آموزش
Android Studio

مولف:مهندس افشین رفوآ

www.tahlildadeh.com

بسم الله الرحمن الرحیم

آموزشگاه تحلیل داده

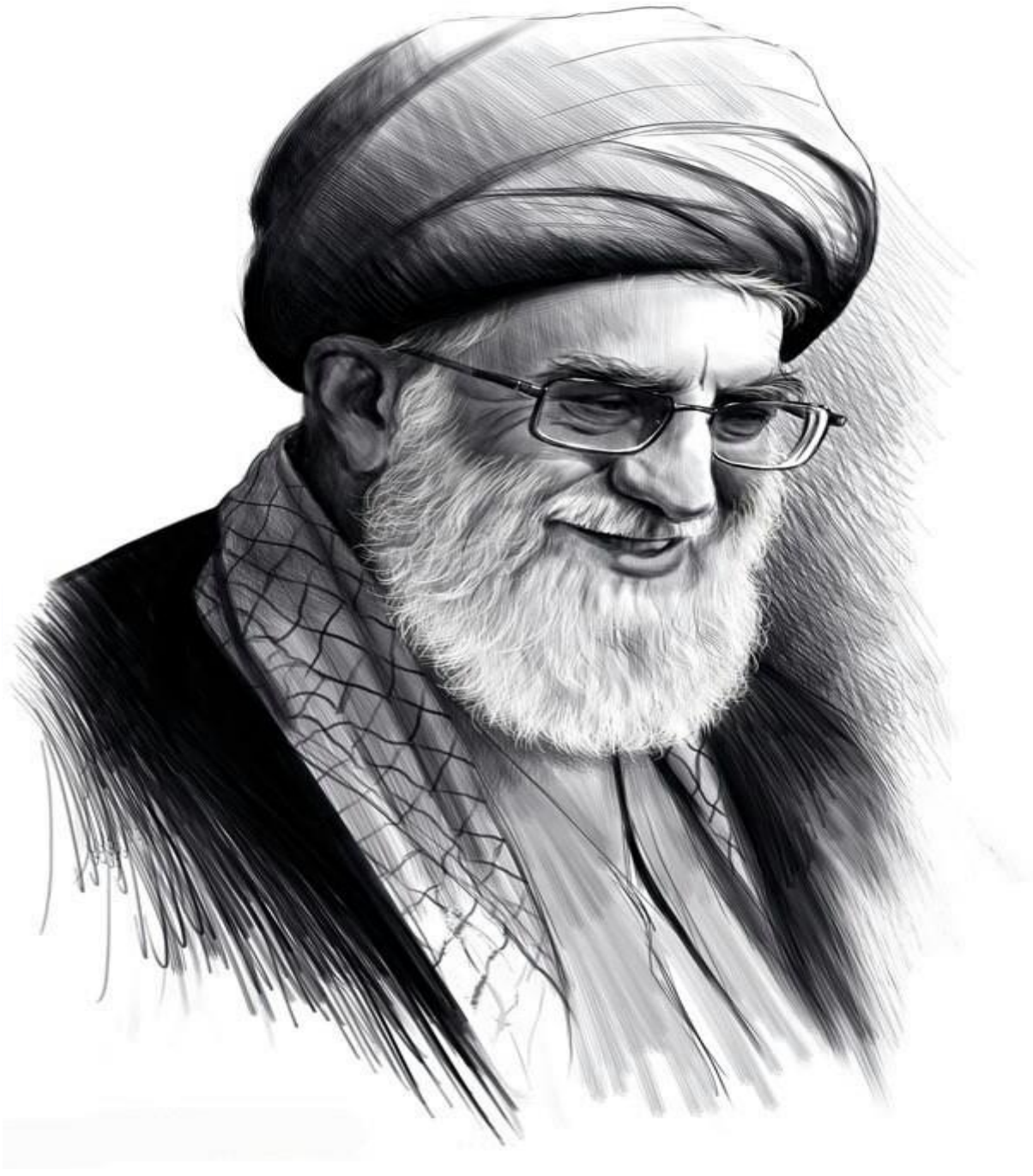
تخصصی ترین مرکز برنامه نویسی و دیتا بیس در ایران

کتاب آموزشی برنامه نویسی اندروید در Android Studio

نویسنده : مهندس افشین رفوآ

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



تقدیم به نائب امام عصر، حضرت آیت الله خامنه ای که عصا زدنش ضرب آهنگ حیدری دارد

ت

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

تقدیم به همه ء جویندگان علم که توان و امکان شرکت در کلاس های ما را ندارند

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فهرست

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فصل اول طراحی و برنامه سازی برای سیستم عامل اندروید با محیط توسعه ی Android Studio

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

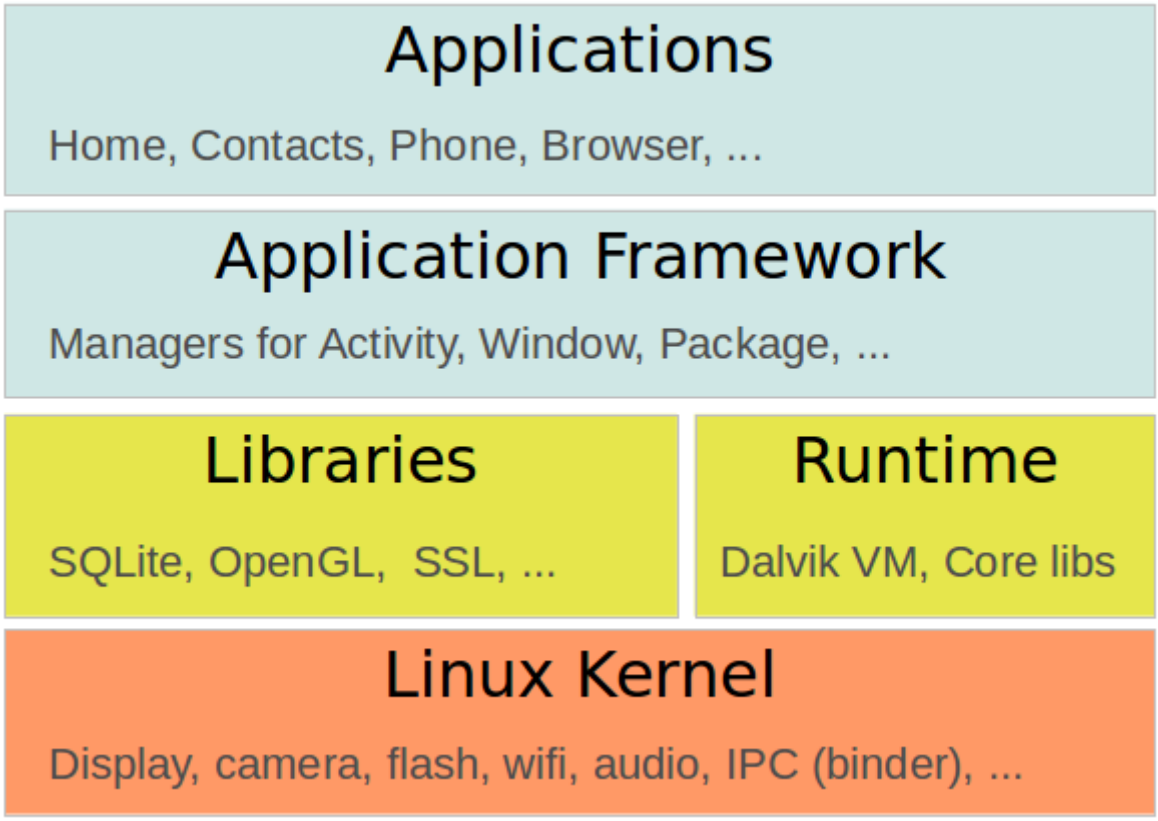
88146323 - 88446780 - 88146330

1-1-مقدمه ای بر برنامه سازی تحت موبایل برای سیستم عامل اندروید

1-1-1-سیستم عامل اندروید

Android یک سیستم عامل تحت موبایل و مبتنی بر هسته ی Linux است. پروژه ی توسعه و ارتقا اندروید AOSP یا پروژه ی کد باز اندروید نام دارد که شرکت Google آن را رهبری می کند.

سیستم عامل اندروید از چهار لایه تشکیل شده است، اما یک توسعه دهنده ی اندروید معمولا با دو لایه ی بالایی (Application و Application Framework) آن سروکار دارد.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

معماری اندروید به شرح زیر می باشد:

- لایه ی Application – پروژه ی کد باز اندروید دربردارنده ی چندین اپلیکیشن پیش فرض همچون مرورگر وب، اپلیکیشن استفاده از دوربین (Camera)، اپلیکیشن پخش موسیقی Music، اپلیکیشن برقراری تماس Phone و غیره ... می باشد. در واقع تمامی برنامه هایی که کاربران از آن ها استفاده می کنند در این لایه نصب شده و در درسترس کاربر قرار می گیرند.
- لایه ی Application framework – یک API است که امکان برقراری تعامل سطح بالا از اپلیکیشن ها با سیستم اندروید را فراهم می آورد. این لایه عمده ی خدمات سطح بالا و مورد نیاز اپلیکیشن ها را در قالب کلاس های جاوا فراهم آورده و به برنامه نویس اجازه ی بهره گیری از این امکانات در توسعه ی اپلیکیشن را می دهد.
- لایه ی Libraries و runtime – کتابخانه هایی که انجام کارهای معمول نظیر نمایش و ارائه ی خروجی گرافیکی (rendering)، ذخیره ی داده ها، قابلیت وبگردی را به صورت آماده فراهم می آورند. این لایه همچنین Runtime اندروید و کتابخانه های اصلی که امکان اجرای اپلیکیشن ها را فراهم می آورد، شامل می شود.
- لایه ی هسته ی Linux – این لایه امکان ارتباط با سخت افزار را مهیا می سازد.

تاکنون ویرایش های متعددی از اندروید منتشر شده که در جدول زیر آن ها را مشاهده می کنید:

Code name	Version	API level
Nougat	N	24
Marshmallow	6.0	23
Lollipop	5.1	22
Lollipop	5.0	21
KitKat	4.4 - 4.4.4	19

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Jelly Bean	4.1.x - 4.3.x	16 - 18
Ice Cream Sandwich	4.0.1 - 4.0.4	14 -15
Honeycomb	3.2.x	13
Honeycomb	3.0 - 3.1	11 - 12
Gingerbread	2.3 - 2.3.7	9-10
Froyo	2.2.x	8
Eclair	2.1	7
Eclair	2.0 - 2.0.1	5 -6
Donut	1.6	4
Cupcake	1.5	3
(no code name)	1.1	2
(no code name)	1.0	1

2-1-1-نحوه ی برنامه سازی برای سیستم عامل تحت موبایل اندروید

زبان برنامه نویسی که برای ساخت و توسعه ی اپلیکیشن اندروید بکار گرفته می شود، زبان شی گرا و همه منظوره ی Java است. به عبارت دیگر، برنامه نویس اندروید تمامی فایل های تنظیمات

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

و منطق اپلیکیشن که برای یک برنامه ی کاربردی اندروید ضروری است را با زبان چند منظوره ی Java می نویسد.

Android development tooling (مجموعه ابزار ساخت و توسعه ی اپلیکیشن های اندروید) فایل هایی که توسط محیط کاری و برنامه نویس تولید شده را به اپلیکیشن اندروید تبدیل می کند. حال اگر برنامه نویس فرایند deployment را آغاز کند، کل اپلیکیشن اندروید به زبان ماشین ترجمه، کلاس های اپلیکیشن پوشه بندی و package شده (تحت یک namespace یا پوشه ی واحد سازمان دهی شده)، سپس برای اجرا تنظیم و آماده می شود (و در نهایت به اجرا در می آید).

(deployment فرایندی است که طی آن اپلیکیشن برای بهره برداری کاملا آماده می شود.)

SDK (مجموعه ابزار ساخت و توسعه ی اپلیکیشن اندروید) و Gradle (سیستم ترجمه و کامپایل کدها به زبان ماشین) ابزار لازم جهت ایجاد، کامپایل و package کردن (سازمان دهی یا دسته بندی کلاس ها تحت یک namespace واحد) اپلیکیشن های اندروید را بر عهده دارد. تیم توسعه دهندگان اندروید افزونه ی Gradle را جهت هدایت اپلیکیشن های اندروید به مرحله ی کامپایل فراهم می آورد که می توانید آن را از اینترنت دانلود کرده و بر روی محیط کاری Android Studio نصب نمایید.

SDK یا مجموعه ابزار ساخت و توسعه ی اپلیکیشن های اندروید، ابزار ADB (پل ارتباطی به دستگاه اندروید و اشکال زدایی اپلیکیشن) را نیز شامل می شود. ADB یک پل ارتباطی به دستگاه های واقعی و مجازی اندروید جهت مدیریت یا اشکال زدایی اپلیکیشن می باشد.

3-1-1-ADT/مجموعه ابزار ساخت و توسعه ی اپلیکیشن اندروید و محیط توسعه ی Android Studio

شرکت Google جهت توسعه و تست اپلیکیشن های اندروید، IDE یا محیط برنامه نویسی Android Studio را ارائه کرده و توصیه می کند. این محیط کاری خود از محیط برنامه نویسی IntelliJ برگرفته شده است.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در ADT ویرایشگرهای اختصاصی ویژه ی فایل های اندروید تعبیه شده. بیشتر فایل های تنظیمات اندروید مبتنی بر XML هستند. ویرایشگرهایی که از آن ها نام برده شد به شما امکان می دهند بین نسخه ی XML (حالت ویرایش) فایل و رابط کاربری ساخت یافته که امکان وارد کردن داده را مهیا می کند، راه گزینی (سویچ) نمایید.

4-1-1-پروسه ی تبدیل و ترجمه از کد برنامه به اپلیکیشن اندروید

توجه: مطالبی که در بخش حاضر مطالعه می کنید مربوط به فرایند کامپایل بوده و برای توسعه ی اپلیکیشن ضروری نمی باشد. در صورت تمایل می توانید از آن رد شوید.

کامپایلر Java فایل های حاوی کد و دستورات (source file) جاوا را به فایل های کلاس (class file) تبدیل می کند. در واقع SDK اندروید در خود ابزاری به نام dx دارد که فایل های Java را به یک فایل اجرایی با پسوند dex. (Dalvik Executable) تبدیل می کند. تمامی فایل های کلاس اپلیکیشن داخل این فایل اجرایی قرار داده می شوند. در طول پروسه ی تبدیل، اطلاعات تکراری و غیر ضروری موجود در فایل های کلاس حذف شده و به طور بهینه سازمان دهی می شوند. به طور مثال، اگر یک متغیر String همزمان در چندین فایل کلاس وجود داشته باشد، فایل dex. تنها یک اشاره گر (reference) به این String را در خود نگه می دارد.

با توجه به توضیح بالا، فایل های dex. بسیار کم حجم تر از فایل های کلاس متناظر هستند.

فایل dex. و محتویات یک پروژه ی اندروید (resource) همچون تصاویر و فایل های XML همگی به صورت پکیج داخل فایل apk. (فایل Android package) قرار داده می شوند. این عملیات را برنامه ای به نام aapt انجام می دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فایل apk. خروجی، تمامی داده های لازم برای اجرای اپلیکیشن را شامل می شود. حال کافی است این فایل را به وسیله ی ابزار adb (پل ارتباطی مدیریت دستگاه اندروید و اشکال زدایی اپلیکیشن) بر روی دستگاه اندروید نصب (deploy) کرد.

از ویرایش 5.0 به بعد اندروید، ART به عنوان سیستم مدیریت اجرای برنامه برای تمامی اپلیکیشن های اندروید مورد استفاده قرار می گیرد. ART از Ahead Of Time compilation (ترجمه ی کل کد برنامه به زبان ماشین به صورت یکجا در ابتدای اجرای اپلیکیشن) بهره می گیرد. حین نصب یک اپلیکیشن بر روی دستگاه اندروید، کد آن اپلیکیشن به زبان ماشین ترجمه می شود. این امر سبب می شود کد پس از کامپایل با افزایش 30 درصدی حجم مواجه شود، اما در زمان راه اندازی برنامه سرعت اجرا را بالا می برد.

از آنجایی که کد اپلیکیشن تنها به هنگام اجرا و راه اندازی اولیه اپلیکیشن به زبان ماشین ترجمه می شود، مصرف باتری به مراتب کاهش می یابد.

ابزار dex2oat فایل dex. تولید شده را به فرمت ELF تبدیل می کند. این فایل کد dex، کد ترجمه شده به زبان ماشین که مستقیما توسط پردازنده اجرا می شود (native code) و meta-data (اطلاعاتی پیرامون اپلیکیشن) را شامل می شود. فایل با نگه داشتن کد dex. امکان استفاده از ابزار جاری را همچنان فراهم می آورد.

عملیات مدیریت حافظه (garbage collection) در ART بهینه سازی شده، از این جهت مدت زمانی که اپلیکیشن ممکن است به طور موقت کند شده یا متوقف شود، کاهش می یابد.

5-1-1-فروشگاه مجازی Google Play

شرکت گوگل سرویسی به نام Google Play را ارائه می دهد که این سرویس یک فروشگاه مجازی است و برنامه نویسان می توانند اپلیکیشن خود را در آن برای استفاده ی کاربران به صورت رایگان و پولی عرضه کنند. کاربران، اپلیکیشن Google Play را نصب کرده و به واسطه ی آن برنامه های کاربردی دلخواه خود را از سرویس Google play دانلود و نصب می کنند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

اپلیکیشن Google Play یک سرویس ویژه کاربران ارائه می دهد که آن ها را از آپدیت و نسخه ی جدید یک نرم افزار آگاه می سازد. برای مثال زمانی که توسعه دهنده یک نسخه ی جدید از اپلیکیشن خود را در فروشگاه بارگذاری می کند، این سرویس بلافاصله کاربران جاری خود را از آن آپدیت مطلع ساخته و این آپدیت را در اختیار آن ها قرار می دهد.

Google play همچنین کتابخانه ها و سرویس هایی را ویژه ی برنامه سازان اندروید ارائه می دهد. به عنوان مثال می توان به سرویسی اشاره کرد که امکان استفاده و نمایش Google Maps را مهیا می سازد. ارائه ی این سرویس ها از طریق Google Play این مزیت را دارد که ویرایش های قدیمی اندروید نیز می توانند از آن ها استفاده کنند. Google می تواند آن ها را بدون اینکه نیازی به بروز آوری کلی سیستم عامل اندروید باشد، بروز رسانی کند.

1-2-نصب محیط برنامه نویسی Android Studio

1-2-1-سیستم مورد نیاز برای نصب Android Studio

می توان بر روی یک سیستم سخت افزاری متوسط به راحتی برای سیستم عامل اندروید برنامه نویسی کرد. هر چند برای شرایط مطلوب، بهتر است محیط سخت افزاری مدرن با پردازنده ی 2.6 گیگاهرتز و رم 8 گیگابایت استفاده نمایید. هارد SSD نیز می تواند به شدت در اجرا و راه اندازی سریع برنامه شبیه ساز اندروید موثر باشد.

2-2-1-ابزار لازم برای نصب و استفاده از سیستم عامل Linux

SDK اندروید 32 بیتی است، به همین جهت بر روی یک سیستم لینوکس 64 بیتی، می بایست پکیج ia32-libs را نصب نمایید. در نسخه ی Ubuntu می توانید این کار را با اجرای دستور زیر انجام دهید:

```
apt-get install ia32-libs
```

چنانچه از نسخه ی دیگر Linux استفاده می کنید، توصیه می شود توضیحات و مستندات ارائه شده همراه با آن نسخه را در خصوص دستور مربوط مطالعه نمایید.

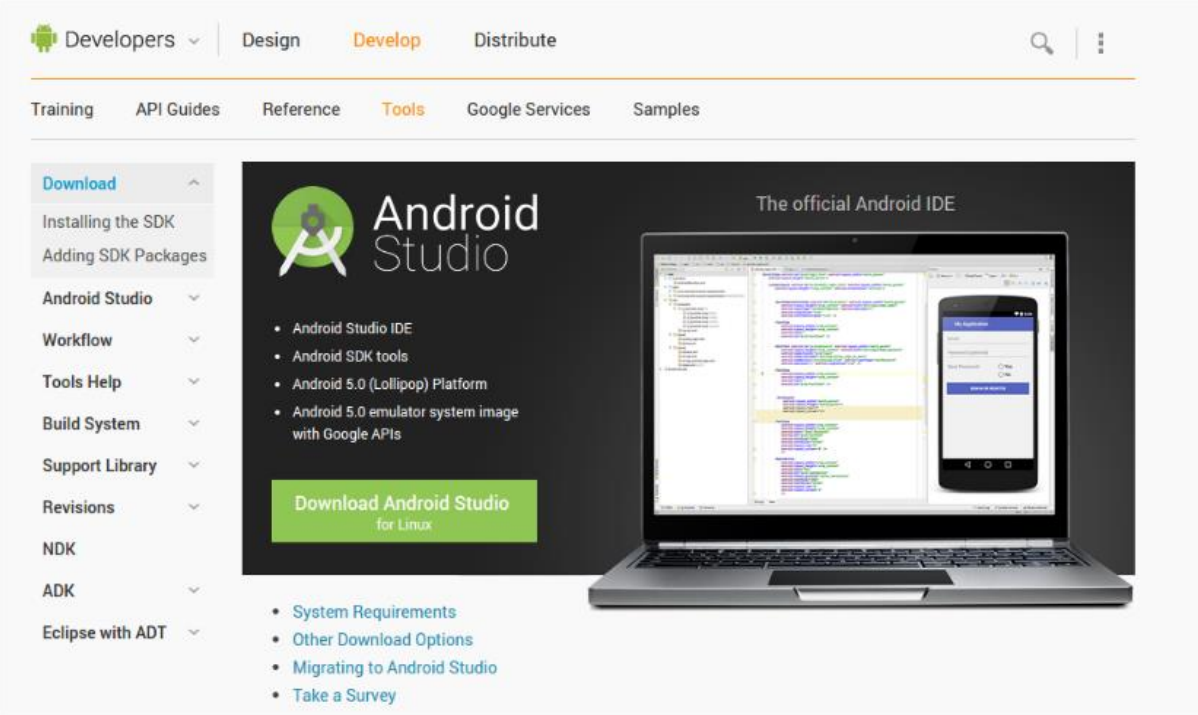
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

3-2-1-دانلود Android Studio از اینترنت

می توانید Android Studio را از آدرس http://developer.android.com/sdk/index.html دانلود نمایید.

دانلود در دو نسخه عرضه می شود: 1. یک نسخه فقط SDK Tools را شامل می شود 2. دیگری کل پکیج های Android Studio Packages را دربر می گیرد. شما می بایست Android Studio Package را دانلود نمایید.



4-2-1-نصب محیط برنامه نویسی Android Studio

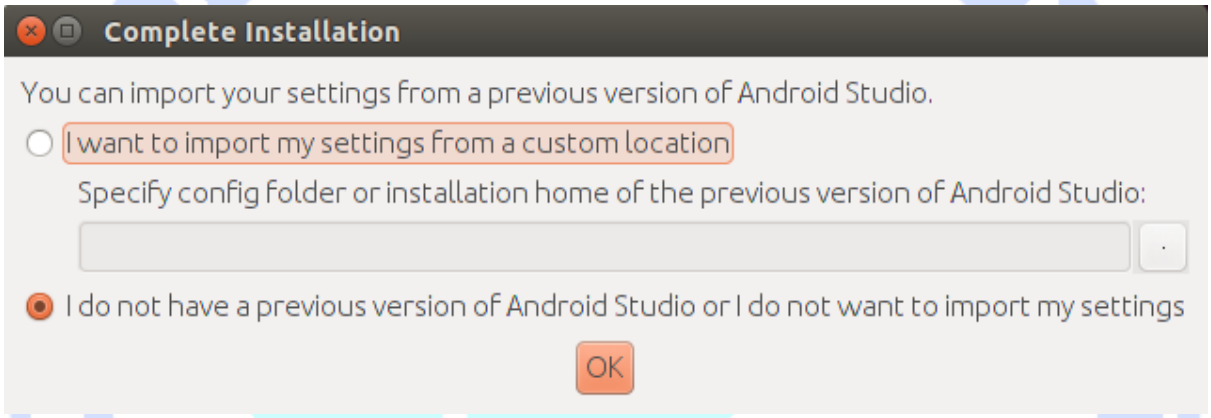
پروسه نصب بر روی سیستم عامل Windows آسان است، کافی است فایل اجرایی exe. را که از اینترنت دانلود کردید، اجرا نمایید. در سیستم عامل Max OSX کافی است Android Studio را با اشاره گر موس کشیده و در پوشه ی Applications جایگذاری نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در سیستم عامل Linux می بایست فایل ZIP دانلود شده را از حالت فشرده خارج ساخته و در مکان مناسب جایگذاری نمایید. به منظور راه اندازی پروسه ی نصب Android Studio، ابتدا به آدرس android-studio/bin/ داخل یک پنجره ی فرمان (terminal) پیمایش نموده و سپس دستور studio.sh را اجرا نمایید.

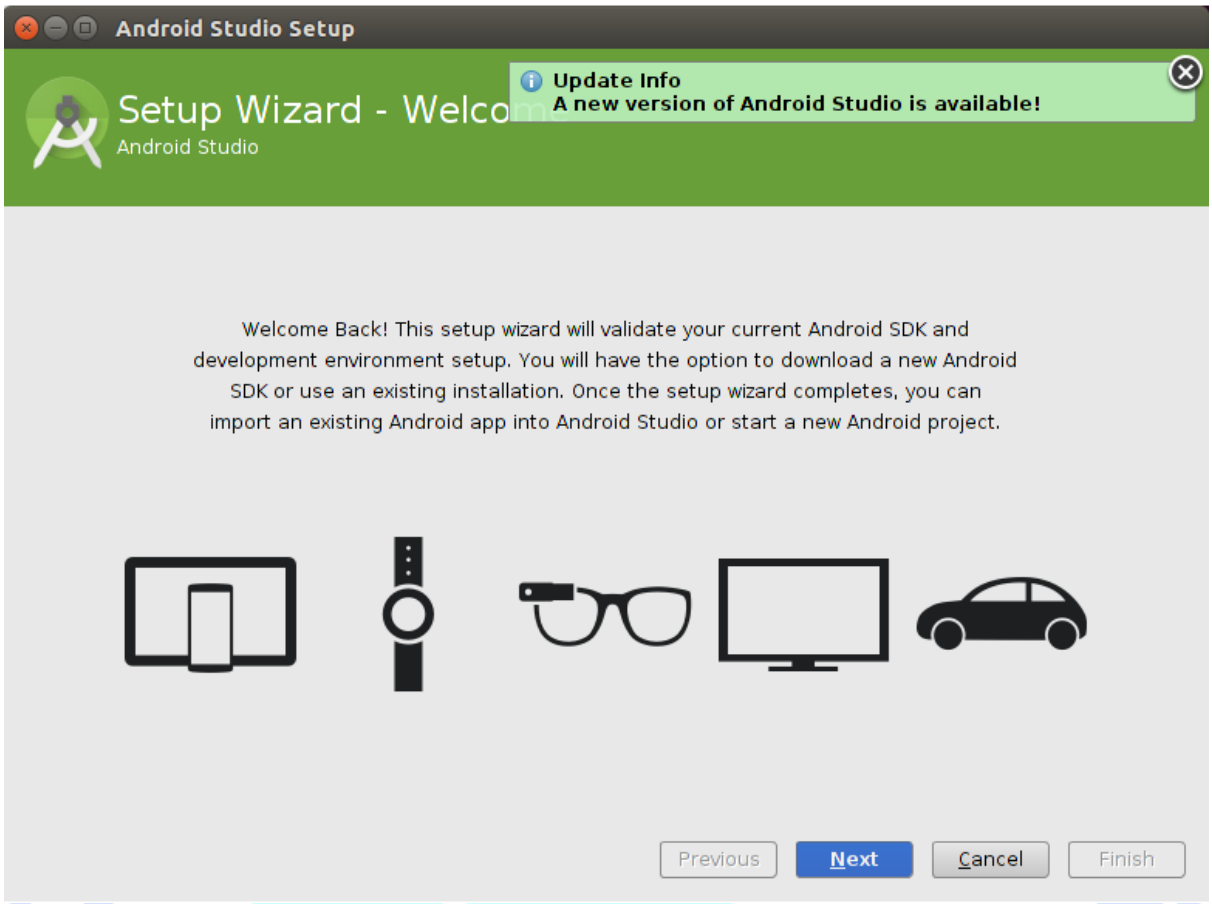
زمانی که پروسه ی نصب Android Studio را راه اندازی می کنید، این امکان در اختیار شما قرار می گیرد که تنظیمات مورد نیاز نسخه ی نصبی موجود را (برای مثال setting ویرایش قدیمی تر Android Studio که قبلا بر روی سیستم خود نصب کردید) به محیط جدید وارد نمایید.



باقی مراحل نصب را به صورت ویزاردی دنبال نمایید.

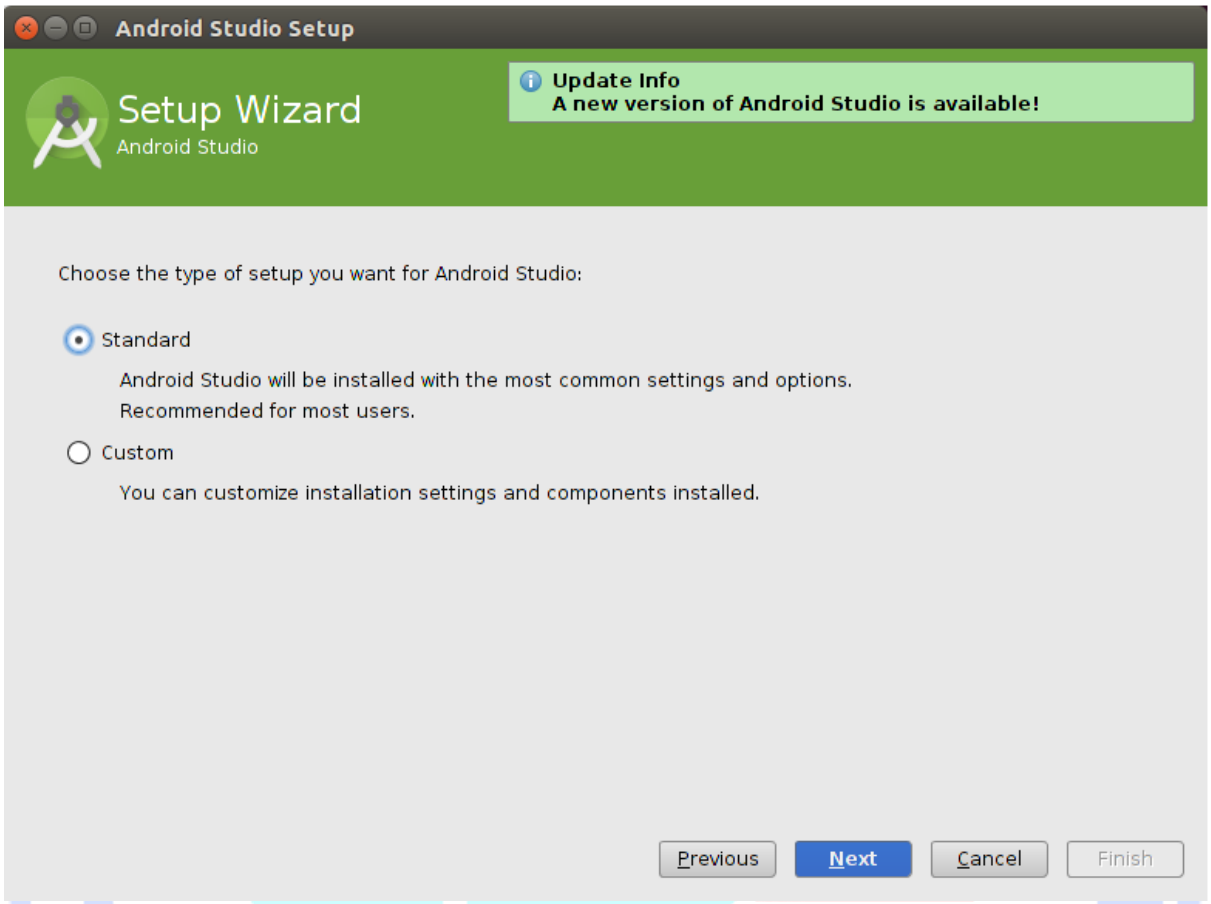
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



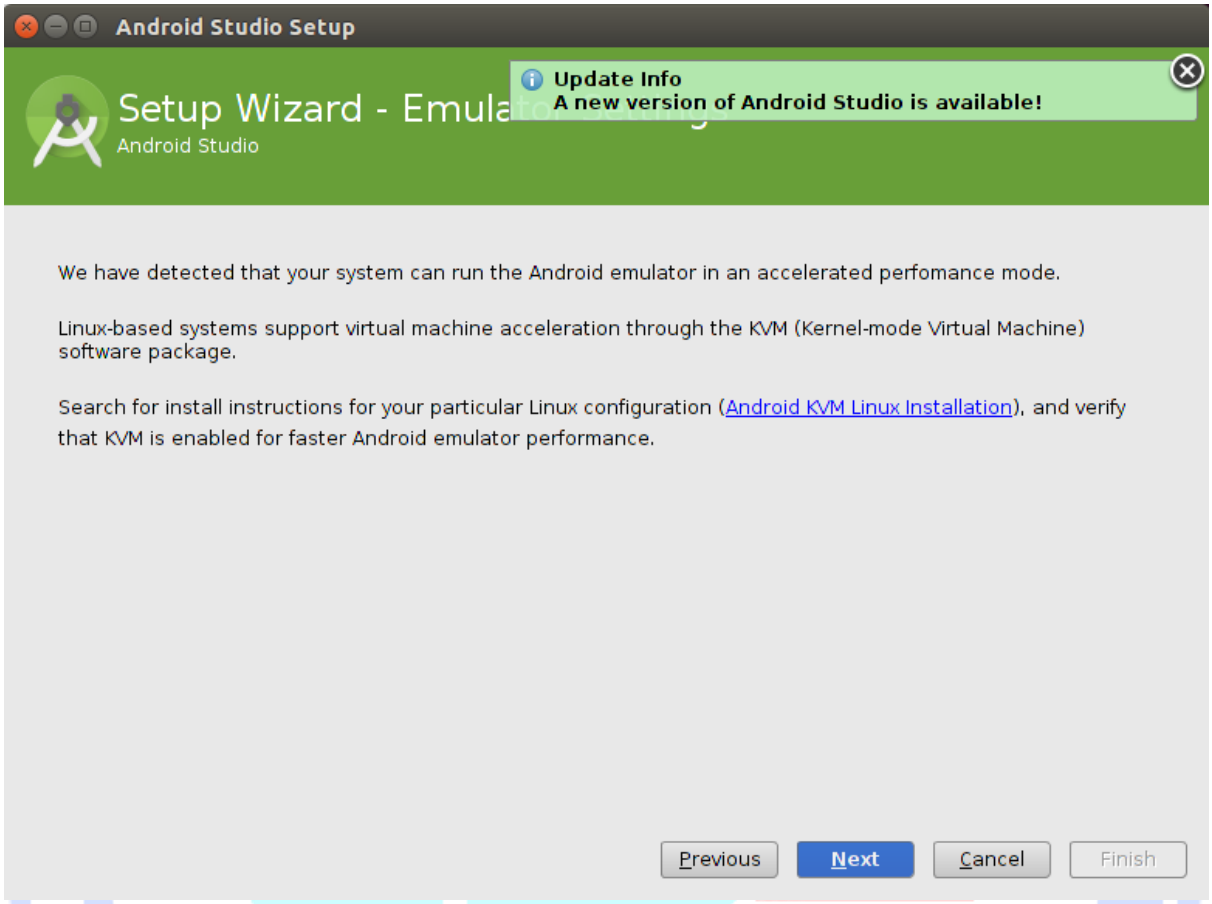
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

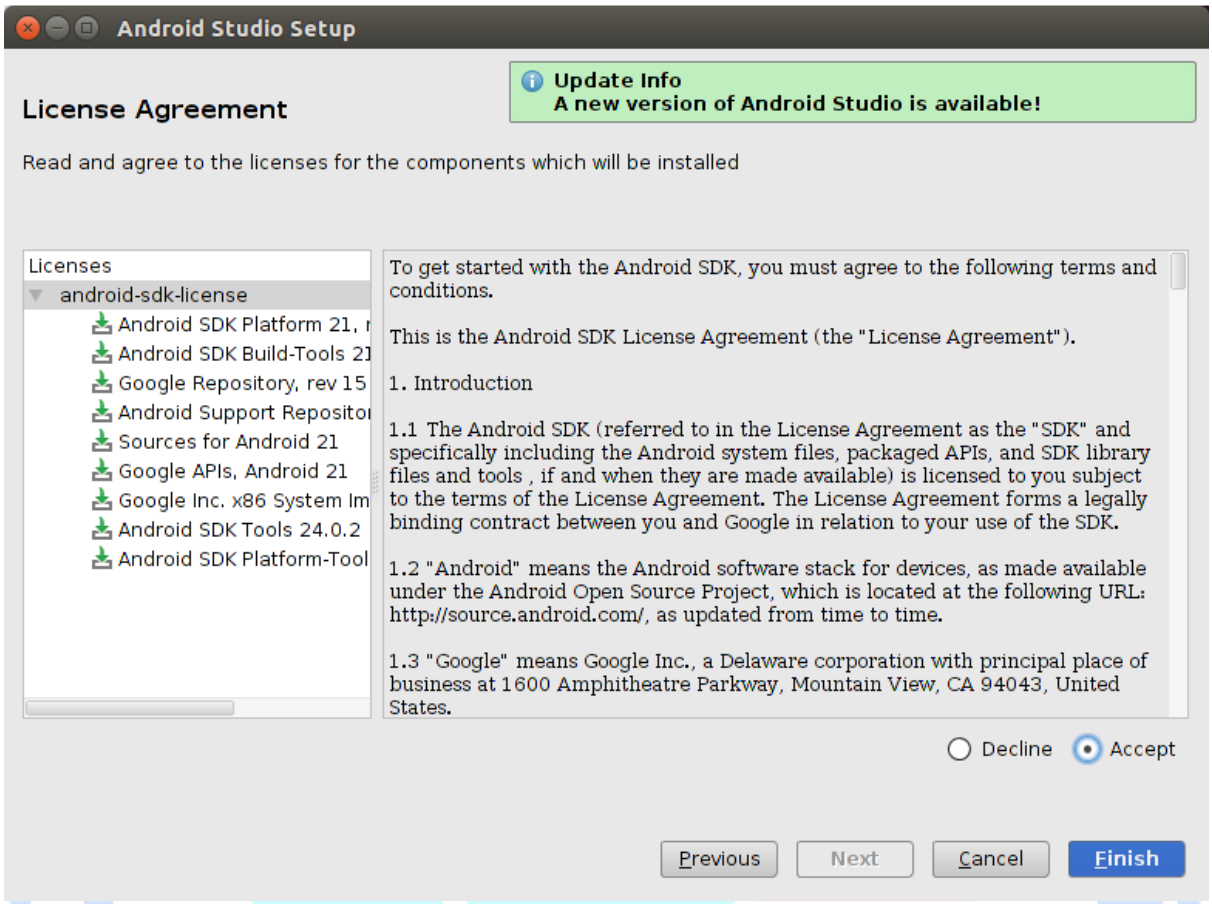
88146323 - 88446780 - 88146330



در آخرین صفحه ی نصب، بر روی دکمه ی Finish کلیک نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



تمرین: شروع به کار با Android Studio

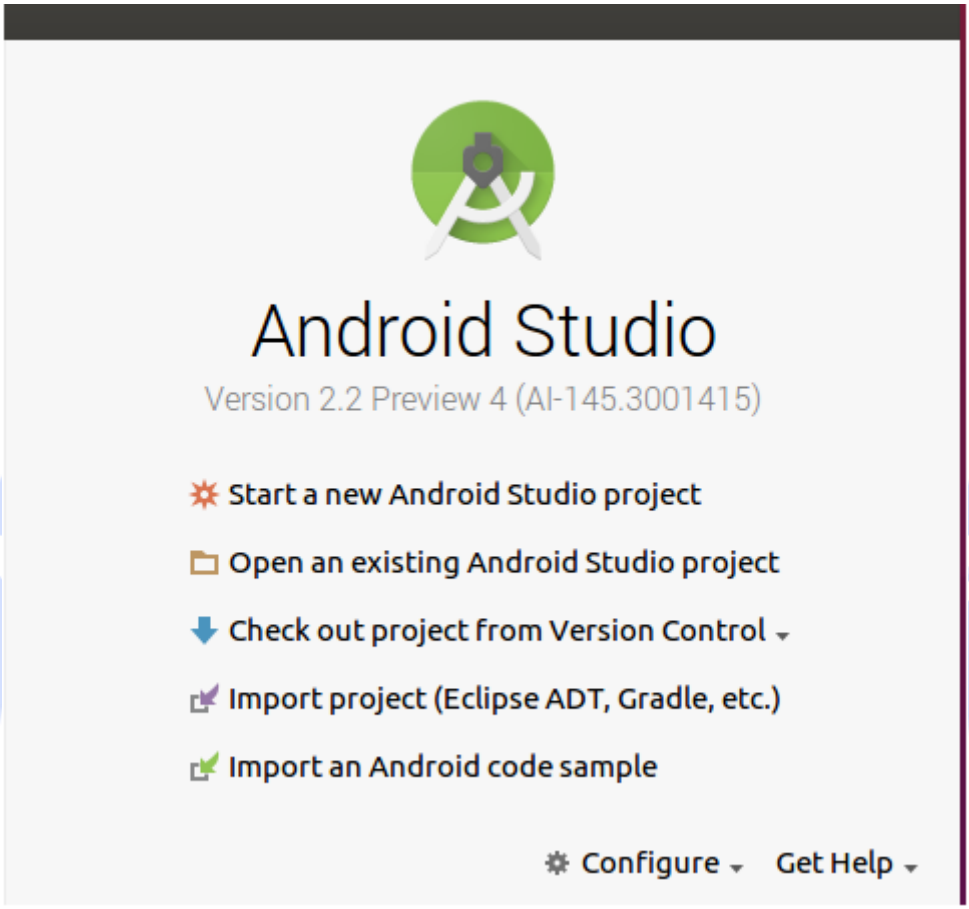
در این تمرین می توانید یک پروژه ی اندروید تعریف کرده، سپس دستگاه مجازی Android (VM) را بر روی آن راه اندازی نمایید.

5-2-1-ایجاد یک پروژه ی اندروید

جهت ساخت یک اپلیکیشن ساده ی اندروید، می توانید بر روی لینک Start a new Android Studio project کلیک نمایید و یا اگر قبلا یک پروژه ی اندروید ایجاد کردید، این مسیر را طی نمایید: File ▸New Project.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



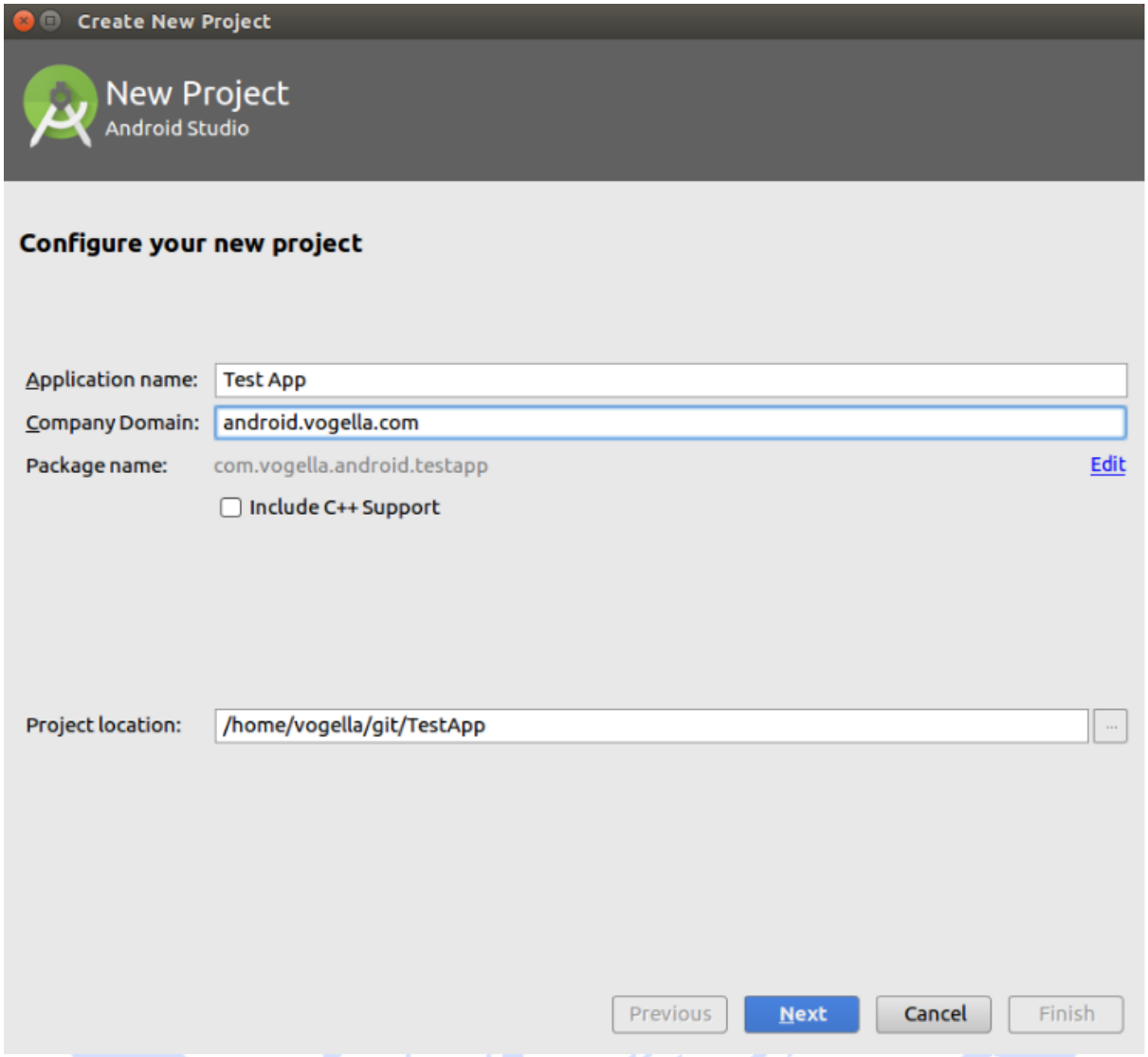
پروژه ی اندروید خود را با مقادیر زیر تنظیم نمایید (فیلدهای پنجره ی Create new project را با مقادیر زیر مقداردهی نمایید). محل ذخیره ی پروژه (مقدار فیلد project location) و اسم پکیج (فیلد package name) از مقادیری که شما وارد می کنید، بر گرفته می شود. برای ویرایش اسم پکیج، کافی است بر روی لینک کوچک Edit کلیک نمایید.

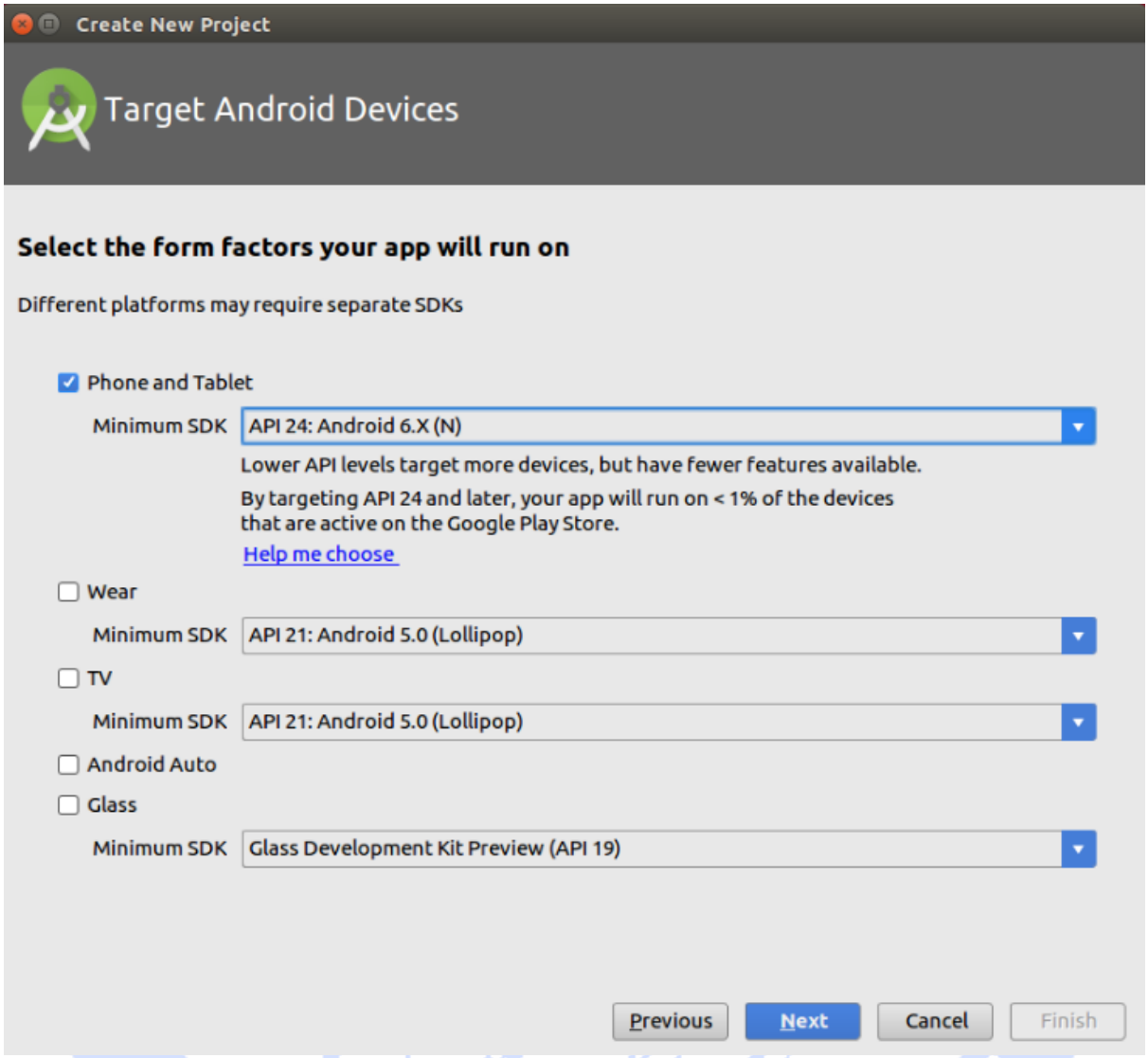
Property	Value
Application name	Test App

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Property	Value
Company Domain	android.vogella.com
Package Name	com.vogella.android.testapp
API (Minimum, Target, Compile with)	Latest
Template	Empty Activity

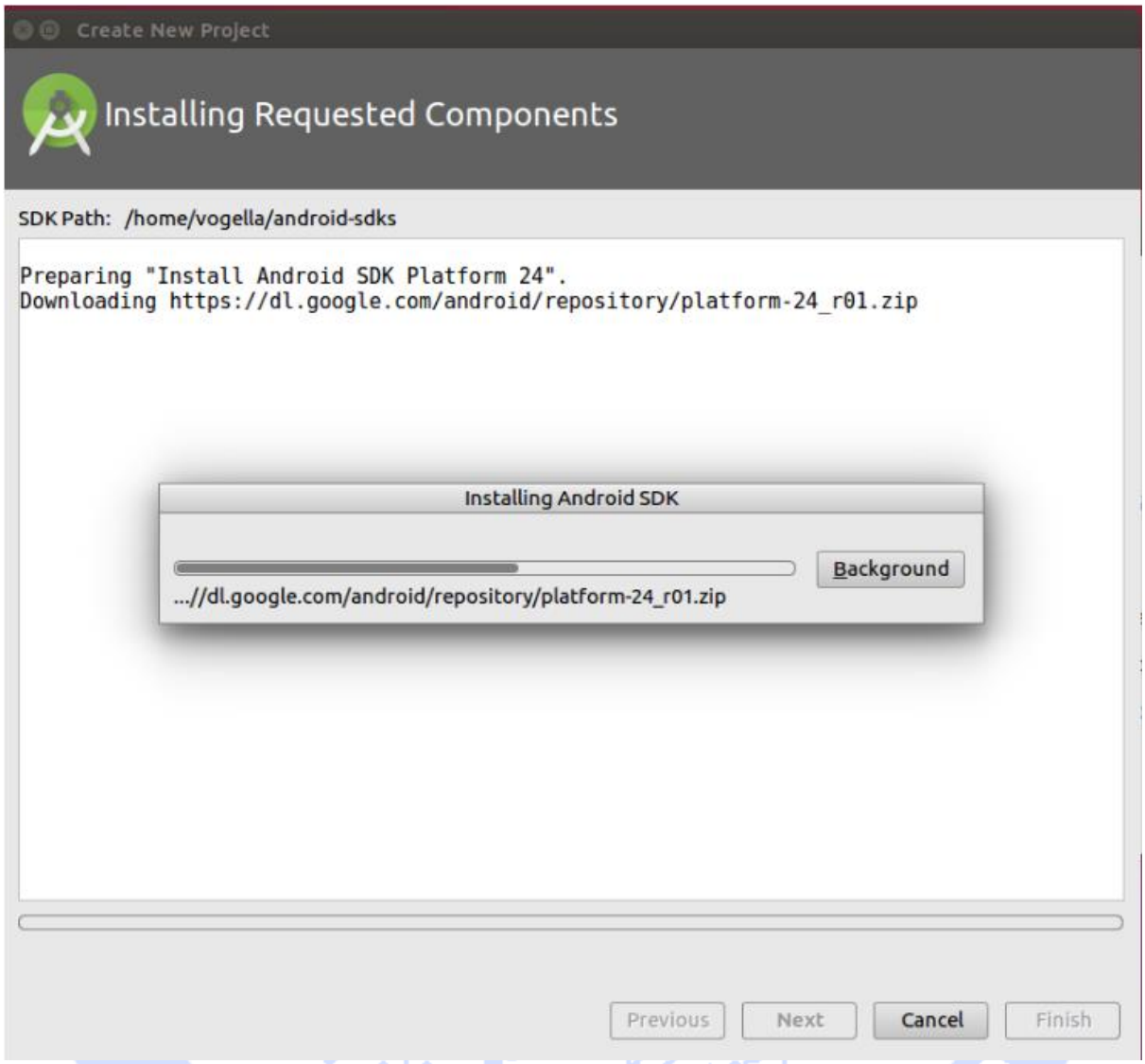




اگر قبلا SDK را دانلود نکرده باشید، Android Studio به صورت خودکار SDK مورد نیاز را از اینترنت بارگیری می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

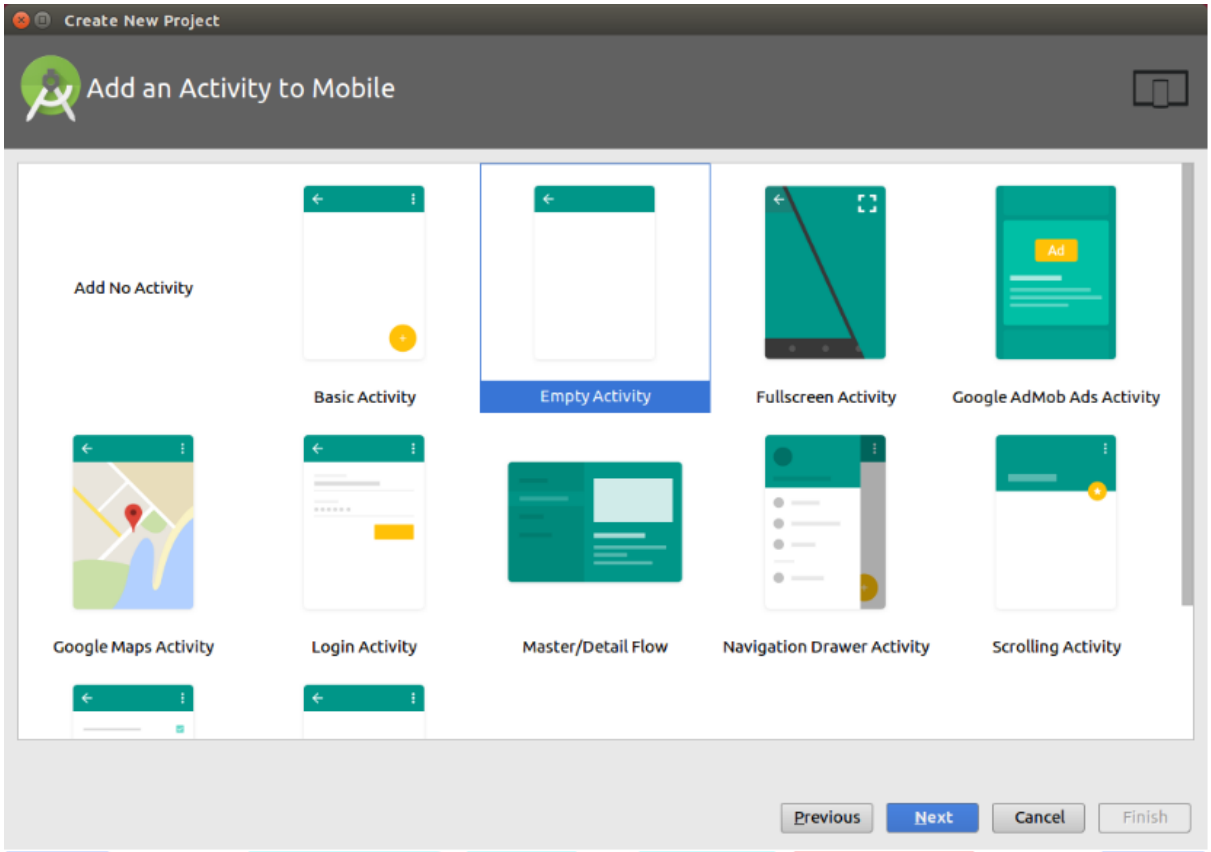
88146323 - 88446780 - 88146330



در پنجره ی Add an Activity to Mobile، قالب آماده ی Empty Activity را انتخاب نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

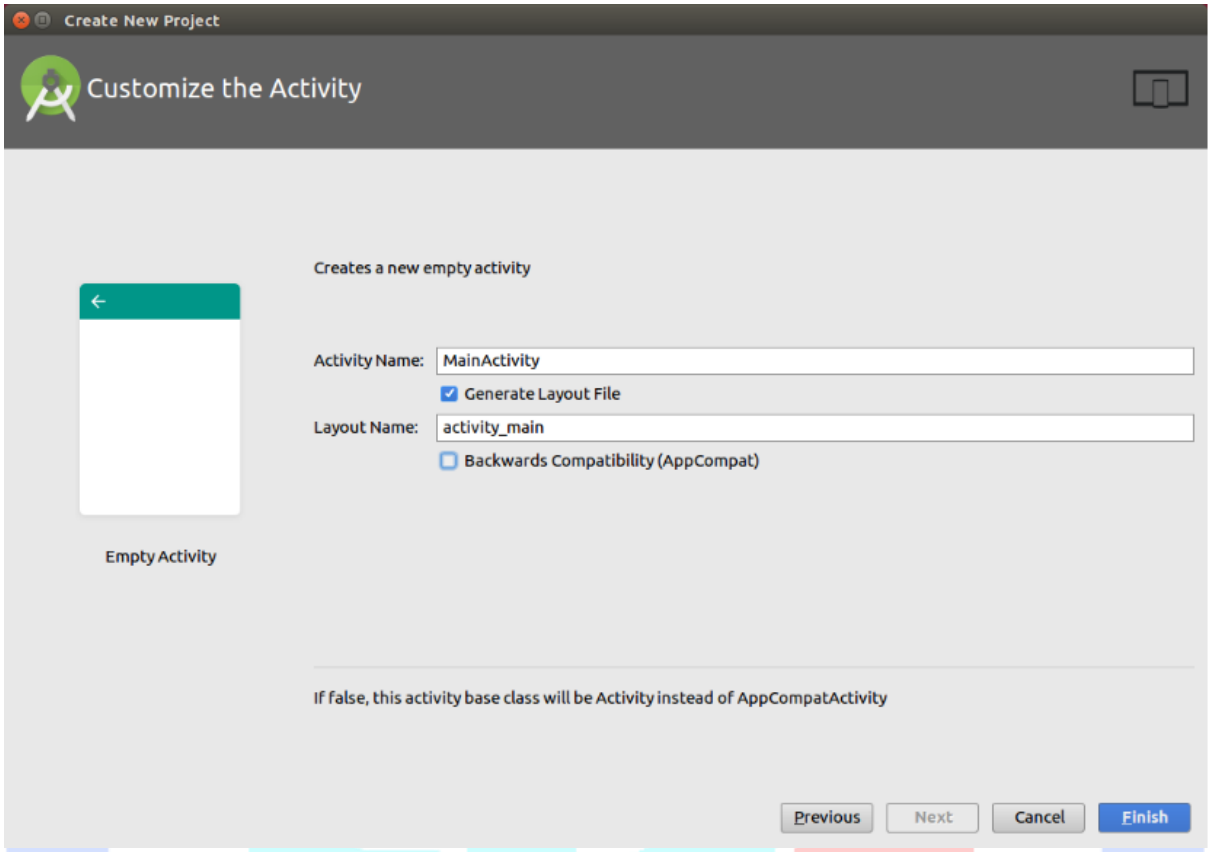
88146323 - 88446780 - 88146330



در آخرین صفحه ی تنظیمات پروژه ی جدید، گزینه ی Backwards Compatibility (قابلیت پشتیبانی از ویرایش های قبلی) را از حالت انتخاب خارج نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

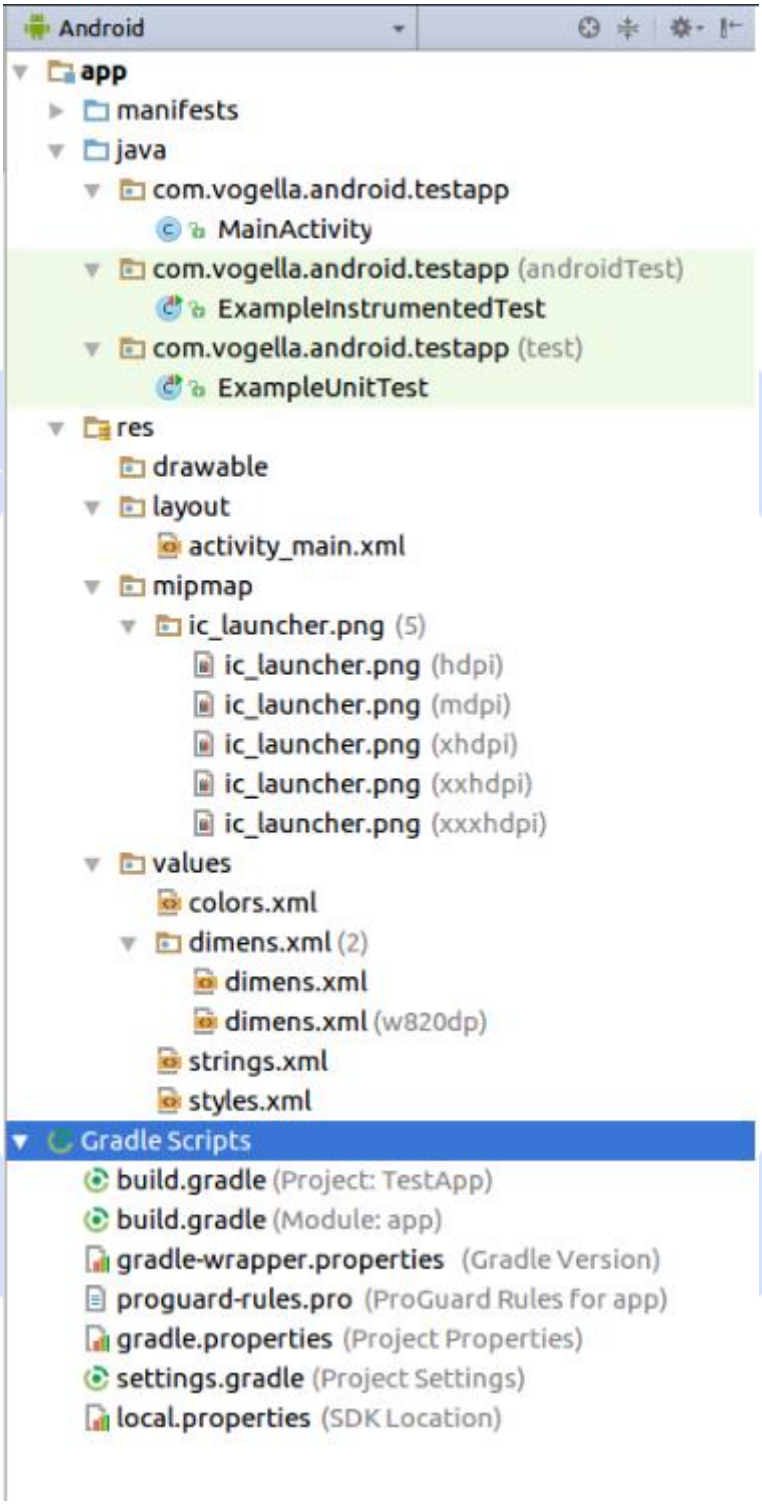


6-2-1-بررسی پروژه ی ایجاد شده

برنامه ی راهنما (wizard) اکنون یک پروژه ی Android برای شما ایجاد می کند. بد نیست ساختار و فایل های تولید شده ی پروژه را بررسی نمایید.

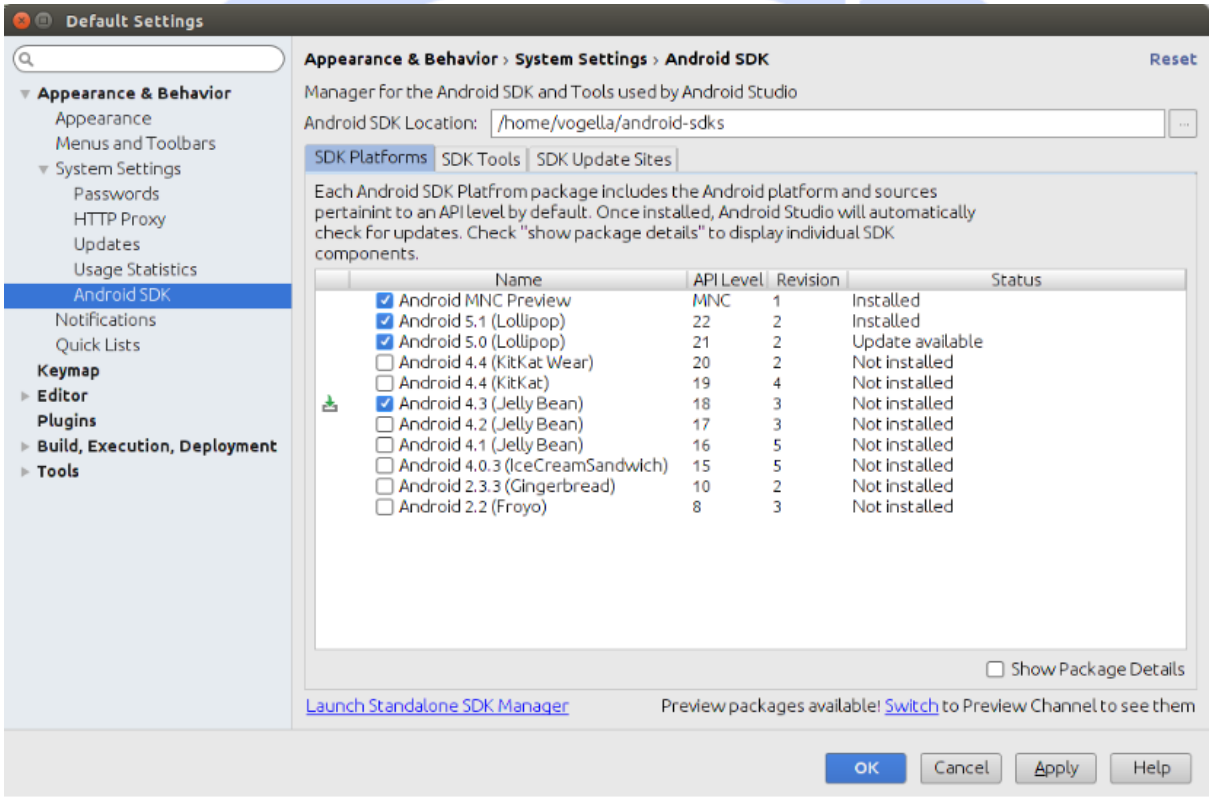
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



7-2-1-نصب نسخه ی مورد انتخاب سیستم عامل اندروید

در SDK Manager، نسخه ی دلخواه اندروید برای نصب را از نمای درختی انتخاب نموده و سپس دکمه ی Install را کیلک نمایید. همان طور که در تصویر زیر مشاهده می کنید، API 18 از کتابخانه های اندروید انتخاب شده و نصب می شود.



با کلیک بر روی دکمه ی OK پروسه ی نصب را آغاز نمایید.

در تب SDK Platforms، می توانید ورژن API یا کتابخانه های اندروید را برای نصب انتخاب نمایید.

در تب SDK Tools نیز می توانید ابزار توسعه و ساخت اپلیکیشن را نصب نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

8-2-1-نصب support library (کتابخانه ی پشتیبانی از API های جدید اپلیکیشن در نسخه های قدیمی تر آن)

Support Library این امکان را برای شما فراهم می آورد تا از قابلیت ها و امکانات ویرایش های بالاتر اندروید در نسخه های پایین تر آن استفاده نمایید. علاوه بر آن، این کتابخانه قابلیت هایی که به صورت جدا از Android عرضه می شود (unbundled)، همچون ابزارک (widget) RecycleView جهت نمایش بهینه ی لیست، را در اختیار شما قرار می دهد.

در حال حاضر ویرایش های متعددی از این کتابخانه وجود دارد: v4، v7 و v13. این کتابخانه ها هر یک برای ورژن های مربوطه قابل استفاده می باشند. به عنوان مثال، support library v7 برای دستگاه های اندروید که از کتابخانه های ورژن 7 (API 7) اندروید استفاده می کنند، قابل استفاده می باشد. لازم به ذکر است که نسخه های بالاتر support library برای عملکرد صحیح خود نیاز به ورژن های پایین تر دارند. به طور مثال، support library v7 برای عملکرد صحیح خود به کتابخانه ی v4 احتیاج دارد.

1-3-تست اپلیکیشن های اندروید بر روی محیط مجازی (ADV) یا دستگاه واقعی

1-3-1-محیط شبیه ساز اندروید (emulator) و دستگاه مجازی اندروید (ADV)

مجموعه ابزار اندروید (Android Tooling) در خود یک نرم افزار شبیه ساز (Android device emulator) دارد. این شبیه ساز AVD را اجرا می کند. AVD می تواند محیط و سخت افزار یک گوشی واقعی اندروید را شبیه سازی کند.

AVD ها این امکان را برای شما فراهم می کنند تا اپلیکیشن های اندروید خود را بر روی ورژن های مختلف اندروید با تنظیمات سخت افزاری مختلف تست کنید. حتی اگر هم یک دستگاه واقعی اندروید برای تست اپلیکیشن های خود دارید، لازم است نحوه ی کار با AVD ها را بیاموزید چرا که

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

دستگاه های مجازی به شما اجازه می دهند برنامه ی کاربردی خود را بر روی ورژن های مختلف اندروید با تنظیمات سخت افزاری خاص اجرا و تست نمایید.

تنظیمات دلخواه دستگاه مجازی را می بایست در طول پروسه ی ایجاد AVD مشخص نمایید. این تنظیمات انتخاب کیفیت تصویر (resolution)، ورژن کتابخانه های اندروید (API version) و تراکم پیکسلی مد نظر را شامل می شود.

می توانید چندین AVD با تنظیمات مختلف تعریف نمایید و آن ها را همزمان اجرا کنید. این کار به شما اجازه می دهد به طور همزمان اپلیکیشن خود را بر روی چندین دستگاه با تنظیمات مختلف امتحان کنید.

نکته: اگر AVD را حین راه اندازی اولیه متوقف نمایید، دستگاه مجازی خراب می شود. راه اندازی اولیه در دستگاه های قدیمی کمی زمان می برد. در دستگاه های جدید فرایند راه اندازی اولیه معمولا 1 تا 3 دقیقه طول می کشد.

پس از اجرای کامل AVD، می توانید GUI را به وسیله ی موس کنترل نمایید. شبیه ساز همچنین با ارائه ی منویی در سمت راست محیط، به شما امکان می دهد تا به دکمه های گوشی دسترسی داشته باشید. سعی کنید در طول پروسه ی توسعه ی اپلیکیشن، AVD را متوقف نکنید. چنانچه در اپلیکیشن خود تغییراتی را اعمال نموده و هم اکنون می خواهید نسخه ی جدیدی از آن را تست نمایید، بهتر است اپلیکیشن خود مجددا بر روی AVD مستقر (deploy) کنید.

2-3-1- Debug certificate (شناسنامه ی تاریخ تولید و ثبت اپلیکیشن) و تاریخ ابطال آن (expiry date)

اپلیکیشن های اندروید می بایست قبل از نصب بر روی دستگاه، امضا شده و کلید منحصر بفردی به آن تخصیص داده شود. در واقع در طول پروسه ی توسعه ی برنامه، محیط کاری Eclipse اپلیکیشن شما را به صورت خودکار با یک certificate یا امضای دیجیتالی خاص به نام debug key نشانه گذاری می کند تا امکان کپی برداری از آن وجود نداشته باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

حال این امضای دیجیتالی که به برنامه ی کاربردی تخصیص داده می شود، حدودا تا 365 روز اعتبار دارد. زمانی که این تاریخ به پایان می رسد، اپلیکیشن یک خطای زمان کامپایل (build error) صادر کرده و به شما اعلان می کند که certificate یا امضای دیجیتالی تخصیص داده شده به اپلیکیشن منقضی شده و دیگر معتبر نیست.

به منظور رفع این مشکل، کافی است فایل debug.keystore را حذف نمایید. محل ذخیره ی فایلی های اپلیکیشن به صورت پیش فرض در سیستم عامل OS X و Linux در آدرس ~/.android/ و در سیستم عامل ویندوز XP در آدرس C:\Documents andSettings\[username]\.android\ و در Vista و 7 در آدرس C:\Users\[username]]\.android\ می باشد.

دفعه ی بعد که اپلیکیشن را کامپایل (build) می کنید، ابزار کامپایل پروژه یک keystore به همراه debug key جدید برای اپلیکیشن تولید کرده و به آن اختصاص می دهد (key store = جهت تایید اعتبار id توسعه دهندگان مورد استفاده قرار می گیرد).

3-3-1-AVD Google در مقایسه با Android AVD

در طول ایجاد یک AVD، این اختیار را دارید که بین Android device یا Google device یکی را انتخاب نمایید.

یک Android AVD اغلب برنامه هایی از پروژه ی کد باز اندروید (Android Open Source Project) را شامل می شود. این در حالی است که یک AVD ساخته شده با API های Google معمولا کدهای اضافی (کتابخانه ها یا توابع اضافی) از Google را دربرمی گیرد.

AVD هایی که برای API های گوگل ایجاد می کنید به شما امکان می دهند تا اپلیکیشن هایی را که از سرویس های Google Play (به عنوان مثال سرویس های موقعیت یابی مانند Google Maps API) استفاده می کنند، تست نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

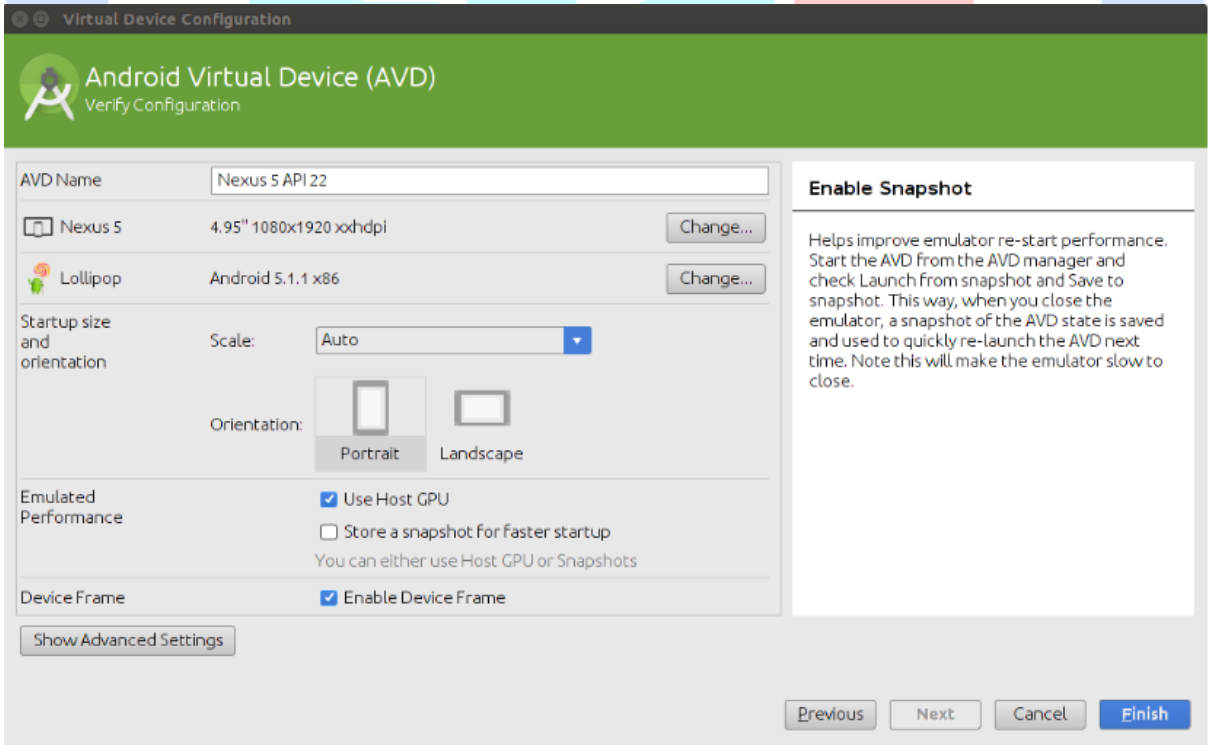
88146323 - 88446780 - 88146330

4-3-1-بهینه سازی و افزایش سرعت اجرا با انتخاب GPU رایانه ی میزبان (تنظیمات آیتم ی Emulated Performance)

در طول فرایند تنظیم و آماده سازی یک محیط شبیه ساز برای تست برنامه (emulator)، می توانید یکی از دو گزینه ی Snapshot یا Use Host GPU را انتخاب نمایید.

توجه: پنجره ی محاوره ای حاضر به شما اجازه می دهد هر دو گزینه را انتخاب نمایید. اما اگر هر دو گزینه را فعال نمایید، با خطا مواجه خواهید شد.

با انتخاب گزینه ی اول، هنگامی که شما برای دومین بار دستگاه را راه اندازی می کنید، می بینید که دستگاه مجازی با سرعت خارق العاده ای اجرا می شود. می دانید چرا؟ به این علت که به هنگام بستن دستگاه مجازی، AVD وضعیت و اطلاعات جاری خود را ذخیره می کند و طبیعتا با اجرای بعدی این وضعیت بلافاصله بازگردانده می شوند. چنانچه گزینه ی Use Host GPU را انتخاب نمایید، AVD مستقیما از کارت گرافیک سیستم میزبان برای پردازش و نمایش گرافیک بهره می گیرد که سرعت برنامه را در شبیه ساز به طور قابل توجهی بالا می برد.



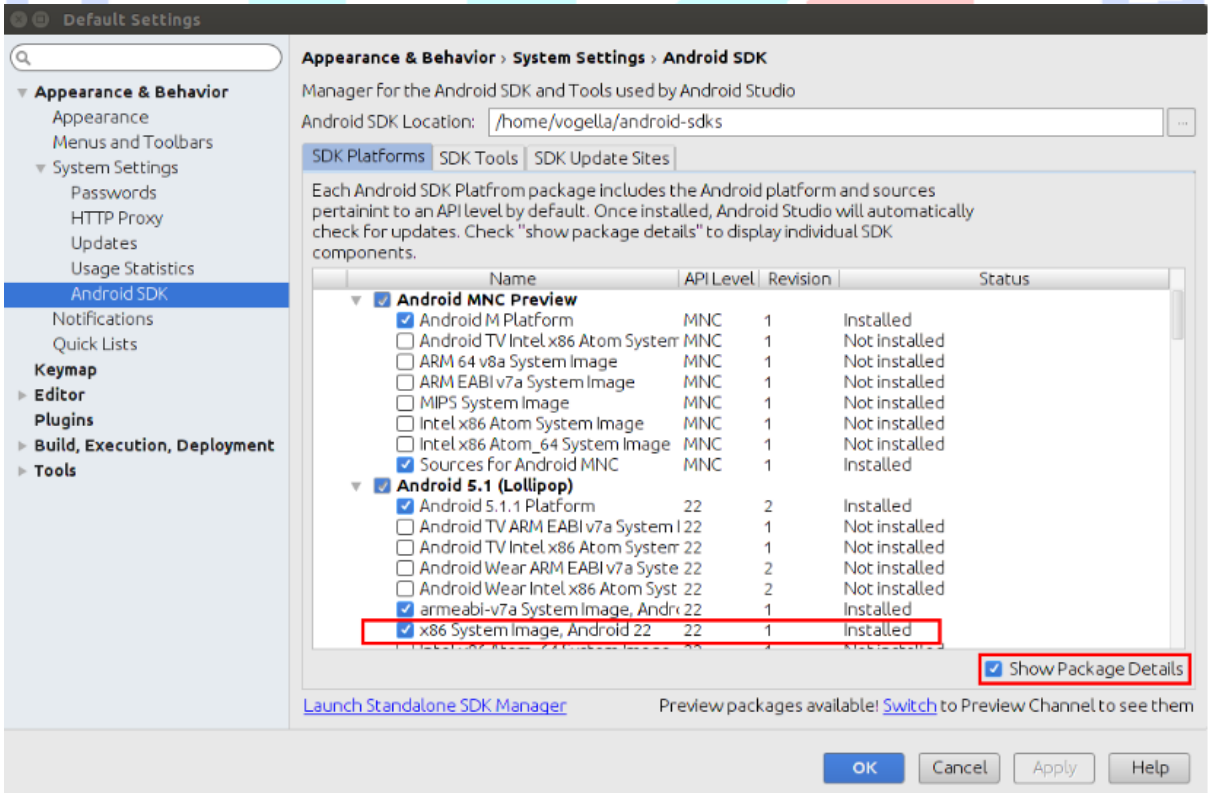
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

5-3-1-افزایش سرعت با انتخاب Intel image system

می توانید یک AVD را با image system مبتنی بر معماری ARM CPU یا Intel CPI اجرا نمایید. دستگاه مجازی که از Intel image system بهره می گیرد، در اجرا بر روی سخت افزار AMD/Intel نسبت به system image مبتنی بر ARM بسیار سریع تر عمل می کند. در شرح علت آن باید گفت که شبیه ساز نیازی ندارد که دستورات پردازنده ی ARM را بر روی کامپیوتر شما به Intel/AMD ترجمه کند. حقیقت این است که image تهیه شده از ARM بیشتر با دستگاه های اندروید مطابقت دارد، اما لازمه ی این است که پردازنده شبیه سازی شود و این افت سرعت را در پی دارد. x86 بسیار سریع تر است چرا که سعی می کند با استفاده از مکانیزم های مربوطه ی AVD کد را به صورت native اجرا کند.

Intel image را می توان به راحتی توسط Android SDK Manager برای API مربوطه نصب کرد. در محیط برنامه نویسی Android Studio، به هنگام ایجاد دستگاه مجازی، این اتفاق به صورت خودکار رخ می دهد. می توان تنظیمات لازم را از طریق package details انجام داد.

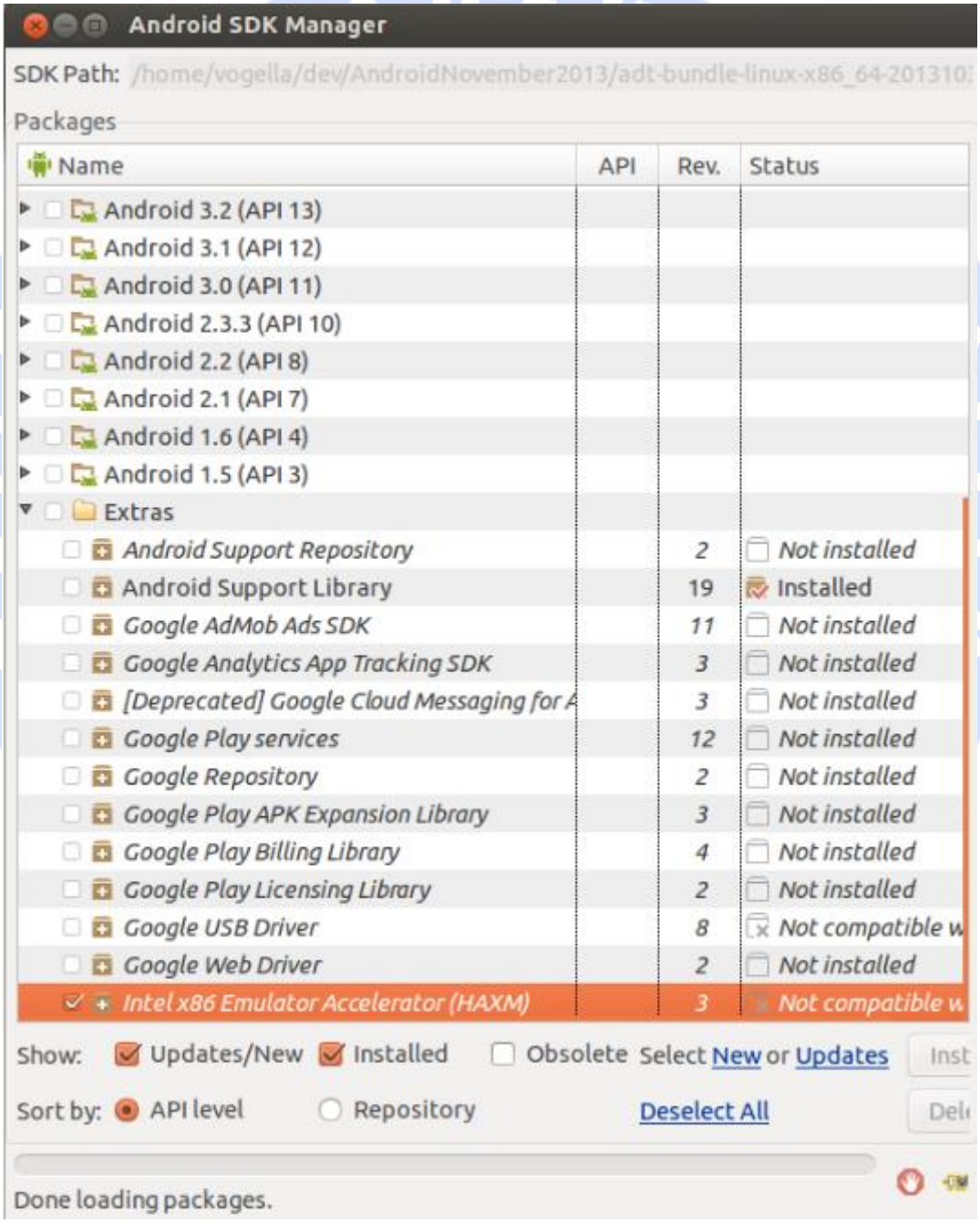


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

نکته: لازم به ذکر است که intel image برای تمامی ورژن های مجموعه کتابخانه های اندروید (API Level) در دسترس نیست.

لازم است برای ویندوز درایورهای بیشتری دانلود و نصب نمایید.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

پس از دانلود، می توانید به فایل مورد نظر در محل قرار گیری فایل های نصبی اندروید در پوشه ی extras/intel دسترسی داشته باشید. برای نصب درایورها می بایست است فایل اجرایی exe. را کلیک نمایید. این مرحله ی اضافی برای سیستم عامل ویندوز ضروری است. صرفا دانلود درایور کافی نیست و باید آن را نصب نمایید.

پس از دانلود، می توانید یک AVD مبتنی بر شبیه ساز Intel ایجاد نمایید. شبیه ساز با سرعت چندان بیشتری راه اندازی نمی شود، اما سرعت اجرای خود اپلیکیشن بسیار سریع تر خواهد بود.

6-3-1-تست برنامه بر روی یک دستگاه واقعی اندروید

ابتدا در دستگاه اندروید خود به بخش setting مراجعه نموده و در آنجا قابلیت USB Debugging را فعال نمایید. برای این منظور مسیر رو به رو را طی کنید: USB-Debugging ▸ Settings ▸ Development Options.

شاید لازم باشد درایور مورد نیاز برای گوشی همراه خود را نیز نصب نمایید. برای سیستم عامل های Linux و Mac OX این درایور از پیش نصب شده و آماده (out of box) در اختیار شما قرار می گیرد، اما برای سیستم عامل Windows شما می بایست درایور مربوطه را خود نصب نمایید.

توجه: پایین ترین ورژن اندرویدی که اپلیکیشن شما بر روی آن قابلیت اجرا دارد و به اصطلاح پشتیبانی می کند، می بایست با نسخه ی اندرویدی که بر روی دستگاه شما نصب است، سازگاری و هم خوانی داشته باشد.

زمانی که چندین دستگاه به کامپیوتر متصل کرده اید، می توانید مشخص کنید کدام یک مورد استفاده قرار گیرد. چنانچه تنها یک دستگاه به کامپیوتر وصل است، اپلیکیشن به صورت خودکار بر روی دستگاه مربوطه مستقر و نصب (deploy) می شود.

1-4-کامپوننت های(اجزای تشکیل دهنده)نرم افزاری یک اپلیکیشن اندروید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-4-1-اپلیکیشن اندروید

اپلیکیشن اندروید در حقیقت یک موجودیت واحد و قابل نصب است که می توان آن را راه اندازی نموده و مستقل از دیگر اپلیکیشن های اندورید مورد استفاده قرار داد. اجزای تشکیل دهنده یا به اصطلاح کامپوننت های نرم افزاری دو اپلیکیشن می توانند بر اساس یک آبجکت به نام Intent به یکدیگر وصل شوند. به تعبیر دیگر با استفاده از آبجکت Intent می توان عملیات یا task هایی تعریف کرده که بین چندین اپلیکیشن رفت و آمد دارند و اطلاعاتی را بین این دو منتقل می کنند.

اپلیکیشن های اندروید به طور کلی از کامپوننت های نرم افزاری، فایل های حاوی دستورات جاوا (source files) و فایل های محتوا (resource files) تشکیل می شوند. کامپوننت های نرم افزاری اندروید هر یک به همراه شرح چیستی و کاربرد آن در زیر عنوان شده اند:

1. کامپوننت Application: هر برنامه ی اندروید می تواند تنها یک کلاس Application داشته باشد که این کلاس قبل از دیگر کامپوننت های اندروید نمونه سازی می شود (یک نمونه یا آبجکت از این کلاس ساخته می شود). این جزء نرم افزاری همچنین آخرین کامپوننتی است که به هنگام خروج و بسته شدن برنامه، متوقف شده و چرخه ی حیات آن به اتمام می رسد.

   اگر خودتان به صورت صریح یک آبجکت از این کلاس نسازید، اندروید خود به صورت خودکار یکی برای شما ایجاد می کند.
2. Activity: جز نرم افزاری که یک صفحه از اپلیکیشن را ارائه داده و کاربر با تعامل با آن فعل خاصی را انجام می دهد، برای مثال با یکی از مخاطبین تماس گرفته، از منظره ای عکس می گیرد، یک نقشه را بر روی نمایشگر گوشی خود مشاهده می کند یا ایمیلی را ارسال می کند. به هر activity یک پنجره اختصاص می یابد که در آن رابط کاربری و ظاهر برنامه (UI) ترسیم می گردد.به عبارت دیگر، Activity جز ء اصلی اپلیکیشن های اندروید است که از دو بخش تشکیل شده: 1. کلاس که کد و رفتار برنامه را در آن پیاده سازی می کنید 2. لایه ی گرافیکی. یک اپلیکیشن می تواند چندین activity داشته باشد. activity ها با

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

استفاده از view ها و fragment ها ظاهر برنامه ی خود را ساخته و با کاربر تعامل برقرار می کنند.

3. Service: یکی از اجزا تشکیل دهنده ی اپلیکیشن های اندروید که عملیاتی را انجام می دهد ولی رابط کاربری یا UI ندارد. به عبارت دیگر، یک کامپوننت نرم افزاری که در پس زمینه کار می کند و با کاربر هیچ تعاملی ندارد. سرویس ها می توانند با دیگر کامپوننت های اندروید ارتباط بر قرار کنند. به عنوان مثال broadcast receiver می تواند از طریق notification framework به کاربر درباره ی رخداد خاصی خبر بدهد.
4. Broadcast receiver: یک Broadcast receiver می تواند به پیغام ها سیستمی و intent ها گوش فرا دهد و به اصطلاح منتظر باشد اتفاق خاصی رخ دهد. در واقع زمانی که event مورد نظر رخ می دهد، receiver ای که به آن گوش فرا می دهد توسط سیستم اندروید مطلع می گردد.

به عنوان مثال می توانید یک receiver تخصیص دهید که کار آن گوش دادن به پیغام های سیستمی یا intent ها است. یا receiver ای اعلان کنید که منتظر تغییر در وضعیت گوشی (زمانی که گوشی زنگ می زند) است.

به عبارت دیگر سیستم اندروید رخدادهایی را که اتفاق می افتد توسط broadcast اعلان می کند و شما می توانید برای این رخدادها receiver تعریف کنید که به رخداد گوش داده و در زمان فعال شدن رخداد از آن مطلع شود.

شما می توانید ویژه ی event مربوط به اتمام فرایند راه اندازی سیستم (boot)، یک receiver تعریف نمایید.

5. Content provider: یک جزء نرم افزاری که که رابط ساخت یافته یا یک پل ارتباطی به داده های اپلیکیشن فراهم می آورد. کامپوننت Content provider این قابلیت را دارد که داده های مورد نیاز چندین اپلیکیشن را در مکانی واحد به اشتراک گذاشته تا برنامه هایی که به این داده ها نیاز دارند بتوانند به آن داده ها دسترسی داشته و از آن ها استفاده کنند.

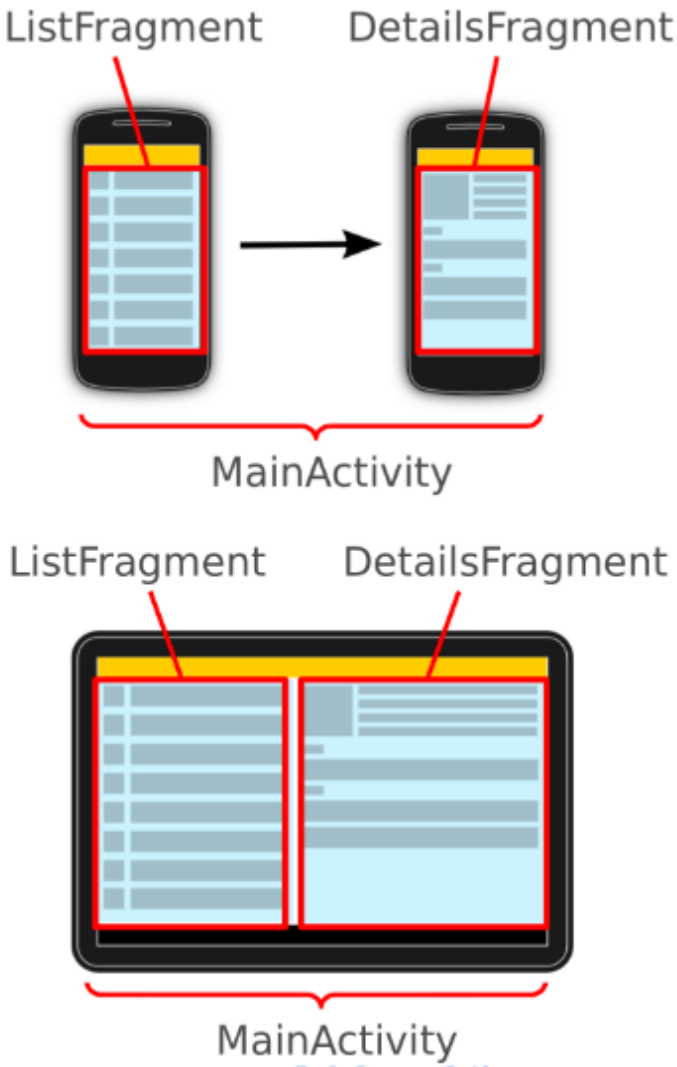
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

سیستم اندروید یک دیتابیس به نام SQLite دارد که به طور مکرر همراه با content provider مورد استفاده قرار می گیرد. دیتابیس داده ها را ذخیره می کند و سپس این داده ها توسط واسطی به نام provider در اختیار اپلیکیشن قرار می گیرد.

2-4-1-طراحی ظاهر و UI برنامه با استفاده از fragment ها، view ها و layout manager ها

Fragment ها کامپوننت هایی هستند که در بستر (context) یک activity که همان فرم در اندروید است، اجرا می شوند. fragment ها با کپسوله سازی کدهای اپلیکیشن در خود امکان استفاده ی مجدد از آن ها در activity های مختلف را فراهم آورده و همچنین توسعه دهنده را قادر می سازد برای دستگاه های اندروید با اندازه صفحه متفاوت UI داینامیک و انعطاف پذیر تعریف کنند. تصویر زیر یک activity به نام MainActivity را به نمایش می گذارد. Activity حاضر، در نمایشگر کوچک، تنها یک fragment را برای کاربر نمایش می دهد. این در حالی است که همین activity، به هنگام نمایش در دستگاه تبلت هر دو fragment را همزمان در یک صفحه به نمایش می گذارد.



View ها ابزارک های رابط کاربری (UI Widget) همچون دکمه یا text field هستند که به وسیله ی attribute می توان رفتار و ظاهرشان را تنظیم و ویرایش کرد.

در اندروید مفهومی به نام ViewGroup وجود دارد. ViewGroup خود یک view است که نقش میزبان یا ظرف را برای دیگر view ها (که در اصطلاح view های فرزند آن خوانده می شوند) ایفا می کند. به این view میزبان در اندروید layout manager نیز گفته می شود چرا که قادر است چینش view های دیگر را مدیریت کند.

کلاس پایه که layout manger ها از آن ارث بری می کنند android.view.ViewGroup است . این کلاس خود از android.view.View که کلاس پدر تمامی view ها است مشتق می شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

می توان با قرار دادن layout manager ها در دل یکدیگر (تودرتو کردن آن ها)، layout یا قالب های ترکیبی و پیچیده تری ایجاد کرد.

3-4-1-ابزارک های رابط کاربری یا widget های مورد استفاده در صفحه ی اصلی (home screen widget)

Widget ها یا ابزارک های رابط کاربری که در صفحه ی اصلی موبایل به نمایش در می آیند، در واقع یک سری broadcast receiver هستند که کامپوننت های تعاملی ارائه می دهند. شما از طریق این کامپوننت ها می توانید به بخش هایی از برنامه در صفحه ی اصلی موبایل (home screen) دسترسی داشته باشید. به عنوان مثال، یک ابزارک می تواند به کاربر اجازه دهد خلاصه ای از ایمیل های جدید را مشاهده کند و زمانی که کاربر بر روی ایمیل دلخواه خود کلیک کرد، کاربر را به صفحه ی مربوطه در اپلیکیشن هدایت کند.

Live wallpapers به شما این امکان را می دهند تا background های متحرک و پویا برای نمایش در صفحه ی اصلی سایت خود ایجاد کنید.

4-4-1-کلاس Context

نمونه یا آبجکت های ایجاد شده از کلاس android.content.Context، اتصال/امکان دسترسی به سیستم اندروید و دستگاهی که اپلیکیشن بر روی آن اجرا می شود را فراهم می آورد. به عبارت بهتر، نمونه های این کلاس به سیستم، منابع و سرویس های اپلیکیشن دسترسی دارند.

برای مثال، می توانید به وسیله ی Context اندازه ی دقیق صفحه نمایش دستگاه جاری را بدست بیاورید.

Activity ها و سرویس ها از کلاس Context ارث بری می کنند.

5-1-فایل تنظیمات اندروید (manifest)

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-5-1-تنظیمات اپلیکیشن های اندروید

کامپوننت ها، تنظیمات و metadata یک اپلیکیشن اندروید در فایل AndroidManifest.xml آن که فایل تنظیمات در سیستم اندروید است، قید می شود.

تمامی activity ها، سرویس ها و کامپوننت های content provider اپلیکیشن باید در این فایل و به صورت static تعریف شوند. اما Broadcast receiver را می توان به صورت static در فایل manifest یا به صورت dynamic و در زمان اجرا در اپلیکیشن تعریف کرد. سیستم اندروید این فایل تنظیمات را به هنگام نصب اپلیکیشن خوانده و سپس با توجه به اطلاعات داخل آن، قابلیت ها و امکانات متعدد اپلیکیشن مورد نظر را شناسایی می کند.

Gradle، سیستم کامپایل و دسته بندی کلاس های اندروید تحت یک پوشه ی واحد (build&packaging)، می تواند فایل manifest را ایجاد کرده و در اختیار شما قرار دهد. برای مثال، ورژن اپلیکیشن معمولا توسط فایل gradle build عرضه می شود.

1-5-2-نمونه ای از فایل تنظیمات اندروید (manifest)

در زیر نمونه ای از یک فایل ساده ی تنظیمات اندروید (manifest) را مشاهده می کنید:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.android.rssreader"
  android:versionCode="1"
  android:versionName="1.0" >
  <uses-sdk
    android:minSdkVersion="16"
    android:targetSdkVersion="19" />
  <uses-permission android:name="android.permission.INTERNET" />
  <application
    android:name="RssApplication"
    android:allowBackup="false"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <activity
      android:name="RssfeedActivity"
      android:label="@string/title_activity_main" >
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
```

```
    <activity
      android:name=".DetailActivity"
      android:label="Details" >
    </activity>
    <activity android:name="MyPreferenceActivity" >
    </activity>
    <service android:name="RssDownloadService" >
    </service>
  </application>
</manifest>
```

3-5-1-خصیصه ی Package و تعین نسخه های اپلیکیشن با استفاده از خصیصه های Version در فایل تنظیمات (manifest)

در فایل تنظیمات attribute یا خصیصه ای به نام package مشاهده می شود که اسم پکیجی که آبجکت ها (کلاس های) جاوا مورد اشاره در این فایل، داخل آن قرار دارند را مشخص می کند. در صورتی که یک آبجکت جاوا داخل پکیج دیگری قرار دارد، در آن صورت لازم است اسم کامل پکیج قید شود.

برای عرضه ی اپلیکیشن در فروشگاه مجازی Google play، آن اپلیکیشن باید یک اسم پکیج اختصاصی و منحصر بفرد داشته باشد. توصیه می شود برای اسم پکیج، اسم دامنه ی اپلیکیشن خود را به صورت وارونه بکار ببرید. بدین وسیله یک معرف یکتا برای اپلیکیشن اعلان می کنید که آن را از دیگر اپلیکیشن ها برای سیستم اندروید تمییز می دهد.

android:versionName و android:versionCode ورژن اپلیکیشن را مشخص می کنند. versionName اسم قابل مشاهده برای کاربرانی است که اپلیکیشن را بارگیری و نصب می کنند و هر می تواند هر مقدار رشته ای باشد.

مقدار versionCode باید یک عدد صحیح یا integer باشد. به وسیله ی این مقدار عددی Google play هم خود اپلیکیشن را شناسایی می کند و هم نسخه های جدید و آپدیت های آن را دنبال می کند. مقدار این خصیصه معمولا در اولین ورژن بر روی 1 تنظیم می شود و همین طور که ویرایش های جدید از آن منتشر می شود، این مقدار بالا می رود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

4-5-1-تگ <application> (یک ظرف برای اجزا) و کامپوننت های دیگر

به شما این امکان را می دهد تا برای اپلیکیشن خود metadata بنویسید (اطلاعاتی را پیرامون برنامه ی خود ارائه دهید) و علاوه بر آن یک کلاس application به صورت صریح تعریف نمایید. این تگ همچنین می تواند ظرفی برای تعریف دیگر کامپوننت های اندروید باشد.

توسط تگ <activity> یک کلاس activity تعریف می شود. در این تگ خصیصه (attribute) name به کلاسی اشاره دارد که اگر به طور کامل قید نشده باشد، نسبت به پکیجی که در خصیصه ی package آن اپلیکیشن قید کردید، آدرسی دهی می شود.

در واقع <action android:name="android.intent.action.MAIN" /> بیانگر این است که اپلیکیشن می تواند با این activity اجرا شود و این activity نقطه ی اجرا برنامه می باشد. پارامتر <category android:name="android.intent.category.LAUNCHER" /> نیز اعلان می دارد که activity حاضر بایستی به لانچر اضافه شود (در لیست اپلیکیشن های دستگاه به صورت فایل اجرایی ارائه شود).

مقدار <@string/app_name> به فایل های محتوا (resource) اشاره دارد که اسم اپلیکیشن در آن تعریف شده است. با بهره گیری از فایل های محتوا می توان به راحتی منابع، محتوای مختلف (مقادیر رشته ای، رنگ و آیکون برنامه) و مورد نیاز هر اپلکیشن را فراهم نموده و همچنین زبان برنامه را به راحتی تنظیم کرد (اطلاعات مربوط به زبان برنامه را فراهم و آن را ترجمه کرد).

تگ service، receiver و provider نیز می توانند مانند activity، نقش ظرف را برای تعریف دیگر کامپوننت و اجزا اندروید ایفا کنند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

5-5-1-تعیین پایین/بالاترین نسخه ی اندروید که برنامه بر روی آن اجرا می شود (خصیصه های minSdkVersion و targetSdkVersion)

بخش uses-sdk در فایل تنظیمات به شما این امکان را می دهد که پایین ترین نسخه و همچنین نسخه ی مورد نظر از اندروید که می خواهید اپلیکیشن بر روی آن قابل اجرا باشد را به ترتیب از طریق attribute های minSdkVersion و targetSdkVersion مشخص نمایید.

1. minSdkVersion: این خصیصه به شما امکان می دهد پایین ترین ویرایش اندروید که اپلیکیشن شما بر روی آن قابل اجرا می باشد را مشخص نمایید. مقداری که اینجا مشخص می کنید در واقع به منزله ی یک فیلتر در فروشگاه مجازی Google Play مورد استفاده قرار می گیرد، به طوری که دستگاه هایی که اندروید نصب شده بر روی آن پایین تر از مقدار مشخص شده در این attribute است، امکان نصب آن را پیدا نمی کنند.
2. targetSdkVersion: مقدار این attribute نشانگر ورژنی از سیستم عامل اندروید است که اپلیکیشن در اصل ویژه ی آن طراحی و بر روی آن تست شده است. لازم به ذکر است که این مقدار برابر ورژن کتابخانه های اندروید یا API version دستگاه نیست. سیستم اندروید ممکن است جهت سازگاری و قابلیت اجرای برنامه بر روی ورژن های جدید یا قدیمی تر تغییراتی را اعمال کند (forward/backward compatibility). توصیه می شود این attribute را جهت بهره وری از آخرین امکانات اندروید، برابر آخرین API version قرار دهید.

5-5-1-تعیین مجوزهای دسترسی با استفاده از تگ permission

فایل تنظیمات اندروید (manifest) همچنین می بایست مجوزهای لازم را مشخص کند. به عنوان مثال، در صورتی که اپلیکیشن به اینترنت نیاز دارد، مجوز دسترسی به آن باید در این فایل تعریف و اعطا شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

اپلیکیشن با استفاده از تگ <uses-permission> به سیستم اندروید اعلان می کند که به مجوز خاصی احتیاج دارد.

برخی از مجوزها، همچون دسترسی به اینترنت، از اندروید 6.0 به بعد به صورت خودکار اعطا می شوند. اما برخی دیگر جهت فعال شدن به تاییدیه ی کاربر نیاز دارند.

7-5-1-تعیین سیستم سخت افزاری مورد نیاز (بخش uses-configuration و uses-feature در فایل تنظیمات)

در فایل تنظیمات (manifest) اندروید همچنین بخشی به نام uses-configuration وجود دارد که روش های دریافت ورودی برای دستگاه را مشخص می کند. برای مثال، تکه کد زیر اعلان می کند که دستگاه مورد نظر بایستی از صفحه کلید مجزا (به صورت سخت افزاری) برخوردار باشد.

```
<uses-configuration android:reqHardKeyboard="true"/>
```

بخش uses-feature به شما این امکان را می دهد تا سیستم سخت افزاری که دستگاه میزبان اپلیکیشن باید داشته باشد را مشخص نمایید. به عنوان مثال، تکه کد زیر اعلان می کند که دستگاه مورد نظر باید برای اجرای اپلیکیشن و بهره وری از قابلیت های آن، دوربین داشته باشد.

```
<uses-feature android:name="android.hardware.camera" />
```

8-5-1-تعیین محل نصب (خصیصه ی installLocation)

به واسطه ی خصیصه ی installLocation می توانید مشخص کنید آیا اپلیکیشن شما اجازه یا امکان نصب بر روی حافظه ی خارجی دستگاه را دارد یا خیر. برای اعطای این مجوز کافی است attribute ذکر شده را با یکی از دو مقدار auto یا preferExternal برابر قرار دهید.

در حقیقت، این امکان به ندرت مورد استفاده قرار می گیرد چرا که اپلیکیشن (مسقر بر روی حافظه ی خارجی) به مجرد وصل شدن به کامپیوتر و استفاده از آن به عنوان یک حافظه ی USB، بلافاصله متوقف می شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

6-1-فایل های محتوا (Resources)

اندروید به شما اجازه می دهد فایل های محتوای پروژه ی خود را به صورت مجزا و جدا از کد برنامه (source code) در قالب منابع static (فایل های تنظیمات مبتنی بر XML ، عکس، متن و غیره ...) تعریف نمایید.

فایل های محتوا بایستی داخل پوشه ی res/ اپلیکیشن شما، در زیرپوشه ای از پیش تعریف شده جایگذاری شود. این زیرپوشه معمولا به نوع محتوایی که در آن قرار داده می شود، بستگی دارد. در صورت لزوم می توانید شناسه های بیشتری به اسم پوشه الصاق نموده و از این طریق مشخص نمایید محتوای مربوطه باید برای تنظیمات خاصی مورد استفاده قرار گیرد. این شناسه ها در اصطلاح resource qualifier خوانده می شوند. به عنوان مثال، می توانید مشخص کنید که فایل layout یا تنظیم چیدمان های المان رابط کاربری فقط برای اندازه صفحه ی خاصی قابل استفاده می باشد.

جدول زیر توضیح مختصری در خصوص منابع (resource) مورد پشتیبانی و پیشوند مربوطه ی هر یک ارائه می دهد.

Resource یا محتوای مورد نظر	پوشه ی مربوطه	شرح
Drawables	*/res/drawables*	فایل های تصویری (برای مثال png، jpeg) drawable/ های برداری (vector)، فایل های XML که به طور خودکار خود را متناسب با تراکم پیکسلی نمایشگر اندازه بندی و تنظیم می کنند را دربرمی گیرد. می توان از طریق کلاس R.drawable به آن ها دسترسی داشت.
Simple Values	*/res/values*	داخل این فایل مقادیر متنی یا رشته ای، رنگ ها، ابعاد و اندازه، استایل ها و آرایه ای از رشته ها یا اعداد صحیح در قالب فایل XML تعریف و ذخیره می

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Resource یا محتوای مورد نظر	پوشه ی مربوطه	شرح
		شوند. به طور قرار دادی، هر نوع در پوشه ی جدا و مختص به خود قرار می گیرد، برای مثال مقادیر رشته ای در آدرس res/values/strings.xml ذخیره می شوند.
Layouts	*/res/layout*	فایل های XML با مقادیر مربوط به layout اپلیکیشن که امکان طراحی محیط کاربری با activity ها و fragment ها را فراهم می آورند در این مسیر نگهداری می شوند.
Styles and themes	*/res/values*	فایل هایی که ظاهر و سبک نمایش اپلیکیشن اندروید را تعریف می کنند.
Animations	*/res/animator*	می توانید دستورات انیمیشن را به صورت xml داخل این فایل (برای کار با animation API) تعریف کنید که به شما اجازه می دهد مقادیر property های آبجکت را جهت اجرای انیمیشن و ادامه ی آن تا مدت زمان خاص، ویرایش نمایید. به عبارت دیگر فایل های ویژگی های انیمیشن را تعریف می کنند، در این مسیر ذخیره می شوند.
Raw data	*/res/raw*	داده هایی که می خواهید با فرمت خام خود ذخیره شوند را در این فایل تعریف می کنید. می توانید از طریق آبجکت InputStream به راحتی به آن ها دسترسی داشته باشید. در واقع بهتر است فایل های که فرمت آن ها با دیگر فایل متفاوت است در این پوشه جایگذاری نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Resource یا محتوای مورد نظر	پوشه ی مربوطه	شرح
Menus	*/res/menu*	فایل های XML ای که منوهای اپلیکیشن را تشکیل می دهند در این پوشه قرار دارند. برای دسترسی به آن ها کافی است از R.menu استفاده نمایید.

1-6-1-نمونه فایل resource (تعریف تعدادی ثابت رشته ای، آرایه ی رشته ای، ثابت رنگ و ابعاد)

به یک نمونه فایل resource می توان به values.xml در پوشه ی /res/values اشاره کرد. این فایل دربردارنده ی تعدادی ثابت رشته ای، آرایه ی از نوع رشته، ثابتی حاوی مقدار رنگ و ثابتی حاوی مقدار ابعاد می باشد.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <string name="app_name">Test</string>
   <string name="action_settings">Settings</string>
   <string name="hello_world">Hello world!</string>
   <string-array name="operationsystems">
      <item>Ubuntu</item>
      <item>Android</item>
      <item>Microsoft Windows</item>
   </string-array>
   <color name="red">#ffff0000</color>
   <dimen name="mymargin">10dp</dimen>
</resources>
```

1-6-2-فایل های resource و R.java

سیستم build و کامپایل اندروید به تمامی فایل های محتوا و resource های مربوطه در پوشه ی res، یک ID منحصر بفرد تخصیص می دهد. اندروید سپس یک فایل به نام R.java ایجاد می کند که مقادیر این ID ها را در خود ذخیره می کند. ID ها (اشاره گرها) در واقع مقادیر static و عددی از نوع int هستند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بلافاصله پس از ایجاد یک فایل resource جدید، ID متناظر با آن به صورت خودکار در فایل R.java ایجاد می شود. نیازی به تغییرات دستی در فایل R.java نیست و در صورت اعمال چنین تغییراتی هم، Android development tooling آن ها را بازنویسی می کند. سیستم اندروید تعدادی متد ارائه می دهد که ID مربوط به یک resource را به عنوان آرگومان پذیرفته و فایل resource متناظر آن را بازیابی می کند.

برای مثال، جهت دسترسی به یک String با ID منحصر بفرد R.string.yourString در کد برنامه، شما می بایست متد getString(R.string.yourString) از کلاس Context را فراخوانی نمایید.

3-6-1-فایل های Layout (فایل های چیدمان رابط کاربری)

Activity ها (یا فرم ها در اندروید) ظاهر و UI خود را با استفاده از view ها (widget یا ابزارک های رابط کاربری) و fragment ها می سازند. در سیستم عامل اندروید کدهای مربوط به چیدمان المان های UI و طراحی رابط کاربری در قالب فایل های محتوا مبتنی بر XML در پوشه ی <filename class="filename">/res/layout_ تعریف می شوند یا در قالب کد جاوا نوشته و ذخیره می گردند. در صورت تمایل می توانید تلفیقی از دو روش را بکار ببرید، هر چند روش اول بیشتر توصیه می شود. بدین وسیله می توان منطق برنامه (programming logic) را از ظاهر آن (layout definition) جدا کرد. همچنین این امکان بوجود می آید که برای هر دستگاه ظاهر و فایل layout ویژه ای تعریف نمایید.

فایل محتوا (resource file) که کدهای UI برنامه از آن خوانده می شوند در اصطلاح برنامه نویسی اندروید layout گفته می شود. Layout یا فایل تنظیم چیدمان المان های رابط کاربری تمامی view ها، ViewGroup ها و رابطه میان آن ها و attribute های آن ها را در قالب فایل های XML تعریف می کنند.

کد زیر یک نمونه ساده از فایل layout را به نمایش می گذارد.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingBottom="@dimen/activity_vertical_margin"
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >
    <TextView
        android:id="@+id/mytext"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello_world" />
</RelativeLayout>
```

فایل layout با فراخوانی متد setContentView به یک activity جهت نمایش در UI تخصیص می یابد، مانند زیر:

```
package com.vogella.android.first;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
public class MainActivity extends Activity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
  }
}
```

1-6-4-افزایش سرعت و کارایی با استفاده از layout های ساده

محاسبه، پردازش و ترسیم view ها یک عملیات سنگین بوده و منابع و محتوای حجیمی را شامل می شود. شما بایستی تا حد امکان از layout های ساده استفاده نمایید تا سرعت اجرا و کارایی کلی برنامه بهبود یابد. برای مثال، بهتر است در شرایطی که یک layout manager ساده کفایت می کند، از تودرتو سازی بیش از حد layout manager ها خودداری نمایید.

1-6-5-رهنمودها و روش های بهینه در خصوص ID های اختصاص داده شده به فایل های محتوا/resource

به منظور دسترسی به یک view از کد جاوا یا XML، لازم است با تنظیم مقدار attribute ای به نام android:id، به view دلخواه ID یا شناسه ی منحصر بفردی تخصیص دهید. جهت تخصیص یک ID جدید به view مورد نظر، کافی است خصیصه ی android:id از المان متناظر را در فایل layout با مقدار دلخواه تنظیم نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

SDK یا مجموعه ابزار ساخت توسعه برنامه های اندرویدی، از سیستم نشانه گذاری camelCase برای نگارش مقدار ID بهره می گیرد.

مثال: buttonRefresh. توصیه می شود شما نیز از همین سیستم استفاده نمایید.

در نمونه ی زیر می بینید که با تنظیم مقدار attribute نام برده (خصیصه ی android:id المان مربوطه) بر روی پارامتر رشته ای "@+id/button1"، به المان دکمه یک شناسه ی منحصر بفرد اختصاص داده می شود. طبق قراردادهای تعریف شده، این دستور یک ID جدید در فایل R.java ایجاد کرده و سپس آن را به view مربوطه انتساب می دهد.

```
<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Show Preferences" >
</Button>
```

توصیه می شود تمامی ID ها را در یک فایل تنظیمات (Config) واحد تعریف نمایید. این فایل معمولا ids.xml نام گذاری شده و در پوشه ی /res/values جایگذاری می شود. با این کار شما می توانید ID های از پیش تعریف شده را در فایل layout مورد استفاده قرار دهید. لازم به ذکر است که شما می توانید ID ها را در فایل جداگانه تعریف نمایید، اما در آن صورت بایستی آیتم id+@ را از فایل های layout خود حذف نمایید چرا که در غیر این صورت با یک پیغام خطا مبنی بر اینکه این فایل ها قبلا ساخته شده اند مواجه می شوید. کد زیر نمونه ای از این فایل را به نمایش می گذارد.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <item name="button1" type="id"/>
</resources>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity" >
   <Button
      android:id="@id/button1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="true"
      android:layout_marginRight="27dp"
      android:text="Button" />
</RelativeLayout>
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

توجه: روشی که هم اکنون از آن نام برده شده برای پروژه های واقعی بسیار کارامد و مناسب می باشد. اما در آموزش حاضر، به علت زمان بر بودن این پروسه، مورد استفاده قرار نمی گیرد.

6-6-1-محتوای و منابعی که توسط سیستم اندروید ارائه می شود (system resources)

سیستم اندروید خود نیز محتوا و resource ارائه می دهد. به این محتویات و منابع system resources گفته می شود. فایل های resource که خود سیستم در اختیار برنامه نویس قرار می دهد توسط پیشوند android از دیگر فایل های محتوا (local resource) متمایز می شوند. به عنوان مثال، android.R.string.cancel یک string است که محیط اندروید (platform) ویژه ی عملیات cancel تعریف می کند.

7-1-مفهوم view در اندروید – المان ها و ابزارک های رابط کاربری یا UI Widget

همان طور که می دانید یک اپلیکیشن اندروید دارای یک یا چند activity هست. Activity همان فرم یا پنجره در برنامه های تحت ویندوز است. حال داخل هر activity شما می توانید المان های رابط کاربری همچون دکمه، کادر متن و غیره داشته باشید. این اجزا رابط کاربری در واقع نمونه هایی از کلاس view یا subclass های (کلاس های ارث بری شده از) ViewGroup هستند.

به عبارت دیگر view کلاس پدر (superclass) است که تمامی کامپوننت های GUI در اندروید از آن ارث بری می شوند. به عنوان مثال، کلاس TextView که یک برچسب متنی را در UI به نمایش می گذارد یک کلاس فرزند از View است (ViewGroup نیز یک کلاس مشتق شده از View است. از خود ViewGroup آبجکت هایی ساخته می شود که این آبجکت ها نقش ظرف را برای گروه بندی نمونه های کلاس View ایفا می کند. به عنوان مثال می توان به کلاس LinearLayout اشاره کرد که یک کلاس فرزند از ViewGroup است.)

هر view در اندروید نماینده ی یک widget (ابزارک یا المان کاربری) همچون دکمه یا یک layout manager هست. SDK سیستم اندروید این ابزارک یا المان های رابط کاربری را از طریق کلاس

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Button، TextView و EditText در اختیار شما قرار می دهد (ListView که لیستی از آیتم ها را در نمایشگر به صورت فهرست ارائه می دهد) .

تمامی view ها از کلاس android.view.View ارث برده می شوند. این کلاس پدر، بسیار سنگین و حاوی حدودا 18 هزار خط کد است که قابلیت های پایه ای مختلفی را در اختیار کلاس های فرزند خود قرار می دهد.

کلاس های پایه ی view ها همگی در پوشه ی android.view قرار دارند و widget های پیش فرض محیط اندروید (platform) داخل پوشه ی android.widget.

1-8- layout manager و ViewGroup

1-8-1- استفاده از layout manager

Layout manager وظیفه ی تنظیم ظاهر خود و مدیریت چیدمان view های فرزندش را بر عهده دارد. در واقع Layout manager یک کلاس مشتق شده از ViewGroup است.

Layout manager های متعددی در سیستم اندروید تعبیه شده که در زیر به پرکاربردین آن ها اشاره می شود:

- ConstraintLayout - توسط یک کتابخانه ی خارجی به اندروید اضافه می شود.
- LinearLayout
- FrameLayout
- RelativeLayout
- GridLayout

1-8-2- تنظیم و ویرایش layout manager ها از طریق attribute ها

تمامی layout manager ها را می توان با ویرایش مقادیر attribute به آسانی طبق نیاز تنظیم کرد. View های موجود در layout manager می توانند attribute هایی داشته باشند و از طریق آن ها ظاهر خود همچون اندازه ی عرض و طول را مشخص کنند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

View های موجود در layout manager می توانند طول و عرض مد نظر خود را به واسطه ی attribute های زیر تنظیم کنند.

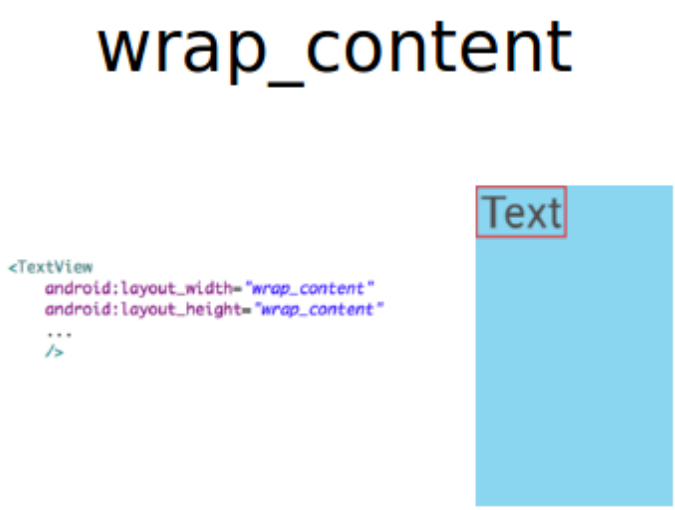
- android:layout_width : عرض widget را تعیین می کند.
- android:layout_height : طول widget را مشخص می کند.

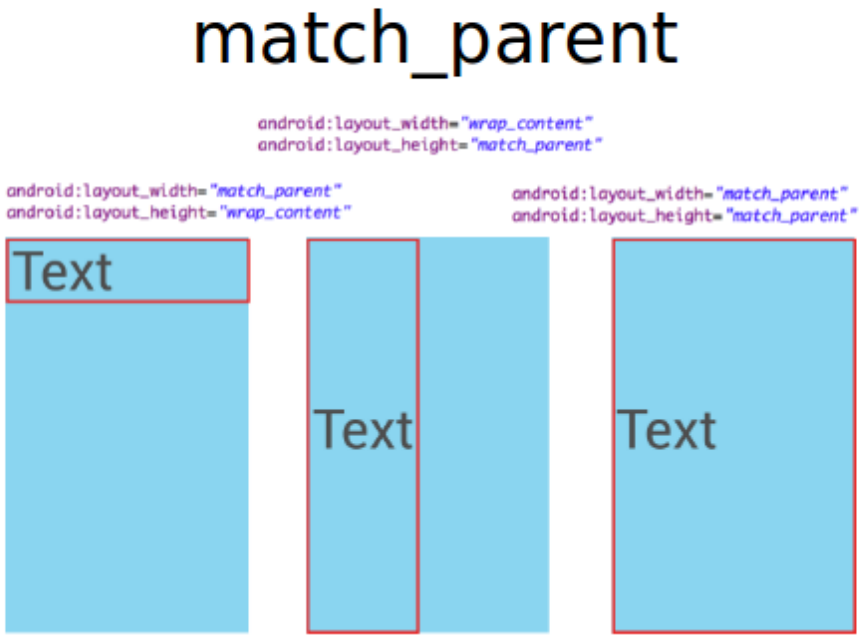
اندازه ی layout manager ها را می توان بر حسب واحد های اندازه گیری متعارف مشخص نمود یا مقادیر آماده و از پیش تعریف شده ی layout را مورد استفاده قرار داد. مثال: 100dp.

مقدار match_parent به اپلیکیشن دستور می دهد widget موجود در layout manager (یا view پدر) را دقیقا از لحاظ اندازه با view پدر (میزبان) برابر قرار دهد. در مقابل این پارامتر، مقدار wrap_content وجود دارد که به layout اعلان می کند تنها به میزان مورد نیاز به widget فضا تخصیص دهد. در تصویر زیر اثر هر دو پارامتر به نمایش گذاشته شده است.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330





Constraint Layout_1_8_3

Constraint layout یکی از ابزار کنترل ظاهر و چیدمان المان های UI است که توسط کتابخانه ی خارجی در اختیار توسعه دهنده قرار می گیرد. این نوع layout به شما امکان استفاده از view hierarchy تخت و flat را می دهد که در افزایش سرعت اجرای اپلیکیشن بسیار موثر است. علاوه بر آن تمامی ابزارهای طراحی (design tool) به راحتی از constraint layout پشتیبانی می کنند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در پروژه های جدید بهتر است که از constraint layout برای طراحی ظاهر اپلیکیشن استفاده شود.

در واقع با constraint layout می توان قالب های انعطاف پذیر، واکنشگرا و پیچیده (تودرتو) تعریف کرد که علاوه بر قابلیت تطبیق خودکار خود با نمایشگرهای مختلف و تغییر جهت نمایش یا وضعیت چیدمان، می تواند سرعت بالایی داشته باشد.

FrameLayout_1_8_4

FrameLayout یک layout manager است که تمامی المان های داخل خود (child element) را بر روی هم ترسیم می کند. این به شما امکان می دهد جلوه های بصری و افکت های جالب طراحی کنید.

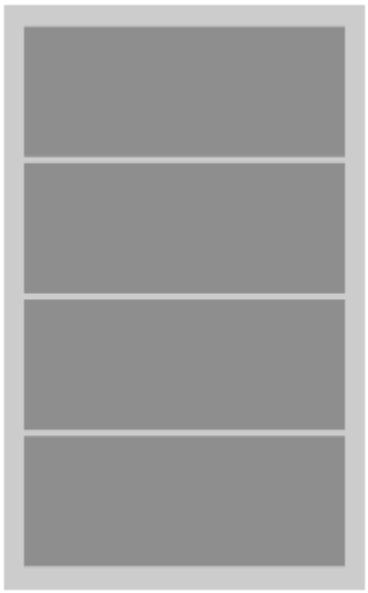
تصویر زیر اپلیکیشن Gmail را به نمایش می گذارد که با استفاده از FrameLayout چندین دکمه را بر روی layout دیگر قرار می دهد. در این layout المان ها می توانند بر روی یکدیگر قرار گیرند.

LinearLayout_1_8_5

LinearLayout تمامی المان های داخل خود را، بر اساس مقدار خصیصه ی orientation، دورن یک ستون یا سطر واحد قرار می دهد. در واقع با استفاده از این layout المان های فرزند همگی پشت سرهم در یک مسیر قرار می گیرند. همان طور که گفته شد، این layout یک attribute به نام orientation دارد که می توان آن را با دو مقدار مختلف تنظیم کرد: vertical و horizontal. مقدار horizontal که المان مربوطه را به صورت افقی نمایش می دهد، حالت پیش فرض می باشد.

در صورت مقداردهی attribute نام برده با horizontal، المان های داخل layout manager به صورت زیر در نمایش گر ارائه می شوند.

در صورت تنظیم مقدار attribute بر روی vertical، المان های داخل layout manager به صورت زیر در نمایشگر نشان داده می شود:



LinearLayout را می توان در حالت nested و تودرتو بکار برد. به عبارت دیگر این امکان وجود دارد که با تعریف کردن چند layout داخل یکدیگر، قالب های پیچیده و جالب تر ایجاد نمود.

در LinearLayout شما می توانید اندازه ی هر یک از المان های حاضر در layout manager را با مقداردهی خصیصه ی android:layout_weight تنظیم نمایید. این مقدار مشخص می کند چه میزان از فضای اضافی موجود به هر یک از المان ها اختصاص داده شود. برای مثال، چنانچه در قالب خود دو widget داشته باشید و خصیصه ی layout_weight یکی را با 1 و دیگری را با 2

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

مقداردهی کنید، در آن صورت اولین المان 1/3 فضای موجود را می گیرد و دومین المان 2/3 فضا را به خود اختصاص می دهد. همچنین می توانید با قرار دادن مقدار layout_width بر روی 0 یک نسبت مشخص را رعایت کنید.

Inflate کردن: تبدیل فایل LAYOUT XML به کلاس view جهت قرار گرفتن در کلاس ACTIVITY.

RelativeLayout_1_8_6

در این نوع layout، المان های رابط کاربری (widget) نسب به محل قرارگیری یکدیگر چیده می شوند. relativelayout را می توانید برای ساخت رابط های کاربری پیچیده مورد استفاده قرار دهید. در واقع RelativeLayout یک layout manager پیچیده و قدرتمند است که فقط باید در شرایط پیچیده مورد استفاده قرار گیرد زیرا این قالب برای چیدمان و تنظیم ظاهر المان های داخل خود به منابع زیاد و محاسبات سنگین احتیاج دارد.

یکی از موارد استفاده ی RelativeLayout زمانی است که می خواهید فقط یک کامپوننت را در مرکز صفحه قرار دهید. برای این منظور کافی است یک کامپوننت به RelativeLayout اضافه نموده و مقدار خصیصه ی android:layout_centerInParent را بر روی true تنظیم کنید.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <ProgressBar
      android:id="@+id/progressBar1"
      style="?android:attr/progressBarStyleLarge"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_centerInParent="true"
       />
</RelativeLayout>
```

GridLayout_1_8_7

GridLayout برای اولین بار در ویرایش 4.0 اندروید ارائه شد. layout نام برده به شما این امکان را می دهد تا فضای جاری را خانه بندی کنید و هر view می تواند یک یا چند خانه را اشغال کند. در واقع GridLayout قالب را مانند یک جدول به خانه، سطر و ستون تقسیم بندی می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

به وسیله ی این layout manager شما می توانید دقیقا تعیین کنید هر View چند ستون را به خود تخصیص دهد، در کدام سطر و ستون قرار گیرد و در نهایت چند سطر و ستون را اشغال کند. اگر خودتان به صورت صریح مشخص نکنید، در آن صورت GridLayout مقادیر پیش فرض را تخصیص می دهد. برای مثال، به هر view یک ستون و یک سطر اختصاص می یابد و محل view نیز به ترتیبی که در کد تعریف کردید مشخص می شود.

فایل XML زیر یک قالب با GridLayout تعریف می کند.

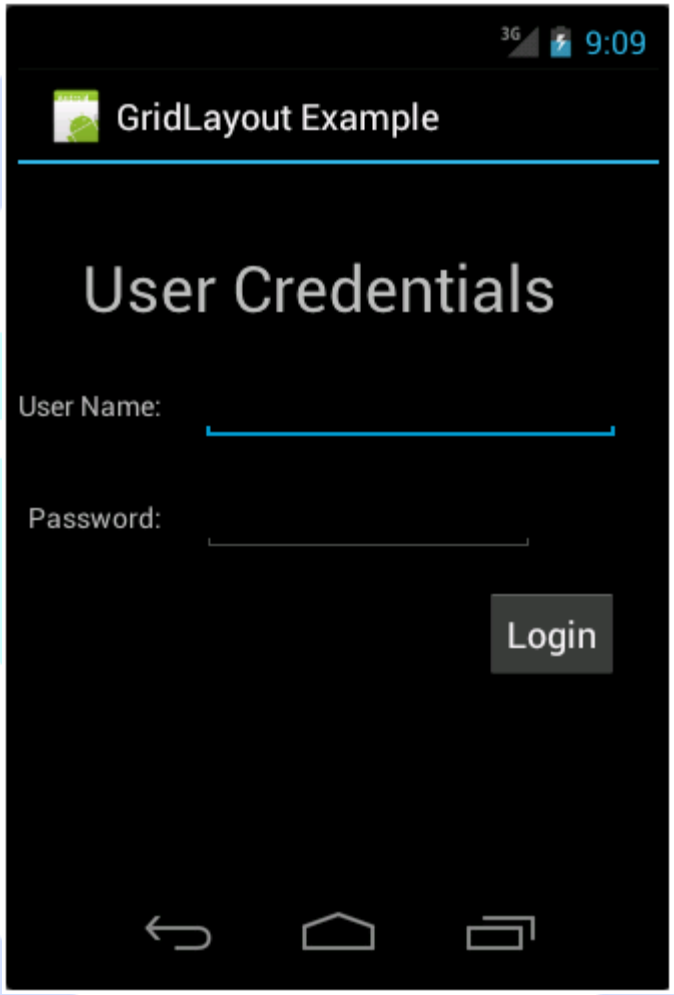
```
<?xml version="1.0" encoding="utf-8"?>
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/GridLayout1"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:columnCount="4"
   android:useDefaultMargins="true" >
   <TextView
      android:layout_column="0"
      android:layout_columnSpan="3"
      android:layout_gravity="center_horizontal"
      android:layout_marginTop="40dp"
      android:layout_row="0"
      android:text="User Credentials"
      android:textSize="32dip" />
   <TextView
      android:layout_column="0"
      android:layout_gravity="right"
      android:layout_row="1"
      android:text="User Name: " >
   </TextView>
   <EditText
      android:id="@+id/input1"
      android:layout_column="1"
      android:layout_columnSpan="2"
      android:layout_row="1"
      android:ems="10" />
   <TextView
      android:layout_column="0"
      android:layout_gravity="right"
      android:layout_row="2"
      android:text="Password: " >
   </TextView>
   <EditText
      android:id="@+id/input2"
      android:layout_column="1"
      android:layout_columnSpan="2"
      android:layout_row="2"
      android:inputType="textPassword"
      android:ems="8" />
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    <Button
        android:id="@+id/button1"
        android:layout_column="2"
        android:layout_row="3"
        android:text="Login" />
</GridLayout>
```

این کد ظاهری مشابه آنچه در تصویر زیر مشاهده می کنید را تعریف می کند.



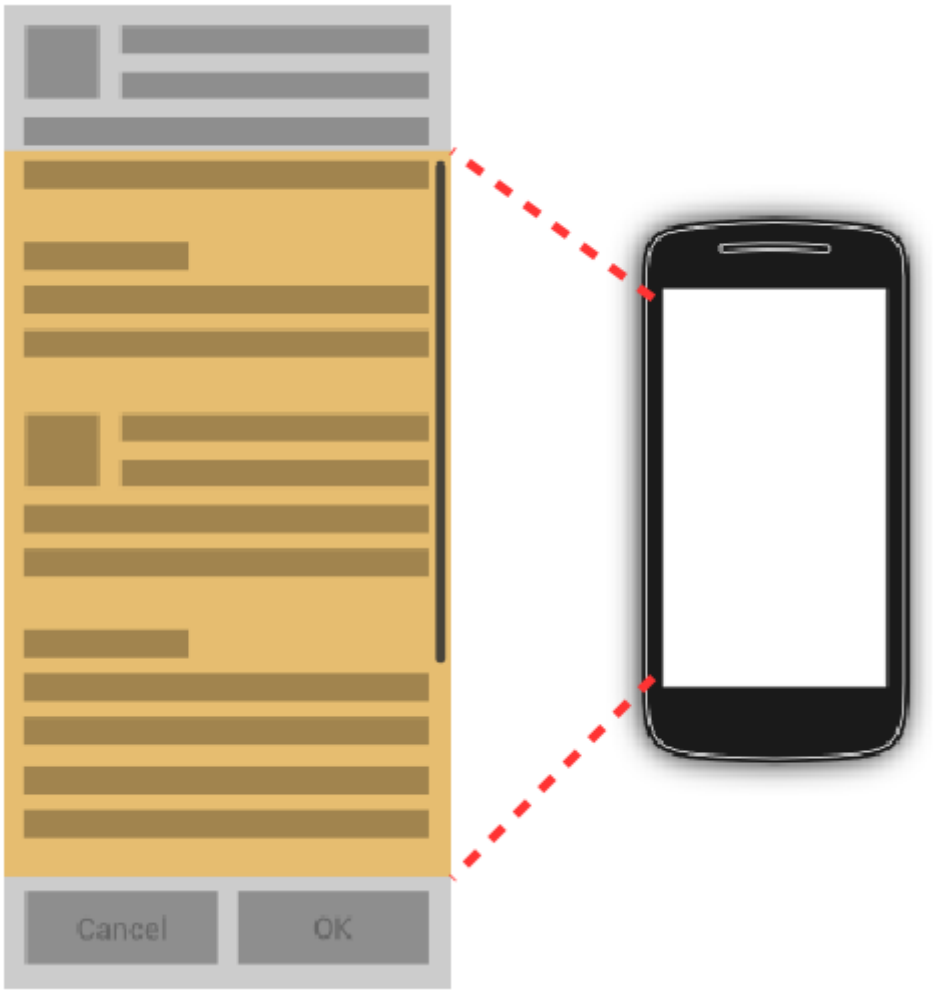
ScrollView-1-8-8

گاهی لازم است تمامی view ها را در UI به نمایش بگذارید، حتی اگر فضای کافی برای نمایش آن ها وجود نداشته باشد. در صورتی که view بیش از حد بزرگ باشد، آنگاه ScrollView یک نوار پیمایش به صفحه اضافه نموده و به شما امکان می دهد به وسیله ی آن کل محتوا یا المان مورد نظر را (با کشیدن نوار به سمت چپ یا راست) در نمایشگر مشاهده نمایید. لازم به ذکر است که ScrollView یا HorizontalScrollView هیچ یک layout manager محسوب نمی

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

شوند، بلکه خود می توانند یک view میزبان باشند و آن view تعدادی فرزند در خود داشته باشد.



در زیر یک فایل XML مشاهده می کنید که در آن از ScrollView (جهت نمایش کل محتوای صفحه با نوار پیمایش) استفاده شده است.

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:fillViewport="true"
   android:orientation="vertical" >
   <TextView
      android:id="@+id/TextView01"
      android:layout_width="wrap_content"
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    android:layout_height="wrap_content"
    android:paddingLeft="8dip"
    android:paddingRight="8dip"
    android:paddingTop="8dip"
    android:text="This is a header"  android:textAppearance="?android:attr/textAppearanceLarge" >
  </TextView>
</ScrollView>
```

خصیصه ی android:fillViewport="true" سبب می شود که ScrollView کل صفحه را در بر گیرد حتی اگر المان ها کوچکتر از کل صفحه باشند و اساسا نیازی به نوار پیمایش وجود نداشته باشد.

تمرین: ویرایش view layout (تنظیم چیدمان المان های رابط کاربری) در زمان اجرا

در این تمرین شما تعدادی radio button به قالب (layout) اپلیکیشن که در تمرین قبلی ایجاد کردید، اضافه خواهید نمود. بسته به انتخابی که کاربر می کند، چیدمان دکمه از افقی به عمودی و بالعکس تغییر می کند.

9-8-1-افزودن radio group و radio button به قالب/layout

فایل layout را باز نمایید و سپس یک radio group با دو radio button به قالب برنامه ی خود اضافه نمایید.

شناسه ی المان ها را بر اساس جدول زیر مقداردهی کنید.

جدول تخصیص ID به المان ها	
ID	View
orientation	Radio Group
horizontal	First radio button

جدول تخصیص ID به المان ها	
ID	View
vertical	Second radio button

در حال حاضر کد موجود در فایل XML می بایست به صورت زیر باشد. با کمی دقت متوجه می شود که تنها تگ RadioGroup در این layout جدید می باشد.

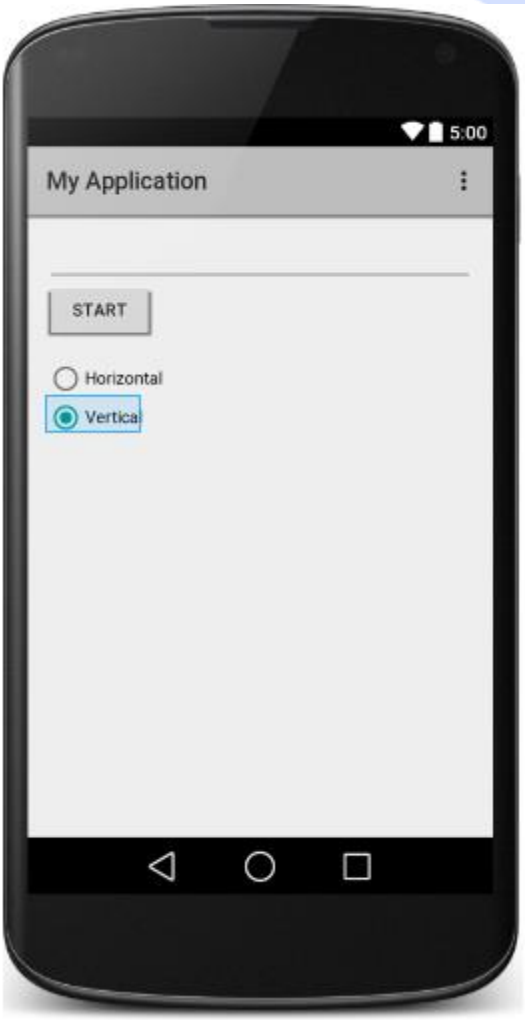
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   tools:context=".MainActivity" >
   <EditText
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:id="@+id/main_input"
      android:layout_alignParentTop="true"
      android:layout_alignParentStart="true"
      android:layout_alignParentEnd="true" />
   <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Start"
      android:id="@+id/button"
      android:layout_below="@id/main_input"
      android:layout_alignParentStart="true"
      android:onClick="onClick"/>
   <RadioGroup
      android:id="@+id/orientation"
      android:layout_below="@id/button"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="16dp">
      <RadioButton
         android:id="@+id/horizontal"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Horizontal" >
      </RadioButton>
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    <RadioButton
      android:id="@+id/vertical"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:checked="true"
      android:text="Vertical" >
    </RadioButton>
  </RadioGroup>
</RelativeLayout>
```

کد فوق خروجی زیر را در UI به نمایش می گذارد:



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

10-8-1-ویرایش orientation/وضعیت چیدمان radio group در زمان اجرا (به صورت dynamic)

تابع onCreate() را در کلاس activity ویرایش نمایید. سپس با فراخوانی متد findViewById() کلاس RadioGroup را در فایل layout (فایل xml که ظاهر برنامه را تعریف می کند) پیدا کنید.

یک Linstener داخل radio group پیاده سازی نمایید که به تغییرات گوش فرا داده و وضعیت چیدمان یا orientation دکمه را بر اساس انتخاب جاری کاربر تغییر دهد. اینکه کدام دکمه انتخاب شده را می توان توسط پارامتر ID تشخیص داد.

RadioGroup را در قالب متغیر در کلاس activity خود اعلان نمایید. سپس RadioGroup.OnCheckedChangeListener را به واسطه ی متد setOnCheckedChangeListener() از پکیج android.widget.RadioGroup اضافه نمایید. این متد (listener) متنظر انتخاب کاربر می ماند و به محض رخداد اتفاق مورد نظر (تغییر انتخاب کاربر) از آن مطلع شده و متعاقبا دستورات پیاده سازی شده در onCheckedChanged را اجرا می کند.

می توانید از قالب آماده ی زیر به عنوان الگو جهت پیاده سازی listener استفاده نمایید.

```
final RadioGroup group1 = (RadioGroup) findViewById(R.id.orientation);
group1.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(RadioGroup group, int checkedId) {
        switch (checkedId) {
            case R.id.horizontal:
                group.setOrientation(LinearLayout.HORIZONTAL);
                break;
            case R.id.vertical:
                group.setOrientation(LinearLayout.VERTICAL);
                break;
        }
    }
});
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-8-11-تست اپلیکیشن

برنامه ی خود را اجرا نموده و انتخاب خود را تغییر دهید. بایستی بر اساس گزینه ی مورد انتخاب orientation دکمه ها تغییر کند.

1-9-دسترسی به محتوای static و استفاده از آن

1-9-1-دسترسی به فایل های محتوا (resources) از طریق کلاس Resources

کلاس Resources به شما اجازه می دهد به منابع و محتوای اپلیکیشن خود به صورت مجزا و فردی دسترسی داشته باشید. می توانید با فراخوانی متد getResources() از کلاس Context به راحتی به نمونه ی کلاس Resources دسترسی داشته باشید. از آنجایی که کلاس های activity و service همگی از Context ارث بری دارند، شما می توانید این متد را مستقیما در پیاده سازی های این کامپوننت ها بکار ببرید.

سایر کلاس های framework اندروید نیز به نمونه ای از کلاس Resources احتیاج دارند. برای مثال، کد زیر برای شما نمایش می دهد چگونه یک فایل Bitmap از ID اشاره گر ایجاد نمایید.

```
// BitmapFactory requires an instance of the Resource class
BitmapFactory.decodeResource(getResources(), R.drawable.ic_action_search);
```

1-9-2-دسترسی به view ها از layout در کلاس activity

طبیعتا برای تنظیم ظاهر view ها، شما لازم دارید که در کد activity و fragment به آن ها دسترسی داشته و property های آن ها را ویرایش نمایید.

جهت دسترسی به view مورد نظر از layout جاری، کافی است متد findViewById(id) را فراخوانی نمایید. پارامتر ورودی id در واقع اشاره به خصیصه ی ID از view مورد نظر در فایل layout دارد. کد زیر کاربرد این متد را به نمایش می گذارد.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    TextView textView = (TextView) findViewById(R.id.mytext);
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    // TODO do something with the TextView
}
```

همچنین این امکان وجود دارد که به وسیله ی متد findViewById(id)، در یک view hierarchy (view های تودرتو در فایل layout) به آسانی view مد نظر را پیدا کرده و به آن دسترسی داشت.

همچنین می توانید یک متد کمکی (utility method) تعریف کنید و به وسیله ی آن آسان تر به view دلخواه دسترسی داشته باشید.

با این روش شما می توانید بدون طی کردن پروسه ی تبدیل (cast) صریح، آسان تر و سریع تر به view مورد نظر دسترسی داشته باشید.

```
package com.example.android.test;
import android.app.Activity;
import android.view.View;
public class UiUtils {
  public static <T extends View> T findView(View root, int id)     {
    return (T) root.findViewById(id); }
  public static <T extends View> T findView(Activity activity, int id)     {
    return (T) activity.getWindow().getDecorView().getRootView().findViewById(id); }
}
// search in the layout of the activity
LinearLayout linearLayout = (LinearLayout) findViewById(R.id.mylayout);
// afterwards search in linearLayout for another view
TextView textView = (TextView) linearLayout.findViewById(R.id.mytext);
// note, you could have directly searched for R.id.mytext, the above coding
// is just for demonstration purposes
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
package com.example.android.test;
import android.app.Activity;
import android.view.View;
public class UiUtils {
  public static <T extends View> T findView(View root, int id)     {
    return (T) root.findViewById(id); }
  public static <T extends View> T findView(Activity activity, int id)     {
    return (T) activity.getWindow().getDecorView().getRootView().findViewById(id); }
}
```

```
Button button = UiUtils.findView(this, R.id.button);
```

3-9-1-دسترسی به محتوای مورد نظر در فایل های XML از دیگر فایل های resource

در فایل های XML خود، برای مثال فایل های layout اپلیکیشن، می توانید به سایر resource ها اشاره کرده و در واقع از داخل فایل مورد نظر به آن ها دسترسی داشته باشید. این کار از طریق درج علامت @ صورت می پذیرد.

برای مثال، جهت دسترسی به مقدار color که در یک فایل XML (resource file) تعریف شده، کافی است این ساختار نگارشی را بکار ببرید: @color/your_id. همچنین ممکن است قبلا یک String با ID یا شناسه ی "titlepage" در فایل XML تعریف کرده باشید و اکنون می خواهید به آن دسترسی داشته باشید. می توانید با دستور @string/titlepage به راحتی به آن متغیر دسترسی داشته باشید.

به منظور دسترسی به محتوایی که خود سیستم اندروید ارائه می دهد (system resource)، لازم است namespace یا پوشه ی android را در اشاره گرها (id هایی که برای دسترسی به آن منبع مورد استفاده قرار می دهید) لحاظ نمایید (مثال:android.R.string.cancel).

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

4-9-1-استفاده از پوشه ی assets و دسترسی به داده های ذخیره شده در آن

در سیستم اندروید، پوشه ای به نام res تعریف شده که دربردارنده ی مقادیر ساخت یافته و سازماندهی شده (با semantics و معانی از پیش تعریف شده) ویژه ی محیط کاری اندروید است. هر نوع داده ای را می توان در این پوشه ذخیره کرد. شما می توانید به فایل های ذخیره شده در پوشه، بر اساس محل قرارگیری آن فایل دسترسی داشته باشید. داخل این پوشه ی assets همچنین می توانید زیرپوشه داشته باشید. برای نگهداری داده هایی که با ساختار مشخصی سازمان دهی و ذخیره نشده اند (unstructured data)، می توانید از پوشه ی /res/raw استفاده نمایید. هر چند بهتر است این نوع اطلاعات را نیز در همان پوشه ی assets ذخیره نمایید (منابعی که در پوشه ی res ذخیره می شود برای کاربر قابل دسترسی هستند).

می توانید با فراخوانی متد ()getAssets که نمونه ای از کلاس AssetManager را برمی گرداند، به فایل های خام ذخیره شده در پوشه ی /res/raw دسترسی داشته باشید. این متد از کلاس انتزاعی Context برگرفته شده است.

در زیر یک آبجکت از کلاس AssetsManager ایجاد می کنید که این کلاس اجازه ی دسترسی به فایل های خام مستقر در پوشه ی assets را می دهد. اسم این آبجکت را manager انتخاب می کنید. سپس با فراخوانی متد ()getAssets نمونه ای از کلاس AssetsManager را برمی گردانید. از کلاس InputStream یک آبجکت به نام open ایجاد می کنید. سپس متد ()open را بر روی آبجکت ایجاد شده از کلاس InputStream فراخوانی می کنید و اسم فایل مورد نظر را به عنوان پارامتر ورودی به متد ()open پاس می دهید. این متد فایل را می خواند و در آبجکت open از کلاس InputStream می ریزد.

```
// get the AssetManager
AssetManager manager = getAssets();
// read the "logo.png" bitmap from the assets folder
InputStream open = null;
try {
    open = manager.open("logo.png");
    Bitmap bitmap = BitmapFactory.decodeStream(open);
    // assign the bitmap to an ImageView in this layout
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
        ImageView view = (ImageView) findViewById(R.id.imageView1);
        view.setImageBitmap(bitmap);
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (open != null) {
                try {
                    open.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
}
```

تمرین: استفاده از محتوا/resource در فایل های XML و در کد

5-9-1-افزودن عکس به اپلیکیشن

در این تمرین نیز اپلیکیشن قبلی را ادامه می دهید. دو تصویر جدید ایجاد می کنید و آن را به پروژه ای به نام ic_tethering و ic_offline اضافه می نمایید.

6-9-1-افزودن view جدید به پروژه

فایل layout خود را باز نموده و یک تگ Button و ImageView به آن اضافه نمایید. سپس فایل ic_offline را از طریق فایل layout به ImageView اضافه کنید.

```
<!--
    NOTE: More attributes are required
    for the correct layout of the ImageView. These are left
    out for brevity
  -->
<ImageView
 android:id="@+id/myicon"
 .... more attributes
 android:src="@drawable/ic_offline" />
```

7-9-1-جایگزین کردن عکس ها با کلیک بر روی دکمه

زمانی که بر روی دکمه کلیک می شود، با استفاده از متد findViewById() به ImageView دسترسی پیدا می کنید. سپس با استفاده از متد SetImageResource() فایل png را (که در زمان اجرای برنامه به صورت یک آبجکت Drawable نمایش داده می شود) به ImageView خود اضافه نمایید. پارامتر ارسال شده به متد SetImageResource() یک شناسه یا ID است که به فایل R.drawable.your_png_file اشاره دارد و شناسه ی متناظر فایل مورد نظر می باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

8-9-1-تست اپلیکیشن

برنامه را اجرا نمایید. حال بر روی دکمه کلیک کنید. در پی کلیک بر روی دکمه ی جدید، تصویری که در UI قابل مشاهده می باشد باید با تصویر دیگری جایگزین شود.

تمرین: استفاده از ScrollView در پروژه

در این تمرین یک کلاس ScrollView به پروژه اضافه می کنید که یک نوار پیمایش به اپلیکیشن اضافه نموده و امکان مشاهده ی تمامی بخش های view را در صفحه ی نمایشگر برای کاربر فراهم می آورد. یک پروژه اندروید به نام de.vogella.android.scrollview ایجاد کرده و سپس کلاس activity با نام ScrollViewActivity را به آن اضافه نمایید.

activity_main.xml را به عنوان فایلی که اپلیکیشن ظاهر برنامه را بر اساس آن طراحی می کند (layout)، انتخاب نمایید.

فایل layout مزبور را که در activity خود فراخوانی می کنید، به صورت زیر ویرایش نمایید:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:fillViewport="true"
  android:orientation="vertical" >
  <LinearLayout
    android:id="@+id/LinearLayout01"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical" >
    <TextView
      android:id="@+id/TextView01"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:paddingLeft="8dip"
      android:paddingRight="8dip"
      android:paddingTop="8dip"
      android:text="This is a header"
      android:textAppearance="?android:attr/textAppearanceLarge" >
    </TextView>
    <TextView
      android:id="@+id/TextView02"
      android:layout_width="wrap_content"
```

```
        android:layout_height="match_parent"
        android:layout_weight="1.0"
        android:text="@+id/TextView02" >
    </TextView>
    <LinearLayout
        android:id="@+id/LinearLayout02"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" >
        <Button
            android:id="@+id/Button01"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1.0"
            android:text="Submit" >
        </Button>
        <Button
            android:id="@+id/Button02"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1.0"
            android:text="Cancel" >
        </Button>
    </LinearLayout>
  </LinearLayout>
</ScrollView>
```

کد کلاس ScrollViewActivity را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.scrollview;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
public class ScrollViewActivity extends Activity {
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    TextView view = (TextView) findViewById(R.id.TextView02);
    String s="";
    for (int i=0; i < 500; i++) {
        s += "vogella.com ";
    }
    view.setText(s);
  }
}
```

اپلیکیشن خود را اجرا کنید. با توجه به کدی که برای برنامه نوشته اید، بایستی بتوانید با استفاده از نوار پیمایش به پایین صفحه دسترسی داشته و دکمه ها را مشاهده نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

9-9-1-تست اپلیکیشن

اکنون می بایست با کلیک بر روی دکمه ی جدید، تصویر قابل مشاهده در نمایشگر را جایگزین نمایید.

تمرین: نوشتن برنامه ی تبدیل دما

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در طی این تمرین شما نحوه ی ایجاد و استفاده از resource و محتوای اندروید را خواهید آموخت.

این اپلیکیشن از قبل نوشته شده و به صورت آماده در آدرس http://play.google.com/store/apps/details?id=de.vogella.android.temperature از طریق Google Play قابل دسترسی می باشد.

می توانید با گوشی اندروید خود بارکد زیر را اسکن کرده، سپس برنامه ی تبدیل دما را از طریق اپلیکیشن Google Play دانلود و نصب نمایید.

10-9-1-ساخت پروژه

یک پروژه ی جدید اندروید بر اساس پارامتر های ارائه شده در جدول زیر ایجاد نمایید.

Table 7. New Android project

Property	Value
Application Name	Temperature Converter
Package name	com.vogella.android.temperatureconverter
API (Minimum, Target, Compile with)	Latest

Table 7. New Android project

Property	Value
Template	Empty Activity
Activity	MainActivity
Layout	activity_main

11-9-1-ایجاد attribute ها

جهت ویرایش فایل res/values/strings.xml آن را انتخاب و سپس بر روی آن دابل کلیک نمایید.

دو متغیر Color و String بر اساس جدول زیر اضافه نمایید.

Table 8. New attributes to add

Type	Name	Value
Color	myColor	#F5F5F5
String	celsius	to Celsius
String	fahrenheit	to Fahrenheit
String	calc	Calculate

مقادیر می بایست به صورت زیر در فایل XML درج شود.

```
<?xml version="1.0" encoding="utf-8"?>
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
<resources>
   <string name="app_name">Temperature Converter</string>
   <string name="action_settings">Settings</string>
   <string name="hello_world">Hello world!</string>
   <color name="myColor">#F5F5F5</color>
   <string name="102ahrenh">to Celsius</string>
   <string name="102ahrenheit">to Fahrenheit</string>
   <string name="calc">Calculate</string>
</resources>
```

12-9-1-ایجاد فایل Layout و تنظیم کننده ی ظاهر اپلیکیشن

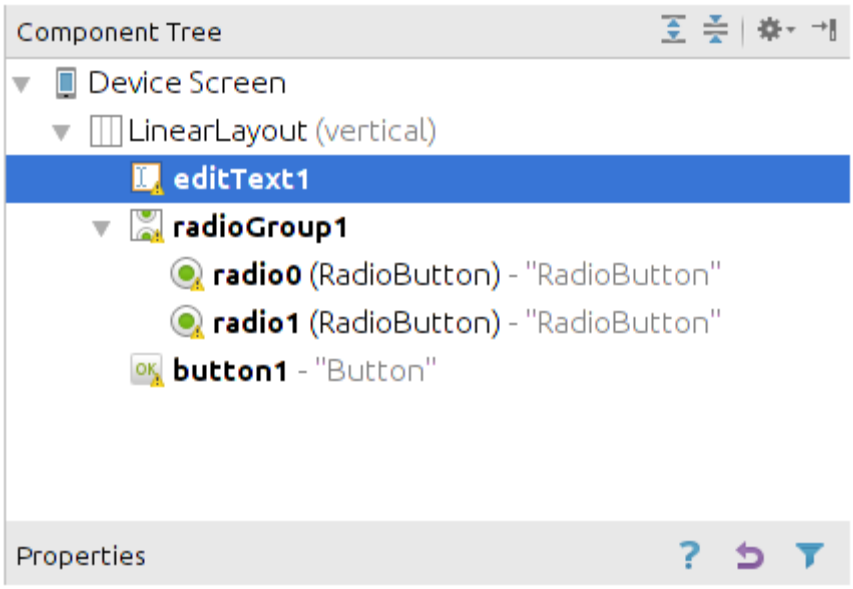
فایل res/layout/activity_main.xml را انتخاب نموده و با دوبار کلیک بر روی آن ویرایشگر متناظر را باز نمایید.

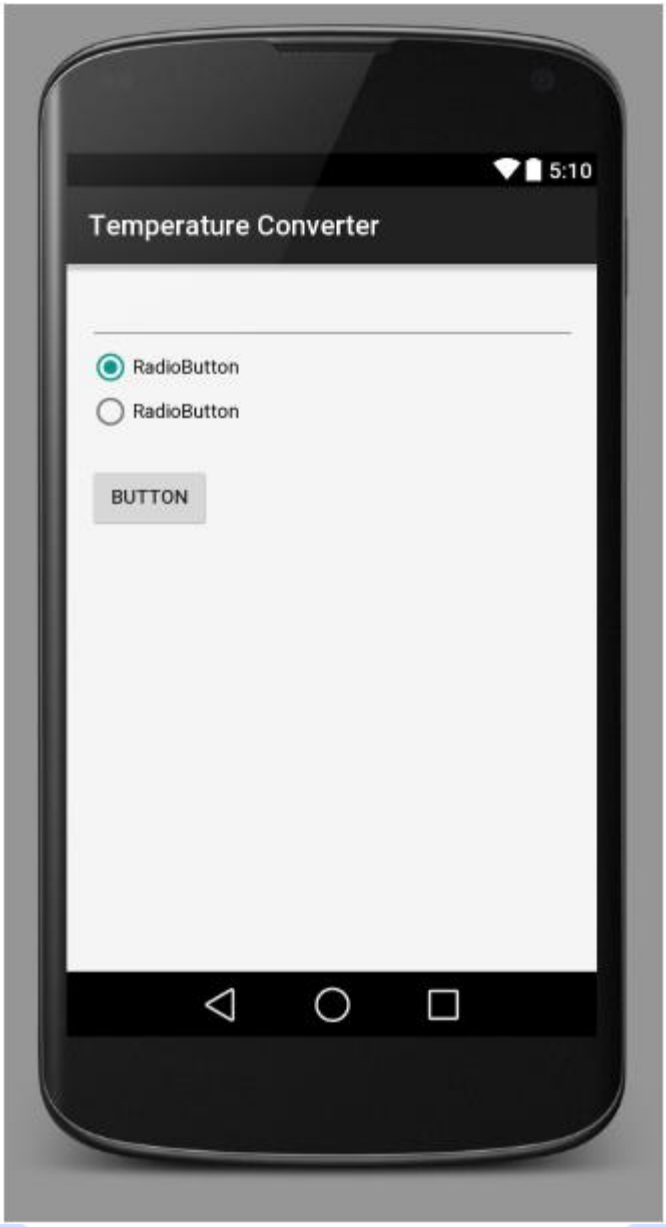
تمامی view ها را از layout حذف نمایید. این کار را می توانید یا به طور مستقیم از طریق فایل XML و یا به وسیله ی ویرایشگر گرافیکی انجام دهید.

سپس یک LinearLayout اضافه نموده و بعد از آن یک LinearLayout دیگر با ViewText و EditText به عنوان فرزند در داخل آن تعریف کنید.

پس از آن یک RadioGroup به همراه دو radio button به layout خود اضافه نمایید. این کار را می توانید یا از طریق فایل XML و یا ویرایشگر گرافیکی انجام دهید. یک روش ساده برای سازمان دهی کامپوننت ها این است که آن ها را با اشاره گر موس کشیده و بر روی Component Tree view جایگذاری نمایید.

خروجی می بایست به صورت زیر باشد.





اکنون بر روی تب XML فایل layout خود کلیک نمایید. آن را با کد زیر تطبیق داده و از مشابه بودن آن اطمینان حاصل نمایید.

توجه: تیم توسعه دهندگان ابزار اندروید این کد را هر چند وقت یکبار تغییر می دهند، به همین خاطر اگر فایل XML شما کمی متفاوت بود، جای تعجب نیست.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity"
    android:background="@color/myColor">
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/editText1" />
    <RadioGroup
        android:id="@+id/radioGroup1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignStart="@+id/editText1"
        android:layout_below="@+id/editText1">
        <RadioButton
            android:id="@+id/radio0"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:checked="true"
            android:text="RadioButton" />
        <RadioButton
            android:id="@+id/radio1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="RadioButton" />
    </RadioGroup>
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignStart="@+id/radioGroup1"
        android:layout_below="@+id/radioGroup1"
        android:layout_marginTop="22dp"
        android:text="Button" />
</LinearLayout>
```

توجه: با تعدادی پیغام هشدار مواجه می شوید. این خطاها را در بخش بعدی تمرین برطرف خواهید نمود.

You see some warning messages. You fix these in the following section of this exercise.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

13-9-1-ویرایش مقادیر property های view

نسخه ی XML فایل را باز نمایید و مقدار @string/Celsius را به property یا خصوصیت android:text از اولین radio button تخصیص دهید. سپس رشته ی farenhiet را به خصوصیت text از دومین radio button انتساب دهید.

```
<RadioGroup
    android:id="@+id/radioGroup1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editText1"
    android:layout_below="@+id/editText1" >

    <RadioButton
        android:id="@+id/radio0"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:checked="true"
        android:text="@string/celsius" />

    <RadioButton
        android:id="@+id/radio1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/fahrenheit" />
</RadioGroup>
```

مقدار خصوصیت Checked را بر روی true تنظیم نمایید و سپس مقدار onClick را به خوصیصت onClick اختصاص دهید.

مقدار @string/calc را به خصوصیت text اولین دکمه اختصاص داده، سپس مقدار onClick را به خصوصیت onClick انتساب دهید.

خوصیصت inputType را در EditText با numberSigned و numberDecimal مقداردهی نمایید. این اصلاحات در فایل XML زیر قابل مشاهده می باشد. در پایان ID آن را به "InputValue" تغییر دهید.

```
<EditText
 android:id="@+id/inputValue"
 android:layout_width="match_parent"
```

```
android:layout_height="wrap_content"
android:layout_alignParentEnd="true"
android:layout_below="@+id/textView"
android:ems="10"
android:inputType="numberSigned|numberDecimal" />
```

تمامی کامپوننت های UI در یک layout با فرمت XML تعریف می شوند. رنگ پس زمینه (background color) را نیز در این فایل اضافه نمایید.

Color را انتخاب نموده و سپس myColor را در کادر محاوره ای انتخاب نمایید. رنگ پس زمینه ی UI در فایل XML به صورت زیر در آخرین خط تنظیم می شود.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   tools:context=".MainActivity"
   android:background="@color/myColor">
```

پس از افزودن این خط کد، پس زمینه بایستی با رنگ whitesmoke تنظیم شود.هر چند تشخیص این تغییر ممکن است با توجه به رنگ انتخابی کمی دشوار باشد.

حال بر روی تب activity_main.xml کلیک نموده و با بررسی آن از صحیح بودن کامل کد اطمینان حاصل نمایید.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   tools:context=".MainActivity"
   android:background="@color/myColor">
   <EditText
      android:layout_width="match_parent"
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
        android:layout_height="wrap_content"
        android:id="@+id/inputValue"
        android:inputType="numberSigned|numberDecimal"/>
    <RadioGroup
        android:id="@+id/radioGroup1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignStart="@+id/editText1"
        android:layout_below="@+id/editText1">
        <RadioButton
            android:id="@+id/radio0"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:checked="true"
            android:text="@string/celsius" />
        <RadioButton
            android:id="@+id/radio1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/fahrenheit" />
    </RadioGroup>
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignStart="@+id/radioGroup1"
        android:layout_below="@+id/radioGroup1"
        android:layout_marginTop="22dp"
        android:text="@string/calc"
        android:onClick="onClick"/>
</LinearLayout>
```

14-9-1-تعریف یک کلاس کمکی (utility class)

یک کلاس تعریف کرده و در آن کد لازم برای انجام عملیات تبدیل از واحد سلسیوس به فارانهایت و بالعکس را بنویسید.

```
package com.vogella.android.temperatureconverter;
public class ConverterUtil {
    // converts to celsius
    public static float convertFahrenheitToCelsius(float fahrenheit) {
        return ((fahrenheit - 32) * 5 / 9);
    }
    // converts to fahrenheit
    public static float convertCelsiusToFahrenheit(float celsius) {
        return ((celsius * 9) / 5) + 32;
    }
}
```

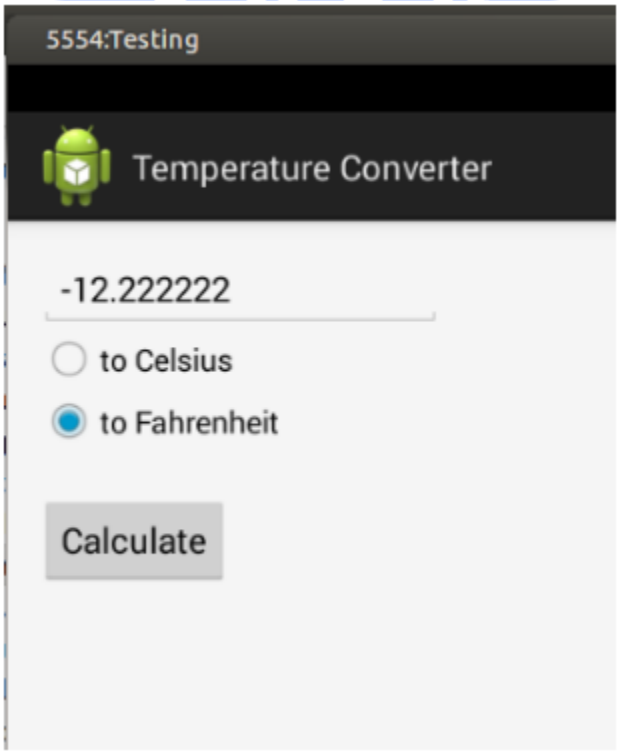
15-9-1-ویرایش کد activity

کد موجود در کلاس MainActivity را به صورت زیر ویرایش کنید:

```
package com.vogella.android.temperatureconverter;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.Toast;
public class MainActivity extends Activity {
  private EditText text;
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    text = (EditText) findViewById(R.id.inputValue);
  }
  // this method is called at button click because we assigned the name to the
  // "OnClick" property of the button
  public void onClick(View view) {
    switch (view.getId()) {
    case R.id.button1:
      RadioButton celsiusButton = (RadioButton) findViewById(R.id.radio0);
      RadioButton fahrenheitButton = (RadioButton) findViewById(R.id.radio1);
      if (text.getText().length() == 0) {
        Toast.makeText(this, "Please enter a valid number",
            Toast.LENGTH_LONG).show();
        return;
      }
      float inputValue = Float.parseFloat(text.getText().toString());
      if (celsiusButton.isChecked()) {
        text.setText(String    .valueOf(ConverterUtil.convertFahrenheitToCelsius(inputValue)));
        celsiusButton.setChecked(false);
        fahrenheitButton.setChecked(true);
      } else {
        text.setText(String      .valueOf(ConverterUtil.convertCelsiusToFahrenheit(inputValue)));
        fahrenheitButton.setChecked(false);
        celsiusButton.setChecked(true);
      }
      break;
    }
  }
}
```

16-9-1-راه اندازی اپلیکیشن

اپلیکیشن اندروید خود را راه اندازی نموده و یک عدد جهت تبدیل در آن وارد نمایید. سپس تبدیل را انتخاب نموده و دکمه ی مربوطه را کلیک نمایید. پس از مشاهده ی نتیجه گزینه ی اول، جهت اطمینان حاصل نمودن از عملکرد صحیح برنامه در تبدیل بالعکس، گزینه ی دیگر را انتخاب نمایید.



10-1-فرایند آماده سازی، عرضه، توزیع و نصب اپلیکیشن (Deployment)

1-10-1-نحوه ی آماده سازی و عرضه ی اپلیکیشن برای نصب

روش های مختلفی برای نصب اپلیکیشن بر روی دستگاه وجود دارد. شما می توانید اپلیکیشن را از طریق USB و در حالت تست بر روی دستگاه مستقر نمایید یا آن را از طریق ایمیل به خود ارسال نموده و نصب کنید و یا اپلیکیشن را بر روی فروشگاه های مجازی عرضه ی اپلیکیشن های تحت موبایل اندروید بارگذاری نموده و از آنجا امکان دانلود و نصب آن را فراهم نمایید. در زیر روش های معمول و پرطرفدار عرضه و نصب اپلیکیشن تشریح شده اند.

2-10-1-مشخص کردن اجزا نرم افزاری و سیستم سخت افزاری مورد نیاز برای اپلیکیشن

شما می توانید با تعریف تگ <uses-feature> در فایل تنظیمات اندروید (manifest) مشخص نمایید اپلیکیشن شما برای اجرای موفقیت آمیز بر روی دستگاه به چه ابزاری نیاز دارد. با تنظیم مقدار خصیصه ی android:required (attribute) بر روی true در واقع شما به دستگاه اعلان می کنید که برای اجرای اپلیکیشن بایستی از سخت افزار و نرم افزار مربوطه برخوردار باشد یا چنانچه آن ویژگی در دستگاه تعبیه شده باشد برنامه بهتر کار می کند اما در عین حال به گونه ای طراحی شده که بتواند بدون حضور آن ویژگی نیز با موفقیت اجرا شود. برای مثال، اپلیکیشنی را در نظر بگیرید که برای عملکرد خود به یک حسگر یا دوربین احتیاج دارد. کافی است تگ نام برده را در فایل تنظیمات این برنامه تعریف کرده و مقدار خصیصه ی android:required آن را بر روی true تنظیم نمایید.

3-10-1-Sign کردن (تخصیص امضای الکترونیکی جهت دیباگ) اپلیکیشن قبل از انتشار و عرضه ی آن در فروشگاه های مجازی

اپلیکیشن های اندروید بایستی قبل از اینکه بر روی دستگاه نصب شوند، امضا شده و کلید یا امضای الکترونیکی منحصربفردی به آن اختصاص یابد. به هنگام ساخت و توسعه ی اپلیکیشن، پروسه ی کامپایل و build برنامه ی شما را با یک debug key (امضای الکترونیکی جهت دیباگ) امضا می کند.

جهت نصب اپلیکیشن اندروید، شما می بایست فایل apk اندروید را با یک کلید منحصر بفرد (signature key) که به صورت خودکار تولید می شود، امضا نمایید.

لازم به ذکر است که برای بروز رسانی اپلیکیشن خود، می بایست از همان signature key در فروشگاه مجازی Google Play استفاده نمایید. اگر این کلید منحصربفرد را از دست بدید، دیگر هیچ وقت نمی توانید برنامه ی خود را بروز رسانی کنید. توصیه می شود از این کلید نسخه ی پشتیبان تهیه نمایید.

4-10-1-خروجی امضا شده دادن (Export) اپلیکیشن از طریق محیط کاری Android Studio

جهت خروجی دادن اپلیکیشن از داخل محیط Android Studio منوی اصلی را باز نموده و سپس مسیر روبرو را طی نمایید: Build ▸ Generate Signed APK.

5-10-1-خروجی امضا شده دادن/Export اپلیکیشن از طریق محیط برنامه نویسی Eclipse (IDE)

جهت خروجی دادن اپلیکیشن از محیط کاری Eclipse، کافی است بر روی آن راست کلیک کرده و سپس این مسیر را طی نمایید: Android Tools ▸ Export Signed Application Package.

ویزارد به شما اجازه می دهد یک کلید جدید ایجاد کنید یا همان کلید از پیش آماده را مورد استفاده قرار دهید.

6-10-1-نصب اپلیکیشن از روش های دیگر

اندروید به شما اجازه می دهد تا اپلیکیشن را به طور مستقیم نصب نمایید. کافی است بر روی لینکی که در یک پیوست ایمیل یا صفحه ی وب به فایل apk. اشاره دارد، کلیک کنید. اندروید از شما می پرسد آیا مایلید اپلیکیشن را نصب نمایید یا خیر.

البته برای این منظور می بایست به بخش تنظیمات دستگاه مراجعه نموده و اجازه ی نصب اپلیکیشن هایی که خارج از فروشگاه های مجازی معتبر عرضه می شود را به طور صریح اعطا نمایید. می توانید به گزینه ی مورد نظر در بخش Security تنظیمات دستگاه خود دسترسی داشته باشید.

7-10-1-فروشگاه مجازی Google Play

جهت عرضه ی اپلیکیشن های اندرویدی خود در Google play می بایست مبلغ 25 دلار را برای یکبار پرداخت کنید. پس از پرداخت این مبلغ می توانید برنامه های خود را به همراه آیکون های مربوطه تحت آدرس https://play.google.com/apps/publish مستقیما در Google Play بارگذاری نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

گوگل ابتدا برنامه را اسکن کرده و از عدم وجود بدافزار در آن اطمینان حاصل می کند. معمولا چند دقیقه پس از بارگذاری، اپلیکیشن برای دانلود و نصب در دسترس قرار می گیرد.

1-11-آشنایی با محیط برنامه نویسی Android Studio

Android Studio در حال حاضر محیط برنامه نویسی مورد تایید شرکت Google است. محیط Android Studio خود بر اساس محیط برنامه نویسی IntelliJ شکل گرفته و در اصل همان کد و دستورات نوشته شده برای محیط برنامه نویسی (code base) مزبور است که قابلیت های جدیدی ویژه ی برنامه سازی برای اندروید به آن اضافه شده است (fork) یا به عبارت دیگر Android Studio از IntelliJ منشعب گردیده و قابلیت های جدیدی ویژه ی توسعه ی برنامه برای سیستم عامل اندروید به آن اضافه شده است.

1-11-1-نوار ابزار محیط برنامه نویسی Android Studio

Toolbar و نوار ابزار محیط کاری Android Studio دربردارنده ی تعداد زیادی گزینه ویژه ی ساخت و توسعه ی اپلیکیشن های اندرویدی می باشد. در زیر به هر یک از ابزار موجود در این ToolBar اشاره شده است.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

ابزار های موجود در toolbar محیط برنامه نویسی اندروید استودیو	
Number	Action
1	AVD Manager
2	Sync Project with Gradle Files
3	Project Structure / Settings
4	Android SDK manager

2-11-1-محل ذخیره سازی پروژه های اندروید

Android Studio تمامی پروژه های شما را به صورت پیش فرض در پوشه ی home کاربر تحت نام AndroidStudioProjects ذخیره می کند.

پوشه ی main دربردارنده ی فایل های تنظیمات محیط کاری و فایل های Gradle build می باشد (عملیات پیکربندی پروژه در فایل گریدل صورت می پذیرد. در این فایل شما می توانید متدها و خواصی را تعریف کرده و وظایف معینی را مشخص نمایید). فایل های مربوط به خود اپلیکیشن در پوشه ی app جایگذاری می شود. این پوشه فایل های جاوایی را شامل شده و پیش فرض دربردارنده ی یک فایل به نام MainActivity.java می باشد که برنامه به هنگام راه اندازی از آن شروع می شود. این ساختار در زیر به نمایش گذاشته شده است.

توجه: Android Studio این فایل ها را با نمای خاصی به نمایش می گذارد. برای مثال فایل های حاوی تعاریف رشته ها، رنگ ها و محتویات (resource) اپلیکیشن را تحت گره (node) res سازمان دهی می کند که با ساختار واقعی فایل یکسان نیست.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

3-11-1-باز کردن پروژه ها و راه گزینی (سوییچ) بین آن ها

محیط برنامه نویسی Android Studio هر پروژه را در پنجره ی متفاوتی به نمایش می گذارد. جهت باز کردن پروژه ی جاری کافی است مسیر File ▸ Open... را طی نمایید. به دنبال آن، پنجره ی جدیدی پدیدار شده و پروژه ی مورد نظر را در خود نمایش می دهد.

جهت بستن پروژه نیز کافی است این مسیر را طی نمایید: File ▸ Close Project. سپس یک پروژه ی جدید ایجاد نموده یا پروژه ی از قبل موجود را به روش یاد شده در بالا باز نمایید.

4-11-1-پاک کردن محتوای پوشه های build و کامپایل مجدد پروژه/هماهنگ سازی پروژه با Gradle

متاسفانه ابزار build در محیط کاری Android Studio تغییرات وارد شده به فایل های محتوا (res) را نمی شناسد و خود را با آن تطبیق نمی دهد. اگر می خواهید پروژه را کاملا rebuild نمایید، کافی است مسیر Build ▸ Clean Project را طی کنید.

تغییرات را در فایل gradle build باید به صورت دستی وارد نمایید. جهت هماهنگ سازی پروژه با Gradle، لازم است این مسیر را طی نمایید: Tools ▸ Android ▸ Sync Project with Gradle و یا بر روی لینک مربوطه در tooltip که ویرایشگر فایل Gradle build در اختیار شما قرار می دهد، کلیک نمایید.

5-11-1-بروز رسانی تنظیمات Android Studio

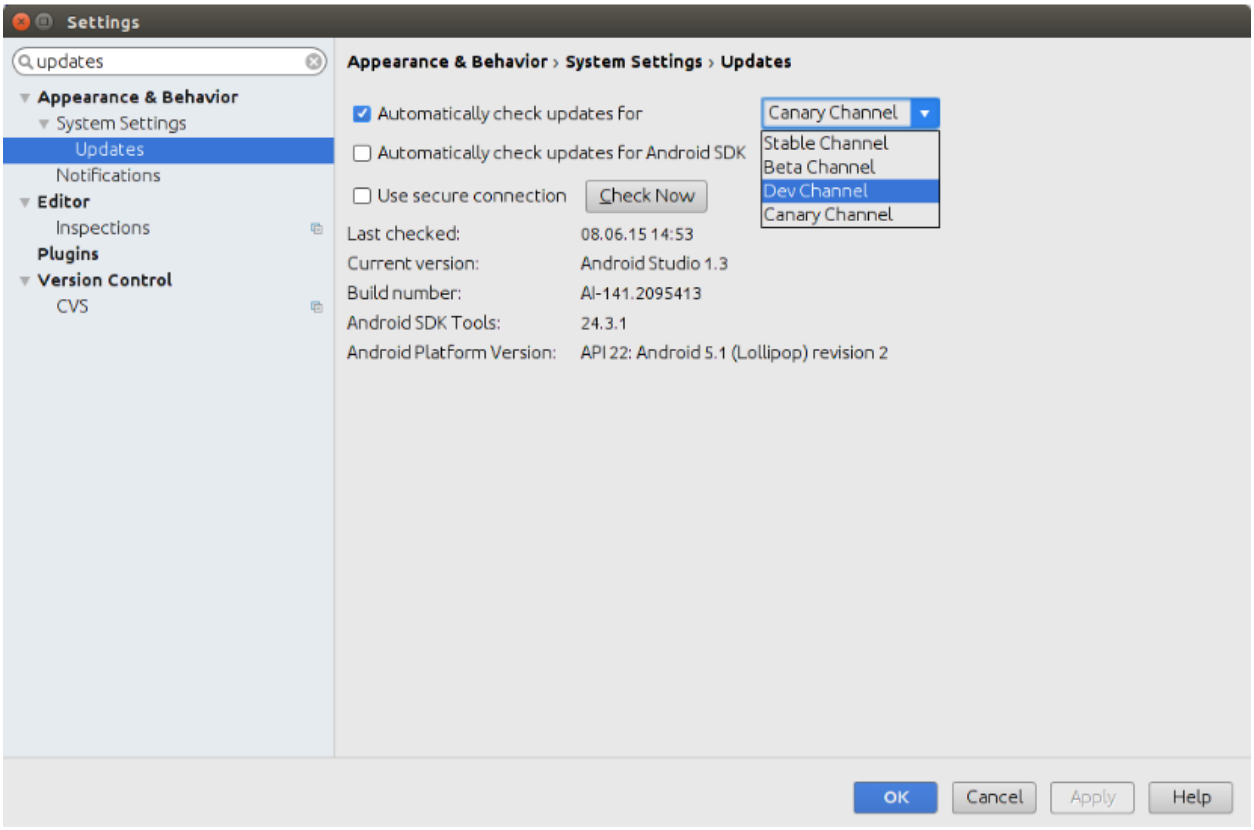
Android Studio در نسخه ها (flavor) مختلف ارائه می شود. برنامه نویس می تواند نسخه ی ثابت و نهایی را مورد استفاده قرار دهد یا ویژگی های نسخه های جدید ولی غیرنهایی و در حال تغییر را تست کند.

1. Canary channel: build های Canary، در واقع نسخه هایی با جدیدترین ویژگی های اضافه شده به محیط کاری هستند و به صورت هفتگی منتشر می شوند. اگر چه این نسخه ها تست شده هستند، اما همچنان احتمال بروز خطا در آن ها وجود دارد چرا که با عجله و با هدف عرضه یا معرفی امکانات جدید انتشار می یابند. از این جهت استفاده از آن برای production (زمان و مرحله ای که کاربر محیط توسعه را بر روی سیستم نصب می کند و در آن به برنامه سازی می پردازد) توصیه نمی شود.
2. Dev channel: build های Dev، نسخه های برتر و دستچین شده از build های قدیمی تر Canary هستند که تست شده و مورد تایید می باشند. این نسخه هر دو هفته یا ماهی یکبار بروز آوری می شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

3. Beta channel: نسخه هایی با سطح کیفیتی beta در این channel عرضه می شوند.
4. Stable channel: جدیدترین نسخه ی ثابت از Android Studio که تمامی خطاهای آن برطرف شده و آماده ی عرضه ی نهایی در اختیار توسعه دهنده و کاربران می باشد.

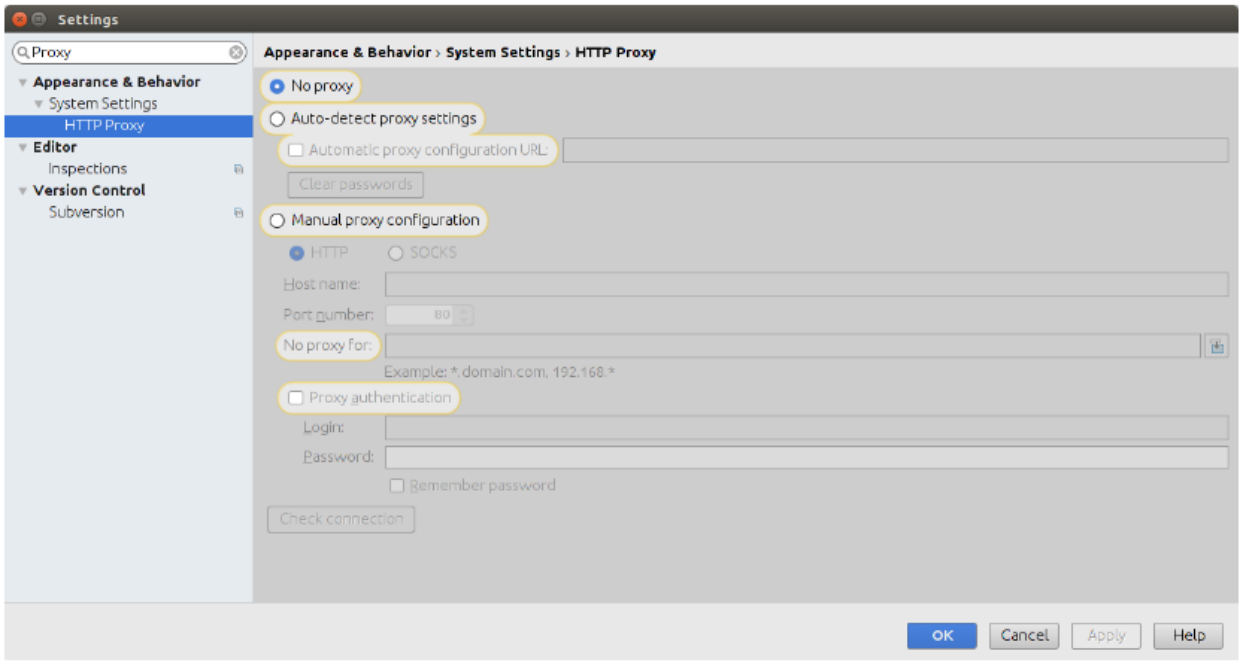


6-11-1-استفاده از Android Studio با proxy

می توانید Android Studio و Gradle را با proxy بکار ببرید. برای این منظور کافی است از مسیر File ▸ Settings... وارد شوید. همچنین تنظیمات Gradle را برای استفاده از این proxy بروز رسانی می کند. از ویرایش 1.4 محیط کاری Android Studio این امکان بهینه شده است.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



7-11-1-ابزار Android Device Monitor

جهت اجرای Android Device Monitor می توانید یکی از دو مسیر رو به رو را طی نمایید:

Tools > Android > Android Device Monitor یا در نوار ابزار محیط کاری Android Studio بر روی Android Device Manager کلیک کنید.

در پی این کار، یک اپلیکیشن مبتنی بر Eclipse به نام Android Device Monitor برای تعامل با دستگاه مجازی اندروید و اپلیکیشن اندرویدی باز می شود.

1-12-شروع به استفاده از Android Studio

المان هایی که در منوهای Android Studio مشاهده می کنید پویا هستند و می توانند بر اساس انتخاب جاری کاربر تغییر کنند. اما متاسفانه این ویژگی در زمان حاضر هنوز کامل نیست و آن طور که باید عمل نمی کند. در صورتی که المان مورد نظر را مشاهده نکردید، سعی کنید دوباره بر روی المان app کلیک نمایید.

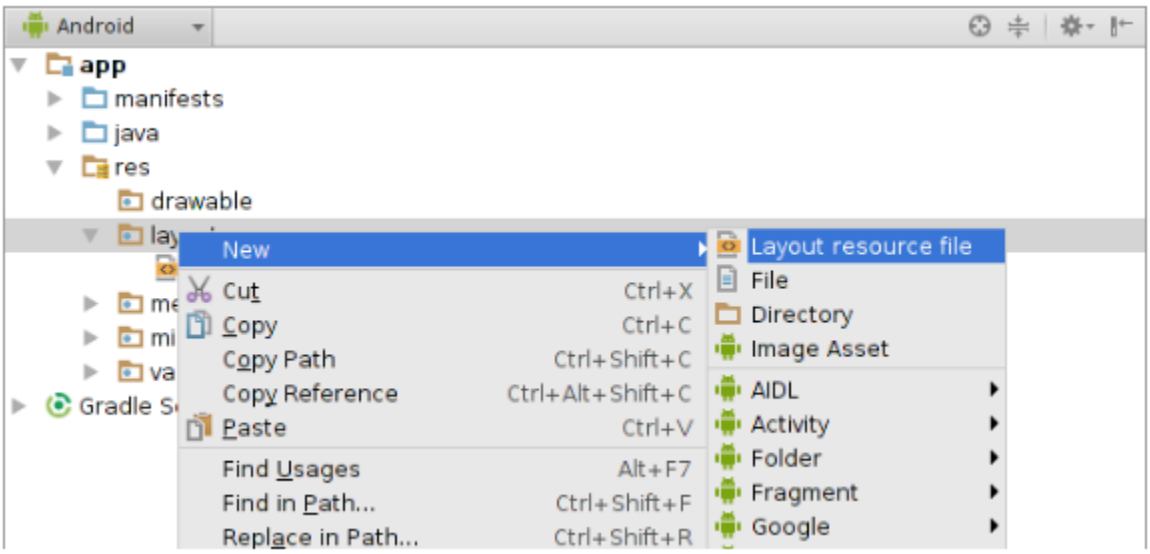
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-12-1-ایجاد یک فایل layout جدید

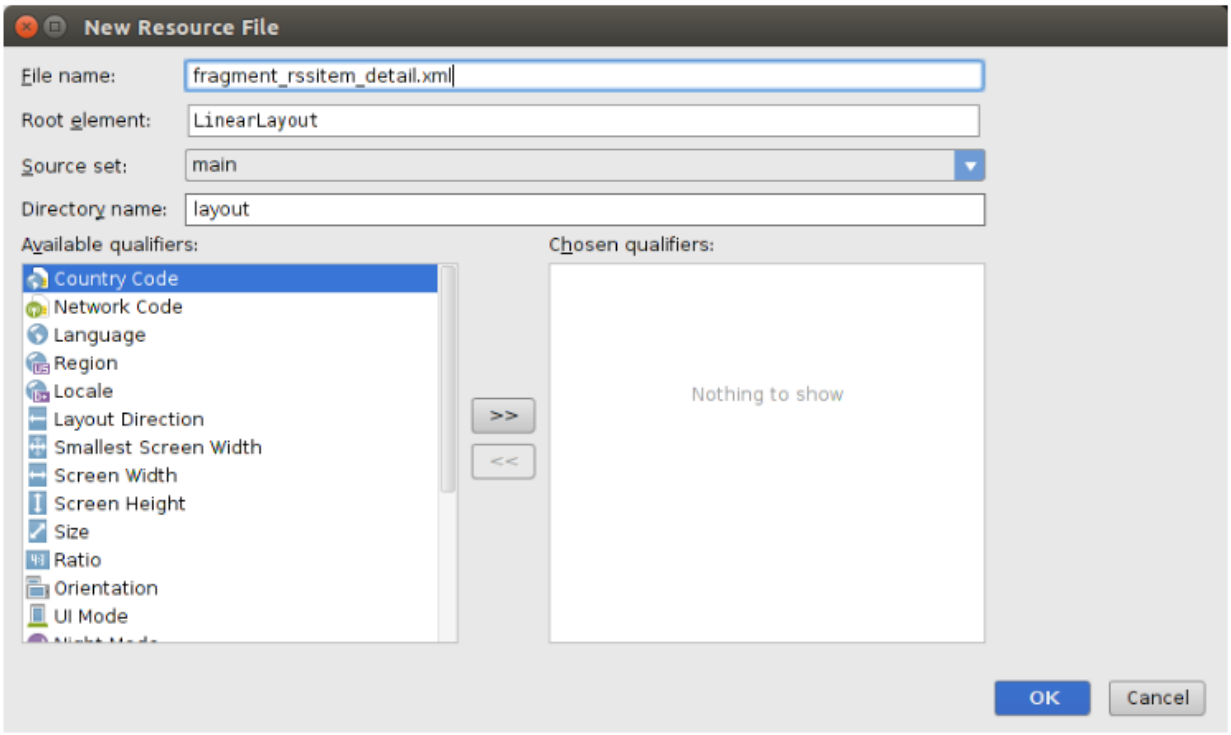
جهت ایجاد یک فایل layout جدید، پروژه را انتخاب نموده، بر روی آن راست کلیک نمایید و سپس این مسیر را طی کنید: File ▸ New ▸ Android resource file . با انتخاب گزینه ی Layout، یک فایل جدید XML برای تعریف المان های رابط کاربری خود (فایل Layout) ایجاد نمایید.

آسان ترین راه برای ایجاد یک فایل حاوی محتوا (resource file) همچون فایل layout که ظاهر برنامه را تعیین می کند، context menu است که با راست کلیک بر روی پوشه ی مربوطه به نمایش در می آید. نحوه ی ایجاد یک فایل layout جدید را در تصویر زیر مشاهده می کنید.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

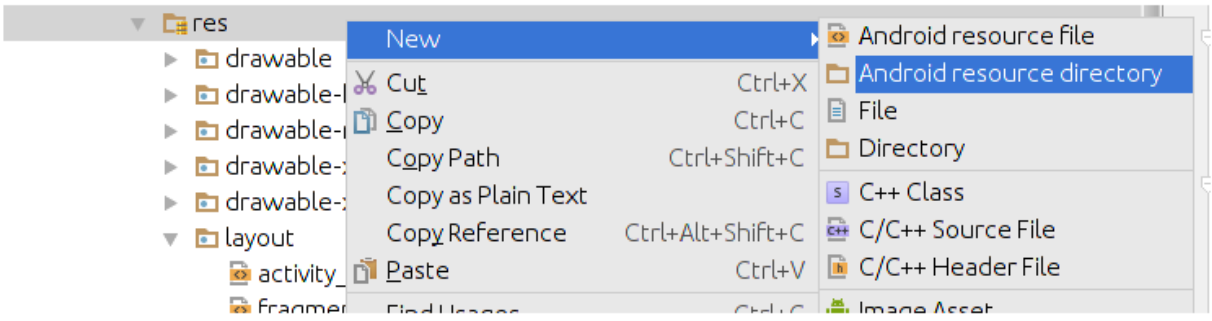
88146323 - 88446780 - 88146330



نکته: اگر می خواهید یک فایل جدید layout برای دستگاه با پیکربندی خاص طراحی کنید، در همین پنجره ی محاوره ای می توانید آن را از بخش Available Qualifier انتخاب نمایید.

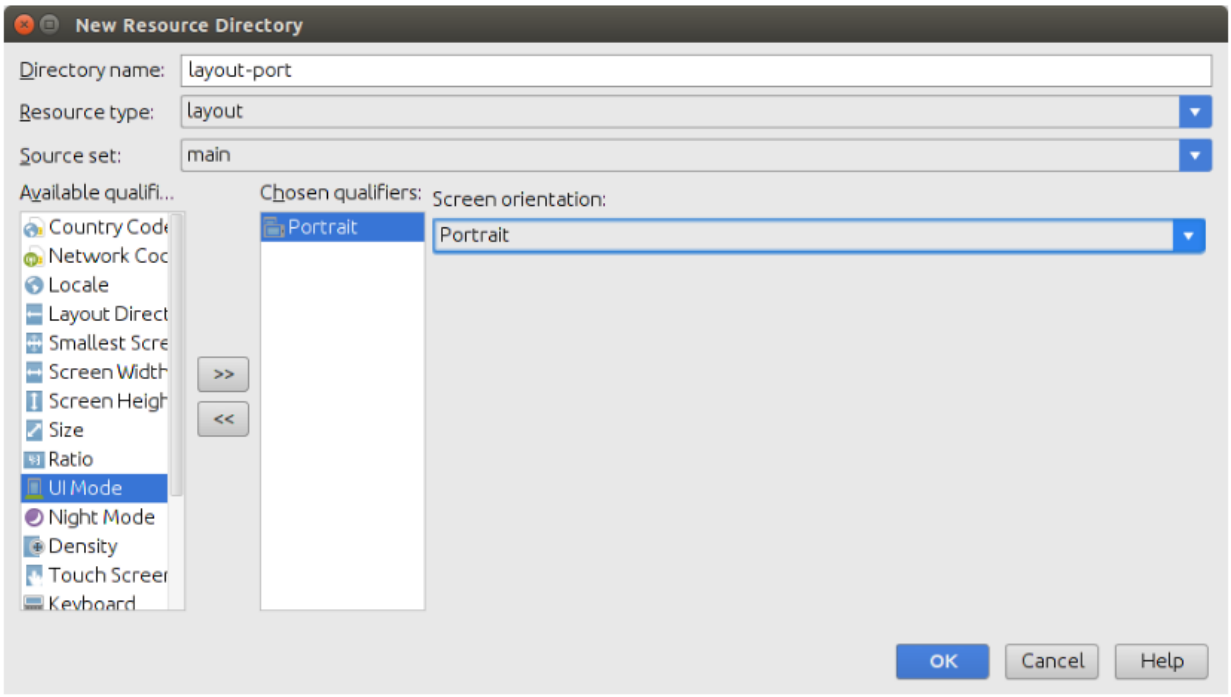
2-12-1-ایجاد یک فایل محتوای جدید (resource file)

در صورتی که فایل محتوای مورد نظر موجود نبود، بایستی آن را ایجاد کنید. در تصویر زیر مشاهده می کنید که فایل محتوای layout-port_folder_ ایجاد می شود.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



3-12-1-کار با فایل های layout

محیط کاری Android Studio یک ویرایشگر ویژوال و متنی برای اعمال تغییر در فایل های layout فراهم می آورد. آسان ترین روش برای افزودن ابزارک های رابط کاربری (UI Widget) به فایل layout، انتخاب آن ها از Palette است. می توانید المان های مورد نظر را با اشاره گر موس کشیده و به راحتی در فایل جایگذاری نمایید. در حالت طراحی ویژوال (visual design mode)، شما می توانید با راست کلیک بر روی view مورد نظر و انتخاب گزینه ی Delete از Context menu، آن view را حذف نمایید.

Android Studio مقادیر جای نگهدار و موقتی (placeholder) را با مقادیر حقیقی جایگزین می کند. این امر سبب می شود مقایسه ی فایل ها کمی دشوار شود. در چنین مواقعی برای پی بردن به مقادیر موقتی یا placeholder، کافی است بر روی مقادیر مورد نظر دوبار کلیک نمایید.

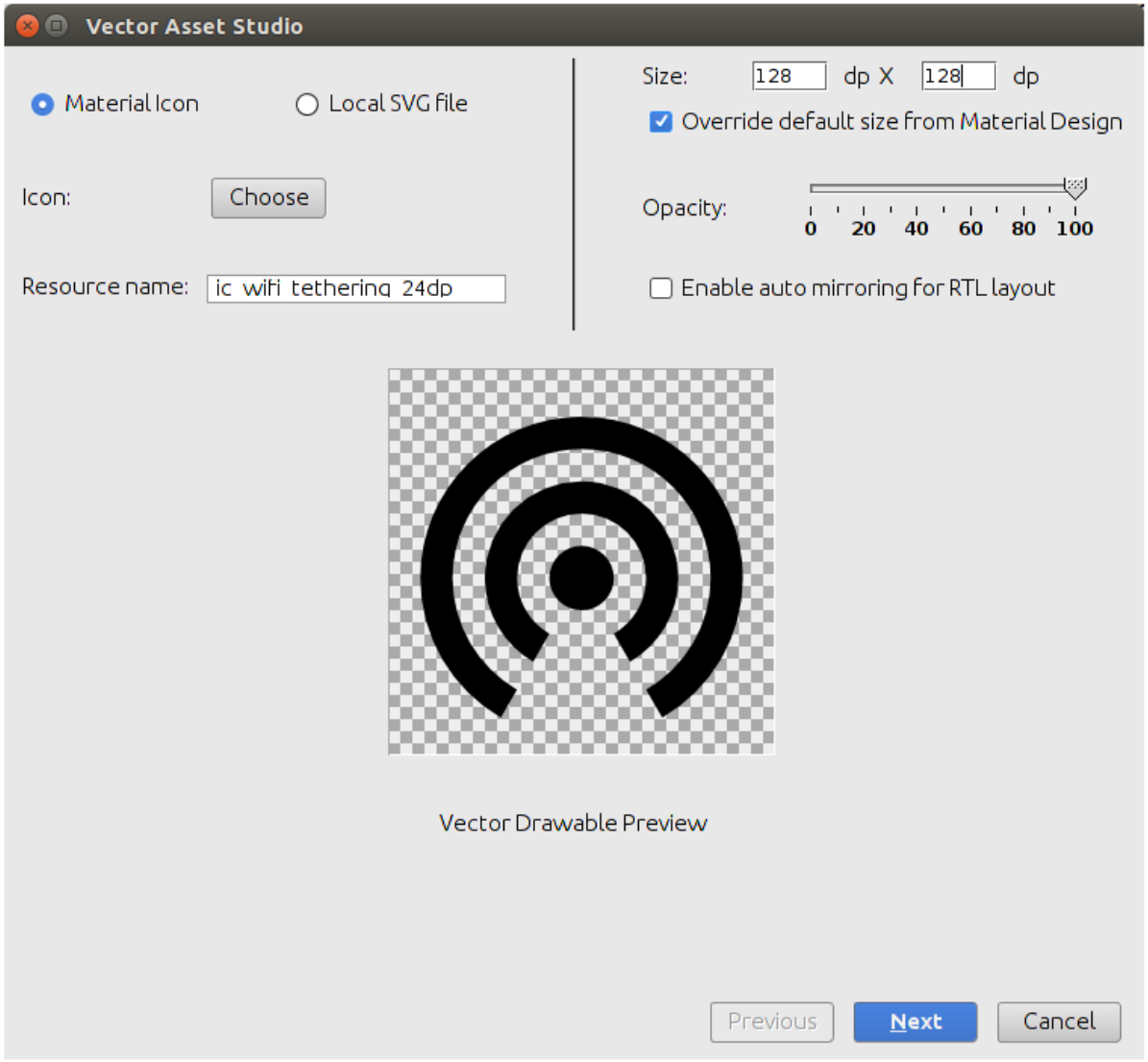
4-12-1-قرار دادن فایل تصویری (image) در پروژه

منوی محیط کاری Android Studio را باز نموده و سپس این مسیر را طی نمایید: File ▸ New ▸ Vector Assert. در پی این کار یک فایل drawable برداری یا vector ایجاد می

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

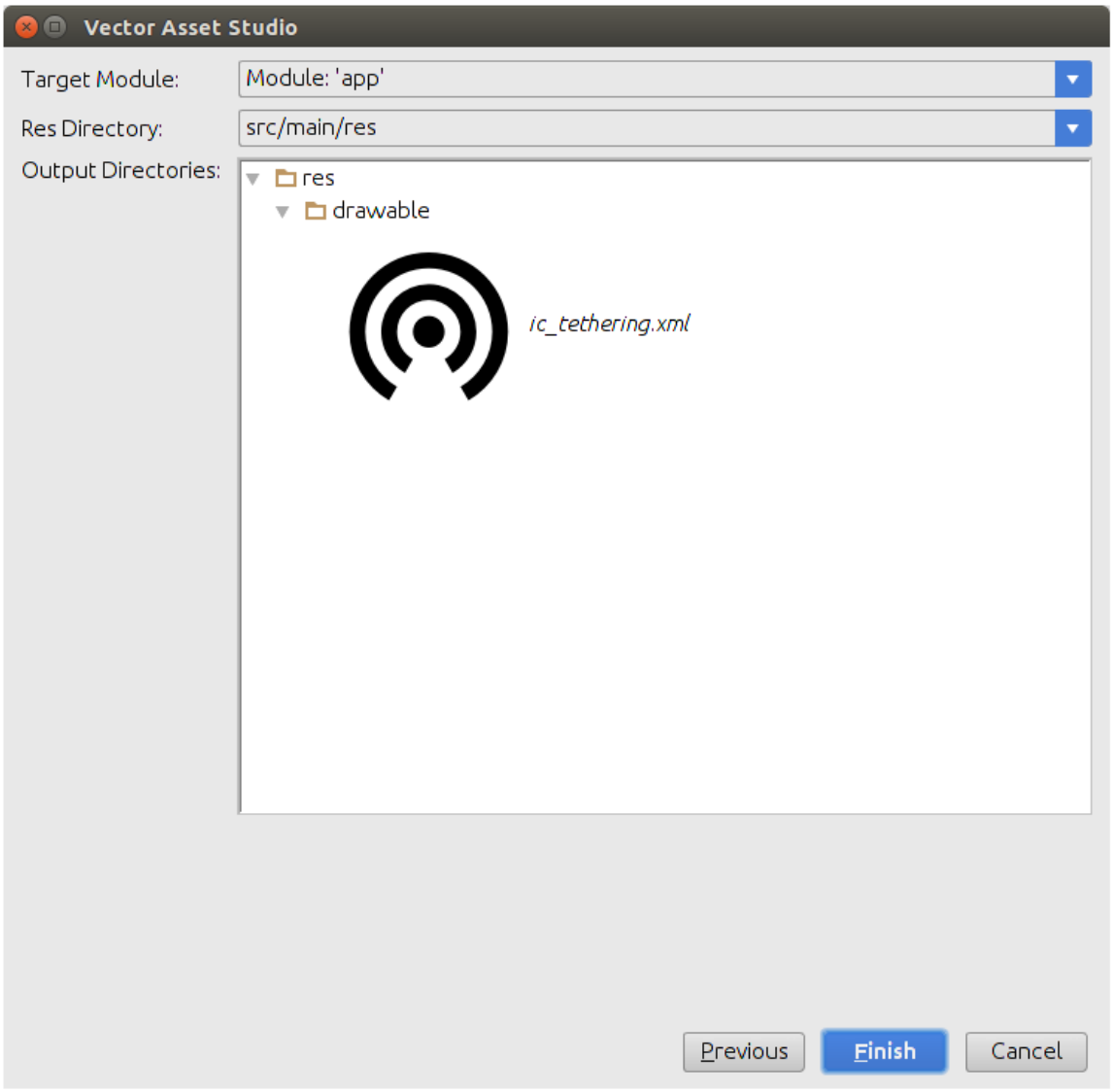
88146323 - 88446780 - 88146330

شود. در زیر پروسه ی ایجاد آیکون ic_tethering در محیط کاری Android Studio به نمایش گذاشته شده است:



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

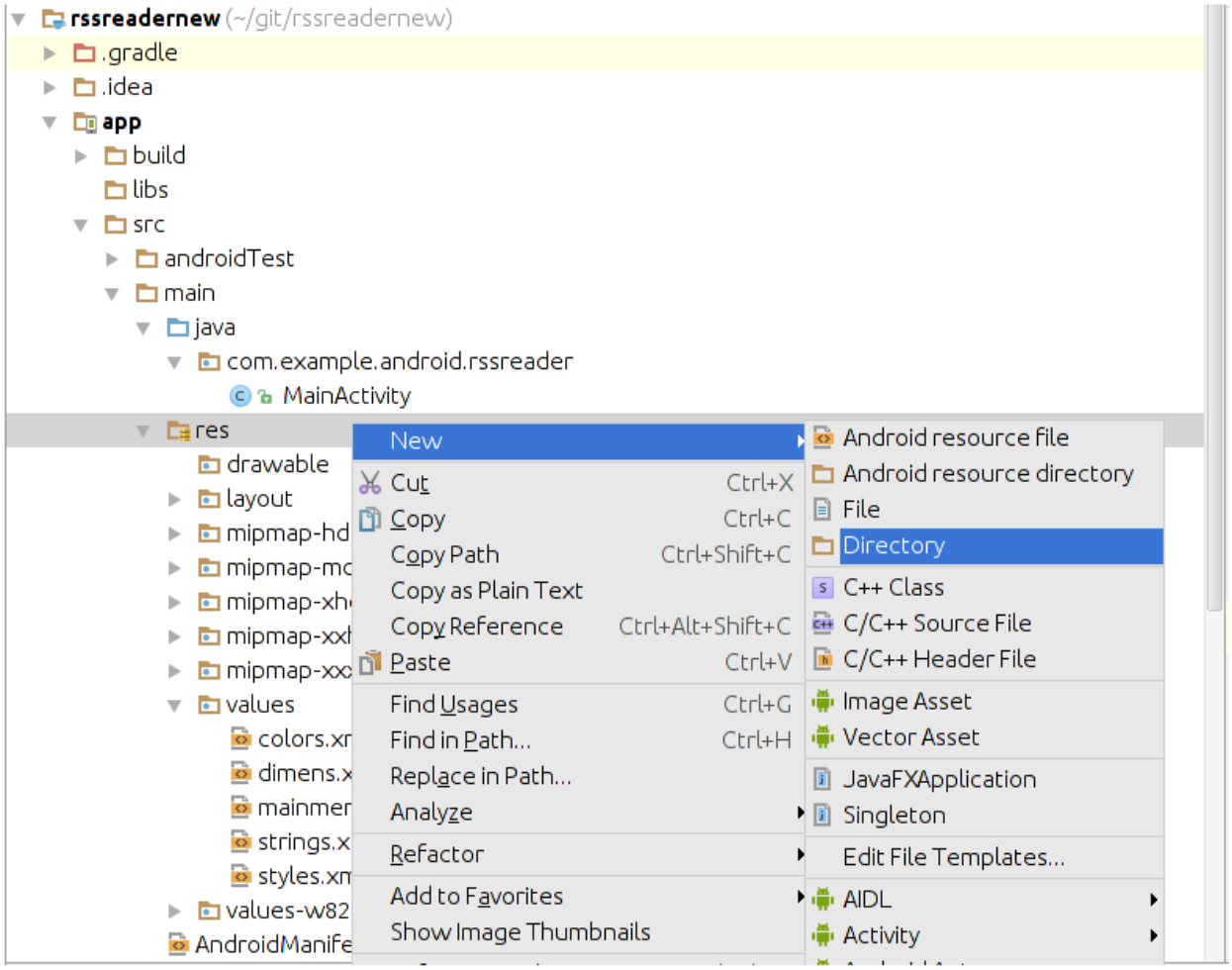


4-12-1-افزودن یک منوی جدید (menu resource)

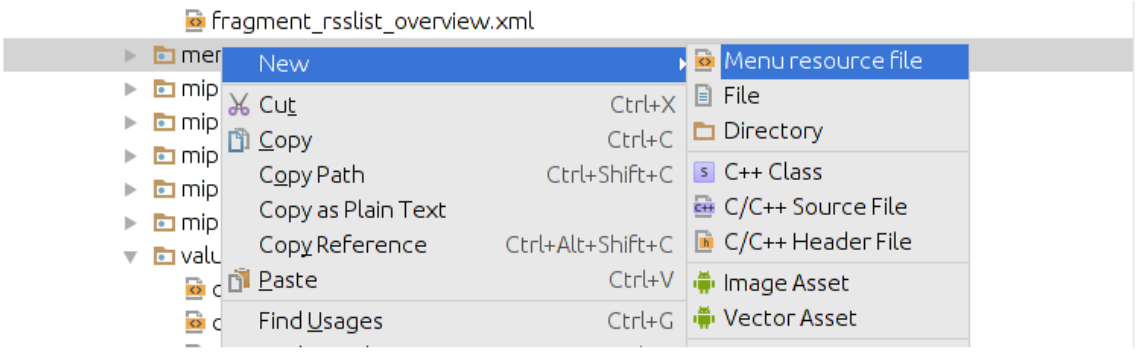
اگر پوشه ای به نام menu تعریف نشده، اکنون آن را ایجاد نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



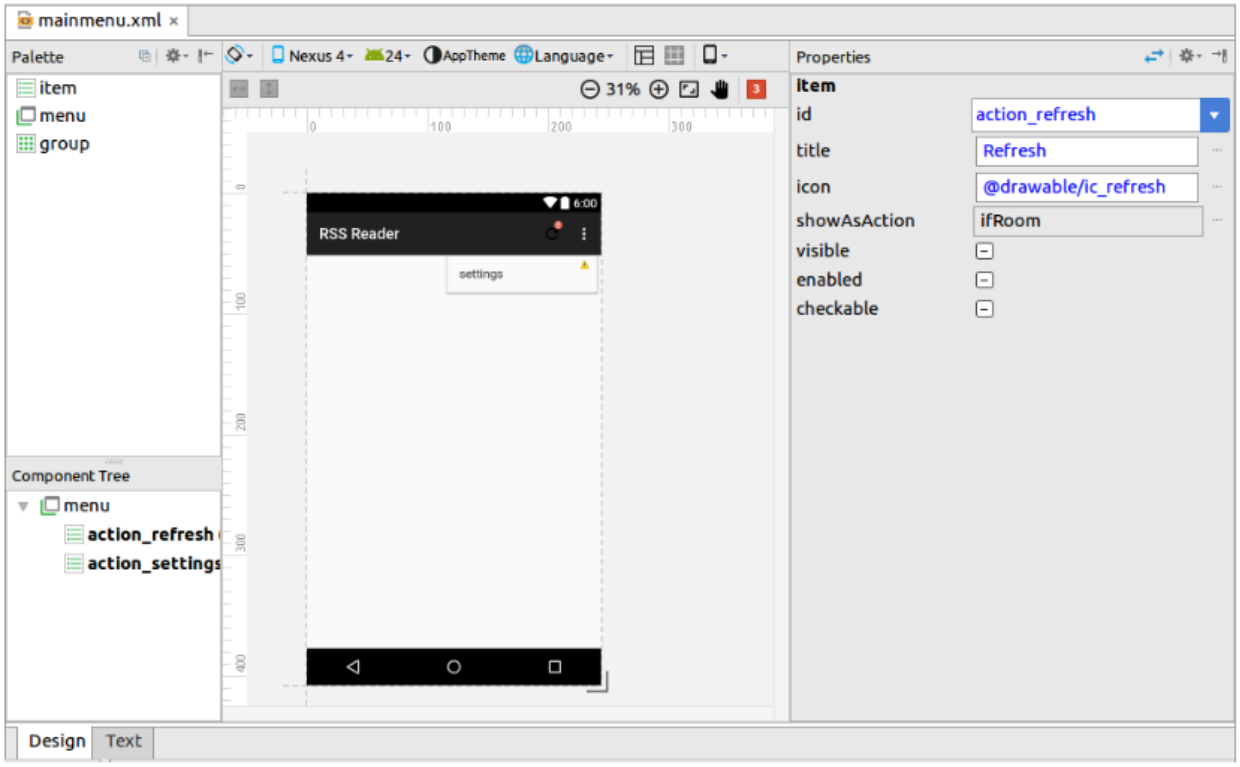
یک فایل XML جدید به نام mainmenu.xml برای این منو تعریف نمایید.



پس از آن می توانید فایل مزبور را از طریق تب Text به صورت دستی ویرایش کنید و یا تب Design را باز نموده و از طریق drag&drop محتوای فایل را ویرایش نمایید.

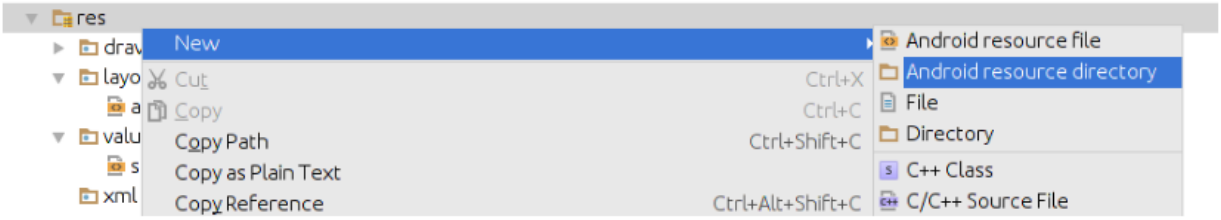
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



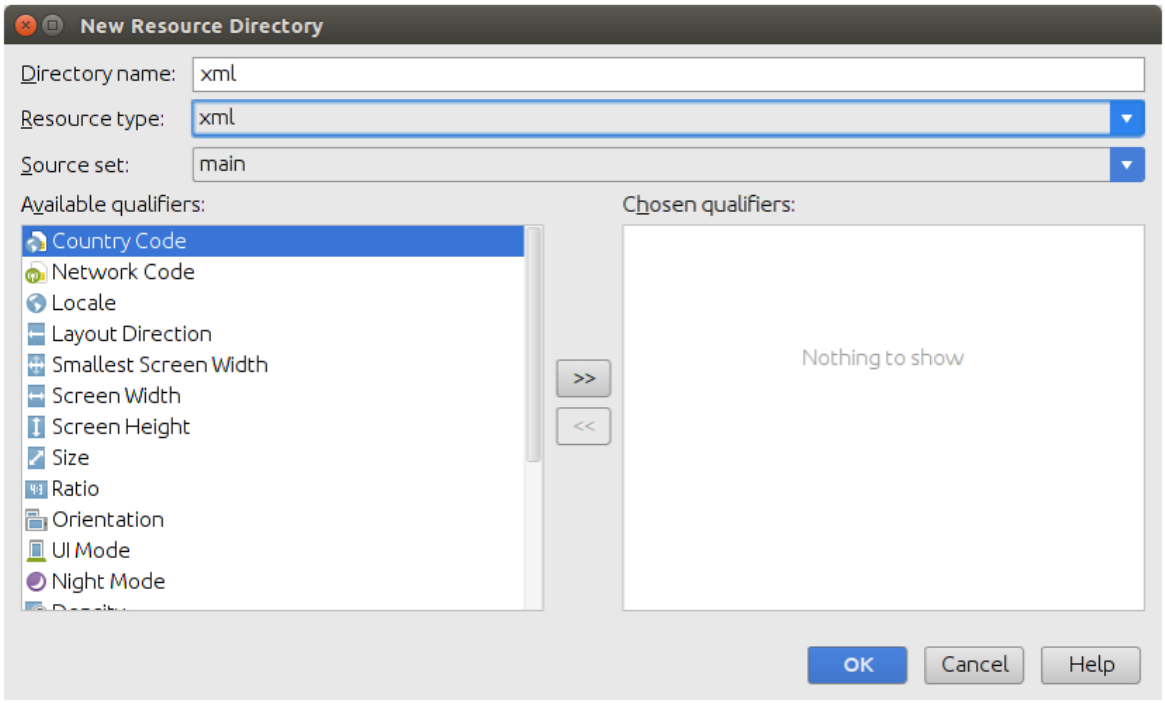
6-12-1-ایجاد فایل preference

اگر پروژه ی شما پوشه ی /app/res/xml ندارد، می بایست آن را ایجاد کنید.

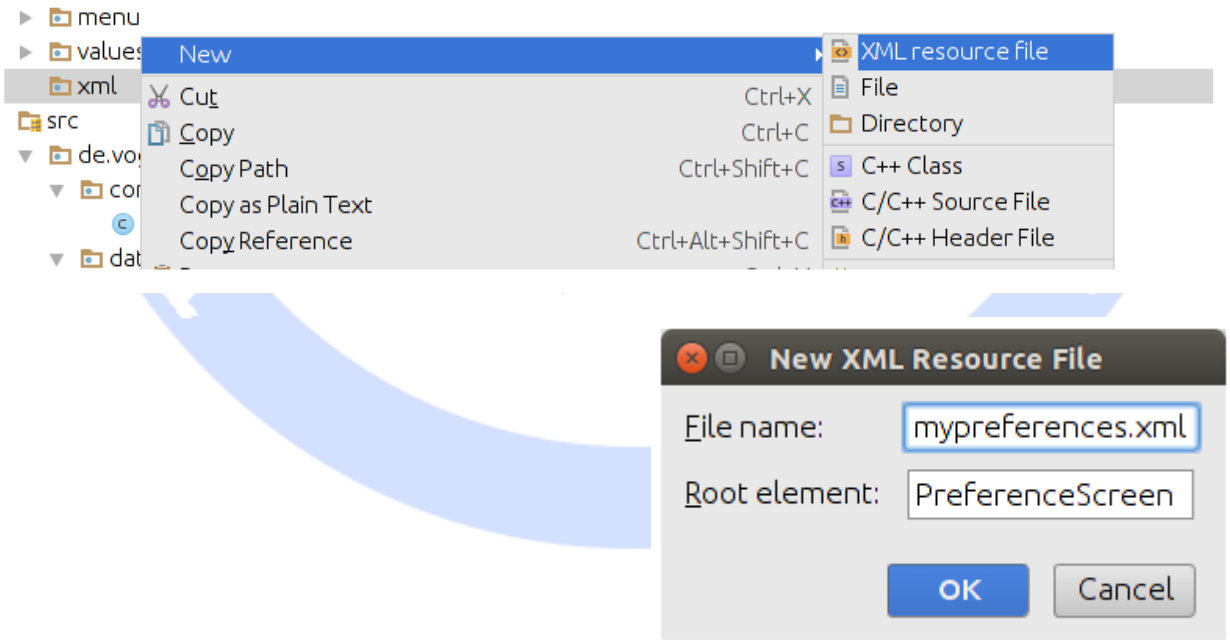


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



اکنون می توانید یک فایل XML در پوشه ی xml ایجاد کنید.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

7-12-1-مشاهده ی محتوای فایل های log اپلیکیشن اندرویدی

جهت مشاهده ی محتوای log و گزارشات ثبت شده از وقایع، کلیدهای Alt+6 را همزمان فشار دهید.

1-13-وظایف و task های مربوط به برنامه نویسی Java

1-13-1-ایجاد کد Getter/Setter، toString ...

جهت دریافت ساختمان یا اسکلت کد getter، setter و توابع سازنده (constructor) به صورت آماده از محیط، ابتدا منوی محیط کاری را باز نموده و سپس مسیر رو به رو را طی نمایید:

Code ▸ Generate.

2-13-1-استفاده از Java 8 در ساخت اپلیکیشن های اندرویدی

جدیدترین افزونه ی Gradle قادر به ترجمه و build کدهای Java 8 است و کاملا با آن سازگار می باشد. از ویرایش Gingerbread (ورژن 9 کتابخانه های اندروید) به بعد می توانید از قابلیت ها و ابزار زیر استفاده نمایید:

1. عبارت های Lambda
2. Java.util.function

از ویرایش 7.0 اندروید به بعد می توانید از قابلیت ها زیر بهره بگیرید:

1. متدهای static و پیش فرض interface (متدهایی که توسط interface برای پیاده سازی در کلاس در اختیار توسعه دهنده قرار می گیرد)
2. Repeatable annotation ها
3. Stream ها
4. Reflection API

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

جهت فعال سازی Java 8 در پروژه، تنظیمات زیر را بکار ببرید. بخش های مهم jackOptions و compileOptions هستند:

```
android {
  ...
  defaultConfig {
    ...
    jackOptions {
      enabled true
    }
  }
  compileOptions {
    sourceCompatibility JavaVersion.VERSION_1_8
    targetCompatibility JavaVersion.VERSION_1_8
  }
}
```

3-13-1-استفاده از live template و تکه کدهای آماده در کد برنامه ی خود

محیط کاری Android Studio برای task ها و وظایف معمول در java و android تکه کدهای آماده ارایه می دهد. لیست زیر پرکاربردترین آن ها را با ذکر کاربرد هر یک، عنوان کرده است:

1. Toast – یک Toast جهت نمایش پیغامی مختصر به صورت زودگذر برای کاربر، تعریف می کند.
2. fbc – متد findViewById را همراه با تبدیل در اختیار شما قرار می دهد.
3. const – یک ثابت یا constant تعریف می کند.

به منظور مشاهده ی لیست کاملی از template های یاد شده، این مسیر را طی نمایید: File ▸ Settings ▸ Editor ▸ Live Templates.

1-14-کار با نرم افزار کنترل نسخه ی Git

1-14-1-Clone یا کپی کردن یک Git repository

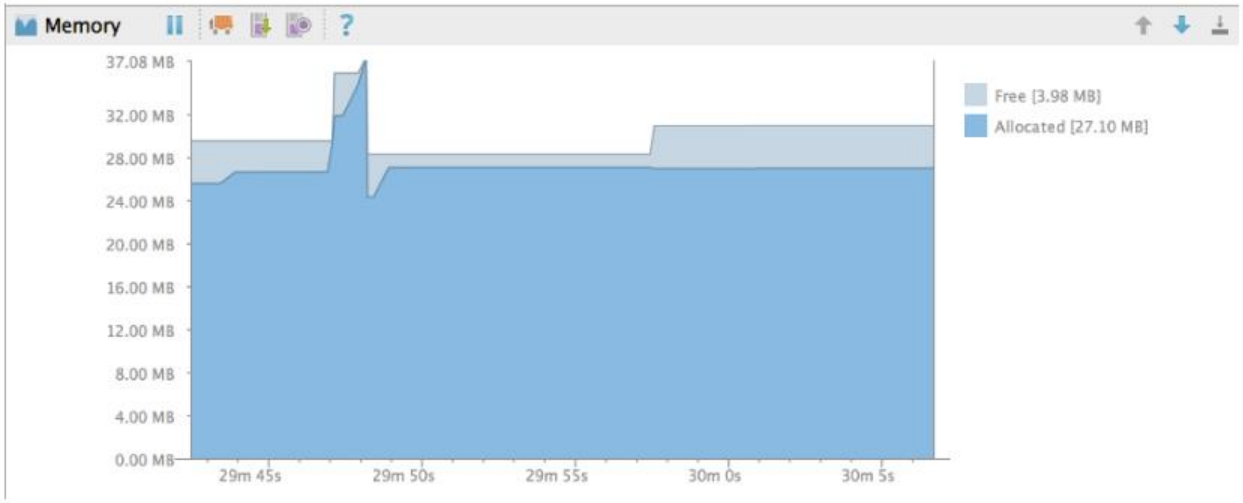
برای ایجاد یک کپی محلی از repository مورد نظر، این مسیر را طی نمایید: VCS ▸ Checkout ▸ Git Version Control

2-14-1-کپی کردن یک repository از Github

Version Control ▸ Github ▸ Checkout

1-15-نظارت بر مصرف حافظه (Memory Monitor)

به منظور مشاهده ی میزان حافظه ای که برنامه مصرف می کند، ابتدا باید اپلیکیشن را اجرا نموده و حین اجرای آن، Android Monitor را باز کنید. پس از کلیک بر روی تب Monitors، Memory Monitor را باز نمایید.



1-16-استفاده از Gradle در محیط برنامه نویسی IntelliJ

1-16-1-محیط توسعه ی IntelliJ و قابلیت پشتیبانی از Gradle

محیط IntelliJ با تنظیمات پیش فرض خود به راحتی از Gradle پشتیبانی می کند.

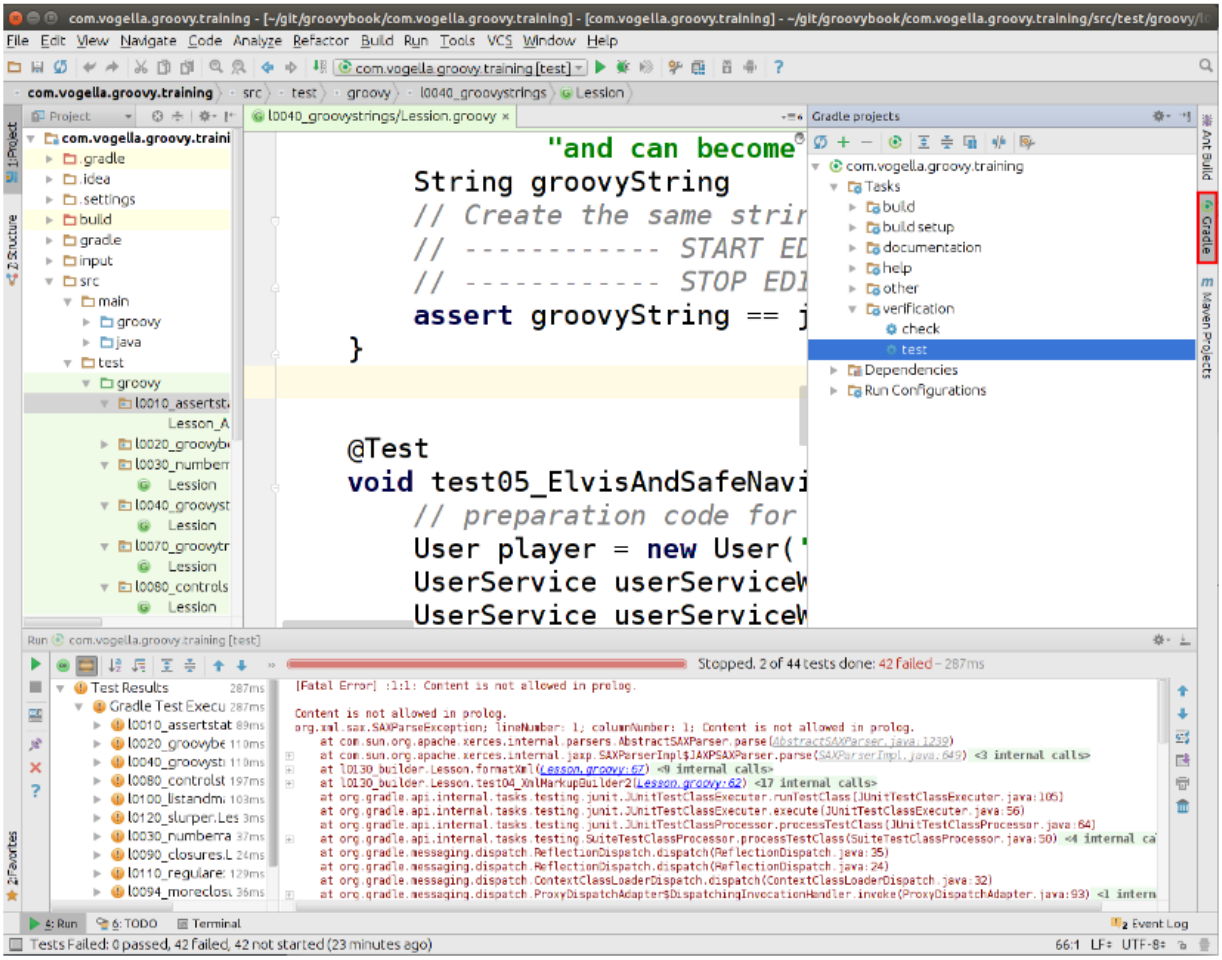
2-16-1-Gradle project view

برای نمایش Gradle projects، این مسیر را طی نمایید: View ▸ Tool Buttons.

با این کار یک نوار ابزار عمودی در سمت راست محیط به نمایش در می آید. با کلیک بر روی دکمه ی میانی، Gradle Projects را باز نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



3-16-1-راه اندازی task های Gradle

بر روی یکی از task ها در Gradle Projects کلیک نمایید تا اجرا شود.

4-16-1-وارد کردن (import) یک پروژه ی آماده ی Gradle در محیط برنامه نویسی IntelliJ

پروژه ی جاری را بسته (File ▸ Close Project) و سپس بر روی لینک Import Project کلیک نمایید.

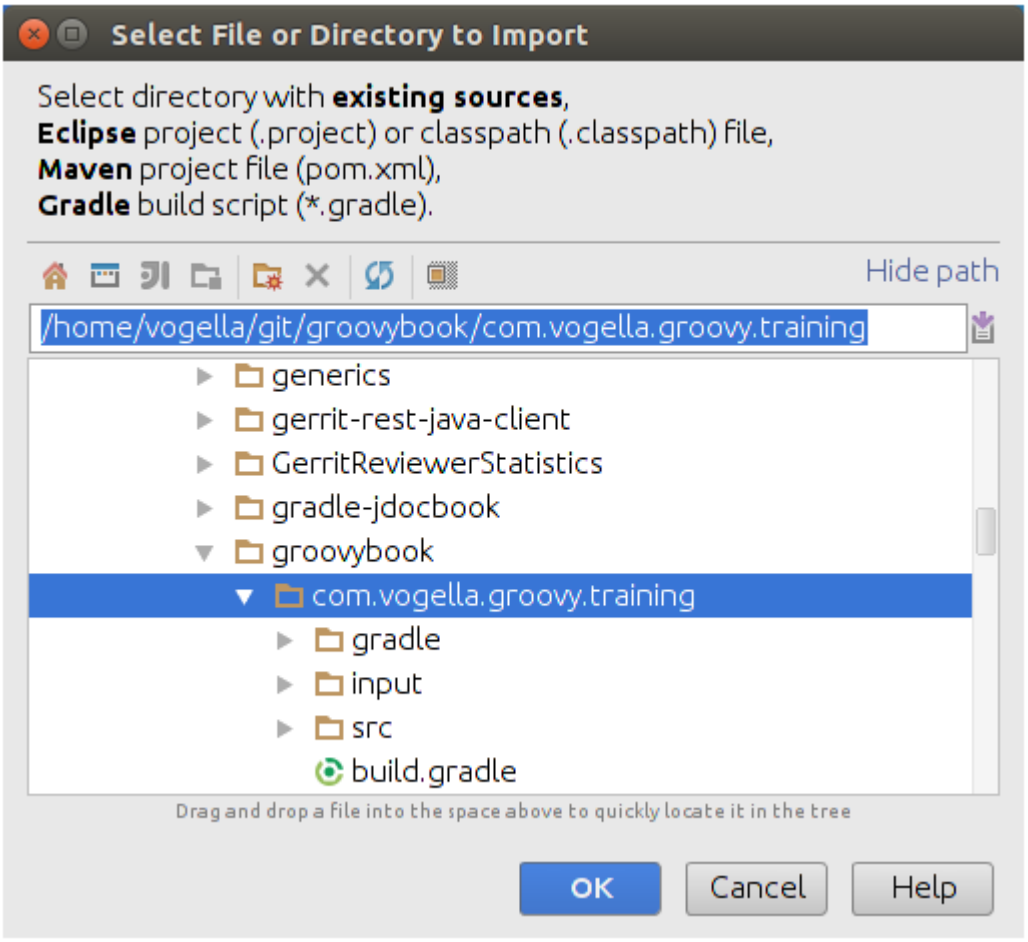
پس از آن مراحلی که در تصاویر زیر نمایش داده شده را به ترتیب دنبال کنید.

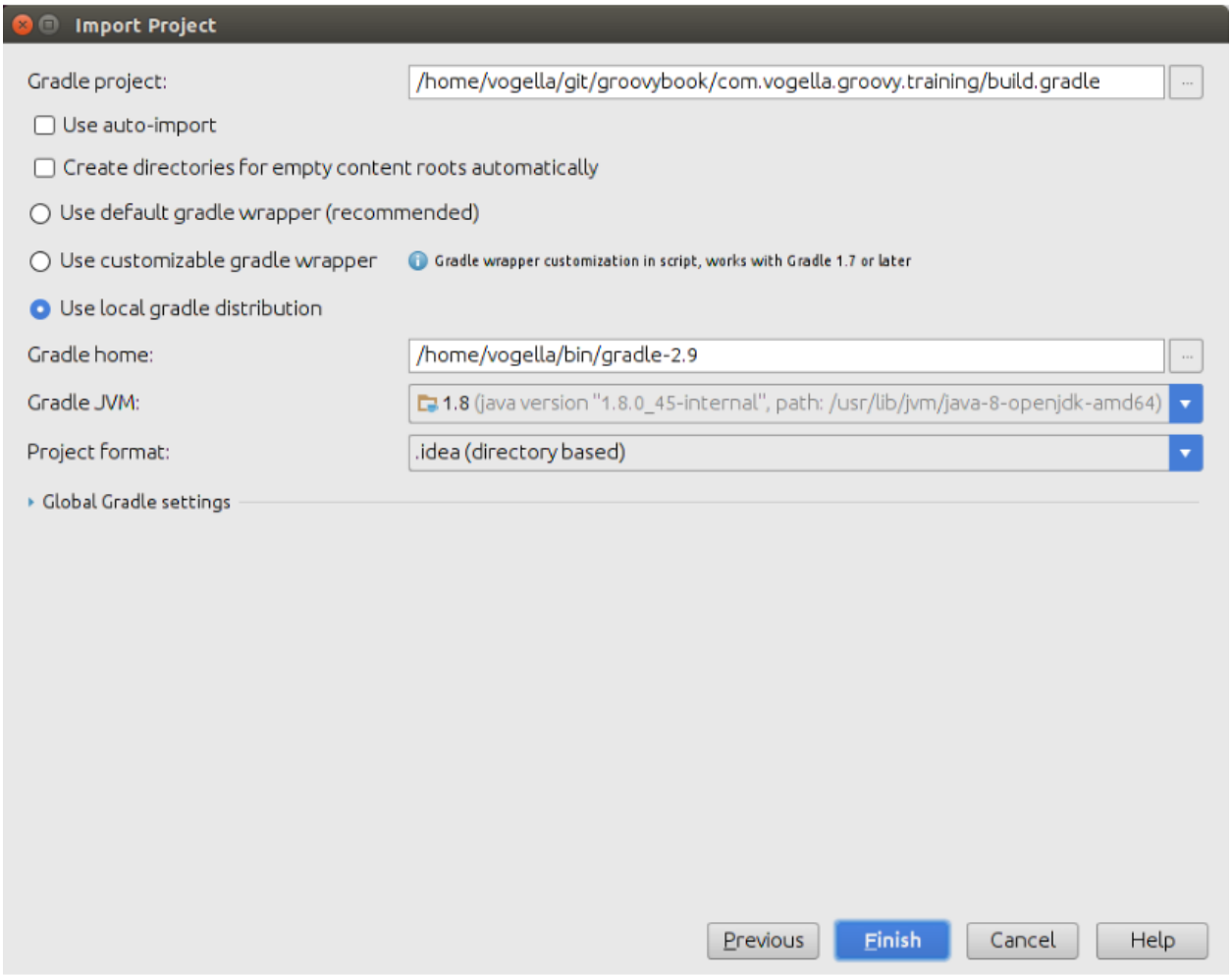
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

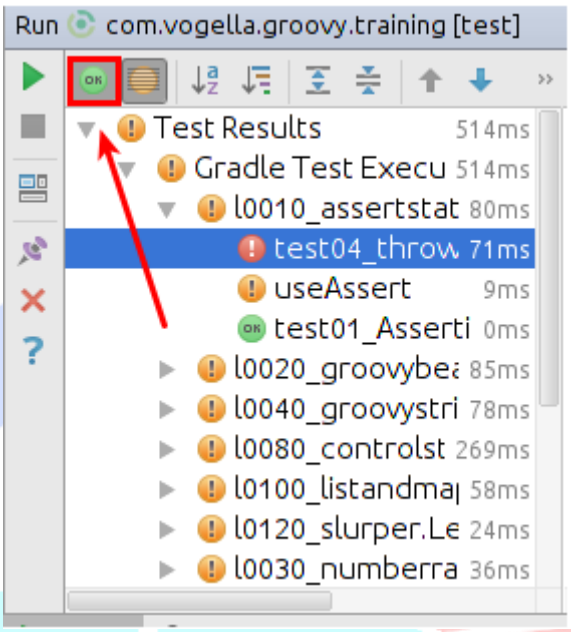
88146323 - 88446780 - 88146330





5-16-1-مشاهده ی نتیجه ی تمامی تست های اجرا شده

بر روی دکمه ی اشاره شده در تصویر زیر کلیک کرده تا نتیجه ی تمامی تست ها (علاوه بر تست های ناموفق) به نمایش درآید.



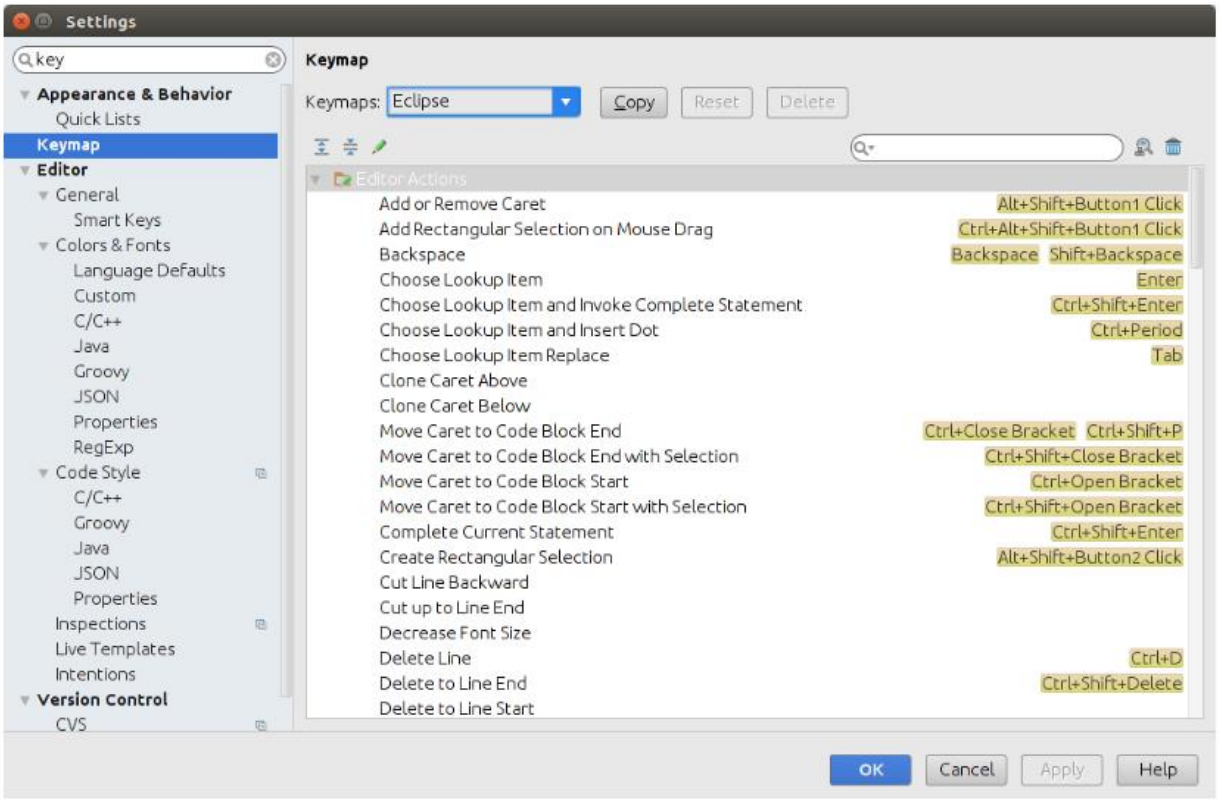
17-1-انتقال پروژه از محیط Eclipse به Android Studio

به منظور انتقال پروژه از محیط کاری Eclipse به Android Studio این مراحل را به ترتیب دنبال نمایید:

1. داخل محیط Android Studio، تمامی پروژه های باز را ببندید. به دنبال این کار صفحه ی Welcome به نمایش در می آید.در این صفحه بر روی لینک Import Non-Android Studio project کلیک نمایید.
2. به پوشه ی پروژه رفته و دکمه ی OK را کلیک نمایید.

17-1-1-ویرایش keybinding جهت استفاده از binding های Eclipse (تغییر تنظیمات صفحه کلید)

برای اینکه سرعت کار شما در محیط جدید افزایش یابد، ابتدا keybinding اندروید استودیو را به keybinding محیط کاری Eclipse تغییر دهید.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فصل دوم آموزش پایه و اساس اندروید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-1-آموزش Intent در Android

این مبحث به آموزش نحوه ی استفاده از Intent جهت برقراری ارتباط بین کامپوننت ها و اجزا نرم افزاری مختلف اندروید می پردازد.

1-1-2-اجرا و راه اندازی دیگر کامپوننت ها و اجزا نرم افزاری اندروید به وسیله ی intent ها (تبادل اطلاعات بین اجزا و کامپوننت های مختلف به وسیله ی آبجکت های intent)

Intent ها پیغام هایی هستند که به واسطه ی آن ها از سیستم تقاضای انجام کار یا عملیات خاصی را می کنید. در واقع با استفاده از آبجکت intent می توانید بین کامپوننت های مختلف یک اپلیکیشن و حتی کامپوننت های نرم افزاری دیگر اپلیکیشن ها تعامل بر قرار نمایید و اطلاعات رد و بدل کنید. برای مثال، یک Activity می تواند activity دیگر را جهت عکس گرفتن اجرا کند.

به وسیله ی intent ها می توانید service ها و Activity ها را اجرا کنید یا مقادیری را از یک activity به activity دیگری ارسال نمایید.

Intent ها آبجکت یا نمونه هایی از جنس کلاس android.content.Intent هستند. کد شما می تواند آبجکت های intent را به سیستم اندروید ارسال کرده و صریحا اعلام کند کدام کامپوننت ها هدف و مد نظر شما هستند. به طور مثال، می توانید یک activity را به کمک متد startActivity() راه اندازی کنید (تعریف کنید که این intent باید فقط به قصد اجرای activity بکار گرفته شود).

داده های مورد نظر را می توانید برای ارسال به activity دیگر داخل Bundle قرار دهید. داده هایی که داخل این آبجکت قرار دادید توسط کامپوننت مد نظر دریافت و مورد استفاده قرار می گیرد.

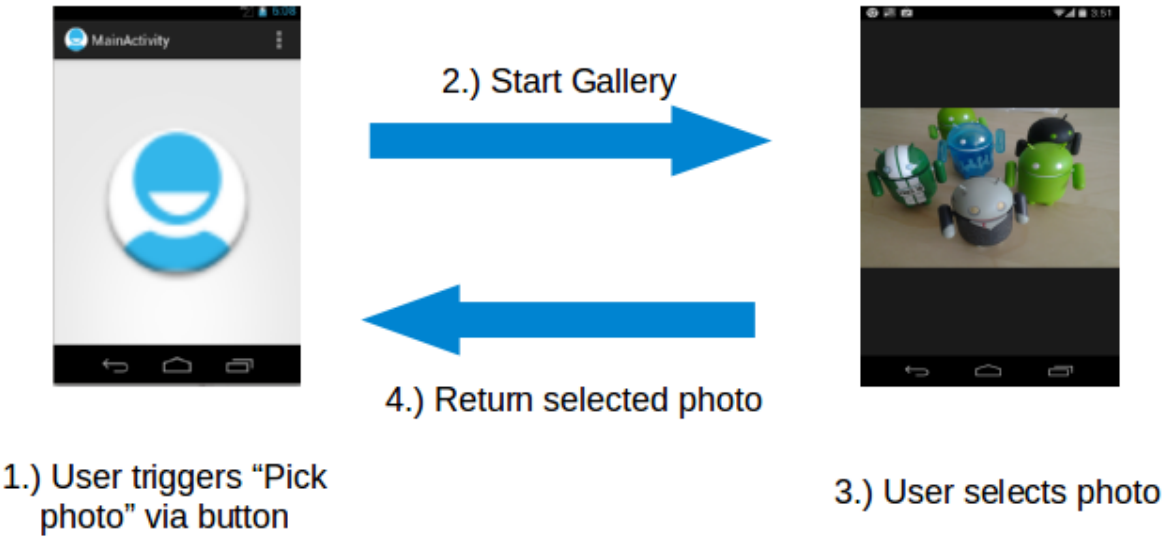
در سیستم اندروید استفاده ی مجدد از دیگر کامپوننت های اپلیکیشن تحت عنوان task شناخته می شود. یک اپلیکیشن می تواند برای انجام عملیات خاصی به دیگر کامپوننت های اندروید دسترسی داشته باشد. به عنوان مثال می توانید از یک کامپوننت اپلیکیشن خود، کامپوننت دیگری در سیستم عامل اندروید را فعال سازی نمایید که فایل های تصویری را مدیریت می کند، با اینکه

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

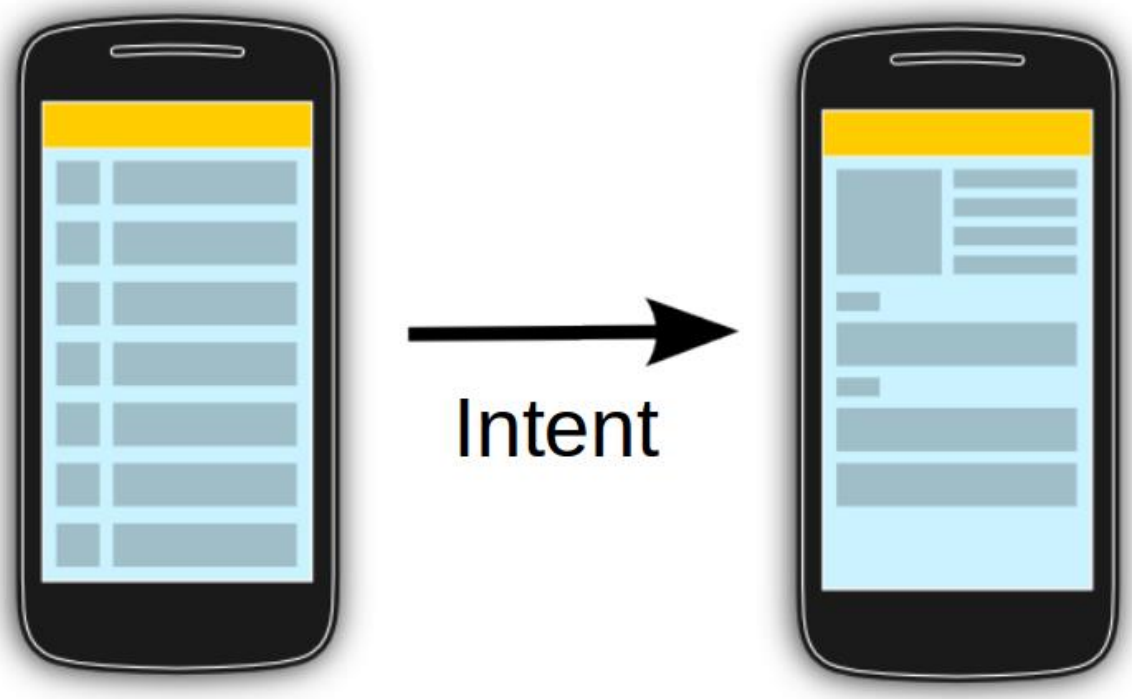
این کامپوننت به اپلیکیشن جاری تعلق ندارد. سپس در این کامپوننت عکس دلخواه را انتخاب کرده و به اپلیکیشن خود برای استفاده از عکس انتخابی باز گردید.

این این جریان تعامل و تبادل اطلاعات در زیر به تصویر کشیده شده است.



2-1-2-راه اندازی Activity ها به وسیله ی آبجکت intent

به منظور راه اندازی یک activity، می توانید متد startActivity(intent) را بکار ببرید. این متد در سطح آبجکت Context تعریف شده است. کلاس Activity از این کلاس پدر ارث بری داشته و متعاقبا متد ذکر شده را نیز از این کلاس به ارث می برد و آن را در بدنه ی خود فراخوانی می کند.



کد زیر نحوه ی راه اندازی یک activity از طریق آبجکت intent را به نمایش می گذارد:

```
# Start the activity connect to the
# specified class
Intent i = new Intent(this, ActivityTwo.class);
startActivity(i);
```

Activity هایی که توسط activity های دیگر راه اندازی می شوند در اصطلاح sub-activity خوانده می شوند.

3-1-2-راه اندازی service ها

گفتنی است که توسط intent ها می توان service ها را نیز راه اندازی نمود. کافی است متد startService(Intent) را فراخوانی نمایید.

4-1-2-ارسال intent های صریح/explicit و ضمنی/implicit

در اندروید دو نوع intent وجود دارد. intent صریح و intent ضمنی.

Intent صریح یا explicit برای صدا زدن و به اجرا در آوردن service یا Activity مشخصی مورد استفاده قرار می گیرد. در این سناریو کاربر نمی تواند تصمیم بگیرد رخداد مورد نظر توسط کدام کامپوننت انجام شود، برای مثال شرایطی را در نظر بگیرید که برنامه ی کاربردی شما دو Activity دارد و شما می خواهید از طریق activity اول به activity دوم راه پیدا کنید. در این حالت لازم است از intent صریح استفاده نمایید. به عبارت دیگر به منظور برقراری ارتباط بین بخش های داخلی یک نرم افزار از intent صریح استفاده می شود. Intent های ضمنی هدف و کامپوننت مد نظر خود را با نام مشخص نمی کنند و معمولا برای راه اندازی کامپوننت دیگر نرم افزار ها مورد استفاده قرار می گیرند. به بیان دیگر زمانی که می خواهید سیستم اندروید عملیاتی را انجام دهد و برای شما چندان اهمیتی ندارد این کار توسط کدام activity یا service انجام می شود، می بایست از intent ضمنی استفاده نمایید. در این سناریو سیستم اندروید تمامی کامپوننت هایی که قابلیت انجام درخواست را دارند، فهرست می کند و کاربر می تواند تصمیم بگیرد درخواست توسط کدام کامپوننت اجرا شود.

کد زیر نحوه ی ایجاد یک intent صریح و ارسال آن به سیستم Android جهت راه اندازی activity دیگر را نمایش می دهد.

```
Intent i = new Intent(this, ActivityTwo.class);
i.putExtra("Value1", "This value one for ActivityTwo ");
i.putExtra("Value2", "This value two ActivityTwo");
```

همان طور که در بالا ذکر شد، هنگامی که intent درخواست انجام عملیات خاصی را از سیستم اندروید می کند، اندروید تمامی نرم افزارهایی که برای اجرای این درخواست و عملیات ثبت شده اند را شناسایی کرده و برای کاربر فهرست می کند. این کار توسط intent filter صورت می پذیرد. برای مثال، کد زیر از اندروید درخواست مشاهده ی یک صفحه ی وب را می کند. تمامی مرورگرهای وب که برای این منظور ثبت شده اند، توسط اندروید شناسایی شده و برای کاربر جهت انتخاب، نمایش داده می شود. این کار توسط intent filter امکان پذیر می باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

کد زیر به سیستم اندروید اعلام می کند که یک صفحه ی وب را باز کند. تمامی مرورگر های نصب شده بر روی سیستم که برای این intent ثبت شده اند، لیست می شوند و کاربر یکی از آن ها را برای مشاهده ی صفحه انتخاب می کند.

```
Intent i = new Intent(Intent.ACTION_VIEW, Uri.parse("http://www.vogella.com"));
startActivity(i);
```

2-2-انتقال و تبادل اطلاعات بین activity ها و service ها

2-2-1-ارسال داده به کامپوننت مقصد

در زمان تعریف یک intent بایستی اطلاعات مرتبط با آن intent را نیز مشخص نمایید. در حقیقت یک intent می تواند دربردارنده ی یک سری اطلاعات پایه ای پیرامون خود باشد همچون اینکه چه عملیاتی باید انجام شود و چه نوع و غیره ... برای هر نوع درخواست نیز ممکن است اطلاعات بیشتری نیاز باشد. یک intent می تواند همچنین داده های اضافی در نمونه ای از کلاس (آبجکت) Bundle داشته باشد که به راحتی از طریق متد getExtras() قابل بازیابی است.

شما می توانید داده های اضافی را مستقیما از طریق نسخه های overload شده ی متد putExtra() از آبجکت های Intent، (نمونه ای از کلاس) Bundle قرار دهید. داده های اضافی در قالب جفت های کلید/مقدار هستند. کلید همیشه از نوع String است و مقادیر آن را می توانید از نوع داده ای اولیه int، float یا آبجکت های از نوع String، Bundle، Parceable و Serializable تنظیم نمایید.

کامپوننت دریافت کننده می تواند به این اطلاعات از طریق توابع getAction() و getData() آبجکت Intent دسترسی داشته باشد. خود آبجکت Intent را نیز می توان از طریق متد getIntent() بازیابی نمود.

کامپوننتی که دریافت کننده ی این intent است، می تواند با فراخوانی getIntent() .getExtras() به داده های اضافی دسترسی داشته باشد. این عملیات در تکه کد زیر به نمایش گذاشته شده است.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
Bundle extras = getIntent().getExtras();
if (extras == null) {
    return;
}
// get data via the key
String value1 = extras.getString(Intent.EXTRA_TEXT);
if (value1 != null) {
    // do something with the data
}
```

نمونه: استفاده از intent جهت به اشتراک گذاری اطلاعات

اغلب اپلیکیشن های اندروید این امکان را به شما می دهند تا داده هایی را با کاربران که عضو اپلیکیشن های Facebbok، G+، Gmail و Twitter هستند به اشتراک بگذارید. در واقع می توانید داده های مورد نظر را از طریق آبجکت Intent به یکی از این کامپوننت ها ارسال کنید. تکه کد زیر روش استفاده از intent جهت به اشتراک گذاری اطلاعات را نمایش می دهد:

```
// this runs, for example, after a button click
Intent intent = new Intent(Intent.ACTION_SEND);
intent.setType("text/plain");
intent.putExtra(android.content.Intent.EXTRA_TEXT, "News for you!");
startActivity(intent);
```

2-2-2-بازیابی اطلاعات از یک subactivity

یک activity را می توانید با اعمال دکمه ی بازگشت در نمایشگر گوشی خود پایان دهید. با فشردن این دکمه، متد finish() فراخوانی می شود. چنانچه activity مورد نظر با صدا خوردن متد startActivity(Intent) اجرا شده باشد، در آن صورت فراخواننده انتظار بازگشت هیچ نتیجه یا بازخوردی از activity مزبور را ندارد.

اگر activity را با فراخوانی متد startActivityForResult() راه اندازی کرده باشید، در آن صورت subactivity طبق انتظار، نتیجه ای را برمی گرداند. با پایان یافتن subactivity، متد onActivityResult() در subactivity فرخوانده می شود و شما می توانید عملیاتی را با توجه به نتیجه بازگشتی انجام دهید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در فراخوانی متد startActivityForResult() شما می توانید کدی (resultcode) جهت شناسایی activity راه اندازی شده تعریف نمایید. این کد به شما بازگردانده می شود. activity فراخوانده شده نیز می تواند کدی داشته باشد که فراخواننده به کمک آن می تواند تشخیص دهد آیا activity لغو شده یا خیر.

Sub-activity با استفاده از finish() یک intent جدید ایجاد کرده و داده ها را در آن می ریزد. سپس به واسطه ی فراخوانی متد setResult() نتیجه را تنظیم می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

نمونه کد زیر نشان می دهد چگونه با اجرای متد startActivityForResult() (و ارسال کلاس ایجاد شده از intent به عنوان آرگومان به این متد)، یک intent را فعال و به اصطلاح trigger نمایید.

```
public void onClick(View view) {
    Intent i = new Intent(this, ActivityTwo.class);
    i.putExtra("Value1", "This value one for ActivityTwo ");
    i.putExtra("Value2", "This value two ActivityTwo");
    // set the request code to any code you like,
    // you can identify the callback via this code
    startActivityForResult(i, REQUEST_CODE);
}
```

زمانی که متد startActivityForResult() را صدا می زنید، activity راه اندازی شده sub-activity خوانده می شود.

هنگامی که subactivity بسته می شود، به دنبالش داده ها را از طریق آبجکت intent به فراخواننده ی خود ارسال می کند. این عملیات داخل بدنه ی finish() پیاده سازی می شود.

```
@Override
public void finish() {
    // Prepare data intent
    Intent data = new Intent();
    data.putExtra("returnKey1", "Swinging on a star. ");
    data.putExtra("returnKey2", "You could be better then you are. ");
    // Activity finished ok, return the data
    setResult(RESULT_OK, data);
    super.finish();
}
```

با بسته شدن یا به پایان رسیدن subactivity، متد onActivityResult() در سطح کلاس activity فراخواننده صدا خورده می شود.

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (resultCode == RESULT_OK && requestCode == REQUEST_CODE) {
        if (data.hasExtra("returnKey1")) {
            Toast.makeText(this, data.getExtras().getString("returnKey1"),
                Toast.LENGTH_SHORT).show();
        }
    }
}
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-3-ثبت و معرفی intent filter برای intent

1-3-2-Intent filter چیست؟

Intent ها رخداد یک اتفاق را به سیستم اندروید اعلان می کنند و در پی آن تمامی Activity و Service هایی که برای آن اتفاق در سیستم ثبت شده اند، صدا خورده و اجرا می شوند. Intent ها معمولا عملیاتی که باید اجرا شوند را شرح داده و اطلاعاتی را نیز درباره ی عملیاتی که باید اجرا شود ارائه می دهد. برای مثال، اپلیکیشن شما می تواند جهت دسترسی به آدرس URL خاص، با استفاده از آبجکت intent یک کامپوننت مرورگر را راه اندازی کند. در تکه کد زیر این عملیات به نمایش گذاشته شده است.

```
String url = "http://www.vogella.com";
Intent i = new Intent(Intent.ACTION_VIEW);
i.setData(Uri.parse(url));
startActivity(i);
```

اینجا یک سوال مطرح می شود: چگونه سیستم اندروید می تواند کامپوننت هایی را که قادر به اجرای درخواست intent خاص و واکنش نشان دادن به آن هستند را شناسایی کند؟

در سیستم اندروید، کامپوننت ها می توانند به وسیله ی intent filter خود را برای عملیات یا داده ای خاص ثبت کنند و در واقع اعلان نمایند که قابلیت انجام درخواست و عملیات intent مورد نظر را دارند. به بیانی دیگر intent filter با تعریف قابلیت های یک کامپوننت، نوع intent ای که یک activity، service یا broadcast receiver قادر به پاسخ گویی آن است را مشخص می کند.

کامپوننت های اندروید intent filter را یا به صورت static و ثابت در فایل AndroidManifest.xml تعریف می کنند و یا در خصوص broadcast reciever به صورت dynamic و از طریق کدنویسی. Intent filter از طریق category، action و data تعریف می شود (در فایل تنظیمات به وسیله ی تگ <intent-filter> می توانید فهرستی از actionها، categoryها و data هایی را که با هر کدام از کامپوننت های activity، service یا broadcast receiver مرتبط هستند را معرفی کنید). intent filter می تواند meta data نیز داشته باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

زمانی که یک intent به سیستم اندروید ارسال می شود، محیط (platform) یا بستر اجرای android دریافت کننده ی آن را شناسایی می کند. این کار را با استفاده از داده های موجود در آبجکت intent انجام می دهد. در صورتی که چندین کامپوننت برای یک intent filter ثبت نام کرده و گوش بدهند، در آن صورت سیستم اندروید لیستی از کامپوننت هایی که باید راه اندازی شده و پردازش یا درخواست را اجرا کنند برای کاربر نمایش داده و به دنبال آن کاربر می تواند تصمیم بگیرد کدام کامپوننت باید اجرا شود.

2-3-2-تعریف intent filter

همان طور که گفته شد سیستم اندروید به وسیله ی intent filter تشخیص می دهد آیا یک برنامه ی کاربردی قابلیت اجرای درخواست معینی را دارد یا خیر. در سیستم اندروید برنامه نویس می تواند با استفاده از intent filter امکاناتی که نرم افزار دارد را اعلان کند.

می توانید کامپوننت های اندرویدی خود را به وسیله ی intent filter برای event و رخدادهای خاص معرفی کنید (در واقع به کامپوننت ها اعلان کنید که به رخدادهای مورد نظر گوش داده و منتظر فعال شدن آن ها باشند). در صورتی که این کار برای کامپوننت معینی انجام نشده باشد، آنگاه فقط intent های صریح (explicit) می توانند آن کامپوننت را فراخوانی کنند. مبحث حاضر نحوه ی ثبت یک کامپوننت برای intent معین را با ذکر مثال شرح می دهد. نکته ی مهم و کلیدی این است که کامپوننت ها برای action، mime-type مناسب ثبت نام کرده و گوش فرادهند و meta-data (اطلاعاتی پایه ای درباره ی intent) مربوطه را دربر گیرد.

اگر شما چنین intent ای را به سیستم اندروید ارسال کنید، سیستم تمامی کامپوننت هایی که برای این intent به ثبت رسیده اند را شناسایی می کند. در صورتی که چندین کامپوننت برای این intent ثبت شده باشند، سیستم اندروید آن ها را برای کاربر لیست می کند و کاربر می تواند تصمیم بگیرد با کامپوننت درخواست اجرا شود.

مثال: ثبت و تخصیص یک activity به عنوان مرورگر

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

کد زیر یک activity را برای intent ثبت می کند. این intent زمانی که کاربر می خواهد یک صفحه ی وب را باز کند، فعال شده و فراخوانی می گردد.

```
<activity android:name=".BrowserActivitiy"
        android:label="@string/app_name">
  <intent-filter>
    <action android:name="android.intent.action.VIEW" />
    <category android:name="android.intent.category.DEFAULT" />
    <data android:scheme="http"/>
  </intent-filter>
</activity>
```

مثال: ثبت و تخصیص یک activity برای intent به اشتراک گذاری و ارسال اطلاعات

مثال زیر یک activity ویژه ی ACTION_SEND ثبت می کند. همان طور که در کد زیر می بینید activity حاضر تنها قادر به پردازش داده هایی از نوع text/plain می باشد (با استفاده از intent-filter خود را مربوط به این داده اعلان یا ثبت کرده است.)

```
<activity
      android:name=".ActivityTest"
    android:label="@string/app_name" >
    <intent-filter>
      <action android:name="android.intent.action.SEND" />
      <category android:name="android.intent.category.DEFAULT" />
      <data android:mimeType="text/plain" />
    </intent-filter>
</activity>
```

اگر یک کامپوننت intent filter تعریف نکند، در آن صورت فقط intent صریح می تواند آن را فراخوانی کند.

2-4-گوش فرا دادن به event ها (ارسال پیغام به سیستم اندروید از طریق intent)

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

با استفاده از intent ها می توان پیغام هایی (broadcast message) را به سیستم اندروید ارسال کرد. Broadcast receiver می تواند به رخدادی گوش فرا داده (register کرده) و زمانی که رخداد اتفاق افتاد، از آن مطلع شود.

اپلیکیشن شما نیز می تواند به رخدادهای سیستم همچون دریافت ایمیل جدید، بالا آمدن سیستم، تماس جدید گوش فرا داده، و متناسب با آن عکس العمل مربوطه را نشان دهد.

2-5-شناسایی intent receiver های مربوطه/بررسی اینکه آیا یک کامپوننت به intent خاصی گوش فرا می دهد/برای آن ثبت شده یا خیر

گاهی لازم می شود برنامه نویس بررسی کرده و مطمئن شوید آیا کامپوننت خاص برای یک intent ثبت شده و به آن گوش می دهد یا خیر. برای مثال، می خواهید ببینید آیا intent receiver خاصی (یک کامپوننت) وجود دارد یا خیر، و در صورت وجود آن کامپوننت قابلیت خاصی را در اپلیکیشن خود فعال نمایید.

برای این منظور می توانید از کلاس PackageManager استفاده کنید.

نمونه کد زیر بررسی می کند آیا کامپوننتی برای intent مورد نظر ثبت شده است یا خیر. کافی است intent خود را ساخته و آن را به متد زیر به عنوان آرگومان ارسال کنید.

```
public static boolean isIntentAvailable(Context ctx, Intent intent) {
    final PackageManager mgr = ctx.getPackageManager();
    List<ResolveInfo> list =
        mgr.queryIntentActivities(intent,
            PackageManager.MATCH_DEFAULT_ONLY);
    return list.size() > 0;
}
```

با توجه به نتیجه ی بازگشتی می توانید اپلیکیشن خود را تنظیم کنید. برای مثال، می توانید برخی از آیتم های منو را غیرفعال ساخته یا پنهان نمایید.

تمرین: راه اندازی activity به وسیله ی intent صریح

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

تمرین زیر نشان می دهد چگونه با استفاده از یک intent صریح، یک subactivity را راه اندازی نموده و به آن داده هایی را ارسال کنید.

1-5-2-ساخت پروژه و فایل layout اصلی

یک پروژه ی جدید اندروید به نام com.vogella.android.intent.explicit ایجاد کرده (این اسم را در فیلد package وارد نمایید) و activity آن را MainActivity نام گذاری نمایید.

حال محتویات فایل layout این activity را به صورت زیر ویرایش نمایید:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   tools:context=".MainActivity" >
   <EditText
      android:id="@+id/inputforintent"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:minHeight="60dip"
      android:text="First Activity"
      android:textSize="20sp" >
   </EditText>
   <Button
      android:id="@+id/startintent"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignLeft="@+id/inputforintent"
      android:layout_below="@+id/inputforintent"
      android:onClick="onClick"
      android:text="Calling an intent" />
</RelativeLayout>
```

2-5-2-ایجاد یک فایل layout جدید

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <TextView
      android:id="@+id/displayintentextra"
      android:layout_width="wrap_content"
```

```
    android:layout_height="wrap_content"
    android:text="Input"
   />
  <EditText
    android:id="@+id/returnValue"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
    <requestFocus />
  </EditText>
</LinearLayout>
```

3-5-2-ایجاد یک activity دیگر

کلاس activity دیگری به نام ResultActivity تعریف نموده و کد موجود در بدنه ی این کلاس را مانند نمونه ی زیر نگارش نمایید.

```
package com.vogella.android.intent.explicit;
import android.app.Activity;
import android.os.Bundle;
public class ResultActivity extends Activity {
    @Override
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_result);
    }
}
```

کلاس activity فوق را به فایل AndroidManifest.xml اضافه نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.vogella.android.first"
  android:versionCode="1"
  android:versionName="1.0" >
  <uses-sdk android:minSdkVersion="14" />
  <application
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name" >
    <activity
      android:label="@string/app_name"
      android:name=".MainActivity" >
      <intent-filter >
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
    <activity
      android:label="Result Activity"
      android:name=".ResultActivity" >
    </activity>
```

```
    </application>
</manifest>
```

نکته: از آنجایی که activity ذکر شده توسط acitivity دیگری (کلاس MainActivity) راه اندازی می شود، از این به بعد تحت عنوان subactivity به آن اشاره می کنیم.

4-5-2-راه اندازی subactivity

Subactivity را با کلیک بر روی یک دکمه در صفحه ی اصلی برنامه که همان کلاس MainActivity محسوب می شود، راه اندازی نمایید. کد زیر شما را در خصوص پیاده سازی این عملیات راهنمایی می کند. با نوشتن دستورات مناسب بجای TODO ها در کد برنامه، اپلیکیشن خود را طوری بنویسید که ResultActivity با فراخوانی متد ()OnClick راه اندازی شود (به محض کلیک بر روی دکمه در صفحه ی اصلی، اپلیکیشن کاربر را به activity دوم هدایت کند).

```
package com.vogella.android.intent.explicit;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
public class MainActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void onClick(View view) {
        EditText text = (EditText) findViewById(R.id.inputforintent);
        // used later
        String value = text.getText().toString();
        // TODO 1 create new Intent(context, class)
        // use the activity as context parameter
        // and "ResultActivity.class" for the class parameter
        // TODO 2 start second activity with
        // startActivity(intent);
    }
}
```

پس از اتمام این بخش از تمرین، اپلیکیشن خود را راه اندازی نموده و مطمئن شوید که کاربر با کلیک بر روی دکمه در صفحه ی اول اپلیکیشن، به activity دوم راه پیدا می کند.

5-5-2-انتقال داده (ارسال مقدار) از activity اول به activity دوم (ResultActivity)

کلاس MainActivity می بایست مقدار EditText را به subactivity ارسال کند. برای این منظور کافی است متد putExtra("yourKey", string) را بر روی آبجکت Intent بکار ببرید.

5-5-2-دریافت اطلاعات ارسالی از activity اول (داده های intent) در ResultActivity

حال داده های ارسالی از activity اصلی اپلیکیشن را از Bundle که بخشی از آبجکت intent است با فراخوانی متد getIntent().getExtras() در کلاس ResultActivity بازیابی نمایید.

حال مقدار extra را با فراخوانی متد extras.getString("yourkey") بر روی آبجکت bundle که در نتیجه ی فراخوانی متد getExtras() بدست آوردید، بازیابی نمایید.

این مقدار می بایست داخل کلاس TextView با ID یا شناسه ی displayintentextra جایگذاری شود.

تمرین: بازیابی داده ها از ResultActivity (انتقال داده ها از activity دوم به activity اصلی برنامه)

در تمرین زیر کدی به متن برنامه اضافه می کنید که با کلیک کاربر بر روی دکمه ی Back در activity دوم، داده هایی را از subactivity به activity اصلی اپلیکیشن ارسال کند.

برای نیل به این هدف، متد finish() را در کلاس ResultActivity پیاده سازی می کنید.

```
@Override
public void finish() {
    // TODO 1 create new Intent
    // Intent intent = new Intent();
    // TODO 2 read the data of the EditText field
    // with the id returnValue
    // TODO 3 put the text from EditText
    // as String extra into the intent
    // use editText.getText().toString();
    // TODO 4 use setResult(RESULT_OK, intent);
    // to return the Intent to the application
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
        super.finish();
}
```

در جایگاه TODO ها در کد فوق، بدنه ی متد finish() را پیاده سازی نمایید.

Solve all TODOs.

7-5-2-دریافت و ارزیابی داده های بازگشتی (ارسالی از activity دوم) در کلاس MainActivity

متد startActivityForResult() را در کلاس MainActivity فراخوانی نمایید تا subactivity راه اندازی شود. این کار به شما امکان می دهد تا متد onActivityResult() را برای دریافت داده از subactivity فراخوانی نمایید. حال داده های ارسالی extra که همراه با آبجکت bundle از activity دوم ارسال شده، استخراج نمایید.

بد نیست همراه داده های extra یک پیغام Toast نیز برای کاربر نمایش داده و اطمینان دهید که داده ها طبق انتظار دریافت شدند. کد زیر شما را در خصوص پیاده سازی این عملیات راهنمایی می کند:

```
package com.vogella.android.intent.explicit;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
public class MainActivity extends Activity {
    // constant to determine which sub-activity returns
    private static final int REQUEST_CODE = 10;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void onClick(View view) {
        EditText text = (EditText) findViewById(R.id.inputforintent);
        String string = text.getText().toString();
        Intent i = new Intent(this, ResultActivity.class);
        i.putExtra("yourkey", string);
        // TODO 2.. now use
        // startActivityForResult(i, REQUEST_CODE);
    }
```

```
    // TODO 3 Implement this method
    // assumes that "returnkey" is used as key to return the result
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        if (resultCode == RESULT_OK && requestCode == REQUEST_CODE) {
            if (data.hasExtra("returnkey")) {
                String result = data.getExtras().getString("returnkey");
                if (result != null && result.length() > 0) {
                    Toast.makeText(this, result, Toast.LENGTH_SHORT).show();
                }
            }
        }
    }
}
```

راه حل: استفاده از intent ها

با تکمیل این تمرین، پیاده سازی activity شما می بایست مشابه کد کلاس های زیر باشد.

```
package com.vogella.android.intent.explicit;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.widget.EditText;
import android.widget.TextView;
public class ResultActivity extends Activity {
    @Override
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_result);
        Bundle extras = getIntent().getExtras();
        String inputString = extras.getString("yourkey");
        TextView view = (TextView) findViewById(R.id.displayintentextra);
        view.setText(inputString);
    }
    @Override
    public void finish() {
        Intent intent = new Intent();
        EditText editText= (EditText) findViewById(R.id.returnValue);
        String string = editText.getText().toString();
        intent.putExtra("returnkey", string);
        setResult(RESULT_OK, intent);
        super.finish();
    }
}
```

```
package com.vogella.android.intent.explicit;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends Activity {
    // constant to determine which sub-activity returns
    private static final int REQUEST_CODE = 10;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void onClick(View view) {
        EditText text = (EditText) findViewById(R.id.inputforintent);
        String string = text.getText().toString();
        Intent i = new Intent(this, ResultActivity.class);
        i.putExtra("yourkey", string);
        startActivityForResult(i, REQUEST_CODE);
    }
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        if (resultCode == RESULT_OK && requestCode == REQUEST_CODE) {
            if (data.hasExtra("returnkey")) {
                String result = data.getExtras().getString("returnkey");
                if (result != null && result.length() > 0) {
                    Toast.makeText(this, result, Toast.LENGTH_SHORT).show();
                }
            }
        }
    }
}
```

تمرین: استفاده از intent به اشتراک گذاری و تبادل اطلاعات

یک دکمه ی دیگر به فایل layout مربوط به activity خود اضافه نمایید. کلیک بر روی این دکمه می بایست این امکان را فراهم بیاورد تا متن EditText را به اشتراک بگذارید. برای ارسال اطلاعات از اپلیکیشن دیگر، می بایست مقدار مربوطه را در ثابت Intent.ACTION_SEND قرار دهید. سپس یک متد با این تعریف public void shareData(View view) بر اساس تکه کد زیر در activity خود ایجاد نمایید.

```
Intent sharingIntent = new Intent(Intent.ACTION_SEND);
sharingIntent.setType("text/plain");
sharingIntent.putExtra(android.content.Intent.EXTRA_TEXT, "REPLACE WITH YOUR TEXT");
startActivity(sharingIntent);
```

این متد را به وسیله ی property یا خصوصیت android:onClick در فایل layout خود به دکمه ی مربوطه متصل نمایید. سپس این برنامه را اجرا نموده و بر روی دکمه ی مورد نظر کلیک کنید. در محیط شبیه ساز، قاعدتا تنها برنامه ای که به این intent گوش فرا می دهد و در سیستم اندروید برای آن ثبت شده و متعاقبا مستقیما راه اندازی می شود، اپلیکیشن ارسال SMS می باشد.

تمرین: تخصیص و ثبت یک activity به عنوان مرورگر

هدف اصلی

در تمرین زیر یک activity را در فایل XML خود به عنوان مرورگر ثبت می کنید. پس از این کار، هر زمان که کاربر می خواهد آدرس URL ای که با http آغاز می شود را مشاهده کند، activity ای ذکر شده، برای پردازش intent مربوطه در دسترس خواهد بود.

activity نام برده کد HTML این صفحه را دانلود کرده و آن را در یک TextView به نمایش می گذارد.

8-5-2-ایجاد پروژه

یک پروژه ی اندروید و یک activity به ترتیب به نام های de.vogella.android.intent.browserfilter و BrowserActivity ایجاد نمایید.

9-5-2-ثبت و تخصیص یک activity به عنوان مرورگر

Attribute های android:name و android:scheme را در فایل تنظیمات AndroidManifest.xml به ترتیب با Intent.Action_VIEW و "http" مقداردهی نمایید. همچنین

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

لازم است در فایل manifest اجازه ی دسترسی به اینترنت را از طریق تگ uses-permission تنظیم نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="de.vogella.android.intent.browserfilter"
   android:versionCode="1"
   android:versionName="1.0" >
   <uses-sdk android:minSdkVersion="15" />
   <uses-permission android:name="android.permission.INTERNET" >
   </uses-permission>
   <application
      android:icon="@drawable/icon"
      android:label="@string/app_name" >
      <activity
         android:name=".BrowserActivity"
         android:label="@string/app_name" >
         <intent-filter>
            <action android:name="android.intent.action.VIEW" />
            <category android:name="android.intent.category.DEFAULT" />
            <data android:scheme="http" />
         </intent-filter>
      </activity>
   </application>
</manifest>
```

حال فایل layout مربوطه را به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   >
<TextView
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:id="@+id/textView"/>
</LinearLayout>
```

کد کلاس activity را نیز به ترتیب زیر ویرایش نمایید.

```
package de.vogella.android.intent.browserfilter;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.URL;
import android.app.Activity;
import android.content.Intent;
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
import android.net.Uri;
import android.os.Bundle;
import android.os.StrictMode;
import android.widget.TextView;
public class BrowserActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // To keep this example simple, we allow network access
        // in the user interface thread
        StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder()
                .permitAll().build();
        StrictMode.setThreadPolicy(policy);
        setContentView(R.layout.main);
        Intent intent = getIntent();
        TextView text = (TextView) findViewById(R.id.textView);
        // To get the action of the intent use
        String action = intent.getAction();
        if (!action.equals(Intent.ACTION_VIEW)) {
            throw new RuntimeException("Should not happen");
        }
        // To get the data use
        Uri data = intent.getData();
        URL url;
        try {
            url = new URL(data.getScheme(), data.getHost(), data.getPath());
            BufferedReader rd = new BufferedReader(new InputStreamReader(
                    url.openStream()));
            String line = "";
            while ((line = rd.readLine()) != null) {
                text.append(line);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

10-5-2-تست برنامه

اپلیکیشن خود را نصب نمایید. زمانی که یک intent را جهت مشاهده ی محتویات صفحه ی URL فراخوانی یا فعال می کنید، کاربر قاعدتا باید بتواند پیاده سازی سفارشی مرورگر شما را انتخاب کند.

در این راستا یک دکمه ی اضافی به اپلیکیشن خود اضافه می کنید که با کلیک بر روی آن کد intent فراخوانی می شود.

```
intent = new Intent(Intent.ACTION_VIEW,Uri.parse("http://www.vogella.com"));
startActivity(intent);
```

پس از انتخاب کامپوننت دلخواه از میان کامپوننت های پیشنهادی توسط سیستم، کد HTML مورد نظر ابتدا در حافظه بارگذاری شده، سپس در المان رابط کاربری TextView به نمایش در می آید.

11-5-2-تبدیل اپلیکیشن به یک مرورگر واقعی

در صورت تمایل می توانید TextView را با یک WebView جایگزین نموده و اپلیکیشن خود را به یک مرورگر واقعی تبدیل نمایید. WebView خود درخواست HTTP را برای شما مدیریت و بارگذاری می کند. کافی است آدرس URL را به واسطه ی متد loadURL به آن تخصیص دهید.

تمرین: انتخاب یک عکس به وسیله ی آبجکت intent

هدف

مثال زیر نمایش می دهد چگونه می توان با استفاده از یک intent، عکس دلخواه را از اپلیکیشن های تخصیص یافته برای این منظور در سیستم اندروید، انتخاب نمود.

ایجاد پروژه

یک پروژه ی جدید اندروید و یک activity به ترتیب با نام های de.vogella.android.imagepick و ImagePickActivity ایجاد نمایید.

فایل layout مربوط به activity_main.xml را به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <Button
      android:id="@+id/button1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:onClick="pickImage"
      android:text="Button" >
   </Button>
   <ImageView
      android:id="@+id/result"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:src="@drawable/icon" >
   </ImageView>
</LinearLayout>
```

کد کلاس activity خود را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.imagepick;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;

import android.app.Activity;
import android.content.Intent;
import android.graphics.Bitmap;
```

```
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
public class ImagePickActivity extends Activity {
	private static final int REQUEST_CODE = 1;
	private Bitmap bitmap;
	private ImageView imageView;
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
		imageView = (ImageView) findViewById(R.id.result);
	}
	public void pickImage(View View) {
		Intent intent = new Intent();
		intent.setType("image/*");
		intent.setAction(Intent.ACTION_GET_CONTENT);
		intent.addCategory(Intent.CATEGORY_OPENABLE);
		startActivityForResult(intent, REQUEST_CODE);
	}
	@Override
	protected void onActivityResult(int requestCode, int resultCode, Intent data) {
		InputStream stream = null;
		if (requestCode == REQUEST_CODE && resultCode == Activity.RESULT_OK)
			try {
				// recyle unused bitmaps
				if (bitmap != null) {
					bitmap.recycle();
				}
				stream = getContentResolver().openInputStream(data.getData());
				bitmap = BitmapFactory.decodeStream(stream);
				imageView.setImageBitmap(bitmap);
			} catch (FileNotFoundException e) {
				e.printStackTrace();
			} finally {
				if (stream != null)
					try {
						stream.close();
					} catch (IOException e) {
						e.printStackTrace();
					}
			}
	}
}
```

تست اپلیکیشن

با توجه به کدی که برای اپلیکیشن خود نوشته اید، باید پس از اجرای آن بتوانید عکس دلخواه را از image library مستقر در دستگاه اندروید خود انتخاب نموده و سپس آن را به ImageView تخصیص دهید.

تمرین: استفاده از intent های ضمنی جهت راه اندازی activity ها

هدف

تمرین زیر نحوه ی استفاده از intent های ضمنی جهت راه اندازی activity ها در سیستم اندروید را برای شما شرح می دهد.

ایجاد پروژه

یک پروژه ی جدید و یک activity به ترتیب به نام های de.vogella.android.intent.implicit و CallIntentsActivity ایجاد نمایید.

در این مثال، با پیاده سازی کلاس Spinner، یک لیست کشویی ایجاد می کنید که به کاربر اجازه می دهد تا خود تصمیم بگیرد کدام intent را فراخوانی کند. محتوای این لیست کشویی (Spinner) را مقادیر static و ثابت تعریف خواهید نمود.

برای این منظور فایل intents.xml زیر را در پوشه ی res/values ایجاد نمایید.

```
<resources>
  <string-array name="intents">
    <item>Open Browser</item>
    <item>Call Someone</item>
    <item>Dial</item>
    <item>Show Map</item>
    <item>Search on Map</item>
    <item>Take picture</item>
    <item>Show contacts</item>
    <item>Edit first contact</item>
  </string-array>
</resources>
```

فایل layout مربوط به activity مورد نظر را به صورت زیر ویرایش نمایید.

Change the layout file of the activity to the following.

```
<?xml version="1.0" encoding="utf-8"?>
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:alignmentMode="alignBounds"
   android:columnCount="1" >
    <Spinner
     android:id="@+id/spinner"
     android:layout_gravity="fill_horizontal"
     android:drawSelectorOnTop="true"
     >
    </Spinner>
   <Button
     android:id="@+id/trigger"
     android:onClick="onClick"
     android:text="Trigger Intent">
   </Button>
</GridLayout>
```

برای اینکه بتوانید از intent های مورد نظر استفاده کنید، لازم است مجوزهای مربوطه را در فایل تنظیمات AndroidManifest.xml ثبت نمایید. در واقع فایل تنظیمات اپلیکیشن شما می بایست مجوزهای زیر را (در تگ uses-permission) در خود به صورت زیر داشته باشد.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="de.vogella.android.intent.implicit"
   android:versionCode="1"
   android:versionName="1.0" >
   <uses-sdk android:minSdkVersion="15" />
   <uses-permission android:name="android.permission.CALL_PRIVILEGED" >
   </uses-permission>
   <uses-permission android:name="android.permission.CALL_PHONE" >
   </uses-permission>
   <uses-permission android:name="android.permission.CAMERA" >
   </uses-permission>
   <uses-permission android:name="android.permission.READ_CONTACTS" >
   </uses-permission>
   <uses-permission android:name="android.permission.INTERNET"/>
   <application
     android:icon="@drawable/icon"
     android:label="@string/app_name" >
     <activity
       android:name=".CallIntentsActivity"
       android:label="@string/app_name" >
       <intent-filter>
         <action android:name="android.intent.action.MAIN" />
         <category android:name="android.intent.category.LAUNCHER" />
       </intent-filter>
```

```
    </activity>
  </application>
</manifest>
```

کد کلاس activity خود را به صورت زیر ویرایش نمایید.

Change your activity class to the following code.

```
package de.vogella.android.intent.implicit;
import android.app.Activity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Spinner;
import android.widget.Toast;
public class CallIntentsActivity extends Activity {
    private Spinner spinner;
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        spinner = (Spinner) findViewById(R.id.spinner);
        ArrayAdapter adapter = ArrayAdapter.createFromResource(this,
                R.array.intents, android.R.layout.simple_spinner_item);
        adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        spinner.setAdapter(adapter);
    }
    public void onClick(View view) {
        int position = spinner.getSelectedItemPosition();
        Intent intent = null;
        switch (position) {
        case 0:
            intent = new Intent(Intent.ACTION_VIEW,
                    Uri.parse("http://www.vogella.com"));
            break;
        case 1:
            intent = new Intent(Intent.ACTION_CALL,
                    Uri.parse("tel:(+49)12345789"));
            break;
        case 2:
            intent = new Intent(Intent.ACTION_DIAL,
                    Uri.parse("tel:(+49)12345789"));
            startActivity(intent);
            break;
        case 3:
            intent = new Intent(Intent.ACTION_VIEW,
                    Uri.parse("geo:50.123,7.1434?z=19"));
            break;
```

```
            case 4:
                intent = new Intent(Intent.ACTION_VIEW,
                        Uri.parse("geo:0,0?q=query"));
                break;
            case 5:
                intent = new Intent("android.media.action.IMAGE_CAPTURE");
                break;
            case 6:
                intent = new Intent(Intent.ACTION_VIEW,
                        Uri.parse("content://contacts/people/"));
                break;
            case 7:
                intent = new Intent(Intent.ACTION_EDIT,
                        Uri.parse("content://contacts/people/1"));
                break;

            }
            if (intent != null) {
                startActivity(intent);
            }
        }
        @Override
        public void onActivityResult(int requestCode, int resultCode, Intent data) {
            if (resultCode == Activity.RESULT_OK && requestCode == 0) {
                String result = data.toURI();
                Toast.makeText(this, result, Toast.LENGTH_LONG);
            }
        }
}
```

تست اپلیکیشن

با اجرای اپلیکیشن می بینید که یک لیست کشویی به برنامه اضافه شده که کلیک بر روی هر یک از آن ها به اجرا و راه اندازی activity های مربوطه منتهی می شود.

توجه: اگر بخاطر داشته باشید، شما هیچ اپلیکیشن مشخصی را به عنوان دریافت کننده ی intent به طور صریح مشخص نکردید (تنها عملیاتی که باید انجام شود را قید نمودید). این امر به شما اجازه می دهد تا task هایی با وابستگی کم (loosely coupled) تعریف نمایید که قابلیت استفاده از کامپونتت های اپلیکیشن های مختلف را فراهم می آورد (در این سناریو، سیستم اندروید تمامی کامپوننت هایی که برای این intent تخصیص یافته و قابلیت اجرای درخواست آن را دارند، شناسایی کرده و لیستی از کامپوننت های مزبور را برای کاربر لیست می کند. سپس کاربر تصمیم می گیرد activity را با کدام کامپوننت باز کند).

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

6-2-مدیریت چرخه ی حیات / lifecycle اپلیکیشن

مدیریت چرخه ی حیات/life cycle اپلیکیشن و activity در اندروید

این مبحث به شرح life cycle یا چرخه ی حیات اپلیکیشن و activity در اندروید می پردازد.

در شرایط ایده ال، کلیه ی اپلیکیشن هایی که کاربر در دستگاه اندروید خود راه اندازی می کند، باید در حافظه باقی بمانند. نا گفته پیدا است که این امر در تسریع راه اندازی مجدد اپلیکیشن بسیار موثر است. اما حقیقت امر این است که حافظه ی دستگاه اندروید همانند اندازه ی خود دستگاه، کوچک بوده و دارای ظرفیت محدود می باشد. به منظور مدیریت بهینه ی این منابع محدود سیستم اندروید این اجازه را دارد که فرایندهای در حال اجرا را متوقف کرده یا کامپوننت های اندروید را بازیابی (recycle) کند.

در صورتی که سیستم اندروید به این منابع نیاز داشته باشد، طبیعتا باید این منابع را آزاد کند. برای آزاد کردن این منابع از تعدادی قانون ساده پیروی می کند. در واقع هر فرایند در حال اجرا دارای اولویتی است که اندروید آن ها را بر اساس این اولویت از حافظه پاک می کند.

Table 1. Priorities

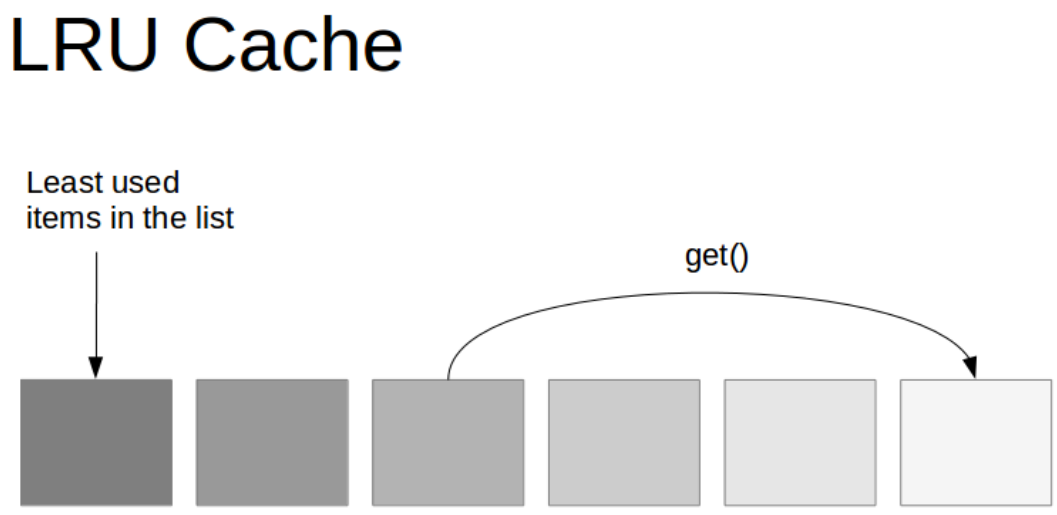
اولویت	شرح	وضعیت فرایند/ process status
1	اپلیکیشنی که کاربر در حال تعامل با activity آن می باشد یا دارای سرویسی است که به این activity متصل است. همچنین سرویسی که در حال اجرای یکی از متدهای مدیریت چرخه ی حیات (lifecycle method) بوده یا broadcast receiver ای که در حال فراخوانی و اجرای متد onReceive() خود می باشد.	پیش/Foreground زمینه

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Table 1. Priorities		
اولویت	شرح	وضعیت فرایند/ process status
2	کاربر دیگر با activity تعامل ندارد، اما activity هنوز (تا اندازه ای) قابل مشاهده است یا اپلیکیشن مورد نظر سرویسی دارد که توسط activity قابل مشاهده اما غیرفعال مورد استفاده قرار می گیرد.	قابل مشاهده/Visible
3	اپلیکیشن با یک سرویس در حال اجرا که ویژگی های 1 یا 2 را ندارد.	Service
4	اپلیکیشنی که تمامی activity های آن متوقف شده و هیچ سرویس یا دریافت کننده ای ندارد. سیستم اندروید تمامی آن ها را در یک لیست LRU نگه داشته و در صورت لزوم آن هایی که از تاریخ آخرین بار استفاده ی آن خیلی می گذرد یا کم ترین استفاده را داشته اند را از حافظه حذف می کند.	پس/Background زمینه
5	اپلیکیشنی که هیچ کامپوننت در حال اجرا و فعالی ندارد.	Empty

تمامی فرایندهایی که از اولویت 6 برخوردار هستند (در رده ی "Empty" قرار می گیرند) به لیست LRU اضافه می شوند و فرایندهایی که در ابتدای این لیست ها قرار دارند، توسط out of memory killer نابود می شوند (برای آزاد سازی منابع از حافظه حذف می شوند). حال چنانچه اپلیکیشنی توسط کاربر مجددا راه اندازی شد، آن اپلیکیشن به انتهای صف هدایت می شود و اولویت آن به 1 تغییر می یابد. این جریان در تصویر زیر به نمایش گذاشته شده است:



Calling get() for an item, moves it to the top of the cache

فراخوانی متد get() برای یک آیتم سبب می شود آن آیتم به انتهای حافظه ی LRU (cache) انتقال یابد.

7-2-آبجکت application

هر زمان که یک کامپوننت راه اندازی می شود، سیستم اندروید به صورت خودکار یک آبجکت application ایجاد می کند که این آبجکت تحت یک فرایند جدید با ID منحصربفرد و مختص به یک کاربر خاص ایجاد می شود. حتی اگر برنامه نویس آبجکت application را در فایل تنظیمات (AndroidManifest.xml)، در تگ application مشخص نکند، سیستم اندروید به صورت پیش فرض یک نمونه از کلاس Application (همان آبجکت application) ایجاد می کند (این آبجکت بیشتر برای نگهداری وضعیت و اطلاعات سراسری اپلیکیشن مورد استفاده قرار می گیرد).

آبجکت ذکر شده یک سری توابع مربوط به مدیریت lifecycle ارائه می دهد که در زیر به هریک همراه با عملکرد آن اشاره می کنیم:

- onCreate() – این متد قبل از اینکه اولین کامپوننت های اپلیکیشن راه اندازی شوند، فراخوانی می گردد. به عبارت دیگر این متد یک تابع callback است و هنگامی که activity برای نخستین بار اجرا می شود، فراخوانی می گردد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

- onLowMemory() – زمانی صدا خورده می شود که کل سیستم با کمبود حافظه مواجه شده و فرایندهای در حال اجرا یا فعال می بایست میزان استفاده ی خود از حافظه را کاهش دهند. سیستم اندروید با فراخوانی این تابع به اپلیکیشن دستور می دهد که باید حافظه را بهینه استفاده کرده و میزان استفاده ی خود را تا حد امکان محدود کند. شما این متد را بایستی زمانی پیاده سازی کنید که لازم باشد cache یا دیگر منابع غیر ضروری که برنامه اشغال کرده را آزاد سازی نمایید.
- onTrimMemory() – این متد زمانی صدا خورده می شود که سیستم عامل احساس کند که فرایند باید میزان استفاده ی خود از حافظه را کاهش دهد. این اتفاق (برای مثال) زمانی می افتد که فرایند در وضعیت background قرار گرفته باشد و در این میان حافظه ی کافی برای فعال نگه داشتن تعداد دلخواه فرایند پس زمینه ای وجود ندارد. به عبارت دیگر، این متد زمانی فراخوانده می شود که سیستم اندروید از اپلیکیشن درخواست استفاده ی بهینه از حافظه/کاهش استفاده از آن را داشته باشد. این متد پارامترهایی دارد که از وضعیت جاری اپلیکیشن خبر می دهند. برای مثال، ثابت TRIM_MEMORY_MODERATE بیانگر این است که فرایند در اواسط لیست LRU قرار دارد. در این شرایط طبیعتا آزاد سازی حافظه به سیستم کمک می کند تا دیگر فرایندهای در حال اجرا را جهت بهبود کارایی کلی سیستم، در انتهای لیست مذکور نگه دارد.
- onTerminate() – تنها به منظور تست در محیط شبیه ساز بکار رفته و به هیچ وجه در بستر اجرا نهایی (محیط production) اپلیکیشن فراخوانی نمی شود.
- onConfigurationChanged() – این متد زمانی فراخوانده می شود که کامپوننت شما در حال اجرا است و در این حین تنظیمات و configuration دستگاه تغییر می کند.

لازم به توضیح است که آبجکت application قبل از راه اندازی هر کامپوننتی ایجاد شده و حداقل تا زمانی که کامپوننت دیگری از اپلیکیشن راه اندازی نشده، به اجرا ادامه می دهد.

LifeCycle-2-8/چرخه ی حیات content provider

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

هر زمان که content provider مورد دسترسی قرار می گیرد، این کامپوننت هیچگاه به صورت انفرادی و جداگانه متوقف نمی شود. به بیانی روشنتر این کامپوننت تنها زمانی کاملا متوقف می شود که کل فرایند اپلیکیشن خاتمه یابد.

Lifecycle-9-2/چرخه ی حیات Activity

1-9-2-وضعیت های مختلف یک activity

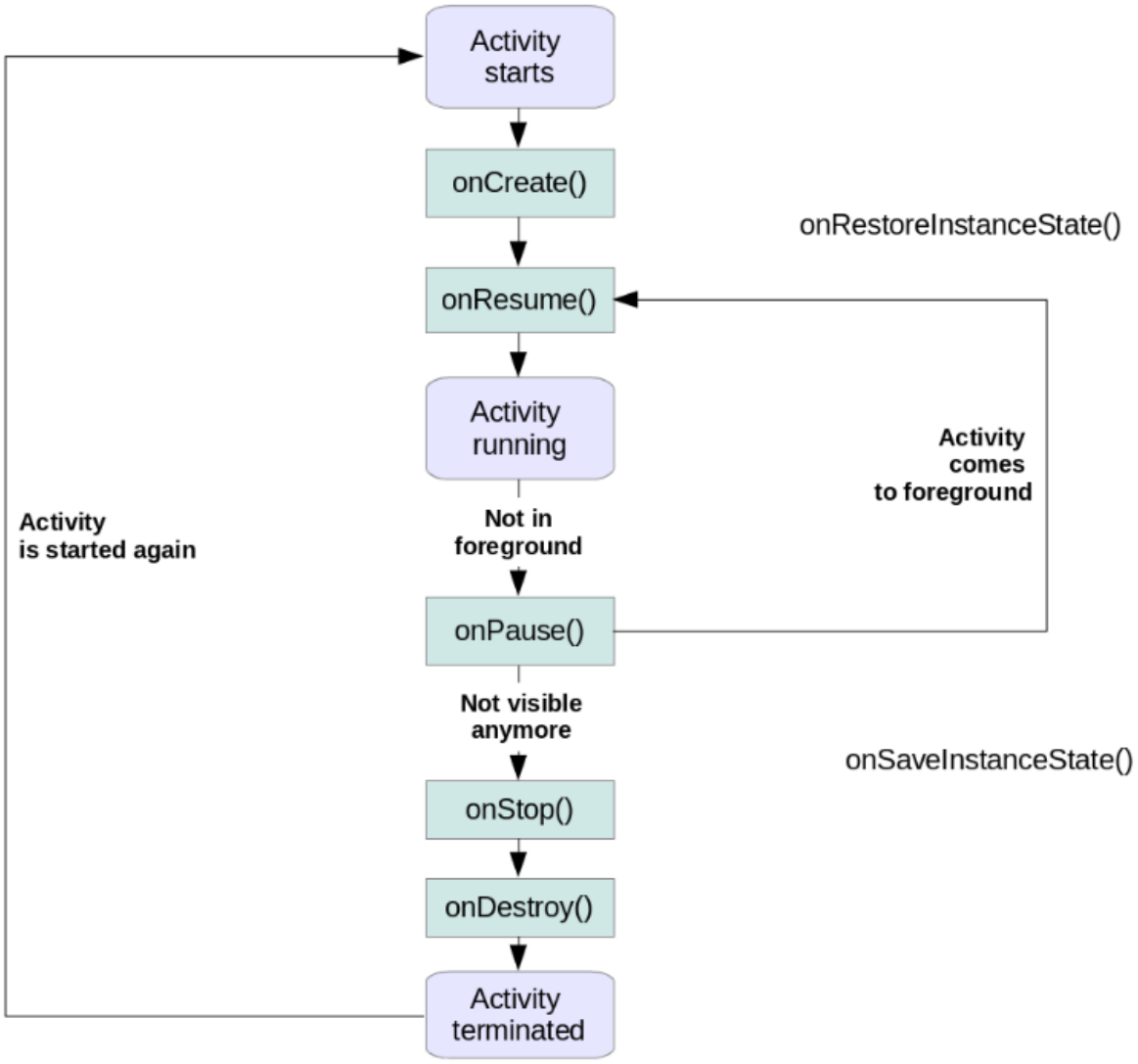
یک Activity می تواند بسته به نحوه ی تعاملی که کاربر با آن دارد، در وضعیت های متفاوت قرار داشته باشد. این وضعیت ها در جدول زیر تشریح شده است:

Table 2. Activity state	
وضعیت/state	شرح
Running (در حال اجرا و قابل تعامل مستقیم)	در این وضعیت، activity کاملا قابل مشاهده بوده و کاربر می تواند با آن مستقیما تعامل داشته باشد.
Paused (متوقف شده اما همچنان قابل رویت)	Activity is still visible but partially obscured, instance is running but might be killed by the system. در این وضعیت، Activity همچنان قابل مشاهده است ، اما نه به طور کامل و به اصطلاح دیگر در foreground قرار ندارد. در واقع نمونه در حال اجرا است، اما ممکن است سیستم اندروید آن را از حافظه پاک کند (kill).
Stopped (متوقف شده و غیر قابل رویت)	Activity is not visible, instance is running but might be killed by the system. Activity قابل مشاهده نیست، نمونه در حال اجرا است اما ممکن است توسط سیستم از حافظه پاک شود.
Killed (کاملا نابود یا به اصطلاح از حافظه پاک شده است)	Activity has been terminated by the system of by a call to its finish() method. در این وضعیت سیستم اندروید متد finish() را صدا زده و activity مورد نظر در پی اجرای شدن این متد از حافظه پاک شده است.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در نمودار زیر چرخه ی حیات یک activity را به همراه متدهای مهم آن مشاهده می کنید.



لازم به ذکر است که اندروید متدهای دیگری نیز جهت چرخه ی حیات در اختیار دارد، اما موارد استفاده از آن ها نادر است. به عنوان مثال می توان به متد onDestroy() اشاره کرد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-9-2-پایان یافتن activity ها توسط سیستم عامل

سیستم اندروید این اجازه را دارد که در راستای بهبود کارایی و بهینه سازی سرعت، با recycle کردن activity ها (استفاده ی مجدد)، منابعی را آزاد کند. گفته می شود که سیستم عامل اندروید می تواند activity ها را فارغ از کل فرایند اپلیکیشن (به صورت منفرد) خاتمه دهد، در حالی که واقعیت چیز دیگری است. سیستم اندروید هیچگاه activity های فردی را recycle نمی کند، بلکه کل فرایند را بر اساس قوانین مشخص پایان می دهد.

توجه: این یک باور غلط است که اندروید activity های منفرد را از حافظه پاک می کند. متاسفانه این توضیح اشتباه در سایت رسمی توسعه اندروید نیز ذکر شده است. اما بنا به گفته ی Dianne Hackborn که عضو تیم طراحان و توسعه دهندگان اندروید، بخش پیاده سازی out of memory killer هست، اندروید هیچگاه activity ها را به صورت منفرد خاتمه نمی دهد، بلکه کل فرایند میزبان activity ها را پایان می دهد.

توجه: یک activity برای اینکه اطلاعات و وضعیت خود را به درستی ذخیره کرده و بعده ها بتواند آن را بازیابی کند، ناگذیر است اطلاعات state را در زمان مناسب ذخیره کند. همچنین، در صورتی که activity های غیر ضروری دیگر قابل مشاهده یا دسترسی نبودند، می بایست عملیات غیر ضروری را به منظور صرفه جویی در منابع سیستم متوقف کند.

برای متوقف کردن listener های framework (گوش فراخوان ها) و بروز رسانی های UI، می بایست متد onPause() را فراخوانی کرده و سپس با استفاده از متد onStop() داده های اپلیکیشن را ذخیره نمایید. دو تابع نام برده حتما پیش از خاتمه یافتن activity صدا خورده می شوند. با فراخوانی onResume() می توانید دوباره listener ها را تخصیص داده و در صورت لزوم، بروز رسانی های UI را بر اساس داده های ذخیره شده از سر بگیرید (مجددا راه اندازی نمایید).

علاوه بر مدیریت منابع (resource management)، اندروید می بایست activity ها را در صورت رخداد تغییر در تنظیمات دستگاه، از نو ایجاد کند. آبجکت Configuration دربردارنده ی تنظیمات و

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

پیکربندی جاری دستگاه می باشد. اگر این تنظیمات تغییر کند، activity های مجددا راه اندازی می شوند چرا که لازم است برای این تنظیمات جدید از منابع/resource متفاوت استفاده کنند.

10-2-داده ها و اطلاعات ذخیره شده از آبجکت activity جهت بازگردانی activity به وضعیت قبلی/Activity instance state

همان طور که در بالا ذکر شد، هنگامی تنظیمات دستگاه تغییر می کند، مثلا وضعیت و جهت نمایش تغییر می یابد، activity برای اینکه متناسب با تنظیمات جدید نمایش داده شود، باید مقادیر و منابع متفاوتی را بخواند و از اینرو از نو ساخته می شود (ابتدا متدonDestroy() صدا خورده و بلافاصله متد onCreate() فراخوانی می شود). حال به منظور اینکه سیستم اندروید بتواند activity را با تنظیمات جدید، اما وضعیت قبلی بازگردانی کند، ناچار نمونه ای از آن کلاس را ذخیره می کند. سپس بر اساس داده های ذخیره شده، activity را به حالت قبلی بازمی گرداند. Instance state در واقع داده های موقتی و غیر ماندگار هستند که باید در بازه ی زمانی تغییر بین دو وضعیت نمایش (برای مثال بخاطر تغییر در تنظیمات دستگاه) بین activity ها (نمونه ی ذخیره شده از activity و نمونه ی مجدد ایجاد شده) پاس داده شود تا اطلاعات و انتخاب های کاربر (اینک با تنظیمات جدید) به حالت قبل بازگردانی شود. این اپلیکیشن است که وظیفه ی بازگردانی خود به حالت قبلی را دارد.

Instance state معمولا جفت های کلید-مقدار هستند که از activity خاتمه یافته، در آبجکت Bundle ذخیره می شوند و جهت بازگردانی وضعیت activity به حالت قبلی (وضعیتی که کاربر در آن اپلیکیشن را ترک کرد) مورد استفاده قرار می گیرند.

سناریویی را در نظر بگیرید که کاربر با نوار پیمایش به پایین یک لیست می رود (ListView) که در آن هزاران آیتم وجود دارد و در این میان activity حذف و مجددا ساخته می شود. طبیعتا کاربر دوست ندارد موقعیت جاری خود را در این لیست بسیار بزرگ از دست بدهد و دوباره مجبور به پیدا کردن آن شود. از اینرو موقعیت جاری کاربر باید حفظ شده و پس از ایجاد activity دوباره کاربر به آن بازگردانده شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

برای این منظور متد onSaveInstanceState() صدا خورده شده و وضعیت نمونه را در قالب یک آبجکت Bundle در خود ذخیره می کند. آبجکت Bundle می تواند انواع داده ی اولیه، آرایه، رشته/String و Object هایی از جنس Parcelable و Serialisable را در خود ذخیره کند.

داده های ذخیره شده در کلاس Bundle، در زمان راه اندازی مجدد activity مورد نظر به عنوان پارامتر به متدهای onCreate() و onRestoreInstanceState() ارسال می شوند.

در صورت بازنویسی (override) پیاده سازی توابع onSaveInstanceState() و onRestoreInstanceState()، لازم است پیاده سازی کلاس پدر (super implementation) را نیز فراخوانی نمایید زیرا view های پیش فرض اندروید داده های خود را به واسطه ی فراخوانی View.onSaveInstanceState از متد onSaveInstanceState() ذخیره می کنند. برای مثال، EditText محتوای خود را از طریق فراخوانی پیش فرض این متد ذخیره می کند.

با استفاده از دو تابع onRestoreInstanceState() و onCreate() می توانید یک instance state activity را (در صورتی که activity نابود شده و مجددا ساخته شد) بازسازی نمایید.

توجه: برای بازگردانی وضعیت activity (بازیابی instance state) توصیه می کنیم از متد onRestoreInstanceState() استفاده نمایید. این روش تنظیم و راه اندازی اولیه activity را از بازگردانی وضعیت آن به حالت قبلی (instance state) جدا می سازد.

اگر کاربر با activity تعامل داشته و سپس دکمه ی Back را فشار دهد یا اینکه متد finish() آن activity صدا خورده شود، activity مورد نظر از پشته ی جاری activity ها حذف شده و recycle می شود. در چنین شرایطی، هیچ instance state (داده ی ذخیره شده ای از وضعیت activity وجود ندارد که بخواهد بازگردانی شود) برای بازگردانی وجود ندارد، از اینرو متد onSaveInstanceState() هیچگاه فراخوانی نمی شود.

حال چنانچه کاربر با activity تعامل داشته و بعد دکمه ی Home را فشار دهد، اطلاعات مربوط به وضعیتactivity / activity instance می بایست ذخیره شود. در پی این رخداد، متد

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

onSaveInstanceState() فراخوانده می شود و اطلاعات مربوط به وضعیت activity را در خود ذخیره می کند. اکنون اگر کاربر به اپلیکیشن بازگردد یا به عبارتی آن را مجددا راه اندازی کند، سیستم عامل اندروید آخرین activity که هنگام ترک اپلیکیشن فعال و در حال اجرا بود را فراخوانی می کند. در واقع زمانی که activity مجددا راه اندازی می شود، آبجکت Bundle (حامل داده های ذخیره شده از وضعیت activity) در اختیار متدهای onRestoreInstanceState() و onCreate() قرار می گیرد.

توجه: در صورتی که کاربر دکمه ی بازگشت را فشار دهد، متد onSaveInstanceState() صدا زده نمی شود. توجه داشته باشید که از این روش نباید به هیچ وجه برای ذخیره ی داده هایی که می بایست ماندگار شوند، استفاده نمایید.

2-11-آبجکت های جاوایی که در صورت تغییر در تنظیمات دستگاه باید بین نمونه های activity پاس داده شوند

در صورتی که تنظیمات دستگاه (configuration) حین اجرای activity تغییر کند، آن activity از بین رفته و مجددا جهت تطبیق خود با شرایط جدید و بارگذاری منابع جدید، مجددا ایجاد می شود. سیستم اطلاعات activity را جهت بازگردانی وضعیت اپلیکیشن به قبل ذخیره کرده و با راه اندازی مجدد activity، به این نمونه (activity) پاس داده می شود. این اطلاعات که از activity قبلی ذخیره شده و به activity جدید پاس داده می شود، در واقع داخل آبجکت های جاوایی ذخیره می گردد. در ویرایش های قبلی اندروید ذخیره و بازگردانی این آبجکت ها توسط getLastNonConfigurationInstance() و onRetainNonConfigurationInstance() قابل دسترسی و بازیابی بودند. اما این دو متد دیگر منسوخ هستند و در API های جدید بهتر است برای ذخیره ی آبجکت هایی که می بایست بر اثر تغییر در تنظیمات یا کانفیگ بین نمونه های کلاس activity رد و بدل شوند، از headless fragment ها استفاده نمایید.

توجه: اگر از متد onRetainNonConfigurationInstance() استفاده می کنید، در آن صورت نباید آبجکتی از activity را به نمونه ی بعدی آن (که خود از قبل اشاره گری به آبجکت اولیه activity دارد) ارسال نمایید. چرا که این کار هدر رفتن حافظه (memory leak) را به دنبال دارد. بدین معنی

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

که garbage collector قادر نخواهد بود نمونه ای از activity را که هنوز اشاره گری به آن در آبجکت جدید وجود دارد، حذف نموده و حافظه را آزاد کند.

12-2-مدیریت configuration و تنظیمات

1-12-2-جلوگیری از نابود و مجدد ایجاد شدن activity بر اثر تغییر در تنظیمات و وضعیت برنامه

همان طور که در بالا ذکر شد، activity بر اثر تغییر در تنظیمات و وضعیت دستگاه (config change) از بین رفته و جهت تطبیق با تنظیمات جدید و همچنین بارگذاری resource های جدید، دوباره ساخته می شود.

تغییر در تنظیمات ممکن است به دنبال یک event که از خود دستگاه اندروید اعلان یا صادر می شود و مربوط به برنامه ی جاری می باشد، اتفاق بیافتد.

اطلاعات مربوط به تنظیمات جاری دستگاه در نمونه ای از کلاس Configuration ذخیره می شود. در واقع Configuration یک کلاس است که تمامی اطلاعات مربوط به تنظیمات جاری دستگاه را تعریف می کند و این اطلاعات می توانند منابع و محتوای مورد نیاز اپلیکیشن را تحت تاثیر قرار دهند. کلاس مزبور هم می تواند تنظیمات تعریف شده توسط کاربر (همچون locale list و scaling = مقیاس بندی به صورت پویا) را شامل شود و هم تنظیمات دستگاه نظیر حالت دریافت ورودی (input mode)، اندازه ی صفحه یا وضعیت و جهت نمایش.

به عنوان مثال زمانی که کاربر با چرخاندن دستگاه، وضعیت نمایش و چیدمان را تغییر می دهد (از نمای عمودی به افقی تغییر می دهد)، در این هنگام سیستم اندروید متوجه می شود که activity جهت تطبیق با شرایط جاری، لازم به بارگذاری منابع جدید دارد. از اینرو activity کنونی را نابود کرده و بار دیگر با تنظیمات جدید آن راه اندازی می کند.

هنگامی که activity نابود و مجدد ساخته می شود، برنامه نویس می بایست اطمینان حاصل کند که نمونه ی جدید activity با اطلاعات مربوط به وضعیت قبلی ایجاد و بارگذاری شود. سیستم

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

اندروید ابزار و روش های مختلفی برای این منظور ارائه می دهد. در محیط emulator می توانید با فشردن کلیدهای Ctrk+F11 به راحتی تغییر در وضعیت و جهت نمایش را شبیه سازی نمایید.

می توانید activity را طوری در فایل XML تعریف کنید که نسبت به برخی از تغییرات در وضعیت و تنظیمات بی تفاوت باشد و به تبع از راه اندازی مجدد activity در موارد مزبور جلوگیری نمایید. کافی است در فایل تنظیمات AndroidManifest.xml خود، داخل تگ activity، خصیصه configChanges (attribute) را با مقدار مربوطه تنظیم نمایید.

Activity زیر طوری تنظیم شده که در صورت رخداد تغییر در وضعیت نمایش یا موقعیت صفحه کلید (قابل مشاهده/پنهان)، از نو راه اندازی نشود و به بیانی دیگر نسبت به این تغییرات بی تفاوت باشد.

```
<activity android:name=".ProgressTestActivity"
   android:label="@string/app_name"
   android:configChanges="orientation|keyboardHidden|keyboard">
</activity>
```

توجه: بهتر است برای مدیریت تغییر در تنظیمات، بجای attribute فوق از روش های دیگر نظیر فریم ورک loader یا fragment های فاقد UI (headless fragments) استفاده نمایید. در آینده به شرح مفاهیم نام برده خواهیم پرداخت.

2-12-2-ثابت کردن وضعیت نمایش (orientation) یک activity

این امکان برای شما وجود دارد که activity را در فایل تنظیمات AndroidManifest.xml طوری تعریف کنید که فقط در یک وضعیت یا نمای (orientation) خاص نمایش داده شود. برای مثال فایل XML زیر یک activity تعریف کرده که تنها در حالت و نمای افقی (landscape) به کاربر نشان داده می شود.

```
<activity
  android:name="com.vogella.android.multitouch.MainActivity"
  android:label="@string/app_name"
  android:screenOrientation="landscape" >
  <intent-filter>
    <action android:name="android.intent.action.MAIN" />
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

تمرین: بررسی Lifecycle

ایجاد پروژه

یک پروژه ی جدید به نام com.vogella.android.lifecycle.activity ایجاد کنید.

کلاسی با پیاده سازی زیر تعریف کنید که از طریق notification ها، رخدادها یا event های مربوط به چرخه ی حیات را گزارش می کند.

```
package com.vogella.android.lifecycle.activity;
import android.app.Activity;
import android.app.Notification;
import android.app.NotificationManager;
import android.os.Bundle;
public class TracerActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        notify("onCreate");
    }
    @Override
    protected void onPause() {
        super.onPause();
        notify("onPause");
    }
    @Override
    protected void onResume() {
        super.onResume();
        notify("onResume");
    }
    @Override
    protected void onStop() {
        super.onStop();
        notify("onStop");
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
        notify("onDestroy");
    }
    @Override
    protected void onRestoreInstanceState(Bundle savedInstanceState) {
        super.onRestoreInstanceState(savedInstanceState);
```

```
        notify("onRestoreInstanceState");
    }
    @Override
    protected void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
        notify("onSaveInstanceState");
    }
    private void notify(String methodName) {
        String name = this.getClass().getName();
        String[] strings = name.split("\\.");
        Notification noti = new Notification.Builder(this)
         .setContentTitle(methodName + " " + strings[strings.length - 1]).setAutoCancel(true)
         .setSmallIcon(R.drawable.ic_launcher)
         .setContentText(name).build();
        NotificationManager notificationManager =
                (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
        notificationManager.notify((int) System.currentTimeMillis(), noti);
    }
}
```

3-12-2-ایجاد activity های مورد نیاز

دو activity دیگر ایجاد کنید که activity بالا را به ارث می برند. Activity اول باید از طریق یک intent، اکتیویتی دوم را راه اندازی کند.

```
package com.vogella.android.lifecycle.activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
public class MainActivity extends TracerActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void onClick(View view) {
        Intent intent = new Intent(this, SecondActivity.class);
        startActivity(intent);
    }
}
```

```
package com.vogella.android.lifecycle.activity;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
public class SecondActivity extends TracerActivity {
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
    }
}
```

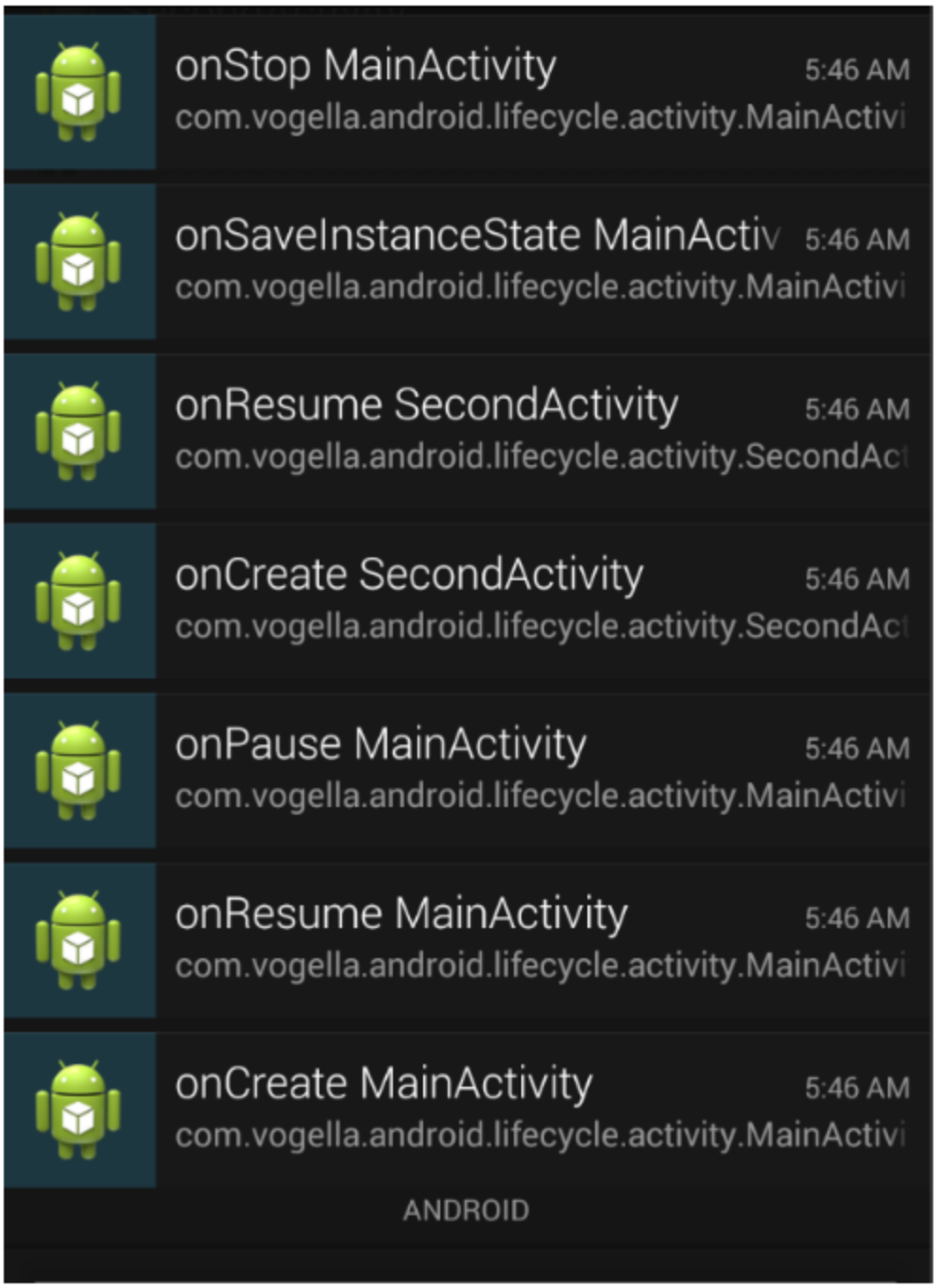
توجه: بایستی هر دو activity را در فایل تنظیمات (manifest) تعریف نمایید.

تست اپلیکیشن

اپلیکیشن خود را اجرا کرده و اطمینان حاصل کنید همه چیز طبق برنامه اتفاق می افتد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



پس از راه یابی به activity دوم و مشاهده ی محتویات آن، دکمه ی بازگشت را در این activity (صفحه ی دوم اپلیکیشن) فشار دهید. خواهید دید که متد onSaveInstanceState() فراخوانی نشده و به تبع آن وضعیت activity جاری ذخیره نمی شود. می توانید تشریح کنید چرا این متد صدا خورده نمی شود!

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

حال در activity دوم یا همان صفحه ی دوم اپلیکیشن، دکمه ی home را فشار دهید. خواهید دید که به دنبال این رخداد، متد onSaveInstanceState() صدا خورده شده و وضعیت activity جاری ذخیره می شود. علت فراخوانده شدن این متد را شرح دهید.

حال activity یا صفحه ی دوم اپلیکیشن را اجرا نمایید. جهت نمایش (orientation) محیط شبیه ساز را تغییر داده و دقت کنید که کدام متدهای مربوط به lifecycle صدا زده می شوند. آیا activity اول نیز دوباره ساخته می شود یا فقط activity دوم مجددا راه اندازی می شود.

گزینه ی Don't keep activities را در بخش تنظیمات مربوط به توسعه دهنده (Developer Options) فعال نمایید. بار دیگر دقت کنید کدام متدها فراخوانده می شوند.

4-12-2-بازگردانی وضعیت نمونه activity

ابتدا آرایه ای از نوع رشته تعریف کرده و سپس یک آبجکت Spinner (لیست کشویی) به activity اول خود اضافه نمایید که مقادیر آن از این آرایه پر می شود. در کد XML زیر، فایل strings.xml و layout مورد استفاده ی activity اول را مشاهده می کنید.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Lifecycle</string>
    <string name="action_settings">Settings</string>
    <string name="hello_world">Hello world!</string>
    <string-array name="operating_systems">
        <item >Ubuntu</item>
        <item >Android</item>
        <item >iOS</item>
    </string-array>
</resources>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >
    <Spinner
```

```
    android:id="@+id/spinner"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginRight="58dp"
    android:entries="@array/operating_systems" />
  <Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="onClick"
    android:layout_gravity="bottom"
    android:text="Start new Activity" />
</LinearLayout>
```

اندروید وضعیت و اطلاعات مربوط به آبجکت Spinner یا همان لیست کشویی را به صورت خودکار بازگردانی می کند. در واقع سیستم اندروید خود وضعیت جاری آبجکت مزبور (همچون انتخاب کاربر) را ذخیره کرده و در صورت تغییر در تنظیمات اپلیکیشن یا دستگاه و همچنین راه اندازی مجدد activity ها (نابود شدن activity و بازسازی آن)، آن را به وضعیت قبلی بازگردانی می کند.

حال مقادیر ثابت آرایه در فایل layout را حذف کرده و آن را در کد برنامه (از طریق source code) به آبجکت Spinner تخصیص دهید.

```
// configure the spinner in code
Spinner spinner = (Spinner) findViewById(R.id.spinner);
String[] values = getResources().getStringArray(R.array.operating_systems);
ArrayAdapter<String> adapter =
    new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, values);
spinner.setAdapter(adapter);
```

مقادیر تخصیص یافته ی ثابت را از فایل layout خود نیز حذف نمایید.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:id="@+id/LinearLayout1"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  android:paddingBottom="@dimen/activity_vertical_margin"
  android:paddingLeft="@dimen/activity_horizontal_margin"
  android:paddingRight="@dimen/activity_horizontal_margin"
  android:paddingTop="@dimen/activity_vertical_margin"
  tools:context=".MainActivity" >
  <Spinner
    android:id="@+id/spinner"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
```

```
        android:layout_marginRight="58dp"
        />
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="onClick"
        android:layout_gravity="bottom"
        android:text="Start new Activity" />
</LinearLayout>
```

اطمینان حاصل نمایید که وضعیت آبجکت Spinner همچنان مانند قبل به صورت خودکار بازگردانی می شود.

13-2-مبحث امنیت و مجوزهای دسترسی در اندروید (Security&Permission)

13-2-1-مفهوم امنیت در اندروید

سیستم اندروید به هنگام نصب یک اپلیکیشن بر روی دستگاه، یک GID (شناسه ی یکتا که به گروه کاربری خاص تخصیص می یابد) و UID (شناسه ی یگانه ی کاربری) به آن اپلیکیشن اختصاص می دهد. در واقع تمامی فایل های یک آپلیکیشن در انحصار این کاربر یا به اصطلاح private هستند و به تبع دیگر اپلیکیشن ها اجازه ی دسترسی به این فایل ها را ندارند. علاوه بر آن، هر اپلیکیشن اندروید تحت فرایند اختصاصی خود اجرا و راه اندازی می شود. از اینرو، به واسطه ی این ویژگی هسته ی Linux، هر اپلیکیشن اندرویدی از دیگر برنامه های کاربردی در حال اجرا کاملا مستقل و مجزا است. در صورتی هم که یک اپلیکیشن اندروید می بایست بر حسب نیاز اطلاعاتی را با یک اپلیکیشن دیگر رد و بدل کرده و آن ها را به اشتراک بگذارد، این اتفاق بایستی به صورت صریح و توسط یک کامپوننت نرم افزاری که وظیفه ی آن انتقال داده می باشد همچون یک service یا content provider صورت گیرد.

13-2-2-مفهوم مجوزهای دسترسی در اندروید

اندروید از یک سیستم تخصیص مجوز برخوردار است که ویژه ی هر task یا عملیات مجوزها و سطوح دسترسی خاصی معین می کند. در حقیقت هر اپلیکیشن می تواند جهت انجام عملیات خاص، مجوزهای لازم را از این سامانه درخواست نماید. برای مثال، یک برنامه ی کاربری می تواند از سیستم تخصیص مجوز، درخواست دسترسی به اینترنت را داشته باشد.

سیستم مزبور سطوح مختلفی برای مجوزها دارد که به آن در اصطلاح protection level گفته می شود.

لازم به توضیح است که مفهوم مجوز از کتابخانه های اندروید، نسخه ی 23 (API level) تغییر کرده است. قبل از ویرایش مذکور، یک اپلیکیشن حین نصب مجوزهای مورد نیاز و درخواست خود را به

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

کاربر اعلان می کرد. اما از API 23 به بعد دیگر تنها در زمان اجرای برنامه مجوزهای لازم را از کاربر درخواست می کند.

اپلیکیشن های اندروید مجوزهای مورد نیاز خود را در فایل تنظیمات اندروید (manifest) اعلان می کنند. البته اپلیکیشن می تواند مجوزهای اضافی در این فایل تعریف کند که دسترسی به برخی کامپوننت های نرم افزار را محدود می سازند.

3-13-2-سیستم مدیریت و تخصیص مجوز در اندروید پیش از API 23

قبل از ویرایش 23 کتابخانه های اندروید، مجوزهای مورد نیاز قبل از نصب اپلیکیشن از کاربر درخواست می شدند. کاربر تصمیم می گرفت که آیا مجوزهای مربوطه باید به برنامه داده شوند یا خیر. چنانچه کاربر مجوز مورد درخواست اپلیکیشن را تایید نمی کرد، در آن صورت اپلیکیشن مورد نظر به هیچ وجه بر روی دستگاه نصب نمی شد. اگر مجوز لازم را حین نصب به اپلیکیشن می داد، اپلیکیشن به بخش های مد نظر خود دسترسی پیدا می کرد و دیگر کاربر نمی توانست این امر را (پس از اتمام نصب برنامه) تغییر دهد.

4-13-2-سیستم تخصیص مجوز از API 23 به بعد (اعطای مجوز به هنگام اجرا برنامه)

ویرایش 6.0 سیستم عامل اندروید، Marshmallow، یک permission model جدید را معرفی کرد که طی آن مجوزهای لازم، در زمان اجرای اپلیکیشن از کاربر درخواست می شود. بنابراین اگر اپلیکیشنی که در دست توسعه دارید، ویژه ی این نسخه طراحی شده (targetsdkversion آن را بر روی 23 تنظیم کرده اید)، لازم است از مدل تخصیص مجوز جدید استفاده نمایید.

در اندروید دو سطح دسترسی و امنیت به شرح زیر تعریف شده است:

- Normal: در این سطح، مجوزهایی درخواست می شود که به اطلاعات، فایل ها شخصی کاربر یا فعالیت و عملیات دیگر اپلیکیشن ها لطمه ای وارد نمی کند (به بخش های حساس

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

دسترسی ندارد). به عنوان مثال می توان به مجوز تنظیم منطقه ی زمانی اشاره کرد. مجوزهایی که در این سطح هستند معمولا به صورت خودکار به اپلیکیشن اعطا می شوند.

- Dangerous: مجوزهای مربوط به این سطح، اغلب با اطلاعات شخصی کاربر سروکار دارند و حتی ممکن است این اطلاعات یا فرایندهای مربوط به اپلیکیشن های دیگر را دستکاری کنند. به عنوان مثال می توان به اجازه ی دسترسی به اطلاعات مخاطبین کاربر اشاره کرد. این دست از مجوزها باید در زمان اجرای اپلیکیشن از کاربر درخواست می شوند.

برای بررسی اینکه آیا مجوز خاصی به اپلیکیشن داده شده یا خیر، می توان از متد checkSelfPermission استفاده نمود.

```
// called in a standard activity, use  ContextCompat.checkSelfPermission for AppCompActivity
int permissionCheck = checkSelfPermission(this, Manifest.permission.WRITE_CALENDAR);
if (!permissionCheck == PackageManager.PERMISSION_GRANTED) {
  // User may have declined earlier, ask Android if we should show him a reason
   if (shouldShowRequestPermissionRationale(thisActivity, Manifest.permission.WRITE_CALENDAR)) {
      // show an explanation to the user
      // Good practise: don't block thread after the user sees the explanation, try again to request the
permission.
   } else {
      // request the permission.
      // CALLBACK_NUMBER is a integer constants
      requestPermissions(thisActivity,  new String[]{Manifest.permission.WRITE_CALENDAR},
CALLBACK_NUMBER);
      // The callback method gets the result of the request.
   }
} else {
// got permission use it
}
```

هنگامی که از کاربر درخواست اعطای مجوز می شود، شما یک callback دریافت می کنید.

```
@Override
public void onRequestPermissionsResult(int requestCode, String permissions[], int[] grantResults) {
   switch (requestCode) {
      case MY_PERMISSIONS_REQUEST_READ_CONTACTS: {
         // If request is cancelled, the result arrays are empty.
         if (grantResults.length > 0
            && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
            // permission was granted, do your work....
         } else {
            // permission denied
            // Disable the functionality that depends on this permission.
         }
```

```
            return;
        }
        // other 'case' statements for other permssions
    }
}
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

14-2-پیاده سازی list یا grid جهت نمایش آیتم ها در اندروید

پیاده سازی list (نمای فهرستی) و grid (نمای جدولی و خانه بندی شده همراه با سطر و ستون) در اپلیکیشن های اندروید به وسیله ی کلاس/ویجت RecyclerView

این آموزش نحوه ی استفاده از widget/کامپوننت رابط کاربری RecyclerView (جهت پیاده سازی لیست های پیچیده) در اپلیکیشن های مبتنی بر اندروید را به شما آموزش می دهد.

14-2-1-استفاده از list یا grid در اپلیکیشن اندروید

نمایش آیتم ها در قالب یک list (با نمای فهرستی) یا grid (با نمای جدولی و به صورت خانه بندی شده) در اپلیکیشن های تحت موبایل یک الگوی کاملا معمول و پرکاربرد است که به فور شاهد آن هستید. کاربر در مقابل خود مجموعه ای از آیتم های مرتب و سازماندهی شده را می بیند که به راحتی به وسیله ی یک نوار اسکرول قابل پیمایش می باشد. حال این مجموعه می تواند یک list، grid یا هر ساختار سازمان دهی شده ی دیگری از داده ها باشد. تصویر زیر یک activity با این نما را به نمایش می گذارد.

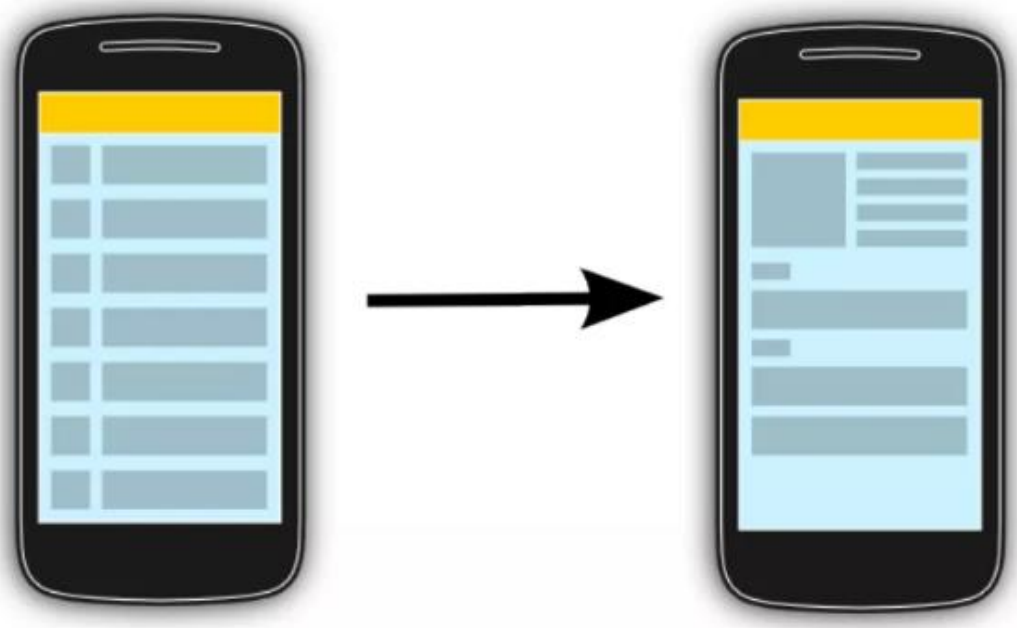
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

کاربر با این مجموعه از آیتم ها از طریق event هایی که با لمس نمایشگر اتفاق می افتند (touch event) یا نوارابزار (toolbar) تعامل می کند. هر یک از این المان ها می تواند برای کاربر قابل انتخاب باشد. در پی کلیک کاربر بر روی یک المان، ممکن است toolbar بروز رسانی شود یا به طور کلی یک صفحه ی جدید با جزئیات فراوان به نمایش در آید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



2-14-2-استفاده از RecyclerView جهت پیاده سازی لیست های پیچیده

RecyclerView یک کلاس است که به برنامه نویس قابلیت پیاده سازی لیست های پیچیده را می دهد. این لیست می تواند حجم بالایی از اطلاعات را نمایش داده که ممکن است در زمان اجرای برنامه (بر اساس افعال کاربر) بارها تغییر کنند.

RecyclerView در واقع نسخه ی جدیدی از ListView و GridView است که توسط چارچوب نرم افزاری (framework) اندروید در اختیار توسعه دهنده قرار می گیرد. Recycler view تعدادی زیادی از مشکلات و کاستی هایی که در widget های قبلی وجود داشت را برطرف نموده و سبکی از برنامه نویسی را اعمال می کند که افزایش کارایی و سرعت در اجرای اپلیکیشن را به دنبال دارد. علاوه بر موارد ذکر شده، این کلاس همراه با انیمیشن های پیش فرض و از قبل تعریف شده ارائه می گردد که المان ها را با انیمیشن از لیست حذف کرده یا به آن اضافه می کند.

RecyclerView به شما اجازه می دهد تا از layout manager های مختلف برای مدیریت چیدمان و موقعیت دهی آیتم ها استفاده نمایید. RecyclerView برای پیاده سازی لیست و نمایش آیتم ها با سه کلاس همکاری دارد. ViewHolder که وظیفه ی نگهداری یک المان در لیست را برعهده دارد

(تمامی المان هایی که در لیست مشاهده می کنید، اشاره گر یا متغیری متناظر در آبجکت view holder دارند). دومین المان، کلاس adapter هست که view یا همان المانی که در لیست قرار می گیرد را ساخته، آن را در viewholder قرار می دهد و در نهایت داده ها را به هر view متصل یا bind می کند. و اما آخرین المان LayoutManager هست که چگونگی چیدمان و نمایش المان ها در لیست را مدیریت می کند (لیست را با نمای افقی، عمودی، به صورت جدول بندی شده نمایش دهد).

ViewHolder یک کلاس static و درون ساخته در adapter است که اشاره گری (reference و متغیری متناظر) به view مربوطه نزد خود نگه می دارد. با نگه داشتن اشاره گر به این view ها در کلاس مزبور، توسعه دهنده می تواند از فراخوانی مکرر متد findViewById() جهت بروز رسانی widget ها با داده های جدید خودداری کند (که در نهایت صرفه جویی در منابع و افزایش کارایی اپلیکیشن را در پی دارد).

3-14-2-Adapter ها

همان طور که در بالا اشاره شد، دومین المان، کلاس adaptor هست که view یا همان المانی که در لیست قرار می گیرد را ساخته، آن را در آبجکت viewholder قرار می دهد و در نهایت داده ها را به هر view متصل یا bind می کند. کلاس adapter در واقع وظیفه ی مدیریت data model را بر عهده داشته و آن را با توجه به هر المان در لیست (ویجت RecyclerView) تنظیم یا بروز رسانی می کند. کلاس نام برده ویژگی ها و متدهای کلاس RecyclerView.Adapter را به ارث برده (این کلاس را extend می کند) و توسط متد RecyclerView.setAdapter به recycler view تخصیص می یابد. ورودی adapter می تواند هر آبجکت جاوایی باشد. بر پایه ی این ورودی، آبجکت adapter می بایست تعداد کل آیتم های لیست (تعداد المان هایی که در لیست یا recycler view قرار است نمایش داده شود) را از طریق متد getItemCount() بازگردانی نماید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

می دانید adapter چگونه چیدمان و ظاهر (layout) آیتم های موجود در لیست را مدیریت و آماده می کند؟ در جواب باید گفت که adapter برای نمایش هر آیتم در لیست، فایل layout متناظر و مربوطه ی آن را inflate می کند. (inflate = عبارت است از تبدیل فایل LAYOUT XML به آبجکت view جهت قرار گرفتن در کلاس ACTIVITY. در واقع هر زمان که بخواهید layout را برای مثال در زمان اجرا در برنامه نمایش دهید، باید آن را inflate نمایید). این کار در متد onCreateViewHolder() انجام می شود. این متد به ازای هر المان بصری داخل recycler view، آبجکتی از نوع ViewHolder را به عنوان خروجی بازمی گرداند.

به واسطه ی این نمونه (instance) می توانید به view ها در layout بارگذاری و به صورت نهایی ارائه /inflate شده دسترسی داشته باشید. هر زمان که لازم باشد یک view جدید ساخته شود، متد onCreateViewHolder صدا خورده می شود. متد نام برده همان طور که از نامش پیدا است، view را می سازد، سپس آن را در ViewHolder قرار می دهد و در نهایت آبجکت مورد نظر را برای نمایش به RecyclerView می دهد.

Adapter هر المان در لیست (recycler view) را با آیتم متناظر و مناسب (در) data model پر می کند. پس از اینکه داده ی (data item) مورد نظر در دسترس قرار می گیرد، adapter این داده را به تک تک widget های مربوطه که قبلا inflate و در حافظه بارگذاری کرده، متصل یا bind می کند. این عملیات در بدنه ی متد onBindViewHolder() انجام می شود. به عبارت دیگر این متد view را می گیرد و داده های مربوطه از data model را به آن به صورت دو طرفه وصل می کند.

برای مثال، آیتمی را در لیست تصور کنید که ممکن است یک عکس در سمت چپ و دو خط متن مانند تصویر زیر در وسط داشته باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فایل layout متناظر این سطر به صورت زیر خواهد بود.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="fill_parent"
   android:layout_height="?android:attr/listPreferredItemHeight"
   android:padding="6dip" >
   <ImageView
      android:id="@+id/icon"
      android:layout_width="wrap_content"
      android:layout_height="fill_parent"
      android:layout_alignParentBottom="true"
      android:layout_alignParentTop="true"
      android:layout_marginRight="6dip"
      android:contentDescription="TODO"
      android:src="@drawable/ic_launcher" />
   <TextView
      android:id="@+id/secondLine"
      android:layout_width="fill_parent"
      android:layout_height="26dip"
      android:layout_alignParentBottom="true"
      android:layout_alignParentRight="true"
      android:layout_toRightOf="@id/icon"
      android:ellipsize="marquee"
      android:singleLine="true"
      android:text="Description"
      android:textSize="12sp" />
   <TextView
      android:id="@+id/firstLine"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:layout_above="@id/secondLine"
      android:layout_alignParentRight="true"
      android:layout_alignParentTop="true"
      android:layout_alignWithParentIfMissing="true"
      android:layout_toRightOf="@id/icon"
      android:gravity="center_vertical"
      android:text="Example application"
      android:textSize="16sp" />
</RelativeLayout>
```

اضافه کردن dependency (کتابخانه ها) به فایل gradle build جهت استفاده از recycler view

ویجت RecyclerView توسط یک کتابخانه ی مجزا در اختیار توسعه دهنده قرار گرفته و در واقع جدا از فریم ورک اندروید ارائه می شود که از ورژن 7 کتابخانه های اندروید به بعد (API 7 و level های

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بالاتر) قابل استفاده می باشد. به منظور استفاده از recycler view در پروژه ی خود، لازم است dependency مربوطه را مانند زیر به فایل gradle build اضافه نمایید.

```
dependencies {
    ...
    compile "com.android.support:recyclerview-v7:23.0.1"
}
```

2-14-4- Layout manager پیش فرض

Layout manager تعیین می کند که داده ها چگونه در لیست یا همان RecyclerView به نمایش در آیند (نحوه ی چیدمان و آرایش المان ها در لیست را مدیریت می کند). کتابخانه ی recycler view یک سری layout manager درون ساخته و پیش فرض ارائه می دهد که در زیر عنوان شده:

- LinearLayoutManager – آیتم ها را در یک لیست عمودی یا افقی قابل پیمایش (دارای نوار اسکرول) به نمایش می گذارد.
- GridLayoutManager – آیتم ها را با نمای جدولی (به صورت خانه بندی شده) و به اصطلاح در یک grid نمایش می دهد.
- StaggeredGridLayoutManager – آیتم ها را در قالب یک staggered grid به نمایش می گذارد (در staggered grid، بر خلاف Grid View معمولی، هر گرید یا خانه برای خود طول و ارتفاع متفاوتی می تواند داشته باشد و به عبارتی خانه های جدول و آیتم هایی که در لیست به نمایش در می آیند می توانند نامتقارن باشند).

2-14-5- کلاس های مورد نیاز برای پیاده سازی RecyclerView

به منظور پیاده سازی RecyclerView لازم است تعدادی کلاس را قبلش پیاده سازی کرده باشید. در جدول زیر به برخی از مهمترین این کلاس ها همراه با شرح کاربرد اشاره شده است.

کلاس	کاربرد	Optional
Adapter	view یا همان المانی که در لیست قرار می گیرد را ساخته، آن را در viewholder قرار می دهد و در نهایت داده ها را به هر view متصل یا bind می کند.	الزامی
ViewHolder	تمامی المان هایی که در لیست مشاهده می کنید، اشاره گر یا متغیری متناظر در آبجکت view holder دارند. این متغیرها با مقدار المان های لیست پر می شوند. به عبارت بهتر این کلاس تک تک view های تشکیل دهنده ی لیست که متناظر آن ها به صورت تگ در فایل Layout قرار دارد را در خود ذخیره کرده و به طور آماده در دسترس قرار می دهد، به گونه ای که دیگر لازم نباشد برای هر با استفاده از view مورد نظر، در فایل layout آن را جستجو کنید.	الزامی
LayoutManager	این کلاس مدیریت نحوه ی چیدمان و نمایش المان ها در لیست را بر عهده دارد. به عبارت دیگر LayoutManager آیتم ها یا view های موجود در لیست را اندازه گیری و موقعیت دهی کرده و همچنین قوانین مربوط به زمان بازیافت و استفاده ی مجدد از view هایی که دیگر برای کاربر قابل مشاهده نیستند را تعیین می کند. با ویرایش LayoutManager، کلاس RecyclerView قادر خواهد بود به راحتی یک لیست افقی قابل پیمایش، یک grid معمولی، staggered grid، لیست های افقی با قابلیت پیمایش و غیره ... پیاده سازی کند.	الزامی است، هرچند پیاده سازی های پیش فرض و آماده دارد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

کلاس	کاربرد	Optional
ItemDecoration	بر روی آیتم های لیست یا پیرامون آن ها decoration (خط های تزئین/خط پیرامون یا جداکننده غیره ..) ترسیم می کند.	دارای رفتار پیش فرض است، هرچند پیاده سازی یا بدنه ی آن قابل بازنویسی یا override می باشد.
ItemAnimator	ItemAnimator این امکان را فراهم می کند تا برای مبنای عملیاتی که بر روی آیتم های لیست اجرا می شود (یک المان اضافه/حذف شده یا ترتیب آن تغییر می کند)، انیمیشن خاصی تعریف نمایید. به عبارت دیگر به شما اجازه می دهد تا انیمیشن هایی تعریف کنید که با تغییر در adapter، بر روی المان های لیست اعمال می شوند.	دارای رفتار پیش فرض بوده، اما قابل بازنویسی می باشد.

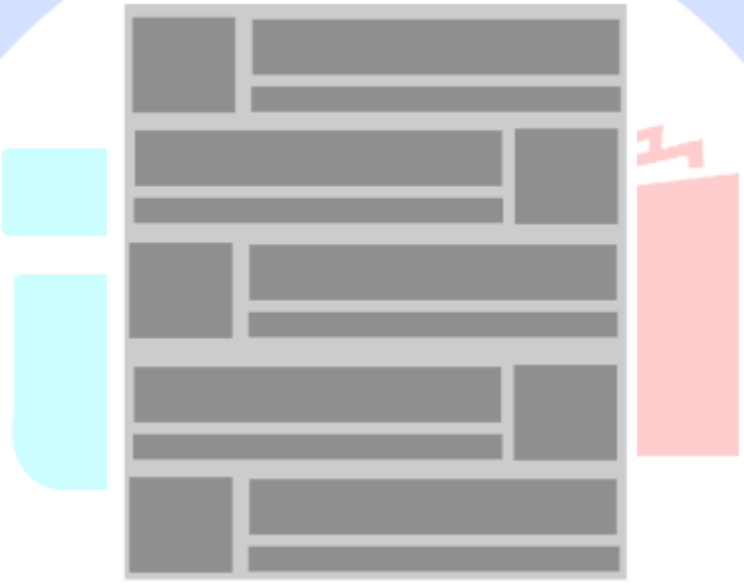
می توانید این animation ها و layout manager ها را به صورت اختصاصی پیاده سازی نمایید.

6-14-2-مدیریت event های مربوط به کلیک یا لمس نمایشگر در لیست/recycler view

رخدادهایی که بر اثر لمس یا کلیک بر روی آیتم های موجود در لیست فعال می شوند، بایستی توسط view ها در recycler view مدیریت شود. در صورتی که view سبب فراخوانی رخدادی در object میزبان (activity یا fragment) شود، می توانید آن را از طریق متد سازنده (constructor) کلاس adapter مدیریت نمایید. این کار به شما امکان می دهد تا اشاره گری (reference) به آبجکت مورد نظر ذخیره کرده و متدهای آن را برای بازخورد صدا بزنید.

7-14-2-استفاده از layout های مختلف در recycler view

Adapter می بایست به ازای هر سطر در لیست، یک layout ایجاد کند. root فایل layout معمولا یک کلاس ViewGroup است که خود چندین view مانند ImageView و TextView را شامل می شود. تصویر زیر لیستی را به نمایش می گذارد که در آن سطرهای زوج و فرد چیدمان و layout متفاوتی دارند.



Recycler view نوع داده ای view را از طریق متد getItemViewType تشخیص می دهد (متد getItemViewType نوع view مستقر در مکان خاص را به منظور recycle و استفاده ی مجدد از آن برمی گرداند). چارچوب نرم افزاری اندروید (framework) خود به صورت پیش فرض متد onCreateViewHolder را، در صورتی که برای این نوع داده ای لازم باشد، صدا می زند. در متد مذکور، layout مناسب را برای نوع مورد نظر inflate کرده و سپس view holder مربوطه را بازگردانی می کنید.

8-14-2-پیاده سازی انیمیشن های اختصاصی

به منظور تنظیم سفارشی و پیاده سازی انیمیشن ها در کلاس RecyclerView، می بایست از کلاس RecyclerView.ItemAnimator ارث بری کرده (آن را extend نمایید) و سپس به وسیله ی متد RecyclerView.setItemAnimator() آن را به widget یا آیتم مورد نظر در لیست خود تخصیص دهید.

9-14-2-فیلتر و مرتب سازی داده ها

کلاس adapter عملیات فیلتر و مرتب سازی داده ها را بر عهده دارد. برای این منظور لازم است منطق مربوطه را در بدنه ی کلاس adapter اختصاصی خود پیاده سازی نمایید.

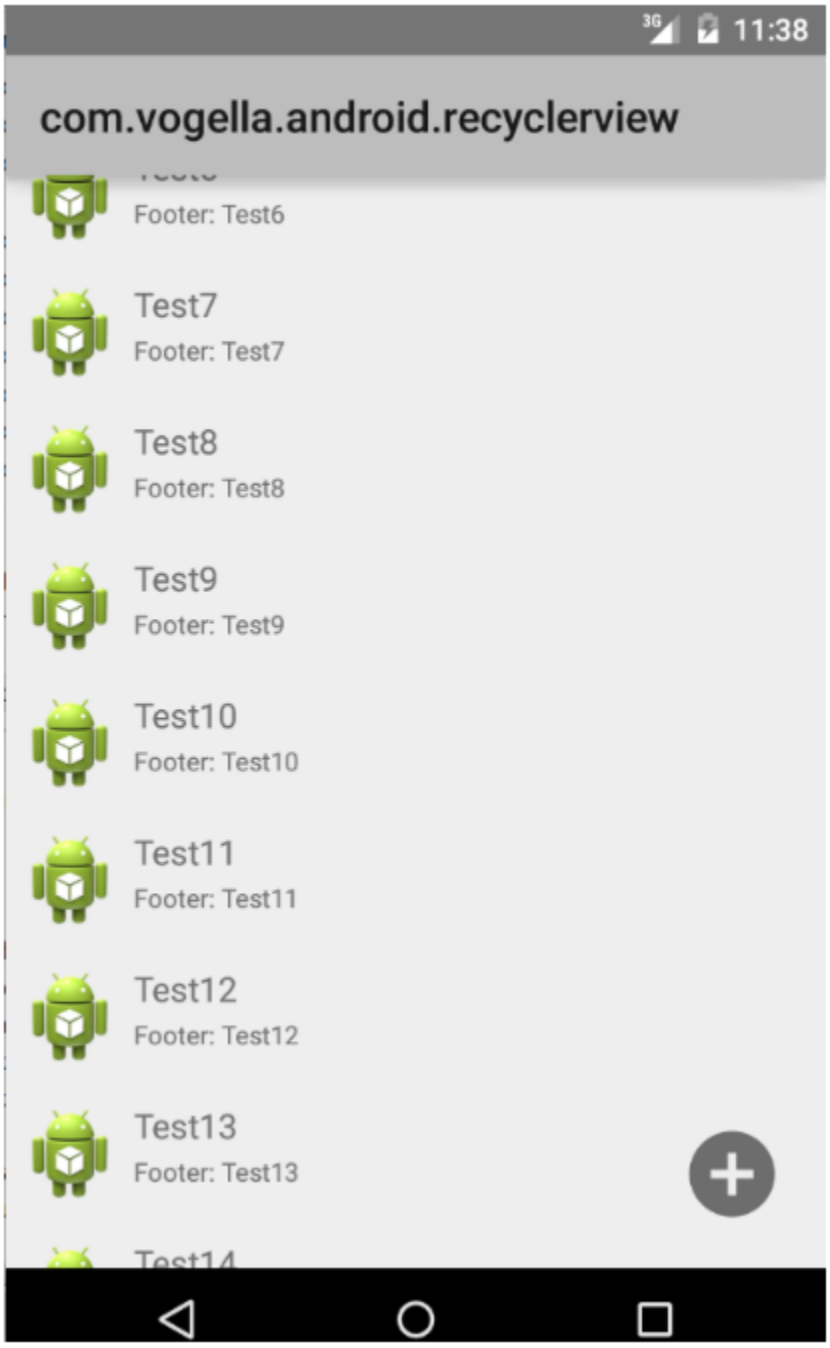
10-14-2-بروز رسانی داده ها در کلاس adapter

به وسیله ی متد notifyItemInserted(position) در کلاس adapter می توان view را مطلع نمود که یک المان جدید در مکان قرار گیری خاصی درج شده است.

به وسیله ی متد notifyItemRemoved(position); می توان view را مطلع کرد که المانی مستقر در مکان خاص، از view حذف شده است.

تمرین: پیاده سازی RecyclerView و استفاده از آن در پروژه

در این تمرین یک پروژه ایجاد خواهید نمود که با استفاده از کلاس RecyclerView لیست پیچیده ای ایجاد نموده و برای کاربر نمایش می دهید.



11-14-2-ایجاد پروژه ها و افزودن dependency مورد نظر در فایل gradle build

ابتدا یک پروژه ی جدید Android با نام پکیج com.vogella.android.recylerview ایجاد نمایید.

جهت استفاده از کلاس RecyclerView، کتابخانه یا Dependency زیر را در بخش dependencies به فایل gradle build اضافه نمایید.

```
dependencies {
    ...
    compile "com.android.support:recyclerview-v7:23.0.1"
}
```

12-14-2-ایجاد فایل های layout مورد نیاز

یک فایل layout به نام activity_main.xml ایجاد کرده و سپس RecyclerView را در آن تعریف نمایید.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="${relativePackage}.${activityClass}" >
    <!-- A RecyclerView with some commonly used attributes -->
    <android.support.v7.widget.RecyclerView
        android:id="@+id/my_recycler_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scrollbars="vertical" />
    <ImageView
        android:id="@+id/imageView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_marginBottom="12dp"
        android:layout_marginRight="12dp"
        android:elevation="2dp"
        android:src="@drawable/ic_add_circle" />
</RelativeLayout>
```

نکته : همان طور که در فایل XML می بینید، المان ImageView یک attribute به نام android:elevation دارد که با پارامتر 2dp مقدار دهی شده است. این ویژگی به اندروید دستور می دهد که برای المان مزبور سایه ایجاد کند.

حال یک فایل layout تعریف می کنید که ویژگی های هر آیتم در لیست را مشخص می کند.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="?android:attr/listPreferredItemHeight"
```

```
    android:padding="6dip" >
    <ImageView
        android:id="@+id/icon"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
        android:layout_alignParentBottom="true"
        android:layout_alignParentTop="true"
        android:layout_marginRight="6dip"
        android:contentDescription="TODO"
        android:src="@drawable/ic_launcher" />
    <TextView
        android:id="@+id/secondLine"
        android:layout_width="fill_parent"
        android:layout_height="26dip"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_toRightOf="@id/icon"
        android:ellipsize="marquee"
        android:singleLine="true"
        android:text="Description"
        android:textSize="12sp" />
    <TextView
        android:id="@+id/firstLine"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_above="@id/secondLine"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:layout_alignWithParentIfMissing="true"
        android:layout_toRightOf="@id/icon"
        android:gravity="center_vertical"
        android:text="Example application"
        android:textSize="16sp" />
</RelativeLayout>
```

نکته: در حال حاضر نمی توان لیستی که با کلاس RecyclerView پیاده سازی شده را در حالت پیش نمایش (preview) مشاهده نمود.

کلاس adapter را به صورت زیر پیاده سازی نمایید.

```
package com.vogella.android.recyclerview;
import java.util.ArrayList;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup;
import android.widget.TextView;
public class MyAdapter extends RecyclerView.Adapter<MyAdapter.ViewHolder> {
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
private ArrayList<String> mDataset;
// Provide a reference to the views for each data item
// Complex data items may need more than one view per item, and
// you provide access to all the views for a data item in a view holder
public class ViewHolder extends RecyclerView.ViewHolder {
      // each data item is just a string in this case
      public TextView txtHeader;
      public TextView txtFooter;
      public ViewHolder(View v) {
            super(v);
            txtHeader = (TextView) v.findViewById(R.id.firstLine);
            txtFooter = (TextView) v.findViewById(R.id.secondLine);
      }
}
public void add(int position, String item) {
      mDataset.add(position, item);
      notifyItemInserted(position);
}
public void remove(String item) {
      int position = mDataset.indexOf(item);
      mDataset.remove(position);
      notifyItemRemoved(position);
}
// Provide a suitable constructor (depends on the kind of dataset)
public MyAdapter(ArrayList<String> myDataset) {
      mDataset = myDataset;
}
// Create new views (invoked by the layout manager)
@Override
public MyAdapter.ViewHolder onCreateViewHolder(ViewGroup parent,
            int viewType) {
      // create a new view
      View v = LayoutInflater.from(parent.getContext()).inflate(R.layout.rowlayout, parent, false);
      // set the view's size, margins, paddings and layout parameters
      ViewHolder vh = new ViewHolder(v);
      return vh;
}
// Replace the contents of a view (invoked by the layout manager)
@Override
public void onBindViewHolder(ViewHolder holder, int position) {
      // - get element from your dataset at this position
      // - replace the contents of the view with that element
      final String name = mDataset.get(position);
      holder.txtHeader.setText(mDataset.get(position));
      holder.txtHeader.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                  remove(name);
            }
        });
      holder.txtFooter.setText("Footer: " + mDataset.get(position));
```

```
    }
    // Return the size of your dataset (invoked by the layout manager)
    @Override
    public int getItemCount() {
        return mDataset.size();
    }
}
```

اکنون شما می توانید recycler view را در activity خود تنظیم نمایید.

```
public class MyActivity extends Activity {
  private RecyclerView mRecyclerView;
  private RecyclerView.Adapter mAdapter;
  private RecyclerView.LayoutManager mLayoutManager;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.my_activity);
    mRecyclerView = (RecyclerView) findViewById(R.id.my_recycler_view);
    // use this setting to improve performance if you know that changes
    // in content do not change the layout size of the RecyclerView
    mRecyclerView.setHasFixedSize(true);
    // use a linear layout manager
    mLayoutManager = new LinearLayoutManager(this);
    mRecyclerView.setLayoutManager(mLayoutManager);
    // specify an adapter (see also next example)
    mAdapter = new MyAdapter(myDataset);
    mRecyclerView.setAdapter(mAdapter);
  }
  ...
}
```

بخش پنجم :

استفاده از کتابخانه ی data binding جهت اتصال دو طرفه داده ها از Model به UI/ data binding در اندروید

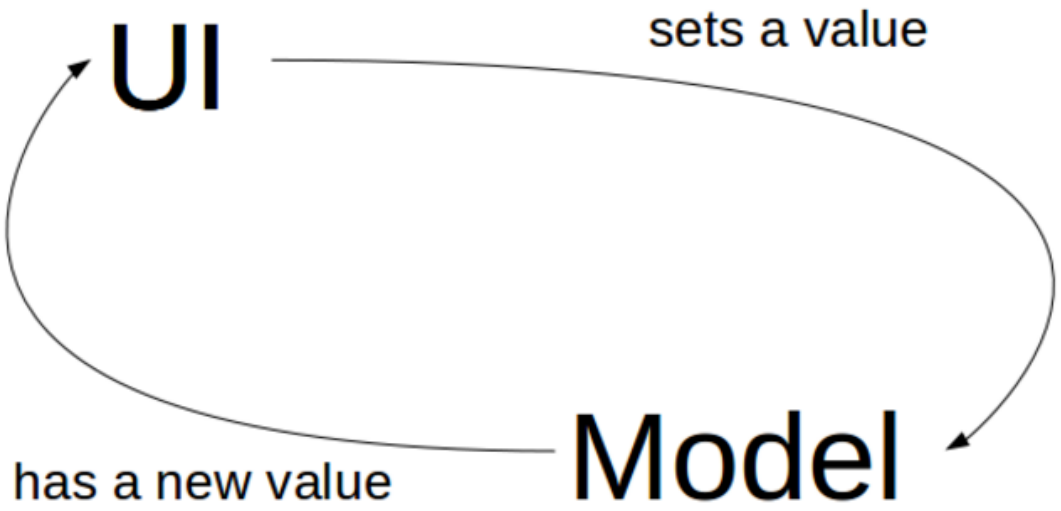
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بر کسی پوشیده نیست که اپلیکیشن های اندروید دائما با داده سروکار دارند. این داده ها قاعدتا باید یا از یک منبع داده همچون دیتابیس خوانده شوند یا از اینترنت گرفته شده و در اپلیکیشن مورد نظر برای کاربر به نمایش در آیند.

طبیعتا داده های برنامه باید با نما و ظاهر خاصی در نمایشگر دستگاه نشان داده شوند. همان طور که می دانید در اندروید توسعه دهنده اغلب با آبجکت های activity، fragment و لایه ها سروکار دارد و ظاهر برنامه را با آیتم های ذکر شده طراحی می کند. در مرحله ی بعدی توسعه دهنده این با جاوا این آبجکت ها را مقداردهی می کند. نا گفته مشخص است که این کار بسیار تکراری است و در فایل های XML انجام می شود. به این دلیل Google کتابخانه ای ارائه داد که به واسطه ی آن برنامه نویس قادر است به راحتی مقداردهی را در فایل های layout و با همان تگ های XML انجام دهد و سپس مقادیر به view های مربوطه متصل شوند. این کتابخانه یک support library هست که با وجود جدید بودن، به راحتی برای ویرایش های قبلی، البته از 2.1 به بالا قابل استفاده می شود.

این مبحث به شما آموزش می دهد چگونه با استفاده از کتابخانه ی Data Binding فایل های layout با فرمت XML بسازید و کد لازم برای متصل کردن منطق اپلیکیشن و فایل های layout (glue code) را به حداقل برسانید.



جهت استفاده از این کتابخانه، لازم است افزونه ی اندروید ویرایش 1.5.0 سیستم کامپایل Gradle یا بالاتر را نصب نمایید.

به منظور استفاده از این کتابخانه برای اتصال داده، می بایست تکه کد زیر را به فایل app/build.gradle اضافه نمایید:

```
android {
    ....
    dataBinding {
        enabled = true
    }
}
```

15-2-فایل های layout اتصال داده از model به UI (Data Binding Layout Files)

فایل های layout ای که data binding انجام می دهند، با تگ layout آغاز شده و المان data را به صورت تودرتو در خود دارند و سپس المان آغازین و میزبان view را شامل می شود. این تگ view

در واقع المانی است که در یک فایل non-binding، ریشه یا root در آن قرار می گرفت.در زیر نمونه ای از یک فایل binding layout را مشاهده می کنید:

```
<?xml version="1.0" encoding="utf-8"?>
<layout xmlns:android="http://schemas.android.com/apk/res/android">
  <data>
    <variable name="user" type="com.vogella.android.databinding.User"/>
  </data>
  <LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@{user.firstName}"/>
    <TextView android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@{user.lastName}"/>
  </LinearLayout>
</layout>
```

اطلاعات data-binding به فایل های layout پروژه ی شما اضافه می شود. در طول پروسه ی build پروژه، کلاس های binding بر اساس اطلاعات این فایل تولید می شوند. به صورت پیش فرض، اسم کلاس binding بر اساس اسم فایل layout ساخته می شود. در واقع اسم فایل با سیستم نوشتاری Pascal نگارش شده و سپس پسوند "Binding" به انتهای آن الصاق می شود. برای مثال اگر اسم فایل layout شما activity_main.xml باشد، اسم کلاس متعاقبا ActivityMainBinding خواهد بود. این کلاس تمامی binding ها از متغیرهای فایل layout (تگ variable که در نمونه فایل فوق مشاهده می کنید) که اشاره گری به View های فایل layout هستند را شامل می شود و همچنین دقیقا می داند چگونه مقادیر را به view ها متصل (bind) کند. المان data که در فایل بالا مشاهده می کنید، داده هایی را توصیف می کند که قرار است به صورت دو طرفه متصل شوند. Expression ها یا عبارات داخل فایل های layout با دستور نگارشی "@{}" در attribute property ها نوشته می شوند.

در این فایل همچنین تگ variable با نام user را می بینید که یک متغیر (property) تعریف می کند و این متغیر می تواند در layout جاری بکار گرفته شود. در واقع این المان اسم یک آبجکت را

مشخص می کند که property های آن در این فایل layout مورد دسترسی قرار می گیرد. برای مثال "@{user.lastName}" بیانگر مقدار فیلد lastName از کلاس User می باشد.

15-2-1- data binding و مدیریت event ها

data binding در اندروید به شما اجازه می دهد تا عبارت هایی بنویسید که event های ارسالی و فعال شده از view ها را مدیریت می کنند (برای مثال می توان به onClick اشاره کرد). اسم attribute های مربوط به event ها بر اساس اسم متد گوش فراخوان آن رخداد (listener method) مشخص می شود. به طور مثال، View.OnLongClickListener یک متد listener به نام onLongClick() دارد و از اینرو attribute مرتبط با این event به این صورت نام گذاری می شود: android:onLongClick.

Event ها می توانند به طور مستقیم به متدهای handler متناظر آن ها متصل باشند، همان طور که android:onClick می تواند به یک متد در activity چسبانده شود.

به منظور تخصیص یک event به handler مربوطه، کافی است از یک عبارت اتصال ساده (binding expression) استفاده کنید. به طوری که مقدار اسم همان متدی باشد که باید با فعال شدن رخداد صدا خورده شود. این عبارت اتصال سپس می تواند listener یا گوش فرادهنده به رخداد کلیک را به view بچسباند.

15-2-2-بروز رسانی رابط کاربری با تغییر رخ داده در data model

به منظور اتصال داده از model به UI می توان از هر آبجکت ساده ی جاوایی بهره گرفت که در اصطلاح POJO خوانده می شوند. اما متاسفانه تغییر این آبجکت ساده ی جاوا به صورت خودکار سبب بروز رسانی UI نمی شود. به منظور فراهم آوردن بروز رسانی UIهمراه با تغییر در model، آبجکت حاوی داده ها (data object) باید بتوانند بخش UI مربوطه را از تغییراتی که در داده ها رخ می دهد، مطلع کنند. در کل سه مکانیزم جهت اطلاع رسانی view ها از تغییرات در model وجود دارد: 1. Observable objects 2. Observable fields 3. Observable collections.

برای روش اول، اندروید کلاس BaseObservable را ارائه می دهد که شما می توانید از آن ارث بری کنید. این کلاس که داده ها را در خود نگه می دارد، موظف است به محض اینکه مقادیر

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

property ها تغییر کردند، view ها از این تغییرات با خبر کند. جهت اعطا نمودن این امکان، کافی است دستور (annotation) @Bindable را در بالای تابعی که داده را می خواند درج نموده و مطلع ساختن view از تغییرات را در تابعی که مقدار را تنظیم می کند (setter)، از طریق تابع notifyPropertyChanged(BR.celsius); انجام دهید.

```
package com.vogella.android.databinding;
import android.databinding.BaseObservable;
import android.databinding.Bindable;
import java.util.Observable;
public class TemperatureData extends BaseObservable {
   private String celsius;
   public TemperatureData(String celsius) {
      this.celsius = celsius;
   }
   @Bindable                              1
   public String getCelsius() {
      return celsius;
   }
   public void setCelsius(String celsius) {
      this.celsius = celsius;             2
      notifyPropertyChanged(BR.celsius);
   }
}
```

1. یک getter جهت خواندن داده ی مربوطه

2. توسط این بخش از کد تمامی listener ها را از تغییرات با خبر می سازید. BR.celsius یک کلاس تولید شده توسط Android studio می باشد.

این listener به محض رخداد هر گونه بروز رسانی و تغییر در داده، فراخوانی می شود و view های متناظر را به تناسب با تغییر رخ داده ویرایش می کند. با این مکانیزم برنامه نویس اطمینان حاصل می کند که هرگاه مقادیر model تغییر کردند، بخش UI مربوطه نیز بروز رسانی می شود.

3-15-2-پیاده سازی data binding و نمونه ای از کاربرد آن

همان طور که در بالا نیز گفته شد، فایل های layout ای که از data binding پشتیبانی می کنند با تگ layout آغاز شده و یک المان data را شامل می شود که تعریف کننده ی داده ای است که قرار است bind یا دوطرفه متصل شوند. پس از المان data، تگ view میزبان و root را داریم که در یک فایل layout ساده و non-binding همان root و عنصر آغازین محسوب می شد. عبارت

های اتصال دو طرفه ی داده ها (binding expressions) با دستور نگارشی "@{}" در فایل مزبور نوشته می شوند.

یک فایل نمونه به صورت زیر نوشته می شود.

```
<?xml version="1.0" encoding="utf-8"?>
<layout xmlns:android="http://schemas.android.com/apk/res/android">
  <data>
    <variable name="user" type="com.vogella.android.databinding.User"/>
  </data>
  <LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@{user.firstName}"/>
    <TextView android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@{user.lastName}"/>
  </LinearLayout>
</layout>
```

متغیر user در المان data، در واقع یک property تعریف می کند که می تواند داخل این layout مورد استفاده قرار گیرد.

آبجکتی که داده ها را در خود کپسوله کرده است (data object) و همراه این layout بکار می رود، می تواند ظاهری مشابه نمونه ی زیر داشته باشد.

```
package com.vogella.android.databinding;
public class User {
  public final String firstName;
  public final String lastName;
  public User(String firstName, String lastName) {
    this.firstName = firstName;
    this.lastName = lastName;
  }
}

package com.vogella.android.databinding;
import android.app.Activity;
import android.databinding.DataBindingUtil;
import android.databinding.ViewDataBinding;
import android.os.Bundle;
import com.vogella.android.databinding.databinding.ActivityMainBinding;
public class MainActivity extends Activity {
  @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        ActivityMainBinding binding = DataBindingUtil.setContentView(this, R.layout.activity_main);
        User user = new User("Test", "User");
        binding.setUser(user);
    }
}
```

تمرین: استفاده از data binding در پروژه

در طی این پروژه ساده خواهید آموخت چگونه بین المان های رابط کاربری (UI widget) به وسیله ی data binding تعامل بر قرار نمایید. ابتدا یک اپلیکیشن جدید اندرویدی با اسم پکیج com.vogella.android.databinding ایجاد نمایید.

4-15-2- افزودن کتابخانه ی data binding به پروژه

فایل app/build.gradle را باز نموده و با افزودن تکه کد زیر به بخش مربوطه در این فایل، قابلیت data binding را فعال نمایید. توجه کنید که باید build file مناسب را انتخاب نمایید (آن فایلی که نود application را دارد).

```
apply plugin: 'com.android.application'
android {
    dataBinding {
        enabled = true
    }
        .... [REST AS BEFORE...]
```

5-15-2-ساخت کلاس جهت مطلع ساختن view از تغییرات در model

کلاس های زیر را تعریف نمایید.

```
package com.vogella.android.databinding;
import android.databinding.BaseObservable;
import android.databinding.Bindable;
import android.databinding.ObservableField;
import java.util.Observable;
public class TemperatureData extends BaseObservable{
    private String celsius;
    public TemperatureData(String celsius) {
        this.celsius = celsius;
    }
    private String fahrenheit;
```

```
    @Bindable
    public String getCelsius() {
        return celsius;
    }
    public void setCelsius(String celsius) {
        this.celsius = celsius;
        notifyPropertyChanged(BR.celsius);
    }
}

package com.vogella.android.databinding;
public interface MainActivityContract {
    public interface Presenter {
        void onShowData(TemperatureData temperatureData);
    }
    public interface View {
        void showData(TemperatureData temperatureData);
    }
}

package com.vogella.android.databinding;
import android.util.Log;
import android.view.View;
import android.widget.Toast;
public class MainActivityPresenter {
    private MainActivityContract.View view;
    public MainActivityPresenter(MainActivityContract.View view) {
        this.view = view;
    }
    public void onShowData(TemperatureData temperatureData) {
        view.showData(temperatureData);
    }
}
```

6-15-2-تنظیم و ویرایش فایل ها برای استفاده از قابلیت data binding

فایل layout را به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<layout xmlns:android="http://schemas.android.com/apk/res/android">
    <data>
        <variable
            name="temp"
          type="com.vogella.android.databinding.TemperatureData" />
        <variable
            name="presenter"        type="com.vogella.android.databinding.MainActivityPresenter"/>
    </data>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
```

```
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin">
        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@={temp.celsius}"
            android:id="@+id/textView" />
        <EditText
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="8dp"
            android:text="@={temp.celsius}" />
        <Button
            android:text="Show data model"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:onClick="@{() -> presenter.onShowData(temp)}"
            android:id="@+id/button" />
    </LinearLayout>
</layout>
```

توجه: ممکن است تعدادی پیغام هشدار در ویرایشگر مشاهده نمایید. برای مثال ممکن است پیغامی مبنی بر اینکه رشته های hard-code شده در پروژه استفاده شده است دریافت نمایید. در بخش های آتی برای شما شرح می دهیم چگونه این پیغام ها را برطرف نمایید.

```
package com.vogella.android.databinding;
import android.app.Activity;
import android.databinding.DataBindingUtil;
import android.os.Bundle;
import android.widget.Toast;
import com.vogella.android.databinding.databinding.ActivityMainBinding;
public class MainActivity extends Activity implements MainActivityContract.View {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        ActivityMainBinding binding = DataBindingUtil.setContentView(this, R.layout.activity_main);
        MainActivityPresenter mainActivityPresenter = new MainActivityPresenter(this);
        TemperatureData temperatureData = new TemperatureData("10");
        binding.setTemp(temperatureData);
        binding.setPresenter(mainActivityPresenter);
    }
    @Override
    public void showData(TemperatureData temperatureData) {
        String celsius = temperatureData.getCelsius();
        Toast.makeText(this, celsius, Toast.LENGTH_SHORT).show();
    }
}
```

کد کلاس MainActivity را برای فعال سازی data binding تنظیم نمایید.

تست اپلیکیشن

می توانید اپلیکیشن خود را در محیط شبیه ساز راه اندازی نمایید. با کلیک بر روی دکمه، یک popup حامل داده های مرتبط به نمایش در می آید.

با تایپ کردن در فیلد EditText، می بینید که مقادیر المان TextView به صورت خودکار بروز رسانی می شود.

بخش ششم :

16-2-استفاده از کلاس ListView در اندروید

پیاده سازی list در اندروید با کلاس ListView

مقاله ی حاضر به شما آموزش می دهد چگونه ListView را با activity و fragment در اندروید بکار ببرید.

نمایش مجموعه ای سازماندهی شده از المان ها در اندروید یک الگوی بسیار معمول در اپلیکیشن های تحت موبایل است. کاربر رو به روی خود لیستی از آیتم ها را می بیند و می تواند به راحتی داخل آن پیمایش کند. چنین activity در تصویر زیر به نمایش گذاشته شده است.

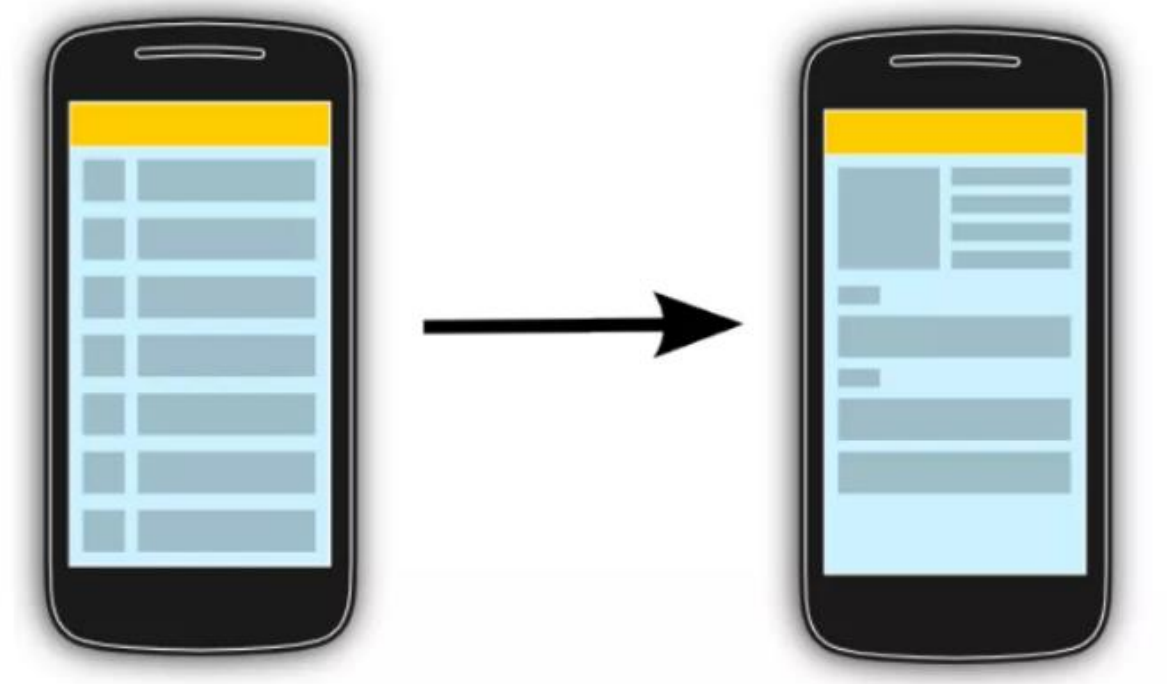
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

به طور معمول، کاربر از طریق نوار ابزار با لیست تعامل بر قرار می کند. به طور مثال، یک دکمه در نوار ابزار که لیست را بروز رسانی نموده و آیتم هایی را از آن حذف یا به آن اضافه می کند. آیتم های فردی لیست را می توان انتخاب نمود که به دنبال این انتخاب نوار ابزار بروز آوری شده یا ممکن است یک صفحه ی جدید با جزئیات فراوان به نمایش در آید. در زیر می بینید که با انتخاب یک آیتم از لیست، activity دیگری راه اندازی شده و برای کاربر به نمایش در می آید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



17-2-اندروید و widget/المان رابط کاربری ListView

17-2-1 View هایی برای مدیریت لیست ها

چارچوب نرم افزاری (framework) اندروید برای پیاده سازی لیست های ساده، دو کلاس ListView و ExpandableListView را ارائه می دهد. با بهره گیری از این دو کلاس، برنامه نویس قادر خواهد بود به راحتی لیست های قابل پیمایش با نوار اسکرول را پیاده سازی کند.

کلاس ExpandableListView علاوه بر سازمان دهی المان ها، آن ها را در دو سطح یا گروه متمایز قرار می دهد، به طوری که کاربر می تواند با کلیک بر روی هر یک، المان های فرزند آن دو را به صورت جداگانه مشاهده کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-17-2-نوع داده ای آیتم های لیست

آیتم هایی که در لیست به نمایش در می آیند، می توانند یک آبجکت جاوا از هر نوعی باشد. کلاس adapter داده های مربوطه را از data object (کلاسی که داده ها در آن کپسوله شده اند) استخراج کرده و داده های واکشی شده را به view های متناظر در ListView تخصیص می دهد.

این آیتم ها به طور معمول همان data model لیست خوانده می شوند. کلاس adapter این داده ها را به عنوان ورودی دریافت می کند. در حقیقت adapter یک پل ارتباطی است که داده ها را از منبع داده (مثلا آرایه) خوانده و آن ها را به Adapter view تحویل می دهد. این داده ها نهایتا در اختیار کامپوننت های UI قرار داده شده و در نمایشگر ارائه می شوند.

2-17-3-Adapter ها

adapter در واقع data model اپلیکیشن را مدیریت می کند و آن را با توجه به المان های فردی در widget بروز رسانی کرده و تطبیق می دهد. کلاس adapter خود از کلاس BaseAdapter مشتق می شود.

هر خط که در widget یا المان رابط کاربری اطلاعاتی را نشان می دهد، خود از یک layout تشکیل شده و می تواند بر اساس نیاز پیچیده یا ساده باشد. ظاهر کلی سطرها در لیست اغلب به این صورت است که در سمت چپ یک عکس قابل مشاهده است و در وسط دو خط متن مانند تصویر زیر را شامل می شود.

فایل layout متناظر این سطر از لیست می تواند به صورت زیر باشد.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
 android:layout_height="?android:attr/listPreferredItemHeight"
  android:padding="6dip" >
```

```
    <ImageView
        android:id="@+id/icon"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
        android:layout_alignParentBottom="true"
        android:layout_alignParentTop="true"
        android:layout_marginRight="6dip"
        android:contentDescription="TODO"
        android:src="@drawable/ic_launcher" />
    <TextView
        android:id="@+id/secondLine"
        android:layout_width="fill_parent"
        android:layout_height="26dip"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_toRightOf="@id/icon"
        android:ellipsize="marquee"
        android:singleLine="true"
        android:text="Description"
        android:textSize="12sp" />
    <TextView
        android:id="@+id/firstLine"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_above="@id/secondLine"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:layout_alignWithParentIfMissing="true"
        android:layout_toRightOf="@id/icon"
        android:gravity="center_vertical"
        android:text="Example application"
        android:textSize="16sp" />
</RelativeLayout>
```

کلاس adapter به ازای هر سطر در لیست، layout مربوطه را inflate می کند (آن را در حافظه بارگذاری و آماده ی نمایش در UI می نماید). در واقع با استفاده از متد getView() هر layout را واکشی و سپس داده های آن را به view های مرتبط در سطر میزبان متصل می کند.

Adapter از طریق متد setAdapter که به آبجکت ListView الحاق شده، به ListView متصل می شود.

توجه: این تنها ListView نیست که از Adapter استفاده می کند، بلکه سایر view هایی که از Adapterview ارث بری دارند، نظیر Spinner، Gallery و StackView نیز برای واکشی داده های خود با این کلاس همکاری دارند.

4-17-2-فیلتر و مرتب سازی داده ها

فیلتر کردن و مرتب سازی داده ها بر عهده ی adapter می باشد. منطق مربوطه را می بایست در بدنه ی adapter اختصاصی خود (که از کلاس پدر ارث بری کردید) پیاده سازی نمایید.

5-17-2-بروز رسانی داده ها در adapter

متد notifyDataSetChanged() در کلاس adapter زمانی فراخوانی می شود که داده ها تغییر کرده باشند یا داده های جدید در دسترس قرار گرفته باشد.

متد notifyDataSetInvalidated() زمانی صدا خورده می شود که داده ها دیگر در دسترس نباشند.

6-17-2-تعریف listener برای آگاهی از تغییرات

برای واکنش نشان دادن به تغییراتی که در لیست رخ می دهد (برای مثال انتخابی که کاربر می کند)، لازم است یک OnItemClickListener به ListView خود الحاق کنید.

```
listView.setOnItemClickListener(new OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view,
        int position, long id) {
        Toast.makeText(getApplicationContext(),
            "Click ListItem Number " + position, Toast.LENGTH_LONG)
            .show();
    }
});
```

18-2-Adapter پیش فرض

1-18-2-کلاس های adapter پیش فرض محیط (platform) اندروید

اندروید تعدادی adapter با پیاده سازی پیش فرض ارائه می دهد که مهم ترین و پرکاربردترین آن ها به شرح زیر می باشد: ArrayAdapter و CursorAdapter.ArrayAdapter که زمانی استفاده می شود که منبع داده ای آرایه باشد و دیگری java.util.List. SimpleCursorAdapter که می تواند به هنگام کار با داده های مستقر در دیتابیس مورد استفاده قرار گیرد (منبع داده ای یک cursor برای ردیف های دیتابیس باشد).

2-18-2-استفاده از ArrayAdapter

کلاس ArrayAdapter می تواند یک لیست یا آرایه ای از آبجکت های جاوا را به عنوان ورودی دریافت کند. هر آبجکت جاوایی به یک سطر نگاشت می شود. در واقع متد ()toString را به ازای هر المان در آرایه صدا می زند و یک شی view می سازد و آن را در TextView قرار می دهد.

شما می توانید ID یا شناسه ی منحصربفرد view را در تابع سازنده (Constructor) کلاس ArrayAdapter تعریف کنید. در غیر این صورت شناسه ی android.R.id.text1 به صورت پیش فرض مورد استفاده قرار می گیرد.

کلاس ArrayAdapter به شما این امکان را می دهد تمامی المان های موجود در منبع داده (data structure) را با فراخوانی متد ()clear حذف نمایید. سپس شما می توانید به وسیله ی متد ()add المان های جدید و به وسیله ی ()addAll یک Collection جدید اضافه نمایید.

این امکان نیز برای توسعه دهنده وجود دارد که محتوای منبع داده (برای مثال آرایه) را ویرایش نموده و سپس با فراخوانی متد ()notifyDataSetChanged در adapter، این آبجکت را از تغییرات در داده ها آگاه نمایید.

توجه: اگر می خواهید داده ها را در adapter خود تغییر دهید، در آن صورت منبع داده ی اصلی (underlying data structure) نیز باید این عملیات را پشتیبانی کند. برای مثال، کلاس ArrayList این امکان را به شما می دهد ولی این امر درمورد آرایه صادق نیست.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

3-18-2-نمونه ی استفاده از ListView با ArrayAdapter

کد XML زیر، یک قطعه از فایل layout به نام activity_listviewexampleactivity.xml را به نمایش می گذارد که کامپوننت رابط کاربری ListView را شامل می شود.

```
<ListView xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/listview"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content" />
```

مثال زیر کاربرد آبجکت ListView در یک activity را به نمایش می گذارد. این مثال برای چیدمان و تنظیم ظاهر سطرها (row layout) از یک layout پیش فرض و آماده ی محیط اندروید (platform) بهره می گیرد. علاوه بر آن مثال حاضر حذف آیتم ها از لیست را همراه با انیمیشن به نمایش می گذارد.

```
package com.vogella.android.listview.withanimation;
public class ListViewExampleActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_listviewexampleactivity);
        final ListView listview = (ListView) findViewById(R.id.listview);
        String[] values = new String[] { "Android", "iPhone", "WindowsMobile",
                "Blackberry", "WebOS", "Ubuntu", "Windows7", "Max OS X",
                "Linux", "OS/2", "Ubuntu", "Windows7", "Max OS X", "Linux",
                "OS/2", "Ubuntu", "Windows7", "Max OS X", "Linux", "OS/2",
                "Android", "iPhone", "WindowsMobile" };
        final ArrayList<String> list = new ArrayList<String>();
        for (int i = 0; i < values.length; ++i) {
            list.add(values[i]);
        }
        final StableArrayAdapter adapter = new StableArrayAdapter(this,
                android.R.layout.simple_list_item_1, list);
        listview.setAdapter(adapter);
        listview.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, final View view,
                    int position, long id) {
                final String item = (String) parent.getItemAtPosition(position);
                view.animate().setDuration(2000).alpha(0)
                        .withEndAction(new Runnable() {
                            @Override
                            public void run() {
                                list.remove(item);
```

```
                                adapter.notifyDataSetChanged();
                                view.setAlpha(1);
                            }
                        });
                }
            });
    }
    private class StableArrayAdapter extends ArrayAdapter<String> {

        HashMap<String, Integer> mIdMap = new HashMap<String, Integer>();
        public StableArrayAdapter(Context context, int textViewResourceId,
                List<String> objects) {
            super(context, textViewResourceId, objects);
            for (int i = 0; i < objects.size(); ++i) {
                mIdMap.put(objects.get(i), i);
            }
        }
        @Override
        public long getItemId(int position) {
            String item = getItem(position);
            return mIdMap.get(item);
        }
        @Override
        public boolean hasStableIds() {
            return true;
        }
    }
}
```

19-2-پیاده سازی adapter های اختصاصی

19-2-1-ساخت یک adapter اختصاصی

برای مدیریت تخصیص داده و پشتیبانی از چندین view، می بایست خود یک کلاس adapter با پیاده سازی اختصاصی تعریف نمایید.

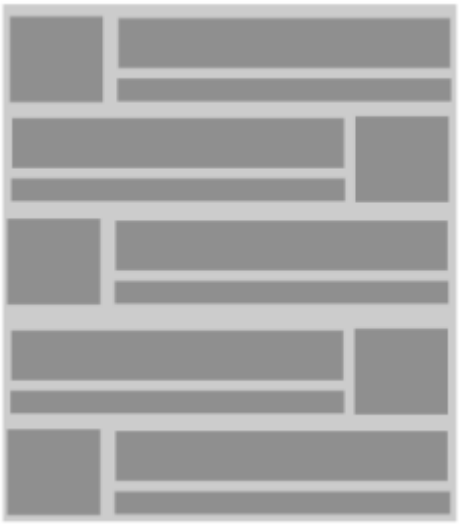
برای این منظور ابتدا لازم است یک کلاس پیاده سازی شده ی adapter را به ارث ببرید و یا به طور مستقیم از کلاس BaseAdapter یک کلاس فرزند ایجاد کرده و بدنه ی آن را خود پیاده سازی کنید.

توجه: اغلب توسعه دهنده از کلاس ArrayAdapter ارث بری کرده و یک adapter اختصاصی پیاده سازی می کند. این کار به مراتب ساده تر از ایجاد کلاس فرزند و ارث بری مستقیم از BaseAdapter است.

2-19-2-آماده سازی سطر از لیست

Adapter می بایست به ازای هر سطر از لیست، یک layout ایجاد کند. آبجکت (ListView (instance متد ()getView را برای هر المان در لیست، در سطح کلاس Adater صدا می زند. به واسطه ی این متد layout adapter مربوط به سطر را ایجاد کرده و داده ها را به view های مربوطه در layout نگاشت می کند.

این root معمولا یک ViewGroup (یک layout manager) است که خود میزبان چندین view دیگر می باشد. به عنوان مثال می توان از ImageView و TextView نام برد. تصویر زیر یک لیست را با layout های متفاوت برای سطرهای زوج و فرد نشان می دهد.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

متد getView() فایل XML را خوانده و آن را به listview تبدیل می کند.

حال با استفاده از متد getView() فایل layout مبتنی بر XML خود را بارگذاری یا inflate کرده و سپس محتوای تک تک view های سطر مورد نظر را بر اساس آبجکت جاوا تنظیم می کنید (آن ها را با داده هایی از آبجکت جاوا پر می کنید). به منظور inflate کردن فایل layout مبتنی بر XML خود، می توانید از سرویس LayoutInflator استفاده نمایید.

توجه: این سرویس که layout ها را خوانده و در حافظه بارگذاری می کند (inflate)، به راحتی به واسطه ی فراخوانی متد getLayoutInflator() در سطح کلاس activity و یا فراخوانی متد context.getSystemService(Context.LAYOUT_INFLATER_SERVICE) به راحتی قابل دسترسی می باشد.

پس از اینکه adapter فایل layout مربوطه را در حافظه بارگذاری و به اصطلاح inflate کرد، view های متناظر در layout را پیدا می کند و متعاقبا آن ها را با داده پر می نماید. همان طور که قبلا نیز به آن اشاره شده، جهت پیدا کردن المان های فردی در layout، کافی است متد findViewById() را بر روی top level view (view میزبان که view مورد نظر در آن تعریف شده است) فراخوانی نمایید.

نمونه ای از پیاده سازی یک adapter اختصاصی و custom

کد زیر بدنه ی یک کلاس adapter با پیاده سازی اختصاصی را نشان می دهد. این آبجکت adapter فرض می گیرد شما دو فایل تصویری png (no.png و ok.png) در یکی از پوشه های res/drawable[[directory خود دارید. کد نوشته شده در این adapter، فایل XML layout را در حافظه بارگذاری (inflate) کرده، view های مربوطه را در layout پیدا می کند و سپس محتوای آن ها را بر اساس داده های ورودی مقداردهی و تنظیم می نماید.

```
package de.vogella.android.listactivity;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
```

```
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

public class MySimpleArrayAdapter extends ArrayAdapter<String> {
	private final Context context;
	private final String[] values;
	public MySimpleArrayAdapter(Context context, String[] values) {
		super(context, -1, values);
		this.context = context;
		this.values = values;
	}
	@Override
	public View getView(int position, View convertView, ViewGroup parent) {
		LayoutInflater inflater = (LayoutInflater) context
				.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
		View rowView = inflater.inflate(R.layout.rowlayout, parent, false);
		TextView textView = (TextView) rowView.findViewById(R.id.label);
		ImageView imageView = (ImageView) rowView.findViewById(R.id.icon);
		textView.setText(values[position]);
		// change the icon for Windows and iPhone
		String s = values[position];
		if (s.startsWith("iPhone")) {
			imageView.setImageResource(R.drawable.no);
		} else {
			imageView.setImageResource(R.drawable.ok);
		}
		return rowView;
	}
}
```

3-19-2-بروز رسانی data model از طریق کلاس adapter

سطر مورد نظر می تواند view هایی داشته باشد که با data model (آبجکتی که داده ها را کپسوله می کند) از طریق کلاس adapter تعامل دارند. به طور مثال، شما می توانید یک Checkbox در layout مربوط به سطر داشته باشید و هر زمان که این Checkbox تیک دار و به اصطلاح انتخاب شد، داده های موجود در آبجکت model نیز همگام با آن تغییر کرده و بروز رسانی شوند.

2-20-ListActivity و ListFragment

1-20-2-ظرف آماده و پیش فرض برای استفاده از ListView

اندروید با ارائه ی fragment ها و کلاس های activity ویژه، پیاده سازی و مدیریت لیست را برای توسعه دهنده بسیار آسان می کند.

برای ایجاد لیست در activity و fragment، اندروید به ترتیب کلاس های ListActivity و ListFragment را ارائه می دهد.

لازم به ذکر است که برای این المان ها نیازی به تخصیص layout نیست چرا که activity یا fragment مورد نظر خود به صورت پیش فرض خود یک ListView آماده دارند.

ListActivity و ListFragment به برنامه نویس این اجازه را می دهند تا با بازنویسی (override) پیاده سازی متد onListItemClick() انتخاب آیتم ها در لیست را مدیریت کند.

هر دو کلاس همچنین این اختیار به برنامه نویس می دهند تا adapter را به وسیله ی متد SetListAdapter() به ListView پیش فرض متصل کند.

نمونه کد زیر یک پیاده سازی ساده از ListFragment را نشان می دهد.

```
package de.vogella.android.fragments;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.app.ListFragment;
public class MyListFragment extends ListFragment {
    @Override
    public void onActivityCreated(Bundle savedInstanceState) {
        super.onActivityCreated(savedInstanceState);
        String[] values = new String[] { "Android", "iPhone", "WindowsMobile",
                "Blackberry", "WebOS", "Ubuntu", "Windows7", "Max OS X",
                "Linux", "OS/2" };
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(getActivity(),
                android.R.layout.simple_list_item_1, values);
        setListAdapter(adapter);
    }
    @Override
    public void onListItemClick(ListView l, View v, int position, long id) {
        // TODO implement some logic
```

```
    }
}
```

نمونه کد بعدی که مشاهده می کنید، استفاده از ListActivity را به نمایش می گذارد.

```
package de.vogella.android.listactivity;
import android.app.ListActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
public class MyListActivity extends ListActivity {
      public void onCreate(Bundle icicle) {
            super.onCreate(icicle);
            String[] values = new String[] { "Android", "iPhone", "WindowsMobile",
                  "Blackberry", "WebOS", "Ubuntu", "Windows7", "Max OS X",
                  "Linux", "OS/2" };
            ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                  android.R.layout.simple_list_item_1, values);
            setListAdapter(adapter);
      }
}
```

2-20-2- ListActivity و layout اختصاصی

شما می توانید برای هر دو کلاس ListActivity و ListFragment فایل layout اختصاصی داشته باشید. در این مثال، fragment یا activity در layout ارائه شده به دنبال ListView ای که مقدار android:id آن به صورت پیش فرض بر روی @android:id/list تنظیم شده است، می گردد. تکه کد زیر این کاربرد را به نمایش می گذارد.

```
<ListView
 android:id="@android:id/list"
 android:layout_width="match_parent"
 android:layout_height="wrap_content" >
</ListView>
```

توجه: در صورتی که از این ID استفاده نکرده یا ListView را در layout خود اضافه نکنید، اپلیکیشن به هنگام نمایش activity یا fragment مربوطه از کار می افتد.

2-20-3- انتخاب یک view مکان نگهدار/placeholdder برای لیست خالی

می توانید از یک view با شاسنه ی @android:id/empty در layout خود استفاده کنید. چنانچه ListView تهی باشد، در آن صورت activity و fragment مربوطه این view را به صورت خودکار

نمایش می دهند و در غیر این صورت آن را به طور کلی مخفی می کند. در چنین view ای شما می توانید یک پیغام خطا را به نمایش بگذارید.

تمرین: استفاده از ListView و ListActivity

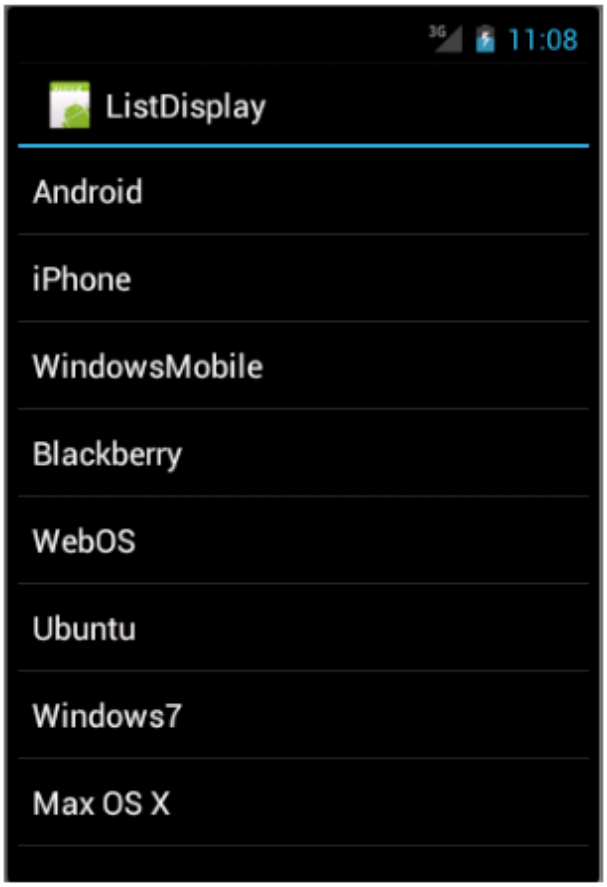
پروژه ی زیر نحوه ی استفاده از یک ListView در ListActivity را نمایش می دهد. در این پروژه شما از کلاس از پیش تعریف شده ی ArrayAdapter و یک layout آماده ی اندروید برای سطرها استفاده می کنید.

یک پروژه ی جدید اندروید به نام de.vogella.android.listactivity و یک کلاس به نام MyListActivity ایجاد کنید.

کلاس MyListActivity را بر اساس نمونه کد زیر ویرایش نمایید. همان طور که می بینید متد setContentView() استفاده نشده است.

```
package de.vogella.android.listactivity;
import android.app.ListActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;
public class MyListActivity extends ListActivity {
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        String[] values = new String[] { "Android", "iPhone", "WindowsMobile",
                "Blackberry", "WebOS", "Ubuntu", "Windows7", "Max OS X",
                "Linux", "OS/2" };
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_1, values);
        setListAdapter(adapter);
    }
    @Override
    protected void onListItemClick(ListView l, View v, int position, long id) {
        String item = (String) getListAdapter().getItem(position);
        Toast.makeText(this, item + " selected", Toast.LENGTH_LONG).show();
```

```
    }
}
```



تمرین: ListActivity با layout اختصاصی

در مثال حاضر، فایل layout مربوط به سطرها را خود به صورت اختصاصی تعریف کرده و آن را در adapter بکار می برید.

ابتدا فایل rowlayout.xml را در پوشه ی directory]]res/layout پروژه ی de.vogella.android.listactivity ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content" >
  <ImageView
    android:id="@+id/icon"
```

```
        android:layout_width="22px"
        android:layout_height="22px"
        android:layout_marginLeft="4px"
        android:layout_marginRight="10px"
        android:layout_marginTop="4px"
        android:src="@drawable/ic_launcher" >
    </ImageView>
    <TextView
        android:id="@+id/label"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@+id/label"
        android:textSize="20px" >
    </TextView>
</LinearLayout>
```

Activity خود را جهت استفاده از layout جدید ویرایش نمایید.

```
package de.vogella.android.listactivity;
import android.app.ListActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;
public class MyListActivity extends ListActivity {
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        String[] values = new String[] { "Android", "iPhone", "WindowsMobile",
                "Blackberry", "WebOS", "Ubuntu", "Windows7", "Max OS X",
                "Linux", "OS/2" };
        // use your custom layout
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                R.layout.rowlayout, R.id.label, values);
        setListAdapter(adapter);
    }
    @Override
    protected void onListItemClick(ListView l, View v, int position, long id) {
        String item = (String) getListAdapter().getItem(position);
        Toast.makeText(this, item + " selected", Toast.LENGTH_LONG).show();
    }
}
```

آموزش: پیاده سازی adapter اختصاصی

برنامه ی زیر از دو فایل تصویری "no.png" و "ok.png" استفاده می کند. این دو فایل در پوشه ی "res/drawable-mdpi" قرار دارند. شما بایستی آیکون های خود را ایجاد کنید. در صورتی که آیکون دلخواه را پیدا نکردید، می توانید "icon.png" را کپی کرده و با استفاده از یک برنامه ی گرافیکی آن را کمی تغییر دهید و دومین آیکون را هم به همین ترتیب ایجاد نمایید.

کلاس MySimpleArrayAdapter را ایجاد کنید. این کلاس adapter و پل ارتباطی بین model و UI خواهد بود.

```
package de.vogella.android.listactivity;
import android.content.Context;
import android.view.LayoutInflater;
```

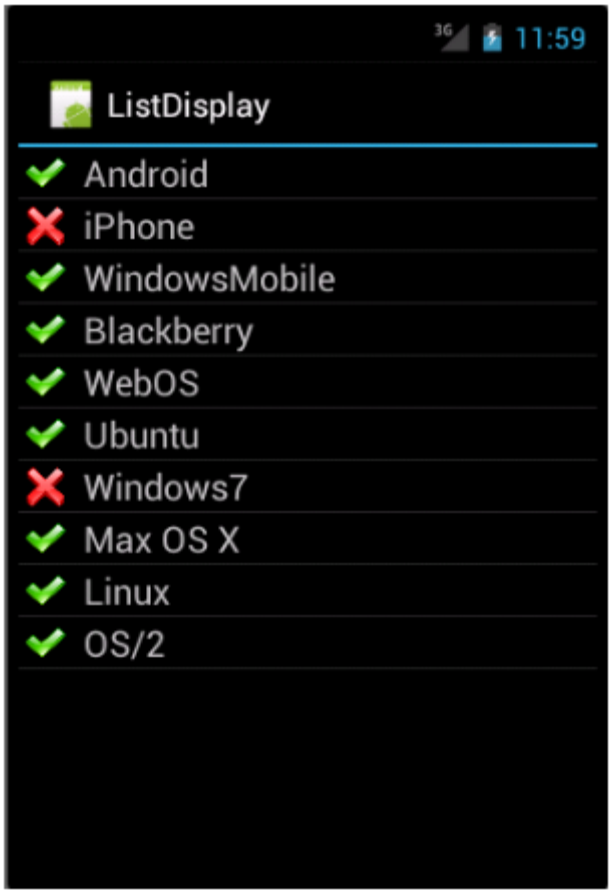
```
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;
public class MySimpleArrayAdapter extends ArrayAdapter<String> {
	private final Context context;
	private final String[] values;
	public MySimpleArrayAdapter(Context context, String[] values) {
		super(context, R.layout.rowlayout, values);
		this.context = context;
		this.values = values;
	}
	@Override
	public View getView(int position, View convertView, ViewGroup parent) {
		LayoutInflater inflater = (LayoutInflater) context
				.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
		View rowView = inflater.inflate(R.layout.rowlayout, parent, false);
		TextView textView = (TextView) rowView.findViewById(R.id.label);
		ImageView imageView = (ImageView) rowView.findViewById(R.id.icon);
		textView.setText(values[position]);
		// Change the icon for Windows and iPhone
		String s = values[position];
		if (s.startsWith("Windows7") || s.startsWith("iPhone")
				|| s.startsWith("Solaris")) {
			imageView.setImageResource(R.drawable.no);
		} else {
			imageView.setImageResource(R.drawable.ok);
		}
		return rowView;
	}
}
```

برای استفاده از این adapter، لازم است activity را به صورت زیر ویرایش نمایید.

To use this adapter, change the *activity* to the following.

```
package de.vogella.android.listactivity;
import android.app.ListActivity;
import android.os.Bundle;
public class MyListActivity extends ListActivity {
	public void onCreate(Bundle icicle) {
		super.onCreate(icicle);
		String[] values = new String[] { "Android", "iPhone", "WindowsMobile",
				"Blackberry", "WebOS", "Ubuntu", "Windows7", "Max OS X",
				"Linux", "OS/2" };
		MySimpleArrayAdapter adapter = new MySimpleArrayAdapter(this, values);
		setListAdapter(adapter);
	}
}
```

با اجرای این برنامه لیستی از آیتم ها را مشاهده می کنید که کنار برخی آیکون ضرب در و کنار برخی دیگر تیک دیده می شود.



ListView-21-2 و مبحث کارایی و سرعت اپلیکیشن

کارایی و سرعت در اندروید از اهمیت بسیار بالایی برخوردار است و کاربران انتظار واکنش سریع و سرعت بالای پاسخگویی از اپلیکیشن را دارند. از لحاظ قدرت سخت افزاری و منابع موجود، یک دستگاه اندروید به هیچ وجه با کامپیوترهای دسکتاپ قابل مقایسه نیست.

در این بخش شرح خواهیم داد چگونه adapter لیست خود را به صورت بهینه پیاده سازی نمایید و عملیات غیر ضروری را جهت افزایش سرعت اپلیکیشن و صرفه جویی در مصرف منابع سیستم

کاهش دهید. گفتنی است که adapter های پیش فرض اندروید همچون ArrayAdapter طوری طراحی شده اند که از لحاظ کارایی کاملا بهینه هستند.

1-21-2-عملیات سنگین و زمان بر

تمامی view هایی که از فایل layout XML خوانده و inflate می شوند، در نهایت به آبجکت های Java تبدیل می شوند. این دو فرایند هر دو علاوه بر زمان بر بودن، حافظه ی قابل توجهی را به خود اختصاص می دهند.

بعلاوه، استفاده از متد findViewById() بسیار زمان بر است، هرچند به اندازه ی inflate کردن فایل های XML سنگین نیست.

2-21-2-جلوگیری از inflation فایل های layout و ایجاد آبجکت های جاوا جهت بهینه سازی برنامه

یک آبجکت ListView معمولا اطلاعات بیشتری نسبت به سطرهای قابل مشاهده در صفحه دربردارد. چنانچه کاربر با اسکرول داخل لیست پیمایش کند، آن هنگام سطرها و view های مربوطه ی هریک از ناحیه ی قابل مشاهده برای کاربر خارج می شوند. آبجکت های جاوا که این سطرها را به نمایش می گذارند و در واقع کد پشت view ها هستند، قابل بازیافت و استفاده ی مجدد برای سطرهایی می باشند که با پیمایش کاربر، در صفحه به نمایش در می آیند.

اگر اندروید تشخیص دهد که سطری دیگر در صفحه جاری قابل مشاهده نیست، به متد getView() از کلاس adapter اجازه می دهد تا view مرتبط را از طریق پارامتر convertView بازیافت کند (reuse).

کلاس adapter می تواند داده های جدید را به view های موجود در سلسله مراتب view ها که با پارامتر convertView در دسترس قرار گرفته، متصل کند. این امر مانع از inflate شدن فایل XML و ایجاد آبجکت های جدید Java می شود.

چنانچه امکان بازیافت یک سطر وجود نداشت (reuse)، سیستم اندروید باید مقدار null را به پارامتر convertView ارسال کند. در واقع، این کد موجود در بدنه ی adapter است که باید مکانیزمی برای مدیریت این سناریو پیاده سازی کند.

3-21-2-استفاده از الگوی توسعه ی View holder در اپلیکیشن جهت بهینه سازی

پیاده سازی الگو توسعه ی ViewHolder این امکان را فراهم می کند تا از فراخوانی متد findViewById() در کلاس adapter جلوگیری شود.

ViewHolder معمولا یک کلاس درون ساخته static در adapter هست که اشاره گرهایی در قالب متغیر به view های مربوطه (که متناظر آن ها در فایل layout تعریف شده) در خود نگهداری می کند. این اشاره گرها به وسیله ی متد setTag() (و به صورت یک تگ) به view متناظر در سطر مورد نظر متصل می شوند.

چنانچه آبجکت convertView را دریافت کردیم، آنگاه می توانیم نمونه ای از ViewHolder را با متد getTag() بازیابی کرده و attribute های جدید را به وسیله ی اشاره گر (متغیر متناظر در ViewHolder به view های مربوطه تخصیص دهیم.

اگرچه این روش کمی پیچیده به نظر می رسد، اما حدودا 15 % سریع تر از فراخوانی متد findViewById() و پیدا کردن تک تک view ها در فایل XML می باشد.

مثال

کد زیر نمونه ای بهینه سازی شده از یک adapter را نشان می دهد که از الگوی توسعه ی view holder استفاده نموده و با بازیافت view های جاری در استفاده از منابع و حافظه صرفه جویی می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
package de.vogella.android.listactivity;
import android.app.Activity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;
public class MyPerformanceArrayAdapter extends ArrayAdapter<String> {
	private final Activity context;
	private final String[] names;
	static class ViewHolder {
		public TextView text;
		public ImageView image;
	}
	public MyPerformanceArrayAdapter(Activity context, String[] names) {
		super(context, R.layout.rowlayout, names);
		this.context = context;
		this.names = names;
	}
	@Override
	public View getView(int position, View convertView, ViewGroup parent) {
		View rowView = convertView;
		// reuse views
		if (rowView == null) {
			LayoutInflater inflater = context.getLayoutInflater();
			rowView = inflater.inflate(R.layout.rowlayout, null);
			// configure view holder
			ViewHolder viewHolder = new ViewHolder();
			viewHolder.text = (TextView) rowView.findViewById(R.id.TextView01);
			viewHolder.image = (ImageView) rowView
					.findViewById(R.id.ImageView01);
			rowView.setTag(viewHolder);
		}
		// fill data
		ViewHolder holder = (ViewHolder) rowView.getTag();
		String s = names[position];
		holder.text.setText(s);
		if (s.startsWith("Windows7") || s.startsWith("iPhone")
				|| s.startsWith("Solaris")) {
			holder.image.setImageResource(R.drawable.no);
		} else {
			holder.image.setImageResource(R.drawable.ok);
		}
		return rowView;
	}
}
```

2-22-ذخیره سازی انتخاب یک view (تعیین وضعیت انتخاب با متد ()setChoiceMode)

به صورت پیش فرض، ListView هیچ حالت انتخاب پیش فرض و فعالی ندارد. شما می توانید این حالت را با فراخوانی متد ()setChoiceMode فعال نمایید. برای اینکه کاربر بتواند چندین آیتم را انتخاب کند، می بایست پارامتر ListView.CHOICE_MODE_MULTIPLE را به متد مذکور ارسال کنید و اگر می خواهید امکان انتخاب یک آیتم را بیشتر نداشته باشد، مقدار ListView.CHOICE_MODE_SINGLE را به عنوان آرگومان به متد مورد نظر پاس می دهید.

جهت بازیابی آیتم های انتخاب شده ی ListView، اگر یک آیتم انتخاب شده باشد، متد getCheckedItemPosition() را فراخوانی می کنید و اگر چندین آیتم انتخاب شده باشد، listView.getCheckedItemPositions() را صدا می زنید. اگر آیتم ها دارای ID مشخصی هستند، می توانید با استفاده از متد getCheckedItemIds() شناسه یا ID آیتم های انتخاب شده را بازگردانی نمایید.

اندروید از قبل برای تنظیم ظاهر لیست هایی که چندین آیتم در آن انتخاب شده باشد، یک layout آماده ارائه می دهد: android.R.layout.simple_list_item_multiple_choice. این فایل دربردارنده ی view کاملا تنظیم شده CheckedTextView می باشد.

در activity های زیر نحوه ی استفاده از این حالت های انتخاب به نمایش گذاشته شده است. لازم به ذکر است که در صورت استفاده از این حالت ها، ListView مقادیر انتخابی را ذخیره می کند. اما اطلاعات مذکور به صورت دائمی در data model ذخیره و ماندگار نمی شوند.

```
package com.vogella.android.listview.selection.multi;
import android.app.ListActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;
import com.vogella.android.listview.selection.R;
public class MainActivity extends ListActivity {
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        String[] values = new String[] { "a", "b", "c", "d", "e", "f", "g",
                "h", "i", "j", "k", "l", "m", "n", "o", "p", "q", "r", "s",
                "t", "u", "w", "x", "y", "z" };
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_multiple_choice, values);
        setListAdapter(adapter);
        getListView().setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        Toast.makeText(this,
                String.valueOf(getListView().getCheckedItemCount()),
                Toast.LENGTH_LONG).show();
        return true;
    }
}

package com.vogella.android.listview.selection.single;
import android.app.ListActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;
public class MainActivity extends ListActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        String[] values = new String[] { "a", "b", "c", "d", "e", "f", "g",
                "h", "i", "j", "k", "l", "m", "n", "o", "p", "q", "r", "s",
                "t", "u", "w", "x", "y", "z" };
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_single_choice, values);
        setListAdapter(adapter);
        getListView().setChoiceMode(ListView.CHOICE_MODE_SINGLE);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
```

```
            Toast.makeText(this,
                    String.valueOf(getListView().getCheckedItemCount()),
                    Toast.LENGTH_LONG).show();
            return true;
        }
}
```

23-2-استفاده از Contextual action mode برای ListView ها (actionbar موقتی که بر روی actionbar اصلی قرار می گیرد)

جهت مطالعه و درک مطالب این بخش، بایستی از قبل با مفهوم ActionBar و contextual action mode آشنا باشید. در زیر به شرح نحوه ی استفاده از contextual action mode برای مدیریت حالت انتخاب در ListView خواهیم پرداخت.

به منظور تخصیص یک contextual action mode به یک آیتم از لیست که با کلیک طولانی مدت روی آیتم مورد نظر بر روی action bar اصلی برنامه (جهت انجام کارهای جزئی تر) نمایان می شود، کافی است متد setOnItemLongClickListener() را بر روی ListView بکار ببرید. این متد اطلاعات آیتم انتخابی را دربر دارد.

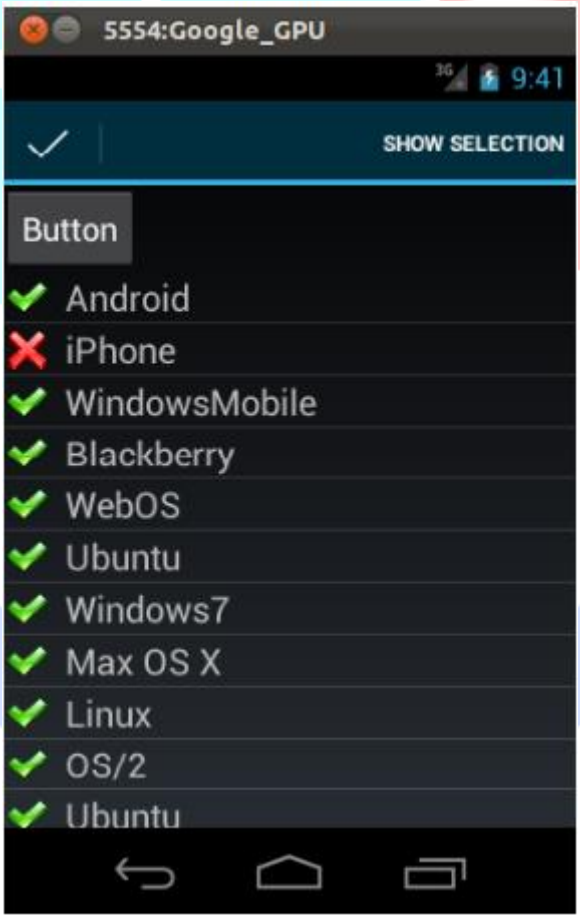
مثال زیر استفاده از contextual action mode را به صورت کاربردی به نمایش می گذارد. برای این مثال لازم است یک فایل منو XML به نام rowselection.xml تعریف کرده باشید که منوی آن تنها یک آیتم با شناسه ی @+id/menuitem1_show داشته باشد.

```
package de.vogella.android.listactivity;
import android.app.ListActivity;
import android.os.Bundle;
import android.view.ActionMode;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemLongClickListener;
import android.widget.Toast;
public class MyListActivityActionbar extends ListActivity
        implements ActionMode.Callback {
    protected Object mActionMode;
    public int selectedItem = -1;
    @Override
    public void onCreate(Bundle icicle) {
```

```
        super.onCreate(icicle);
        String[] values = new String[] { "Android",
                        "iPhone", "WindowsMobile",
                        "Blackberry", "WebOS", "Ubuntu",
                        "Windows7", "Max OS X", "Linux", "OS/2",
                        "Ubuntu", "Windows7", "Max OS X",
                        "Linux", "OS/2", "Ubuntu",
                        "Windows7", "Max OS X",
                        "Linux", "OS/2" };
        MySimpleArrayAdapter adapter = new MySimpleArrayAdapter(this, values);
        setListAdapter(adapter);
        getListView().setOnItemLongClickListener(new OnItemLongClickListener() {
            @Override
            public boolean onItemLongClick(AdapterView<?> parent, View view, int position, long id) {
                if (mActionMode != null) {
                    return false;
                }
                selectedItem = position;
                // Start the CAB using the ActionMode.Callback defined above
                mActionMode =
MyListActivityActionbar.this.startActionMode(MyListActivityActionbar.this);
                view.setSelected(true);
                return true;
            }
        });
    }
    private void show() {
        Toast.makeText(MyListActivityActionbar.this, String.valueOf(selectedItem),
Toast.LENGTH_LONG).show();
    }
    // Called when the action mode is created; startActionMode() was called
    @Override
    public boolean onCreateActionMode(ActionMode mode, Menu menu) {
        // Inflate a menu resource providing context menu items
        MenuInflater inflater = mode.getMenuInflater();
        // Assumes that you have "contexual.xml" menu resources
        inflater.inflate(R.menu.rowselection, menu);
        return true;
    }
    // Called each time the action mode is shown. Always called after
    // onCreateActionMode, but
    // may be called multiple times if the mode is invalidated.
    @Override
    public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
        return false; // Return false if nothing is done
    }
    // Called when the user selects a contextual menu item
    @Override
    public boolean onActionItemClicked(ActionMode mode, MenuItem item) {
        switch (item.getItemId()) {
        case R.id.menuitem1_show:
```

```
                show();
                // Action picked, so close the CAB
                mode.finish();
                return true;
            default:
                return false;
            }
        }
        // Called when the user exits the action mode
        @Override
        public void onDestroyActionMode(ActionMode mode) {
            mActionMode = null;
            selectedItem = -1;
        }
}
```

حال اگر اپلیکیشن خود را اجرا کرده و بر روی یک آیتم در آن طولانی مدت کلیک کنید، contextual action bar شما به نمایش در می آید.



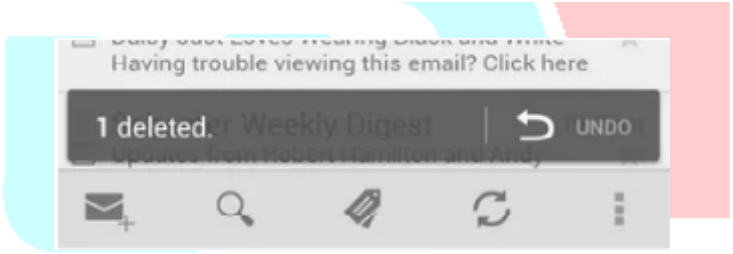
24-2-پیاده سازی قابلیت لغو/undo action

1-24-2-چه زمانی باید قابلیت لغو عملیات را به برنامه اضافه کنید

لازم است قابلیت لغو عملیات حساس را برای کاربر اپلیکیشن خود فراهم کنید. برای مثال ممکن است کاربر از حذف آیتم های انتخابی لیست منصرف شده و بخواهد این عملیات را لغو کند.

روش معمول و پر طرفدار این است که یک گزینه در انتهای صفحه برای کاربر به نمایش بگذارید که امکان انتخاب این گزینه با گذشت مدت زمان خاص یا هنگامی که کاربر به ادامه ی تعامل با اپلیکیشن می پردازد، از میان برداشته می شود.

به عنوان نمونه می توان به اپلیکیشن Gmail اشاره کرد که دقیقا همین رفتار را برای لغو عملیات (حذف) پیاده سازی می کند.



مثال

نمونه ی زیر قابلیت undo جهت لغو عملیات حذف دائمی یک آیتم را پیاده سازی می کند. برای این منظور یک نوار موقتی نمایان می شود که کاربر برای کلیک بر روی آن مدت زمان محدودی دارد و پس از گذشت آن بازه ی زمانی، نوار حاوی دکمه ی undo با انیمیشن از صفحه محو می شود.

یک پروژه ی جدید به نام com.vogella.android.userinterface.undo بر اساس قالب آماده BlankTemplate (template) ایجاد کنید.

فایل layout زیر را برای activity خود ایجاد کنید. این فایل با استفاده از FrameLayout دو بخش مجزای رابط کاربری را نمایش می دهد. نواری که دکمه را به نمایش می گذارد، در ابتدا مخفی می باشد. دکمه برای عکس خود از یک drawable استفاده می کند.

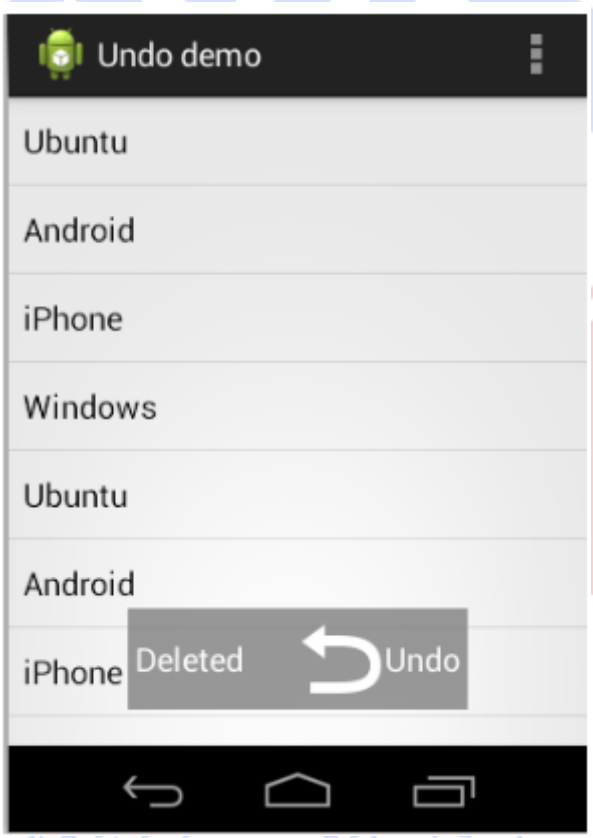
```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent" >
    <RelativeLayout
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       tools:context=".MainActivity" >
       <ListView
          android:id="@+id/listview"
          android:layout_width="match_parent"
          android:layout_height="match_parent" >
       </ListView>
    </RelativeLayout>
    <LinearLayout
       android:id="@+id/undobar"
       android:visibility="gone"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_gravity="bottom|center_horizontal"
       android:layout_margin="20dp"
       android:alpha="100"
       android:background="#808080"
       android:dividerPadding="11dp"
       android:padding="4dp" >
       <TextView
          android:id="@+id/undobar_message"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:text="Deleted"      android:textAppearance="?android:attr/textAppearanceMedium"
          android:textColor="#fff" />
       <Button
          android:id="@+id/undobar_button"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:layout_marginLeft="30dp"
          android:onClick="onClick"
          android:background="#808080"
          android:drawableLeft="@drawable/ic_undobar_undo"
          android:text="Undo"      android:textAppearance="?android:attr/textAppearanceMedium"
          android:textColor="#fff" />
    </LinearLayout>
</FrameLayout>
```

کلاس activity را به صورت زیر ویرایش کنید. ویزارد پروژه خود یک المان ActionBar تولید می کند. این المان در کد زیر بکار گرفته شده است. در صورت تمایل می توانید ActionBar دلخواه خود را ایجاد کنید.

```
package com.vogella.android.userinterface.undo;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;
import com.vogella.android.actionbar.undo.R;
public class MainActivity extends Activity {
      private View viewContainer;
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
            ListView l = (ListView) findViewById(R.id.listview);
            String[] values = new String[] { "Ubuntu", "Android", "iPhone","Windows", "Ubuntu", "Android",
"iPhone", "Windows"};
            ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1, values);
            viewContainer = findViewById(R.id.undobar);
            l.setAdapter(adapter);
      }
      @Override
      public boolean onCreateOptionsMenu(Menu menu) {
            // Inflate the menu; this adds items to the action bar if it is present.
            getMenuInflater().inflate(R.menu.activity_main, menu);
            return true;
      }
      @Override
      public boolean onOptionsItemSelected(MenuItem item) {
            showUndo(viewContainer);
            return true;
      }
      public void onClick(View view) {
            Toast.makeText(this, "Deletion undone", Toast.LENGTH_LONG).show();
            viewContainer.setVisibility(View.GONE);
      }
      public static void showUndo(final View viewContainer) {
            viewContainer.setVisibility(View.VISIBLE);
            viewContainer.setAlpha(1);
        viewContainer.animate().alpha(0.4f).setDuration(5000)
                        .withEndAction(new Runnable() {
```

```
                    @Override
                    public void run() {viewContainer.setVisibility(View.GONE);
                    }
                });
    }
}
```

با انتخاب المان موجود در action bar، نواری حامل دکمه ی Undo به مدت 5 ثانیه نمایش داده می شود.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-25-بهینه سازی کارایی و سرعت اپلیکیشن

نمونه ی زیر نسخه ی بهینه از adapter مثال قبلی که سرعت و کارایی نسبتا بهتری دارد را پیاده سازی می کند.

ابتدا کلاس زیر به نام MyPerformanceArrayAdapter را ایجاد کنید.

```
package de.vogella.android.listactivity;
import android.app.Activity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;
public class MyPerformanceArrayAdapter extends ArrayAdapter<String> {
    private final Activity context;
    private final String[] names;
    static class ViewHolder {
        public TextView text;
        public ImageView image;
    }
    public MyPerformanceArrayAdapter(Activity context, String[] names) {
        super(context, R.layout.rowlayout, names);
        this.context = context;
        this.names = names;
    }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        View rowView = convertView;
        // reuse views
        if (rowView == null) {
            LayoutInflater inflater = context.getLayoutInflater();
            rowView = inflater.inflate(R.layout.rowlayout, null);
            // configure view holder
            ViewHolder viewHolder = new ViewHolder();
            viewHolder.text = (TextView) rowView.findViewById(R.id.TextView01);
            viewHolder.image = (ImageView) rowView
                    .findViewById(R.id.ImageView01);
            rowView.setTag(viewHolder);
        }
        // fill data
        ViewHolder holder = (ViewHolder) rowView.getTag();
        String s = names[position];
        holder.text.setText(s);
        if (s.startsWith("Windows7") || s.startsWith("iPhone")
            || s.startsWith("Solaris")) {holder.image.setImageResource(R.drawable.no);
        } else {holder.image.setImageResource(R.drawable.ok);
```

```
        }
        return rowView;
    }
}
```

Adapter جدید خود را به activity اضافه کنید. با اجرای برنامه متوجه می شوید که ظاهر آن تغییری نکرده، با این وجود اپلیکیشن به ویژه در خواندن و کار با dataset های بزرگ بسیار سریع تر عمل می کند.

```
package de.vogella.android.listactivity;
import android.app.ListActivity;
import android.os.Bundle;
public class MyListActivity extends ListActivity {
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        String[] values = new String[] { "Android", "iPhone", "WindowsMobile",
                "Blackberry", "WebOS", "Ubuntu", "Windows7", "Max OS X",
                "Linux", "OS/2" };
        setListAdapter(new MyPerformanceArrayAdapter(this, values));
    }
}
```

26-2-آموزش نحوه ی نمایش دو آیتم در یک ListView

می توانید کلاس SimpleAdapter را جهت نمایش داده های دو المان بکار ببرید. این کلاس به آرایه ای از رشته ها (داده های متغیر آرایه ای from) احتیاج دارد که در آن فیلدهایی که قرار است داده ها از بیرون وارد آن شوند (input data) تعریف شده است. کلاس مزبور همچنین به آرایه ای از اعداد صحیح (int array) نیاز دارد که این اعداد ID یا شناسه ی منحصربفرد widget هایی در فایل layout می باشند که فیلدها در سطر به آن نگاشت (map) می شوند.

داده ها در این برنامه لیستی از نوع Map ها (خانواده ای برای نگهداری جفت های کلید و مقدار) است. Map به ازای هر فیلد در From یک مقدار تعریف می کند.

در زیر مثالی را می بینید که برای تنظیم ظاهر سطر مربوطه از یک layout آماده و از پیش ساخته شده ی اندروید استفاده می کند.

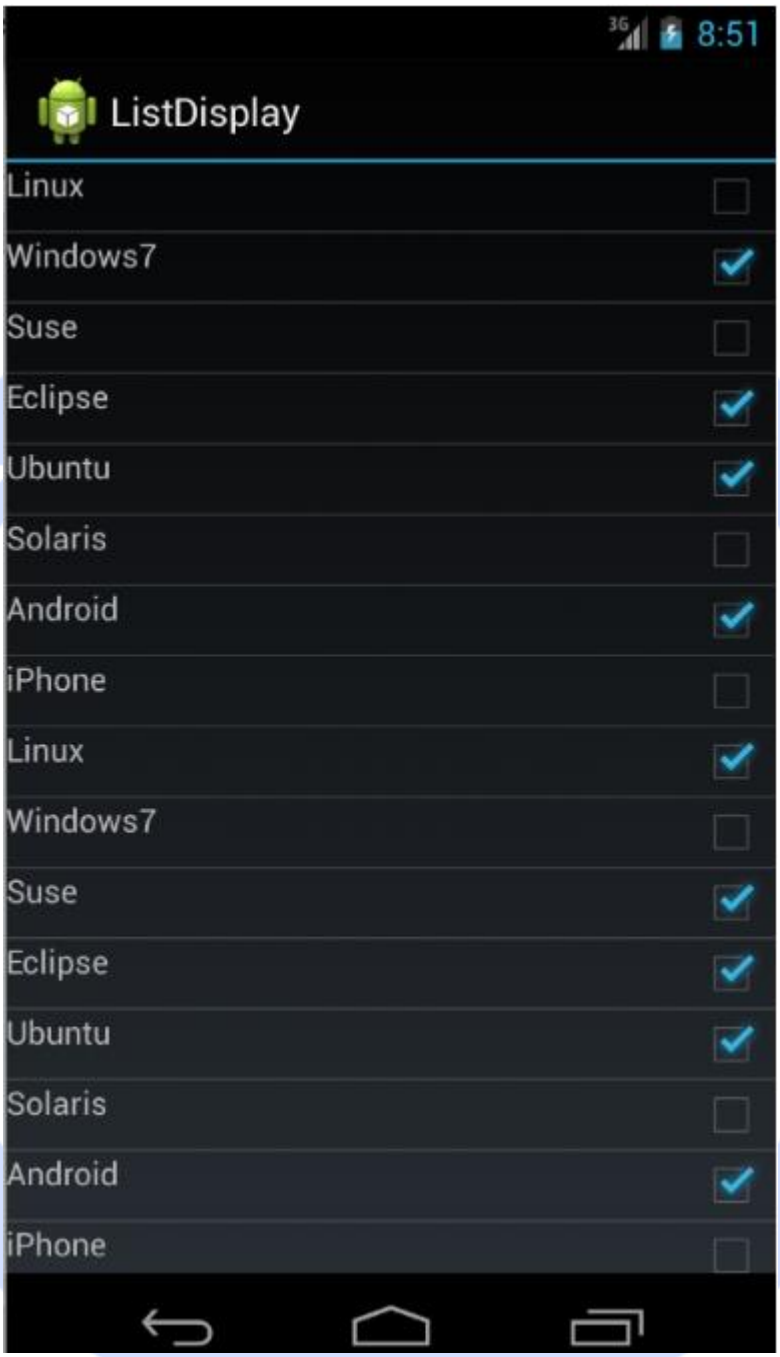
```
package de.vogella.android.listactivity;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import android.app.ListActivity;
```

```
import android.os.Bundle;
import android.widget.SimpleAdapter;
public class MyTwoListItemsActivity extends ListActivity {
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		ArrayList<Map<String, String>> list = buildData();
		String[] from = { "name", "purpose" };
		int[] to = { android.R.id.text1, android.R.id.text2 };
		SimpleAdapter adapter = new SimpleAdapter(this, list,
				android.R.layout.simple_list_item_2, from, to);
		setListAdapter(adapter);
	}
	private ArrayList<Map<String, String>> buildData() {
		ArrayList<Map<String, String>> list = new ArrayList<Map<String, String>>();
		list.add(putData("Android", "Mobile"));
		list.add(putData("Windows7", "Windows7"));
		list.add(putData("iPhone", "iPhone"));
		return list;
	}
	private HashMap<String, String> putData(String name, String purpose) {
		HashMap<String, String> item = new HashMap<String, String>();
		item.put("name", name);
		item.put("purpose", purpose);
		return item;
	}
}
```

2-27-انتخاب چندین آیتم در ListView

1-27-2-تعامل بین model و ListView

پس از ایجاد لیست، قاعدتا می خواهید آیتم هایی را در آن انتخاب نمایید. از آنجایی که (به هنگام پیمایش به پایین و بالای لیست) سطرهای ListView بازیافت شده (recycle) و مجددا مورد استفاده قرار می گیرند، نمی توان انتخاب (اطلاعات مربوط به وضعیت و انتخاب) را در سطح View ذخیره کرد.



انتخاب آیتم ها در لیست (و تبادل داده بین UI و Model)، تنها یک نمونه از تعامل بین سطر (UI) و داده های متناظر در model است و می توان سناریوهای دیگری را تصور کرد.

به منظور ماندگارسازی و ذخیره ی دائمی انتخاب، شما بایستی data model خود را با اطلاعات مربوط به وضعیت انتخاب شده بروز رسانی نمایید.

به منظور بروز رسانی data model در ListView، ابتدا می بایست کلاس Adapter اختصاصی خود را تعریف نمایید. در سطح این کلاس، شما یک listener به View متصل می کنید که وظیفه ی انتخاب المان متناظر در model را بر عهده دارد. پس از انتخاب المان مورد نظر در لیست، اطلاعات المان مربوطه را در model بروز رسانی می کنید. حال برای بروز رسانی و دسترسی به اطلاعات المان متناظر در model، آن را به صورت یک تگ به view اضافه می کنید.

مثال زیر نشان می دهد چگونه از آبجکت های استاندارد جاوا استفاده کرده و چگونه بین Views و model تعاملو ارتباط دو طرفه برقرار نمایید.

تمرین آموزشی: تعامل و ارتباط دو طرفه بین Domain model و سطرهای ListView

در این تمرین نیز پروژه ی قبلی، de.vogella.android.listactivity، را ادامه می دهید.

ابتدا Model زیر را ایجاد نمایید. این کلاس اسم و اطلاعات را، در صورتی که المان در حال حاضر انتخاب شده باشد، در خود ذخیره می نماید.


```
package de.vogella.android.listactivity;
public class Model {
	private String name;
	private boolean selected;
	public Model(String name) {
		this.name = name;
		selected = false;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
        this.name = name;
    }
    public boolean isSelected() {
        return selected;
    }
    public void setSelected(boolean selected) {
        this.selected = selected;
    }
}
```

یک فایل layout به نام rowbuttonlayout.xml مانند زیر ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="wrap_content" >
  <TextView
    android:id="@+id/label"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@+id/label"
    android:textSize="30px" >
  </TextView>
  <CheckBox
    android:id="@+id/check"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentRight="true"
    android:layout_marginLeft="4px"
    android:layout_marginRight="10px" >
  </CheckBox>
</RelativeLayout>
```

یک کلاس Adapter با پیاده سازی زیر ایجاد نمایید. کلاس نام برده یک listener (گوش فراخوان به رخدادها) بر روی آبجکت Checkbox اضافه می کند. به دنبال انتخاب checkbox، داده های مربوطه در model نیز بروز رسانی می شوند. Checkbox به وسیله ی متد ()getTag، داده های المان متناظر خود در data model را بازیابی می کند.

```
package de.vogella.android.listactivity;
import java.util.List;
import android.app.Activity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.CheckBox;
import android.widget.CompoundButton;
import android.widget.TextView;
public class InteractiveArrayAdapter extends ArrayAdapter<Model> {
    private final List<Model> list;
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    private final Activity context;
    public InteractiveArrayAdapter(Activity context, List<Model> list) {
        super(context, R.layout.rowbuttonlayout, list);
        this.context = context;
        this.list = list;
    }
    static class ViewHolder {
        protected TextView text;
        protected CheckBox checkbox;
    }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        View view = null;
        if (convertView == null) {
            LayoutInflater inflator = context.getLayoutInflater();
            view = inflator.inflate(R.layout.rowbuttonlayout, null);
            final ViewHolder viewHolder = new ViewHolder();
            viewHolder.text = (TextView) view.findViewById(R.id.label);
            viewHolder.checkbox = (CheckBox) view.findViewById(R.id.check);
            viewHolder.checkbox
                    .setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {
                        @Override
                        public void onCheckedChanged(CompoundButton buttonView,
                                boolean isChecked) {
                            Model element = (Model) viewHolder.checkbox
                                    .getTag();
                            element.setSelected(buttonView.isChecked());
                        }
                    });
            view.setTag(viewHolder);
            viewHolder.checkbox.setTag(list.get(position));
        } else {
            view = convertView;
            ((ViewHolder) view.getTag()).checkbox.setTag(list.get(position));
        }
        ViewHolder holder = (ViewHolder) view.getTag();
        holder.text.setText(list.get(position).getName());
        holder.checkbox.setChecked(list.get(position).isSelected());
        return view;
    }
}
```

حال کد activity را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.listactivity;
import java.util.ArrayList;
import java.util.List;
import android.app.ListActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
```

```
public class MyList extends ListActivity {
    /** Called when the activity is first created. */
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        // create an array of Strings, that will be put to our ListActivity
        ArrayAdapter<Model> adapter = new InteractiveArrayAdapter(this,
                getModel());
        setListAdapter(adapter);
    }
    private List<Model> getModel() {
        List<Model> list = new ArrayList<Model>();
        list.add(get("Linux"));
        list.add(get("Windows7"));
        list.add(get("Suse"));
        list.add(get("Eclipse"));
        list.add(get("Ubuntu"));
        list.add(get("Solaris"));
        list.add(get("Android"));
        list.add(get("iPhone"));
        // Initially select one of the items
        list.get(1).setSelected(true);
        return list;
    }
    private Model get(String s) {
        return new Model(s);
    }
}
```

برنامه را اجرا کرده و آیتم ها را علامت گذاری کنید. با توجه به کد اپلیکیشن، تمامی تغییرات در UI اپلیکیشن، بلافاصله در model منعکس شده و در واقع آبجکت model به صورت خودکار با داده های جدید بروز رسانی می شود.

2-28-پیاده سازی یک ListView کشویی و قابل بسط

ExpandableListView_2_28_1

ExpandableListView مشابه ListView است با این تفاوت که به شما این امکان را می دهد تا برای المان ها گروه و جزئیات بیشتر ایجاد کنید. برای این منظور group.ExpandableListView انتظار یک آبجکت adapter از جنس BaseExpandableListAdapter را دارد.

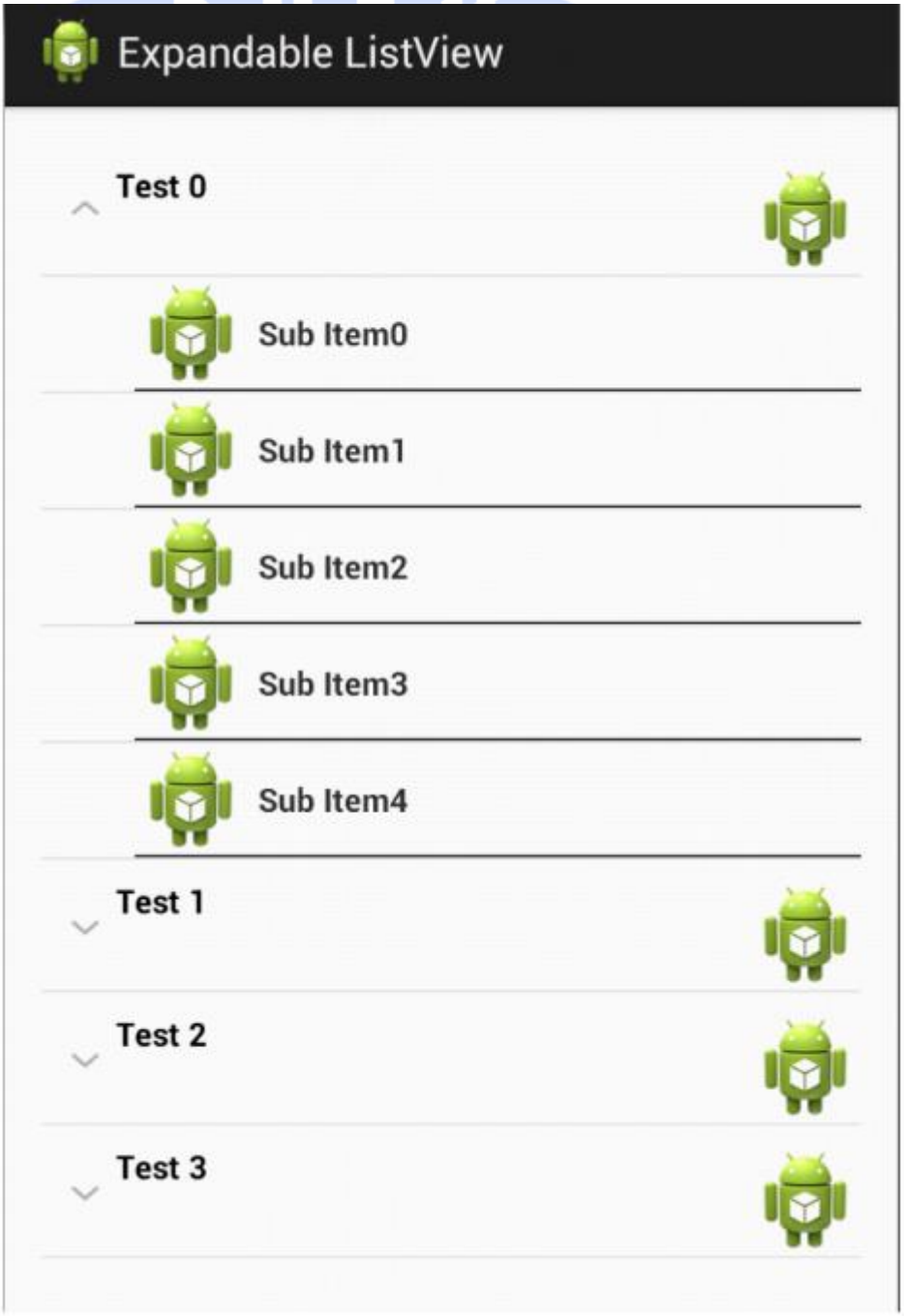
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در این سناریو بایستی دو layout تعریف کنید: یکی برای گروه و دیگری برای سطری که جزئیات را نمایش می دهد.

مثالی از پیاده سازی یک ExpandableListView

در این مثال، یک لیست کشویی یا ExpandableListView مشابه تصویر زیر ایجاد خواهید نمود.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-28-2-پیاده سازی یک ExpandableListView

یک پروژه جدید به نام com.vogella.android.listview.expandable همراه با activity ای به نام MainActivity ایجاد کنید.

فایل های layout زیر را ایجاد نمایید. ابتدا فایل layout/activity_main.xml را تعریف کنید.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/LinearLayout1"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   tools:context=".MainActivity" >
   <ExpandableListView
      android:id="@+id/listView"
      android:layout_width="match_parent"
      android:layout_height="wrap_content" >
   </ExpandableListView>
</LinearLayout>
```

سپس فایل layout/listrow_group.xml را ایجاد نمایید.

```
<CheckedTextView xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/textView1"
   android:layout_width="wrap_content"
   android:layout_height="?android:attr/listPreferredItemHeight"
   android:layout_marginLeft="8dp"
   android:drawableRight="@drawable/ic_launcher"
   android:gravity="left"
   android:paddingLeft="32dp"
   android:paddingTop="8dp"
   android:text="Test"
   android:textSize="14sp"
   android:textAlignment="textEnd"
   android:textStyle="bold" />
```

آخرین فایل layout ای که باید تعریف کنید، layout/listrow_details.xml می باشد.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="40dp"
```

```
    android:clickable="true"
    android:orientation="vertical"
    android:paddingLeft="40dp"
    tools:context=".MainActivity" >
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:drawableLeft="@drawable/ic_launcher"
        android:drawablePadding="5dp"
        android:gravity="center_vertical"
        android:text="@string/hello_world"
        android:textSize="14sp"
        android:textStyle="bold" >
    </TextView>
    <View
        android:layout_width="match_parent"
        android:layout_height="1dp"
        android:background="@android:color/black" />
</LinearLayout>
```

کلاسی با پیاده سازی زیر را تعریف نمایید که به عنوان domain model آبجکت ExpandableListView شما ایفای نقش می کند.

```
package com.vogella.android.listview.expandable;
import java.util.ArrayList;
import java.util.List;
public class Group {
        public String string;
        public final List<String> children = new ArrayList<String>();
        public Group(String string) {
                this.string = string;
        }
}
```

حال کلاس adapter با پیاده سازی زیر را ایجاد نموده و کد کلاس activity را به صورت زیر ویرایش نمایید.

```
package com.vogella.android.listview.expandable;
import android.app.Activity;
import android.util.SparseArray;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup;
import android.widget.BaseExpandableListAdapter;
import android.widget.CheckedTextView;
import android.widget.TextView;
import android.widget.Toast;
```

```
public class MyExpandableListAdapter extends BaseExpandableListAdapter {
    private final SparseArray<Group> groups;
    public LayoutInflater inflater;
    public Activity activity;
    public MyExpandableListAdapter(Activity act, SparseArray<Group> groups) {
        activity = act;
        this.groups = groups;
        inflater = act.getLayoutInflater();
    }
    @Override
    public Object getChild(int groupPosition, int childPosition) {
        return groups.get(groupPosition).children.get(childPosition);
    }
    @Override
    public long getChildId(int groupPosition, int childPosition) {
        return 0;
    }
    @Override
    public View getChildView(int groupPosition, final int childPosition,
            boolean isLastChild, View convertView, ViewGroup parent) {
        final String children = (String) getChild(groupPosition, childPosition);
        TextView text = null;
        if (convertView == null) {
            convertView = inflater.inflate(R.layout.listrow_details, null);
        }
        text = (TextView) convertView.findViewById(R.id.textView1);
        text.setText(children);
        convertView.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(activity, children,
                        Toast.LENGTH_SHORT).show();
            }
        });
        return convertView;
    }
    @Override
    public int getChildrenCount(int groupPosition) {
        return groups.get(groupPosition).children.size();
    }
    @Override
    public Object getGroup(int groupPosition) {
        return groups.get(groupPosition);
    }
    @Override
    public int getGroupCount() {
        return groups.size();
    }
    @Override
    public void onGroupCollapsed(int groupPosition) {
        super.onGroupCollapsed(groupPosition);
    }
```

```
    @Override
    public void onGroupExpanded(int groupPosition) {
        super.onGroupExpanded(groupPosition);
    }
    @Override
    public long getGroupId(int groupPosition) {
        return 0;
    }
    @Override
    public View getGroupView(int groupPosition, boolean isExpanded,
            View convertView, ViewGroup parent) {
        if (convertView == null) {
            convertView = inflater.inflate(R.layout.listrow_group, null);
        }
        Group group = (Group) getGroup(groupPosition);
        ((CheckedTextView) convertView).setText(group.string);
        ((CheckedTextView) convertView).setChecked(isExpanded);
        return convertView;
    }
    @Override
    public boolean hasStableIds() {
        return false;
    }
    @Override
    public boolean isChildSelectable(int groupPosition, int childPosition) {
        return false;
    }
}
package com.vogella.android.listview.expandable;
import java.util.ArrayList;
import android.app.Activity;
import android.os.Bundle;
import android.util.SparseArray;
import android.view.Menu;
import android.widget.ExpandableListView;
public class MainActivity extends Activity {
    // more efficient than HashMap for mapping integers to objects
    SparseArray<Group> groups = new SparseArray<Group>();
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        createData();
        ExpandableListView listView = (ExpandableListView) findViewById(R.id.listView);
        MyExpandableListAdapter adapter = new MyExpandableListAdapter(this,
                groups);
        listView.setAdapter(adapter);
    }
    public void createData() {
        for (int j = 0; j < 5; j++) {
            Group group = new Group("Test " + j);
            for (int i = 0; i < 5; i++) {
```

```
                group.children.add("Sub Item" + i);
            }
            groups.append(j, group);
        }
    }
}
```

2-29-آموزش متفرقه

1-29-2-تعریف کردن یک گوش فراخوان به کلیک طولانی مدت (LongItemClickListener) به آیتم های لیست

بد نیست یک LongItemClickListener به وسیله ی setOnItemLongClickListener() برای آیتم های لیست خود تعریف کنید که با کلیک طولانی کاربر بر روی آیتم های لیست صدا خورده می شود.

```
package de.vogella.android.listactivity;
import android.widget.AdapterView.OnItemLongClickListener;
public class MyList extends Activity {
    /** Called when the activity is first created. */
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        // configure your list view as before
        // ListView is assigned to local variable list...
        list.setOnItemLongClickListener(new OnItemLongClickListener() {
            @Override
            public boolean onItemLongClick(AdapterView<?> parent, View view,
                    int position, long id) {
                Toast.makeText(MyList.this,
                        "Item in position " + position + " clicked",
                        Toast.LENGTH_LONG).show();
                // Return true to consume the click event. In this case the
                // onListItemClick listener is not called anymore.
                return true;
            }
        });
    }
}
```

2-29-2-اضافه کردن Header و Footer به لیست

می توانید المان های دلخواه خود را به لیست اضافه نمایید. به عنوان مثال می توانید یک layout یا قالب با این چیدمان تعریف نمایید: دو text view تعریف کرده و یک لیست بین این دو قرار

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

دهید. در این نوع چیدمان یک text view در بالای لیست و در بخش header صفحه و دیگری در پایین text view و در بخش footer قابل مشاهده می باشد. برای نمایش header یا footer در ابتدا و انتهای لیست، کافی است دو متد addHeaderView() یا addFooterView() را در سطح کلاس ListView (بر روی آبجکت listView در قطعه کد زیر) تعریف نمایید.

```
// configuration as before
ListView listView = (ListView) findViewById(R.id.list);
View header = getLayoutInflater().inflate(R.layout.header, null);
View footer = getLayoutInflater().inflate(R.layout.footer, null);
listView.addHeaderView(header);
listView.addFooterView(footer);
listView.setAdapter(new ArrayAdapter<String>(this,
        android.R.layout.simple_list_item_single_choice,
        android.R.id.text1, names));
    }
}
```

SimpleCursorAdapter-30-2

چنانچه لازم است داده های برنامه ی خود را از Content provider دریافت کنید یا آن را مستقیم از دیتابیس بخوانید، در آن صورت می توانید با استفاده از کلاس SimpleCursorAdapter اطلاعات مورد نیاز لیست (ListView) خود را مشخص نمایید (داده ها را از cursor به view ها متصل نمایید). توضیحات زیر برای شما شرح می دهد چگونه می توان بهcontent provider Contacts (یک دیتابیس مقایس پذیر با اطلاعات مربوط به مخاطبین یا contacts تعریف می کند) دسترسی داشت و اطلاعات مورد نیاز لیست را در اختیار آن قرار داد (یادآوری: content provider به شما اجازه می دهد تا داده های خود را در نقطه ی مرکزی برای دسترسی اپلیکیشن هایی که به آن نیاز دارند به اشتراک بگذارید).

یک پروژه ی جدید اندروید به نام "de.vogella.android.listactivity.cursor" به همراه activity ای به نام MyListActivity تعریف نمایید. کلاس MyListActivity را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.listactivity.cursor;
import android.app.ListActivity;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.provider.ContactsContract;
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
import android.widget.ListAdapter;
import android.widget.SimpleCursorAdapter;
public class MyListActivity extends ListActivity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Cursor mCursor = getContacts();
        startManagingCursor(mCursor);
        // now create a new list adapter bound to the cursor.
        // SimpleListAdapter is designed for binding to a Cursor.
        ListAdapter adapter = new SimpleCursorAdapter(this, // Context.
                android.R.layout.two_line_list_item, // Specify the row template
                                                     // to use (here, two
                                                     // columns bound to the
                                                     // two retrieved cursor
                                                     // rows).
                mCursor, // Pass in the cursor to bind to.
                // Array of cursor columns to bind to.
                new String[] { ContactsContract.Contacts._ID,
                        ContactsContract.Contacts.DISPLAY_NAME },
                // Parallel array of which template objects to bind to those
                // columns.
                new int[] { android.R.id.text1, android.R.id.text2 });
        // Bind to our new adapter.
        setListAdapter(adapter);
    }
    private Cursor getContacts() {
        // Run query
        Uri uri = ContactsContract.Contacts.CONTENT_URI;
        String[] projection = new String[] { ContactsContract.Contacts._ID,
                ContactsContract.Contacts.DISPLAY_NAME };
        String selection = ContactsContract.Contacts.IN_VISIBLE_GROUP + " = '"
                + ("1") + "'";
        String[] selectionArgs = null;
        String sortOrder = ContactsContract.Contacts.DISPLAY_NAME
                + " COLLATE LOCALIZED ASC";
        return managedQuery(uri, projection, selection, selectionArgs,
                sortOrder);
    }
}
```

لازم است به اپلیکیشن خود مجوز خواندن اطلاعات contacts را بدهید (مقدار خصیصه ی android:name: را در فایل تنظیمات یا manifest بر روی "android.permission.READ_CONTACTS" قرار دهید).

31-2-سایر کتابخانه های کد باز (Open Source libraries)

کاربر مجبور است برای بروز رسانی ActionBar بر روی دکمه ی refresh کلیک کند که ممکن است گاهی تجربه ی کاربری ضعیفی را به دنبال داشته باشد. Chris Banes یک کتابخانه ی کد باز پیاده سازی کرده که الگوی pull to refresh را برای ListView در اپلیکیشن شما implement می کند. برای دسترسی به آن می توانید به این آدرس مراجعه نمایید: https://github.com/chrisbanes/Android-PullToRefresh.

علاوه بر آن شاید لازم باشد با پیاده سازی قابلیت swipe در اپلیکیشن، به کاربر این امکان را بدهید که آیتم ها را با کشیدن انگشت بر روی نمایشگر از لیست حذف کند. Roman Nurik برای این منظور یک کتابخانه ی آماده نوشته و Jake Wharton این کتابخانه را برای پشتیبانی از ورژن های قبلی کتابخانه های اندروید (و افزودن قابلیت های جدید به API های قدیمی تر) backport نموده است. برای دسترسی به هریک می توانید به ترتیب به آدرس های https://github.com/romannurik/android-swipetodismiss و https://github.com/JakeWharton/SwipeToDismissNOA مراجعه نمایید.

بخش هفتم :

2-32-ثبت وقایع و گزارش گیری (Logging) در اندروید

ثبت وقایع و گزارش گیری با استفاده از ابزار Android logging/LogCat

آموزش حاضر نحوه ی ایجاد و استفاده از دستورات log در اپلیکیشن های اندروید را شرح می دهد.

1-33-2-سیستم گزارش گیری اندروید (log system)

اندروید برای ثبت گزارشات (جهت اشکال زدایی پروژه) از یک سیستم مرکزی بهره می گیرد. البته برنامه نویس این امکان را نیز دارد که پیغام های دلخواه خود را تنظیم کند. ابزاری که برای توسعه و طراحی اندروید در اختیار شما قرار می گیرند به شما این امکان را می دهند تا با تعریف فیلتر تنها از دستوراتی که کارایی مد نظر شما را ارائه می دهد، استفاده نمایید.

2-33-2-ایجاد دستورات گزارش گیری (log statement)

به منظور استفاده از دستورات گزارش گیری (log statement)، می بایست از کلاس android.util.Log و متدهای زیر استفاده نمایید.

- Log.v()
- Log.d()
- Log.i()
- Log.w()
- Log.e()
- Log.wtf()

این توابع بر اساس اولویت و میزان کاربرد فهرست شده اند به طوری که Log.i () از کم ترین میزان اهمیت و کاربرد برخوردار است. اولین پارامتر ورودی این متدها category و دومین آرگومان پیغام مربوطه می باشد.

معمولا برنامه نویس ابتدا یک الگو پیاده سازی Constants (interface) در اپلیکیشن اندرویدی خود تعریف می کند و سپس log category را در قالب یک فیلد در بدنه ی آن interface ارائه می دهد.

```
// package declaration left out, use your application package
public interface Constants {
    String LOG = "com.vogella.testapp";
}
```

اندروید به توسعه دهندگان توصیه می کند که از داشتن کدهای مربوط به گزارشات و وقایع (logging) در اپلیکیشن نصب شده (deployed) خودداری کنند. مجموعه ابزار طراحی و توسعه ADT برای این منظور BuildConfig.DEBUG (flag) را ارائه می دهد. زمانی که اپلیکیشن اندرویدی خود را به صورت یک خروجی امضا شده (export)، نهایی و آماده برای نصب منتشر می کنید، مقدار این flag به صورت خودکار بر روی false تنظیم می شود. در طول توسعه ی اپلیکیشن، مقدار این flag بر روی true قرار می گیرد، بدین معنا که برنامه نویس می تواند دستورات و کدهای مربوط به گزارش از برنامه را حین نوشتن کدهای برنامه مشاهده کند.

مثال زیر نشان می دهد چگونه یک پیغام خطا را بنویسید.

```
if (BuildConfig.DEBUG) {
    Log.e(Constants.TAG, "onCreate called");
}
```

بخش هشتم :

2-34-پیاده سازی dialog در اپلیکیشن به وسیله ی DialogFragments

نمایش پنجره ی محاوره/dialog به وسیله ی DialogFragments

این آموزش نحوه ی پیاده سازی و استفاده از dialog یا پنجره ی محاوره ای (که به صورت کادر شناور بر روی پنجره ی activity نمایش داده می شود) در اپلیکیشن های اندرویدی را شرح می دهد.

پروژه ی این مبحث داخل محیط کاری Android Studio نوشته شده و مبتنی بر ویرایش 5.0 سیستم عامل اندروید می باشد.

1-34-2-به نمایش گذاشتن dialog در برنامه

جهت به نمایش گذاشتن یک کادر محاوره ای در Activity اپلیکیشن، بایستی نمونه ای از کلاس DialogFragment را پیاده سازی نمایید. DialogFragment در واقع یک fragment (آبجکتی که تکه ای از یک activity کل را تشکیل می دهد) است که به برنامه نویس این امکان را می دهد تا پنجره ی محاوره ای را به صورت شناور بر روی پنجره ی اصلی که activity اپلیکیشن مربوطه می باشد، به نمایش بگذارد.

به منظور به نمایش گذاشتن dialog، می توانید یا یک کلاس dialog از پیش پیاده سازی شده در fragment را باز گردانی نمایید و یا یک layout اختصاصی برای آن تعریف کنید که به هنگام اجرا برنامه فراخوانی و به نمایش گذاشته می شود.

2-34-2-استفاده از dialog های آماده و از قبل موجود

DialogFragment برنامه شما می تواند متد onCreateDialog را پیاده سازی نموده و dialog از پیش موجود را بازگردانی کند. کلاس Dialog، در واقع کلاس پایه یا پدر است که برای پیاده سازی dialog باید از آن ارث بری نمایید.

برنامه نویس برای نمایش دادن dialog در اپلیکیشن خود اغلب یکی از کلاس های مشتق شده از آن همچون AlertDialog، ProgressDialog، DatePickerDialog یا TimePickerDialog را مورد استفاده قرار می دهد.

اندروید همچنین یک ProgressDialog ارائه می دهد که به راحتی با فراخوانی متد ProgressDialog.open() می توان آن را باز کرده و به نمایش گذاشت.

3-34-2-تعریف layout اختصاصی برای DialogFragment

جهت ایجاد dialog اختصاصی خود، لازم است یک فایل layout ویژه ی آن تعریف نمایید. این فایل layout را می توان inflate نموده و با فراخوانی متد onCreateView() از fragment مربوطه بازگردانی نمود.

4-34-2-تعامل DialogFragment با activity

آبجکت DialogFragment درست مشابه دیگر fragment ها با activity مربوطه ارتباط برقرار کرده و اطلاعاتی را رد و بدل می کند. یک روش صحیح و بهینه این است که fragment یک interface معرفی کرده و activity مورد نظر این interface را پیاده سازی می کند. fragment سپس می تواند توسط activity صدا خورده شود، بدون اینکه اطلاعی از جزئیات پیاده سازی activity داشته باشد.

تمرین: پیاده سازی کلاس DialogFragment

در پروژه ی زیر نحوه ی استفاده از کلاس DialogFragment در activity را شرح خواهیم داد. این مثال هم از dialog از پیش ساخته و آماده استفاده می کند و هم برای آن یک layout اختصاصی تعریف می نماید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

5-34-2-ایجاد پروژه و فایل های layout

ابتدا یک پروژه ی اندروید به نام com.android.vogella.com.dialogfragmentexample ایجاد نمایید.

در فایل layout ای که activity فراخوانی می کند، دو دکمه تعریف کنید به طوری که در خصیصه android:onClick (attribute) خود به متد onClick اشاره کنند.

حال یک فایل layout به نام fragment_username.xml با بدنه ی زیر تعریف نمایید.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/edit_name"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_gravity="center"
   android:orientation="vertical">
   <TextView
      android:id="@+id/lbl_your_name"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Enter user name" />
   <EditText
      android:id="@+id/username"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:imeOptions="actionDone"
      android:inputType="text" />
</LinearLayout>
```

6-34-2-تعریف fragment و تنظیم activity

دو کلاس fragment زیر را ایجاد نمایید.

```
package dialogfragmentexample.android.vogella.com.dialogfragmentexample;
import android.app.AlertDialog;
import android.app.Dialog;
import android.app.DialogFragment;
import android.content.DialogInterface;
import android.os.Bundle;
import android.widget.Toast;
public class MyAlertDialogFragment extends DialogFragment {
   @Override
   public Dialog onCreateDialog(Bundle savedInstanceState) {
      return new AlertDialog.Builder(getActivity())
            // set dialog icon
            .setIcon(android.R.drawable.stat_notify_error)
            // set Dialog Title
            .setTitle("Alert dialog fragment example")
```

```
            // Set Dialog Message
            .setMessage("This is a message")
            // positive button
            .setPositiveButton("OK", new DialogInterface.OnClickListener() {
               public void onClick(DialogInterface dialog, int which) {
                  Toast.makeText(getActivity(), "Pressed OK", Toast.LENGTH_SHORT).show();
               }
            })
            // negative button
            .setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
               public void onClick(DialogInterface dialog, int which) {
                  Toast.makeText(getActivity(), "Cancel", Toast.LENGTH_SHORT).show();
               }
            }).create();
   }
}


package dialogfragmentexample.android.vogella.com.dialogfragmentexample;
import android.app.DialogFragment;
import android.os.Bundle;
import android.view.KeyEvent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.view.WindowManager;
import android.widget.EditText;
import android.widget.TextView;
public class MyDialogFragment extends DialogFragment implements TextView.OnEditorActionListener {
   private EditText mEditText;
   public interface UserNameListener {
      void onFinishUserDialog(String user);
   }
   // Empty constructor required for DialogFragment
   public MyDialogFragment() {
   }
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
                  Bundle savedInstanceState) {
      View view = inflater.inflate(R.layout.fragment_username, container);
      mEditText = (EditText) view.findViewById(R.id.username);
      // set this instance as callback for editor action
      mEditText.setOnEditorActionListener(this);
      mEditText.requestFocus();
      getDialog().getWindow().setSoftInputMode(
            WindowManager.LayoutParams.SOFT_INPUT_STATE_VISIBLE);
      getDialog().setTitle("Please enter username");
      return view;
   }
   @Override
   public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
      // Return input text to activity
```

```
        UserNameListener activity = (UserNameListener) getActivity();
        activity.onFinishUserDialog(mEditText.getText().toString());
        this.dismiss();
        return true;
    }
}
```

کد activity خود را به صورت زیر ویرایش نمایید.

```
package dialogfragmentexample.android.vogella.com.dialogfragmentexample;
import android.app.Activity;
import android.app.Fragment;
import android.app.FragmentManager;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;
public class MainActivity extends Activity implements MyDialogFragment.UserNameListener {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    @Override
    public void onFinishUserDialog(String user) {
        Toast.makeText(this, "Hello, " + user, Toast.LENGTH_SHORT).show();
    }
    public void onClick(View view) {
        // close existing dialog fragments
        FragmentManager manager = getFragmentManager();
        Fragment frag = manager.findFragmentByTag("fragment_edit_name");
        if (frag != null) {
            manager.beginTransaction().remove(frag).commit();
        }
        switch (view.getId()) {
            case R.id.showCustomFragment:
                MyDialogFragment editNameDialog = new MyDialogFragment();
                editNameDialog.show(manager, "fragment_edit_name");
                break;
            case R.id.showAlertDialogFragment:
                MyAlertDialogFragment alertDialogFragment = new MyAlertDialogFragment();
                alertDialogFragment.show(manager, "fragment_edit_name");
                break;
        }
    }
}
```

تست اپلیکیشن

اکنون زمانی که اپلیکیشن را اجرا نموده و بر روی دکمه ی مربوطه در UI اپلیکیشن کلیک می کنید، پنجره ی محاوره ای (dialog) حاوی پیغام مورد نظر برای شما به نمایش در می آید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فصل سوم
استفاده از
Fragmant ها و
ActionBar ها

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش اول :

طراحی صفحات تک قطعه و چند قطعه (single-pane & multi-pane) با استفاده از fragment ها در اندروید/ایجاد UIهای انعطاف پذیر به وسیله ی fragment ها

این مبحث برای شما شرح می دهد چگونه می توان با استفاده از کلاس fragment، اپلیکیشن هایی با رابط کاربری مقیاس و انعطاف پذیر ایجاد نمایید.

3-1-Fragment ها

3-1-1-شرح Layout های تک قطعه (single-pane) و چند قطعه (multi-pane)

همان طور که می دانید دستگاه های اندروید دارای نمایشگرهایی با اندازه و تراکم پیکسلی متفاوت هستند.

Panel یا pane در اندروید، عبارت است از یک بخش یا قطعه از کل صفحه (UI) که کاربر با آن تعامل دارد. pane در واقع یک واژه ی کلی است که بیانگر قابلیت اندروید برای پشتیبانی از چندین view در کنار هم و در قالب یک view مرکب (واحد) بوده که ممکن است بسته به اندازه ی فضای موجود در نمایشگر از دستگاه به دستگاهی دیگر متفاوت باشد.

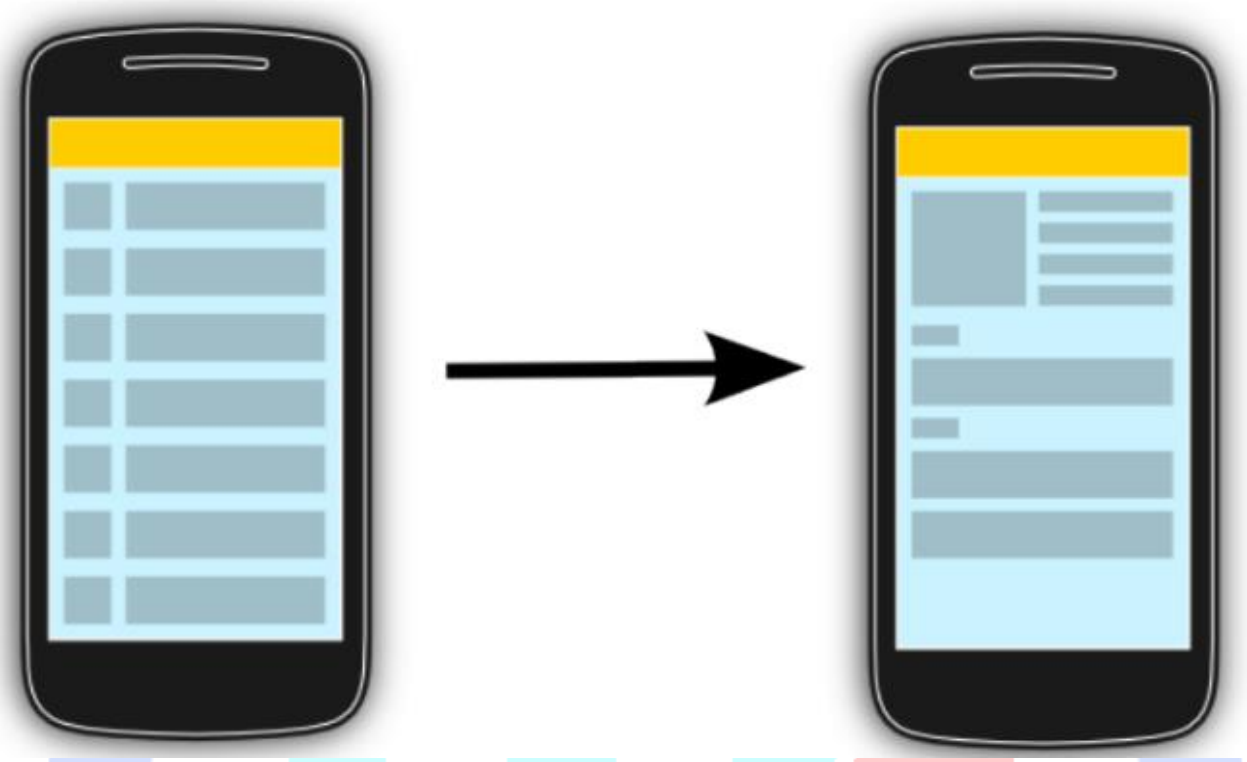
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در صورت عدم وجود فضای کافی، تنها یک panel به نمایش در می آید. از این معمولا تحت عنوان چیدمان یا قالب (layout) تک قطعه نام برده می شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



اما در دستگاه هایی که صفحه ی نمایش عریض تری دارند و در نتیجه فضای بیشتری در دسترس است (همچون تبلت)، طراحی چند قطعه این امکان را برای برنامه فراهم می کند تا چندین panel را همزمان به نمایش بگذارد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-1-3-شرح مفهوم Fragment

Fragment در حقیقت یک کامپوننت UI مستقل در اندروید است که زیرمجموعه ی یک activity می باشد و بخشی از کل صفحه یا فرم را تشکیل می دهد (در نمایشگرهای بزرگ چند fragment و در نمایشگرهای کوچک تنها یک fragment در لحظه نمایش داده شده و صفحه ی قابل مشاهده برای کاربر را تشکیل می دهند). به عبارت دیگر fragment در اندروید به شما اجازه می دهد تا UI انعطاف پذیر و سازگار با انواع نمایشگرها (کوچک و بزرگ) برای برنامه ی خود تعریف نمایید و اجزا رابط کاری اپلیکیشن را در قالب ماژول های مجزا کپسوله سازی نمایید.

Fragment با کپسوله سازی اجزا رابط کاربری و رفتار activity ها در ماژول های مجزا، این امکان را برای اپلیکیشن فراهم می کند تا از این ماژول های مجزا در چندین activity استفاده کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Fragment در بستر یک activity اجرا می شود، با این وجود دارای چرخه ی حیات و رابط کاربری اختصاصی خود است. البته یک fragment می تواند فاقد ظاهر و UI باشد که در اصطلاح headless fragment خوانده می شود.

3-1-3- Fragment ها و دسترسی به Context

Fragment ها از کلاس Context مشتق (subclass) نمی شوند. بنابراین برای دسترسی به activity میزبان بایستی متد getActivity() را صدا بزنید.

3-1-4-مزایای استفاده از fragment

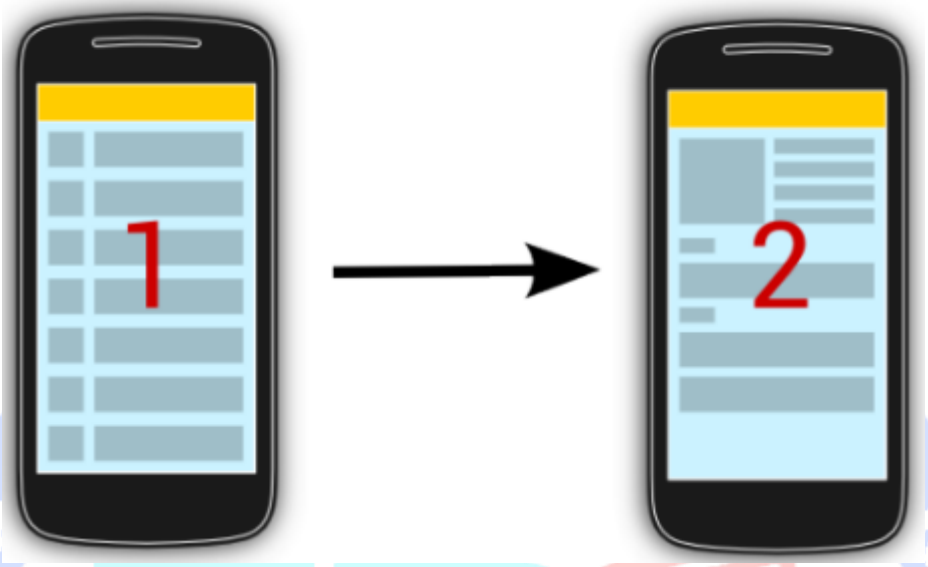
1. fragment ها چرخه ی حیات و رفتار خود را دارند.
2. می توان آن ها را در زمان اجرای اپلیکیشن و به صورت داینامیک کم و زیاد کرد.
3. می توان چندین fragment را با یکدیگر ترکیب کرده و UI های چند قطعه/چند پنجره ای و انعطاف پذیر ایجاد کرد.
4. این امکان نیز وجود دارد که یک fragment را در چندین activity بکار برد.

به کمک fragment می توانید برای دستگاه های کوچک UI یک قطعه ایجاد نموده و برای نمایشگرهای بزرگ رابط کاربری چند پنجره طراحی نمایید. همچنین می توانید با استفاده از fragment هر دو جهت نمایش (landscape= نمای افقی و portrait= عمودی) در گوشی ها را مدیریت نمایید.

یکی از موارد کاربرد fragment در طراحی UI را می توان در ساخت لیست مشاهده کرد. اگر بر روی یک آیتم از لیست کلیک کنید، در صفحه ی تبلت، جزئیات مربوطه در همان صفحه، برای مثال در سمت راست به نمایش در می آیند. اما همین لیست در نمایشگر موبایل، کاربر را به صفحه ی مجزا حاوی جزئیات مرتبط هدایت می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



برای این مبحث اپلیکیشن شما باید دو fragment داشته باشد: main و detail. البته شما می توانید بسته به نیاز خود fragment های بیشتری تعریف نمایید. بعلاوه اپلیکیشن شما می بایست دو activity به شرح زیر داشته باشد: main activity و detailed activity. در نمایشگر تبلت main activity هر دو fragment را در layout خود به نمایش می گذارد. این در حالی است که main activity در صفحه نمایش موبایل، فقط main fragment را در لحظه به نمایش می گذارد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

5-1-3-طراحی اپلیکیشنی با UI انعطاف پذیر و سازگار با نمایشگرهای مختلف به وسیله ی fragment ها

Fragment ها را می توان به صورت static تعریف کنید، بدین معنی که به وسیله ی تگ <fragment> در فایل layout می توانید مشخص کنید که activity از چندین fragment و قطعه تشکیل شود.

همان طور که گفته شد، شما می توانید fragment های یک activity را در زمان اجرا ویرایش کنید (کم یا زیاد کنید) که به آن تعریف به صورت پویا و dynamic definition گفته می شود.

برای اینکه بتوانید با توجه به فضای موجود در نمایشگر خود چند fragment را نمایش دهید، کافی است بر اساس یکی از روش های زیر اقدام نمایید:

1. از یک activity استفاده کنید که در نمایشگرهای بزرگ (تبلت) و کوچک (گوشی همراه) دو fragment را برای کاربر در لحظه نمایش می دهد و در زمان لازم، fragment های قابل مشاهده در activity (صفحه ی جاری) را به هنگام اجرای برنامه تغییر دهید. برای این منظور توصیه می شود دو نمونه از کلاس FrameLayout به عنوان placholder (مکان نگهدار موقتی) در layout خود تعریف نموده، سپس fragment ها را در زمان اجرا به آن دو کلاس اضافه کنید.

- در دستگاه های کوچک همچون گوشی، هر fragment را در activity جدا بگنجانید. برای مثال، اگر UI تبلت از دو fragment در یک activity استفاده می کند و در آن واحد دو قطعه را به نمایش می گذارد، همین activity را برای نمایشگر گوشی استفاده می کنید اما این بار فایل layout مجزا ارائه می دهید که تنها یک fragment را شامل می شود. چنانچه detailed fragment از قبل در نمایشگر حاضر باشد، آنگاه main activity به fragment دستور بروز رسانی خود را می دهد. اگر detail fragment در دسترس نبود، در آن صورت main activity، detail activity را راه اندازی می کند.

اینکه کدام گزینه را انتخاب کنید کاملا به نیاز و مورد استفاده ی شما بستگی دارد. به طور معمول ویرایش fragment در زمان اجرا منعطف تر است، اما پیاده سازی آن کمی پیچیده تر می باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

3-2-تعریف و استفاده از fragment ها

3-2-1-تعریف fragment

به منظور تعریف یک fragment جدید می توانید از کلاس android.app.Fragment و یا یکی از کلاس های فرزند (subclass) آن ارث بری (extend) نمایید. کلاس های فرزندی که می توانید مورد استفاده قرار دهید عبارتند از: ListFragment، DialogFragment، PreferenceFragment یا WebViewFragment. کد زیر یک نمونه پیاده سازی را نمایش می دهد:

```
package com.example.android.rssreader;
import android.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
public class DetailFragment extends Fragment {
   public static final String EXTRA_URL ="url";
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
                    Bundle savedInstanceState) {
      View view = inflater.inflate(R.layout.fragment_rssitem_detail,
          container, false);
      return view;
   }
   @Override
   public void onActivityCreated(Bundle savedInstanceState) {
      super.onActivityCreated(savedInstanceState);
      Bundle bundle = getArguments();
      if (bundle != null) {
         String link = bundle.getString("url");
         setText(link);
      }
   }
      public void setText(String url) {
            TextView view = (TextView) getView().findViewById(R.id.detailsText);
            view.setText(url);
      }
}
```

2-2-3-تعامل اپلیکیشن با fragment ها

برای اینکه بتوان امکان استفاده ی مجدد از fragment ها را افزایش داد و به اصطلاح آن ها را بازیافت نمود، لازم است هر گونه تعامل بین این fragment ها منحصرا از طریق activity میزبان انجام شود. به بیان دیگر، fragment ها نباید مستقیما و بدون واسطه با یکدیگر ارتباط داشته باشند.

برای این منظور fragment می بایست یک interface را به صورت inner و داخل یک کلاس تعریف کند. fragment ایجاب می کند که activity میزبان، این interface را پیاده سازی کند. بدین وسیله شما اطمینان حاصل می کنید که fragment هیچ اطلاعی از activity میزبان (activity که از آن fragment استفاده می کند) خود ندارد. Fragment مورد نظر سپس به وسیله ی متد onAttach() خود بررسی می کند که آیا activity به درستی interface مورد نظر را پیاده سازی می کند یا خیر.

به عنوان مثال، فرض کنید fragment ای دارد که باید مقداری را به activity میزبان (parent) خود ارسال کند. این کار را می توان به صورت زیر پیاده سازی نمود.

```
package com.example.android.rssreader;
import android.app.Activity;
import android.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
public class MyListFragment extends Fragment {
    private OnItemSelectedListener listener;
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
            Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_rsslist_overview,
                container, false);
        Button button = (Button) view.findViewById(R.id.button1);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                updateDetail("fake");
            }
        });
        return view;
    }
```

```
    public interface OnItemSelectedListener {
        public void onRssItemSelected(String link);
    }
    @Override
    public void onAttach(Context context) {
        super.onAttach(context);
        if (context instanceof OnItemSelectedListener) {
            listener = (OnItemSelectedListener) context;
        } else {
            throw new ClassCastException(context.toString()
                    + " must implemenet MyListFragment.OnItemSelectedListener");
        }
    }
    @Override
    public void onDetach() {
        super.onDetach();
        listener = null;
    }
    // may also be triggered from the Activity
    public void updateDetail(String uri) {
        // create a string just for testing
        String newTime = String.valueOf(System.currentTimeMillis());
        // inform the Activity about the change based
        // interface defintion
        listener.onRssItemSelected(newTime);
    }
}
```

3-2-3-ارسال پارامتر به fragment ها

Activity می تواند یک bundle را به عنوان پارامتر به fragment ارسال کند.

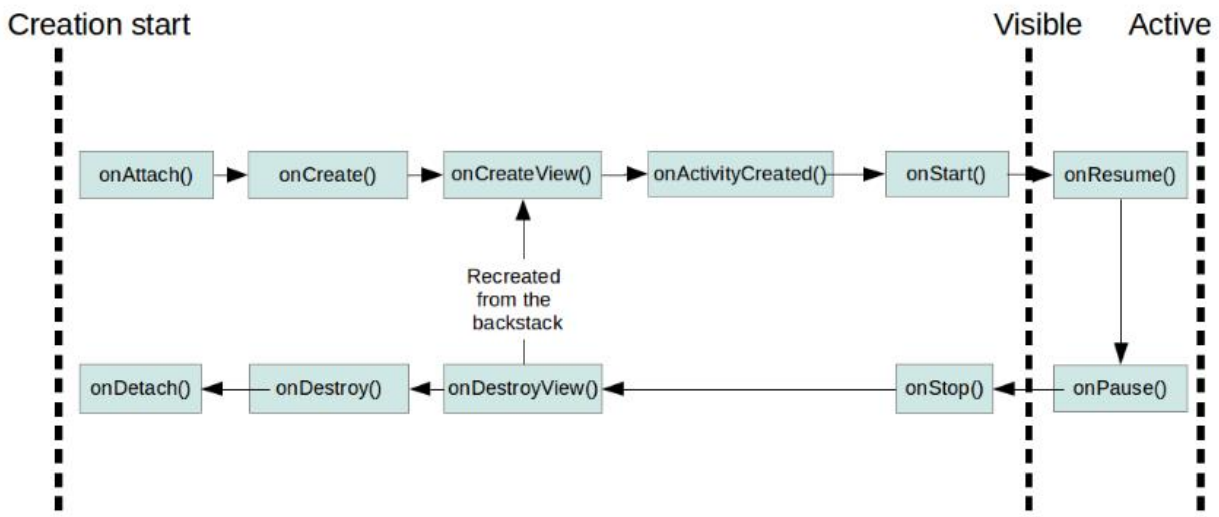
```
detailFragment = new DetailFragment();
// configure link
Bundle bundle = new Bundle();
bundle.putString("link", link);
detailFragment.setArguments(bundle);
```

fragment آبجکت bundle را در متد onActivityCreated خود دریافت می کند.

```
@Override
public void onActivityCreated(Bundle savedInstanceState) {
    super.onActivityCreated(savedInstanceState);
    Bundle bundle = getArguments();
    if (bundle != null) {
        setText(bundle.getString("link"));
    }
}
```

3-3-چرخه ی حیات fragment (life cycle)

در اندروید هر fragment چرخه ی حیات مختص به خود را دارد. با این وجود چرخه ی حیات آن همیشه به activity میزبان آن متصل است.



زمانی که activity متوقف می شود، همگام با آن fragment های زیرمجموعه ی این activity نیز متوقف می شوند. اگر activity از حافظه کلا پاک شود (destroy)، fragment های آن نیز همزمان از بین می روند.

متد	شرح کارکرد
onAttach()	کاربرد متد حاضر این است که fragment را به activity می شناساند. زمانی که onAttach() صدا خورده می شود، activity و fragment هیچ یک ایجاد نشده اند. شما می توانید داخل این متد عملیاتی که می خواهید پیش از ایجاد fragment انجام شود، تعریف نمایید. نمونه ی activity به نمونه ی fragment متصل است. به هنگام فراخوانی این متد activity و fragment، هنوز راه اندازی نشده اند. معمولا برنامه نویس اشاره گری به activity داخل این

شرح کارکرد	متد
متد قرار می دهید که از fragment برای مقداردهی بیشتر استفاده می کند.	
اندروید زمانی این متد را صدا می زند که در حال ساختن fragment باشد. برنامه نویس می تواند کامپوننت هایی از fragment را که می خواهد در زمان ادامه یافتن activity، بعد از توقف حفظ شود را مقداردهی اولیه نمایید. در واقع در این مرحله، fragment ایجاد شده است. این متد بعد از فراخوانده شدن ()onCreate مربوط به activity و قبل از متد ()onCreateView مربوط به fragment صدا خورده می شود.	onCreate()
نمونه ی fragment، زنجیره یا سلسله مراتب view های خود را ایجاد می کند. به عبارت دیگر fragment در متد ()onCreateView، رابط کاربری یا ظاهر خود را تعریف می کند. در این متد با فراخوانی متد ()inflate از آبجکت Inflator که به عنوان پارامتر به این متد فرستاده می شود، برنامه نویس می تواند یک layout را بارگذاری کرده و نمایش (inflate) دهد. در این متد نباید با activity تعامل داشته باشید. Activity هنوز کاملا ساخته و مقداردهی اولیه نشده است. لزومی ندارد این متد را برای fragment های فاقد headless/UI فراخوانی کنید. view های inflate شده بخشی از زنجیره ی view های activity میزبان محسوب می شوند. به عبارت دیگر، هنگامی که fragment می خواهد view های خود را بسازد این متد را صدا	onCreateView()

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

شرح کارکرد	متد
می زند. به منظور پیاده سازی UI یا رابط کاری، برنامه نویس می بایست یک آبجکت view که پوشه ی حاوی fragment را مشخص می کند، به عنوان خروجی این متد تعریف کنید. اگر fragment رابط کاربری نداشته باشد، می توانید خروجی متد را null تعریف کنید.	
متد جاری پس از فراخوانی متد onCreateView() و زمانی که activity میزبان ایجاد شده است، صدا خورده می شود. نمونه ی Activity و fragment همراه با زنجیره ی view های activity ساخته شده اند. در این برهه، می توان با فراخوانی متد findViewById() به view دلخواه دسترسی داشت. در این متد شما می توانید از آبجکت هایی که به یک آبجکت Context نیاز دارند، نمونه سازی نمایید. به عبارت دیگر، این متد بعد از onCreateView() و هنگامی که activity میزبان ایجاد شده، call می شود. activity و fragment به صورت یک آبجکت view سلسله مراتبی برای activity تولید می شوند.	onActivityCreated()
The onStart() method is called once the fragment gets visible. متد onStart() زمانی صدا زده می شود که fragment نمایان و قابل مشاهده شود.	onStart()
زمانی که Fragment در حال اجرا است.	onResume()

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

متد	شرح کارکرد
onPause()	Fragment قابل مشاهده است اما دیگر فعال نیست مانند زمانی که activity دیگری، با انیمیشن بر روی activity حامل fragment قرار می گیرد. Fragment با کاربر تعاملی ندارد. حال یا activity آن در حال متوقف شدن است یا یک عملیات مربوط به fragment در حال ویرایش آن در activity است.
onStop()	Fragment دیگر برای کاربر قابل مشاهده نیست.
onDestroyView()	View یا المان های رابط کاربری قابل مشاهد در fragment را پاک می کند. اگر fragment از backstack ساخته شده باشد، در آن صورت این متد فراخوانی می شود و پس از آن onCreateView صدا زده می شود. در واقع پس از فراخوانی این متد UI و ظاهر fragment به کلی از بین می رود.
onDestroy()	این متد cleanup و پاک سازی نهایی fragment ها را انجام می دهد. لازم به ذکر است که این متد ممکن است اصلا فراخوانی نشود.

3-4-تعریف fragment برای activity

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

3-4-1-تعریف fragment برای activity خود در فایل layout به صورت static

برای استفاده از fragment جدید خود، می توانید آن را به صورت static و به وسیله ی تگ fragment در فایل layout اضافه نمایید. خصیصه ی (attribute) android:name به کلاس مربوطه اشاره دارد.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:baselineAligned="false"
   android:orientation="horizontal" >
   <fragment
      android:id="@+id/listFragment"
      android:layout_width="0dp"
      android:layout_weight="1"
      android:layout_height="match_parent"
      class="com.example.android.rssreader.MyListFragment"
      tools:layout="@layout/fragment_rsslist_overview">
   </fragment>
   <fragment
      android:id="@+id/detailFragment"
      android:layout_width="0dp"
      android:layout_weight="2"
      android:layout_height="match_parent"
      class="com.example.android.rssreader.DetailFragment"
      tools:layout="@layout/fragment_rssitem_detail">
   </fragment>
</LinearLayout>
```

استفاده از این روش برای زمانی توصیه می شود که چندین فایل layout ثابت و static برای تعریف ظاهر برنامه دارید که هریک ویژه ی یک دستگاه با config خاص طراحی شده است.

3-4-2-مدیریت fragment در زمان اجرا و به صورت dynamic

کلاس FragmentManager به شما این امکان را می دهد تا fragment ها را در activity layout اضافه/حذف یا جایگزین نمایید. این کلاس به وسیله ی متد getFragmentManager() قابل دسترسی می باشد. اصلاحات و تغییراتی که می خواهید اعمال کنید باید در طی یک transaction (تراکنش) به وسیله ی کلاس FragmentTransaction انجام شود. به منظور ویرایش fragment

ها در یک activity می بایست یک مکان نگهدار (placeholder) FrameLayout در فایل layout تعریف نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="horizontal" >
   <FrameLayout
      android:id="@+id/listcontainer"
      android:layout_width="match_parent"
      android:layout_height="match_parent" />
   <FrameLayout
      android:id="@+id/detailscontainer"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:visibility="gone" />
</LinearLayout>
```

با استفاده از FragmentManager می توانید fragment موجود در container را با fragment دیگری جایگزین نمایید.

```
// get fragment manager
FragmentManager fm = getFragmentManager();
// add
FragmentTransaction ft = fm.beginTransaction();
ft.add(R.id.your_placehodler, new YourFragment());
// alternatively add it with a tag
// trx.add(R.id.your_placehodler, new YourFragment(), "detail");
ft.commit();
// replace
FragmentTransaction ft = fm.beginTransaction();
ft.replace(R.id.your_placehodler, new YourFragment());
ft.commit();
// remove
Fragment fragment = fm.findFragmentById(R.id.your_placehodler);
FragmentTransaction ft = fm.beginTransaction();
ft.remove(fragment);
ft.commit();
```

یک fragment جدید جایگزین fragment جاری در این container می شود.

می توانید یک تراکنش یا transaction را با استفاده از متد addToBackStack() به backstack اندروید اضافه نمایید. با این کار عملیات را به history stack از activity مورد نظر اضافه می کنید و به کاربر این امکان را می دهید تا تغییرات را از طریق دکمه ی بازگشت (back) به حالت قبلی بازگرداند.

3-4-3-بررسی اینکه آیا یک fragment در layout حاضر است یا خیر

به منظور بررسی اینکه آیا یک fragment در layout مورد نظر حضور دارد یا خیر، می توانید از کلاس FragmentManager کمک بگیرید. کافی است متد ()isLayout را فراخوانی کرده و بررسی کنید آیا fragment مورد نظر از طریق layout به activity اضافه شده است یا خیر.

```
DetailFragment fragment = (DetailFragment) getFragmentManager().
    findFragmentById(R.id.detail_frag);
if (fragment==null || ! fragment.isInLayout()) {
    // start new Activity
    }
else {
    fragment.update(...);
}
```

3-4-4-بررسی تعداد fragment ها

منطق برنامه در activity به طور قطع به سناریو و شرایط جاری بستگی دارد (اینکه آیا صفحه تک قطعه است یا از چند fragment و قطعه تشکیل شده). می توان با نوشتن یک قطعه کد شرطی به تعداد fragment در اپلیکیشن پی برد. روش های متعددی برای اجرای این عملیات وجود دارد. روش اول این است که یک فایل تنظیمات/configuration در پوشه ی محتوا و منابع پروژه (values) ایجاد نمایید. جفت کلید/مقدار (key/value) به صورت پیش فرض بر روی false تنظیم می شوند. سپس یک فایل تنظیمات دیگر این مقدار را جهت سازگاری با اندازه ی دلخواه نمایشگر بر روی true قرار می دهد.

نمونه ی زیر یک فایل تنظیمات پیش فرض به نام config.xml را نمایش می دهد.

```
<resources>
  <item type="bool" name="twoPaneMode">false</item>
</resources>
```

حال همین فایل را در پوشه ی res/values-land با مقداری متفاوت (این بار true) ایجاد نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
<resources>
   <item type="bool" name="twoPaneMode">true</item>
</resources>
```

می توانید با استفاده نوشتن دستور زیر، به اطلاعات مربوط به وضعیت جاری (state) دسترسی داشته باشید.

```
getResources().getBoolean(R.bool.twoPaneMode);
```

5-4-3-افزودن آبجکت fragment transition به backstack

می توانید آبجکت FragmentTransition را به backstack اضافه نموده و به کاربر این امکان را بدهید تا با استفاده از دکمه ی بازگشت (back) به fragment قبلی بازگردد.

برای این منظور شما می توانید متد addToBackStack() را بر روی آبجکت FragmentTransition فراخوانی کنید.

6-4-3-استفاده از افکت های تعریف شده توسط Property Animation API برای حرکت بین fragment ها

در طول اجرای یک تراکنش fragment، می توانید با استفاده از متد setCustomAnimations() از مجموعه توابع Property Animation، به راحتی animation تعریف نمایید (برای حرکت و انتقال بین fragment ها انیمیشن اعمال نمایید).

البته می توانید با فراخوانی متد setTranstion()، بسیاری از انیمیشن های استاندارد اندروید را برای انتقال بین fragment ها اعمال نمایید. برای پیاده سازی انیمیشن می بایست constant هایی که با FragmentTransaction.TRANSIT_FRAGMENT_* شروع می شوند را تعریف نمایید.

هر دو متد نام برده به شما امکان می دهند تا یک انیمیشن آغاز (entry animation) و یک انیمیشن پایان (exit) تعریف نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

7-4-3-ماندگارسازی و ذخیره ی دائمی داده در fragment ها

اغلب برنامه نویس لازم دارد که داده هایی از اپلیکیشن را در حافظه به طور دائمی ذخیره کند. برای این منظور کافی است از یک فایل ساده یا دیتابیس SQLite استفاده کند.

8-4-3-نگهداری و بازگردانی اطلاعات پس از تغییر در تنظیمات و config

اگر می خواهید تغییرات در تنظیمات و پیکربندی را ماندگارسازی نمایید، در آن صورت می توانید از آبجکت اپلیکیشن نیز استفاده کنید.

علاوه بر روش ذکر شده، شما می توانید متد setRetainState(true) را بر روی fragment فراخوانی نمایید. با این کار اطلاعات مربوط به وضعیت نمونه ی fragment بین تغییرات در تنظیمات fragment ماندگار می شود. لازم به ذکر است که این رویکرد تنها زمانی کارگر واقع می شود که fragment ها به backstack اضافه نشده باشند. در این صورت، کافی داده ها را به عنوان یک عضو (در قالب یک فیلد) ذخیره نمایید.

توجه: Google استفاده از این متد را برای fragment هایی که UI دارند به هیچ وجه توصیه نمی کند.

می توانید با فراخوانی متد onSaveInstanceState() داده ها را در آبجکت Bundle قرار دهید. سپس می توانید این داده ها را با استفاده از متد onActivityCreated() بازیابی نمایید.

3-5-Fragment ها و پردازش در پس زمینه (background processing)

1-5-3-Fragment های فاقد UI/headless fragments

می توانید fragment هایی تعریف نمایید که اصلا UI و ظاهر ندارند. به این fragment ها در اصطلاح headless گفته می شود. جهت پیاده سازی چنین fragment ای کافی است مقدار null را از متد onCreateView() در fragment برگردانید.

توجه: توصیه می شود برای پردازش پس زمینه ای از service ها استفاده نمایید. اگر می خواهید این کار را از طریق fragment های خود انجام دهید، دومین راه حل پیشنهادی ما (بعد از

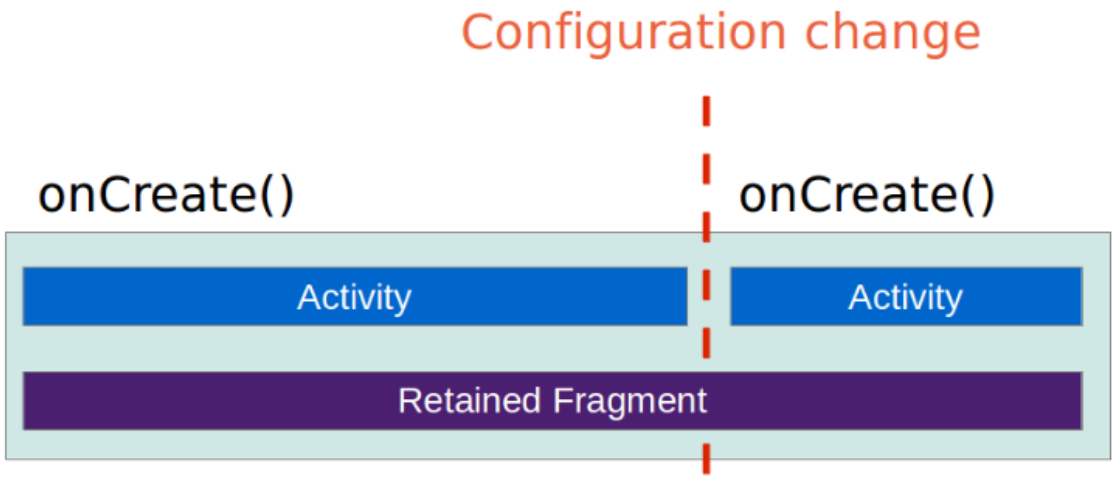
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

سرویس ها) headless fragment ها همراه با فراخوانی متد setRetainInstance() هست. بدین وسیله دیگر شما مجبور نیستید تغییرات در تنظیمات را حین پردازش ناهمزمان (asynchronous processing)، خود مدیریت نمایید.

2-5-3- Retained headless fragment ها جهت مدیریت تغییر در تنظیمات (config changes)

Headless fragment ها معمولا به منظور کپسوله سازی (encapsulate) و نگهداری اطلاعات مربوط به وضعیت fragment ها، مابین تغییرات در تنظیمات و نحوه ی پیکربندی اپلیکیشن، استفاده می شوند. مورد کاربرد دیگری که می توان به آن اشاره کرد: برای task هایی که پردازش هایی را در پس زمینه انجام می دهند (background processing task) نیز مورد استفاده قرار می گیرند. برای کپسوله سازی داده ها در fragment ها، می توانید headless fragment را retained تعریف نمایید. retained fragment حین تغییر در تنظیمات و نحوه ی پیکربندی (configuration changes) از بین نمی روند (در واقع این fragment ها قادرند اشاره گرهایی به آبجکت های پایدار و stateful که شما می خواهید اطلاعات آن ها را حفظ نمایید، نزد خود نگه دارند).



برای اینکه fragment اطلاعات مربوط به وضعیت را نگه دارند و به اصطلاح retained باشند، کافی است متد setRetainInstance() را صدا بزنید.

به منظور افزودن چنین fragment ای به activity مورد نظر، می توانید متد add() از کلاس FragmentManager را فراخوانی کنید. برای اینکه در آینده به این Fragment دسترسی داشته باشید (به آن اشاره کنید)، لازم است آن را با یک تگ در فایل XML اضافه نمایید. این کار به شما اجازه می دهد، با استفاده از متد findFragmentByTag() از FragmentManager به آن دسترسی داشته باشید.

توجه داشته باشید که استفاده از متد onRetainNonConfigurationInstance() در activity دیگر منسوخ شده است و جای خود را به retained headless fragment ها داده اند.

تمرین: پیاده سازی fragment ها در اپلیکیشن ها

مبحث زیر نحوه ی استفاده از fragment ها در یک اپلیکیشن متعارف اندروید، بدون استفاده از support library (کتابخانه ی پشتیبانی از API های جدید در ویرایش های قدیمی تر اندروید) را تشریح می کند. اپلیکیشن بسته به حالت نمایش (نمای افقی/عمودی)، از layout با fragment های متفاوت بهره می گیرد.

در نمای عمودی، RssfeedActivity در لحظه تنها یک fragment را به نمایش می گذارد. از این fragment، کاربر می تواند به activity دیگری که میزبان fragment مورد نظر است راه پیدا کند. در نمای افقی، activity مزبور هر دو fragment را درکنار هم برای کاربر به نمایش می گذارد.

ایجاد پروژه

یک پروژه ی جدید اندروید ایجاد نموده و آن را به صورت زیر مقداردهی نمایید. توجه داشته باشید که نباید از لایه ی سازگاری (Compatibility layer) استفاده کنید.

جدول 1

Property	مقدار

Application Name	RSS Reader
Company Domain (اسم دامنه ی شرکت)	android.example.com
Package name (اسم)	com.example.android.rssreader
Template (الگو یا قالب آماده)	Empty Activity
Activity	RssfeedActivity
Layout	activity_rssfeed

3-5-3-تعریف فایل های layout برای fragment ها

یک فایل layout جدید به نام fragment_rssitem_detail.xml در پوشه ی /res/layout ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <TextView
      android:id="@+id/detailsText"
      android:layout_width="wrap_content"
      android:layout_height="match_parent"
     android:layout_gravity="center_horizontal|center_vertical"
      android:layout_marginTop="20dip"
      android:text="Default Text"
    android:textAppearance="?android:attr/textAppearanceLarge"
      android:textSize="30dip" />
</LinearLayout>
```

توجه: id باید صحیح باشد چرا که fragment با استفاده از این شناسه به المان TextView دسترسی دارد.

یک فایل layout جدید به نام fragment_rsslist_overview.xml ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <Button
      android:id="@+id/updateButton"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Press to update"
       />
</LinearLayout>
```

توجه: در اینجا هم id باید دقیق باشد.

4-5-3-ایجاد کلاس های fragment

کلاس های زیر را ایجاد نمایید. این کلاس ها به عنوان fragment مورد استفاده قرار می گیرند.

ابتدا کلاس DetailFragment را تعریف نمایید.

```
package com.example.android.rssreader;
import android.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
public class DetailFragment extends Fragment {
   public static final String EXTRA_URL ="url";
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
                    Bundle savedInstanceState) {
      View view = inflater.inflate(R.layout.fragment_rssitem_detail,
          container, false);
      return view;
   }
   @Override
   public void onActivityCreated(Bundle savedInstanceState) {
      super.onActivityCreated(savedInstanceState);
      Bundle bundle = getArguments();
      if (bundle != null) {
        String link = bundle.getString("url");
        setText(link);
      }
   }
      public void setText(String url) {
          TextView view = (TextView) getView().findViewById(R.id.detailsText);
          view.setText(url);
      }
}
```

کلاس MyListFragment را ایجاد نمایید. این کلاس برخلاف اسمش، هیچ لیستی را به نمایش نمی گذارد، بلکه صرفا یک دکمه ارائه می دهد که با کلیک کاربر بر روی آن، مقدار زمان جاری به fragment ای به نام DetailsFragment ارسال می گردد.

```
package com.example.android.rssreader;
import android.app.Fragment;
import android.content.Context;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
```

```
import android.view.ViewGroup;
import android.widget.Button;
public class MyListFragment extends Fragment {
   private OnItemSelectedListener listener;
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
                     Bundle savedInstanceState) {
      View view = inflater.inflate(R.layout.fragment_rsslist_overview,
            container, false);
      Button button = (Button) view.findViewById(R.id.updateButton);
      button.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            updateDetail("fake");
         }
      });
      return view;
   }
   public interface OnItemSelectedListener {
      void onRssItemSelected(String link);
   }
   @Override
   public void onAttach(Context context) {
      super.onAttach(context);
      if (context instanceof OnItemSelectedListener) {
         listener = (OnItemSelectedListener) context;
      } else {
         throw new ClassCastException(context.toString()
               + " must implement MyListFragment.OnItemSelectedListener");
      }
   }
   // triggers update of the details fragment
   public void updateDetail(String uri) {
      // create fake data
      String newTime = String.valueOf(System.currentTimeMillis());
      // send data to activity
      listener.onRssItemSelected(newTime);
   }
}
```

5-5-3-ویرایش فایل layout اصلی

فایل activity_rssfeed.xml فعلی را ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:baselineAligned="false"
   android:orientation="horizontal" >
   <fragment
```

```
        android:id="@+id/listFragment"
        android:layout_width="0dp"
        android:layout_weight="1"
        android:layout_height="match_parent"
        class="com.example.android.rssreader.MyListFragment"
        tools:layout="@layout/fragment_rsslist_overview">
    </fragment>
    <fragment
        android:id="@+id/detailFragment"
        android:layout_width="0dp"
        android:layout_weight="2"
        android:layout_height="match_parent"
        class="com.example.android.rssreader.DetailFragment"
        tools:layout="@layout/fragment_rssitem_detail">
    </fragment>
</LinearLayout>
```

6-5-3-ویرایش کلاس RssfeedActivity

کلاس RssfeedActivity را ویرایش کنید به طوری که نقش یک تابع callback (بازفراخوان) را برای ListFragment ایفا کند و متعاقبا DetailsFragment را بروز رسانی نماید.

```
package com.example.android.rssreader;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
public class RssfeedActivity extends Activity implements MyListFragment.OnItemSelectedListener{
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_rssfeed);
    }
    @Override
    public void onRssItemSelected(String link) {
        DetailFragment fragment = (DetailFragment) getFragmentManager()
                .findFragmentById(R.id.detailFragment);
        fragment.setText(link);
    }
}
```

تست اپلیکیشن و کسب اطمینان از اجرای موفقیت آمیز آن

اپلیکیشن خود را اجرا نمایید. هر دو fragment باید در نمای افقی و عمودی نمایش داده شوند. می توانید با استفاده از کنترل های شبیه ساز (emulator)، جهت نمایش را تغییر دهید. به دنبال فشرده شدن دکمه در ListFragment، اطلاعات DetailFragment بروز آوری می شوند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

تمرین: نمایش fragment ها بر اساس تنظیمات و نحوه ی پیکربندی

چنانچه اپلیکیشن در نمای افقی راه اندازی شود، قاعدتا باید دو fragment (دو قطعه یا pane) را در UI برای کاربر به نمایش بگذارد. در تمرین حاضر اپلیکیشن را طوری تنظیم کنید که از این قابلیت یا رفتار پشتیبانی کند.

3-5-7-تعریف activity layout ویژه ی نمای عمودی

فایل activity_rssfeed.xml را با تنظیمات زیر تعریف نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".RssfeedActivity"
   android:orientation="vertical">
   <FrameLayout
      android:id="@+id/fragment_container"
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:orientation="horizontal"
      android:layout_weight="1"
      android:paddingBottom="@dimen/activity_vertical_margin"
      android:paddingLeft="@dimen/activity_horizontal_margin"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="@dimen/activity_vertical_margin"
      >
   </FrameLayout>
</LinearLayout>
```

3-5-8-تعریف یک flag یا گزینه ی بولی مستقل از resource selector (گزینشگر منابع)

یک فایل به نام config.xml در پوشه ی res/values با تنظیمات زیر ایجاد نمایید.

```
<resources>
   <item type="bool" name="twoPaneMode">false</item>
</resources>
```

همین فایل را در پوشه ی res/values-land با مقداری متفاوت تعریف نمایید.

```
<resources>
   <item type="bool" name="twoPaneMode">true</item>
</resources>
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

9-5-3-تنظیم و ویرایش کلاس RssfeedActivity

پیاده سازی کلاس RssfeedActivity را طوری ویرایش نمایید که در صورت قرار گرفتن اپلیکیشن در وضعیت تک قطعه/قابه (single-pane)، fragment جاری را جایگزین کند.

```
package com.example.android.rssreader;
import android.app.Activity;
import android.app.Fragment;
import android.app.FragmentTransaction;
import android.os.Bundle;
import android.widget.FrameLayout;
public class RssfeedActivity extends Activity implements
    MyListFragment.OnItemSelectedListener {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_rssfeed);
    if (getResources().getBoolean(R.bool.twoPaneMode)) {
      // all good, we use the fragments defined in the layout
      return;
    }
    // if savedInstanceState is null we do some cleanup
    if (savedInstanceState != null) {                    (1
      // cleanup any existing fragments in case we are in detailed mode
      getFragmentManager().executePendingTransactions();
      Fragment fragmentById = getFragmentManager().
          findFragmentById(R.id.fragment_container);
      if (fragmentById!=null) {
        getFragmentManager().beginTransaction()
            .remove(fragmentById).commit();
      }
    }
    MyListFragment listFragment = new MyListFragment();
    FrameLayout viewById = (FrameLayout) findViewById(R.id.fragment_container);
    getFragmentManager().beginTransaction()
        .replace(R.id.fragment_container, listFragment).commit();
  }
  @Override
  public void onRssItemSelected(String link) {
    if (getResources().getBoolean(R.bool.twoPaneMode)) {
      DetailFragment fragment = (DetailFragment) getFragmentManager()
          .findFragmentById(R.id.detailFragment);
      fragment.setText(link);
    } else {
      // replace the fragment
      // Create fragment and give it an argument for the selected article
      DetailFragment newFragment = new DetailFragment();
      Bundle args = new Bundle();
```

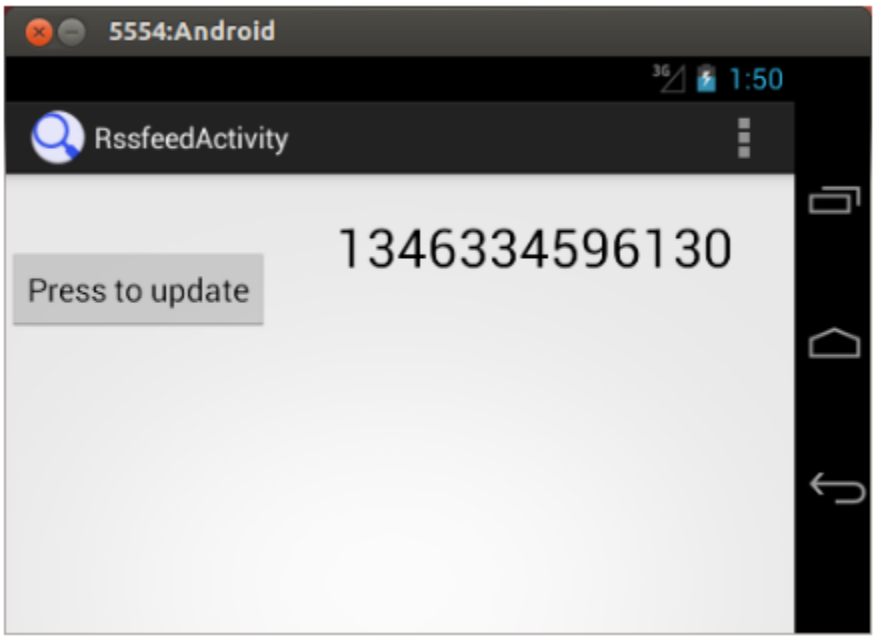
```
            args.putString(DetailFragment.EXTRA_URL, link);
            newFragment.setArguments(args);
            FragmentTransaction transaction = getFragmentManager().beginTransaction();
            // Replace whatever is in the fragment_container view with this fragment,
            // and add the transaction to the back stack so the user can navigate back
            transaction.replace(R.id.fragment_container, newFragment);
            transaction.addToBackStack(null);
            // Commit the transaction
            transaction.commit();
        }
    }
}
```

توجه: این فرایند پاک سازی ضروری است، چراکه اپلیکیشن ما به صورت dynamic و در زمان اجرا بین دو حالت تک قطعه یا نمایش یک fragment در لحظه و نمایش دو قطعه یا fragment در آن واحد، تغییر وضعیت می دهد. البته سناریوی حاضر غیرمعمول بوده و معمولا انتخاب یک یا دو fragment برای اپلیکیشن بستگی به حداقل عرض نمایشگر دارد و این پیکربندی در زمان اجرای برنامه تغییر نمی یابد. FragmentManager سعی دارد با cache یا ذخیره ی موقتی fragment کارایی را افزایش دهد، از اینرو لازم است detailed fragment موجود را حذف نمایید.

تست اپلیکیشن و کسب اطمینان از درستی پیاده سازی آن

اپلیکیشن را اجرا نمایید. اگر اپلیکیشن را در نمای عمودی (portrait mode) اجرا کنید، طبیعتا باید تنها یک fragment را در نمایشگر مشاهده نمایید. در نمای افقی (landscape mode) شما هر دو fragment را در همزمان در UI مشاهده خواهید نمود.

حال وضعیت نمایش شبیه ساز را تغییر دهید. بر روی دکمه ی مورد نظر در نمای عمودی (portrait mode) و سپس در نمای افقی (landscape mode) کلیک نمایید. اطمینان حاصل نمایید که detail activity، زمان جاری را نمایش می دهد.



تمرین: ذخیره ی اطلاعات مربوط به وضعیت در یک headless retained fragment

در تمرین حاضر قصد دارید که آخرین انتخاب کاربر را ذخیره کرده و بیاد آورید. برای این منظور می بایست از یک headless fragment استفاده نمایید.

ایجاد headless fragment

```
package com.example.android.rssreader;
import android.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
public class SelectionStateFragment extends Fragment {
   public String lastSelection = "";
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
                    Bundle savedInstanceState) {
      return null;
```

```
    }
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setRetainInstance(true);
    }
}
```

10-5-3-ذخیره ی آخرین مقدار انتخابی در یک headless fragment

```
package com.example.android.rssreader;
import android.app.Activity;
import android.app.Fragment;
import android.app.FragmentTransaction;
import android.os.Bundle;
import android.widget.FrameLayout;
public class RssfeedActivity extends Activity implements
        MyListFragment.OnItemSelectedListener {
    SelectionStateFragment stateFragment;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_rssfeed);
        stateFragment =
                (SelectionStateFragment) getFragmentManager()
                        .findFragmentByTag("headless");
        if(stateFragment == null) {
            stateFragment = new SelectionStateFragment();
            getFragmentManager().beginTransaction()
                    .add(stateFragment, "headless").commit();
        }
        if (findViewById(R.id.fragment_container) == null) {
            // restore state
            if (stateFragment.lastSelection.length()>0) {
                onRssItemSelected(stateFragment.lastSelection);
            }
            // all good, we use the fragments defined in the layout
            return;
        }
        // if savedInstanceState is null we do some cleanup
        if (savedInstanceState != null) {
            // cleanup any existing fragments in case we are in detailed mode
            getFragmentManager().executePendingTransactions();
            Fragment fragmentById = getFragmentManager().
                    findFragmentById(R.id.fragment_container);
            if (fragmentById!=null) {
                getFragmentManager().beginTransaction()
                        .remove(fragmentById).commit();
            }
        }
        MyListFragment listFragment = new MyListFragment();
        FrameLayout viewById = (FrameLayout) findViewById(R.id.fragment_container);
```

```
        getFragmentManager().beginTransaction()
                .replace(R.id.fragment_container, listFragment).commit();
    }
    @Override
    public void onRssItemSelected(String link) {
        stateFragment.lastSelection = link;
        if (getResources().getBoolean(R.bool.twoPaneMode)) {
            DetailFragment fragment = (DetailFragment) getFragmentManager()
                    .findFragmentById(R.id.detailFragment);
            fragment.setText(link);
        } else {
            // replace the fragment
            // Create fragment and give it an argument for the selected article
            DetailFragment newFragment = new DetailFragment();
            Bundle args = new Bundle();
            args.putString(DetailFragment.EXTRA_URL, link);
            newFragment.setArguments(args);
            FragmentTransaction transaction = getFragmentManager().beginTransaction();
            // Replace whatever is in the fragment_container view with this fragment,
            // and add the transaction to the back stack so the user can navigate back
            transaction.replace(R.id.fragment_container, newFragment);
            transaction.addToBackStack(null);
            // Commit the transaction
            transaction.commit();
        }
    }
}
```

3-5-11-تعریف یک flag یا گزینه ی بولی مستقل از گزینشگر resource

یک فایل در پوشه ی res/values به نام config.xml با تنظیمات زیر تعریف نمایید.

```
<resources>
    <item type="bool" name="dual_pane">false</item>
</resources>
```

همین فایل را در پوشه ی res/values-land با مقداری متفاوت ایجاد نمایید.

```
<resources>
    <item type="bool" name="dual_pane">true</item>
</resources>
```

3-5-12-تنظیم و ویرایش کلاس RssfeedActivity

بدنه ی کلاس RssfeedActivity را طوری ویرایش نمایید که در صورت قرار گرفتن اپلیکیشن در وضعیت تک قطعه (single-pane)، fragment جاری را جایگزین کند.

```
package com.example.android.rssreader;
```

```
import android.app.Activity;
import android.app.Fragment;
import android.app.FragmentTransaction;
import android.os.Bundle;
import android.widget.FrameLayout;
public class RssfeedActivity extends Activity implements
      MyListFragment.OnItemSelectedListener {
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_rssfeed);
      if (getResources().getBoolean(R.bool.twoPaneMode)) {
         // all good, we use the fragments defined in the layout
         return;
      }
      // if savedInstanceState is null we do some cleanup
      if (savedInstanceState != null) {            (1
         // cleanup any existing fragments in case we are in detailed mode
         getFragmentManager().executePendingTransactions();
         Fragment fragmentById = getFragmentManager().
               findFragmentById(R.id.fragment_container);
         if (fragmentById!=null) {
            getFragmentManager().beginTransaction()
                  .remove(fragmentById).commit();
         }
      }
      MyListFragment listFragment = new MyListFragment();
      FrameLayout viewById = (FrameLayout) findViewById(R.id.fragment_container);
      getFragmentManager().beginTransaction()
            .replace(R.id.fragment_container, listFragment).commit();
   }
   @Override
   public void onRssItemSelected(String link) {
      if (getResources().getBoolean(R.bool.twoPaneMode)) {
         DetailFragment fragment = (DetailFragment) getFragmentManager()
               .findFragmentById(R.id.detailFragment);
         fragment.setText(link);
      } else {
         // replace the fragment
         // Create fragment and give it an argument for the selected article
         DetailFragment newFragment = new DetailFragment();
         Bundle args = new Bundle();
         args.putString(DetailFragment.EXTRA_URL, link);
         newFragment.setArguments(args);
         FragmentTransaction transaction = getFragmentManager().beginTransaction();
         // Replace whatever is in the fragment_container view with this fragment,
         // and add the transaction to the back stack so the user can navigate back
         transaction.replace(R.id.fragment_container, newFragment);
         transaction.addToBackStack(null);
         // Commit the transaction
         transaction.commit();
      }
```

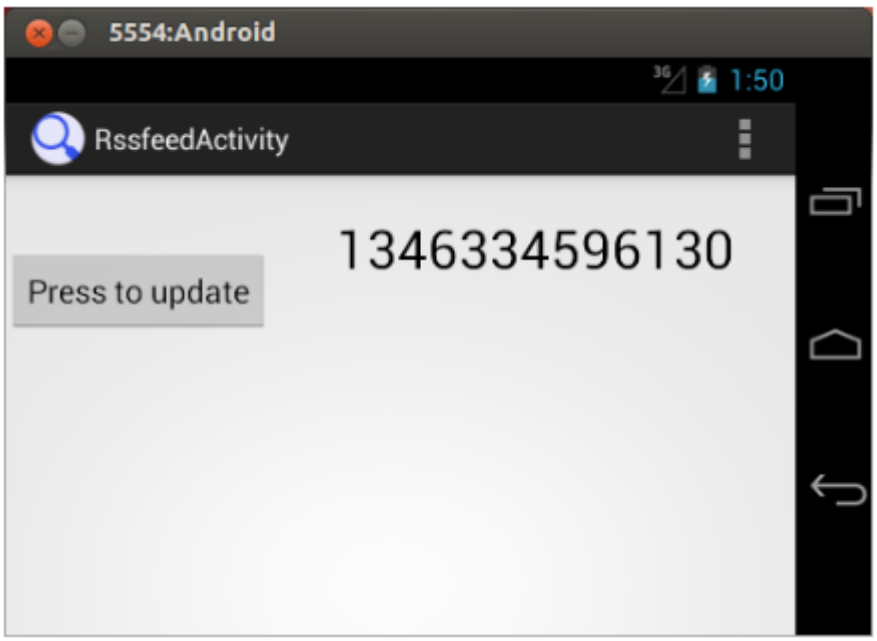
```
    }
}
```

این فرایند پاک سازی ضروری است، چراکه اپلیکیشن ما به صورت dynamic و در زمان اجرا بین دو حالت تک قطعه یا نمایش یک fragment در لحظه و نمایش دو قطعه یا fragment در آن واحد، تغییر وضعیت می دهد. البته سناریوی حاضر غیرمعمول بوده و معمولا انتخاب یک یا دو fragment برای اپلیکیشن بستگی به حداقل عرض نمایشگر دارد و این پیکربندی در زمان اجرای برنامه تغییر نمی یابد. FragmentManager سعی دارد با cache یا ذخیره ی موقتی fragment کارایی را افزایش دهد، از اینرو لازم است detailed fragment موجود را حذف نمایید.

تست اپلیکیشن و اعتبارسنجی رفتارهای آن

اپلیکیشن خود را اجرا نمایید. اکنون اگر اپلیکیشن خود را در نمای عمودی (portrait mode) اجرا نمایید، طبیعتا باید در لحظه تنها یک fragment در نمایشگر قابل مشاهده باشد. در نمای افقی می بایست هر دو fragment همزمان برای کاربر نمایش داده شود.

جهت یا وضعیت نمایش شبیه ساز را تغییر دهید. دکمه را در دو حالت نمای عمودی و افقی فشار داده و مطمئن شوید که detail activity زمان حاضر را به درستی نمایش می دهد.



بخش دوم :

استفاده از نوارابزار در اپلیکیشن های اندروید/Action bar در اندروید

این مبحث تمرکز خود را بر روی آموزش نحوه ی پیاده سازی و استفاده از toolbar در اپلیکیشن های اندرویدی قرار می دهد. آموزش حاضر همچنین چگونگی استفاده از widget/کامپوننت رابط کاربری toolbar را برای شما شرح می دهد. این آموزش مبتنی بر ویرایش 6.0 سیستم عامل اندروید می باشد (= API یا کتابخانه های اندروید ورژن 23).

3-6-شرح مفهوم Toolbar

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-6-3-Toolbar در اندروید چیست و چه کاربردی دارد؟ (action bar)

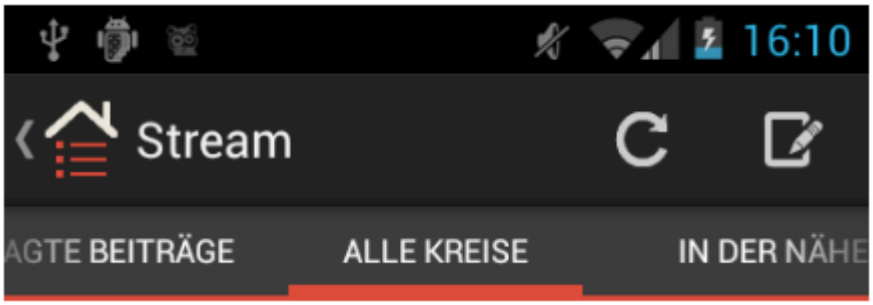
Toolbar در ویرایش 5.0 سیستم عامل اندروید جای actionbar در اندروید 4 را گرفت تا محدودیت های جاری را برطرف ساخته و راه حلی نوین برای طراحی اپلیکیشن را در اختیار برنامه نویس قرار دهد. این widget یا کامپوننت UI از طریق view group به نام Toolbar در دسترس توسعه دهنده قرار می گیرد.

می توان toolbar را به راحتی در فایل layout جایگذاری نمود. این کامپوننت UI قادر است عنوان activity، آیکون، عملیاتی که قابل فعال سازی هستند (action ها) و حتی view ها و سایر آیتم های تعاملی را دربرگیرد. مورد دیگر استفاده ی toolbar در پیمایش و راهبری بین بخش های مختلف اپلیکیشن است.

قبل از ویرایش 5.0 اندروید، مکان قرار گیری نوار ابزار (همان actionbar در ویرایش های قبلی) به صورت ثابت در بالای صفحه یا activity بوده (در آنجا hard code شده بود).

اکنون این امکان وجود دارد که نوار ابزار را در پوسته ی مورد استفاده (theme) غیرفعال نمود. کامپوننت UI نام برده در تمامی پوسته های درون ساخته ی اندروید به صورت پیش فرض فعال می باشد.

تصویر زیر نوارابزار اپلیکیشن اندرویدی Google+ را به همراه آیتم های تعاملی و نوار راهبری (navigation) نشان می دهد.در بالای آن سمت چپ، علامتی قابل مشاهده است که کاربر می تواند با کلیک بر روی آن نوار راهبری را باز کرده و در اپلیکیشن پیمایش کند.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Actionbar-3-6-2 بر روی دستگاه هایی که ورژن کتابخانه های اندروید مورد استفاده در آن ها پایین تر از 21 است (API 21)

Toolbar در ویرایش 5.0 اندروید (ورژن کتابخانه های اندروید 21) برای اولین بار معرفی شد و در واقع جایگزینی برای actionbar در ورژن های قبلی این سیستم عامل گردید. اگر می خواهید از این کامپوننت در ویرایش های قدیمی تر اندروید استفاده نمایید، در آن صورت کافی است از downport ای که توسط appcompat-v7 در اختیار توسعه دهنده قرار می گیرد، استفاده نمایید.

جهت استفاده از toolbar در چنین دستگاه هایی، دستوری مانند نمونه ی زیر را به فایل Gradle build خود اضافه نمایید:

```
compile `com.android.support:appcompat-v7:22.2.0`
```

برای آموزش نحوه ی نصب v7 library به آدرس http://developer.android.com/tools/support-library/setup.html مراجعه نمایید.

Options menu-3-6-3

اپلیکیشن هایی که target SDK آن ها بر روی پایین تر از API 11 تنظیم شده، از options menu استفاده می کنند. البته اگر چنین دکمه ای در دستگاه موجود باشد. این کامپوننت UI با کلیک کاربر بر روی دکمه ی Options به نمایش در می آید. Toolbar نسبت به options menu کاربر پسندتر است زیرا action bar همیشه قابل مشاهده و در دسترس است. در حالی که options menu فقط در صورت درخواست نمایان می شود. در واقع options menu یک کاستی برجسته دارد: ممکن است کاربر متوجه وجود گزینه ها و تنظیمات ارائه شده توسط اپلیکیشن نشود.

3-7-استفاده از Toolbar

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-7-3-تعریف آیتم های نوار ابزار/actions در toolbar

المان هایی که در نوارابزار مشاهده می کنید در اندروید actions خوانده می شوند. اگرچه می توان آیتم های نوار ابزار را با کدنویسی تعریف کرد، با این حال توسعه دهندگان معمولا آن ها را در یک فایل محتوا XML تحت پوشه ی /res تعریف می کنند.

تعریف منوها در یک فایل مجزا تحت پوشه ی res/menu ذخیره می شوند. ابزار اندروید خود به صورت خودکار یک اشاره گر به آیتم های منو جهت دسترسی به منابع آن در فایل R ایجاد می کند.

یک activity معمولا entry ها را از طریق متد onCreateOptionsMenu() به نوارابزار تزریق می کند.

خصیصه showAsAction (attribute) به شما این اجازه را می دهد تا نحوه ی نمایش و ظاهر آیتم مورد نظر در نوارابزار را تعیین نمایید. به عنوان مثال، خصیصه ی ifRoom مشخص می کند آیا فضای کافی برای نمایش المان وجود دارد یا خیر.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
  <item
    android:id="@+id/action_refresh"
    android:orderInCategory="100"
    android:showAsAction="always"
    android:icon="@drawable/ic_action_refresh"
    android:title="Refresh"/>
  <item
    android:id="@+id/action_settings"
    android:title="Settings">
  </item>
</menu>
```

کلاس MenuInflator این امکان را فراهم می آورد تا action های تعریف شده در فایل XML را inflate نموده (آن ها را جهت قرار گرفتن در activity به آبجکت view تبدیل کرده) سپس آن ها را به action bar اضافه نمایید. می توانید با فراخوانی getMenuInflator() در سطح activity به کلاس MenuInflator دسترسی داشته باشید. نمونه کد زیر نحوه ی پیاده سازی و ایجاد action ها را نمایش می دهد.

```
@Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.mainmenu, menu);
        return true;
    }
```

نکته: اگرچه می توان action ها را در کد برنامه نیز تعریف کرد، با این حال توصیه می شود این کار را در فایل های XML انجام دهید چرا که میزان کدهای تکراری (boilerplate code) را کاهش می دهد.

2-7-3-عملیاتی که در پی انتخاب آیتم های نوارابزار رخ می دهند (واکنش نشان دادن به انتخاب action ها)

زمانی که action ای کلیک می شود، متد onOptionsItemSelected() در activity مربوطه فراخوانی می گردد. این متد action انتخابی را به عنوان پارامتر ورودی دریافت می کند. کاربرد این متد در نمونه ی زیر به نمایش گذاشته شده است.

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
  switch (item.getItemId()) {
  // action with ID action_refresh was selected
  case R.id.action_refresh:
        Toast.makeText(this, "Refresh selected", Toast.LENGTH_SHORT)
          .show();
        break;
  // action with ID action_settings was selected
  case R.id.action_settings:
        Toast.makeText(this, "Settings selected", Toast.LENGTH_SHORT)
          .show();
        break;
  default:
        break;
  }
  return true;
}
```

3-7-3-جستجو برای یک action یا آیتم در نوارابزار/action bar

به منظور جستجو برای یک action یا آیتم در منو، متد findItem() از کلاس Menu را فراخوانی نمایید. به وسیله ی این متد می توانید با استفاده از id یک آیتم به آن دسترسی داشته باشید.

3-7-4-ویرایش منو

متد onCreateOptionsMenu() تنها یکبار صدا خورده می شود. در صورت نیاز به ویرایش منو در آینده، لازم است متد invalidateOptionsMenu() را فراخوانی نمایید. پس از آن تابع onCreateOptionsMenu() بار دیگر صدا زده می شود.

3-7-5-Contextual action mode و toolbar (وابسته کردن toolbar به قرائن با استفاده از contextual action mode)

contextual action mode یک نوارابزار (وابسته به قرائن و بستر جاری) است که برای انجام subtask ها و وظایف کوچک برای مدت زمان معینی، به صورت موقتی بر روی نوارابزار اصلی اپلیکیشن ظاهر می شود.

contextual action mode معمولا با انتخاب یک آیتم یا کلیک طولانی مدت بر روی آن ظاهر می شود.

به منظور پیاده سازی این نوع toolbar، کافی است متد startActionMode() را بر روی یک viewer در activity خود فراخوانی نمایید. متد نام برده آبجکت ActionMode.Callback را دریافت می کند. این آبجکت وظیفه ی مدیریت چرخه ی حیات contextual action bar را بر عهده است. همچنین می توانید با استفاده از متد registerForContexMenu(view)، یک منوی وابسته به قرائن/context menu به view مورد نظر اختصاص دهید. Context menu نیز با کلیک طولانی مدت بر روی view مورد نظر فعال می شود.

هر بار که یک context menu فعال می شود، متد onCreateContextMenu() نیز فراخوانی می شود. می دانید چرا؟ context menu پس از هر بار استفاده کاملا دور انداخته می شود.

توصیه می شود تا حد امکان بجای context menu از contextual action mode استفاده نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

3-7-6-اضافه نمودن آیتم به action bar با استفاده از fragment ها

Fragment ها نیز می توانند آیتم و المان هایی را به نوارابزار یا toolbar اپلیکیشن اضافه کنند. برای نیل به این هدف، می بایست متد setHasOptionsMenu(true) را در متد onCreate() از fragment مورد نظر فراخوانی نمایید. framework یا چارچوب نرم افزاری Android در این شرایط متد onCreateOptionsMenu() را در کلاس fragment فراخوانی می کند.در این حالت fragment می تواند آیتم های منو را به toolbar اضافه کند.

3-7-7-تنظیم قابلیت رویت (visibility) و دسترسی toolbar

می توانید قابلیت رویت و visibility نوارابزار را در زمان اجرای برنامه تنظیم نمایید. کد زیر نحوه ی انجام این کار را به نمایش می گذارد.

```
ActionBar actionBar = getActionBar();
actionBar.hide();
// more stuff here...
actionBar.show();
```

شما می توانید متنی که همراه با آیکون اپلیکیشن در زمان اجرای برنامه (runtime) نمایش داده می شود را ویرایش نمایید. تکه کد زیر نحوه ی انجام این کار را نمایش می دهد.

```
ActionBar actionBar = getActionBar();
actionBar.setSubtitle("mytest");
actionBar.setTitle("vogella.com");
```

3-7-8-تخصیص یک عکس drawable به action bar

این امکان نیز برای شما وجود دارد که یک Drawable را از طریق متد ActionBar.setBackgroundDrawable() به عنوان پس زمنیه و background نوارابزار خود تنظیم نمایید.

Toolbar اپلیکیشن خود عکس را مقیاس دهی کرده و با توجه فضای جاری اندازه بندی می کند. از اینرو بهتر است که یک فایل drawable انعطاف پذیر و قابل تنظیم فراهم می کنید. برای مثال می توانید یک عکس 9-patch یا یک drawable با فرمت XML را به عنوان عکس پس زمینه ارائه نمایید.

از ویرایش 4.2 سیستم عامل اندروید، پس زمینه ی action bar را نیز می توان با استفاده از AnimationDrawable متحرک یا پویا نمایی کرد.

9-7-3-پنهان یا کم رنگ کردن دکمه ی پیمایش (diming navigation button)

شما می توانید دکمه ی پیمایش نرم افزاری مورد نظر را در اپلیکیشن اندرویدی خود پنهان نموده و بدین وسیله فضای بیشتری در دسترس داشته باشید. زمانی که کاربر بر روی دکمه ی صفحه کلیک می کند، دکمه ی پیمایش به صورت خودکار نمایان می شود.

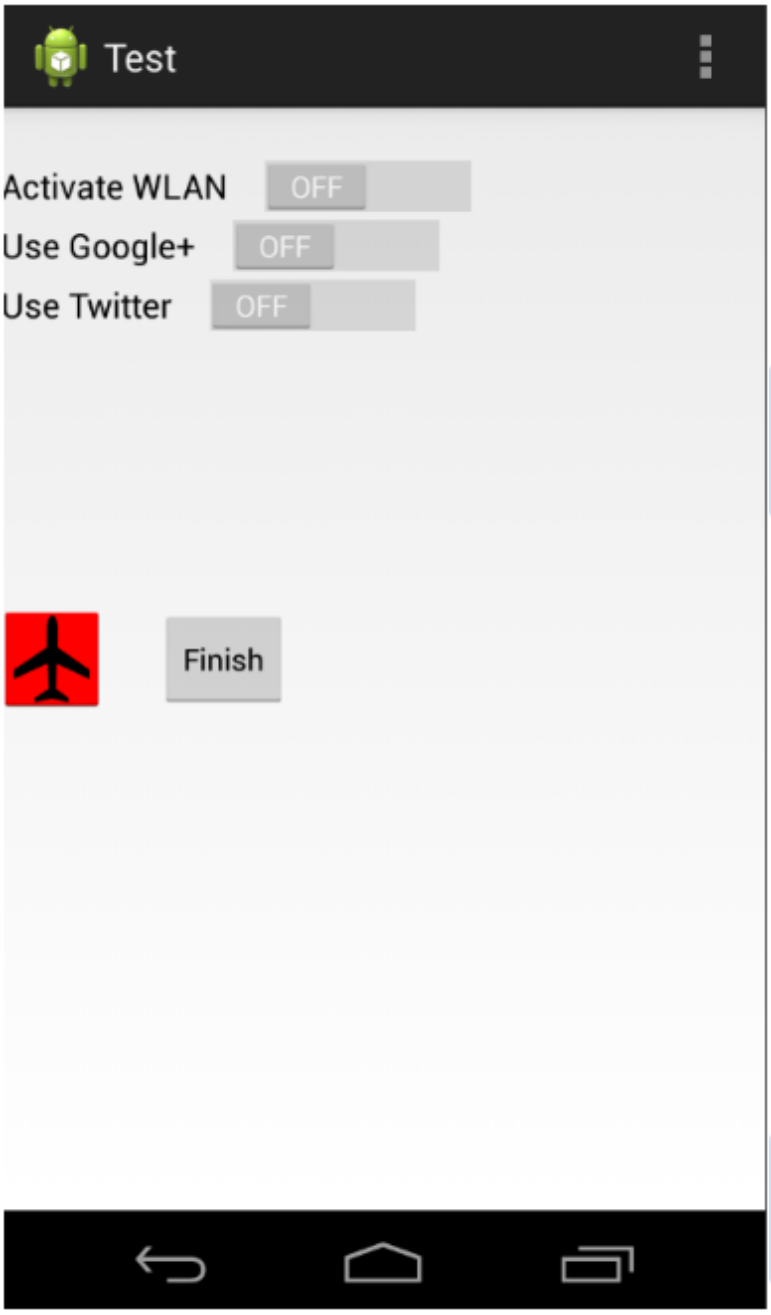
کد لازم برای پیاده سازی این قابلیت در زیر به نمایش گذاشته شده است.

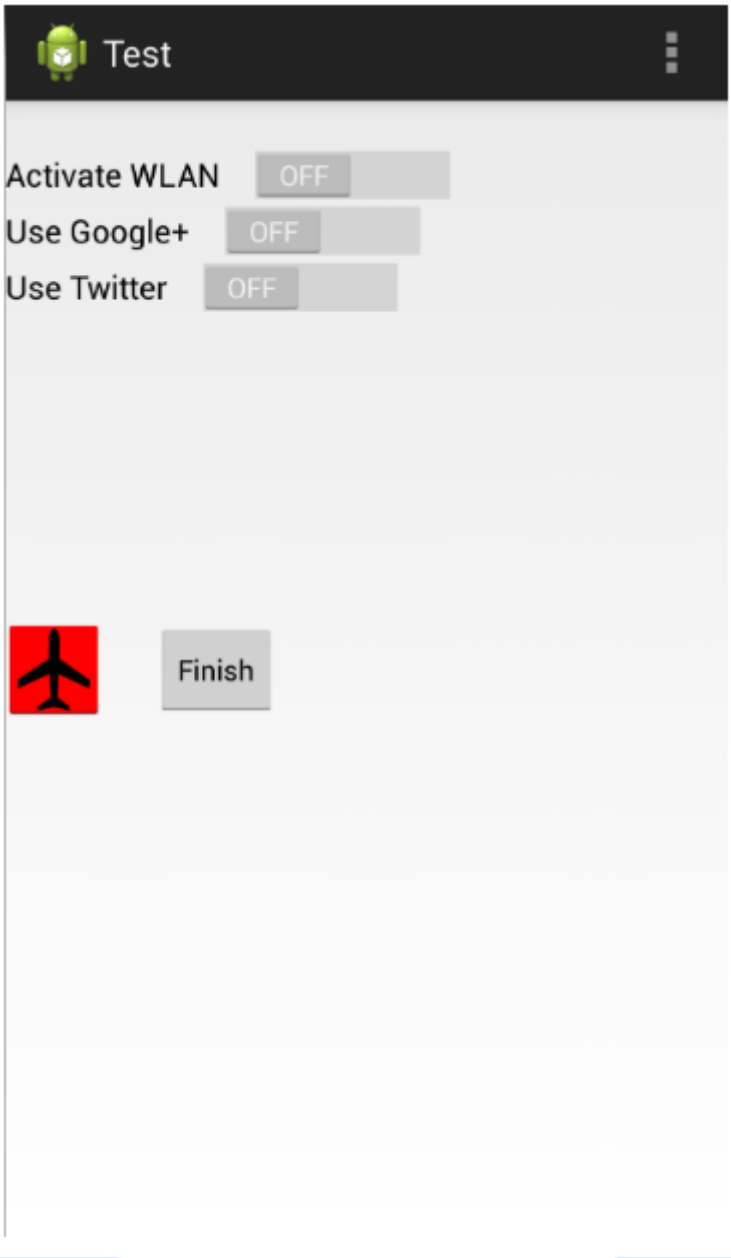
```
getWindow().
    getDecorView().
    setSystemUiVisibility(View.SYSTEM_UI_FLAG_HIDE_NAVIGATION);
```

تصاویر زیر یک اپلیکیشن را با/بدون دکمه های پیمایش نمایش می دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330





10-7-3-استفاده از حالت نمایش تمام صفحه (immersive full screen mode)

از ویرایش 4.4 (ورژن کتابخانه های اندروید یا API 19) به بعد، شما می توانید اپلیکیشن خود را در حالت نمایش تمام صفحه قرار دهید (immersive full screen mode). اولین باری که این اتفاق می افتد، سیستم اندروید اطلاعاتی را به کاربر نشان می دهد. این اطلاعات حاکی از این است که

کاربر می تواند با حرکت downward swipe (لمس سطح نمایشگر به سمت پایین در ناحیه ای که نوارابزار سیستم معمولا قابل مشاهده می باشد) اپلیکیشن را به حالت قبلی بازگردانده و system bar ها را بار دیگر به نمایش بگذارد.

به عنوان مثال، متد زیر activity یا صفحه ی جاری اپلیکیشن را در حالت تمام صفحه قرار می دهد.

```
// This method hides the system bars and resize the content
    private void hideSystemUI() {
        getWindow().getDecorView().setSystemUiVisibility(
                        View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN
                            | View.SYSTEM_UI_FLAG_HIDE_NAVIGATION // hide nav bar
                            | View.SYSTEM_UI_FLAG_FULLSCREEN // hide status bar
                            // remove the following flag for version < API 19
                            | View.SYSTEM_UI_FLAG_IMMERSIVE
                            );
    }
```

11-7-3-پیاده سازی قابلیت دو نیم کردن نوار ابزار (split toolbar)

می توانید به سیستم اعلان نمایید که در صورت عدم وجود فضای کافی، نوارابزار را به صورت خودکار به دو نیم تقسیم کند. جهت فعال سازی این قابلیت می توانید پارامتر android:uiOptions="SplitActionBarWhenNarrow" را در تعریف activity اپلیکیشن خود داخل فایل AndroidManifest.xml لحاظ نمایید.

توجه: با فعال سازی گزینه ی مزبور، اندروید می تواند نوار ابزار را به دو بخش تقسیم کند. سپس سیستم بر اساس فضای جاری و در زمان اجرای برنامه تصمیم می گیرد آیا نوارابزار را به دو بخش تقسیم کند یا خیر.

تمرین: استفاده از Contextual action mode در اپلیکیشن اندرویدی خود

هدف

در تمرین جاری، شما یک contextual action mode به اپلیکیشن خود اضافه خواهید نمود.

12-7-3-ایجاد محتوا و منابع مربوطه/menu resource در پوشه ی res/menu

برای این منظور، یک فایل XML حامل محتوای منو (xml resource menu) با نام actionmode.xml ایجاد نمایید. برای جزئیات بیشتر به <<androidstudio_createmenu" /> مراجعه کنید.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
   <item
      android:id="@+id/action_share"
      android:title="Share it"
      >
   </item>
</menu>
```

13-7-3-تنظیم و ویرایش کد برنامه

ابتدا interface مربوط به fragment را به صورت زیر ارث بری نمایید.

```
public class MyListFragment extends Fragment {
   public void goToActionMode(RssItem item) {
      listener.goToActionMode(item);
   }
   public interface OnItemSelectedListener {
         public void onRssItemSelected(String link);
      public void goToActionMode(RssItem item);
   }
```

کلاس activity خود را جهت پیاده سازی متد جدید و ActionMode.Callback به صورت زیر ویرایش نمایید.

```
package com.example.android.rssreader;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.ActionMode;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.widget.Toast;
import android.widget.Toolbar;
import com.example.android.rssfeedlibrary.RssItem;
public class RssfeedActivity extends Activity
      implements MyListFragment.OnItemSelectedListener,
                  ActionMode.Callback {
   private RssItem selectedRssItem;
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
   Toolbar tb = (Toolbar) findViewById(R.id.toolbar);
   setActionBar(tb);
}
@Override
public boolean onCreateOptionsMenu(Menu menu) {
   Toolbar tb = (Toolbar) findViewById(R.id.toolbar);
   tb.inflateMenu(R.menu.mainmenu);
   tb.setOnMenuItemClickListener(
         new Toolbar.OnMenuItemClickListener() {
            @Override
            public boolean onMenuItemClick(MenuItem item) {
               return onOptionsItemSelected(item);
            }
         });
   return true;
}
//NEW
@Override
public boolean onOptionsItemSelected(MenuItem item) {
   switch (item.getItemId()) {
      case R.id.action_refresh:
            MyListFragment fragment = (MyListFragment) getFragmentManager()
            .findFragmentById(R.id.listFragment);
            fragment.updateListContent();
         break;
      case R.id.action_settings:
         Intent intent = new Intent(this, MyPreferences.class);
         startActivity(intent);
         Toast.makeText(this, "Action Settings selected", Toast.LENGTH_SHORT)
               .show();
         break;
      default:
         break;
   }
   return true;
}
@Override
public void onRssItemSelected(String link) {
   if (getResources().getBoolean(R.bool.twoPaneMode)) {
      DetailFragment fragment = (DetailFragment) getFragmentManager()
            .findFragmentById(R.id.detailFragment);
      fragment.setText(link);
   } else {
      Intent intent = new Intent(getApplicationContext(),
            DetailActivity.class);
      intent.putExtra(DetailActivity.EXTRA_URL, link);
      startActivity(intent);
   }
```

```
    }
    @Override
    public void showContextMenu(RssItem item) {
        this.selectedRssItem = item;
        startActionMode(this);
    }
    @Override
    public void goToActionMode(RssItem item) {
        this.selectedRssItem = item;
        startActionMode(this);
    }
    @Override
    public boolean onCreateActionMode(ActionMode mode, Menu menu) {
        MenuInflater inflater = mode.getMenuInflater();
        inflater.inflate(R.menu.actionmode, menu);
        return true;
    }
    @Override
    public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
        return false;
    }
    @Override
    public boolean onActionItemClicked(ActionMode mode, MenuItem item) {
        Intent intent = new Intent(Intent.ACTION_SEND);
        intent.putExtra(Intent.EXTRA_TEXT, "I found this interesting link" +
                selectedRssItem.getLink());
        intent.setType("text/plain");
        startActivity(intent);
        mode.finish(); // Action picked, so close the CAB
        selectedRssItem = null;
        return true;
    }
    @Override
    public void onDestroyActionMode(ActionMode mode) {
    }
}
```

در کلاس adapter خود، یک LongClickListener پیاده سازی نمایید. با پیاده سازی interface مزبور، contextual action mode فعال و راه اندازی می شود.

```
package com.example.android.rssreader;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;
import com.bumptech.glide.Glide;
import com.example.android.rssfeedlibrary.RssItem;
import java.util.List;
import java.util.Random;
```

```
public class RssItemAdapter
      extends RecyclerView.Adapter<RssItemAdapter.ViewHolder> {
   private List<RssItem> rssItems;
   private MyListFragment myListFragment;
   @Override
   public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
      View v= null;
      v = LayoutInflater.
            from(parent.getContext()).
            inflate(R.layout.rowlayout, parent, false);
      return new ViewHolder(v);
   }
   public static class ViewHolder extends RecyclerView.ViewHolder {
      public View mainLayout;
      public TextView txtHeader;
      public TextView txtFooter;
      public ImageView imageView;
      public ViewHolder(View v) {
         super(v);
         mainLayout = v;
         txtHeader = (TextView) v.findViewById(R.id.rsstitle);
         txtFooter = (TextView) v.findViewById(R.id.rssurl);
         imageView = (ImageView) v.findViewById(R.id.icon);
      }
   }
   @Override
   public void onBindViewHolder(final ViewHolder holder, final int position) {
      final RssItem rssItem = rssItems.get(position);
      holder.txtHeader.setText(rssItem.getTitle());
      holder.txtFooter.setText(rssItem.getLink());
      holder.mainLayout.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            myListFragment.updateDetail(rssItem.getLink());
         }
      });
      holder.mainLayout.setOnLongClickListener(new View.OnLongClickListener() {
         @Override
         public boolean onLongClick(View v) {
            myListFragment.goToActionMode(rssItem);
            return true;
         }
      });
   }
   @Override
   public int getItemCount() {
      return rssItems.size();
   }
   public RssItemAdapter(List<RssItem> rssItems, MyListFragment myListFragment) {
      this.rssItems = rssItems;
      this.myListFragment = myListFragment;
   }
```

}

Dynamic-8-3 تعریف کردن action bar

View–3–8–1 هایی با پیاده سازی اختصاصی در action bar

می توانید یک view اختصاصی به action bar اضافه نمایید. برای مثال، یک button یا text field در نوار ابزار اپلیکیشن خود داشته باشید.

برای این منظور، لازم است متد setCustomView از کلاس ActionView را بکار ببرید. سپس بایستی با فراخوانی متد setDisplayOptions() و ارسال پارامتر ActionBar.DISPLAY_SHOW_CUSTOM به آن، امکان نمایش view های اختصاصی در toolbar را فعال سازی نمایید.

به طور مثال، می توانید یک فایل layout تعریف کنید که دربردارنده ی المان EditText می باشد.

```
<?xml version="1.0" encoding="utf-8"?>
<EditText xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/searchfield"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:inputType="textFilter" >
</EditText>
```

این layout را می توان به وسیله ی کد زیر به action bar در activity مورد نظر متصل کرد. تکه کد زیر علاوه بر متصل کردن layout، یک گوش فراخوان یا listener نیز به custom view الحاق می کند.

```
package com.vogella.android.actionbar.customviews;
import android.app.ActionBar;
import android.app.Activity;
import android.os.Bundle;
import android.view.KeyEvent;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.TextView.OnEditorActionListener;
import android.widget.Toast;
public class MainActivity extends Activity {
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

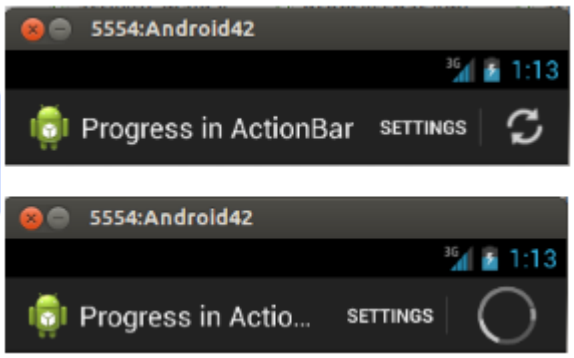
88146323 - 88446780 - 88146330

```
        ActionBar actionBar = getActionBar();
        // add the custom view to the action bar
        actionBar.setCustomView(R.layout.actionbar_view);
        EditText search = (EditText) actionBar.getCustomView().findViewById(
                R.id.searchfield);
        search.setOnEditorActionListener(new OnEditorActionListener() {
            @Override
            public boolean onEditorAction(TextView v, int actionId,
                    KeyEvent event) {
                Toast.makeText(MainActivity.this, "Search triggered",
Toast.LENGTH_LONG).show();
                return false;
            }
        });
        actionBar.setDisplayOptions(ActionBar.DISPLAY_SHOW_CUSTOM
                | ActionBar.DISPLAY_SHOW_HOME);
    }
}
```

Action view-3-8-2

Action view یک کامپوننت UI یا widget هست که بجای دکمه ی آیتم (action در نوارابزار) نمایش داده می شود. با استفاده از این ویژگی می توانید یک آیتم (action) را با آبجکت Progressbar یا هر view مجاز دیگری جایگزین نمایید. به منظور تعریف یک action view جایگزین برای آیتم (action) در toolbar، می توانید خصیصه (attribute) android:actionLayout را جهت ایجاد فایل layout و خصیصه ی android:actionViewClass را جهت تعریف widget class مورد استفاده قرار دهید.

این جایگزینی در تصاویر زیر به نمایش گذاشته شده است.



Activity زیر آیکون را در زمان اجرا برنامه با action view ای که حامل یک آبجکت ProgressBar می باشد، جایگزین می نماید.

```
package com.vogella.android.actionbar.progress;
import android.app.ActionBar;
import android.app.Activity;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
public class MainActivity extends Activity {
	private MenuItem menuItem;
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_main);
		ActionBar actionBar = getActionBar();
		actionBar.setDisplayOptions(ActionBar.DISPLAY_SHOW_HOME
				| ActionBar.DISPLAY_SHOW_TITLE | ActionBar.DISPLAY_SHOW_CUSTOM);
	}
	@Override
	public boolean onCreateOptionsMenu(Menu menu) {
		getMenuInflater().inflate(R.menu.main, menu);
		return true;
	}
	@Override
	public boolean onOptionsItemSelected(MenuItem item) {
		switch (item.getItemId()) {
		case R.id.menu_load:
			menuItem = item;
			menuItem.setActionView(R.layout.progressbar);
			menuItem.expandActionView();
			TestTask task = new TestTask();
			task.execute("test");
			break;
		default:
			break;
		}
		return true;
	}
	private class TestTask extends AsyncTask<String, Void, String> {
		@Override
		protected String doInBackground(String... params) {
			// Simulate something long running
			try {
				Thread.sleep(2000);
			} catch (InterruptedException e) {
				e.printStackTrace();
			}
			return null;
		}
		@Override
		protected void onPostExecute(String result) {
			menuItem.collapseActionView();
			menuItem.setActionView(null);
```

```
        }
    };
}
```

کد زیر، محتوای layout مورد استفاده ی action view را نشان می دهد.

```
<?xml version="1.0" encoding="utf-8"?>
<ProgressBar xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/progressBar2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
</ProgressBar>
```

کد زیر فایل های XML که برای منو استفاده می شود را نشان می دهد.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
  <item
    android:id="@+id/menu_settings"
    android:orderInCategory="100"
    android:showAsAction="always"
    android:title="Settings"
    />
  <item
    android:id="@+id/menu_load"
    android:icon="@drawable/navigation_refresh"
    android:orderInCategory="200"
    android:showAsAction="always"
    android:title="Load"/>
</menu>
```

Action provider-9-3

Action provider_3_9_1 چیست و چه کاربردی دارد؟

Action provider در واقع همان action است که layout اختصاصی خود را دارد. Action در ابتدا ممکن است به صورت یک button یا آیتم در منو ظاهر شود، اما زمانی که کاربر بر روی action کلیک می کند، action provider رفتار action را بر اساس تعریف شما، کنترل و هدایت می کند. برای مثال، action bar می تواند با نمایش یک منو به کلیک کاربر واکنش نشان دهد. به عبارت دیگر، action provider به شما امکان می دهد تا قابلیت تعامل و رفتار پیچیده را در قالب تنها یک action به راحتی پیاده سازی نمایید. این action قادر است action view هایی تولید کند که در action bar استفاده می شوند، زیرمنوها را به صورت پویا و در زمان اجرای اپلیکیشن با داده پر کند و فراخوانی های پیش فرض action را مدیریت نماید.

کلاسی که action bar از آن مشتق می شود، کلاس پایه/پدر ActionProvider می باشد.

در حال حاضر، محیط اندروید (Android platform) دو action provider زیر را در اختیار توسعه دهندگان قرار می دهد: 1. MediaRouteActionProvider 2. ShareActionProvider.

تمرین: استفاده از ShareActionProvider

تمرین زیر استفاده از ShareActionProvider را نمایش می دهد. این action provider به شما اجازه می دهد تا محتوای انتخابی را از اپلیکیشن هایی که Intent.ACTION_SEND را ثبت کرده اند، دریافت نمایید.

جهت استفاده از ShareActionProvider، می بایست یک آیتم در منو ویژه ی آن تعریف کرده و intent ای به آن تخصیص دهید که داده های مورد نظر برای اشتراک را دربرمی گیرد.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
  <item android:id="@+id/menu_share"
      android:title="Share"
      android:showAsAction="ifRoom"
android:actionProviderClass="android.widget.ShareActionProvider" />
  <item
    android:id="@+id/item1"
    android:showAsAction="ifRoom"
    android:title="More entries...">
  </item>
</menu>
@Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu);
        // Get the ActionProvider for later usage
        provider = (ShareActionProvider) menu.findItem(R.id.menu_share)
                .getActionProvider();
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case R.id.menu_share:
            doShare();
            break;
        default:
            break;
        }
        return true;
```

```
}
public void doShare() {
    // populate the share intent with data
    Intent intent = new Intent(Intent.ACTION_SEND);
    intent.setType("text/plain");
    intent.putExtra(Intent.EXTRA_TEXT, "This is a message for you");
    provider.setShareIntent(intent);
}
```

10-3-پیمایش در اپلیکیشن از طریق آیکون

10-3-1-استفاده از آیکون اپلیکیشن در action bar جهت راهبری به صفحه ی اصلی

در action bar آیکونی از اپلیکیشن خود را مشاهده می کنید. آیتم مزبور home icon نام داشته و شما می بایست یک action به آن اختصاص دهید. بهتر است کاربر با کلیک بر روی آیکون، به صفحه ی اصلی برنامه (main activity) هدایت شود.

پس از انتخاب شدن action مورد نظر، متد onOptionsItemSelected() همراه با آیتم یا action ای صدا خورده می شود که شناسه یا ID آن android.R.id.home باشد.

پیش از ویرایش 4.1 اندروید، توسعه دهنده می بایست شناسه android.R.id.home را در متد onOptionMenuItemSelected() بکار می برد و امکان انتخاب دکمه ی home را فعال سازی می کرد. این عملیات در کد زیر به نمایش گذاشته شده است. آیکون home در SecondActivity کاربر را به MainActivity در اپلیکیشن هدایت می کند.

```
package com.vogella.android.actionbar.homebutton;
import android.os.Bundle;
import android.app.ActionBar;
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.MenuItem;
```

```
public class SecondActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // enable the home button
        ActionBar actionBar = getActionBar();
        actionBar.setHomeButtonEnabled(true);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case android.R.id.home:
            Intent intent = new Intent(this, MainActivity.class);
            intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
            startActivity(intent);
            break;
            // Something else
        case R.id.action_settings:
            intent = new Intent(this, ThirdActivity.class);
            startActivity(intent);
        default:
            break;
        }
        return super.onOptionsItemSelected(item);
    }
}
```

نکته: از Android 4.1 به بعد، برنامه نویس می تواند با برابر قرار دادن مقدار parentActivityName با نام activity پدر (برای مثال "MainActivity") در فایل تنظیمات اپلیکیشن (AndroidManifest.xml)، به راحتی کاربر را از activity دوم به activity اصلی هدایت کرده و تعیین کند که کاربر با کلیک بر روی آیکون home به activity اصلی برنامه ارجاع داده شود.

```
<activity
 android:name="SecondActivity"
 android:label="@string/app_name"
 android:parentActivityName="MainActivity" >
</activity>
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-10-3-استفاده از آیکون اپلیکیشن به عنوان دکمه ی Up

می توانید از آیکون اپلیکیشن به عنوان دکمه ی Up استفاده نمایید به طوری که کاربر با کلیک بر روی آن به activity میزبان که پدر activity کنونی هست، هدایت شود. دکمه ی back در دستگاه همیشه کاربر را به activity یا صفحه ی قبلی بازگشت می دهد.

رفتار و کاربرد هر دو دکمه را می توان متفاوت تعریف کرد، برای مثال زمانی که کاربر آپشنی را فعال کند که به او اجازه ی نوشتن ایمیل از صفحه ی اصلی را می دهد، برنامه را گونه ای تعریف کرده باشید که کلیک روی دکمه ی back کاربر را به صفحه ی اصلی و کلیک بر روی دکمه ی Up کاربر را به activity ای که نمایی مختصر از تمامی ایمیل ها ارائه می دهد، هدایت نماید.

به منظور فعال سازی Up، می توانید تکه کد زیر را در activity خود اضافه نمایید.

```
actionBar.setDisplayUseLogoEnabled(false);
actionBar.setDisplayHomeAsUpEnabled(true);
```

نکته: درج تکه کد بالا در کلاس activity صرفا Up را بر روی آیکون home فعال می سازد. شما بایستی رفتار لازم را در کلاس activity خود و داخل متد onOptionsItemSelected() پیاده سازی نمایید. action مربوطه همچنان با همان شناسه ی android.R.id.home فراخوانی می شود.

تمرین: پیاده سازی ActionBar

پروژه

در این بخش action و آیتم هایی تعاملی در action bar ایجاد خواهید نمود که به انتخاب کاربر واکنش نشان می دهند.

تمرینی که در بخش حاضر دنبال می کنیم مبتنی بر آموزش fragment است. اگر قبلا پروژه ی آن را نوشته اید می توانید همان را دوباره استفاده کنید.

تمرین: اضافه کردن toolbar به اپلیکیشن

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

افزودن آیکون refresh در نوار ابزار

پروژه ی RSS Reader را ادامه دهید. با طی نمودن میسر زیر یک drawable به نام ic_refresh ایجاد نمایید: File ▸ New ▸ Vector Asset. یک تصویر مناسب از نمونه های جاری انتخاب نمایید.

3-10-3-اضافه کردن منابع لازم برای منو

یک فایل XML جدید به نام mainmenu.xml ایجاد نمایید. سپس یک آیتم به فایل مورد نظر اضافه کنید، طوری که فایل XML نهایی ظاهری مشابه زیر داشته باشد. می بایست این آیتم ها را به صورت دستی تایپ کنید چرا که محیط کاری Android Studio تا زمان نگارش آموزش حاضر، این قابلیت را ندارد که کدهای ساخت منوها را به صورت خودکار برای توسعه دهنده تکمیل کند.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
  <item
     android:id="@+id/action_settings"
     android:title="Settings"
     android:showAsAction="never"
     >
  </item>
</menu>
```

یک فایل XML جدید به نام listfragment_menu.xml برای منوی اپلیکیشن خود تولید نمایید. سپس یک آیتم مانند زیر به فایل اضافه کنید تا فایل XML نهایی ظاهری مشابه زیر داشته باشد. می بایست این آیتم ها را خود به صورت دستی بنویسید چرا که تا زمان نگارش آموزش حاضر، محیط کاری Android Studio قابلیت تکمیل خودکار کدها را ندارد.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:android="http://schemas.android.com/apk/res/android">
   <item android:title="Refresh"
      android:icon="@drawable/ic_refresh"
      android:id="@+id/action_refresh"
      android:enabled="false"
      android:showAsAction="ifRoom" />
</menu>
```

4-10-3-غیرفعال کردن action bar پیش فرض از طریق تگ style

از آنجایی که برای پروژه از toolbar اختصاصی خود استفاده می کنید، لازم است action bar پیش فرض را در فایل res/values/styles.xml غیر فعال نمایید.

```
<resources>
    <!-- Base application theme. -->
    <style name="AppTheme" parent="android:Theme.Material.Light.NoActionBar">
        <!-- Customize your theme here. -->
    </style>
</resources>
```

5-10-3-اضافه کردن یک آبجکت (view) toolbar به layout activity

فایل های layout مربوط به RssfeedActivity خود را طوری ویرایش کنید که آیتم (toolbar entry) زیر را دربرداشته باشد.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:orientation="horizontal"
        android:layout_weight="1"
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        >
        <fragment
            android:id="@+id/listFragment"
            class="com.example.android.rssreader.MyListFragment"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1"          tools:layout="@layout/fragment_rsslist_overview"></fragment>
        <fragment
            android:id="@+id/detailFragment"
            class="com.example.android.rssreader.DetailFragment"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="2"
            tools:layout="@layout/fragment_rssitem_detail" />
    </LinearLayout>
```

```
    <Toolbar
      android:id="@+id/toolbar"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      >
    </Toolbar>
</LinearLayout>
```

همین کار را برای نمای عمودی (portrait mode) نیز انجام دهید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".RssfeedActivity"
    android:orientation="vertical">
    <FrameLayout
      android:id="@+id/fragment_container"
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:layout_weight="1"
      android:orientation="horizontal"
      android:paddingBottom="@dimen/activity_vertical_margin"
      android:paddingLeft="@dimen/activity_horizontal_margin"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="@dimen/activity_vertical_margin"
      >
    </FrameLayout>
    <Toolbar
      android:id="@+id/toolbar"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      >
    </Toolbar>
</LinearLayout>
```

بدنه ی کلاس RssfeedActivity خود را جهت تنظیم آبجکت toolbar مانند زیر ویرایش نمایید.

```
package com.example.android.rssreader;
import android.app.Activity;
import android.app.Fragment;
import android.app.FragmentTransaction;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.FrameLayout;
import android.widget.Toast;
import android.widget.Toolbar;
public class RssfeedActivity extends Activity implements
      MyListFragment.OnItemSelectedListener {
    // as before
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_rssfeed);
        //
        Toolbar tb = (Toolbar) findViewById(R.id.toolbar);
        setActionBar(tb);
        // more code as before
    }
    //
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        Toolbar tb = (Toolbar) findViewById(R.id.toolbar);
        tb.inflateMenu(R.menu.mainmenu);
        tb.setOnMenuItemClickListener(
                new Toolbar.OnMenuItemClickListener() {
                    @Override
                    public boolean onMenuItemClick(MenuItem item) {
                        return onOptionsItemSelected(item);
                    }
                });
        return true;
    }
    //
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_settings:
                Toast.makeText(this, "Action Settings selected", Toast.LENGTH_SHORT).show();
                return true;
            default:
                break;
        }
        return super.onOptionsItemSelected(item);
    }
    @Override
    public void onRssItemSelected(String link) {
        // more code as before
    }
}
```

تست اپلیکیشن

اپلیکیشن خود را اجرا نموده و هر دو action را انتخاب کنید. بایستی با انتخاب آیتم های مختلف در نوارابزار یک پیغام Toast برای کاربر به نمایش گذاشته شود.

تمرین: فعال سازی عملیات refresh/بروز رسانی آیتم های MyListFragment از طریق toolbar

برای تست اپلیکیشن شما می خواهید recycler view تنها زمانی که کاربر refresh را از toolbar انتخاب می کند (از طریق swipe-to-refresh یا قابلیت لمس اپلکیشن به پایین جهت بروز رسانی آیتم های لیست) با داده پر شود. بنابراین، recycler view یا لیست شما در ابتدای امر بایستی تهی باشد.

6-10-3-حذف داده های اولیه

در پروژه ی RSS Reader خود، فراخوانی updateListContent() را از بدنه ی متد onCreate() کلاس MyListFragment خود حذف نمایید.

7-10-3-بروز رسانی MyListFragment، در صورتی که refresh انتخاب شد

منطق Refresh را در کلاس RssfeedActivity خود پیاده سازی نمایید. حال زمانی که کاربر refresh را انتخاب می کند، متد updateListContent() در MyListFragment صدا خورده می شود.

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
      switch (item.getItemId()) {
    case R.id.action_refresh:
    //TODO
    // find fragment via FragmentManager and call updateListContent() on it.
  break;
    default:
  break;
 }
      return true;
}
```

بخش سوم :

پیاده سازی الگو توسعه و قابلیت Swipe-To-Refresh (کشیدن صفحه به پایین جهت بروز رسانی لیست) در اپلیکیشن

آموزش حاضر به شرح پیاده سازی الگو توسعه ی swipe to refresh در اپلیکیشن می پردازد.

3-11-پیاده سازی قابلیت بروز رسانی آیتم های لیست با استفاده از swipe to refresh

اندروید همچنین یک کامپوننت UI یا widget ارائه می دهد که در واقع الگو توسعه ی swipe-to-refresh را پیاده سازی کرده و به کاربر این امکان را می دهد تا با کشیدن صفحه به پایین (swipe عمودی بر روی نمایشگر) آیتم های لیست را بروز رسانی کند (عملیات update را فراخوانی نماید).

این قابلیت توسط widget ای به نام SwipeRefreshLayout پیاده سازی می شود. این کامپوننت UI در حقیقت عمل کشیدن صفحه به پایین یا swipe عمودی را تشخیص داده، یک progress bar مجزا به نمایش می گذارد و متدهای callback را در اپلیکیشن فراخوانی می کند.

جهت پشتیبانی از widget نام برده، ابتدا می بایست آن را به عنوان میزبان و پدر view مربوطه در فایل layout مورد نظر تعریف نمایید و سپس رفتار یا عملیات بروز رسانی (refresh) که با swipe عمودی کاربر بر روی نمایشگر فعال می شود را پیاده سازی کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

به منظور استفاده از swipe to refresh لازم است کتابخانه ی مربوطه (dependency to the support library) را در فایل Gradle build اپلیکیشن خود اضافه کرده باشید.

```
dependencies {
   compile 'com.android.support:support-v4:24.0.0'
}
```

```
<android.support.v4.widget.SwipeRefreshLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/swiperefresh"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <android.support.v7.widget.RecyclerView
      android:id="@+id/my_recycler_view"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:scrollbars="vertical" />
</android.support.v4.widget.SwipeRefreshLayout>
```

توجه: پس از پیاده سازی الگو توسعه ی swipe-to-refresh، لازم است برای آن دسته از کاربرانی که قابلیت لمس نمایشگر به پایین را ندارند، یک دکمه ی refresh به overflow menu خود داخل toolbar اضافه نمایید.

داخل بدنه ی متد SwipeRefreshLayout.OnRefreshListener، منطق بروز رسانی view را پیاده سازی نمایید.

```
mySwipeRefreshLayout.setOnRefreshListener(
   new SwipeRefreshLayout.OnRefreshListener() {
      @Override
      public void onRefresh() {
         doYourUpdate();
      }
   }
);
private void doYourUpdate() {
  // TODO implement a refresh
  setRefreshing(false); // Disables the refresh icon
}
```

چنانچه action مزبور (آیتم refresh) را در منوی overflow مستقر در toolbar خود نیز اضافه کردید، در آن صورت لازم است آیکون refresh را نیز در UI به نمایش بگذارید.

```
public boolean onOptionsItemSelected(MenuItem item) {
   switch (item.getItemId()) {
      case R.id.menu_refresh:
        // signal SwipeRefreshLayout to start the progress indicator
```

```
            mySwipeRefreshLayout.setRefreshing(true);
            doYourUpdate();

            return true;
        }
        return super.onOptionsItemSelected(item);
    }
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فصل سوم

کار با فایل ، شبکه و پردازش های Asynchronous

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش اول :

دیتابیس SQLite و content provider

این مبحث به شرح مفهوم دیتابیس در اندروید می پردازد. سپس به ترتیب نحوه ی استفاده از دیتابیس SQLite در اپلیکیشن، استفاده از ContentProvider جاری و ایجاد نمونه های جدید از آن را برای شما تشریح می کند. در نهایت نمونه ای از کاربرد چارچوب نرم افزاری (framework) Loader جهت بارگذاری داده به صورت ناهمزمان (asynch) در برنامه ی اندرویدی را برای شما به نمایش می گذارد.

پروژه ی این مبحث در محیط کاری Eclipse 4.2 نوشته شده، مبتنی بر ویرایش 1.6 زبان Java و ورژن 4.2 سیستم عامل اندروید می باشد.

4-1-SQLite و Android

1-1-4-SQLite چیست و چه کاربردی دارد؟

SQLite یک دیتابیس کد باز (Open Source) بوده، از امکانات و ویژگی های متعارف دیتابیس های رابطی نظیر ساختار نگارشی SQL (syntax)، تراکنش ها (transactions) و دستورات آماده پشتیبانی می کند. دیتابیس مزبور در زمان اجرای اپلیکیشن حافظه ی کمی را مصرف می کند (حدودا 250 کیلوبایت). این امر سبب شده که SQLite گزینه ی خوبی برای جاسازی در سایر runtime ها تلقی شود.

SQLite از انواع داده ای همچون TEXT (معادل String در جاوا)، INTEGER (معادل long در جاوا) و REAL (مشابه double در جاوا) پشتیبانی می کند. سایر نوع های داده ای می بایست قبل از قابلیت ذخیره در دیتابیس، به یکی از فیلدهای نام برده تبدیل شوند.

SQLite به صورت خودکار قابلیت اعتبارسنجی انواع داده ای درج شده در ستون ها را ندارد. به عنوان مثال، ممکن است مقداری از نوع integer را در ستون رشته ای ذخیره کرده یا بالعکس. خواهید دید که دیتابیس مذکور از شما ایراد نخواهد گرفت.

4-1-2- SQLite در اندروید

SQLite تقریبا در تمامی دستگاه های اندروید جاسازی شده است. برای استفاده از این دیتابیس، توسعه دهنده نیازی به نصب و تنظیم آن ندارد. مدیریت (admin) دیتابیس نیز به صورت خودکار توسط خود اندروید انجام می شود.

کافی است دستورات SQL ایجاد و آپدیت دیتابیس را تعریف نموده و فراخوانی کنید. پس از آن باقی عملیات لازم را خود محیط و بستر اجرای اندروید (platform) به دست می گیرد.

دسترسی به دیتابیس SQLite به صورت خودکار دسترسی به سیستم فایل را نیز به دنبال دارد و به همین دلیل عملیات مرتبط با دیتابیس می تواند کند باشد.

دیتابیس اپلیکیشن به صورت خودکار در آدرس DATA/data/APP_NAME/databases/FILENAME ذخیره می شود.

بخش های آدرس فوق بر اساس قوانین و قواعد خاصی ساخته می شوند. DATA آدرس یا مسیری است که متد Environment.getDataDirectory() برمی گرداند. APP_NAME اسم اپلیکیشن شما است و در پایان، FILENAME اسمی است که در کد اپلیکیشن خود برای دیتابیس تعریف می کنید.

4-2-معماری SQLite

4-2-1- Package ها

پکیج android.database تمامی اطلاعات ضروری برای کار با دیتابیس را شامل می شود. پکیج android.database.sqlite تمامی کلاس های ویژه ی SQLite را دربرمی گیرد.

2-2-4-ایجاد و آپدیت دیتابیس با SQLiteOpenHelper

به منظور ایجاد و ارتقا دیتابیس در اپلیکیشن اندرویدی خود، لازم است یک کلاس فرزند (subclass) از SQLiteOpenHelper ایجاد نمایید. سپس در تابع سازنده (constructor) کلاس فرزند، متد super() از SQLiteOpenHelper را فراخوانی نموده و اسم و ورژن دیتابیس را مشخص نمایید.

در این کلاس می بایست پیاده سازی متدهای زیر را جهت ایجاد و آپدیت دیتابیس خود، بازنویسی (override) نمایید.

- onCreate() – چارچوب نرم افزاری اندروید (framework) این متد را زمانی صدا می زند که می خواهید به دیتابس دسترسی داشته باشید، اما دیتابیس هنوز ایجاد نشده است.
- onUpgrade() – هنگامی فراخوانی می شود که ورژن دیتابیس در کد اپلیکیشن ارتقا یافته باشد. این متد به شما اجازه می دهد تا schema دیتابیس جاری را بروز رسانی نموده یا دیتابیس جاری را حذف کنید و سپس آن را مجددا با متد onCreate() ایجاد نمایید.

هر دو متد یک آبجکت SQLiteDatabase به عنوان پارامتر می گیرند. این آبجکت جاوا در واقع همان دیتابیس است.

کلاس SQLiteOpenHelper با ارائه ی دو متد getReadableDatabase() و getWriteableDatabase() دسترسی به آبجکت SQLiteDatabase را به ترتیب به صورت فقط خواندنی و نوشتنی برای توسعه دهنده فراهم می آورد.

جداول دیتابیس بایستی id_ را به عنوان کلید اصلی (primary key) بکار ببرند. بسیاری از رفتارها و توابع اندروید از این استاندارد پیروی می کنند.

نکته: توصیه می شود برای هر کلاس یک جدول جداگانه ایجاد نمایید. این کلاس ها متدهای استاتیک onCreate() و onUpgrade() را تعریف می کند. توابع مذکور در متدهای متناظر داخل SQLiteOpenHelper فراخوانی می شوند. با پیروی از این روش، حتی اگر تعداد جداول دیتابیس بسیار بالا باشد، پیاده سازی SQLiteOpenHelper به راحتی قابل دسترسی و خواندن خواهد بود.

3-2-4-کلاس پدر SQLiteDatabase

SQLiteDatabase کلاس پایه ای است که برای کار با دیتابیس می بایست از آن ارث بری کنید. این کلاس متدهایی را جهت بازکردن، کوئری گرفتن، آپدیت کردن و بستن دیتابیس در اختیار توسعه دهنده قرار می دهد. به عبارت دقیق تر، SQLiteDatabase سه متد بسیار مهم insert()، update() و delete() را ارائه می دهد.

علاوه بر آن، کلاس نام برده با ارائه ی متد execSQL() این امکان را برای برنامه نویس فراهم کرده تا دستور SQL را به صورت مستقیم اجرا کند.

آبجکت ContentValues جفت های کلید/مقدار (key/value) را ایجاد می کنند. Key همان id یا شناسه ی ستون جدول است و value نیز محتوا یا مقدار رکورد جدول در این ستون می باشد. از این آبجکت برای درج مقدار جدید و آپدیت مقادیر جاری دیتابیس استفاده می شود.

جهت ایجاد query ها و درخواست داده از دیتابیس می توانید به دو روش اقدام کنید: 1. می توانید یکی از دو متد rawQuery() و query() را فراخوانی نمایید 2. از کلاس SQLiteQueryBuilder استفاده کنید.

rawQuery() یک دستور SQL را مستقیما به عنوان ورودی می پذیرد.

query() یک interface ساخت یافته و الگوی پیاده سازی آماده برای تعریف دستور SQL ارائه می دهد.

SQLiteQueryBuilder یک کلاس کمکی (Convenience) است که این امکان را به شما می دهد تا query های sql را به راحتی بسازید.

مثالی از پیاده سازی متد rawQuery()

نمونه کد زیر متد rawQuery() را فراخوانی می کند.

```
Cursor cursor = getReadableDatabase().
      rawQuery("select * from todo where _id = ?", new String[] { id });
```

مثالی از پیاده سازی متد query()

نمونه کد زیر متد query() را فراخوانی می کند.

```
return database.query(DATABASE_TABLE,
    new String[] { KEY_ROWID, KEY_CATEGORY, KEY_SUMMARY, KEY_DESCRIPTION },
    null, null, null, null, null);
```

متد query() پارامترهای زیر را به عنوان ورودی می گیرد.

جدول پارامترهای ورودی متد query()	
پارامتر	شرح
String dbName	اسم جدولی که از آن کوئری گرفته می شود.
String[] columnNames	لیستی از ستون ها که باید در خروجی بازگردانده شوند. با ارسال مقدار "null" به عنوان پارامتر ورودی، تمامی ستون ها در خروجی بازگردانی می شوند.
String whereClause	عبارت شرطی Where که به مثابه ی فیلتری جهت واکشی داده های مورد نیاز استفاده می شود. ارسال مقدار null، تمامی داده ها را در خروجی برمی گرداند.
String[] selectionArgs	می توانید s? را در عبارت ""whereClause" نیز بکار ببرید. این مکان نگهدار (placeholder) سپس با مقادیر واقعی از آرایه ی selectionArgs جایگزین می شوند.
String[] groupBy	یک فیلتر است که نحوه ی گروه بندی سطرها را مشخص می کند. ارسال مقدار null به عنوان پارامتر سبب می شود که سطرها مرتب نشوند.
String[] having	این پارامتر نتیجه را به آن گروه از سطرها که با شرط اعمال شده منطبق هستند، محدود می کند. به عبارت دیگر برای اعمال شرط و محدود کردن خروجی در گروه ها مورد استفاده قرار می گیرد.
String[] orderBy	ستون هایی از جدول که برای مرتب سازی داده ها مورد استفاده قرار می گیرد را شامل می شود. با ارسال null به عنوان پارامتر هیچ مرتب سازی صورت نمی گیرد.

اگر شرطی برای اعمال و محدودسازی نتایج نیاز نباشد، می توانید null را به عنوان آرگومان ارسال نمایید. به عنوان مثال چنانچه لزومی برای اعمال شرط بر روی گروه ها نیست، می توانید null را پاس دهید.

در متد query() همان طور که مشاهده می کنید، بجای عبارت شرطی "whereClause" از این ساختار استفاده شده است: "_id=19 and summary=?". در واقع عملگر "=" جایگزین عبارت where می شود.

اگر با استفاده از ?مقادیر موقتی و مکان نگهدار تعریف نمایید (placeholder)، در آن صورت می بایست آن ها را به عنوان selectionArgs به کوئری ارسال کنید.

4-2-4-آبجکت Cursor

کوئری در خروجی یک آبجکت Cursor برمی گرداند. Cursor همان نتیجه ی کوئری و سطرهای واکشی شده از دیتابیس است و اساسا به یک سطر از نتیجه ی کوئری اشاره دارد. این روش بازیابی اطلاعات به اندروید اجازه می دهد تا نتایج کوئری را به راحتی buffer کند چرا که در این صورت سیستم دیگری نیازی به بارگذاری تمامی داده ها در حافظه ندارد.

برای بدست آوردن تعداد المان های کوئری خروجی می توانید متد getCount() را بکار ببرید.

به منظور حرکت بین سطرهای فردی، می توانید moveToFirst() و moveToNext() را فراخوانی نمایید. با استفاده از متد isAfterLast() می توانید بررسی کنید آیا به انتهای نتیجه ی کوئری رسیده اید یا خیر.

Cursor مجموعه متدهای (typed) وابسته به نوع get*() را در اختیار توسعه دهنده قرار می دهد. از جمله می توان به getLong(columnIndex)، getString(columnIndex) جهت دسترسی به مقادیر و داده های ستون با توجه به اندیس کنونی نتیجه اشاره کرد. getString(columnIndex) برای بازیابی داده از نوع رشته و getLong(columnIndex) برای واکشی داده هایی از نوع Long استفاده می شود. پارامتر "columnIndex" همان طور که از نامش پیدا است، شماره ی مکان قرار گیری و اندیس ستون مورد درخواست می باشد که داده را بر اساس آن استخراج می کند.

Cursor همچنین متدی به نام getColumnIndexOrThrow(String) را ارائه می دهد. این متد اسم ستون را به عنوان ورودی گرفته و در خروجی شماره ی مکان قرار گیری ستون را برمی گرداند.

لازم به ذکر است که آبجکت Cursor را می بایست با فراخوانی متد close() ببندید.

4-2-5- ListView ها، ListActivityها و SimpleCursorAdapter

ListView ها آبجکت های view ای هستند که لیستی از المان ها را برای کاربر به نمایش می گذارند.

ListActivitiy ها نیز activity های ویژه ای هستند که استفاده از ListView ها را آسان تر می سازند.

برای کار با دیتابیس ها و ListView ها می توانید از SimpleCursorAdapter استفاده نمایید. SimpleCursorAdapter به شما این امکان را می دهد تا برای هر سطر از ListView ها یک layout تنظیم نمایید.

شما بایستی آرایه ای تعریف کنید که دربردارنده ی اسم ستون ها می باشد و سپس آرایه ی دیگری جهت نگهداری ID و شناسه ی View هایی که باید با داده پر شوند ایجاد نمایید.

کلاس SimpleCursorAdapter ستون ها را بر اساس Cursor ارسالی، به view ها نگاشت می کند.

جهت دسترسی به Cursor می بایست کلاس Loader را بکار ببرید.

4-3-آموزش: استفاده از SQLite

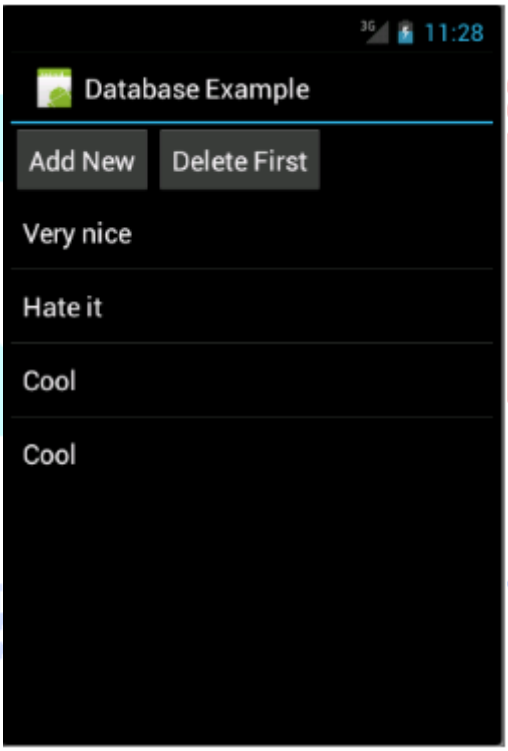
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-3-4-مقدمه ای بر پروژه

در این بخش نحوه ی کار با دیتابیس SQLite را به صورت کاربردی خواهید آموخت. برای مدیریت داده ها از یک DAO (آبجکت دسترسی به داده ها) استفاده خواهید نمود. DAO وظیفه ی مدیریت اتصال به دیتابیس، دسترسی و ویرایش داده ها را بر عهده دارد. طی این مکانیزم همچنین آبجکت های دیتابیس به آبجکت های متعارف جاوا (Java Objects) تبدیل می شوند و از این طریق کد UI دیگر با لایه ی ذخیره ی دائمی داده ها یا persistence layer تعامل نخواهد داشت.

در پایان اپلیکیشن به صورت زیر خواهد بود:



گفتنی است که استفاده از الگو DAO همیشه و در همه ی شرایط بهترین روش نیست. در واقع این الگو، model object های جاوا (آبجکت هایی از روی جداول دیتابیس) ایجاد می کند و منابع بیشتری مصرف می کند. استفاده ی مستقیم از یک دیتابیس یا دسترسی به داده ها از طریق یک ContentProvider معمولا بهینه تر بوده و از آنجایی که در این دو روش دیگر نیازی به ساخت آبجکت های جاوا نیست، میزان حافظه مورد استفاده کاهش می یابد.

همان طور که گفته شد در این روش از DAO استفاده می شود. برای این مثال ورژن کتابخانه های اندروید را بر روی API 15 تنظیم نمایید که معادل اندروید 4.0 می باشد. در غیر این صورت می بایست از کلاس Loader برای مدیریت Cursor استفاده می شد که از اندروید 3.0 به بعد پشتیبانی می شود. بعلاوه استفاده از این کلاس پیچیدگی خاص خود را دارد.

ایجاد پروژه

یک پروژه ی جدید اندروید و یک activity به ترتیب به نام های de.vogella.android.sqlite.first و TestDatabaseActivity ایجاد نمایید.

2-3-4-ساخت Database و Data Model

ابتدا کلاس MySQLiteHelper را ایجاد نمایید. این کلاس دیتابیس اپلیکیشن را ایجاد می کند.

اگر به خاطر داشته باشید، متد onUpgrade() به کلی داده های فعلی بانک اطلاعاتی را حذف کرده و جدول را مجددا ایجاد می کند. سپس تعدادی ثابت (constant) جهت نگهداری مقادیر اسم جدول و ستون های جدول ایجاد می نماید.

```
package de.vogella.android.sqlite.first;
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;
public class MySQLiteHelper extends SQLiteOpenHelper {
      public static final String TABLE_COMMENTS = "comments";
      public static final String COLUMN_ID = "_id";
      public static final String COLUMN_COMMENT = "comment";
      private static final String DATABASE_NAME = "commments.db";
      private static final int DATABASE_VERSION = 1;
      // Database creation sql statement
      private static final String DATABASE_CREATE = "create table "
                  + TABLE_COMMENTS + "( " + COLUMN_ID
                  + " integer primary key autoincrement, " + COLUMN_COMMENT
                  + " text not null);";
      public MySQLiteHelper(Context context) {
            super(context, DATABASE_NAME, null, DATABASE_VERSION);
      }
      @Override
      public void onCreate(SQLiteDatabase database) {
            database.execSQL(DATABASE_CREATE);
      }
      @Override
```

```
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        Log.w(MySQLiteHelper.class.getName(),
                "Upgrading database from version " + oldVersion + " to "
                        + newVersion + ", which will destroy all old data");
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_COMMENTS);
        onCreate(db);
    }
}
```

کلاس Comment را ایجاد نمایید. این کلاس به منزله ی model (آبجکتی ساخته شده از جدول دیتابیس) خواهد بود و تمامی داده هایی که در دیتابیس ذخیره شده و برای کاربر به نمایش در می آید را دربرخواهد گرفت.

```
package de.vogella.android.sqlite.first;
public class Comment {
    private long id;
    private String comment;
    public long getId() {
        return id;
    }
    public void setId(long id) {
        this.id = id;
    }
    public String getComment() {
        return comment;
    }
    public void setComment(String comment) {
        this.comment = comment;
    }
    // Will be used by the ArrayAdapter in the ListView
    @Override
    public String toString() {
        return comment;
    }
}
```

کلاس CommentDataSource را ایجاد نمایید. این کلاس همان آبجکتی است که برای دسترسی به دیتابیس (DAO) از آن استفاده خواهید نمود. آبجکت ذکر شده اتصال به دیتابیس را نگه داشته و امکان درج comment های جدید و واکشی تمامی comment های جاری را برای اپلیکیشن فراهم می کند.

```
package de.vogella.android.sqlite.first;
import java.util.ArrayList;
import java.util.List;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
```

```
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
public class CommentsDataSource {
	// Database fields
	private SQLiteDatabase database;
	private MySQLiteHelper dbHelper;
	private String[] allColumns = { MySQLiteHelper.COLUMN_ID,
			MySQLiteHelper.COLUMN_COMMENT };
	public CommentsDataSource(Context context) {
		dbHelper = new MySQLiteHelper(context);
	}
	public void open() throws SQLException {
		database = dbHelper.getWritableDatabase();
	}
	public void close() {
		dbHelper.close();
	}
	public Comment createComment(String comment) {
		ContentValues values = new ContentValues();
		values.put(MySQLiteHelper.COLUMN_COMMENT, comment);
		long insertId = database.insert(MySQLiteHelper.TABLE_COMMENTS, null,
				values);
		Cursor cursor = database.query(MySQLiteHelper.TABLE_COMMENTS,
				allColumns, MySQLiteHelper.COLUMN_ID + " = " + insertId, null,
				null, null, null);
		cursor.moveToFirst();
		Comment newComment = cursorToComment(cursor);
		cursor.close();
		return newComment;
	}
	public void deleteComment(Comment comment) {
		long id = comment.getId();
		System.out.println("Comment deleted with id: " + id);
		database.delete(MySQLiteHelper.TABLE_COMMENTS, MySQLiteHelper.COLUMN_ID
				+ " = " + id, null);
	}
	public List<Comment> getAllComments() {
		List<Comment> comments = new ArrayList<Comment>();
		Cursor cursor = database.query(MySQLiteHelper.TABLE_COMMENTS,
				allColumns, null, null, null, null, null);
		cursor.moveToFirst();
		while (!cursor.isAfterLast()) {
			Comment comment = cursorToComment(cursor);
			comments.add(comment);
			cursor.moveToNext();
		}
		// make sure to close the cursor
		cursor.close();
		return comments;
	}

	private Comment cursorToComment(Cursor cursor) {
```

```
        Comment comment = new Comment();
        comment.setId(cursor.getLong(0));
        comment.setComment(cursor.getString(1));
        return comment;
    }
}
```

3-3-4-ساخت ظاهر و UI اپلیکیشن

فایل main.xml در folder <filename class="directory">res/layout_ را به صورت زیر ویرایش نمایید. این layout دو دکمه برای افزودن/حذف comment ها و یک ListView (لیست ساده) جهت نمایش comment های جاری تعریف می کند. متن comment بعدها در کلاس activity و به وسیله ی یکrandom generator تولید خواهد شد.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
  <LinearLayout
    android:id="@+id/group"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" >
    <Button
      android:id="@+id/add"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Add New"
      android:onClick="onClick"/>
    <Button
      android:id="@+id/delete"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Delete First"
      android:onClick="onClick"/>
  </LinearLayout>
  <ListView
    android:id="@android:id/list"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
  />
</LinearLayout>
```

کلاس TestDatabaseActivity را به صورت زیر ویرایش نمایید. برای نمایش آسان لیست ساده ای از اطلاعات می توانید از یک ListActivity استفاده نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
package de.vogella.android.sqlite.first;
import java.util.List;
import java.util.Random;
import android.app.ListActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
public class TestDatabaseActivity extends ListActivity {
      private CommentsDataSource datasource;
      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main);
            datasource = new CommentsDataSource(this);
            datasource.open();
            List<Comment> values = datasource.getAllComments();
            // use the SimpleCursorAdapter to show the
            // elements in a ListView
            ArrayAdapter<Comment> adapter = new ArrayAdapter<Comment>(this,
android.R.layout.simple_list_item_1, values);
            setListAdapter(adapter);
      }
      // Will be called via the onClick attribute
      // of the buttons in main.xml
      public void onClick(View view) {
            @SuppressWarnings("unchecked")
            ArrayAdapter<Comment> adapter = (ArrayAdapter<Comment>) getListAdapter();
            Comment comment = null;
            switch (view.getId()) {
            case R.id.add:
                  String[] comments = new String[] { "Cool", "Very nice", "Hate it" };
                  int nextInt = new Random().nextInt(3);
                  // save the new comment to the database
                  comment = datasource.createComment(comments[nextInt]);
                  adapter.add(comment);
                  break;
            case R.id.delete:
                  if (getListAdapter().getCount() > 0) {
                        comment = (Comment) getListAdapter().getItem(0);
                        datasource.deleteComment(comment);
                        adapter.remove(comment);
                  }
                  break;
            }
            adapter.notifyDataSetChanged();
      }
      @Override
      protected void onResume() {
            datasource.open();
            super.onResume();
      }
      @Override
```

```
    protected void onPause() {
        datasource.close();
        super.onPause();
    }
}
```

4-3-4-نصب و اجرای اپلیکیشن

اپلیکیشن خود را نصب نموده، دکمه های Add و Delete را کلیک نمایید. سپس اپلیکیشن را restart کرده و اطمینان حاصل نمایید که داده ها پس از راه اندازی مجدد همچنان در جای خود هستند.

4-4-شرح مفهوم Content Provider

4-4-1-Content Provider چیست و چه کاربردی دارد؟

جهت به اشتراک گذاری داده ها بین چندین اپلیکیشن، می توانید از Content Provider استفاده نمایید. می توانید به Provider به چشم یک منبع داده ای نگاه کنید که اطلاعات موجود در آن به اپلیکیشنی که از آن استفاده می کند، وابسته نمی باشد. این امکان وجود دارد که از Provider مانند یک دیتابیس کوئری بگیرید، داده های آن را بروز رسانی و حتی حذف نمایید.علاوه بر آن provider ها قابلیتی دارند که به آن ها اجازه ی ذخیره داده ها در فایل، دیتابیس و شبکه را می دهد.

Provider اطلاعات را بر اساس URI مربوطه کپسوله سازی می کند. هر URI ای که با content:// شروع می شود در واقع به منابع و محتوایی اشاره دارد که از طریق Provider قابل دسترسی است.

آدرس URI یک منبع به شما این امکان را می دهد تا از طریق Provider عملیات ساده ی CRUD (ایجاد، بازیابی، بروز رسانی و حذف) را بر روی داده های منبع مورد نظر اجرا نمایید.

همان طور که قبلا اشاره شد، provider داده های مورد نیاز اپلیکیشن را در اختیار آن قرار می دهد. این داده ها ممکن است در یک دیتابیس SQLite، در سیستم فایل، داخل فایل های flat یا بر روی سرور راه دور (remote server) مستقر باشند.

اگرچه از Provider می توان برای دسترسی به داده ها در یک اپلیکیشن نیز استفاده کرد، اما معمولا توسعه دهندگان از آن برای به اشتراک گذاری داده ها بین چندین اپلیکیشن بهره می گیرند. داده های اپلیکیشن به صورت پیش فرض دارای سطح دسترسی private می باشند. اینجا است که content provider به یاری شما می آید؛ این کامپوننت نرم افزاری دسترسی به داده های سایر اپلیکیشن ها را بر اساس یک interface ساخت یافته به راحتی مهیا می سازد.

Content provider را می بایست داخل فایل تنظیمات اپلیکیشن اندرویدی خود (manifest) تعریف نمایید.

2-4-4-قالب پایه ای و الگوی تعریف URI جهت دسترسی به Content Provider

برای دسترسی به content provider می بایست ابتدا پیشوند متعارف content:// و سپس namespace مربوط به provider را ارائه نمایید (namespace همان نام content provider حامل داده های مورد نظر می باشد). لازم است namespace را داخل فایل manifest، تگ provider و در attribute (خصیصه) android:authorities تعریف و مقداردهی نمایید. مثال: content://test/.

در زیر بخش ها مختلف یک URI را شرح می دهیم.

قالب و ساختار کلی URI به این صورت می باشد:

<standard_prefix>://<authority>/<data_path>/<id>

1. standard prefix یا پیشوندی که همیشه ثابت است: content://.
2. authority یا نام content provider.
3. data path نوع درخواست را مشخص می کند.
4. id رکورد مورد نظر را مشخص می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

URI پایه امکان دسترسی به مجموعه ای از منابع (برای مثال کل یک جدول) را فراهم می کند. چنانچه در بخش نهایی URI شناسه ی نمونه ذکر شود، در آن صورت تنها content ای که id آن ذکر شده، قابل دسترسی می باشد. مثال: content://test/2.

4-4-3-دسترسی به content provider

از آنجایی که داشتن URI های یک provider برای دسترسی به آن ضروی است، توصیه می شود URI ها را داخل constant هایی با سطح دسترسی public قرار داده، آن ها را به صورت مستند در اختیار دیگر توسعه دهندگان قرار دهید.

بسیاری از منابع داده ای اندروید همچون contacts (اطلاعات مخاطبین) از طریق content provider قابل دسترسی هستند.

4-4-4-پیاده سازی content provider اختصاصی

به منظور ایجاد content provider اختصاصی خود بایستی یک کلاس که از android.content.ContentProvider ارث می برد (extend می کند)، ایجاد نمایید. سپس این کلاس را به عنوان content provider در فایل تنظیمات (Android manifest) معرفی کنید. تگ مربوطه در فایل تنظیمات باید خصیصه android:authorities (attribute) را داشته باشد. مقدار این attribute را برابر اسم content provider قرار می دهید. Authority در واقع مبنای URI را تشکیل داده و برای دسترسی به content provider مورد استفاده قرار می گیرد، از اینرو باید منحصر بفرد باشد.

```
<provider   android:authorities="de.vogella.android.todos.contentprovider"
    android:name=".contentprovider.MyTodoContentProvider" >
</provider>
```

در بدنه ی Content provider شما باید تعداد زیادی متد را پیاده سازی کنید. برای مثال می توان به query()، insert()، update()، delete()، getType() و onCreate() اشاره کرد. در صورتی که از متد خاصی

پشتیبانی نمی کنید، توصیه می شود UnsupportedOperationException() را فراخوانی نمایید.

متد query() در خروجی یک آبجکت Cursor برمی گرداند.

5-4-4-Content provider و مبحث امنیت

تا ویرایش 4.2 اندروید، content provider به صورت پیش فرض برای دیگر اپلیکیشن های اندرویدی قابل دسترسی می باشد. اما از ورژن 4.2 به بعد اندروید، برای دسترسی به content provider لازم است آن را به صورت صریح export نمایید.

به منظور تنظیم قابلیت رویت و دسترسی content provider خود، کافی است پارامتر android:exported=false|true را در تعریف content provider خود داخل فایل AndroidManifest.xml ذکر نمایید.

نکته: توصیه می شود پارامتر android:exported را همیشه در فایل تنظیمات قید کنید تا رفتار اپلیکیشن در ورژن های مختلف اندروید یکسان باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

6-4-4-Thread safety (برطرف سازی مشکل همزمانی با استفاده از کلیدواژه ی synchronized)

چنانچه مستقیما با دیتابیس تعامل داشته باشید و همزمان از چند thread در دیتابیس مقادیری نوشته شوند (از چند thread همزمان چندین writer وجود داشته باشد) در آن صورت طبیعتا با مشکل مدیریت همروندی و concurrency مواجه می شوید.

یک content provider می تواند همزمان از چندین برنامه مورد دسترسی قرار گیرد. به این دلیل لازم است دسترسی thread-safe را پیاده سازی نمایید. آسان ترین راه، استفاده از کلیدواژه ی synchronized در مقابل تمامی متدهای provider است. بدین وسیله تنها یک thread در لحظه می تواند به این متدها دسترسی داشته باشد.

(thread-safe = دسترسی چند thread به یک data structure بدون اینکه در پروسه ی کاری thread ها اختلالی به وجود آید).

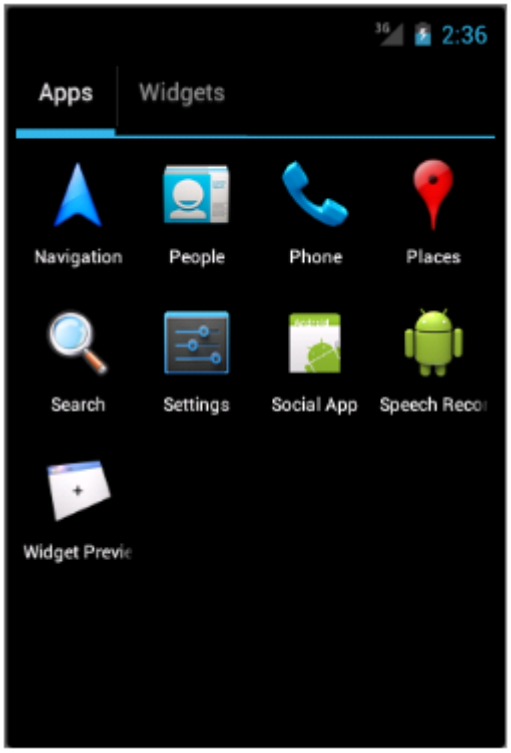
اگر لزومی ندارد که اندروید دسترسی به provider را sync کند، مقدار خصیصه (attribute) android:multiprocess= را در تگ <provider>، داخل فایل تنظیمات اپلیکیشن خود برابر true قرار دهید. این کار سبب می شود در هر یک از process های مربوط به کلاینت (پاسخ دهی به درخواست های کاربر)، یک نمونه از provider ایجاد شده و نیاز به اجرای IPC به کلی از میان برداشته شود (IPC = عبارت است از مجموعه ای از روش ها برای تبادل داده بین چندین پردازش خرد و کوچک/thread. بنابراین هر یک از این روشها می تواند برای ارتباط بین یک یا چند پردازش بکار رود).

آموزش: استفاده ی کاربردی از ContentProvider

مثال زیر از content provider آماده ی اپلیکیشن People استفاده می کند.

6-4-4-ایجاد contact ها (مخاطبین) در محیط شبیه ساز

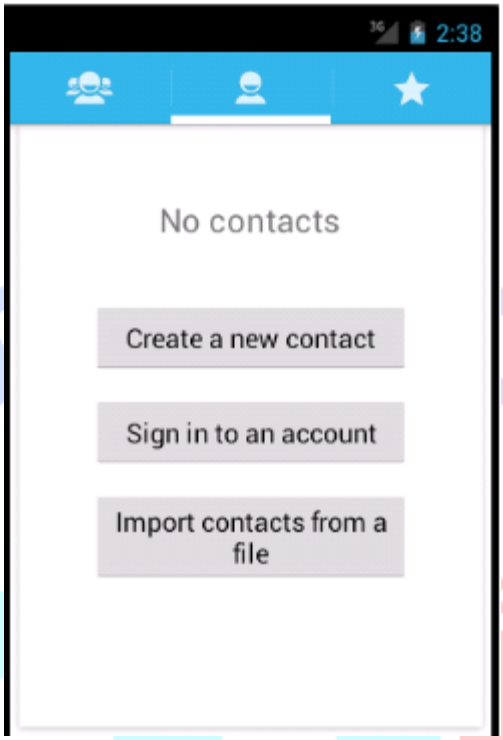
برای این مثال لازم است تعدادی contact ثابت داشته باشید. منوی home را انتخاب نموده و سپس المان People را برای ایجاد contacts انتخاب کنید.

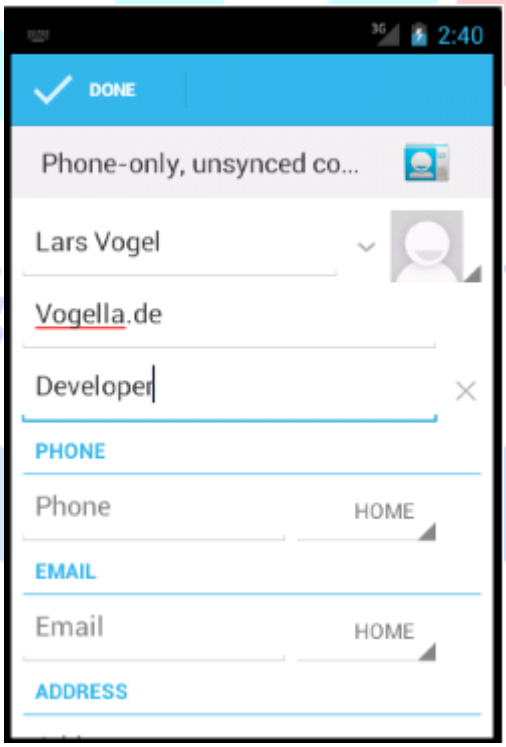


اپلیکیشن از شما می پرسد آیا میلی به ورود و login در برنامه دارید یا خیر. می توانید login کنید یا "Not now" را انتخاب نمایید. حال بر روی دکمه ی "Create a new contact" کلیک نمایید. می توانید local contatcs را ایجاد کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330





پس از اینکه اولین contact را ایجاد کردید، اپلیکیشن به شما اجازه می دهد تا با کلیک بر روی دکمه ی "+" contact های بیشتری ایجاد کنید. در پایان چندین contact به اپلیکیشن خود اضافه کرده اید.

7-4-4-استفاده از Contact Content Provider

یک پروژه و activity جدید اندروید به ترتیب به نام های de.vogella.android.contentprovider و ContactsActivity ایجاد نمایید.

فایل layout مربوطه را نیز در آدرس <filename class="directory">res/layout_ folder ویرایش نمایید. ID المان TextView را به contactview تغییر داده و متن پیش فرض را حذف نمایید.

فایل layout خروجی می بایست دارای ظاهری مشابه زیر باشد.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <TextView
      android:id="@+id/contactview"
      android:layout_width="match_parent"
      android:layout_height="match_parent" />
</LinearLayout>
```

از آنجایی که تمامی اپلیکیشن ها نباید به اطلاعات شخصی مخاطبین دسترسی داشته باشند، لازم است برای دستیابی به contact ContentProvider، در فایل تنظیمات permission خاصی تنظیم نمایید. برای این منظور ابتدا فایل AndroidManifest.xml را باز نموده و تب Permissions را انتخاب کنید. در این تب، بر روی دکمه ی Add کلیک کرده و Uses Permission را انتخاب نمایید. از لیست کشویی (drop-down) حاضر، آیتم android.permission.READ_CONTACTS را انتخاب کنید.

پیاده سازی activity را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.contentprovider;
import android.app.Activity;
import android.database.Cursor;
```

```
import android.net.Uri;
import android.os.Bundle;
import android.provider.ContactsContract;
import android.widget.TextView;

public class ContactsActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_contacts);
        TextView contactView = (TextView) findViewById(R.id.contactview);
        Cursor cursor = getContacts();
        while (cursor.moveToNext()) {
            String displayName = cursor.getString(cursor
                    .getColumnIndex(ContactsContract.Data.DISPLAY_NAME));
            contactView.append("Name: ");
            contactView.append(displayName);
            contactView.append("\n");
        }
    }
    private Cursor getContacts() {
        // Run query
        Uri uri = ContactsContract.Contacts.CONTENT_URI;
        String[] projection = new String[] { ContactsContract.Contacts._ID,
                ContactsContract.Contacts.DISPLAY_NAME };
        String selection = ContactsContract.Contacts.IN_VISIBLE_GROUP + " = '"
                + ("1") + "'";
        String[] selectionArgs = null;
        String sortOrder = ContactsContract.Contacts.DISPLAY_NAME
                + " COLLATE LOCALIZED ASC";
        return managedQuery(uri, projection, selection, selectionArgs,
                sortOrder);
    }
}
```

زمانی که اپلیکیشن را اجرا کنید، داده های لازم از کامپوننت نرم افزاری ContentProvider که اطلاعات اپلیکیشن People را حمل می کند، بارگذاری شده و نهایتا در المان رابط کاربری TextView به نمایش گذاشته می شود. جهت ارائه ی چنین اطلاعاتی به کاربر، توسعه دهندگان معمولا از لیست ساده که توسط ListView پیاده سازی می شود، استفاده می کنند.

Unresolved directive in 001_article.adoc - include::../AndroidBackgroundProcessing/content/120_loader.adoc[]

4-5-Cursor ها و Loader ها

یکی از مشکلات و چالش های دسترسی به دیتابیس و خواندن اطلاعات، پایین بودن سرعت آن است. بعلاوه، اپلیکیشن می بایست life cycle کامپوننت ها را در نظر گرفته و آن را مدیریت کند. برای مثال می توان به باز کردن و بستن cursor در صورت تغییر در تنظیمات و نحوه ی پیکربندی اشاره کرد.

تا ویرایش 3.0 سیستم عامل اندروید، توسعه دهندگان با فراخوانی متد managedQuery() در کلاس های activity، چرخه ی حیات یا lifecycle کامپوننت ها را اداره می کردند.

اما از ویرایش 3.0 به بعد اندروید، استفاده از این متد دیگر توصیه نمی شود و توسعه دهنده برای دسترسی به ContentProvider می بایست از چارچوب نرم افزاری Loader (framework) استفاده کند.

کلاس SimpleCursorAdapter که همراه با ListViews نیز قابل استفاده می باشد، متد swapCursor() را در اختیار شما قرار می دهد. کلاس Loader اپلیکیشن شما می تواند با فراخوانی این متد، Cursor را در متد onLoadFinished() بروز رسانی کند.

کلاس CursorLoader، آبجکت Cursor را پس از تغییر در تنظیمات و نحوه ی پیکربندی مجددا متصل می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آموزش: SQLite، ContentProvider و Loader با پیاده سازی اختصاصی

مرور کلی

نمونه اپلیکیشنی که در زیر مشاهده می کنید، در Android Market نیز به صورت آماده در دسترس است. برای اینکه کاربران بیشتری بتوانند از آن استفاده کنند، اپلیکیشن به ویرایش 2.3 سیستم عامل اندروید downport شده و بر روی آن ها قابل اجرا می باشد. در صورتی که اپلیکیشن بارکد خوان را بر روی دستگاه خود نصب کرده باشید، می توانید با اسکن کد QR زیر به سرعت اپلیکیشن مورد نظر را در Android Market پیدا کنید. لازم به توضیح است که اپلیکیشن نام برده در ورژن های مختلف اندروید ممکن است ظاهر و رفتار متفاوتی داشته باشد. برای مثال ممکن است بجای OptionMenu در اپلیکیشن ActionBar را داشته باشید و پوسته (theme) برنامه کمی متفاوت باشد.

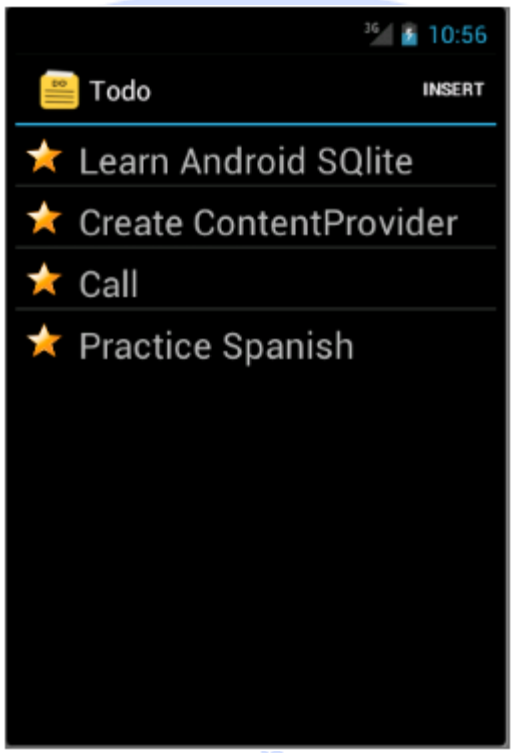
در بخش آموزشی حاضر یک اپلیکیشن به نام "To-do" ایجاد خواهید نمود. این برنامه به کاربر اجازه می دهد تا task یا کارهایی که باید انجام شوند را در لیستی وارد نماید. آیتم های ورودی در دیتابیس SQLite ذخیره شده و از طریق ContentProvider قابل دسترسی می باشد.

در نمونه اپلیکیشن جاری، کارها یا task هایی که داخل لیست درج می شوند، "todos" نام دارند.

اپلیکیشن مورد نظر دارای دو activity می باشد که یکی برای مشاهده ی آیتم های todo و دیگری ویژه ی ایجاد و ویرایش آیتم مورد نظر تعبیه شده است. دو activity این اپلیکیشن از طریق آبجکت های Intent با یکدیگر تعامل کرده و اطلاعات مورد نیاز را رد و بدل می نمایند.

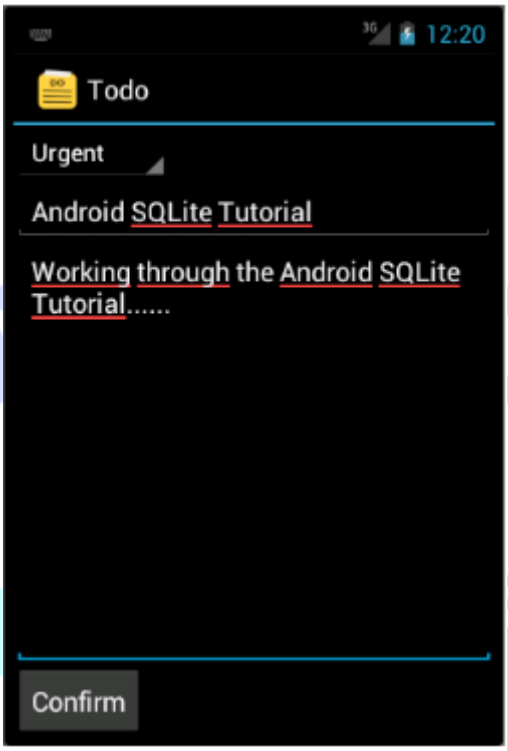
به منظور بارگذاری و مدیریت Cursor به طور ناهمزمان (asynch)، activity اصلی اپلیکیشن از کلاس Loader بهره می گیرد.

در پایان اپلیکیشن دارای ظاهری مشابه زیر خواهد بود.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



ایجاد پروژه

یک پروژه و activity جدید به ترتیب به نام های de.vogella.android.todos و TodosOverviewActivity ایجاد نمایید. حال activity دیگری به نام TodoDetailActivity در پروژه ی جاری ایجاد نمایید.

1-5-4-تعریف کلاس های مدیریت Database

پکیج de.vogella.android.todos.database را ایجاد نمایید. این پکیج کلاس هایی که عملیات مربوط به دیتابیس را اداره می کنند، در خود نگه می دارد.

همان طور که قبلا گفته شد، بهتر است ویژه ی هر جدول در دیتابیس، یک کلاس جداگانه تعریف نمایید. هر چند نمونه اپلیکیشن حاضر یک جدول بیشتر ندارد، اما توصیه می شود از این روش

پیروی کنید تا در صورت گسترش schema دیتابیس (پوشه ای که جداول دیتابیس در آن قرار می گیرند) تمهیدات لازم را از قبل اندیشیده و برای آن آماده باشید.

کلاس زیر را ایجاد نمایید. این کلاس مقادیر مربوط به اسم جدول و ستون های آن را در قالب چند constant نگه می دارد.

```
package de.vogella.android.todos.database;
import android.database.sqlite.SQLiteDatabase;
import android.util.Log;
public class TodoTable {
      // Database table
      public static final String TABLE_TODO = "todo";
      public static final String COLUMN_ID = "_id";
      public static final String COLUMN_CATEGORY = "category";
      public static final String COLUMN_SUMMARY = "summary";
      public static final String COLUMN_DESCRIPTION = "description";
      // Database creation SQL statement
      private static final String DATABASE_CREATE = "create table "
                  + TABLE_TODO
                  + "("
                  + COLUMN_ID + " integer primary key autoincrement, "
                  + COLUMN_CATEGORY + " text not null, "
                  + COLUMN_SUMMARY + " text not null,"
                  + COLUMN_DESCRIPTION
                  + " text not null"
                  + ");";
      public static void onCreate(SQLiteDatabase database) {
            database.execSQL(DATABASE_CREATE);
      }
      public static void onUpgrade(SQLiteDatabase database, int oldVersion,
                  int newVersion) {
            Log.w(TodoTable.class.getName(), "Upgrading database from version "
                        + oldVersion + " to " + newVersion
                        + ", which will destroy all old data");
            database.execSQL("DROP TABLE IF EXISTS " + TABLE_TODO);
            onCreate(database);
      }
}
```

کلاس TodoDatabaseHelper زیر را ایجاد نمایید. این کلاس از SQLiteOpenHelper ارث برده و متدهای static کلاس کمکی (helper) TodoTable فراخوانی می کند.

```
package de.vogella.android.todos.database;
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
public class TodoDatabaseHelper extends SQLiteOpenHelper {
      private static final String DATABASE_NAME = "todotable.db";
```

```
    private static final int DATABASE_VERSION = 1;
    public TodoDatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }
    // Method is called during creation of the database
    @Override
    public void onCreate(SQLiteDatabase database) {
        TodoTable.onCreate(database);
    }
    // Method is called during an upgrade of the database,
    // e.g. if you increase the database version
    @Override
    public void onUpgrade(SQLiteDatabase database, int oldVersion,
            int newVersion) {
        TodoTable.onUpgrade(database, oldVersion, newVersion);
    }
}
```

برای دسترسی به دیتابیس این بار مانند مثال قبلی از یک DAO (آبجکت دسترسی به داده) استفاده نخواهید نمود، بلکه یک ContentProvider جهت دسترسی به داده های مورد نیاز پیاده سازی خواهید کرد.

2-5-4-ایجاد ContentProvider

ابتدا پکیج de.vogella.android.todos.contentprovider را ایجاد نمایید. سپس کلاس MyTodoContentProvider را ایجاد کنید که از ContentProvider ارث بری (extend) می کند.

```
package de.vogella.android.todos.contentprovider;
import java.util.Arrays;
import java.util.HashSet;
import android.content.ContentProvider;
import android.content.ContentResolver;
import android.content.ContentValues;
import android.content.UriMatcher;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteQueryBuilder;
import android.net.Uri;
import android.text.TextUtils;
import de.vogella.android.todos.database.TodoDatabaseHelper;
import de.vogella.android.todos.database.TodoTable;
public class MyTodoContentProvider extends ContentProvider {
    // database
    private TodoDatabaseHelper database;
    // used for the UriMacher
    private static final int TODOS = 10;
    private static final int TODO_ID = 20;
    private static final String AUTHORITY = "de.vogella.android.todos.contentprovider";
```

```
private static final String BASE_PATH = "todos";
public static final Uri CONTENT_URI = Uri.parse("content://" + AUTHORITY
        + "/" + BASE_PATH);
public static final String CONTENT_TYPE = ContentResolver.CURSOR_DIR_BASE_TYPE
        + "/todos";
public static final String CONTENT_ITEM_TYPE = ContentResolver.CURSOR_ITEM_BASE_TYPE
        + "/todo";
private static final UriMatcher sURIMatcher = new UriMatcher(
        UriMatcher.NO_MATCH);
static {
    sURIMatcher.addURI(AUTHORITY, BASE_PATH, TODOS);
    sURIMatcher.addURI(AUTHORITY, BASE_PATH + "/#", TODO_ID);
}
@Override
public boolean onCreate() {
    database = new TodoDatabaseHelper(getContext());
    return false;
}
@Override
public Cursor query(Uri uri, String[] projection, String selection,
        String[] selectionArgs, String sortOrder) {
    // Uisng SQLiteQueryBuilder instead of query() method
    SQLiteQueryBuilder queryBuilder = new SQLiteQueryBuilder();
    // check if the caller has requested a column which does not exists
    checkColumns(projection);
    // Set the table
    queryBuilder.setTables(TodoTable.TABLE_TODO);
    int uriType = sURIMatcher.match(uri);
    switch (uriType) {
    case TODOS:
        break;
    case TODO_ID:
        // adding the ID to the original query
        queryBuilder.appendWhere(TodoTable.COLUMN_ID + "="
                + uri.getLastPathSegment());
        break;
    default:
        throw new IllegalArgumentException("Unknown URI: " + uri);
    }
    SQLiteDatabase db = database.getWritableDatabase();
    Cursor cursor = queryBuilder.query(db, projection, selection,
            selectionArgs, null, null, sortOrder);
    // make sure that potential listeners are getting notified
    cursor.setNotificationUri(getContext().getContentResolver(), uri);
    return cursor;
}
@Override
public String getType(Uri uri) {
    return null;
}
@Override
public Uri insert(Uri uri, ContentValues values) {
```

```
        int uriType = sURIMatcher.match(uri);
        SQLiteDatabase sqlDB = database.getWritableDatabase();
        long id = 0;
        switch (uriType) {
        case TODOS:
            id = sqlDB.insert(TodoTable.TABLE_TODO, null, values);
            break;
        default:
            throw new IllegalArgumentException("Unknown URI: " + uri);
        }
        getContext().getContentResolver().notifyChange(uri, null);
        return Uri.parse(BASE_PATH + "/" + id);
    }
    @Override
    public int delete(Uri uri, String selection, String[] selectionArgs) {
        int uriType = sURIMatcher.match(uri);
        SQLiteDatabase sqlDB = database.getWritableDatabase();
        int rowsDeleted = 0;
        switch (uriType) {
        case TODOS:
            rowsDeleted = sqlDB.delete(TodoTable.TABLE_TODO, selection,
                    selectionArgs);
            break;
        case TODO_ID:
            String id = uri.getLastPathSegment();
            if (TextUtils.isEmpty(selection)) {
                rowsDeleted = sqlDB.delete(
                        TodoTable.TABLE_TODO,
                        TodoTable.COLUMN_ID + "=" + id,
                        null);
            } else {
                rowsDeleted = sqlDB.delete(
                        TodoTable.TABLE_TODO,
                        TodoTable.COLUMN_ID + "=" + id
                        + " and " + selection,
                        selectionArgs);
            }
            break;
        default:
            throw new IllegalArgumentException("Unknown URI: " + uri);
        }
        getContext().getContentResolver().notifyChange(uri, null);
        return rowsDeleted;
    }
    @Override
    public int update(Uri uri, ContentValues values, String selection,
            String[] selectionArgs) {
        int uriType = sURIMatcher.match(uri);
        SQLiteDatabase sqlDB = database.getWritableDatabase();
        int rowsUpdated = 0;
        switch (uriType) {
        case TODOS:
```

```
            rowsUpdated = sqlDB.update(TodoTable.TABLE_TODO,
                        values,
                        selection,
                        selectionArgs);
            break;
        case TODO_ID:
            String id = uri.getLastPathSegment();
            if (TextUtils.isEmpty(selection)) {
                rowsUpdated = sqlDB.update(TodoTable.TABLE_TODO,
                            values,
                            TodoTable.COLUMN_ID + "=" + id,
                            null);
            } else {
                rowsUpdated = sqlDB.update(TodoTable.TABLE_TODO,
                            values,
                            TodoTable.COLUMN_ID + "=" + id
                            + " and "
                            + selection,
                            selectionArgs);
            }
            break;
        default:
            throw new IllegalArgumentException("Unknown URI: " + uri);
        }
        getContext().getContentResolver().notifyChange(uri, null);
        return rowsUpdated;
    }
    private void checkColumns(String[] projection) {
        String[] available = { TodoTable.COLUMN_CATEGORY,
                TodoTable.COLUMN_SUMMARY, TodoTable.COLUMN_DESCRIPTION,
                TodoTable.COLUMN_ID };
        if (projection != null) {
            HashSet<String> requestedColumns = new HashSet<String>(
                    Arrays.asList(projection));
            HashSet<String> availableColumns = new HashSet<String>(
                    Arrays.asList(available));
            // check if all columns which are requested are available
            if (!availableColumns.containsAll(requestedColumns)) {
                throw new IllegalArgumentException(
                        "Unknown columns in projection");
            }
        }
    }
}
```

MyTodoContentProvider توابع update()، insert()، delete() و query() را پیاده سازی می کند. این متدها به طور مستقیم از interface یا الگوی پیاده سازی SQLiteDatabase مشتق می شوند.

این کلاس همچنین با فراخوانی متدی به نام checkColumns()، اطمینان حاصل می کند که کوئری فقط و فقط ستون های معتبر و مجاز را درخواست می کند.

ContentProvider اپلیکیشن خود را در فایل AndroidManifest.xml ثبت نمایید.

```
<application
      <!-- Place the following after the Activity
         Definition
      -->
  <provider
     android:name=".contentprovider.MyTodoContentProvider"
 android:authorities="de.vogella.android.todos.contentprovider" >
   </provider>
</application>
```

3-5-4-منابع (Resource ها)

اپلیکیشن جاری به منابع و محتوای متعددی احتیاج دارد. ابتدا یک منوی listmenu.xml در پوشه ی res/menu تعریف نمایید. اگر از ویزارد Android resource برای ایجاد فایل "listmenu.xml" استفاده نمایید، پوشه ی مورد نیاز به صورت خودکار برای شما ایجاد می شود. چنانچه فایل را خود به صورت دستی ایجاد کنید، در آن صورت پوشه را نیز باید خود بسازید.

این فایل XML برای تعریف ظاهر option menu در اپلیکیشن بکار می رود. با تنظیم مقدار android:showAsAction= بر روی"always"، آیتم مورد نظر در ActionBar اپلیکیشن به نمایش گذاشته می شود.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
   <item
      android:id="@+id/insert"
      android:showAsAction="always"
      android:title="Insert">
   </item>
</menu>
```

کاربر می تواند برای آیتم ها اولویت قرار دهد. برای priorities یک آرایه ی رشته ای ایجاد خواهید نمود.

ابتدا فایل priority.xml زیر را در پوشه ی res/values ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <string-array name="priorities">
      <item>Urgent</item>
      <item>Reminder</item>
   </string-array>
</resources>
```

لازم است برای اپلیکیشن خود ثابت های رشته ای متعددی تعریف کنید. فایل strings.xml تحت پوشه ی res/values را ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, Todo!</string>
    <string name="app_name">Todo</string>
    <string name="no_todos">Currently there are no Todo items maintained</string>
    <string name="menu_insert">Add Item</string>
    <string name="menu_delete">Delete Todo</string>
    <string name="todo_summary">Summary</string>
    <string name="todo_description">Delete Todo</string>
    <string name="todo_edit_summary">Summary</string>
    <string name="todo_edit_description">Description</string>
    <string name="todo_edit_confirm">Confirm</string>
</resources>
```

4-5-4-تعریف فایل های layout

سه فایل layout تعریف خواهید کرد. یکی از فایل های layout برای نمایش و تنظیم ظاهر سطر در لیست مورد استفاده قرار می گیرد و دو فایل دیگر توسط activity های اپلیکیشن فراخوانی می شوند.

Row layout به آیکونی به نام reminder اشاره دارد. آیکونی از نوع "png" به نام "reminder.png" در پوشه های res/drawable (drawable-hdpi, drawable-mdpi, drawable-ldpi) ایجاد نمایید.

اگر خود آیکون مناسبی در دست ندارید، می توانید آیکونی که ویزارد اندروید (ic_launcher.png in the res/drawable* folders) در اختیار شما قرار می دهد را بکار ببرید یا اسم (reference) اشاره گر

به آن را در فایل layout ویرایش نمایید. لازم به ذکر است که ADT ممکن است اسم آیکون تولید شده را تغییر دهد. از اینرو اگر اسم فایل شما "ic_launcher.png" نبود، تعجب نکنید.

در صورت تمایل می توانید تعریف آیکون را از فایل "todo_row.xml" به کلی حذف کنید. این فایل را در مرحله ی بعدی ایجاد خواهید نمود.

فایل "todo_row.xml" را در پوشه ی res/layout_ <filename class="directory"> ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="wrap_content" >
   <ImageView
      android:id="@+id/icon"
      android:layout_width="30dp"
      android:layout_height="24dp"
      android:layout_marginLeft="4dp"
      android:layout_marginRight="8dp"
      android:layout_marginTop="8dp"
      android:src="@drawable/reminder" >
   </ImageView>
   <TextView
      android:id="@+id/label"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="6dp"
      android:lines="1"
      android:text="@+id/TextView01"
      android:textSize="24dp"
      >
   </TextView>
</LinearLayout>
```

حال فایل todo_list.xml را ایجاد نمایید. این فایل ظاهر و چیدمان لیست را تعیین می کند.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <ListView
      android:id="@android:id/list"
      android:layout_width="match_parent"
      android:layout_height="wrap_content" >
   </ListView>
```

```
    <TextView
        android:id="@android:id/empty"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/no_todos" />
</LinearLayout>
```

فایل todo_edit.xml را ایجاد کنید. این فایل layout برای ویرایش و تنظیم ظاهر آیتم های فردی todo در TodoDetailActivity مورد استفاده قرار می گیرد.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <Spinner
        android:id="@+id/category"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:entries="@array/priorities" >
    </Spinner>
    <LinearLayout
        android:id="@+id/LinearLayout01"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
        <EditText
            android:id="@+id/todo_edit_summary"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:hint="@string/todo_edit_summary"
            android:imeOptions="actionNext" >
        </EditText>
    </LinearLayout>
    <EditText
        android:id="@+id/todo_edit_description"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:gravity="top"
        android:hint="@string/todo_edit_description"
        android:imeOptions="actionNext" >
    </EditText>
    <Button
        android:id="@+id/todo_edit_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/todo_edit_confirm" >
    </Button>
```

```
</LinearLayout>
```

5-5-4-تنظیم Activity ها

کد activity های خود را به صورت زیر ویرایش کنید. ویرایش را از فایل TodosOverviewActivity.java آغاز نمایید.

```
package de.vogella.android.todos;
import android.app.ListActivity;
import android.app.LoaderManager;
import android.content.CursorLoader;
import android.content.Intent;
import android.content.Loader;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.ContextMenu.ContextMenuInfo;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView.AdapterContextMenuInfo;
import android.widget.ListView;
import android.widget.SimpleCursorAdapter;
import de.vogella.android.todos.contentprovider.MyTodoContentProvider;
import de.vogella.android.todos.database.TodoTable;
/*
 * TodosOverviewActivity displays the existing todo items
 * in a list
 *
 * You can create new ones via the ActionBar entry "Insert"
 * You can delete existing ones via a long press on the item
 */
public class TodosOverviewActivity extends ListActivity implements
        LoaderManager.LoaderCallbacks<Cursor> {
    private static final int ACTIVITY_CREATE = 0;
    private static final int ACTIVITY_EDIT = 1;
    private static final int DELETE_ID = Menu.FIRST + 1;
    // private Cursor cursor;
    private SimpleCursorAdapter adapter;
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.todo_list);
        this.getListView().setDividerHeight(2);
        fillData();
```

```
        registerForContextMenu(getListView());
    }
    // create the menu based on the XML defintion
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.listmenu, menu);
        return true;
    }
    // Reaction to the menu selection
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case R.id.insert:
            createTodo();
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
    @Override
    public boolean onContextItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case DELETE_ID:
            AdapterContextMenuInfo info = (AdapterContextMenuInfo) item
                    .getMenuInfo();
            Uri uri = Uri.parse(MyTodoContentProvider.CONTENT_URI + "/"
                    + info.id);
            getContentResolver().delete(uri, null, null);
            fillData();
            return true;
        }
        return super.onContextItemSelected(item);
    }
    private void createTodo() {
        Intent i = new Intent(this, TodoDetailActivity.class);
        startActivity(i);
    }
    // Opens the second activity if an entry is clicked
    @Override
    protected void onListItemClick(ListView l, View v, int position, long id) {
        super.onListItemClick(l, v, position, id);
        Intent i = new Intent(this, TodoDetailActivity.class);
        Uri todoUri = Uri.parse(MyTodoContentProvider.CONTENT_URI + "/" + id);
        i.putExtra(MyTodoContentProvider.CONTENT_ITEM_TYPE, todoUri);
        startActivity(i);
    }
    private void fillData() {
        // Fields from the database (projection)
        // Must include the _id column for the adapter to work
        String[] from = new String[] { TodoTable.COLUMN_SUMMARY };
        // Fields on the UI to which we map
        int[] to = new int[] { R.id.label };
```

```
        getLoaderManager().initLoader(0, null, this);
        adapter = new SimpleCursorAdapter(this, R.layout.todo_row, null, from,
                to, 0);
        setListAdapter(adapter);
    }
    @Override
    public void onCreateContextMenu(ContextMenu menu, View v,
            ContextMenuInfo menuInfo) {
        super.onCreateContextMenu(menu, v, menuInfo);
        menu.add(0, DELETE_ID, 0, R.string.menu_delete);
    }

    // creates a new loader after the initLoader () call
    @Override
    public Loader<Cursor> onCreateLoader(int id, Bundle args) {
        String[] projection = { TodoTable.COLUMN_ID, TodoTable.COLUMN_SUMMARY };
        CursorLoader cursorLoader = new CursorLoader(this,
                MyTodoContentProvider.CONTENT_URI, projection, null, null, null);
        return cursorLoader;
    }
    @Override
    public void onLoadFinished(Loader<Cursor> loader, Cursor data) {
        adapter.swapCursor(data);
    }
    @Override
    public void onLoaderReset(Loader<Cursor> loader) {
        // data is not available anymore, delete reference
        adapter.swapCursor(null);
    }
}
```

سپس فایل TodoDetailActivity.java را به صورت زیر ویرایش کنید.

```
package de.vogella.android.todos;
import android.app.Activity;
import android.content.ContentValues;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.Toast;
import de.vogella.android.todos.contentprovider.MyTodoContentProvider;
import de.vogella.android.todos.database.TodoTable;
/*
 * TodoDetailActivity allows to enter a new todo item
 * or to change an existing
```

```
 */
public class TodoDetailActivity extends Activity {
	private Spinner mCategory;
	private EditText mTitleText;
	private EditText mBodyText;
	private Uri todoUri;
	@Override
	protected void onCreate(Bundle bundle) {
		super.onCreate(bundle);
		setContentView(R.layout.todo_edit);
		mCategory = (Spinner) findViewById(R.id.category);
		mTitleText = (EditText) findViewById(R.id.todo_edit_summary);
		mBodyText = (EditText) findViewById(R.id.todo_edit_description);
		Button confirmButton = (Button) findViewById(R.id.todo_edit_button);
		Bundle extras = getIntent().getExtras();
		// check from the saved Instance
		todoUri = (bundle == null) ? null : (Uri) bundle
				.getParcelable(MyTodoContentProvider.CONTENT_ITEM_TYPE);
		// Or passed from the other activity
		if (extras != null) {
			todoUri = extras
					.getParcelable(MyTodoContentProvider.CONTENT_ITEM_TYPE);
			fillData(todoUri);
		}
		confirmButton.setOnClickListener(new View.OnClickListener() {
			public void onClick(View view) {
				if (TextUtils.isEmpty(mTitleText.getText().toString())) {
					makeToast();
				} else {
					setResult(RESULT_OK);
					finish();
				}
			}
		});
	}
	private void fillData(Uri uri) {
		String[] projection = { TodoTable.COLUMN_SUMMARY,
				TodoTable.COLUMN_DESCRIPTION, TodoTable.COLUMN_CATEGORY };
		Cursor cursor = getContentResolver().query(uri, projection, null, null,
				null);
		if (cursor != null) {
			cursor.moveToFirst();
			String category = cursor.getString(cursor
					.getColumnIndexOrThrow(TodoTable.COLUMN_CATEGORY));
			for (int i = 0; i < mCategory.getCount(); i++) {
				String s = (String) mCategory.getItemAtPosition(i);
				if (s.equalsIgnoreCase(category)) {
					mCategory.setSelection(i);
				}
			}
			mTitleText.setText(cursor.getString(cursor
					.getColumnIndexOrThrow(TodoTable.COLUMN_SUMMARY)));
```

```
            mBodyText.setText(cursor.getString(cursor
                    .getColumnIndexOrThrow(TodoTable.COLUMN_DESCRIPTION)));
            // always close the cursor
            cursor.close();
        }
    }
    protected void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
        saveState();
        outState.putParcelable(MyTodoContentProvider.CONTENT_ITEM_TYPE, todoUri);
    }
    @Override
    protected void onPause() {
        super.onPause();
        saveState();
    }
    private void saveState() {
        String category = (String) mCategory.getSelectedItem();
        String summary = mTitleText.getText().toString();
        String description = mBodyText.getText().toString();
        // only save if either summary or description
        // is available
        if (description.length() == 0 && summary.length() == 0) {
            return;
        }
        ContentValues values = new ContentValues();
        values.put(TodoTable.COLUMN_CATEGORY, category);
        values.put(TodoTable.COLUMN_SUMMARY, summary);
        values.put(TodoTable.COLUMN_DESCRIPTION, description);
        if (todoUri == null) {
            // New todo
            todoUri = getContentResolver().insert(
                    MyTodoContentProvider.CONTENT_URI, values);
        } else {
            // Update todo
            getContentResolver().update(todoUri, values, null, null);
        }
    }
    private void makeToast() {
        Toast.makeText(TodoDetailActivity.this, "Please maintain a summary",
                Toast.LENGTH_LONG).show();
    }
}
```

محتوای فایل AndroidManifest.xml در نهایت به صورت زیر خواهد بود.

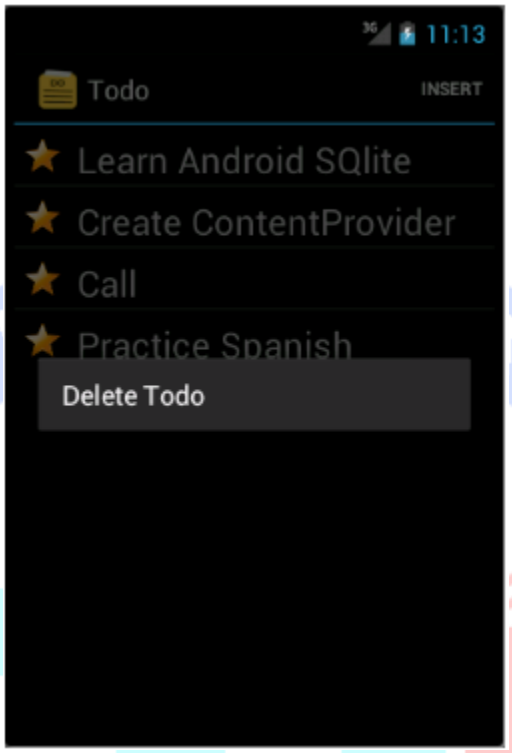
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="de.vogella.android.todos"
  android:versionCode="1"
  android:versionName="1.0" >
  <uses-sdk android:minSdkVersion="15" />
```

```
  <application
    android:icon="@drawable/icon"
    android:label="@string/app_name" >
    <activity
      android:name=".TodosOverviewActivity"
      android:label="@string/app_name" >
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
    <activity
      android:name=".TodoDetailActivity"        android:windowSoftInputMode="stateVisible|adjustResize"
>
    </activity>
    <provider        android:name=".contentprovider.MyTodoContentProvider"
android:authorities="de.vogella.android.todos.contentprovider" >
    </provider>
  </application>
</manifest>
```

تست اپلیکیشن

اپلیکیشن خود را راه اندازی کنید. با کلیک بر روی دکمه ی "Insert" در ActionBar طبیعتا باید بتوانید آیتم جدیدی را در لیست درج نمایید.

جهت حذف آیتم دلخواه از لیست کافی است بر روی آن طولانی مدت کلیک نمایید.



به منظور ویرایش آیتم جاری در لیست، بر روی سطر میزبان آن کلیک نمایید. با این کار activity دوم راه اندازی می شود.

6-5-4-محل ذخیره سازی دیتابیس SQLite

SQLite دیتابیس را همراه با تمامی اطلاعات آن داخل یک فایل ذخیره می کند. در صورتی که به این فایل دسترسی دارید، می توانید به طور مستقیم با دیتابیس کار کنید. دسترسی مسقیم به فایل مزبور تنها در محیط شبیه ساز و دستگاه های root شده امکان پذیر می باشد.

یک دستگاه استاندارد اندروید اجازه ی دسترسی و خواندن (read-access) با فایل دیتابیس را نمی دهد.

7-5-4-دسترسی به دیتابیس از راه دور به وسیله ی command line (Shell access)

در محیط شبیه ساز یا دستگاه root شده اندروید، می توان از طریق پنجره ی فرمان یا command line به دیتابیس SQLite دسترسی داشت. جهت اتصال به دستگاه کافی است دستور زیر را در پنجره ی فرمان اجرا کنید.

```
adb shell
```

Command adb در پوشه ی نصب Android SDK و زیرپوشه ی "platform-tools" مسقر می باشد.

سپس دستور "cd" را برای تغییر پوشه ی دیتابیس فراخوانی نموده و با اجرای دستور "sqlite3" به دیتابیس متصل شوید. برای مثال در اپلیکیشن جاری دستورات لازم به شرح زیر است:

```
# Switch to the data directory
cd /data/data
# Our application
cd de.vogella.android.todos
# Switch to the database dir
cd databases
# Check the content
ls
# Assuming that there is a todotable file
# connect to this table
sqlite3 todotable.db
```

پرکاربرد ترین دستورات SQLite به شرح زیر می باشند:

شرح	دستور
تمامی دستورات و آپشن ها را لیست می کند.	.help
از دستور sqlite3 خارج می شود.	.exit
دستورات CREATE که برای ایجاد جداول دیتابیس جاری فراخوانی شدند را لیست می کند.	.schema

بهینه سازی اپلیکیشن

تغییراتی که در SQLite اعمال می شوند همگی ACID هستند (Atomicity: تراکنش ها باید کامل اجرا شوند. Consistency: تمامی تراکنش ها باید دیتابیس را از یک حالت کامل و قابل قبول به یک حالت بی عیب و کامل دیگر منتقل کند. isolation: تراکنش هایی که موازی اجرا می شوند باید کاملا از هم مجزا بوده و در جامعیت دیتابیس اختلالی ایجاد نکند. Durability: کلیه ی تغییراتی که تراکنش ها بر روی دیتابیس اعمال می کنند همگی باید ذخیره شده و در صورت رخداد خطای ناگهانی آن ها را به حالت اول بازگرداند). در نتیجه تمامی عملیات update، insert و delete ای که بر روی پایگاه داده اجرا می شوند، از این چهار ویژگی برخوردار بوده و از آن ها تبعیت می کنند. در پی این ویژگی، به ناگذیر مقداری overhead و سربار در پردازش های دیتابیس وارد شده و کارایی را پایین می آورد. به همین جهت توصیه می شود عملیات update در SQLite را در قالب یک تراکنش (transaction) گنجانده (wrap) و پس از انجام موفقیت آمیز تمامی عملیات لازم تراکنش را ثبت نهایی (commit) نمایید. با این اقدام سرعت و کارایی اپلیکیشن به طور چشم گیری افزایش می یابد.

کد زیر این افزایش کارایی را به نمایش می گذارد.

```
db.beginTransaction();
try {
  for (int i= 0; i< values.lenght; i++){
        // TODO prepare ContentValues object values
        db.insert(your_table, null, values);
        // In case you do larger updates
        yieldIfContededSafely()
    }
    db.setTransactionSuccessful();
   } finally {
    db.endTransaction();
}
```

برای آپدیت های زیاد می بایست متد yieldIfContendedSafely() را فراخوانی نمایید. SQLite در طول اجرای تراکنش بر روی دیتابیس قفل می گذارد. حال اگر سرویس گیرنده ی دیگری از داده

های مورد نظر کوئری بگیرد، با فراخوانی این متد، اندروید تراکنش را متوقف کرده و یک تراکنش جدید باز می کند. از این طریق فرایند دوم می تواند به داده ها دسترسی پیدا کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش دوم :

پردازش فایل های XML در اندروید با استفاده از تحلیلگر نحوی xmlPullParser (Parser)

مبحث حاضر به شرح نحوه ی پردازش فایل های XML در اندروید می پردازد.

6-4-پردازش فایل های XML با استفاده از تحلیلگر نحوی XmlPullParser

زبان برنامه نویسی Java تعدادی کتابخانه ی استاندارد برای پردازش فایل های XML ارائه می دهد. تحلیگرهای نحوی SAX و DOM XML که در محیط Android نیز قابل استفاده می باشند.

API یا توابع تحلیلگر نحوی SAX و DOM در زبان جاوا و محیط Android یکسان هستند. البته لازم به توضیح است که دو API نام برده محدودیت های خودشان را دارند و برای استفاده در اندروید توصیه نمی شوند. به این دلیل در آموزش جاری مثالی از کاربرد این کتابخانه عنوان نخواهد شد.

Java همچنین parser یا تحلیل گر نحوی Stax را ارائه می دهد که بین جاوا و بستر اجرای اندروید (platform) مشترک نیست.

محیط Android برای تحلیل گرامری (parse) و نوشتن در فایل XML از کلاس xmlPullParser استفاده می کند. این parser به گونه ای معادل Stax در جاوا است (اما جزئی از بستر اجرای جاوا یا Java SE نمی باشد). برای دسترسی به xmlPullParser می توانید به آدرس http://www.xmlpull.org/ مراجعه نمایید.

جهت تحلیل گرامری و نوشتن در فایل های XML در محیط اجرای اندروید (Android Platform)، کلاس XmlPullParser توصیه می شود. این کتابخانه در مقایسه با SAX و DOM از توابع و API

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

ساده تری برخوردار بوده، سرعت اجرای آن بالا است و نسبت به DOM API حافظه ی کمتری را اشغال می کند.

مثالی از کاربرد XmlPullParser

Javadoc این کتابخانه یک مثال کاربردی و مناسب در خصوص نحوه ی استفاده از آن ارائه می دهد (javadoc از توضیحات و مستندات نوشته در java source فایل HTML ایجاد می کند و برای دیگر توسعه دهندگان این امکان را فراهم می کند تا کد شما بهتر بفهمند).

```
import java.io.IOException;
 import java.io.StringReader;
 import org.xmlpull.v1.XmlPullParser;
 import org.xmlpull.v1.XmlPullParserException.html;
 import org.xmlpull.v1.XmlPullParserFactory;
public class SimpleXmlPullApp
{
   public static void main (String args[])
       throws XmlPullParserException, IOException
    {
       XmlPullParserFactory factory = XmlPullParserFactory.newInstance();
       factory.setNamespaceAware(true);
       XmlPullParser xpp = factory.newPullParser();
       xpp.setInput( new StringReader ( "<foo>Hello World!</foo>" ) );
       int eventType = xpp.getEventType();
       while (eventType != XmlPullParser.END_DOCUMENT) {
        if(eventType == XmlPullParser.START_DOCUMENT) {
           System.out.println("Start document");
        } else if(eventType == XmlPullParser.END_DOCUMENT) {
           System.out.println("End document");
        } else if(eventType == XmlPullParser.START_TAG) {
           System.out.println("Start tag "+xpp.getName());
        } else if(eventType == XmlPullParser.END_TAG) {
           System.out.println("End tag "+xpp.getName());
        } else if(eventType == XmlPullParser.TEXT) {
           System.out.println("Text "+xpp.getText());
        }
        eventType = xpp.next();
       }
   }
}
```

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

ذخیره ی ماندگار داده ها در اپلیکیشن اندروید با استفاده از preferences API (ذخیره و بازیابی اطلاعات مربوط به تنظیمات کاربر) و File API

این آموزش تمرکز خود را بر شرح نحوه ی ذخیره سازی جفت های کلید-مقدار در فایل های اندروید (File based persistence) با فراخوانی preference API قرار می دهد. سپس چگونگی خواندن/نوشتن فایل در اندروید را تشریح می کند.

7-4-File based persistence (ذخیره داده ها در سیستم فایل)

7-4-1-روش های ذخیره ی ماندگار داده ها به صورت محلی (local data persistence)

اندروید این امکان را فراهم می آورد تا داده های اپلیکیشن را در سیستم فایل ذخیره نمایید. در واقع سیستم اندروید به ازای هر اپلیکیشن یک پوشه ی [data/data/[application package ایجاد می کند.

اندروید برای ذخیره ی داده ها در سیستم فایل محلی روش های زیر را پشتیبانی می کند:

- Files – می توان داده ها را در فایل ذخیره کرده و بروز رسانی نمود (روشی قدیمی).
- Preferences – اندروید به شما اجازه می دهد جفت های کلید-مقدار از نوع داده ای اولیه (primitive data type) را به صورت دائمی ذخیره نمایید (به شما امکان می دهد اطلاعات مربوط به تنظیمات انتخابی کاربر را با استفاده از آبجکت sharedpreferences به صورت کلید-مقدار ذخیره کنید).

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

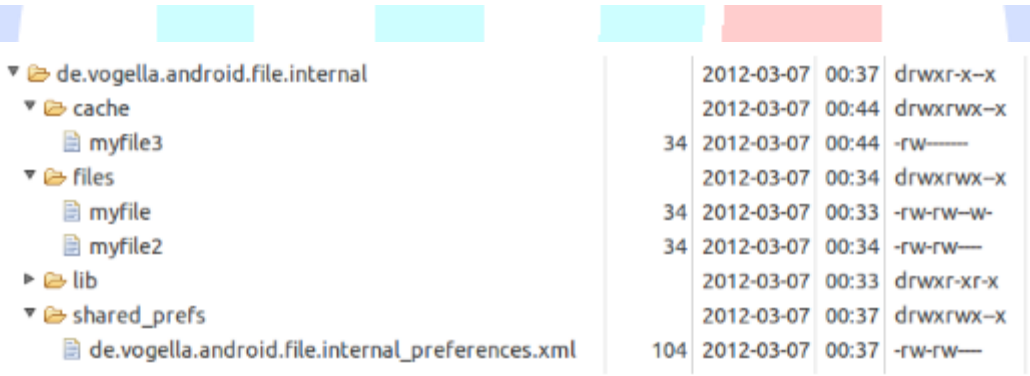
88146323 - 88446780 - 88146330

- SQLite database – می توان نمونه یا نسخه های موجود از دیتابیس SQLite حامل داده های اپلیکیشن را در سیستم فایل و به صورت محلی ذخیره کرد (ذخیره ی داده های اپلیکیشن در سیستم مدیریت متحدالشکل دیتابیس که اندروید توسط SQLite پشتیبانی می کند).

فایل ها در پوشه ی files و تنظیمات اپلیکیشن در قالب فایل های XML داخل پوشه ی shared-prefs ذخیره و نگهداری می شوند.

اگر اپلیکیشن شما یک دیتابیس SQLite ایجاد کند، اندروید به صورت پیش فرض این دیتابیس را در دایرکتوری اصلی اپلیکیشن و تحت پوشه ی databases ذخیره می نماید.

تصویر زیر یک سیستم فایل را به نمایش می گذارد که فایل های ساده، فایل های کش شده (cache file) و اطلاعات مربوط به وضعیت تنظیمات انتخابی کاربر (preferences) را شامل می شود.



تنها اپلیکیشن است که اجازه ی درج داده و نوشتن در پوشه ی اصلی اپلیکیشن (app-directory) را دارد. اپلیکیشن همچنین این قابلیت را دارد که زیرپوشه های (sub-directory) لازم را در پوشه ی اصلی (application directory) ایجاد نماید. جهت ایجاد این زیرپوشه ها، اپلیکیشن مورد نظر می تواند مجوزهای لازم در سطح خواندن و نوشتن را به دیگر برنامه ها اعطا کند.

4-7-2-ذخیره سازی internal داده در مقایسه با ذخیره ی داده ها به صورت external

اندروید دارای دو حافظه ی داخل و خارجی است و داده ها را به صورت internal و external ذخیره می کند. حافظه ی خارجی دارای سطح دسترسی شخصی نبوده (private نیست) و همیشه در دسترس نیست. به عنوان مثال، زمانی که دستگاه اندروید از طریق USB به کامپیوتر وصل می شود، این حافظه ی خارجی به طور موقت از دسترس اپلیکیشن های اندروید خارج می گردد.

4-7-3-جایگذاری اپلیکیشن در حافظه ی خارجی

از ویرایش 8 مجموعه ابزار ساخت و توسعه ی اپلیکیشن های اندروید (SDK level 8) می توان در فایل تنظیمات (manifest) مشخص نمود که اپلیکیشن مورد نظر امکان نصب بر روی حافظه ی خارجی را داشته باشد یا اینکه اپلیکیشن مزبور بایستی بر روی حافظه ی خارجی جایگذاری شود. برای نیل به این هدف کافی است مقدار android:installLocation را در فایل تنظیمات برابر preferExternal یا auto قرار دهید.

در این شرایط، ممکن است برخی از کامپوننت های نرم افزاری و تشکیل دهنده اپلیکیشن بر روی mount point یا محل ذخیره سازی رمزنگاری شده (encrypted) خارجی جایگذاری شوند. دیتابیس و سایر داده های private (با سطح دسترسی شخصی) همچنان فقط بر روی حافظه ی داخلی سیستم (storage system) ذخیره می شوند.

4-8-Preferences (ذخیره و بازگردانی اطلاعات مربوط به تنظیمات کاربر)

4-8-1-ذخیره جفت های کلید-مقدار از نوع داده ای اولیه

اندروید با ارائه و پشتیبانی از کلاس SharedPreferences این امکان را مهیا می کند تا جفت های کلید-مقدار از جنس داده ای اولیه را در سیستم فایل اندروید ذخیره نمایید (تنظیمات انتخابی کاربر همچون اندازه ی فونت و رنگ متن که داده های کوچک هستند را به صورت key-value ذخیره نمایید).

Preferences و تنظیمات انتخابی کاربر را می توان به راحتی داخل فایل های منبع XML تعریف نمود.

کلاس PreferenceManager با ارائه ی توابعی به توسعه دهنده این امکان را می دهد تا اطلاعات مربوط به تنظیمات کاربر، مستقر در یک فایل XML را بازیابی کند. کد زیر نحوه ی دسترسی به تنظیمات ذخیره شده در یک فایل XML را به نمایش می گذارد.

```
# getting preferences from a specified file
SharedPreferences settings = getSharedPreferences("Test", Context.MODE_PRIVATE);
```

Preferences می بایست به صورت private ویژه ی اپلیکیشن مورد نظر ایجاد شود. تمامی کامپوننت ها و اجزا تشکیل دهنده ی اپلیکیشن می توانند به اطلاعات preferences دسترسی داشته باشند. به اشتراک گذاری اطلاعات با یک اپلیکیشن دیگر که دارای فایل preference سراسری با دسترسی در سطح خواندن و نوشتن (world readable/writable) می باشد، امری است که به ندرت اتفاق می افتد. چرا که در این صورت کامپوننت خارجی می بایست از اسم و محل قرار گیری دقیق فایل مطلع باشد.

Preferences پیش فرض به راحتی با فراخوانی متد PreferenceManager.getDefaultSharedPreferences(this) در اختیار دیگر کامپوننت های اپلیکیشن قرار می گیرد. برای دسترسی به مقدار preference کافی است کلید مربوطه و آبجکتی از کلاس SharedPreferences را مانند زیر مورد استفاده قرار دهید.

```
SharedPreferences settings = PreferenceManager.getDefaultSharedPreferences(getActivity();
String url = settings.getString("url", "n/a");
```

به منظور ایجاد و ویرایش مقادیر preferences لازم است متد edit() را بر روی آبجکت SharedPreferences فراخوانی نمایید. پس از ویرایش مقدار، بایستی متد apply() را فراخوانی کنید تا مقادیر جدید به صورت ناهمزمان در سیستم فایل اعمال شوند. لازم به توضیح است که متد commit() تغییرات را به صورت همزمان به فایل سیستم اعمال می کند، از این جهت استفاده از متد مذکور در این سناریو توصیه نمی شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
Editor edit = preferences.edit();
edit.putString("username", "new_value_for_user");
edit.apply();
```

2-8-4-گوش فراخوانی به تغییرات در تنظیمات کاربر به وسیله ی preference listener

می توانید با فراخوانی متد registerOnSharedPreferenceChangeListener() بر روی آبجکت SharedPreference به تغییرات در preferences گوش داده و از آن ها مطلع شوید.

```
SharedPreferences prefs =
    PreferenceManager.getDefaultSharedPreferences(this);
// Instance field for listener
listener = new SharedPreferences.OnSharedPreferenceChangeListener() {
  public void onSharedPreferenceChanged(SharedPreferences prefs, String key) {
    // Your Implementation
  }
};

prefs.registerOnSharedPreferenceChangeListener(listener);
```

توجه داشته باشید که SharedPreferences تمامی listener یا گوش فراخوان ها را داخل WeakHashMap نگه می دارد، به همین علت چنانچه کد شما اشاره گری به آن تعریف نکرده باشد (آن را داخل متغیر نگه ندارد)، listener ها بازیافت می شوند و اطلاعات مورد نظر شما از دست خواهد رفت.

آموزش: آنچه لازم است

پروژه ی این بخش مبتنی بر نمونه اپلیکیشن "de.vogella.android.socialapp" از آموزش action bar در مباحث قبلی می باشد.

تمرین: آنچه لازم است

برای تمرین زیر لازم است یک پروژه ی اندرویدی به نام com.example.android.rssfeed همراه با یک المان Settings در action bar ایجاد نمایید.

تمرین: ذخیره ی تنظیمات کاربر برای RSSfeed

در تمرین جاری، برنامه را گونه ای می نویسید که یک activity دیگر باز شده و به کاربر این امکان را بدهد تا URL دلخواه خود برای RSS feed را وارد نماید. این activity ظاهر و UI خود را از یک فایل xml که حاوی تنظیمات انتخابی کاربر است (xml preference file)، می خواند.

4-8-3-ایجاد فایل preference

یک فایل XML به نام mypreferences.xml در پوشه ی XML ایجاد نمایید. المان های زیر را به آن اضافه نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android" >
   <EditTextPreference
      android:key="url"
      android:title="Rss feed URL"
      android:inputType="textUri"/>
   <CheckBoxPreference android:title="Aktiv" android:key="active"/>
</PreferenceScreen>
```

4-8-4-ایجاد activity جهت دریافت ورودی از کاربر

ابتدا کلاس SettingsActivity که از PreferenceActivity ارث بری می کند را ایجاد نمایید. این activity فایل preference را بارگذاری نموده و به کاربر اجازه می دهد تا مقادیر را ویرایش کند.

```
package com.example.android.rssreader;
import android.os.Bundle;
import android.preference.PreferenceActivity;
public class SettingsActivity extends PreferenceActivity {
      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            addPreferencesFromResource(R.xml.mypreferences);
      }
}
```

این کلاس را به عنوان یک activity در فایل تنظیمات اپلیکیشن AndroidManifest.xml ثبت و معرفی نمایید.

4-8-5-متصل کردن activity مربوط به تنظیمات و دریافت ورودی کاربر

Activity مربوط به preference و دریافت تنظیمات دلخواه کاربر را با فراخوانی متد onOptionsItemSelected() باز نمایید.

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
```

```
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.mainmenu, menu);
        return true;
}
// This method is called once the menu is selected
@Override
public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case R.id.preferences:
                // Launch settings activity
                Intent i = new Intent(this, SettingsActivity.class);
                startActivity(i);
                break;
                // more code...
        }
        return true;
}
```

6-8-4-بارگذاری RSS feed با استفاده از مقدار preference

تکه کد زیر نشان می دهد چگونه می توان از یک متد به مقدار preference دسترسی پیدا کرد.

```
// if you use this in a service or activity you can use this
// if you use this in a fragment use getActivity() or getContent() as parameter
SharedPreferences settings = PreferenceManager.getDefaultSharedPreferences(this);
String url = settings.getString("url", "http://www.vogella.com/article.rss");
```

تست اپلیکیشن

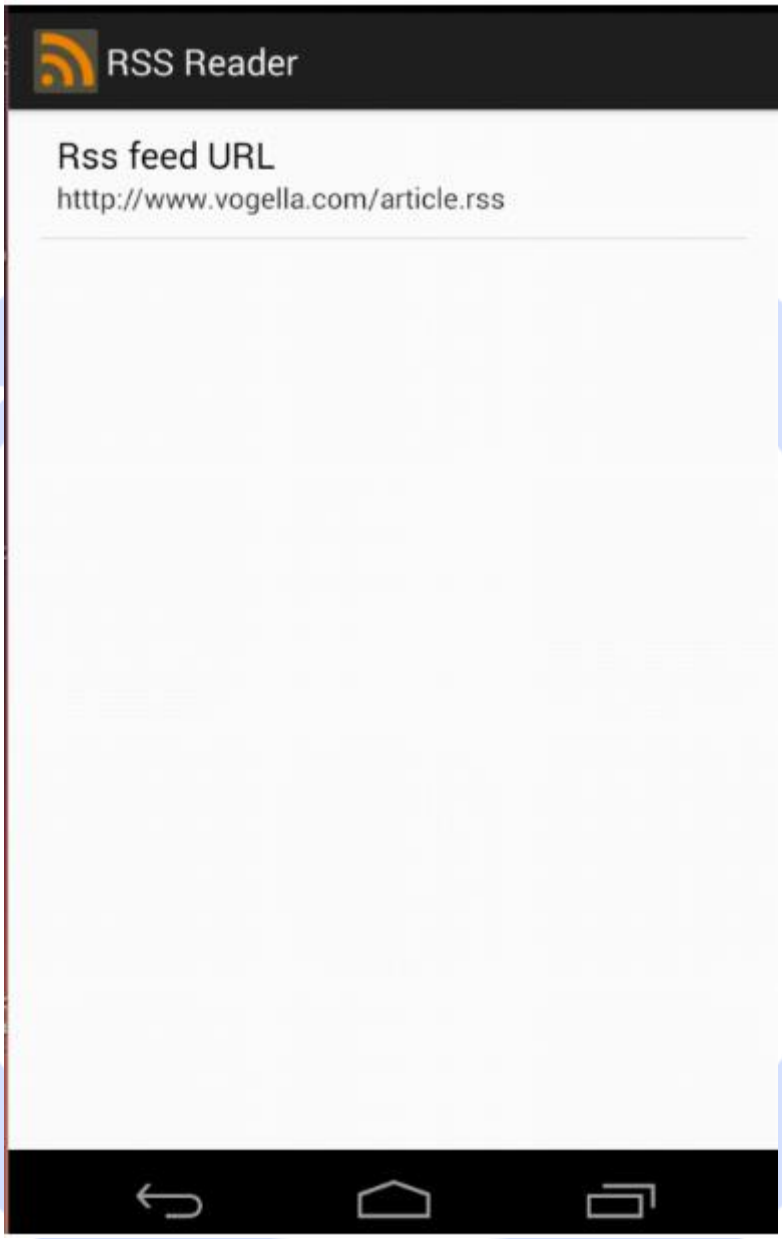
اپلیکیشن خود را اجرا نمایید. آیتم Settings (action) را از action bar انتخاب کنید. بایستی یک activity دیگر به نمایش در آمده و شما بتوانید آدرس URL دلخواه را وارد کنید. پس از کلیک بر روی دکمه ی بازگشت (back) و بروز رسانی (refresh)، با بررسی activity اطمینان حاصل کنید که مقدار url در activity بکار برده شده است.

تمرین اضافی: نمایش مقدار جاری در settings

تکه کد زیر نشان می دهد چگونه می توان مقدار جاری را در صفحه ی preference نمایش داد.

```
package com.example.android.rssreader;
import android.content.SharedPreferences;
```

```
import android.content.SharedPreferences.OnSharedPreferenceChangeListener;
import android.os.Bundle;
import android.preference.EditTextPreference;
import android.preference.ListPreference;
import android.preference.Preference;
import android.preference.PreferenceActivity;
import android.preference.PreferenceCategory;
public class SettingsActivity extends PreferenceActivity implements
        OnSharedPreferenceChangeListener {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        addPreferencesFromResource(R.xml.mypreferences);
        // show the current value in the settings screen
        for (int i = 0; i < getPreferenceScreen().getPreferenceCount(); i++) {
            initSummary(getPreferenceScreen().getPreference(i));
        }
    }
    @Override
    protected void onResume() {
        super.onResume();
        getPreferenceScreen().getSharedPreferences()
                .registerOnSharedPreferenceChangeListener(this);
    }
    @Override
    protected void onPause() {
        super.onPause();
        getPreferenceScreen().getSharedPreferences()
                .unregisterOnSharedPreferenceChangeListener(this);
     }
    @Override
    public void onSharedPreferenceChanged(SharedPreferences sharedPreferences,
            String key) {
        updatePreferences(findPreference(key));
    }
    private void initSummary(Preference p) {
        if (p instanceof PreferenceCategory) {
            PreferenceCategory cat = (PreferenceCategory) p;
            for (int i = 0; i < cat.getPreferenceCount(); i++) {
                initSummary(cat.getPreference(i));
            }
        } else {
            updatePreferences(p);
        }
    }
    private void updatePreferences(Preference p) {
        if (p instanceof EditTextPreference) {
            EditTextPreference editTextPref = (EditTextPreference) p;
            p.setSummary(editTextPref.getText());
        }
    }
}
```



File API-9-4

1-9-4-استفاده از File API

برای دسترسی به سیستم فایل می توانید از طریق کلاس های java.io اقدام نمایید. اندروید کلاس های کمکی (helper) در اختیار توسعه دهنده قرار می دهد که ایجاد و دسترسی به فایل ها و

پوشه های جدید را آسان می سازد. برای مثال، getDir(String, int) پوشه ی مورد نیاز را ایجاد نموده و دسترسی به آن را فراهم می سازد. متد openFileInput(String s) یک فایل برای ورودی باز کرده و متد openFileOutput(String s, Context.MODE_PRIVATE) نیز یک فایل جدید ایجاد می نماید.

اندروید از فایل های خواندنی و نوشتنی سراسری (world readable/writable) پشتیبانی می کند. اما به طور کلی توصیه می شود دسترسی به فایل ها را فقط به خود اپلیکیشن محدود نمایید (فایل های اپلیکیشن را private نگه دارید) و در صورت نیاز به اشتراک گذاری داده ها بین اپلیکیشن ها از کامپوننت نرم افزاری content provider استفاده کنید.

مثال زیر استفاده از API مورد نظر را به نمایش می گذارد.

```
public class Util {
    public static void writeConfiguration(Context ctx ) {
        try (FileOutputStream openFileOutput =
              ctx.openFileOutput( "config.txt", Context.MODE_PRIVATE);) {

            openFileOutput.write("This is a test1.".getBytes());
            openFileOutput.write("This is a test2.".getBytes());
        } catch (Exception e) {
            // not handled
        }
    }
}
public void readFileFromInternalStorage(String fileName) {
    String eol = System.getProperty("line.separator");
    try (BufferedReader input = new BufferedReader(new InputStreamReader(
          openFileInput(fileName))); ){
     String line;
     StringBuffer buffer = new StringBuffer();
     while ((line = input.readLine()) != null) {
         buffer.append(line + eol);
     }
    } catch (Exception e) {
          // we do not care
    }
}
```

2-9-4-حافظه و محل ذخیره سازی خارجی (external storage)

اندروید از حافظه ی خارجی مانند SD و دسترسی به آن نیز پشتیبانی می کند. تمامی فایل ها و پوشه های مستقر در حافظه ی خارجی برای کلیه ی اپلیکیشن هایی که دارای مجوز لازم هستند، قابل دسترسی می باشد.

جهت خواندن داده از حافظه ی خارجی، اپلیکیشن شما می بایست از مجوز android.permission.READ_EXTERNAL_STORAGE برخوردار باشد.

به منظور درج و نوشتن داده در حافظه ی خارجی، اپلیکیشن شما می بایست از مجوز android.permission.WRITE_EXTERNAL_STORAGE برخوردار باشد. برای دسترسی به حافظه ی خارجی، کافی است متد Environement.getExternalStorageDirectory() را فراخوانی نمایید.

با فراخوانی متد زیر شما می توانید وضعیت حافظه ی خارجی را بررسی کنید. چنانچه دستگاه اندروید از طریق USB به کامپیوتر متصل شده است، در آن صورت کارت SD که به عنوان حافظه خارجی مورد استفاده قرار گرفته است، از دسترس خارج می شود.

```
Environment.getExternalStorageState().equals(Environment.MEDIA_MOUNTED)
```

تکه کد زیر مثالی از خواندن اطلاعات از حافظه ی خارجی را به نمایش می گذارد.

```
private void readFileFromSDCard() {
    File directory = Environment.getExternalStorageDirectory();
    // assumes that a file article.rss is available on the SD card
    File file = new File(directory + "/article.rss");
    if (!file.exists()) {
        throw new RuntimeException("File not found");
    }
    Log.e("Testing", "Starting to read");
    BufferedReader reader = null;
    try {
        reader = new BufferedReader(new FileReader(file));
        StringBuilder builder = new StringBuilder();
        String line;
        while ((line = reader.readLine()) != null) {
            builder.append(line);
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        if (reader != null) {
```

```
                try {
                    reader.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
}
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

اتصال به اینترنت، اجرای عملیات HTTP و دسترسی به منابع موجود در سطح وب در اندروید/ Android networking

این آموزش نحوه ی دسترسی به منابع و محتوا از طریق HTTP را برای شما شرح می دهد.

پروژه ی آموزش حاضر در محیط برنامه نویسی Eclipse نوشته شده، مبتنی بر ویرایش 1.6 زبان Java و ورژن 5.0 سیستم عامل اندروید می باشد.

4-10-مروری بر اتصال به اینترنت و دسترسی به منابع از اینترنت در اندروید

1-10-4-دسترسی به اینترنت در اندروید

اندروید پکیج java.net را شامل می شود. با وارد کردن این پکیج در پروژه می توانید به منابع و محتوای مورد نیاز از طریق شبکه دسترسی پیدا کنید. کلاس پایه برای اتصال HTTP در پکیج java.net، کلاس HttpURLConnection است.

اندروید حاوی اطلاعات استاندارد شبکه جاوا است .

Net Package می تواند از منابع دسترسی به شبکه موجود استفاده کند ، به عنوان کلاس اصلی شبکه دسترسی HTTP در جاوا است ، NetPackage یک کلاس HttpURLConnection است.

اجرای عملیات مربوط به شبکه با API جاوا طاقت فرسا است. بدین معنی که توسعه دهنده می بایست connection را باز کند/ببندد، cache را فعال نموده و مطمئن شود که عملیات مربوط به شبکه در background thread (به صورت موازی و در پس زمینه) اجرا می شوند.

تعداد زیادی کتابخانه ی کد باز (open source) وجود دارد که در اختیار برنامه نویس قرار گرفته و انجام عملیات مزبور را آسان می سازد. پرکاربردترین این کتابخانه ها به شرح زیر می باشند:

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

- OkHttp – OkHttp efficient HTTP access برای اتصال و دسترسی بهینه HTTP
- Retrofit – Retrofit REST based clients برای سرویس گیرنده های مبتنی بر REST
- Glide – Glide image processing برای پردازش تصویر

10-4-2-مجوز اتصال به اینترنت

به منظور اتصال به اینترنت، اپلیکیشن شما به مجوز android.permission.INTERNET احتیاج دارد. در ورژن های جدید کتابخانه های اندروید (API version های نوین)، این مجوز به صورت خودکار به اپلیکیشن اعطا می شود.

10-4-3-بررسی وضعیت اتصال به اینترنت

دستگاه های اندروید همیشه به اینترنت دسترسی ندارند. برای بررسی اینکه آیا دستگاه مورد نظر به شبکه و اینترنت دسترسی دارد یا خیر، اپلیکیشن شما بایستی مجوز android.permission.ACCESS_NETWORK_STATE را داشته باشد.

جهت بررسی وضعیت اتصال به اینترنت، می توانید از کد زیر استفاده کنید.

```
public boolean isNetworkAvailable() {
   ConnectivityManager cm = (ConnectivityManager)
         getSystemService(Context.CONNECTIVITY_SERVICE);
   NetworkInfo networkInfo = cm.getActiveNetworkInfo();
   // if no network is available networkInfo will be null
   // otherwise check if we are connected
   if (networkInfo != null && networkInfo.isConnected()) {
      return true;
   }
   return false;
}
```

همان طور که در بالا گفته شد، برای بررسی وضعیت اتصال به شبکه، اپلیکیشن شما به مجوز ACCESS_NETWORK_STATE احتیاج دارد.

10-4-4-روش های بهینه برای اتصال و دسترسی به اینترنت در اندروید

در اپلیکیشن های اندرویدی، شما باید از اجرای عملیات طولانی و سنگین در UI thread خودداری نمایید. از جمله ی این عملیات می توان به اتصال به اینترنت و دسترسی به فایل اشاره کرد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

از ویرایش 3.0 به بعد اندروید، سیستم طوری تعبیه شده است که اگر دسترسی به اینترنت در UI thread رخ دهد، خطای NetworkOnMainThreadException صادر شده و به تبع آن سیستم به طور ناگهانی از کار می افتد.

روش معمول و بهینه برای اتصال به اینترنت و دسترسی به منابع سطح وب (در یک اپلیکیشن با کیفیت)، استفاده از یک سرویس است. گفتنی است که می توان از یک activity یا fragment نیز به اینترنت دسترسی را انجام داد، اما استفاده از سرویس برای نیل به این هدف طراحی بهینه تری را برای اپلیکیشن شما رقم می زند و همچنین به ساده نگه داشتن کد activity شما کمک شایانی می نماید.

توجه: جهت تست می توانید با درج تکه کد زیر در ابتدی متد onCreate() کلاس activity، دسترسی به اینترنت را در thread اصلی اپلیکیشن فراهم کنید.

```
StrictMode.ThreadPolicy policy = new StrictMode.
ThreadPolicy.Builder().permitAll().build();
StrictMode.setThreadPolicy(policy);
```

Unresolved directive in 001_article.adoc - include::../JavaNetworking/010_overview.adoc[] == Web Sockets

Web Socket یک استاندارد مبتنی بر HTTP برای تبادل ناهمزمان پیغام (asynchronous message-based communication) بین سرویس گیرنده و سرویس دهنده است. به منظور راه اندازی این ارتباط، یک درخواست HTTP GET با یک HTTP header خاص ایجاد نمایید.اگر سرویس دهنده این درخواست را پذیرفت، سرویس گیرنده و سرویس دهنده می توانند با هم پیغام (داده هایی را) رد و بدل کنند.

این پیغام ها می توانند متن یا داده های binary باشند. لازم به ذکر است که داده های مورد تبادل می بایست کم حجم باشند چرا که پروتکل web socket اساسا به منظور انتقال payload و داده های کم حجم تعبیه و طراحی شده است.

بهترین روش برای تبادل پیغام و داده بین کلاینت و سرور، قرار دادن آن در قالب JSON می باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

پیام می توانند متنی و یا داده های باینری و نسبتا کوچک باشند ، به عنوان یک وب سوکت پروتکل در نظر گرفته شوندکه با محموله های کوچک در داده استفاده می شوند.

این تمرین خوبی به استفاده از JSON به عنوان فرمت داده ها را برای پیام است .

بخش پنجم :

استفاده از کتابخانه ی Retrofit 2.0 به عنوان REST Client

Retrofit-4-11

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Retrofit عبارت است از یک REST Client برای Java و Android که توسط Square ارائه می شود. این کتابخانه، بازیابی و بارگذاری JSON یا هر داده ی ساخت یافته ی دیگری را از طریق یک وب سرویس مبتنی بر REST انجام می دهد. Retrofit را می توان با یک converter تنیظم نموده و برای serialize داده ها مورد استفاده قرار داد. معمولا برای داده هایی که در فرمت JSON ذخیره شده اند از Gson استفاده می شود، با این حال شما می توانید converter های اختصاصی و دلخواه خود را جهت پردازش XML یا دیگر پروتکل ها مورد استفاده قرار دهید. Retrofit از کتابخانه ی OkHttp برای مدیریت درخواست های HTTP بهره می گیرد.

Retrofit به شما این امکان را می دهد از Converter های زیر استفاده نمایید.

- Gson: com.squareup.retrofit:converter-gson
- Jackson: com.squareup.retrofit:converter-jackson
- Moshi: com.squareup.retrofit:converter-moshi
- Protobuf: com.squareup.retrofit:converter-protobuf
- Wire: com.squareup.retrofit:converter-wire
- Simple XML: com.squareup.retrofit:converter-simplexml

برای کار با Retrofit به سه کلاس زیر احتایج دارید.

1. کلاس model ویژه ی نگاشت داده های JSON.
2. Interface هایی که عملیات و توابع HTTP را تعریف می کنند.
3. کلاس Retrofit.Builder – نمونه ای که از این interface استفاده می کند. Builder API امکان تعریف آدرس URL (آدرس سرویس) endpoint را برای عملیات HTTP فراهم می آورد.

می توانید با مراجعه به سایت زیر، داده های مبتنی بر JSON خود را به فرمت POJO (آبجکت های ساده و بدون متد جاوا) تبدیل نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

تمرین: استفاده از Retrofit برای کوئری گرفتن و پرس و جو از Stackoverflow در اندروید

هدف از این تمرین

StackOverflow یک سایت پر بازدید است که برنامه نویس ها مشکلات برنامه نویسی خود را در آن مطرح می کنند. این سایت یک REST API ارائه می کند که به خوبی مستندسازی شده و با توضیحات لازم در اختیار توسعه دهندگان قرار می گیرد. Query ها را می توان با استفاده از این API ساخت. می توانید جهت مشاهده ی مستندات این API و استفاده از قابلیت های آن به آدرس https://api.stackexchange.com/docs/search مراجعه نمایید. در تمرین زیر با کتابخانه ی Retrofit REST و بر اساس تگ هایی که مشخص می شود، می توانید از سوال های پاسخ داده یا تایید نهایی نشده کوئری بگیرید.

در مثال جاری URL کوئری زیر را بکار می بریم. تگ ها از طریق کد ما مشخص می شوند.

https://api.stackexchange.com/2.2/search?order=desc&sort=activity&tagged=android&site=stackoverflow

1-11-4-ساخت پروژه و تنظیمات اولیه

یک اپلیکیشن اندروید به نام com.vogella.android.retrofitstackoverflow ایجاد نمایید. این اپلیکیشن بایستی اسم com.vogella.android.retrofitstackoverflow را به عنوان top level package مورد استفاده قرار دهد.

کتابخانه ی زیر را به فایل build.gradle اضافه نمایید.

```
compile 'com.squareup.retrofit:retrofit:2.0.0-beta2'
compile 'com.squareup.retrofit:converter-gson:2.0.0-beta2'
```

مجوز لازم برای دسترسی به اینترنت را در فایل تنظیمات (manifest) اپلیکیشن تنظیم نمایید.

2-11-4-تعریف API و کلاس Retrofit adapter

در پاسخ JSON ای که از StackOverflow دریافت می شود، تنها به title و link نیاز داریم. کلاس های زیر را ایجاد نمایید.

```
package android.vogella.com.retrofitstackoverflow;
// This is used to map the JSON keys to the object by GSON
public class Question {
   String title;
   String link;
   @Override
   public String toString() {
      return(title);
   }
}


package android.vogella.com.retrofitstackoverflow;
import java.util.List;
public class StackOverflowQuestions {
   List<Question> items;
}
```

با پیاده سازی interface زیر REST API را برای Retrofit تعریف نمایید.

توجه: از آنجایی stackoverflow پاسخ را در یک آبجکت جاسازی می کند، لازم است ساختار داده ای مورد نیاز را از نوع list تعریف نمایید.

```
package com.vogella.android.retrofitstackoverflow;
import retrofit.Callback;
import retrofit.http.GET;
import retrofit.http.Query;
import retrofit.Call;
public interface StackOverflowAPI {
   @GET("/2.2/questions?order=desc&sort=creation&site=stackoverflow")
   Call<StackOverflowQuestions> loadQuestions(@Query("tagged") String tags);
}
```

یک منوی XML مشابه زیر با نام main_menu.xml ایجاد نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
   <item
      android:id="@+id/menu_load"
      android:title="Load" />
</menu>
```

کد کلاس activity خود را طوری ویرایش نمایید که به شما امکان کوئری گرفتن از سوال هایی را بدهد که با رشته ی "android" علامت یا تگ گذاری شده اند.

```
package com.vogella.android.retrofitstackoverflow;
import android.app.Activity;
import android.app.ListActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.Window;
import android.widget.ArrayAdapter;
import android.widget.Toast;
import java.util.ArrayList;
import retrofit.Call;
import retrofit.Callback;
import retrofit.GsonConverterFactory;
import retrofit.Response;
import retrofit.Retrofit;
public class MainActivity extends ListActivity implements Callback<StackOverflowQuestions> {
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      requestWindowFeature(Window.FEATURE_INDETERMINATE_PROGRESS);
      requestWindowFeature(Window.FEATURE_PROGRESS);
      ArrayAdapter<Question> arrayAdapter =
            new ArrayAdapter<Question>(this,
                  android.R.layout.simple_list_item_1,
                  android.R.id.text1,
                  new ArrayList<Question>());
      setListAdapter(arrayAdapter);
      setProgressBarIndeterminateVisibility(true);
      setProgressBarVisibility(true);
   }
   @Override
   public boolean onCreateOptionsMenu(Menu menu) {
      getMenuInflater().inflate(R.menu.main_menu, menu);
      return true;
   }
   @Override
   public boolean onOptionsItemSelected(MenuItem item) {
      setProgressBarIndeterminateVisibility(true);
      Retrofit retrofit = new Retrofit.Builder()
            .baseUrl("https://api.stackexchange.com")
```

```
            .addConverterFactory(GsonConverterFactory.create())
            .build();
    // prepare call in Retrofit 2.0
    StackOverflowAPI stackOverflowAPI = retrofit.create(StackOverflowAPI.class);
    Call<StackOverflowQuestions> call = stackOverflowAPI.loadQuestions("android");
    //asynchronous call
    call.enqueue(this);
    // synchronous call would be with execute, in this case you
    // would have to perform this outside the main thread
    // call.execute()
    // to cancel a running request
    // call.cancel();
    // calls can only be used once but you can easily clone them
    //Call<StackOverflowQuestions> c = call.clone();
    //c.enqueue(this);
    return true;
  }
  @Override
  public void onResponse(Response<StackOverflowQuestions> response, Retrofit retrofit) {
    setProgressBarIndeterminateVisibility(false);
    ArrayAdapter<Question> adapter = (ArrayAdapter<Question>) getListAdapter();
    adapter.clear();
    adapter.addAll(response.body().items);
  }
  @Override
  public void onFailure(Throwable t) {
    Toast.makeText(MainActivity.this, t.getLocalizedMessage(), Toast.LENGTH_SHORT).show();
  }
}
```

تمرین: استفاده از Retrofit برای دسترسی به توابع Github (API) در اندروید

تنظیم پروژه

به منظور تست Retrofit، یک اپلیکیشن به نام Retrofit Github ایجاد نمایید. اسم com.vogella.android.retrofitgithub را به عنوان top level package مورد استفاده قرار دهید.

برای استفاده از Retrofit در اپلیکیشن مورد نظر، کتابخانه (dependency) زیر را به فایل build.gradle خود اضافه نمایید.

```
compile 'com.squareup.retrofit:retrofit:2.0.0-beta2'
compile 'com.squareup.retrofit:converter-gson:2.0.0-beta2'
```

مجوز لازم برای دسترسی به اینترنت را در فایل تنظیمات اپلیکیشن (manifest) تنظیم نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.vogella.android.retrofitgithub" >
   <uses-permission android:name="android.permission.INTERNET"/>
   <application
      android:allowBackup="true"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >
      <activity
         android:name=".MainActivity"
         android:label="@string/app_name" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
   </application>
</manifest>
```

Interface و کلاس زیر را برای Retrofit API ایجاد نمایید.

```
package com.vogella.android.retrofitgithub;

// This is used to map the JSON keys to the object by GSON
public class GithubRepo {

   String name;
   String url;

   @Override
   public String toString() {
      return(name + " " +  url);
   }
}
package com.vogella.android.retrofitgithub;

// This is used to map the JSON keys to the object by GSON
public class GithubUser {

   String login;
   String name;
   String email;

   @Override
   public String toString() {
      return(login);
   }
}
package com.vogella.android.retrofitgithub;
```

```
import java.util.List;

import retrofit2.Call;
import retrofit2.http.GET;
import retrofit2.http.Path;


public interface GithubAPI {
   String ENDPOINT = "https://api.github.com";

   @GET("/users/{user}")
   Call<GithubUser> getUser(@Path("user") String user);

   @GET("users/{user}/repos")
   Call<List<GithubRepo>> getRepos(@Path("user") String user);

}
package com.vogella.android.retrofitgithub;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;

import com.google.gson.Gson;
import com.google.gson.GsonBuilder;

import java.util.List;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

public class MainActivity extends Activity implements Callback<GithubUser> {

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

   }
   public void onClick(View view) {
      Gson gson = new GsonBuilder()
            .setDateFormat("yyyy-MM-dd'T'HH:mm:ssZ")
            .create();
      Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(GithubAPI.ENDPOINT)
            .addConverterFactory(GsonConverterFactory.create(gson))
            .build();
```

```
    GithubAPI githubUserAPI = retrofit.create(GithubAPI.class);

    switch (view.getId()) {
      case R.id.loadUserData:
        // prepare call in Retrofit 2.0

        Call<GithubUser> callUser = githubUserAPI.getUser("vogella");
        //asynchronous call
        callUser.enqueue(this);
        break;
      case R.id.loadRepositories:
        Call<List<GithubRepo>> callRepos = githubUserAPI.getRepos("vogella");
        //asynchronous call
        callRepos.enqueue(repos);
        break;
    }
}

Callback repos = new Callback<List<GithubRepo>>(){

    @Override
    public void onResponse(Call<List<GithubRepo>> call, Response<List<GithubRepo>> response) {
      if (response.isSuccessful()) {
        List<GithubRepo> repos = response.body();
        StringBuilder builder = new StringBuilder();
        for (GithubRepo repo: repos) {
          builder.append(repo.name + " " + repo.toString());
        }
        Toast.makeText(MainActivity.this, builder.toString(), Toast.LENGTH_SHORT).show();

      } else
      {
        Toast.makeText(MainActivity.this, "Error code " + response.code(), Toast.LENGTH_SHORT).show();
      }

    }

    @Override
    public void onFailure(Call<List<GithubRepo>> call, Throwable t) {
      Toast.makeText(MainActivity.this, "Did not work " +  t.getMessage(), Toast.LENGTH_SHORT).show();
    }
};

@Override
public void onResponse(Call<GithubUser> call, Response<GithubUser> response) {
    int code = response.code();
    if (code == 200) {
      GithubUser user = response.body();
      Toast.makeText(this, "Got the user: " + user.email, Toast.LENGTH_LONG).show();
    } else {
      Toast.makeText(this, "Did not work: " + String.valueOf(code), Toast.LENGTH_LONG).show();
```

```
    }
  }

  @Override
  public void onFailure(Call<GithubUser> call, Throwable t) {
    Toast.makeText(this, "Nope", Toast.LENGTH_LONG).show();

  }
}
```

url مقابل https://api.github.com/users/vogella را در نوار آدرس مرورگر وارد نمایید. پس از پیمایش به آدرس مذکور، یک کلاس data model و interface با داده های مورد نظر ایجاد کنید. حال این داده ها را با Retrofit خوانده و در یک لیست به نمایش بگذارد.

بخش ششم :

آموزش RxJava 2.0

RxJava 2.0-12-4

RxJava_4_12_1 و شرح مفهوم برنامه نویسی Reactive یا ناهمزمان

هر برنامه ای که دارای چهار ویژگی یا پارامتر (قابلیت مدیریت خطاها و ارائه ی بهترین سرویس) resilience، (انعطاف و مقایس پذیری) scalability، (واکنش گرا و تعاملی) responsive و (رویداد محور) Event-driven باشد را در اصطلاح reactive می نامند. در مدل برنامه نویسی reactive، سرویس گیرنده یا کاربر به محض ورود داده به آن واکنش نشان می دهند. به همین خاطر برنامه نویسی asynch را برنامه نویسی reactive نیز می خوانند. مدل برنامه نویسی reactive این امکان را فراهم می آورد تا تغییرات event ها را به observer ها (توابع) گوش فرا دهنده و ثبت شده، منتشر (propagate) و اعمال نمایید.

در واقع RxJava امکان پیاده سازی وظایف غیرهمزمان را به راحتی برای توسعه دهنده فراهم می آورد.

RxJava پیاده سازی از ReativeX برای Java می باشد که NetFlix توسعه داده و در بین برنامه نویسان جاوا محبوبیت زیادی پیدا کره است. Rxjava در سال 2014 کد باز (open source) شده و تحت آدرس http://reactivex.io/ میزبانی می شود. ورژن جاوایی آن تحت https://github.com/ReactiveX/RxJava قابل دسترسی می باشد. این کتابخانه تحت لیسانس Apache 2.0 منتشر می شود.

RxJava خود را تحت عنوان یک API که برای برنامه نویسی ناهمزمان طراحی شده (تعریف تسک ها و وظایف ناهمزمان) و از stream observable (آبجکت های observable اطلاعاتی را منتشر کرده و آن ها را به توابع گوش فرادهنده به تغییرات که همان observer ها هستند ارسال می کنند) استفاده می کند، معرفی می نماید.

2-12-4-اضافه کردن کتابخانه RxJava 2.0

از زمان نگارش این مقاله، ویرایش جدیدتری از کتابخانه ی RxJava ارائه شده است. مقدار g.a.v را با 2.0.1 یا نسخه ی جدیدتر جایگزین نمایید.

برای افزودن کتابخانه ی مزبور ویژه ی سیستم کامپایل Maven، می توانید تکه کد زیر را در بخش dependency ها اضافه نمایید.

```
<dependency>
   <groupId>io.reactivex.rxjava2</groupId>
   <artifactId>rxjava</artifactId>
   <version>g.a.v</version>
</dependency>
```

برای سیستم کامپایل Gradle، می توانید RxJava را از طریق دستور زیر اضافه نمایید.

```
compile group: 'io.reactivex.rxjava2', name: 'rxjava', version: 'g.a.v'
```

3-12-4-برنامه نویسی ناهزمان (Async)

امروزه برنامه نویسی به سبک دستوری/imperative طوری که اپلیکیشن در لحظه تنها یک عملیات را پردازش کند (single-thread) دیگر به هیچ وجه کارامد نیست و ممکن است UI های unresponsive را مسدود کند و در نهایت تجربه ی کاربری ضعیفی را در پی داشته باشد.

با مدیریت رخدادهای پیش بینی نشده به صورت ناهمزمان (async)، می توان از این اتفاق جلوگیری کرد. برای مثال، چنانچه لازم باشد منتظر فراخوانی وب سرویس یا کوئری دیتابیس باشید، اگر شبکه پاسخگو نبود (responsive)، قطعا اپلیکیشن هنگ خواهد کرد.

مثالی در این زمینه:

```
public List<Todo> getTodos() {
    List<Todo> todosFromWeb = // query a webservice (with bad network latency)
    return todosFromDb;
}
```

فراخوانی متد getTodos() از thread اصلی یا یک UI thread سبب می شود تا زمان رسیدن todosFromWeb، اپلیکیشن nonresponsive بوده و تجربه ی کاربری ضعیفی برای کاربر رقم خورده شود.

برای بهبود کارایی این query که مدت زمان اجرای آن مشخص نیست، می بایست آن را در یک thread دیگر اجرا نموده و سپس به هنگام رسیدن نتیجه، thread اصلی را مطلع نمایید.

```
public void getTodos(Consumer<List<Todo>> todosConsumer) {
    Thread thread = new Thread(()-> {
        List<Todo> todosFromWeb = // query a webservice
        todosConsumer.accept(todosFromWeb);
    });
    thread.start();
}
```

حال پس از فراخوانی متد getTodos(Consumer<List<Todo>> todosConsumer)، نخ یا thread اصلی می تواند بدون متوقف شدن همچنان به اجرا ادامه دهد و زمانی که متد accept از consumer صدا خورده شد، واکنش نشان دهد.

4-12-4- Observable ها، Observer ها و Subscription ها

RxJava از مفهوم Observable ها و Observer ها استفاده می کند. Observer ها توابعی هستند که با گوش فراخوانی و subscribe کردن به Observable ها که آبجکت هستند، از تغییرات رخ داده بلافاصله آگاه شده و آن تغییرات را در خود منعکس می نمایند. Observer ها به مجرد اینکه یک observable مقداری را emit (ارسال) می کند، از آن مطلع می شوند. همچنین زمانی که observable اطلاعاتی را مبنی بر اینکه دیگر مقادیری وجود ندارد ارسال می کند، observer ها از آن مطلع می شوند. زمانی هم که observable با خطایی مواجه می شود، متد onError(Throwable e) خورده شده و باز observer از آن آگاه می شود. توابع مربوطه onNext()، onError() و onCompleted() هستند که همگی از interface ای به نام Observer ارث بری می شوند. آبجکت یا نمونه ای از Subscription بیانگر اتصال و ارتباط بین یک observer و observable است. اگر متد unsubscribe() را بر روی این نمونه صدا بزنید، observable از عضویت متد بیرون آمده و اتصال بین آن ها قطع می شود. این متد می تواند زمانی مفید باشد که شما می خواهید بروز رسانی و آپدیت یک المان رابط کاربری یا widget خاتمه یافته (dispose) و از حافظه حذف شود.

4-12-5- تعریف thread اجرا و یک thread برای گوش دادن به تغییرات (observe)

می توانید thread ای که observable در آن قرار است اجرا شود را با فراخوانی متد subscribeOn() تعریف نمایید و thread ای که observer ها در آن اجرا می شوند را با اجرای متد observeOn() اعلان کنید.

4-13- Operator ها

می توانید عملیاتی را بر روی observer های خود ثبت کنید که به شما اجازه می دهد نتیجه و emission یک observable را پیش از ارسال آن به observer ویرایش و دستکاری نمایید. متد map به شما این امکان را می دهد یک Func1 ثبت کنید که ورودی را ترجمه و تبدیل می کند.

14-4-استفاده از delay و به تعویق انداختن تولید و ارسال نتیجه

با فراخوانی متد debounce(delay, TimeUnit.MILLISECONDS) بر روی observer، می توانید اعلان کنید که observer تنها زمانی تغییرات را ارسال یا به عنوان نتیجه ارسال (emit) نماید که مقدار مورد نظر پس از گذشت مدت زمان از پیش تعیین شده تغییر نکرده باشد.

14-4-1-ایجاد Observable ها و Observer ها

RxJava متدهای زیادی برای ایجاد observable در اختیار توسعه دهنده قرار می دهد. این متدها عبارت اند از:

- Observable.just() – یک Observable به عنوان ظرف ایجاد می کند که انواع داده ای را دربرمی گیرد. به عبارت دیگر این متد یک آبجکت را به observable ارسال می کند یا یک یا چند object را به یک Observable تبدیل می کند و سپس Observable آن آبجکت یا آبجکت ها را به observer می فرستد.
- Observable.from() – یک collection یا آرایه گرفته، مقادیر آن ها را به ترتیب موجود در ساختار داده ای مربوطه، emit یا به عنوان نتیجه ارسال می کند. در واقع این متد یک آرایه دریافت کرده، به ازای هر المان در ساختار داده ای مزبور، تابع observer را صدا می زند و مقدار را می فرستد.
- Observable.fromCallable() – به شما امکان ساخت یک observable برای Callable را می دهد.

به منظور ساخت observer ها:

- Action1 را پیاده سازی نمایید – به شما این امکان را می دهد تا observer ساده ایجاد کنید که یک متد به نام call دارد. در صورت تولید یا ارسال (emit) آبجکت جدید، این متد صدا خورده می شود.

مثالی از فراخوانی متد Observable.just()

تابع Observable.just() یک Observable ایجاد می کند و زمانی که یک observer به این Observable گوش می دهد یا برای آن subscribe می کند، متد onNext() از Observer با آرگومان ورودی Observable.just() فراخوانی می شود.

```
import java.util.Arrays;
import java.util.List;
import rx.Observable;
public class RxJavaExample {
      public static void main(String[] args) {
            List<String> list = Arrays.asList("Android", "Ubuntu", "Mac OS"); 1
            Observable<List<String>> listObservable = Observable.just(list);  2
            listObservable.subscribe(new Observer<List<String>>() {
                                                                  3
          @Override
          public void onCompleted() {}
          @Override
          public void onError(Throwable e) {}
          @Override
          public void onNext(List<String> list) {
                        System.out.println(list);
          }
      });
      }
}
```

1. یک لیست ایجاد می کند.

2. Observable را ایجاد می کند.

3. Observer را برای گوش دادن به Observable ثبت می کند.

4-15-انجام عملیات تبدیل

در زیر مثال ساده ای از RxJava را مشاهده می کنید که نوعی عملیات تبدیل در آن رخ داده است.

```
import java.util.Arrays;
import java.util.List;
import rx.Observable;
public class RxJavaExample {
	public static void main(String[] args) {
		List<String> list = Arrays.asList("Hello", "Streams", "Not");
		Observable.from(list).
				filter(s -> s.contains("e")).
				map(s -> s.toUpperCase()).
				reduce(new StringBuilder(), StringBuilder::append).
				subscribe(System.out::print, e -> {},
				() -> System.out.println("!"));
	}
}
```

Subject_4_14_2 ها

Subject ها در واقع یک نوع پل ارتباطی یا پیشکار (proxy) هستند که خود هر دو نقش Observable و Observer را یکجا ایفا می کنند. Subject ها را می توان به عنوان یک pipe یا پل ارتباطی که داده ها را تبدیل و ترجمه می کنند در نظر گرفته و مورد استفاده قرار داد. به عنوان مثال می توان به PublishSubject اشاره کرد. همین که داده ای به PublishSubject ارسال می شود، بلافاصله این داده به بیرون ارسال می گردد. به عبارت دیگر Subject یک آبجکت در RxJava هست که از تمامی ویژگی ها و property های Observable و Subscriber/Observer برخوردار می باشد. هم می تواند به Observable ها گوش داده و از آن ها داده تحویل بگیرد و هم قادر است داده ها را به Observer هایی که به آن گوش می دهند، داده به عنوان خروجی ارسال کند. همچنین می تواند داده هایی را با فراخوانی مستقیم متد onNext() ارسال کند. از این طریق، Subject می تواند نقش واسط را ایفا کند که داده ها را دریافت کرده، عملیاتی را بروی آن انجام می دهد و سپس آن را به گوش فراخوان یا subscriber های خود ارسال می کند.

4-15-آبجکت Single یا همان Promise (مکان نگهدار مقدار مورد نظر)

مدل برنامه نویسی reactive برای برقراری ارتباط دو طرفه و واکنشی از دو مفهوم Observer ها و Observable ها استفاده می کند.

امروزه استفاده از promise ها در فراخوانی های ناهمزمان (async) به ویژه همراه با زبان سمت سرویس گیرنده ی JavaScript، از محبوبیت ویژه و کاربرد فراوانی برخودار شده است. Promise در اصل جانگهدار یا placeholder برای یک مقدار دلخواه است که observer ها یا متدهای گوش فرا دهنده ی خود را به مجرد دریافت مقدار مورد انتظار و به اصطلاح موعود با خبر می سازد. RxJava نیز آبجکت یا نوعی به نام Single دارد که از لحاظ کاربرد بسیار شبیه به Promise است.

نمونه یا آبجکت هایی که از کلاس Single ساخته می شود در واقع وظیفه ای مشابه وظیفه ی آبجکت Observable ایفا می نماید، با این تفاوت که دو متد بازفراخوان بیشتر به نام های onSuccess() و onError() ندارد.

```
import io.reactivex.Single;
public Single<List<Todo>> getTodos() {
    return Single.create(emitter -> {
        Thread thread = new Thread(() -> {
            try {
                List<Todo> todosFromWeb = // query a webservice
                emitter.onSuccess(todosFromWeb);
            } catch (Exception e) {
                emitter.onError(e);
            }
        });
        thread.start();
    });
}
```

نحوه ی استفاده از نمونه ی کلاس Single در قطعه کد زیر به نمایش گذاشته شده است.

```
import io.reactivex.Single;
import io.reactivex.disposables.Disposable;
import io.reactivex.observers.DisposableSingleObserver;
Single<List<Todo>> todosSingle = getTodos();
todosSingle.subscribeWith(new DisposableSingleObserver<List<Todo>>() {
    @Override
    public void onSuccess(List<Todo> todos) {
        // work with the resulting todos
    }
    @Override
    public void onError(Throwable e) {
        // handle the error case
    }
});
```

16-4-دور انداختن subscription ها (قطع اتصال بین observer و observable) و استفاده از کلاس CompositeDisposable

گوش فراخوان ها یا subscriber هایی که به آبجکت مربوطه متصل بوده و از تغییرات آن آگاه می شوند، طبیعتا قرار نیست تا ابد به همین کار ادامه دهند. برای مثال هم ممکن است به خاطر تغییر کوچکی در وضعیت، event ارسالی از کلاس Observable دیگر مورد نیاز و درخور توجه نباشد.

```
import io.reactivex.Single;
import io.reactivex.disposables.Disposable;
import io.reactivex.observers.DisposableSingleObserver;
Single<List<Todo>> todosSingle = getTodos();
Disposable disposable = todosSingle.subscribeWith(new DisposableSingleObserver<List<Todo>>() {
    @Override
    public void onSuccess(List<Todo> todos) {
        // work with the resulting todos
    }
    @Override
    public void onError(Throwable e) {
        // handle the error case
    }
});
// continue working and dispose when value of the Single is not interesting any more
disposable.dispose();
```

توجه: کلاس Single و دیگر کلاس های observable توابع متعددی جهت subscribe و گوش فراخوانی به تغییرات ارائه می دهند که همگی در خروجی یک آبجکت Disposable بازگردانی می نمایند.

به هنگام کار با چندین subscription (نشانگر ارتباط دو طرفه بین observable و observer) که ممکن است به دلیل تغییر در وضعیت، رخداد ارسالی از آن ها دیگر لازم و جالب توجه نباشد، می توان با بهره گیری از کلاس CompositeDisposable همزمان چندین subscription را دور انداخته و به اصطلاح از حافظه پاک نمود (dispose).

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
import io.reactivex.Single;
import io.reactivex.disposables.Disposable;
import io.reactivex.observers.DisposableSingleObserver;
import io.reactivex.disposables.CompositeDisposable;
CompositeDisposable compositeDisposable = new CompositeDisposable();
Single<List<Todo>> todosSingle = getTodos();
Single<Happiness> happiness = getHappiness();
compositeDisposable.add(todosSingle.subscribeWith(new DisposableSingleObserver<List<Todo>>() {
    @Override
    public void onSuccess(List<Todo> todos) {
        // work with the resulting todos
    }
    @Override
    public void onError(Throwable e) {
        // handle the error case
    }
}));
compositeDisposable.add(happiness.subscribeWith(new DisposableSingleObserver<List<Todo>>() {
    @Override
    public void onSuccess(Happiness happiness) {
        // celebrate the happiness :-D
    }
    @Override
    public void onError(Throwable e) {
        System.err.println("Don't worry, be happy! :-P");
    }
}));
// continue working and dispose all subscriptions when the values from the Single objects are not interesting any more
compositeDisposable.dispose();
```

17-4-ذخیره ی موقتی مقدار Observable های ارسال شده (completed obervables)

هنگام کار با چند Observable و فراخوانی های ناهمزمان تمامی subscription های یک observable (observer هایی که برای گوش دادن به آن ثبت شده اند) ضروری نبوده و منابع زیادی را مصرف می کند.

Observable ها معمولا به بخش های مختلف اپلیکیشن پاس داده می شوند و هر بار لازم نیست با اضافه شدن یک subscription (و گوش دادن یک observer به observable) این فراخوانی سنگین اتفاق بیافتد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

کد زیر چهار با از یک وب سرویس کوئری می گیرد، هر چند یک بار کوئری گرفتن هم کفایت می کند. در واقع همان آبجکت های Todo نمایش داده می شوند اما هر بار به روش های مختلف.

```
Single<List<Todo>> todosSingle = Single.create(emitter -> {
    Thread thread = new Thread(() -> {
        try {
            List<Todo> todosFromWeb = // query a webservice
            System.out.println("Called 4 times!");
            emitter.onSuccess(todosFromWeb);
        } catch (Exception e) {
            emitter.onError(e);
        }
    });
    thread.start();
});
todosSingle.subscribe(... " Show todos times in a bar chart " ...);
showTodosInATable(todosSingle);
todosSingle.subscribe(... " Show todos in gant diagram " ...);
anotherMethodThatsSupposedToSubscribeTheSameSingle(todosSingle);
```

تکه کد بعدی از متد cache استفاده می کند، به همین خاطر آبجکت Single، پس از اینکه ارسال برای بار اول با موفقیت انجام شد، در حافظه ی موقت ذخیره می شود و از وب سرویس تنها یک بار کوئری گرفته می شود.

```
Single<List<Todo>> todosSingle = Single.create(emitter -> {
    Thread thread = new Thread(() -> {
        try {
            List<Todo> todosFromWeb = // query a webservice
            System.out.println("I am only called once!");
            emitter.onSuccess(todosFromWeb);
        } catch (Exception e) {
            emitter.onError(e);
        }
    });
    thread.start();
});
// cache the result of the single, so that the web query is only done once
Single<List<Todo>> cachedSingle = todosSingle.cache();
cachedSingle.subscribe(... " Show todos times in a bar chart " ...);
showTodosInATable(cachedSingle);
cachedSingle.subscribe(... " Show todos in gant diagram " ...);
anotherMethodThatsSupposedToSubscribeTheSameSingle(cachedSingle);
```

4-18-Flowable<T> و Backpressure

در RxJava ممکن است به کرات با شرایطی مواجه شوید که در آن تعداد آیتم های ارسالی از یک Observable بیشتر از میزانی است که یک Observer قادر به استفاده از آن هست. در چنین شرایطی این مسئله مطرح می شود که چگونه می توان این انباشت آیتم های استفاده نشده را مدیریت نمود. به این رخداد در اصطلاح برنامه نویسی back pressure گفته می شود.

RxJava 2.0 یک نوع جدید به نام Flowable<T> را معرفی کرد که از لحاظ عملکرد بسیار شبیه به Observable<T> می باشد با این تفاوت که Flowable<T> قابلیت مدیریت back pressure دارد و Observable<T> همچنین امکانی را فراهم نمی کند.

در RxJava 1.0 این مفهوم کمی دیر به نوع Observable<T> اضافه شد. اما یک مشکل وجود داشت: برخی خطای زمان اجرای MissingBackpressureException را می دادند. به همین خاطر Flowable<T> اضافه شده و این مشکل را برطرف نمود.

جدا از نوع Observable<T>، نوع هایی همچون Maybe<T>، Single<T> و Completable<T> نیز back pressure را اصلا پشتیبانی نمی کنند.

4-19-تبدیل نوعی به نوع دیگر

می توان به راحتی یک نوع از RxJava را به نوع دیگر تبدیل نمود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Table 1. Conversion between types					
From / To	Flowable	Observable	Maybe	Single	Completable
Flowable		toObservable()	reduce() elementAt() firstElement() lastElement() singleElement()	scan() elementAt() first()/firstOrError() last()/lastOrError() single()/singleOrError() all()/any()/count() (and more...)	ignoreElements()
Observable	toFlowable()		reduce() elementAt() firstElement() lastElement() singleElement()	scan() elementAt() first()/firstOrError() last()/lastOrError() single()/singleOrError() all()/any()/count() (and more...)	ignoreElements()
Maybe	toFlowable()	toObservable()		toSingle() sequenceEqual()	toCompletable()
Single	toFlowable()	toObservable()	toMaybe()		toCompletable()
Completable	toFlowable()	toObservable()	toMaybe()	toSingle() toSingleDefault()	

20-4-تست Observable و Subscription های RxJava

1-20-4-تست observable ها

کتابخانه ی RxJava یک کلاس به نام TestSubscriber در اختیار توسعه دهنده قرار می دهد که با استفاده از آن می توان یک observable را تست نمایید.

```
Observable<String> obs = ...// assume creation code here
TestSubscriber<String> testSubscriber = new TestSubscriber<>();
obs.subscribe(testSubscriber);
testSubscriber.assertNoErrors();
List<String> chickens = testSubscriber.getOnNextEvents();
// TODO assert your string integrity...
```

RxJava مکانیزمی را در اختیار شما قرار می دهد که به واسطه ی آن می توانید scheduler های ارائه (exposed) شده را طوری بازنویسی نمایید که به صورت همزمان (sync) فراخوانی شوند. برای مثالی در این زمینه می توانید به آدرس http://fedepaol.github.io/blog/2015/09/13/testing-rxjava-observables-subscriptions/ مراجعه نمایید.

بخش هفتم :

آموزش پیاده سازی پردازش پس زمینه ای و ناهمگام با Handler ، AsyncTask و Loader

این آموزش به شرح مفاهیم Thread ها، Handler ها و استفاده از کلاس AsyncTask می پردازد. همچنین مفهوم پردازش ناهمگام در اپلیکیشن های اندرویدی را تشریح می نماید. سپس نحوه ی مدیریت چرخه ی حیات/lifecycle با thread ها و پردازش پس زمینه ای را تحت پوشش قرار می دهد.

پروژه های این مبحث در محیط برنامه نویسی Android Studio نوشته و تست می شوند.

4-21-پردازش پس زمینه ای در اندروید

1-21-4-چرا استفاده از مفهوم همروندی/concurrency توصیه می شود؟

در حالت پیش فرض، یک اپلیکیشن اندرویدی در thread اصلی اجرا می شود. از اینرو تمامی دستورات به صورت پشت سرهم (خط به خط) و طبق یک ترتیب خاص اجرا می شوند. در چنین وضعیتی، اگر اپلیکیشن یک فرایند طولانی را راه اندازی کند، اپلیکیشن تا زمانی که عملیات مزبور به اتمام نرسیده، (thread اصلی مسدود شده) قادر به پردازش درخواست دیگری نخواهد بود.

به منظور ارائه ی تجربه ی کاربری بهینه، تمامی عملیات کند و طولانی می بایست در اپلیکیشن اندرویدی به صورت ناهمزمان (async) اجرا شوند. برای نیل به این هدف می توانید از ساختارهای مدیریت همروندی (concurrency) جاوا یا چارچوب نرم افزاری (framework) اندروید استفاده نمایید. از عملیات کند می توان به پردازش های مربوط به اینترنت، فایل و دسترسی به داده از دیتابیس و محاسبات پیچیده اشاره کرد.

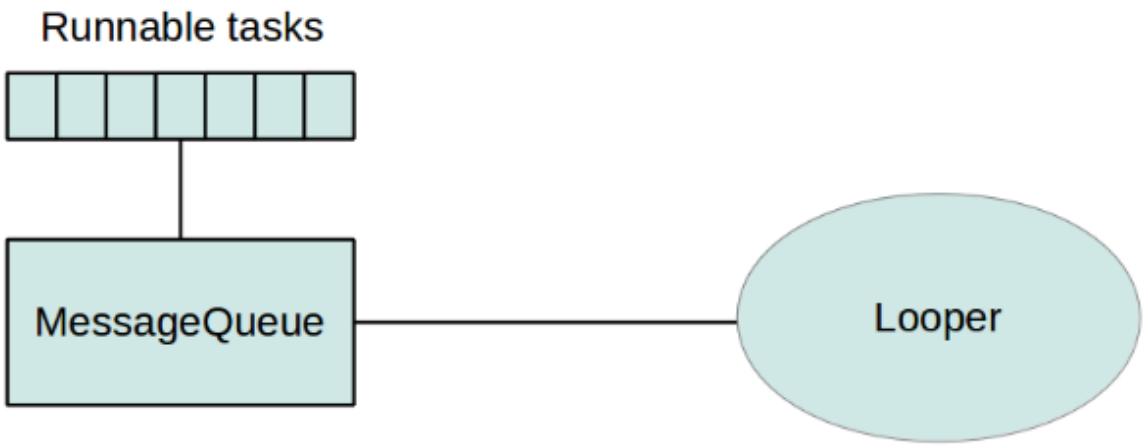
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

نکته: در اندروید چنانچه یک activity قادر نباشد با گذشت مدت 5 ثانیه از درخواست کاربر، پاسخ مناسب را ارائه دهد، سیستم بلافاصله یک پنجره محاوره ی حاوی پیغام Application not responding (ANR)را به نمایش می گذارد. در این پنجره کاربر می تواند با انتخاب گزینه ی مربوطه، اپلیکیشن را کاملا متوقف کند.

Thread-4-21-2 اصلی یا UI thread

اندروید تمامی تسک ها و رخدادهای ورودی را در یک thread واحد به نام UI thread یا main thread مدیریت می نماید. در واقع سیستم اندروید تمامی رخدادها را در این thread، داخل یک صف جمع کرده و سپس event های قرار گرفته در این صف را با کمک نمونه ای از کلاس Looper پردازش می نماید. لازم به ذکر است که thread اصلی نمی تواند چندین عملیات را اداره کرده و در لحظه می تواند تنها یک رخداد ورودی را پردازش کند.



3-21-4-پردازش ناهمزمان و استفاده از thread ها در اندروید

اندروید برای پردازش async و ناهمزمان از کلاس Thread و جهت پردازش همزمان چند عملیات در پس زمینه پکیج java.util.concurrent را در اختیار توسعه دهنده قرار می دهد. این پکیج کلاس های ThreadPools و Executor را برای پردازش پس زمینه فراهم می نماید.

در صورت نیاز به آپدیت UI از یک Thread دیگر، لازم است با main thread همزمانی انجام دهید. به خاطر وجود این محدودیت ها، توسعه دهندگان اندروید اغلب از کدها و ساختارهای اختصاصی خود اندروید بهره می گیرند.

اندروید تعدادی ساختار ویژه جهت مدیریت همروندی (انجام چند پردازش در پس زمینه) ارائه می دهد که جدا از کدهای جاوا است.

از ساختارهای اختصاصی خود اندروید می توان از کلاس های android.os.Handler یا AsyncTasks نام برد. روش های پیچیده تری ویژه ی مدیریت همروندی وجود دارد که لازمه ی آن، استفاده از کلاس Loader، retained fragment ها و سرویس ها است.

توجه: تا حد امکان از بکار بردن پنجره ی ProgressBar یا روش های مشابه که تعامل با UI را تا اتمام فرایند معینی مسدود می کند، خودداری نمایید. توصیه می شود برای این منظور یک توضیح مختصر UI همچنان قابلیت تعامل با کاربر را داشته باشد.

4-21-4-ارائه ی feedback و توضیح مختصر برای کاربر در طول اجرای عملیات طولانی

در طول اجرای عملیات طولانی، بهتر است توضیح مختصری در خصوص این عملیات برای کاربر به نمایش بگذارید.

می توانید این توضیحات را در قالب action bar یا action view در UI به نمایش بگذارید. روش دیگر این است که یک ProgressBar در layout تعریف کرده، آن را بر روی visible تنظیم نمایید و سپس در طول اجرای عملیات بروز رسانی کنید. هرچند توصیه می شود از روش اول استفاده

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

نمایید چرا که در آن صورت ارتباط کاربر با UI قطع نمی شود و اپلیکیشن همچنان responsive می ماند.

4-22-کلاس Handler

4-22-1-استفاده از کلاس Handler جهت مدیریت همروندی

ابتدا جهت پیاده سازی مفهوم همروندی در موبایل و ایجاد یک thread دیگر سوای thread اصلی، می بایست یک آبجکت از کلاس Thread ایجاد نمایید. سپس می توانید با کمک کلاس های handler و Message خروجی thread پس زمینه را به thread اصلی اطلاع دهید.

آبجکت Handler خود را داخل thread ای که در آن ایجاد می شود، ثبت می نماید. این آبجکت یک پل ارتباطی ایجاد کرده و داده ها را از طریق آن به thread اصلی تحویل می دهد. به طور مثال، اگر شما یک نمونه ی جدید از Handler در متد onCreate() کلاس activity خود ایجاد نمایید، در آن صورت می توانید از آبجکت ذکر شده به راحتی داده های مورد نیاز را به thread اصلی ارسال نمایید. داده هایی که کلاس Handler ارسال می کند، می تواند آبجکتی از کلاس Message یا Runnable باشد.

Handler به ویژه در شرایطی کارامد تلقی می شود که لازم باشد داده هایی را چند بار به thread اصلی ارسال (post) نمایید.

4-22-2-ایجاد و استفاده ی مجدد از thread اصلی

به منظور پیاده سازی کلاس handler، می بایست یک کلاس فرزند (subclass) از آن ایجاد کرده و متد handleMessage() آن را جهت پردازش پیغام ها بازنویسی (override) نمایید. سپس می توانید به راحتی با فراخوانی توابع sendMessage(Message) یا sendEmptyMessage() پیغام های مورد نیاز را به آن ارسال کنید. جهت ارسال یک آبجکت Runnable به کلاس مذکور، لازم است متد post() را بکار ببرید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

به منظور جلوگیری از ایجاد غیر ضروری آبجکت و اشغال حافظه، می توانید آبجکت Handler جاری کلاس activity خود را مجددا استفاده یا به اصطلاح بازیافت نمایید.

```
// Reuse existing handler if you don't
// have to override the message processing
handler = getWindow().getDecorView().getHandler();
```

کلاس View به شما این امکان را می دهد تا آبجکت هایی از جنس Runnable را به وسیله ی متد post() ارسال (post) نمایید (این متد به شما امکان می دهد تا پیغام پایان کار را به thread اصلی اعلان نمایید).

مثال

کد زیر نحوه ی استفاده از یک handler از view را نمایش می دهد. فرض کنید activity شما از فایل layout زیر جهت تنظیم ظاهر خود استفاده می کند.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
  <ProgressBar
    android:id="@+id/progressBar1"
    style="?android:attr/progressBarStyleHorizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:indeterminate="false"
    android:max="10"
    android:padding="4dip" >
  </ProgressBar>
     <TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="" >
   </TextView>
  <Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="startProgress"
    android:text="Start Progress" >
```

```
    </Button>
</LinearLayout>
```

با توجه به کد activity زیر که برای برنامه نوشته شده، زمانی که کاربر بر روی دکمه یا آبجکت Button کلیک می کند، آبجکت ProgressBar بلافاصله بروز رسانی می شود.

```
package de.vogella.android.handler;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.ProgressBar;
import android.widget.TextView;
public class ProgressTestActivity extends Activity {
      private ProgressBar progress;
      private TextView text;
      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main);
            progress = (ProgressBar) findViewById(R.id.progressBar1);
            text = (TextView) findViewById(R.id.textView1);
      }
      public void startProgress(View view) {
            // do something long
            Runnable runnable = new Runnable() {
                  @Override
                  public void run() {
                        for (int i = 0; i <= 10; i++) {
                              final int value = i;
                               doFakeWork();
                              progress.post(new Runnable() {
                                    @Override
                                    public void run() {
                                          text.setText("Updating");
                                          progress.setProgress(value);
                                    }
                              });
                        }
                  }
            };
            new Thread(runnable).start();
      }
      // Simulating something timeconsuming
      private void doFakeWork() {
            try {
                  Thread.sleep(2000);
            } catch (InterruptedException e) {
                  e.printStackTrace();
            }
```

```
    }
}
```

4-23-کلاس AsyncTask

4-23-1-هدف استفاده از کلاس AsyncTask

کلاس AsyncTask به شما اجازه می دهد تا دستوراتی را در پس زمینه به اجرا بگذارید و سپس آن ها را با thread اصلی همگام نموده و همچنین گزارشاتی مربوط به تسک های در حال اجرا به thread اصلی تحویل دهید. لازم به ذکر است که از کلاس AsyncTasks اغلب برای اجرای عملیات در پس زمینه که UI را بروز رسانی می کنند، استفاده می شود.

در واقع کلاس AsyncTask یک کلاس انتزاعی و abstract است که به برنامه های اندرویدی امکان می دهد تا thread اصلی (UI) را به صورت بهینه مدیریت نمایید. AsyncTask به توسعه دهنده کمک می کند تا تسک هایی که چند ثانیه بیشتر طول نمی کشند را در پس زمینه به طور ناهمزمان اجرا نموده و نتیجه ی آن را در thread اصلی نمایش دهد.

4-23-2-استفاده ی کاربردی از کلاس AsyncTask

برای استفاده از کلاس AsyncTask لازم است از آن یک کلاس فرزند ایجاد نمایید (از AsyncTask یک کلاس مشتق نمایید). کلاس AsyncTask از generic و varargs استفاده می کند. پارامترهای ورودی عبارتند از: AsyncTask <TypeOfVarArgParams, ProgressValue, ResultValue>.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

کلاس AsyncTask با صدا خوردن متد ()execute راه اندازی می شود. متد ()execute خود به تبع متدهای ()doInBackground و ()onPostExecute را فراخوانی می کند.

TypeOfVarArgParams به عنوان آرگومان ورودی به تابع ()doInBackground فرستاده می شود. ProgressValue ویژه ی اطلاعات پیشرفت فرایند بکار برده می شود و ResultValue باید به عنوان خروجی از متد ()doInBackground بازگردانی شود. این آرگومان خود به عنوان پارامتر ورودی به تابع ()onPostExecute پاس داده می شود.

متد ()doInBackground دربردارنده ی دستورات و کدهایی است که می بایست در یک thread مجزا در پس زمینه اجرا شوند.

متد ()onPostExecute خود را با Thread اصلی (UI) همگام ساخته و بروز رسانی اطلاعات آن را به دنبال دارد. زمانی که عملیات متد ()doInBackground به پایان می رسد، خود چارچوب نرم افزاری اندروید (FRAMEWORK) این متد را فراخوانی می کند.

3-23-4-اجرای همزمان چندین AsyncTasks

سیستم اندروید قبل از ویرایش 1.6 و بار دیگر از ورژن 3.0، تسک های مربوط به AsyncTask را به صورت پیش فرض به ترتیب (و پشت سرهم) اجرا می نماید. شما می توانید به سیستم اندروید اعلان نمایید که این تسک ها را به طور موازی با اجرای متد ()executeOnExecutor، در حالی که AsyncTask.THREAD_POOL_EXECUTOR به عنوان اولین پارامتر به آن ارسال شده، اجرا نماید.

تکه کد زیر این عملیات را نمایش می گذارد.

```
// ImageLoader extends AsyncTask
ImageLoader imageLoader = new ImageLoader( imageView );
// Execute in parallel
imageLoader.executeOnExecutor( AsyncTask.THREAD_POOL_EXECUTOR, "http://url.com/image.png" );
```

نکته: AsyncTask تغییرات در نحوه ی پیکربندی و تنظیمات را به صورت خودکار مدیریت نمی کند، بدین معنی که چنانچه activity از نو ساخته شود، برنامه نویس باید آن را با نوشتن کد مربوطه

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

مدیریت نماید. یک راه حل پرطرفدار و مورد استفاده ی اغلب توسعه دهندگان این است که کلاس AsyncTask را در یک retained headless fragment (fragment های فاقد UI که اطلاعات خود را بین تغییرات در نحوه ی پیکربندی حفظ می کنند) تعریف نمایید.

مثال: پیاده سازی AsyncTask

کد زیر چگونگی استفاده از کلاس AsyncTask جهت دانلود محتوای یک صفحه ی وب را به نمایش می گذارد.

یک پروژه ی جدید اندروید به نام de.vogella.android.asynctask و یک activity به نام ReadWebpageAsyncTask ایجاد نمایید. مجوز android.permission.INTERNET را در فایل تنظیمات اپلیکیشن (manifest) تنظیم نمایید.

Layout زیر را ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
  <Button
    android:id="@+id/readWebpage"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:onClick="onClick"
    android:text="Load Webpage" >
  </Button>
  <TextView
    android:id="@+id/TextView01"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:text="Placeholder" >
  </TextView>
</LinearLayout>
```

بدنه ی activity خود را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.asynctask;
// imports cut out for brevity
public class ReadWebpageAsyncTask extends Activity {
      private TextView textView;
      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main);
```

```
        textView = (TextView) findViewById(R.id.TextView01);
    }
    private class DownloadWebPageTask extends AsyncTask<String, Void, String> {
        @Override
        protected String doInBackground(String... urls) {
            // we use the OkHttp library from https://github.com/square/okhttp
            OkHttpClient client = new OkHttpClient();
            Request request =
                    new Request.Builder()
                    .url(urls[0])
                    .build();
                Response response = client.newCall(request).execute();
                if (response.isSuccessful()) {
                    return response.body().string();
                }
            }
            return "Download failed";
        }
        @Override
        protected void onPostExecute(String result) {
            textView.setText(result);
        }
    }
    public void onClick(View view) {
        DownloadWebPageTask task = new DownloadWebPageTask();
        task.execute(new String[] { "http://www.vogella.com/index.html" });

    }
}
```

اپلیکیشن خود را اجرا نموده و دکمه ی مربوطه را کلیک نمایید. اطلاعات صفحه وب موردنظر در پس زمینه خوانده می شود. به مجرد اتمام این فرایند، می بینید که آبجکت TextView با اطلاعات جدید بروز آوری می شود.

4-24-پردازش پس زمینه ای و مدیریت چرخه ی حیات (lifecycle)

1-24-4-حفظ اطلاعات مربوط به وضعیت (state) بین تغییراتی که در نحوه ی پیکربندی رخ می دهد

یکی از مشکلات و چالش های استفاده از thread، در نظر گرفتن و مدیریت چرخه ی حیات اپلیکیشن می باشد. سیستم اندروید ممکن است activity را به کلی از بین برده، از حافظه پاک کند یا با اعمال تغییری در نحوه ی پیکربندی سبب شود که activity مجددا ایجاد گردد.

بعلاوه توسعه دهنده می بایست مدیریت محاوره های (dialog) باز را در نظر بگیرد چرا که این محاوره ها در نهایت به activity میزبان وصل هستند. در صورتی که activity مجددا راه اندازی شده و شما قصد دسترسی به محاوره ی جاری را بکنید، خواهید دید که یک خطای زمان اجرا View not attached to window manager از سمت سیستم صادر می شود.

به منظور ذخیره ی یک آبجکت، شما می توانید متد onRetainNonConfigurationInstance() را مورد استفاده قرار دهید. در واقع چنانچه قرار است activity مورد نظر از بین رفته و مجددا راه اندازی شود، شما می توانید با فراخوانی متد مزبور اطلاعات نمونه ی جاری را در یک آبجکت نگه داشته و دوباره با ساخت activity، داده های ذخیره شده در آبجکت را به آن تحویل دهید. برای بازیابی این آبجکت می توانید متد getLastNonConfigurationInstance() را فراخوانی کنید. از این طریق شما قادر خواهید بود، حتی زمانی که activity جاری از بین رفته و مجددا راه اندازی می شود، آبجکت حامل اطلاعات activity (برای مثال یک thread در حال اجرا) را نگه داشته و محتوای آن را به activity از نو ساخته شده پاس دهید.

در صورتی که activity برای اولین بار راه اندازی شده و یا با صدا خورده شدن متد finish() خاتمه می یابد، متد مذکور در خروجی مقدار null را برخواهد گرداند.

البته لازم به توضیح است که متد onRetainNonConfigurationInstance() از ورژن 13 کتابخانه های اندروید (API 13) تقریبا منسوخ تلقی می شود. بهتر است برای بدست آوردن نتیجه ی مورد نظر از fragment ها استفاده نموده و متد setRetainInstance() را جهت نگه داشتن اطلاعات بین تغییرات در نحوه ی پیکربندی بکار ببرید.

2-24-4-استفاده از آبجکت application جهت ذخیره اطلاعات چندین آبجکت

چنانچه لازم است چندین آبجکت را جهت نگهداری اطلاعات activity ها و بازگردانی آن ها پس از تغییرات در نحوه ی پیکربندی بکار ببرید، در آن صورت می توانید کلاس Application که اطلاعات

مربوط به وضعیت سراسری اپلیکیشن را در خود ذخیره نگه می دارد را در برنامه ی خود پیاده سازی نمایید.

برای نیل به این هدف، کافی است مقدار classname را به خصیصه android:name (attribute) در فایل تنظیمات اپلیکیشن خود تخصیص دهید.

```
<application android:icon="@drawable/icon" android:label="@string/app_name"
      android:name="MyApplicationClass">
       <activity android:name=".ThreadsLifecycleActivity"
            android:label="@string/app_name">
             <intent-filter>
                  <action android:name="android.intent.action.MAIN" />
                  <category android:name="android.intent.category.LAUNCHER" />
             </intent-filter>
       </activity>
</application>
```

لایه ی runtime اندروید کلاس application را به صورت خودکار ایجاد کرده و تا زمانی که کل فرایند اپلیکیشن متوقف نشده، این آبجکت همچنان در دسترس باقی خواهند ماند.

آبجکت اپلیکیشن می تواند اطلاعات مربوط به وضعیت سراسری اپلیکیشن را نگه دارد. این آبجکت قبل از اینکه فرایند مربوط به اپلیکیشن یا پکیج ایجاد شود، نمونه سازی شده و در دسترس قرار می گیرد.

از کلاس نام برده می توان جهت نگهداری و بازگردانی اطلاعات آبجکت هایی که در چندین activity هستند یا برای چرخه ی حیات کلی اپلیکیشن قابل دسترسی می باشد، استفاده نمایید. داخل بدنه ی متد onCreate() می توانید آبجکت هایی ایجاد کرده و آن ها را از طریق فیلدهای public یا توابع getter در دسترس قرار دهید.

متد onTerminate() در کلاس اپلیکیشن فقط و فقط به منظور تست بکار می رود. همان طور که از اسمش پیدا است، این متد فرایند تخصیص یافته به اپلیکیشن را به طور کلی از بین می برد. بنابراین چنانچه این متد فراخوانی شده و فرایند میزبان خاتمه یابد، در آن صورت تمامی منابع اختصاص یافته به اپلیکیشن به صورت خودکار آزاد می شوند و اطلاعات از حافظه ی دستگاه حذف می گردد.

جهت دسترسی به آبجکت Application می توانید متد getApplication() را در سطح activity فراخوانی نمایید.

4-25-Fragment ها و پردازش موازی در پس زمینه (background processing)

4-25-1-نگهداری اطلاعات آبجکت جهت بازگردانی آن پس از تغییر در نحوه ی پیکربندی

می توانید از fragment های بدون UI استفاده نموده و اطلاعات آن ها را با فراخوانی متد setRetainInstance() ذخیره نگه دارید و پس از تغییر در تنظیمات اپلیکیشن آن ها را بازگردانی کنید.

از این طریق آبجکت Thread یا AsyncTask با تغییر در تنظیمات و نحوه ی پیکربندی اپلیکیشن در حافظه نگه داشته می شود و شما می توانید بدون اینکه نگران مدیریت چرخه ی حیات اپلیکیشن خود بوده و آن را به صورت صریح درنظر بگیرید، عملیاتی را در پس زمنیه و در موازات هم به اجرا بگذارید.

4-25-2-Fragment های فاقد UI (headless fragments)

اگر پردازش هایی را در پس زمینه به اجرا گذاشته اید، در آن صورت می توانید در زمان اجرا (به صورت dynamic) یک headless fragment به اپلیکیشن خود متصل کرده و setRetainInstance() را بر روی true تنظیم نمایید. به دنبال این کار، fragment (اطلاعات آبجکت) با تغییر در نحوه ی پیکربندی حفظ شده و شما می توانید پردازش ناهمزمان در آن اجرا کنید.

4-26-کلاس Loader

1-26-4-هدف از کاربرد کلاس Loader

کلاس Loader به شما این امکان را می دهد تا داده های مورد نیاز را به صورت ناهمزمان (async) در activity یا fragment مربوطه بارگذاری نمایید. این کلاس می تواند منبع داده ها را رصد کرده (زیر نظر داشته) و زمانی که محتویات تغییر کردند، نتایج جدید را تحویل دهد. همچنین این امکان را فراهم می کند تا داده ها با تغییر در نحوه ی پیکربندی ذخیره شده و در زمان لازم (پس از ساخته شدن fragment) بازگردانی شوند.

داده ها را می توان به وسیله ی Loader در حافظه ی نهان نگه داشت (cache کرد) و زمانی که تغییری در نحوه ی پیکربندی رخ می دهد، داده های ذخیره شده را بازگردانی نمود. کلاس های Loader از اندروید 3.0 معرفی شده و بخشی از compatibility layer (جهت پشتیبانی از نسخه های قدیمی) برای ورژن 1.6 به بعد اندروید محسوب می شود.

2-26-4-پیاده سازی کلاس Loader

می توانید از کلاس AsyncTaskLoader به عنوان یک پایه برای پیاده سازی های Loader خود استفاده نمایید. LoaderManager یک activity یا fragment می تواند چندین نمونه از Loader را مدیریت کند. نحوه ی فراخوانی Loader در زیر به نمایش گذاشته شده است.

```
# start a new loader or re-connect to existing one
getLoaderManager().initLoader(0, null, this);
```

اولین پارامتر یک ID یا شناسه ی منحصربفرد است که کلاس callback برای شناسایی Loader مورد استفاده قرار می دهد. دومین پارامتر ارسالی یک آبجکت bundle است که برای اطلاعات بیشتر به متد پاس داده می شود. سومین پارامتر ورودی متد ()initLoader، کلاسی است که به محض شروع مقداردهی اولیه (کلاس callback) فراخوانی می شود. این کلاس می بایست اینترفیس LoaderManager.LoaderCallbacks را پیاده سازی کند. در واقع مرسوم است activity یا fragment ای که از Loader استفاده می کند، همراه با آن اینترفیس LoaderManager.LoaderCallbacks را نیز پیاده سازی نماید.

Loader به طور مستقیم با فراخوانی متد getLoaderManager().initLoader() ایجاد نمی شود، بلکه کلاس callback یا بازفراخوان آن را در بدنه ی تابع onCreateLoader() خود ایجاد می کند.

هنگامی که Loader خواندن داده ها به صورت ناهمزمان را به اتمام می رساند، متد onLoadFinished() از کلاس callback صدا خورده می شود. در اینجا (داخل بدنه ی این متد) شما می تواند UI و ظاهر اپلیکیشن خود را با داده های جدید بروز آوری نمایید.

3-24-4-دیتابیس SQLite و پیاده سازی کلاس CursorLoader

چارچوب نرم افزاری اندروید (framework) یک کلاس به نام CursorLoader در اختیار توسعه دهنده قرار می دهد که پیاده سازی Loader را به صورت پیش فرض دربرداشته و اتصال به دیتابیس SQLite (database connection) را خود مدیریت می نماید.

برای کوئری گرفتن و درخواست داده از یک ContentProvider که مبتنی بر دیتابیس SQLite می باشد، برنامه نویسان اغلب از کلاس CursorLoader استفاده می کنند. این کلاس Loader در پس زمینه (background thread) از دیتابیس کوئری می گیرد، به همین جهت تعامل اپلیکیشن با کاربر مختل نشده و در نتیجه UI مسدود نمی شود.

کلاس CursorLoader جایگزین آبجکت های Activity-managed cursor که در ویرایش های قبلی اندروید بکار می رفتند، محسوب می شود.

در صورتی که Cursor نامعتبر شود، متد onLoadReset() در کلاس callback فراخوانی می گردد.

تمرین: پیاده سازی Loader اختصاصی برای مدیریت و بارگذاری ناهمگام جفت های کلید-مقدار (preferences)

4-24-4-پیاده سازی کلاس

در تمرین زیر یک کلاس loader اختصاصی برای بارگذاری و مدیریت جفت های کلید-مقدار (preferences) پیاده سازی خواهد کرد. با هر بار بارگذاری (load)، مقدار preference افزایش می یابد.

یک پروژه ی جدید به نام com.Vogella.android.loader.preferences و همچنین یک activity جدید به نام MainActivity ایجاد نمایید.

با تعریف کلاس زیر که یک پیاده سازی اختصاصی از AsyncTaskLoader هست شما در واقع جفت های کلید-مقدار (shared preferences) را به صورت ناهمگام مدیریت می نمایید (shared preferences برای ذخیره ی کلید/مقدارهای کوچک بکار می رود تا برای مثال با بسته شدن اپلیکیشن یا خاموش شدن دستگاه اطلاعات مربوطه همچون تنظیمات اپلیکیشن از دست نرود).

```
package com.vogella.android.loader.preferences;
import android.content.AsyncTaskLoader;
import android.content.Context;
import android.content.SharedPreferences;
import android.preference.PreferenceManager;
public class SharedPreferencesLoader extends AsyncTaskLoader<SharedPreferences>
        implements SharedPreferences.OnSharedPreferenceChangeListener {
    private SharedPreferences prefs = null;
    public static void persist(final SharedPreferences.Editor editor) {
        editor.apply();
    }
    public SharedPreferencesLoader(Context context) {
        super(context);
    }
    // Load the data asynchronously
    @Override
    public SharedPreferences loadInBackground() {
        prefs = PreferenceManager.getDefaultSharedPreferences(getContext());
        prefs.registerOnSharedPreferenceChangeListener(this);
        return (prefs);
    }
    @Override
    public void onSharedPreferenceChanged(SharedPreferences sharedPreferences,
            String key) {
        // notify loader that content has changed
        onContentChanged();
    }
```

```
    /**
     * starts the loading of the data
     * once result is ready the onLoadFinished method is called
     * in the main thread. It loader was started earlier the result
     * is return directly
     * method must be called from main thread.
     */
    @Override
    protected void onStartLoading() {
        if (prefs != null) {
            deliverResult(prefs);
        }
        if (takeContentChanged() || prefs == null) {
            forceLoad();
        }
    }
}
```

نمونه کد زیر استفاده از این loader را در کلاس activity به نمایش می گذارد.

```
package com.vogella.android.loader.preferences;
import android.annotation.SuppressLint;
import android.app.Activity;
import android.app.LoaderManager;
import android.content.Loader;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.widget.TextView;
public class MainActivity extends Activity implements
        LoaderManager.LoaderCallbacks<SharedPreferences> {
    private static final String KEY = "prefs";
    private TextView textView;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        textView = (TextView) findViewById(R.id.prefs);
        getLoaderManager().initLoader(0, null, this);
    }
    @Override
    public Loader<SharedPreferences> onCreateLoader(int id, Bundle args) {
        return (new SharedPreferencesLoader(this));
    }
    @SuppressLint("CommitPrefEdits")
    @Override
    public void onLoadFinished(Loader<SharedPreferences> loader,
            SharedPreferences prefs) {
        int value = prefs.getInt(KEY, 0);
        value += 1;
        textView.setText(String.valueOf(value));
        // update value
```

```
        SharedPreferences.Editor editor = prefs.edit();
        editor.putInt(KEY, value);
        SharedPreferencesLoader.persist(editor);
    }
    @Override
    public void onLoaderReset(Loader<SharedPreferences> loader) {
        // NOT used
    }
}
```

تست اپلیکیشن

پس از هر بار تغییر در نحوه ی پیکربندی/تنظیمات اپلیکیشن، کلاس LoaderManager متد onLoadFinished() را به صورت خودکار در activity صدا می زند. اپلیکیشن را اجرا نموده و مطمئن شوید که با هر بار تغییر در نحوه ی پیکربندی (config change) مقدار ذخیره شده در shared preferences افزایش می یابد.

4-25-استفاده از service ها

می توانید از سرویس های اندروید برای اجرای عملیات و تسک های پس زمینه ای بهره بگیرید. برای این منظور می توانید به مبحث آموزش سرویس ها در اندروید مراجعه نمایید.

تمرین: چرخه ی حیات activity و thread ها

نمونه برنامه ی زیر یک فایل تصویری را در thread پس زمینه از اینترنت دانلود کرده و یک پنجره ی محاوره به نمایش می گذارد که تا اتمام پروسه ی دانلود در UI باقی می ماند. برنامه را طوری خواهید نوشت که با از بین رفتن و ساخت مجدد activity نیز thread حفظ شده و پنجره ی محاوره به درستی نمایش داده/بسته شود.

برای این مثال یک پروژه و activity جدید به ترتیب به نام های de.vogella.android.threadslifecycle و ThreadsLifecycleActivity ایجاد نمایید. همچنین لازم است مجوز استفاده از اینترنت را نیز برای اپلیکیشن در فایل manifest تنظیم نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

محتویات فایل تنظیمات اپلیکیشن (AndroidManifest.xml) شما می بایست ظاهری مشابه زیر داشته باشد.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="de.vogella.android.threadslifecycle"
  android:versionCode="1"
  android:versionName="1.0" >
  <uses-sdk android:minSdkVersion="10" />
  <uses-permission android:name="android.permission.INTERNET" >
  </uses-permission>
  <application
    android:icon="@drawable/icon"
    android:label="@string/app_name" >
    <activity
      android:name=".ThreadsLifecycleActivity"
      android:label="@string/app_name" >
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>
</manifest>
```

محتوای فایل main.xml را به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
  <LinearLayout
    android:id="@+id/linearLayout1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
    <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:onClick="downloadPicture"
      android:text="Click to start download" >
    </Button>
    <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:onClick="resetPicture"
      android:text="Reset Picture" >
    </Button>
  </LinearLayout>
  <ImageView
    android:id="@+id/imageView1"
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
    android:src="@drawable/icon" >
  </ImageView>
</LinearLayout>
```

حال بدنه ی کلاس activity را ویرایش نمایید. داخل این کلاس thread ذخیره شده و محاوره به محض از بین رفتن activity، بسته می شود.

```
package de.vogella.android.threadslifecycle;
import java.io.IOException;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.StatusLine;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpUriRequest;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.util.EntityUtils;
import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.view.View;
import android.widget.ImageView;
public class ThreadsLifecycleActivity extends Activity {
    // Static so that the thread access the latest attribute
    private static ProgressDialog dialog;
    private static Bitmap downloadBitmap;
    private static Handler handler;
    private ImageView imageView;
    private Thread downloadThread;
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        // create a handler to update the UI
        handler = new Handler() {
            @Override
            public void handleMessage(Message msg) {
                imageView.setImageBitmap(downloadBitmap);
                dialog.dismiss();
            }
        };
        // get the latest imageView after restart of the application
```

```
        imageView = (ImageView) findViewById(R.id.imageView1);
        Context context = imageView.getContext();
        System.out.println(context);
        // Did we already download the image?
        if (downloadBitmap != null) {
            imageView.setImageBitmap(downloadBitmap);
        }
        // check if the thread is already running
        downloadThread = (Thread) getLastNonConfigurationInstance();
        if (downloadThread != null && downloadThread.isAlive()) {
            dialog = ProgressDialog.show(this, "Download", "downloading");
        }
    }
    public void resetPicture(View view) {
        if (downloadBitmap != null) {
            downloadBitmap = null;
        }
        imageView.setImageResource(R.drawable.icon);
    }
    public void downloadPicture(View view) {
        dialog = ProgressDialog.show(this, "Download", "downloading");
        downloadThread = new MyThread();
        downloadThread.start();
    }
    // save the thread
    @Override
    public Object onRetainNonConfigurationInstance() {
        return downloadThread;
    }
    // dismiss dialog if activity is destroyed
    @Override
    protected void onDestroy() {
        if (dialog != null && dialog.isShowing()) {
            dialog.dismiss();
            dialog = null;
        }
        super.onDestroy();
    }
    // Utiliy method to download image from the internet
    static private Bitmap downloadBitmap(String url) throws IOException {
        HttpUriRequest request = new HttpGet(url);
        HttpClient httpClient = new DefaultHttpClient();
        HttpResponse response = httpClient.execute(request);
        StatusLine statusLine = response.getStatusLine();
        int statusCode = statusLine.getStatusCode();
        if (statusCode == 200) {
            HttpEntity entity = response.getEntity();
            byte[] bytes = EntityUtils.toByteArray(entity);
            Bitmap bitmap = BitmapFactory.decodeByteArray(bytes, 0,
                    bytes.length);
            return bitmap;
        } else {
```

```
                throw new IOException("Download failed, HTTP response code "
                        + statusCode + " - " + statusLine.getReasonPhrase());
            }
        }
        static public class MyThread extends Thread {
            @Override
            public void run() {
                try {
                    // Simulate a slow network
                    try {
                        new Thread().sleep(5000);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                    downloadBitmap =
downloadBitmap("http://www.devoxx.com/download/attachments/4751369/DV11");
                    // Updates the user interface
                    handler.sendEmptyMessage(0);
                } catch (IOException e) {
                    e.printStackTrace();
                } finally {
                }
            }
        }
}
```

اپلیکیشن خود را اجرا نموده و جهت راه اندازی فرایند دانلود، بر روی دکمه ی start کلیک نمایید. برای اینکه مطمئن شوید رفتار اپلیکیشن در خصوص مدیریت lifecycle یا چرخه ی حیات صحیح می باشد، می توانید در محیط شبیه ساز با فشردن کلید های ctrl+F11 جهت و وضعیت (orientation) نمایش را تغییر دهید.

لازم به ذکر است که Thread یک کلاس static و inner (ایستا و تعریف شده در دل یک کلاس دیگر) می باشد. به طور کلی لازم است برای پردازش های پس زمینه ای یک کلاس static inner را بکار ببرید چرا که در غیر این صورت کلاس inner (درون ساخته در دل کلاس دیگر) اشاره گری (reference) به کلاسی که در آن تعریف شده را در خود نگه می دارد. زمانی که thread به نمونه ی جدید از activity ارسال می شود، از آنجایی که این thread همچنان به نمونه ی قبلی activity اشاره دارد، هدر رفت حافظه و memory leak رخ خواهد داد.

بخش هشتم :

JSON-26-4 و اندروید

در این آموزش نحوه ی پردازش JSON در اندروید را خواهید آموخت.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-26-4-کتابخانه ی پیش فرض و درون ساخته ی اندروید برای پردازش JSON

محیط (platform) اندروید خود یک کتابخانه ی درون ساخته به نام json.org ویژه ی پردازش و ساخت فایل های JSON دارد. در صورت تمایل شما می توانید سایر کتابخانه های کد باز نظیر GSON یا Moshi را برای پردازش و ساخت JSON مورد استفاده قرار دهید.

مثال: خواندن فرمت JSON

تبدیل یک رشته ی JSON به آبجکت JSON بسیار ساده است.

کافی است کد زیر را برای activity خود درج نمایید.

```
import org.json.JSONArray;
import org.json.JSONObject;
String jsonString = readJsonObjectFromSomeWhere();          1
try {
    JSONObject json = new JSONObject(jsonString);
    } catch (Exception e) {
        e.printStackTrace();
    }
```

متدی است که یک رشته از جنس JSON را می خواند. به منظور کوتاهی کد معمولا از نوشتن آن خودداری می شود.

نکته: نمونه ی کد فوق در thread اصلی قابل اجرا نیست. لازم است این تکه کد را خارج از thread اصلی اجرا نمایید.

2-26-4-ساخت JSON

نوشتن JSON بسیار ساده است. کافی است یک JSONObject یا JSONArray ایجاد نموده و متد toString() را فراخوانی نمایید.

```
public void writeJSON() {
    JSONObject object = new JSONObject();
    try {
        object.put("name", "Jack Hack");
```

```
            object.put("score", new Integer(200));
            object.put("current", new Double(152.32));
            object.put("nickname", "Hacker");
        } catch (JSONException e) {
            e.printStackTrace();
        }
        System.out.println(object);
}
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فصل پنجم

طراحی پیشرفته رابط کاربری در اندروید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش اول :

پیاده سازی قابلیت drag&drop (کشیدن و جایگذاری)/کار با drag and drop

5-1-استفاده از drag & drop در اندروید

قابلیت drag & drop (کشیدن و جایگذاری) یک view بر روی view یا view group های دیگر از ویرایش 4.0 اندروید پشتیبانی می شود.

5-1-1-پیاده سازی قابلیت کشیدن view

به منظور پیاده سازی قابلیت drag، شما می بایست OnTouchListener یا LongClickListener را بر روی view ای که قرار است کشیده یا drag شود، تنظیم نمایید. به عبارت دیگر یکی از این دو interface را پیاده سازی کرده و متدهای آن را در کلاس بازنویسی کنید.

متد startDrag الحاق شده به view (فراخوانی شده بر روی آن) در واقع عملیات drag و کشیده شدن آیتم را آغاز می کند. در این متد می بایست view یا مکانی که view جاری قرار است بر روی آن کشیده و جایگذاری شود را به واسطه ی نمونه ای از کلاس ClipData مشخص نمایید.

سپس می بایست نمونه ای از کلاس DragShadowBuilder را به عنوان آرگومان به متد startDrag ارسال نمایید. این آبجکت عکسی که برای عملیات drag استفاده می شود را مشخص می نماید (به عبارت دیگر با استفاده از این آبجکت می توان برای آیتم در حال drag سایه ایجاد کنید). حال می

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

توانید این view را مستقیما به عنوان پارامتر ارسال کنید که در این صورت تصویری از view مورد نظر در طول کشیده شدن آیتم نمایش داده می شود.

نحوه ی تنظیم و راه اندازی عملیات کشیدن در touch listener با کد زیر به صورت عملی به نمایش گذاشته شده است.

```
// Assign the touch listener to your view which you want to move
findViewById(R.id.myimage1).setOnTouchListener(new MyTouchListener());
// This defines your touch listener
private final class MyTouchListener implements OnTouchListener {
    public boolean onTouch(View view, MotionEvent motionEvent) {
        if (motionEvent.getAction() == MotionEvent.ACTION_DOWN) {
            ClipData data = ClipData.newPlainText("", "");
            DragShadowBuilder shadowBuilder = new View.DragShadowBuilder(
                    view);
            view.startDrag(data, shadowBuilder, view, 0);
            view.setVisibility(View.INVISIBLE);
            return true;
        } else {
        return false;
        }
    }
}
```

2-1-5-تعریف مشخصات و اطلاعات محل جایگذاری view (مشخص کردن drop target)

View هایی که قرار است از قابلیت drop پشتیبانی کنند، می بایست نمونه ای از کلاس OnDragListener به آن ها اختصاص یابد. به عبارت دیگر، لازم است نمونه ای از onDragListener را به view ای که قرار است محل جایگذاری view دیگر باشد، تخصیص دهید. در این گوش فراخوان به رویداد drop (listener)، چنانچه از قبل event های مرتبط با drag&drop تعریف شده باشد، callback یا توابع بازفراخوانی را دریافت خواهید کرد. *

DragEvent.ACTION_DRAG_STARTED * DragEvent.ACTION_DRAG_ENTERED *

DragEvent.ACTION_DRAG_EXITED * DragEvent.ACTION_DROP *

. DragEvent.ACTION_DRAG_ENDED

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

View ای که OnDragListener برای آن پیاده سازی شده در واقع محل قرارگیری یک view دیگر بوده و از قابلیت drop پشتیبانی می کند. این setOnDragListener است که نمونه ای از OnDragListener را به این آبجکت تخصیص می دهد.

```
findViewById(R.id.bottomright).setOnDragListener(new MyDragListener());
class MyDragListener implements OnDragListener {
    Drawable enterShape = getResources().getDrawable(
            R.drawable.shape_droptarget);
    Drawable normalShape = getResources().getDrawable(R.drawable.shape);
    @Override
    public boolean onDrag(View v, DragEvent event) {
        int action = event.getAction();
        switch (event.getAction()) {
        case DragEvent.ACTION_DRAG_STARTED:
        // do nothing
            break;
        case DragEvent.ACTION_DRAG_ENTERED:
            v.setBackgroundDrawable(enterShape);
            break;
        case DragEvent.ACTION_DRAG_EXITED:
            v.setBackgroundDrawable(normalShape);
            break;
        case DragEvent.ACTION_DROP:
            // Dropped, reassign View to ViewGroup
            View view = (View) event.getLocalState();
            ViewGroup owner = (ViewGroup) view.getParent();
            owner.removeView(view);
            LinearLayout container = (LinearLayout) v;
            container.addView(view);
            view.setVisibility(View.VISIBLE);
            break;
        case DragEvent.ACTION_DRAG_ENDED:
            v.setBackgroundDrawable(normalShape);
            default:
            break;
        }
        return true;
    }
}
```

تمرین: پیاده سازی Drag and drop در اپلیکیشن به صورت عملی

در این تمرین شما تعدادی view group تعریف می کنید که از قابلیت drag&drop (کشیدن آیتم از یکی و جایگذاری آن در دیگری) پشتیبانی می کند.

ایجاد پروژه

یک پروژه و activity اندروید به ترتیب به نام های com.vogella.android.draganddrop و DragActivity ایجاد نمایید.

3-1-5-ایجاد فایل های Drawable (فایل های تصویری تعریف شده در فرمت XML)

در این بخش یکی از منابع مورد استفاده ی شما، فایل های XML drawable خواهند بود.

بنابراین ابتدا بایستی فایل های XML drawable مورد نیاز پروژه ی خود را در پوشه ی res/drawable ایجاد نمایید.

اولین فایلی که در آدرس مذکور ایجاد می کنید، shape.xml با محتویات زیر خواهد بود.

```
<?xml version="1.0" encoding="UTF-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
  android:shape="rectangle" >
  <stroke
    android:width="2dp"
    android:color="#FFFFFFFF" />
  <gradient
    android:angle="225"
    android:endColor="#DD2ECCFA"
    android:startColor="#DD000000" />
  <corners
    android:bottomLeftRadius="7dp"
    android:bottomRightRadius="7dp"
    android:topLeftRadius="7dp"
    android:topRightRadius="7dp" />
</shape>
```

سپس فایل shape_droptarget.xml را با محتویات زیر در پوشه ی نام برده ایجاد نمایید.

```
<?xml version="1.0" encoding="UTF-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
  android:shape="rectangle" >
  <stroke
    android:width="2dp"
    android:color="#FFFF0000" />
  <gradient
    android:angle="225"
    android:endColor="#DD2ECCFA"
    android:startColor="#DD000000" />
  <corners
    android:bottomLeftRadius="7dp"
    android:bottomRightRadius="7dp"
    android:topLeftRadius="7dp"
    android:topRightRadius="7dp" />
```

```
</shape>
```

4-1-5-Activity و تنظیم فایل layout مربوطه

محتویات layout کلاس activity خود را به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:columnCount="2"
   android:columnWidth="320dp"
   android:orientation="vertical"
   android:rowCount="2"
   android:stretchMode="columnWidth" >
   <LinearLayout
      android:id="@+id/topleft"
      android:layout_width="160dp"
      android:layout_height="160dp"
      android:layout_row="0"
      android:background="@drawable/shape" >
      <ImageView
         android:id="@+id/myimage1"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:src="@drawable/ic_launcher" />
   </LinearLayout>
   <LinearLayout
      android:id="@+id/topright"
      android:layout_width="160dp"
      android:layout_height="160dp"
      android:background="@drawable/shape" >
      <ImageView
         android:id="@+id/myimage2"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:src="@drawable/ic_launcher" />
   </LinearLayout>
   <LinearLayout
      android:id="@+id/bottomleft"
      android:layout_width="160dp"
      android:layout_height="160dp"
      android:background="@drawable/shape" >
      <ImageView
         android:id="@+id/myimage3"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:src="@drawable/ic_launcher" />
   </LinearLayout>
   <LinearLayout
      android:id="@+id/bottomright"
```

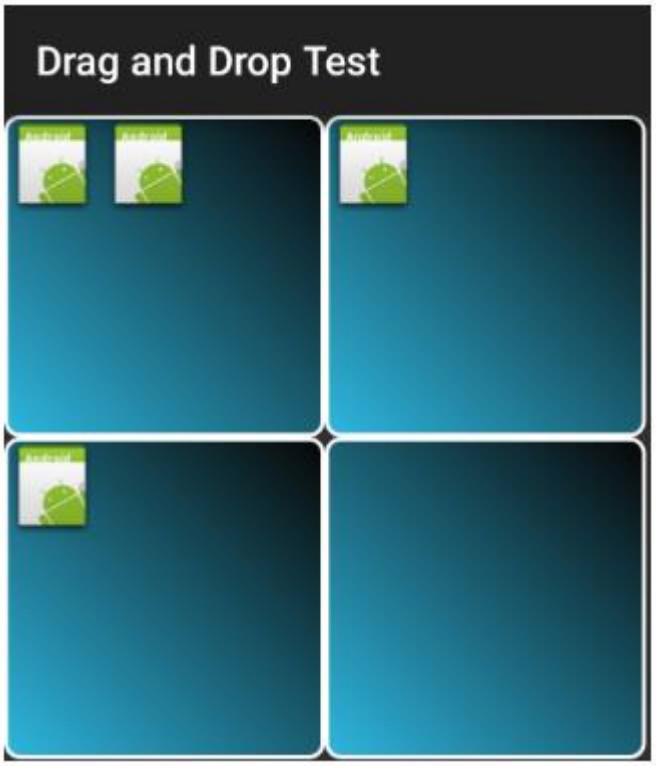
```
        android:layout_width="160dp"
        android:layout_height="160dp"
        android:background="@drawable/shape" >
        <ImageView
            android:id="@+id/myimage4"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/ic_launcher" />
    </LinearLayout>
</GridLayout>
```

حال بدنه ی کلاس activity خود را به صورت زیر ویرایش نمایید.

```
package com.vogella.android.draganddrop;
import android.app.Activity;
import android.content.ClipData;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
import android.view.DragEvent;
import android.view.MotionEvent;
import android.view.View;
import android.view.View.DragShadowBuilder;
import android.view.View.OnDragListener;
import android.view.View.OnTouchListener;
import android.view.ViewGroup;
import android.widget.LinearLayout;
public class DragActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        findViewById(R.id.myimage1).setOnTouchListener(new MyTouchListener());
        findViewById(R.id.myimage2).setOnTouchListener(new MyTouchListener());
        findViewById(R.id.myimage3).setOnTouchListener(new MyTouchListener());
        findViewById(R.id.myimage4).setOnTouchListener(new MyTouchListener());
        findViewById(R.id.topleft).setOnDragListener(new MyDragListener());
        findViewById(R.id.topright).setOnDragListener(new MyDragListener());
        findViewById(R.id.bottomleft).setOnDragListener(new MyDragListener());
        findViewById(R.id.bottomright).setOnDragListener(new MyDragListener());
    }
    private final class MyTouchListener implements OnTouchListener {
        public boolean onTouch(View view, MotionEvent motionEvent) {
            if (motionEvent.getAction() == MotionEvent.ACTION_DOWN) {
                ClipData data = ClipData.newPlainText("", "");
                DragShadowBuilder shadowBuilder = new View.DragShadowBuilder(
                        view);
                view.startDrag(data, shadowBuilder, view, 0);
                view.setVisibility(View.INVISIBLE);
                return true;
            } else {
```

```
                return false;
            }
        }
    }
    class MyDragListener implements OnDragListener {
        Drawable enterShape = getResources().getDrawable(
                R.drawable.shape_droptarget);
        Drawable normalShape = getResources().getDrawable(R.drawable.shape);
        @Override
        public boolean onDrag(View v, DragEvent event) {
            int action = event.getAction();
            switch (event.getAction()) {
            case DragEvent.ACTION_DRAG_STARTED:
                // do nothing
                break;
            case DragEvent.ACTION_DRAG_ENTERED:
                v.setBackgroundDrawable(enterShape);
                break;
            case DragEvent.ACTION_DRAG_EXITED:
                v.setBackgroundDrawable(normalShape);
                break;
            case DragEvent.ACTION_DROP:
                // Dropped, reassign View to ViewGroup
                View view = (View) event.getLocalState();
                ViewGroup owner = (ViewGroup) view.getParent();
                owner.removeView(view);
                LinearLayout container = (LinearLayout) v;
                container.addView(view);
                view.setVisibility(View.VISIBLE);
                break;
            case DragEvent.ACTION_DRAG_ENDED:
                v.setBackgroundDrawable(normalShape);
            default:
                break;
            }
            return true;
        }
    }
}
```

پس از راه اندازی این activity، می توانید ImageViews را کشیده و در ظرف دیگری جایگذاری نمایید.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش دوم :

کار با drawable ها/ فایل های تصویری تعریف شده در فرمت XML

این آموزش به شرح مفهوم Drawable ها و نحوه ی استفاده از آن ها در اندروید می پردازد.

Drawable-2-5 چیست؟

Drawable یک مفهوم کلی در اندروید برای هر فایل گرافیکی قابل ترسیم می باشد یا به عبارت دیگر drawable هر چیزی است که بتوان آن را کشید. به عنوان ساده ترین نمونه می توان به یک فایل گرافیکی (bitmap) اشاره کرد که در اندروید توسط کلاس BitmapDrawable در اختیار توسعه دهنده قرار می گیرد.

هر drawable در قالب یک فایل مجزا در مسیر res/drawable ذخیره می شود. معمولا فایل های drawable در قالب فایل های bitmap در وضوح مختلف داخل زیرپوشه های mdpi– ، hdpi- xhdpi – xxhdpi مسیر res/drawable ذخیره می شوند. ویزارد و راهنمای ساخت پروژه ی ADT این پوشه ها را خود به صورت پیش فرض ایجاد می کند. اگر bitmap ها در پوشه ی متفاوتی ذخیره شده باشند، سیستم اندروید خود تنظیمات دستگاه و اندازه ی صفحه نمایش را ارزیابی کرده و بر اساس آن گزینه ی مناسب را انتخاب می کند.

چنانچه عکس مورد نظر را در تمامی وضوح (تراکم پیکسلی و با کیفیت مناسب) نداشته باشید، در آن صورت سیستم اندروید تصویر مربوطه را جهت سازگاری کامل با صفحه نمایش دستگاه میزبان کوچک یا بزرگ می کند. البته این قابلیت ظاهر چندان مناسبی را به وجود نمی آورد چرا که در بیشتر موارد عکس مورد نظر تار شده و از وضوح و کیفیت مناسب برخوردار نخواهد بود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

علاوه بر فایل های گرافیکی، اندروید از drawable های مبتنی بر XML و 9-patch پشتیبانی می کند. drawable های که در فایل های XML تعریف می شوند به برنامه نویس این امکان را می دهد تا رنگ، حاشیه، شیب/طیف رنگ و گوشه ها را در قالب تگ shape و همچنین state (جهت تعریف عکس خاص برای هر وضعیت ای که view در آن قرار می گیرد)، transition (برای تعریف انیمیشن) فایل گرافیکی مورد نظر را تعریف کند.

به وسیله ی فایل های گرافیکی 9-patch می توان مشخص کرد که در صورت بزرگ تر بودن view میزبان، کدام بخش از فایل گرافیکی مورد نظر می بایست کشیده (بزرگتر) شود.

فایل های Drawable را می توان با کدهای Java نوشت. هر آبجکتی که توابع اینترفیس Drawable را پیاده سازی می کند را می توان به صورت فایل ترسیم شونده/Drawable در کد بکار برد.

3-5-استفاده از drawable ها در view ها

جهت دسترسی و اشاره به drawable ها در XML می توان از این ساختار استفاده نمود: @drawable/filename. در این ساختار نگارشی filename همان اسم فایل بدون پسوند می باشد. برای مثال جهت دستیابی به فایل drawable در مسیر res/drawable/hello.png و درج آن در پس زمینه ی UI، کافی است ساختار @drawable/hello را به صورت زیر استفاده نمایید.

```
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/textView1"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:background="@drawable/hello"
   android:text="@string/hello_world" />
```

می توان با کدنویسی drawable ها را به view ها اختصاص داد. بیشتر view ها این قابلیت را دارند که ID فایل منبع مورد نظر را به عنوان پارامتر ورودی بپذیرند. کد زیر نحوه ی تخصیص یک فایل drawable را به عنوان تصویر پس زمینه به ImageView نمایش می دهد.

```
ImageView imageView = (ImageView) findViewById(R.id.image);
imageView.setImageResource(R.drawable.hello);
```

4-5-بارگذاری Bitmap ها و drawable ها

جهت استفاده از bitmap ها در اندروید می توانید از کلاس Bitmap استفاده نمایید. در این بخش خواهید آموخت چگونه با کد جاوا آبجکت های Bitmap ایجاد نموده و سپس آن را به آبجکت های Drawable و بالعکس تبدیل نمایید.

این امکان برای شما وجود دارد که فایل bitmap موجود را با کدنویسی بارگذاری نموده و متعاقبا آن ها را به آبجکت های Drawable تبدیل کرد.

مثال زیر نمایش می دهد چگونه می توان یک آبجکت Bitmap در پوشه ی assets ایجاد نموده و آن را به المان رابط کاربری ImageView تخصیص داد.

```
AssetManager manager = getAssets();
        // read a Bitmap from Assets
        InputStream open = null;
        try {
            open = manager.open("logo.png");
            Bitmap bitmap = BitmapFactory.decodeStream(open);
            // Assign the bitmap to an ImageView in this layout
            ImageView view = (ImageView) findViewById(R.id.imageView1);
            view.setImageBitmap(bitmap);
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (open != null) {
                try {
                    open.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
```

همچنین می توانید به فایل های Drawable از پوشه ی res/drawable به صورت آبجکت های Bitmap در کد برنامه دسترسی پیدا کنید. کد زیر این قابلیت را به نمایش می گذارد.

```
Bitmap b = BitmapFactory.decodeResource(getResources(), R.drawable.ic_action_search);
```

تعریف نمایید.

```
Bitmap originalBitmap = <initial setup>;
Bitmap resizedBitmap =
  Bitmap.createScaledBitmap(originalBitmap, newWidth, newHeight, false);
```

به منظور تبدیل یک آبجکت Bitmap به Drawable می توانید کد زیر را بکار ببرید.

```
# Convert Bitmap to Drawable
Drawable d = new BitmapDrawable(getResources(),bitmap);
```

Drawable-5-5 های مبتنی بر XML

Shape-5-5-1

Shape ها نیز یک نوع drawable هستند که در فرمت XML تعریف شده و به توسعه دهنده این امکان را می دهند تا آبجکت یا اشکال هندسی با رنگ، حاشیه و شیب رنگ تعریف نمایند.

می توانید این اشکال هندسی را به view های مورد نظر تخصیص داده و از آن ها در المان های رابط کاربری استفاده نمایید.

مزیت استفاده از shape در این است که خود را به صورت اتوماتیک به مقیاس مناسب تنظیم می کند.

کد زیر نمونه ای از یک Shape را نشان می دهد.

```
<?xml version="1.0" encoding="UTF-8"?>
<shape
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <stroke
        android:width="2dp"
        android:color="#FFFFFFFF" />
    <gradient
        android:endColor="#DDBBBBBB"
        android:startColor="#DD777777"
        android:angle="90" />
    <corners
        android:bottomRightRadius="7dp"
        android:bottomLeftRadius="7dp"
        android:topLeftRadius="7dp"
        android:topRightRadius="7dp" />
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
</shape>
```

می توانید drawable مزبور را به ویژگی background (property) در فایل layout تخصیص داده و آن را به عنوان تصویر پس زمینه ی UI تنظیم نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:background="@drawable/myshape"
   android:orientation="vertical" >
   <EditText
      android:id="@+id/editText1"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      >
   </EditText>
   <RadioGroup
      android:id="@+id/radioGroup1"
      android:layout_width="match_parent"
      android:layout_height="wrap_content" >
      <RadioButton
         android:id="@+id/radio0"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:checked="true"
         android:text="@string/celsius" >
      </RadioButton>
      <RadioButton
         android:id="@+id/radio1"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="@string/fahrenheit" >
      </RadioButton>
   </RadioGroup>
   <Button
      android:id="@+id/button1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/calc"
      android:onClick="myClickHandler">
   </Button>
</LinearLayout>
```

Drawable-5-5-2 های مبتنی بر state

می توان برای هر وضعیت یک drawable ویژه تعریف کرد و سپس بسته به وضعیت جاری، drawable مربوطه را به view تخصیص داد. برای مثال تکه کد زیر وضعیت دکمه را در نظر می گیرد و سپس با توجه به وضعیت آن، drawable مربوطه را به view اختصاص می دهد.

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:drawable="@drawable/button_pressed"
        android:state_pressed="true" />
    <item android:drawable="@drawable/button_checked"
        android:state_checked="true" />
    <item android:drawable="@drawable/button_default" />
</selector>
```

Drawable-5-5-3 هایی که طی انتقال جایگزین drawable دیگری می شوند (transition drawable)

شما می توانید در تگ transition، افکت انتقال و جابجایی را تعریف کرده و آن را در کد فعال نمایید. از این طریق یک تصویر طی transition جایگزین تصویری قبلی می شود.

```
<?xml version="1.0" encoding="utf-8"?>
<transition xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:drawable="@drawable/first_image" />
    <item android:drawable="@drawable/second_image" />
</transition>

final ImageView image = (ImageView) findViewById(R.id.image);
final ToggleButton button = (ToggleButton) findViewById(R.id.button);
button.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(final View v) {
        TransitionDrawable drawable = (TransitionDrawable) image.getDrawable();
        if (button.isChecked()) {
            drawable.startTransition(500);
        } else {
            drawable.reverseTransition(500);
        }
    }
});
```

Drawable-6-5 های برداری/توسعه پذیر بدون از دست رفت کیفیت (vector drawable)

از ویرایش 5.0، اندروید به توسعه دهندگان این امکان را می دهد تا drawable های برداری/vector مانند فایل های SVG تعریف کنند. کد زیر کاربرد آن را در فایل vectordrawable.xml به نمایش می گذارد. استفاده از drawable های برداری این مزیت را دارد که خود را با توجه به تراکم پیکسلی (density) و کیفیت دستگاه میزبان تنظیم می کند.

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:height="64dp"
    android:width="64dp"
    android:viewportHeight="600"
    android:viewportWidth="600" >
    <group
      android:name="rotationGroup"
      android:pivotX="300.0"
      android:pivotY="300.0"
      android:rotation="45.0" >
      <path
        android:name="v"
        android:fillColor="#000000"
        android:pathData="M300,70 l 0,-70 70,70 0,0 -70,70z" />
    </group>
</vector>
```

از ویرایش 5.0، کلاس جدیدی به نام AnimatedVectorDrawable به کتابخانه های اندروید اضافه شد که به برنامه نویس اجازه می دهد تا drawable های برداری را با انیمیشن ترکیب کند (ویژگی های drawable را با انیمیشن های تعریف شده توسط ObjectAnimator یا AnimatorSet متحرک و پویا کند). برای مشاهده ی مثال هایی بیشتر می توانید به آدرس http://blog.sqisland.com/2014/10/first-look-at-animated-vector-drawable.html مراجعه نمایید.

برای استفاده از drawable های برداری در platform یا ورژن های قدیمی تر چارچوب نرم افزاری اندروید، می توانید از کتابخانه ی VectorDrawableCompat (یک کتابخانه که امکان استفاده از drawable های برداری در ورژن های قبلی اندروید را فراهم می آورد) استفاده نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Drawable animation-7-5 (تعریف انیمیشن با بارگذاری یک drawable پس از دیگری)

می توانید animation drawable تعریف کرده و با استفاده از متد setBackgroundResource() آن را به View تخصیص دهید. اندروید به شما این امکان را می دهد تا drawable هایی را با حالت انیمیشن یکی پس از دیگری نمایش دهید.

```
<!-- Animation frames are phase*.png files inside the
 res/drawable/ folder -->
 <animation-list android:id="@+id/selected" android:oneshot="false">
    <item android:drawable="@drawable/phase1" android:duration="400" />
    <item android:drawable="@drawable/phase2" android:duration="400" />
    <item android:drawable="@drawable/phase3" android:duration="400" />
 </animation-list>
ImageView img = (ImageView)findViewById(R.id.yourid);
img.setBackgroundResource(R.drawable.your_animation_file);
 // Get the AnimationDrawable object.
 AnimationDrawable frameAnimation = (AnimationDrawable) img.getBackground();
// Start the animation (looped playback by default).
frameAnimation.start();
```

همچنین می توانید آبجکتی پیاده سازی کنید که از کلاس Drawable ارث بری کرده و توابع اینترفیس Animatable را پیاده سازی می کند.

Drawable های برداری را می توان با استفاده از کتابخانه ی VectorDrawableCompat در ورژن های قبلی های اندروید نیز بکار برد.

Drawable-5-7-1 های nine-patch (فایل های ترسیم شونده ی منعطف با کناره های بسط پذیر)

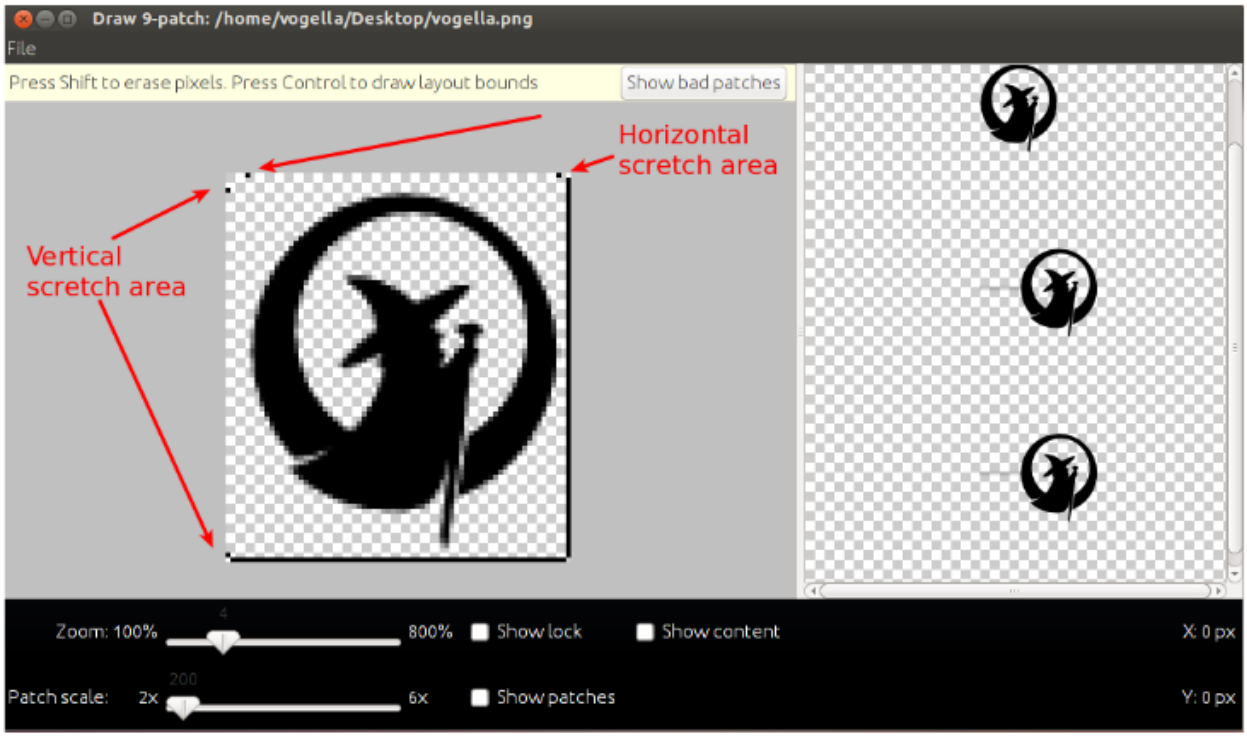
تصور کنید یک عکس دارید و زمانی که شما این تصویر را می کشید، کناره یا لبه ی عکس تار می شود. حال اگر شما تصویری داشته باشید که وسط آن تغییر نکرده و ثابت بماند (با کشیدن تصویر تار نشود)، می توانید آن بخش هایی که با بزرگتر شدن عکس مات نمی شوند را 9-patch تعریف کنید. سیستم اندروید در زمان اجرای برنامه بخش های علامت گذاری نشده را از نظر اندازه تغییر

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

نمی دهد، اما سایر بخش ها را کش داده و عکس را بزرگ می کند. بدین وسیله زمانی که تصویر برای کاربر به نمایش در می آید، کیفیت آن کاهش نمی یابد.

Drawable های 9-patch، فایل های ترسیم شونده تصویری هستند که یک پیکسل اضافی در حاشیه دارند. در بالا و سمت چپ می توانید ناحیه ای تعریف کنید که متناسب با view اندازه بندی و در صورت لزوم بزرگ می شوند. این ناحیه stretch area است.



چنانچه drawable قرار است در یک view قرار گیرد که اجازه ی نوشتن بر روی آن را داشته باشد (همچون آبجکت Button)، در آن صورت می توانید در کناره های سمت راست و پایین ناحیه ای تعریف کنید که در آن متن امکان درج را داشته باشد.

ADT برنامه ای به نام draw9patch در پوشه ی android-sdk/tools ارائه می دهد که ساخت drawable های انعطاف پذیر و 9-patch را آسان می سازد.

Drawable-8-5 های اختصاصی

می توانید drawable های اختصاصی تعریف کنید که برای نمایش در UI از Canvas API بهره می گیرند. می توانید با استفاده از تمامی توابع Canvas API/ فایل های ترسیم شونده یا drawable را متناسب با نیاز خود تنظیم و طراحی کنید.

9-5-ساخت drawable های اختصاصی

پروژه جدید به نام com.vogella.android.drawables.custom ایجاد کرده و قالب آماده ی Empty Activity را برای توسعه ی پروژه ی خود انتخاب نمایید.

کلاس اختصاصی Drawable را به صورت زیر پیاده سازی نمایید.

```
package com.vogella.android.drawables.custom;
import android.graphics.Bitmap;
import android.graphics.BitmapShader;
import android.graphics.Canvas;
import android.graphics.ColorFilter;
import android.graphics.Paint;
import android.graphics.PixelFormat;
import android.graphics.RectF;
import android.graphics.Shader;
import android.graphics.drawable.Drawable;
public class MyRoundCornerDrawable extends Drawable {
    private Paint paint;
    public MyRoundCornerDrawable(Bitmap bitmap) {
        BitmapShader shader;
        shader = new BitmapShader(bitmap, Shader.TileMode.CLAMP,
                Shader.TileMode.CLAMP);
        paint = new Paint();
        paint.setAntiAlias(true);
        paint.setShader(shader);
    }
    @Override
    public void draw(Canvas canvas) {
        int height = getBounds().height();
        int width = getBounds().width();
        RectF rect = new RectF(0.0f, 0.0f, width, height);
        canvas.drawRoundRect(rect, 30, 30, paint);
    }
    @Override
    public void setAlpha(int alpha) {
        paint.setAlpha(alpha);
    }
    @Override
    public void setColorFilter(ColorFilter cf) {
        paint.setColorFilter(cf);
```

```
    }
    @Override
    public int getOpacity() {
        return PixelFormat.TRANSLUCENT;
    }
}
```

برای استفاده از این کلاس در متن پروژه، فایل layout خود را به صورت زیر ویرایش نمایید.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".MainActivity" >
  <ImageView
    android:id="@+id/image"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:contentDescription="TODO" />
</RelativeLayout>
```

پیاده سازی کلاس MainActivity خود را به صورت زیر ویرایش نمایید. کد فرض را بر این می گذارد شما یک فایل bitmap به نام dog.png در پوشه ی drawable خود دارید.

```
package com.vogella.android.drawables.custom;
import java.io.InputStream;
import android.app.Activity;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.view.Menu;
import android.widget.ImageView;
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ImageView button = (ImageView) findViewById(R.id.image);
        InputStream resource = getResources().openRawResource(R.drawable.dog);
        Bitmap bitmap = BitmapFactory.decodeStream(resource);
        button.setBackground(new MyRoundCornerDrawable(bitmap));
    }
}
```

بخش سوم :

طراحی اپلیکیشن بر اساس اصول و قواعد مشخص/با استفاده از theme ها و style ها

این آموزش اصول طراحی بهینه ی اپلیکیشن را برای شما تشریح می کند. سپس شرح می دهد چگونه می توانید در اپلیکیشن های خود style و theme ایجاد کرده و بکار ببرید.

5-10-اصول طراحی UI در اندروید

جهت تعریف UI کارامد برای اپلیکیشن های خود بهتر است از اصول زیر پیروی نمایید. این اصول به تفصیل تحت آدرس http://developer.android.com/design/index.html شرح داده شده اند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1. UI را طوری طراحی کنید که با توجه به قابلیت لمس بهینه و کارامد باشد.
2. تنها آنچه لازم است نمایش دهید.
3. لزومی ندارد برای انجام هر عملیاتی از کاربر اجازه بگیرید، با این وجود لازم است امکان لغو عملیات یا بازگرداندن آن را برای کاربر مهیا کنید.
4. تنها در صورت لزوم تعامل کاربر با UI را مختل نمایید.
5. پیغام ها را تا حد امکان مختصر نگه داشته و از عکس برای رساندن مفهوم مورد نظر استفاده نمایید.
6. برنامه و ظاهر آن را طوری طراحی نمایید که اطلاعات کاربر به طور امن در آن نگهداری شده و هیچگاه از دست نرود.
7. به کاربران این امکان را بدهید تا آیتم های لازم را سریع ایجاد کنند.
8. اگر ظاهر یکسان است، در آن صورت رفتار نیز باید یکسان باشد.
9. در تصمیم گیری به کاربر کمک کنید اما تصمیم نهایی را به او واگذار نمایید.

در خصوص طراحی و توسعه ی برنامه نیز روش های بهینه وجود دارد که در زیر به آن ها اشاره می کنیم:

- در طراحی اپلیکیشن راندمان و کارایی بهینه در مرتبه ی اول قرار می گیرد – یک اپلیکیشن که به صورت بهینه طراحی شده باشد طبیعتا با سرعت قابل توجهی اجرا می شود. زمان اجرای (اولیه) اپلیکیشن به طور متوسط نباید بیش از 1 ثانیه به طول بیانجامد. بعلاوه تمامی عملیات طولانی باید به طور ناهمزمان اجرا شوند.
- اپلیکیشن خود را طوری طراحی کنید که با سرعت با کاربر تعامل کند – بازخورد می بایست سریعاً در اختیار کاربر قرار گیرد. در صورت اجرای عملیات طولانی بهتر است پیغام کوتاهی در UI برای کاربر به نمایش بگذارید.
- در طراحی اپلیکیشن حداقل مصرف باتری را در نظر بگیرید – اپلیکیشن شما می بایست حداقل میزان مصرف باتری را داشته باشد. در صورتی که اپلیکیشن در UI فعال و قابل مشاهده نیست، تمامی آپدیت های مربوط به رابط کاربری و گوش فراخوان ها (listener به رخدادها) را غیرفعال نمایید. می توانید به event هایی نظیر متصل بودن دستگاه به

شارژر گوش داده و به محض اتفاق افتادن این رخداد، آپدیت های طولانی را فعال نمایید. چنانچه لازم است داده های حجیمی را از طریق آپدیت از سرور خارجی واکشی نمایید، بهترین گزینه سرویس Google push notification می باشد. با پیاده سازی این سرویس شما تنها زمانی اجازه ی اتصال به اینترنت را می دهید که داده های لازم در دسترس باشند.

- در دسترس قرار دادن داده های اخیر در اختیار کاربر در زمان راه اندازی اولیه اپلیکیشن – اپلیکیشنی که شما می نویسید می بایست در صورت امکان داده های اخیر را به محض بالا آمدن برنامه در اختیار کاربر خود قرار دهد. بنابراین توصیه می شود برای واکشی داده ها از سرویس دهنده های خارجی (external servers) از service ها بهره بگیرید تا از این طریق عملیات بازیابی اطلاعات و لایه ی UI اپلیکیشن از هم جدا شوند.
- بی مورد به اینترنت خودداری کند (push notification یک سرویس است که ارتباط بین سرویس دهنده و سرویس گیرنده را فراهم می آورد. موارد کاربرد آن عبارت است: همگام سازی، اعمال تغییرات بلادرنگ بر روی سرویس گیرنده، چت سرویس دهنده. به عبارت دیگر push notification پیامی است که به کاربر خارج از اپلیکیشن ارائه می دهید). همچنین توصیه می شود وضعیت جاری اتصال به اینترنت را بررسی نمایید. زمانی که دستگاه به wifi دسترسی دارد، طبیعتا اپلیکیشن شما امکان دانلود اطلاعات بیشتری را خواهد داشت.

برای کسب اطلاعات بیشتر در خصوص اصول طراحی اپلیکیشن های اندرویدی، می توانید به آدرس http://developer.android.com/design/index.html مراجعه نمایید.

نکته: در خصوص انتخاب اسم (filename) برای آیکون ها چند نکته لازم به توضیح است: 1. نباید در اسم آیکون ها از حروف و کاراکترهای خاص استفاده نمایید 2. اسم آیکون نباید با عدد آغاز شود 3. لازم است اسم آیکون با حروف کوچک نوشته شود.

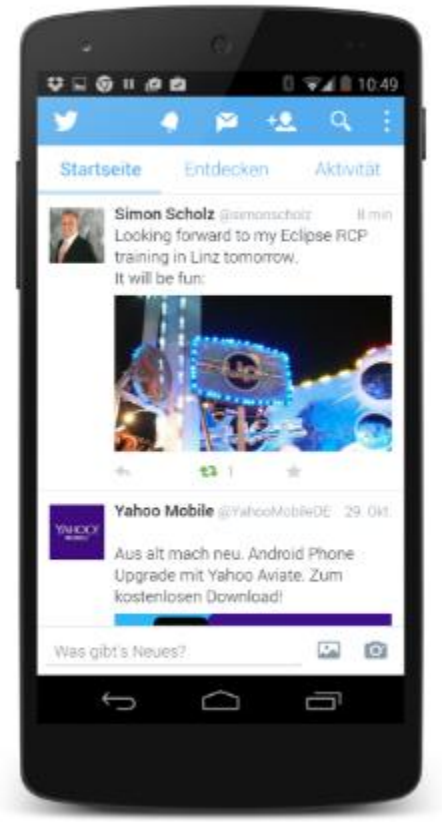
ساخت محتوای تبلیغاتی:

سایت - http://developer.android.com/distribute/tools/promote/device-art.html Device Art Generator به شما این امکان را می دهد تا محتوای تبلیغاتی برای اپلیکیشن خود

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

ایجاد نمایید و عکس برنامه ی خود را در یک قاب زیبا جایگذاری نمایید. کافی است یک تصویر از اپلیکیشن در صفحه نمایش تهیه نموده و سپس آن را بر روی دستگاه مناسب جایگذاری نمایید.



1-10-5-طراحی انعطاف پذیر و واکنش گرا برای اپلیکیشن (Responsive design)

امروزه قابلیت بزرگ یا کوچک شدن اپلیکیشن متناسب با نمایشگر و عرض صفحه ی دستگاه میزبان یک ویژگی پایه ای قملداد می شود که تقریبا هر برنامه ای باید از آن برخوردار باشد. از این رو توصیه می شود حتما UI اپلیکیشن خود را طوری طراحی نمایید که با توجه به نمایشگر دستگاه خود را اندازه بندی کرده و ظاهری زیبا را در راستای تجربه ی کاربری مطلوب از خود ارائه دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در یک نمایشگر کوچک، اپلیکیشن می بایست در لحظه تنها یک fragment را به نمایش بگذارد. در حالی که در صورت عریض بودن دستگاه میزبان بایستی بتواند دو یا حتی طور سه fragment را در آن واحد نمایش دهد.

در تصویر زیر این رویکرد در طراحی لایه ی رابط کاربری اپلیکیشن به راحتی مشاهده می شود.

چنانچه عرض صفحه نمایش دستگاه میزبان از یک حد معین بزرگتر می شود، بهتر است برای ارائه ی نوشته ها در UI از حالت نمایش تمام صفحه استفاده نکنید. این حد معمولا بالای w1000dp می باشد. تحقیقات انجام گرفته حاکی از آن است که در این حالت کاربر مجبور می شود جهت خواندن محتوا سر خود را بیش از حد به چپ یا راست حرکت دهد که در کل تجربه ی کاربری ضعیفی را به دنبال خواهد داشت.

یک راه برای پیاده سازی margin points این است که به فایل res/values/dimens.xml دسترسی پیدا کرده و برای خطوط حاشیه (margin ها) مقدار یا اندازه ی معینی را تعریف نمایید. پس از آن می توانید با استفاده از resource qualifier فایل مربوطه، برای نمایشگرهای عریض تر مقادیر margin متفاوت تعیین نمایید.

5-11-استفاده از style و theme در اپلیکیشن

1-11-5-طراحی ساده و بهینه برای اپلیکیشن های اندرویدی با material design/تم گذاری و material design

اصول و رهنمودهای طراحی UI برای اپلیکیشن های اندرویدی طی سالیان تغییر زیادی کرده است. اولین تغییر بزرگ در طراحی ظاهر اپلیکیشن های اندرویدی با ویرایش 3.0 اندروید و تحت نام Holo style معرفی شد. از ویرایش 5.0 سیستم عامل اندروید، طراحی UI بار دیگر با ارئه ی material design (طراحی ساده و تا حد امکان بدون عکس) متحول شد. این الگوی طراحی رابط کاربری، مفهوم depth را در layout معرفی نموده و جهت تعامل با کاربر بیشتر از انیمیشن بهره می گیرد.

آدرس زیر تعداد زیادی منبع قابل دانلود همچون مجموعه آیکون ها را در اختیار توسعه دهندگان قرار می دهد: https://developer.android.com/design/downloads/index.html.

از ویرایش 5.0 اندروید (API 21)، الگوی material design برای طراحی ظاهر اپلیکیشن توصیه می شود.

Material design یک الگو جهت طراحی UI است که هدف اصلی آن ادغام اصول قدیمی طراحی با قابلیت ها و فناوری های نوین می باشد. این راهنمای جامع همچنین توانسته تعادلی میان کارایی و زیبایی را به ارمغان بیاورد و سازوکاری جالب برای طراحی سایت های جدید ارائه دهد. محیط و بستر اجرای اندروید (android platform) تم نوین، المان های رابط کاربری (widget) و API جدید برای سایه انداختن (shadow های اختصاصی) و پیاده سازی انیمیشن در اختیار توسعه دهندگان قرار می دهد.

Material design با تخصیص elevation level (میزان فاصله ی بین دو المان که بر روی یکدیگر قرار می گیرند) به view ها، این امکان را برای طراح مهیا می کند تا المان های رابط کاربری را بر روی یکدیگر ترسیم نماید. Elevation level را باید بر حسب dp به view تخصیص دهید. حال جهت تنظیم میزان فاصله ی دو المان که بر روی هم قرار می گیرند، می بایست خصیصه android:elevation (attribute) را در تعریف فایل layout تنظیم نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

جهت مقداردهی میزان elevation بین دو المان با کدنویسی (در کلاس های activity)، لازم است متد View.setElelvation() را فراخوانی نمایید. اندروید بر اساس مقدار اختصاص یافته به elevation، سایه های اختصاصی (shadow) بر روی المان ها می اندازد.

Material design همچنین API ها و توابع بهینه تری را برای پیاده سازی انیمیشن ارائه نموده و همچنین تنوع انیمیشن ها را نسبت به قبل افزایش داده است.

2-11-5-شرح مفهوم style و theme

اندروید به شما این امکان را می دهد تا ظاهر کامپوننت های اندرویدی خود (همچون رنگ و فونت) را در فایل های XML تعریف نمایید. این امکان به شما اجازه می دهد تا تمامی attribute هایی که ظاهر اپلیکیشن را تعریف و سبک دهی می کنند به صورت یکجا و در یک مکان واحد مقداردهی نمایید.

آن دسته از المان هایی که در فایل منابع و محتوا (resource) برای سبک دهی به (تنظیم و طراحی ظاهر) view ها مورد استفاده قرار می گیرند، style نامیده می شوند و آن دسته از المان هایی که ویژه ی سبک دهی و تنظیم ظاهر یک activity یا اپلیکیشن تعبیه شده باشند، theme نام دارند.

به منظور تعریف style یا theme، لازم است یک فایل XML در پوشه ی اصلی /res/values پروژه ی خود ذخیره نمایید. گره اول و اصلی (root node) فایل XML مورد نظر می بایست تگ <resources> می باشد. به منظور تعریف style و theme، می بایست یک المان با تگ style و یک خصیصه ی name (attribute) تعریف نمایید. این المان می تواند یک یا چندین آیتم در دل خود داشته باشد و مقادیر attribute های نام گذاری شده را تعیین نماید.

کد زیر نمونه ای از تعریف یک style را نمایش می دهد.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <style name="text">
      <item name="android:padding">4dip</item>
```

```
    <item name="android:textAppearance">?android:attr/textAppearanceLarge</item>
    <item name="android:textColor">#000000</item>
  </style>
  <style name="layout">
    <item name="android:background">#C0C0C0</item>
  </style>
</resources>
```

به منظور تعریف style برای المان های موجود در فایل layout، می بایست به این صورت اقدام نمایید: style="@style/text".

Style ها و theme ها می توانند به واسطه ی attribute پدر (میزبان) تگ style، ویژگی های والد خود را به ارث برده و از inheritance پشتیبانی نمایند. در واقع بدین وسیله style مورد نظر تمامی تنظیمات را از style پدر خود به ارث برده و می تواند attribute های انتخابی را بازنویسی (overwrite) کند.

3-12-5-دسترسی و اشاره به attribute ها در theme جاری

اندروید تمامی attribute های استاندارد که امکان style دهی آن ها وجود دارد را در فایلی به نام R.attr فهرست کرده است. این فایل تحت آدرس http://developer.android.com/reference/android/R.attr.html قابل دسترسی می باشد.

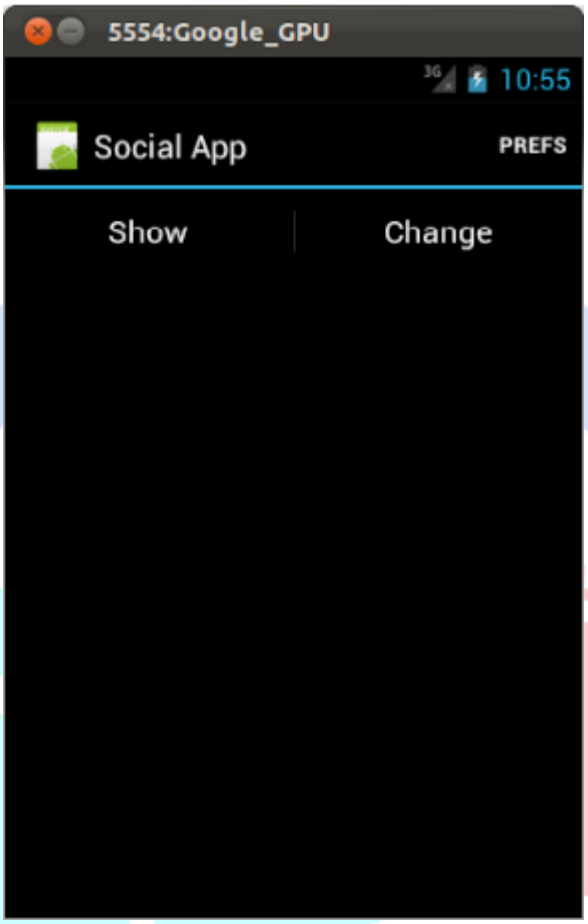
می توانید با استفاده از دستور نگارشی (notation) ?android:attr، به attribute های theme جاری اندروید دسترسی داشته باشید. به عبارت دیگر شما می توانید با استفاده از این دستور به خصیصه ی style (attribute) در theme جاری اشاره کنید.

به عنوان مثال کد ?android:attr/listPreferredItemHeight به اندروید دستور می دهد تا مقدار اختصاص یافته به خصیصه ی listPreferredItemHeight را در تم جاری بکار ببرد.

فایل layout زیر به المان های دکمه در UI استایل دکمه های ویرایش 4.0 اندروید را اعمال می کند.

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
  <LinearLayout
    style="?android:attr/buttonBarStyle"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal" >
    <Button
      android:id="@+id/Button01"
      style="?android:attr/buttonBarButtonStyle"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="1"
      android:text="Show" />
    <Button
      android:id="@+id/Button02"
      style="?android:attr/buttonBarButtonStyle"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="1"
      android:text="Change" />
  </LinearLayout>
  <EditText
    android:id="@+id/myView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10" >
    <requestFocus />
  </EditText>
</LinearLayout>
```



Theme–5–12–4 چیست؟

Theme عبارت است یک style که به جای یک view به کل یک activity (یک صفحه از اپلیکیشن) یا اپلیکیشن اعمال می شود. روش تعریف آن با تعریف style تفاوتی ندارد.

مثال بعدی نحوه ی تعریف یک theme اختصاصی را با ارث بری از theme درون ساخته ی محیط اندروید (platform) شرح می دهد.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <style name="MyTheme" parent="android:Theme.Light">
      <item name="android:windowNoTitle">true</item>
      <item name="android:windowBackground">@color/translucent_red</item>
      <item name="android:listViewStyle">@style/MyListView</item>
   </style>
```

```
    <style name="MyListView" parent="@android:style/Widget.ListView">
        <item name="android:listSelector">@drawable/ic_menu_home</item>
    </style>
</resources>
```

13-5-استفاده از theme های خود محیط اندروید

1-13-5-استفاده از material design و کتابخانه ی لازم برای پشتیبانی از آن در طراحی اپلیکیشن اندرویدی

از API/ورژن 21 به بعد کتابخانه های اندروید، توسعه دهندگان به استفاده از material design جهت طراحی ظاهر اپلیکیشن ترغیب می شوند.

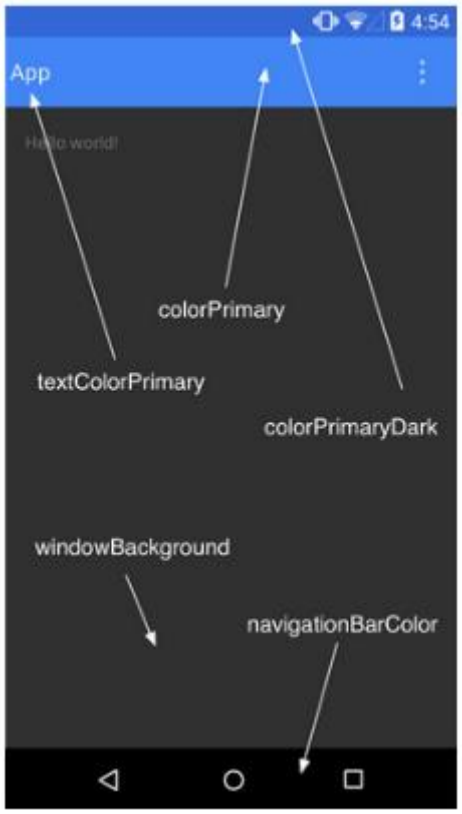
- @android:style/Theme.Material (dark version)
- @android:style/Theme.Material.Light (light version)
- @android:style/Theme.Material.Light.DarkActionBar

برای پیاده سازی material design در طراحی اپلیکیشن هایی که برای ویرایش های قدیمی اندروید نوشته شده اند، بایستی از یک Support library استفاده نمایید که theme و style های این الگوی طراحی UI را در اختیار شما قرار دهد. برای دسترسی به این کتابخانه می توانید به آدرس https://developer.android.com/training/material/compatibility.html مراجعه فرمایید. با استفاده از این کتابخانه می توانید از material design در طراحی ظاهر برنامه در ورژن های قدیمی تر اندروید بهره بگیرید.

2-13-5-تنظیم رنگ های پایه ی theme (Styling the color palette)

با معرفی material design شما می توانید رنگ های پایه ی تم ها را سفارشی تنظیم نمایید.

تصویر زیر تعداد زیادی از این رنگ ها را نمایش می دهد.



کد زیر نحوه ی استایل دهی و تنظیم رنگ تصویر بالا در فایل XML را نمایش می دهد (این فایل در پوشه ی res/values/ تعریف شده است).

```
<resources>
  <style name="AppTheme" parent="android:Theme.Material">
    <!-- Main theme colors -->
    <!--   your app branding color for the app bar -->
    <item name="android:colorPrimary">@color/primary</item>
    <!--   darker variant for the status bar and contextual app bars -->
    <item name="android:colorPrimaryDark">@color/primary_dark</item>
    <!--   theme UI controls like checkboxes and text fields -->
    <item name="android:colorAccent">@color/accent</item>
  </style>
</resources>
```

می توانید با مقداردهی android:statusBarColor ظاهر status bar (نوار نمایشگر وضعیت کلی دستگاه) را نیز تنظیم و به اصطلاح سبک دهی نمایید. به صورت پیش فرض، android:statusBarColor مقدار android:colorPrimaryDark را به ارث می برد.

3-13-5-سبک دهی و تنظیم ظاهر view های فردی و view group ها

از ویرایش 5.0 اندروید این امکان برای برنامه نویس فراهم شده تا android:theme را برای یک view تنظیم و مقداردهی کند. با استفاده از این attribute می توان theme و ظاهر کلی یک view و view های فرزند آن را تغییر داد.

تمرین: تعریف theme ها و استفاده از آن ها

هدف تمرین

در تمرین حاضر علاوه بر استفاده از theme های پیش فرض خود اندروید، theme های اختصاصی و دلخواه خود را تعریف خواهید نمود. می توانید برای این تمرین از یک اپلیکیشن آماده استفاده کنید و یا با استفاده از ویزارد ساخت پروژه محیط برنامه نویسی اندروید (IDE) یک پروژه ی جدید تعریف نمایید.

14-5-استفاده از theme از پیش تعریف شده و آماده

اپلیکیشن را جهت استفاده از android:Theme.Material.Light.DarkActionBar تنظیم نموده و آن را تست نمایید. سپس یک theme دیگر ایجاد کرده و آن را نیز تست کنید.

می توانید theme دلخواه خود را انتخاب نمایید.

تمرین: پیاده سازی یک theme اختصاصی

نکته: تیم توسعه دهندگان اندروید در Google قالب های آماده و الگوی های ساخت پروژه (template) را به طور مداوم تغییر می دهند. به دنبال این تغییر ممکن است فایل های سبک دهی (style resource) از قبل ایجاد شده باشند یا theme های پایه که تم های دیگر از آن ارث می برند (extend می شوند) به کلی تغییر کرده باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

ابتدا فایل styles.xml را برای ورژن اندرویدی که برنامه را بر روی آن تست می کنید، ایجاد نمایید. برای این منظور لازم است بخش qualifier ورژن (version qualifier) مربوطه را برای پوشه ای که دربردارنده ی فایل مورد نظر است، ارائه نمایید.

یک ثابت رنگ به نام my-color اضافه نموده که مقدار رنگ #b0b0ff را در فایل تعریف می کند.

AppTheme را ایجاد نمایید. با اضافه شدن این style می بایست مقدار خصوصیت (property) android:windowBackground با مقدار رنگی که شما تعریف می کنید جایگزین شود.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="my_color">#b0b0ff</color>
    <style name="AppTheme" parent="@android:style/Theme.Material.Light.DarkActionBar">
        <item name="android:windowBackground">@color/my_color</item>
    </style>
    <style name="ToolbarStyling">
        <item name="android:background">@color/colorPrimary</item>
    </style>
</resources>
```

این تم را به اپلیکیشن خود اعمال نمایید. هم اکنون رنگ پس زمینه ی اپلیکیشن بایستی تغییر کرده باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش چهارم :

آموزش ساخت wallpaper/تصاویر زنده در اندروید

آموزش حاضر به شرح نحوه ی ساخت تصاویر زنده برای سیستم عامل اندروید می پردازد. پروژه های این مبحث در محیط برنامه نویسی Eclipse 4.2، با ویرایش 1.6 زبان Java نوشته شده و مبتنی بر ورژن 4.1 سیستم عامل اندروید می باشد.

5-15-مرور کلی

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-15-5- Live Wallpapers / تصاویر زنده

Live wallpaper تصاویر پس زمینه ای پویا و تعاملی هستند که در صفحه ی اصلی سیستم عامل/home screen می نشینند. این تصاویر زنده از بسیاری جهات شبیه به دیگر اپلیکیشن های اندرویدی بوده و قادر هستند از اغلب قابلیت های سیستم اندروید مانند سایر برنامه ها بهره ببرند.

2-15-5-نحوه ی ساخت live wallpaper

به منظور ساخت live wallpaper لازم است یک فایل XML ایجاد نموده و ویژگی های کلی آن را در فایل مزبور اعلان نمایید. فایل نام برده می بایست توصیفی کلی از اپلیکیشن، یک پیش نمایش و لینک به تنظیمات اکتیویتی Activity که در قالب preferences تعریف می شود را دربرداشته باشد. شما بعده ها از طریق preferences می توانید activity را مطابق نیاز تنظیم نمایید.

سپس یک سرویس تعریف می کنید که کلاس WallpaperService را به ارث می برد. در واقع تمامی تصاویر زنده در سیستم اندروید از این کلاس پایه ارث بری کرده و مشتق می شوند. متد onCreateEngine() را پیاده سازی کرده و یک آبجکت از جنس
android.service.wallpaper.WallpaperService.Engine به عنوان خروجی از متد بازگردانی نمایید. این آبجکت ها رخدادهای مربوط به چرخه ی حیات (lifecycle event)، انیمیشن (متحرک سازی) و ترسیم live wallpaper را مدیریت می کنند. کلاس Engine متدهای مربوط به چرخه ی حیات را در اختیار توسعه دهنده قرار می دهد که از جمله ی آن می توان به
onCreate()،onSurfaceCreated()،onVisibilityChanged()،onOffsetsChanged ،
onTouchEvent() و onCommand() اشاره کرد.

برای استفاده از سرویس مورد نظر می بایست داخل فایل تنظیمات اپلیکیشن (manifest) مقدار android:permission را (داخل تگ service) برابر android.permission.BIND_WALLPAPER قرار داده و سپس داخل تگ intent-filter مقدار خصیصه ی android:name المان action را بر روی android.service.wallpaper.WallpaperService تنظیم نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

همچنین لازم است داخل فایل تنظیمات (AndroidManifest.xml) اعلان نمایید که اپلیکیشن از امکان android.software.live_wallpaper استفاده می کند. به دنبال این تنظیم، دستگاه هایی که از live wallpaper پشتیبانی نمی کنند، قابلیت نصب نرم افزار شما را نخواهد داشت.

3-15-5-استفاده از intent برای تنظیم wallpaper

می توانید یک دکمه تعریف کنید که با کلیک بر روی آن یک intent فعال شده و این intent سبب راه اندازی activity دیگری می شود. سپس در activity دوم امکان انتخاب Wallpaper را برای کاربر فراهم آورید.

```
// Button to set the Wallpaper
public void onClick(View view) {
    Intent intent = new Intent(          WallpaperManager.ACTION_CHANGE_LIVE_WALLPAPER);
intent.putExtra(WallpaperManager.EXTRA_LIVE_WALLPAPER_COMPONENT,
        new ComponentName(this, MyWallpaperService.class));
    startActivity(intent);
}
```

مثالی از پیاده سازی Live Wallpaper در اندروید

یک پروژه ی جدید به نام de.vogella.android.wallpaper ایجاد نمایید. لازم به ایجاد activity نیست.

پوشه ی /res/xml و فایل mywallpaper.xml نیز را ایجاد نمایید.

```
<?xml version="1.0" encoding="UTF-8"?>
<wallpaper
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:thumbnail="@drawable/icon"
    android:description="@string/wallpaper_description"
android:settingsActivity="de.vogella.android.wallpaper.MyPreferencesActivity"/>
```

این فایل یک پیش نمایش (در قالب یک تصویر کوچک) و توصیف از wallpaper را شامل می شود. می توانید یک لینک به activity که امکان تنظیم آن را برای کاربر فراهم می کند، در فایل XML لحاظ نمایید. این فایل محتوا به فایل تنظیمات اپلیکیشن (AndroidManifest.xml) متصل بوده و به آن لینک می شود.

با مقداردهی android:thumbnail در فایل حاضر قادر خواهید بود یک تصویر کوچک به عنوان پیش نمایش از live wallpaper فعال ارائه نمایید. مقداری که به این attribute انتساب می دهید در واقع به یک drawable اشاره می کند.

محتوای فایل AndroidManifest.xml را جهت تعریف سرویس MyWallpaperService به صورت زیر ویرایش نمایید. سپس تگ uses-feature را جهت افزودن قابلیت جدید به اپلیکیشن در این فایل اضافه کنید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="de.vogella.android.wallpaper"
   android:versionCode="1"
   android:versionName="1.0" >
   <application
      android:icon="@drawable/icon"
      android:label="@string/app_name" >
      <service
         android:name="MyWallpaperService"
         android:enabled="true"
         android:label="Wallpaper Example "
android:permission="android.permission.BIND_WALLPAPER" >
         <intent-filter>
            <action android:name="android.service.wallpaper.WallpaperService" >
            </action>
         </intent-filter>
         <meta-data
            android:name="android.service.wallpaper"
            android:resource="@xml/mywallpaper" >
         </meta-data>
      </service>
      <activity
         android:name=".MyPreferencesActivity"
         android:exported="true"
         android:label="@string/app_name"
android:theme="@android:style/Theme.Light.WallpaperSettings" >
      </activity>
      <activity
         android:name=".SetWallpaperActivity"
         android:label="@string/app_name"
android:theme="@android:style/Theme.Light.WallpaperSettings" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
```

```
    </application>
    <uses-sdk android:minSdkVersion="10" />
    <uses-feature
      android:name="android.software.live_wallpaper"
      android:required="true" >
    </uses-feature>
</manifest>
```

کلاس MyPoint را تعریف نموده و المان های ترسیم شده را در آن ذخیره نمایید.

```
package de.vogella.android.wallpaper;
public class MyPoint {
      String text;
      private int x;
      private int y;
      public MyPoint(String text, int x, int y) {
            this.text = text;
            this.x = x;
            this.y = y;
      }
}
```

یک activity جدید ایجاد نمایید. سپس فایل prefs.xml را در جهت نگهداری اطلاعات مربوط به تنظیمات در پوشه ی res/xml ایجاد کنید.

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android">
      <CheckBoxPreference android:key="touch"
            android:title="Enable Touch"></CheckBoxPreference>
      <EditTextPreference android:key="numberOfCircles"
            android:title="Number of Circles"></EditTextPreference>
</PreferenceScreen>
```

یک activity جدید به نام MyPreferencesActivity به همراه کلاس زیر ایجاد نمایید.

```
package de.vogella.android.wallpaper;
import android.os.Bundle;
import android.preference.Preference;
import android.preference.Preference.OnPreferenceChangeListener;
import android.preference.PreferenceActivity;
import android.widget.Toast;
public class MyPreferencesActivity extends PreferenceActivity {
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            addPreferencesFromResource(R.xml.prefs);
            // add a validator to the "numberofCircles" preference so that it only
            // accepts numbers
            Preference circlePreference = getPreferenceScreen().findPreference(
```

```
                "numberOfCircles");
        // add the validator          circlePreference.setOnPreferenceChangeListener(numberCheckListener);
    }
    /**
     * Checks that a preference is a valid numerical value
     */
    Preference.OnPreferenceChangeListener numberCheckListener = new OnPreferenceChangeListener() {
        @Override
        public boolean onPreferenceChange(Preference preference, Object newValue) {
            // check that the string is an integer
            if (newValue != null && newValue.toString().length() > 0
                    && newValue.toString().matches("\\d*")) {
                return true;
            }
            // If now create a message to the user                Toast.makeText(MyPreferencesActivity.this,
"Invalid Input",
                    Toast.LENGTH_SHORT).show();
            return false;
        }
    };
}
```

بدنه ی کلاس سرویس Wallpaper را به صورت زیر پیاده سازی کنید.

```
package de.vogella.android.wallpaper;
import java.util.ArrayList;
import java.util.List;
import android.content.SharedPreferences;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.os.Handler;
import android.preference.PreferenceManager;
import android.service.wallpaper.WallpaperService;
import android.view.MotionEvent;
import android.view.SurfaceHolder;
public class MyWallpaperService extends WallpaperService {
    @Override
    public Engine onCreateEngine() {
        return new MyWallpaperEngine();
    }
    private class MyWallpaperEngine extends Engine {
        private final Handler handler = new Handler();
        private final Runnable drawRunner = new Runnable() {
            @Override
            public void run() {
                draw();
            }
        };
        private List<MyPoint> circles;
        private Paint paint = new Paint();
```

```
        private int width;
        int height;
        private boolean visible = true;
        private int maxNumber;
        private boolean touchEnabled;
        public MyWallpaperEngine() {
            SharedPreferences prefs = PreferenceManager
.getDefaultSharedPreferences(MyWallpaperService.this);
            maxNumber = Integer
.valueOf(prefs.getString("numberOfCircles", "4"));
            touchEnabled = prefs.getBoolean("touch", false);
            circles = new ArrayList<MyPoint>();
            paint.setAntiAlias(true);
            paint.setColor(Color.WHITE);
            paint.setStyle(Paint.Style.STROKE);
            paint.setStrokeJoin(Paint.Join.ROUND);
            paint.setStrokeWidth(10f);
            handler.post(drawRunner);
        }
        @Override
        public void onVisibilityChanged(boolean visible) {          this.visible = visible;
            if (visible) {
                handler.post(drawRunner);
            } else {                    handler.removeCallbacks(drawRunner);
            }
        }
}

@Override
        public void onSurfaceDestroyed(SurfaceHolder holder) {
            super.onSurfaceDestroyed(holder);
            this.visible = false;
            handler.removeCallbacks(drawRunner);
        }
        @Override
        public void onSurfaceChanged(SurfaceHolder holder, int format,
                int width, int height) {
            this.width = width;
            this.height = height;
            super.onSurfaceChanged(holder, format, width, height);
        }
        @Override
        public void onTouchEvent(MotionEvent event) {
            if (touchEnabled) {
                float x = event.getX();
                float y = event.getY();
                SurfaceHolder holder = getSurfaceHolder();
                Canvas canvas = null;
                try {
                    canvas = holder.lockCanvas();
                    if (canvas != null) {                       canvas.drawColor(Color.BLACK);
                        circles.clear();
```

```
                        circles.add(new MyPoint(
                                String.valueOf(circles.size() + 1), x, y));
drawCircles(canvas, circles);
                    }
                } finally {
                    if (canvas != null)
                        holder.unlockCanvasAndPost(canvas);
                }
                super.onTouchEvent(event);
            }
        }
        private void draw() {
            SurfaceHolder holder = getSurfaceHolder();
            Canvas canvas = null;
            try {
                canvas = holder.lockCanvas();
                if (canvas != null) {
                    if (circles.size() >= maxNumber) {
                        circles.clear();
                    }
                    int x = (int) (width * Math.random());
                    int y = (int) (height * Math.random());
                    circles.add(new MyPoint(String.valueOf(circles.size() + 1),
                            x, y));
                    drawCircles(canvas, circles);
                }
            } finally {
                if (canvas != null)
                    holder.unlockCanvasAndPost(canvas);
            }
            handler.removeCallbacks(drawRunner);
            if (visible) {
                handler.postDelayed(drawRunner, 5000);
            }
        }
        // Surface view requires that all elements are drawn completely
        private void drawCircles(Canvas canvas, List<MyPoint> circles) {
            canvas.drawColor(Color.BLACK);
            for (MyPoint point : circles) {
                canvas.drawCircle(point.x, point.y, 20.0f, paint);
            }
        }
    }
```

یک کلاس Activity به صورت زیر ایجاد نمایید که ظاهر خود را از یک فایل layout حاوی آبجکت یا المان Button می خواند. این Button از خصوصیت onClick (property) برای اشاره و فراخوانی متد onClick استفاده می کند.

```
package de.vogella.android.wallpaper;
import android.app.Activity;
import android.app.WallpaperManager;
import android.content.ComponentName;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
public class SetWallpaperActivity extends Activity {
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main);
      }
      public void onClick(View view) {
            Intent intent = new Intent(
WallpaperManager.ACTION_CHANGE_LIVE_WALLPAPER);
            intent.putExtra(WallpaperManager.EXTRA_LIVE_WALLPAPER_COMPONENT,
                        new ComponentName(this, MyWallpaperService.class));
            startActivity(intent);
      }
}
```

پس از راه اندازی، اپلیکیشن بایستی به شما اجازه ی تنظیم wallpaper را بدهد. تصویر پس زمینه مشابه زیر خواهد بود. چنانچه قابلیت Touch را از طریق تنظیمات preferences فعال کرده باشید، در آن صورت می توانید با کلیک بر روی صفحه دایره های جاری را حذف نمایید. علاوه بر آن می توانید تعداد دایره هایی که در UI به نمایش در می آید را تعیین نمایید.

بخش پنجم :

Widget های home screen در اندروید/ابزارک های رابط کاربری صفحه ی اصلی

آموزش حاضر نحوه ی ایجاد widget ها در صفحه ی اصلی دستگاه اندروید را شرح می دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

16-5-شرح مفهوم Widgetدر اندروید

16-5-1-مروری بر AppWidgets

Widget ها اپلیکیشن های کوچک هستند و به راحتی بر روی یک میزبان که اغلب home screen یا lock screen (صفحه ی قفل نمایشگر) دستگاه اندروید هست، قابل جایگذاری می باشد. می توانید به widget ها به دید یک نمونه ی کوچک از کل اپلیکیشن و قابلیت های آن که از طریق صفحه ی اصلی دستگاه قابل دسترسی می باشد، نگاه کرد.

یک widget به عنوان بخشی از فرایند میزبان خود اجرا می شود. این امر لازمه ی آن است که widget از مجوزهای اپلیکیشن میزبان خود برخوردار بوده و آن ها را حفظ کنند.

Widget برای ساخت ظاهر و UI خود در صفحه اصلی دستگاه از کلاس RemoteViews استفاده می کند. کلاس RemoteView این قابلیت را دارد که با همان مجوزهای اپلیکیشن اصلی توسط فرایند دیگر راه اندازی شود. در واقع Widget ها با بهره گیری از این کلاس قادرند با مجوزهای اپلیکیشن اصلی که ابزارک ها به آن متصل هستند، اجرا شوند.

Widget ها برای ساخت ظاهر (UI) خود از broadcast receiver نیز کمک می گیرند. در حقیقت receiver یک آبجکت از جنس RemoteViews را با محتوای layout مربوطه پر می کند (آن را در این آبجکت inflate می کند). آبجکت نام برده سپس به اندروید تحویل داده شده و از آنجا در قالب widget مستقر در صفحه ی اصلی (home screen) برای کاربر به نمایش در می آید.

16-5-2-مراحل ساخت یک widget

1. یک فایل layout تعریف نمایید.
2. یک فایل XML (AppWidgetProviderInfo) ایجاد نموده که property ها و خصوصیت های widget همچون اندازه، زمان و دفعات تکرار بروز رسانی اطلاعات را مشخص کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

3. یک BroadcastReceiver ایجاد کنید که در ساخت رابط کاربری و ظاهر widget مورد استفاده قرار می گیرد.
4. تنظیمات و کانفیگ Widget را در فایل AndroidManifest.xml درج نمایید.
5. در صورت تمایل می توانید یک activity جهت انجام تنظیمات (configuration activity) تعریف نمایید که به مجرد اضافه شدن نمونه ای از widget به میزبان (widget host همچون صفحه ی اصلی)، صدا خورده می شود.

3-16-5-اندازه ی widget

قبل از ویرایش 3.1 اندروید، یک widget همیشه تعداد مشخصی خانه (cell) را در صفحه ی اصلی دستگاه به خود اختصاص می داد. هر خانه معمولا به اندازه ی یک آیکون فضا فراهم می کند.
برای محاسبه ی میزان فضای مورد نیاز یک widget می بایست از فرمول رو به رو استفاده کنید:
(Number of columns / rows)) * 74) – 2 . این اندازه بر حسب واحد dip (device independent pixel) محاسبه می شود. مقدار 2- برای اجتناب از خطاهای مربوط به گرد کردن لحاظ شده است.

از ویرایش 3.1 به بعد اندروید، کاربران این اجازه را دارند که اندازه ی widget را مطابق نیاز تنظیم نمایند. جهت فعال سازی این قابلیت، می توانید مقدار android:resizeMode را در فایل تنظیمات XML برای این widget برابر "horizontal|vertical"= قرار دهید.

5-17-ایجاد Broadcast receiver برای widget

1-17-5-ساخت و تنظیم widget

به منظور تعریف widget، لازم است یک broadcast receiver با intent filter ای که المان action آن بر روی

android.appwidget.action.APPWIDGET_UPDATE تنظیم شده باشد، ایجاد نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
<receiver
    android:icon="@drawable/icon"
    android:label="Example Widget"
    android:name="MyWidgetProvider" >
    <intent-filter >
        <action android:name="android.appwidget.action.APPWIDGET_UPDATE" />
    </intent-filter>
    <meta-data
      android:name="android.appwidget.provider"
      android:resource="@xml/widget_info" />
</receiver>
```

می توان به receiver یک label و آیکون تخصیص داد. این دو در لیست widget های موجود در launcher اندروید و به عنوان فایلی که با کلیک بر روی آن اپلیکیشن اجرا می شود، لیست می گردد.

می توانید با مقداردهی android:name="android.appwidget.provider"، برای widget مورد نظر meta-data تعریف نمایید. فایل تنظیماتی که meta data به آن اشاره می کند، تمامی تنظیمات و کانفیگ widget مورد نظر را شامل می شود. این تنظیمات می تواند مربوط به تناوب و دفعات بروز رسانی UI، اندازه و layout (ظاهر، چیدمان کلی) اولیه ی widget باشد.

```
<?xml version="1.0" encoding="utf-8"?>
<appwidget-provider xmlns:android="http://schemas.android.com/apk/res/android"
  android:initialLayout="@layout/widget_layout"
  android:minHeight="72dp"
  android:minWidth="146dp"
  android:updatePeriodMillis="1800000" >
</appwidget-provider>
```


2-17-5-View ها و layout های موجود و قابل استفاده

Widget در استفاده از کلاس های View با محدودیت مواجه است. برای تنظیم چیدمان و طرح کلی (layout) می توانید از FrameLayout، LinearLayout و RelativeLayout استفاده نمایید. به عنوان view ها می توانید از AnalogClock، Button، Chromometer، ImageButton، ImageView، ProgressBar و TextView استفاده کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

از ویرایش 3.0 اندروید به بعد view های بیشتری در اختیار توسعه دهنده قرار گرفته است که از میان آن ها می توان از GridView، ListView، StackView، ViewFlipper و AdapterViewFlipper نام برد.

برای استفاده از این adapter view ها (کلاس های مشتق شده از AdapterView) شما ملزوم به تعریف یک collection view widget هستید.

تنها پل ارتباطی و وسیله ی تعامل با view های یک widget از طریق event یا رخداد onClickListener می باشد. این event را می توان در widget ثبت نموده و به محض تعامل کاربر با widget، اتفاق افتادن آن را اعلان کرد.

AppWidgetProvider

پیاده سازی کلاس BroadcastReceiver معمولا اعضا و توابع کلاس AppWidgetProvider را به ارث می برد.

کلاس AppWidgetProvider متد ()onReceive را پیاده سازی کرده، اطلاعات لازم را استخراج می کند و در نهایت متدهای مدیریت چرخه ی حیات widget را صدا می زند.

از آنجایی که می توانید چندین نمونه از یک widget را به صفحه ی اصلی (home screen) اضافه نمایید، متدهای مربوط به مدیریت چرخه ی حیات widget به دو دسته تقسیم می شوند: 1. متدهایی که تنها برای اولین نمونه اضافه/حذف شده فراخوانده می شوند 2. متدهایی که به ازای هر نمونه از widget صدا خورده می شوند.

متد	شرح
onEnabled()	زمانی فراخوانی می گردد که نمونه ی widget برای اولین بار به home screen اضافه می شود.

تنها یکبار زمانی که آخرین نمونه ی widget از صفحه ی اصلی حذف می شود، فراخوانی می گردد.	onDisabled()
این متد به ازای هر بار بروز رسانی widget فراخوانی می شود. متد مذکور شناسه های appWidgetId آن widget هایی که باید با اطلاعات جدید بروز آوری شوند را به عنوان پارامتر ورودی در قالب آرایه دریافت می کند. توجه داشته باشید که این می تواند تمامی نمونه های AppWidget کلاس Provider را شامل شود یا صرفا زیر مجموعه ای از آن را دربرگیرد.	onUpdate()
با فراخوانی این متد نمونه ی widget از صفحه ی اصلی (widget instance) حذف می گردد.	onDeleted()

3-17-5-Receiver و پردازش ناهمزمان

Widget همان محدودیت هایی را در زمان اجرای برنامه دارد که یک broadcast receiver معمولی با آن مواجه می شود. برای مثال می بایست پردازش خود را نهایتا در عرض 5 ثانیه به اتمام برساند.

یک receiver می بایست عملیات طولانی در سرویس انجام داده و سپس widget ها را از آن سرویس بروز رسانی کند.

18-5-بروز رسانی widget

یک widget داده های جدید خود را بر اساس یک جدول زمانی و در فواصل زمانی مشخص دریافت می کند. برای بروز رسانی یک widget دو روش وجود دارد: 1. یکی مبتنی بر فایل تنظیمات XML بوده 2. دیگری توسط سرویس AlarmManager اندروید صورت می گیرد.

در فایل تنظیمات widget، شما می توانید یک فاصله ی زمانی مشخص تعیین نمایید که بروزرسانی بر اساس آن صورت گیرد. سیستم پس از گذشت این زمان مشخص بیدار شده و broadcast receiver را جهت بروز رسانی widget با داده های جدید فراخوانی می کند. کم ترین فاصله ی زمانی که آپدیت بر اساس آن رخ می دهد، حدودا 1800000 میلی ثانیه معادل 30 دقیقه می باشد. بدین معنی که widget هر 30 دقیقه آپدیت می شود.

AlarmManager این امکان را برای شما فراهم می کند تا در استفاده از منابع صرفه جویی نموده و widget خود را تعداد دفعات بیشتری بروز رسانی نمایید. برای استفاده از این روش، ابتدا یک سرویس تعریف نموده و سپس با استفاده از کلاس AlarmManager آن را طوری زمان بندی می کنید که در فواص زمانی معین اجرا شود. این سرویس متعاقبا widget را به طور مرتب بروز رسانی می کند.

توجه داشته باشید که بروز رسانی در فواصل زمانی بیشتر (با تعداد دفعات بروز رسانی بیشتر) ناگذیر سیستم را از حالت صرفه جویی در مصرف باتری خارج ساخته و در نتیجه widget مورد نظر انرژی بیشتری را مصرف خواهد نمود.

تمرین: پیاده سازی widget و بروز رسانی آن در فواصل زمانی معین

هدف تمرین

در زیر widget ای خواهید ساخت که یک عدد تصادفی را در UI نمایش می دهد. این عدد تصادفی هر 30 دقیقه یکبار بروز رسانی می شود. سپس یک onClickListener ثبت می کنید که با کلیک کاربر بر روی آن، بروز رسانی می شود.

Widget در نهایت ظاهری مشابه زیر خواهید داشت.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

ساخت پروژه و پیاده سازی widget

یک پروژه ی جدید اندروید به نام de.vogella.android.widget.example ایجاد نموده سپس یک activity جدید در پکیج de.vogella.android.widget.example ایجاد کنید.

یک فایل جدید به نام myshape.xml در پوشه ی <filename class="directory">/res/drawable ایجاد نمایید. این فایل در واقع drawable یا فایل تصویری ترسیم شونده ای که به عنوان پس زمینه در widget مورد استفاده قرار می گیرد را تعریف می نماید.

```
<?xml version="1.0" encoding="UTF-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
   android:shape="rectangle" >
   <stroke
      android:width="2dp"
      android:color="#FFFFFFFF" />
   <gradient
```

```
    android:angle="225"
    android:endColor="#DD2ECCFA"
    android:startColor="#DD000000" />
  <corners
    android:bottomLeftRadius="7dp"
    android:bottomRightRadius="7dp"
    android:topLeftRadius="7dp"
    android:topRightRadius="7dp" />
</shape>
```

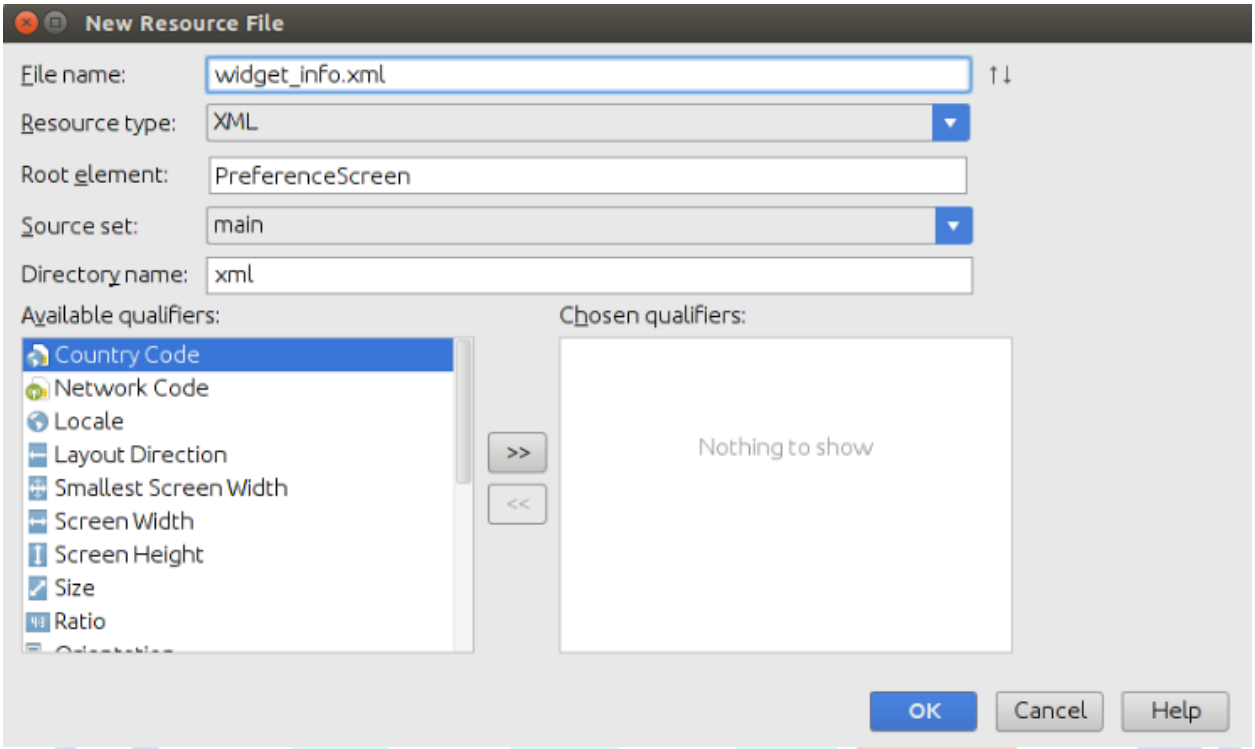
فایل widget_layout.xml با محتویات زیر را تحت پوشه ی <filename class="directory">res/layout ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:id="@+id/layout"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:layout_margin="8dip"
  android:background="@drawable/myshape" >
  <TextView
    android:id="@+id/update"
    style="@android:style/TextAppearance.Medium"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="center"
    android:gravity="center_horizontal|center_vertical"
    android:layout_margin="4dip"
    android:text="Static Text" >
  </TextView>
</LinearLayout>
```

یک فایل محتوا (resource) دیگر به نام widget_info.xml با کلیک راست بر روی پوشه ی res ایجاد نموده و سپس مسیر رو به رو را طی نمایید: New ▸ Android resource file.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330




```
<?xml version="1.0" encoding="utf-8"?>
<appwidget-provider      xmlns:android="http://schemas.android.com/apk/res/android"
  android:initialLayout="@layout/widget_layout"
  android:minHeight="72dp"
  android:minWidth="300dp"
  android:updatePeriodMillis="300000" >
</appwidget-provider>
```

یک کلاس receiver با پیاده سازی زیر ایجاد نمایید که با هر بار بروز رسانی widget صدا خورده می شود.

```
package de.vogella.android.widget.example;
import java.util.Random;
import android.app.PendingIntent;
import android.appwidget.AppWidgetManager;
import android.appwidget.AppWidgetProvider;
import android.content.ComponentName;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
import android.widget.RemoteViews;
public class MyWidgetProvider extends AppWidgetProvider {
      private static final String ACTION_CLICK = "ACTION_CLICK";
      @Override
      public void onUpdate(Context context, AppWidgetManager appWidgetManager,
                  int[] appWidgetIds) {
```

```
        // Get all ids
        ComponentName thisWidget = new ComponentName(context,
                MyWidgetProvider.class);
        int[] allWidgetIds = appWidgetManager.getAppWidgetIds(thisWidget);
        for (int widgetId : allWidgetIds) {
            // create some random data
            int number = (new Random().nextInt(100));
            RemoteViews remoteViews = new RemoteViews(context.getPackageName(),
                    R.layout.widget_layout);
            Log.w("WidgetExample", String.valueOf(number));
            // Set the text
            remoteViews.setTextViewText(R.id.update, String.valueOf(number));
            // Register an onClickListener
            Intent intent = new Intent(context, MyWidgetProvider.class);
intent.setAction(AppWidgetManager.ACTION_APPWIDGET_UPDATE);
intent.putExtra(AppWidgetManager.EXTRA_APPWIDGET_IDS, appWidgetIds);
            PendingIntent pendingIntent = PendingIntent.getBroadcast(context,
                    0, intent, PendingIntent.FLAG_UPDATE_CURRENT);
            remoteViews.setOnClickPendingIntent(R.id.update, pendingIntent);
            appWidgetManager.updateAppWidget(widgetId, remoteViews);
        }
    }
}
```

فایل AndroidManifest.xml را باز نموده و سپس widget خود را به صورت نمایش داده شده در کد زیر ثبت و تعریف نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="de.vogella.android.widget.example"
   android:versionCode="1"
   android:versionName="1.0" >
   <application
      android:icon="@drawable/icon"
      android:label="@string/app_name" >
      <receiver android:name="MyWidgetProvider" >
         <intent-filter >
            <action              android:name="android.appwidget.action.APPWIDGET_UPDATE" />
         </intent-filter>
         <meta-data
            android:name="android.appwidget.provider"
            android:resource="@xml/widget_info" />
      </receiver>
   </application>
   <uses-sdk android:minSdkVersion="8" />
</manifest>
```

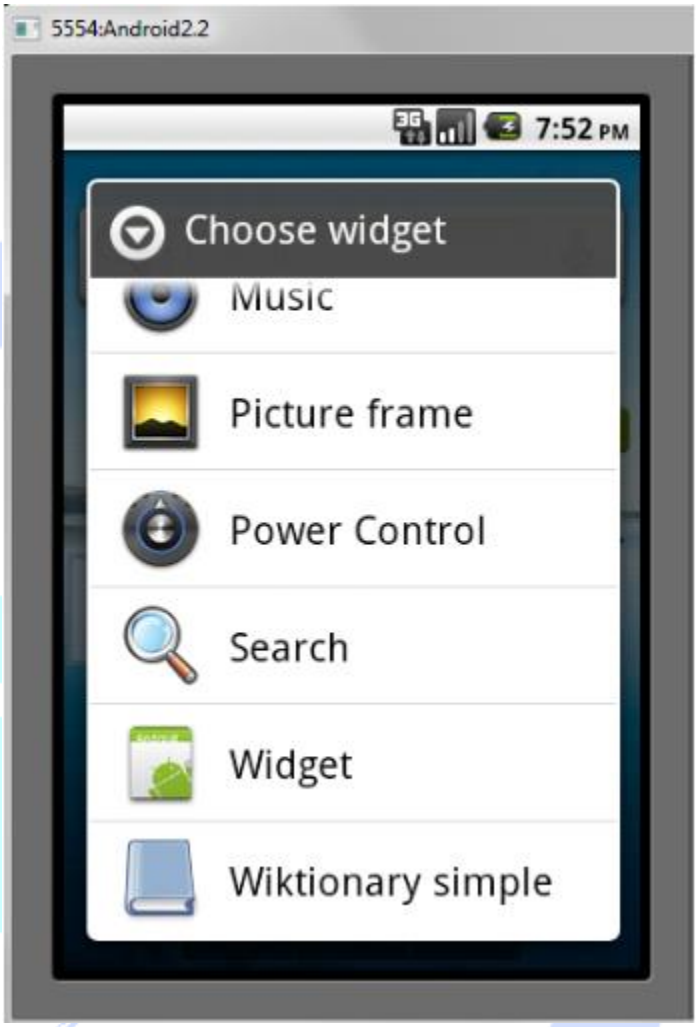
این attribute اعلان می کند که AppWidgetProvider برادکست ACTION_APPWIDGET_UPDATE را پذیرفته و نیز metadata مربوط به widget را مشخص می کند.

تست اپلیکیشن

اپلیکیشن خود را بر روی دستگاه اندروید نصب (deploy) نمایید. پس از نصب اپلیکیشن، با استفاده از launcher اندروید widget را بر روی صفحه ی اصلی دستگاه نصب و آن را تست کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



Collection View Widget-19-5

پیش از هر چیز لازم است درباره ی collection widget توضیح مختصری را در اختیار شما قرار دهیم. Collection widget ویژه ی نمایش مجموعه ای از المان های یکسان مورد استفاده قرار می گیرد. برای مثال می توان به تعدادی عکس از اپلیکیشن gallery، تعدادی مقاله از اپلیکیشن خبرخوان یا مجموعه ای از پیغام/ایمیل ها از یک اپلیکیشن ارتباطات سخن گفت. Collection widget معمولا به دو منظور مورد استفاده قرار می گیرد: 1. پیمایش در مجموعه ای از آیتم ها 2. باز کردن آیتمی از مجموعه جهت مشاهده ی صفحه ی اصلی و جزئیات آن. Collection widget دارای نوار اسکرول در کناره ی سمت راست بوده که به شما اجازه ی پیمایش به صورت عمودی را می دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

collection view widget به شما این امکان را می دهد تا از کلاس های ListView (جهت پیاده سازی لیست ساده)، stackview، GridView (لیستی مانند جدول خانه بندی شده) در بستر widget استفاده نمایید.

برای پیاده سازی collection view widget شما به دو فایل layout احتیاج دارید: 1. یکی برای widget 2. دیگری برای هر آیتم در widget collection.

آیتم های widget توسط نمونه ای از RemoteViewsFactory ساخته شده و با کلاس factory پر می شوند (Factory class = در مفهوم شی گرایی، آبجکتی که برای ساخت آبجکت دیگر تعبیه شده در اصطلاح factory خوانده می شود).

کلاس factory به عنوان یک الگو برای ساخت آبجکت های دیگر ایفای نقش می کند. این کلاس در اصل توسط یک سرویس اندروید که خود اعضا و توابع کلاس RemoteViewsFactory را به ارث می برد، در اختیار توسعه دهنده قرار می گیرد. لازم به ذکر است که سرویس مزبور برای فعالیت به تنظیم مجوز android.permission.BIND_REMOTEVIEWS در فایل تنظیمات اپلیکیشن نیاز دارد.

جهت متصل کردن view های خود به سرویس، لازم است متد ()onUpdate را در پیاده سازی widget بکار ببرید.

لازم است یک intent تعریف نموده که به سرویس مورد نظر دسترسی دارد (به آن اشاره داشته) و بعد متد setRemoteAdapter را در سطح کلاس RemoteViews بکار ببرید.

```
Intent intent = new Intent(context, YourRemoteViewsService.class);
intent.putExtra(AppWidgetManager.EXTRA_APPWIDGET_ID, appWidgetId);
views.setRemoteAdapter(
    apppWidgetId,
    R.id.widget_your_id_to_collectionview,
    intent)
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

20-5-فعال سازی یک widget برای قفل نمایشگر دستگاه (lock screen)

از اندروید 4.2 این امکان فراهم شده تا widget های صفحه ی اصلی دستگاه را در صفحه ی قفل (lock screen) نیز جایگذاری نمایید. برای اینکه widget قابلیت قرار گیری در صفحه ی قفل را داشته باشد، لازم است مقدار خصیصه (attribute) android:widgetCategory را در فایل AppWidgetProviderInfo برابر android:widgetCategory قرار دهید. کد زیر نمونه ای را به نمایش می گذارد.

```
<appwidget-provider xmlns:android="http://schemas.android.com/apk/res/android"
android:widgetCategory="keyguard|home_screen"
    ...
>
...
</appwidget-provider>
```

در مثال جاری، یک widget تعریف می کنید که قابلیت قرار گیری در صفحه ی اصلی و صفحه ی قفل نمایشگر را داشته باشد. اگر اپلیکیشن خود را هم اکنون recompile و راه اندازی نمایید، خواهید توانست widget را در همین برهه از زمان در صفحه ی قفل نمایشگر قرار دهید.

همچنین می توانید widget category را در زمان اجرای برنامه تشخیص دهید (اینکه آیا زمانی که widget در صفحه ی قفل قرار می گیرد با زمانی که در صفحه ی اصلی نمایش داده می شود دارای ظاهر متفاوتی باشد). برای این منظور، در متد AppWidgetProvider.onUpdate() شما می توانید با استفاده از کد زیر category option یک widget را بررسی نمایید.

```
Bundle options = appWidgetManager.getAppWidgetOptions(widgetId);
int category = options.getInt(AppWidgetManager.OPTION_APPWIDGET_HOST_CATEGORY, -1);
boolean isLockScreen = category == AppWidgetProviderInfo.WIDGET_CATEGORY_KEYGUARD;
```

با این روش شما می توانید در زمان اجرا تصمیم بگیرید (مشخص کنید) آیا widget ای که اپلیکیشن شما در UI ارائه می دهد بایستی به هنگام نمایش در صفحه ی قفل ظاهری متفاوت داشته باشد یا خیر.

درست مانند زمانی که شما از خصیصه android:initialLayout (attribute) جهت تعریف layout اولیه و چیدمان widget های صفحه ی اصلی استفاده می کنید، android:initialKeyguardLayout را نیز برای تعریف ظاهر widget در صفحه ی قفل نمایشگر، داخل فایل AppWidgetProviderInfo بکار می برید. این layout بلافاصله پس از اینکه widget اضافه می شود، پدیدار شده و زمانی که widget به معنای واقعی مقدار دهی اولیه/راه اندازی می شود با layout و ظاهر اصلی جایگزین می گردد.

تمرین: بروز رسانی widget از طریق یک سرویس

تمرین جاری نحوه ی استفاده از یک سرویس جهت بروز رسانی widget با داده های جدید را به نمایش می گذارد.

کلاس زیر را در پروژه ی خود ایجاد نمایید.

```
package de.vogella.android.widget.example;
import java.util.Random;
import android.app.PendingIntent;
import android.app.Service;
import android.appwidget.AppWidgetManager;
import android.content.ComponentName;
import android.content.Intent;
import android.os.IBinder;
import android.util.Log;
import android.widget.RemoteViews;
public class UpdateWidgetService extends Service {
	private static final String LOG = "de.vogella.android.widget.example";
	@Override
	public void onStart(Intent intent, int startId) {
		AppWidgetManager appWidgetManager = AppWidgetManager.getInstance(this
				.getApplicationContext());
		int[] allWidgetIds = intent
				.getIntArrayExtra(AppWidgetManager.EXTRA_APPWIDGET_IDS);
//		ComponentName thisWidget = new ComponentName(getApplicationContext(),
//				MyWidgetProvider.class);
//		int[] allWidgetIds2 = appWidgetManager.getAppWidgetIds(thisWidget);
		for (int widgetId : allWidgetIds) {
			// create some random data
			int number = (new Random().nextInt(100));
			RemoteViews remoteViews = new RemoteViews(this
					.getApplicationContext().getPackageName(),
					R.layout.widget_layout);
			Log.w("WidgetExample", String.valueOf(number));
```

```
            // Set the text
            remoteViews.setTextViewText(R.id.update,
                    "Random: " + String.valueOf(number));
            // Register an onClickListener
            Intent clickIntent = new Intent(this.getApplicationContext(),
                    MyWidgetProvider.class);
            clickIntent.setAction(AppWidgetManager.ACTION_APPWIDGET_UPDATE);
            clickIntent.putExtra(AppWidgetManager.EXTRA_APPWIDGET_IDS,
                    allWidgetIds);
            PendingIntent pendingIntent = PendingIntent.getBroadcast(
                    getApplicationContext(), 0, clickIntent,
                    PendingIntent.FLAG_UPDATE_CURRENT);
            remoteViews.setOnClickPendingIntent(R.id.update, pendingIntent);
            appWidgetManager.updateAppWidget(widgetId, remoteViews);
        }
        stopSelf();
        super.onStart(intent, startId);
    }
    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }
}
```

این کلاس را به عنوان یک سرویس (با تگ service) در فایل تنظیمات AndroidManifest.xml خود تعریف نمایید.

```
<service android:name=".UpdateWidgetService"></service>
```

MyWidgetProvider را به صورت زیر ویرایش نمایید. در حال حاضر کد زیر سرویس را ساخته و آن را راه اندازی می کند.

```
package de.vogella.android.widget.example;
import android.appwidget.AppWidgetManager;
import android.appwidget.AppWidgetProvider;
import android.content.ComponentName;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
public class MyWidgetProvider extends AppWidgetProvider {
    private static final String LOG = "de.vogella.android.widget.example";
    @Override
    public void onUpdate(Context context, AppWidgetManager appWidgetManager,
            int[] appWidgetIds) {
        Log.w(LOG, "onUpdate method called");
        // Get all ids
        ComponentName thisWidget = new ComponentName(context,
```

```
                MyWidgetProvider.class);
        int[] allWidgetIds = appWidgetManager.getAppWidgetIds(thisWidget);
        // Build the intent to call the service
        Intent intent = new Intent(context.getApplicationContext(),
                UpdateWidgetService.class);
        intent.putExtra(AppWidgetManager.EXTRA_APPWIDGET_IDS, allWidgetIds);
        // Update the widgets via the service
        context.startService(intent);
    }
}
```

پس از فراخوانی، این سرویس تمامی widget ها را بروز رسانی می کند. در واقع می توانید با کلیک بر روی یکی از widget های جاری، تمامی widget ها را یکجا بروز آوری نمایید.

بخش ششم :

ساخت و پیاده سازی view های سفارشی و ترکیبی در اندروید

این آموزش به شرح ساخت view های اختصاصی و ترکیبی در اندروید می پردازد.

View-21-5 های اختصاصی

View–5–21–1 های پیش فرض محیط اندروید

چارچوب نرم افزاری (framework) اندروید تعدادی view درون ساخته و پیش فرض ارائه می دهد که توسعه دهنده می تواند از آن ها به صورت آماده استفاده نماید. کلاسی که تمامی view ها (همان کنترل ها و المان های رابط کاربری) از آن مشتق و ارث بری می شوند، View می باشد.

View ها موظف هستند خود و تمامی المان های داخل خود (view های فرزند که در ViewGroup با آن مواجه می شوید) را اندازه گیری، طرح بندی (layout) و ترسیم نمایند.

View ها همچنین می بایست اطلاعات مربوط به وضعیت (state) UI را ذخیره کرده و event هایی که با تعامل کاربر با المان های رابط کاربری (لمس نمایشگر) فعال می شوند را مدیریت نمایند.

البته توسعه دهندگان این امکان را هم دارند که view های اختصاصی تعریف نموده و آن را در اپلیکیشن خود بکار ببرند.

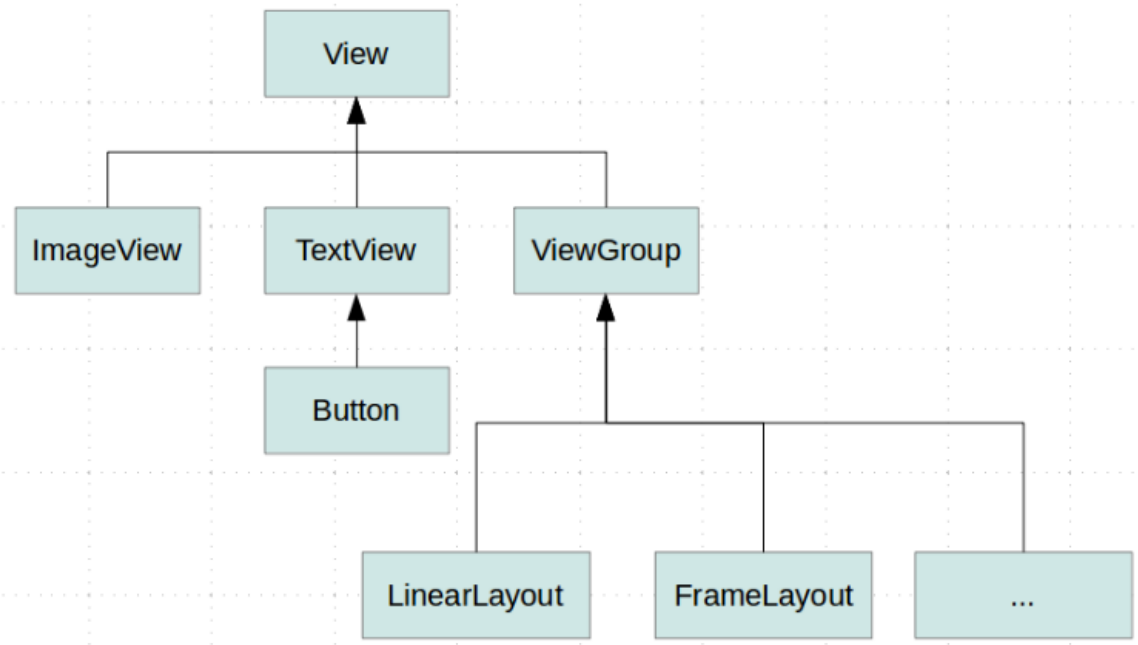
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

به منظور تعریف view های سفارشی و دلخواه خود می توانید به یکی از روش های زیر اقدام نمایید:

- Compound view – تلفیق view ها با اتصال پیش فرض (default wiring)
- Custom view – ساخت view های اختصاصی و دلخواه خود
- با ارث بری از یک view ی آماده همچون Button
- با ارث بری از کلاس View

تصویر زیر زنجیره ی ارث بری و سلسله مراتب view های پیش فرض اندروید را به نمایش می گذارد.



View های سفارشی معمولا با هدف ارائه ی تجربه ی کاربری ویژه و دلخواه که با view های پیش فرض و آماده ی خود اندروید امکان پذیر نیست ساخته می شوند. بعلاوه پیاده سازی view های اختصاصی این امکان را برای توسعه دهنده فراهم می کند تا با ابتکار خود کارایی را افزایش دهد برای مثال در خصوص پیاده سازی layout اختصاصی، برنامه نویس می تواند layout manager را با توجه به نیاز خود تنظیم و بهینه نماید.

2-21-5-اندروید چگونه view hierarchy را ترسیم می نماید!

پس از اینکه انتخاب و تمرکز UI بر روی یک activity متمرکز می شود، در همان لحظه بایستی root node (گره یا عنصر اصلی و آغازین) سلسه مراتب layout خود را در اختیار سیستم اندروید قرار دهد. پس از آن سیستم اندروید فرایند ترسیم را آغاز می نماید.

روند ترسیم layout دو مرحله را پشت سر می گذارد:

- measuring pass (مرحله ی سنجش و اندازه گیری) – توسط متد `measure(int, int)` پیاده سازی می شود. این مرحله به صورت پیمایش از بالای سلسله مراتب view تا پایین آن اتفاق می افتد. هر view اندازه های خود را ذخیره می کند.
- layout pass (مرحله ی تنظیم چیدمان و طرح بندی) – این مرحله توسط متد layout(int, int, int, int) پیاده سازی می شود. پیمایش مرحله ی جاری نیز از بالای سلسله مراتب view به پایین آن رخ می دهد. طی این مرحله هر layout manager مسئول چیدمان و موقعیت دهی فرزندان خود (view های داخل خود) می باشد. لازم به ذکر است که متد فوق برای تنظیم موقعیت view ها از اندازه های بدست آمده در مرحله ی اول استفاده می کند.

نکته: مرحله ی measure (اندازه گیری) و layout (چیدمان) همیشه همزمان اتفاق می افتند.

Layout manager می تواند مرحله ی اندازه گیری را چندین بار اجرا نماید. برای مثال، LinearLayout از attribute ای به نام weight پشتیبانی می کند که فضای خالی باقی مانده را بین view ها پخش نموده و RelativeLayout تمامی المان یا view های فرزند خود را چندین بار اندازه گیری می کند تا constraint های تعیین شده در فایل layout همگی برآورده شوند.

View یا activity هر یک می توانند مرحله ی اندازه ی گیری و چیدمان را با فراخوانی متد requestLayout() راه اندازی نماید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

پس از انجام محاسبات مربوط به اندازه گیری و تنظیم چیدمان المان ها، view ها اقدام به ترسیم خود می نمایند. جهت راه اندازی این عملیات کافی است متد invalidate() از کلاس View فراخوانی شود.

3-21-5-استفاده از view های جدید در فایل های layout

View های اختصاصی و ترکیبی (پیچیده) می توانند در فایل های layout تعریف و استفاده شوند. برای نیل به این هدف لازم است اسم view ها را به صورت تمام و کامل در فایل ذکر شده قید نمایید. برای مثال می بایست اسم کلاس و پکیج (پوشه ی اصلی پروژه) را ارائه کنید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <Button
      android:id="@+id/button1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Button" />
   <de.vogella.android.ownview.MyDrawView
      android:id="@+id/myDrawView1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content" />
</LinearLayout>
```

توجه: در صورت تمایل می توانید name space خود را در فایل layout اعلان نمایید، مانند name space اندروید.

4-21-5-تهیه ی تصویر آنی (screenshot) از view ها

تمامی کلاس های View این قابلیت را دارند از وضعیت و ظاهر کنونی خود تصویر آنی تهیه نمایند.

```
# Build the Drawing Cache
view.buildDrawingCache();
# Create Bitmap
Bitmap cache = view.getDrawingCache();
```

```
# Save Bitmap
saveBitmap(cache);
view.destroyDrawingCache();
```

view-22-5 های ترکیبی (Compound views)

View های ترکیبی (یا کامپوننت های ترکیبی) به Viewهای گفته می شوند که از ترکیب چند view دیگر پدیده آمده باشد.

Compound view ها به شما این امکان را می دهند تا API ها و توابع اختصاصی خود را جهت بروز رسانی و کوئری گرفتن از اطلاعات مربوط به وضعیت view بکار ببرید.

برای این control یک فایل layout تعریف می کنید و آن را به compound view خود تخصیص می دهید. در پیاده سازی compound view، بایستی ارتباط متقابل view ها را تعریف کرده باشید. ابتدا یک فایل layout تعریف می کنید که اعضا و توابع کلاس ViewGroup مربوطه را به ارث می برد. در این کلاس فایل layout را بارگذاری نموده (inflate) و منطق (کد) اتصال و ارتباط View را پیاده سازی می نمایید.

نکته: برای افزایش کارایی و سرعت اجرا، بهتر است view سفارشی که از کلاس View ارث بری کرده ایجاد نمایید. از این طریق می توانید سلسله مراتب view خود و زیرشاخه های آن را به صورت خطی نمایش دهید (flatten view). چرا که در این حالت ترسیم view به پیمایش کمتری نیاز دارد و در صورتی که به شکل درستی پیاده سازی شود، بسیار سریع تر اجرا خواهد شد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

5-23-ساخت view های اختصاصی

5-23-1-پیاده سازی view های اختصاصی

با ارث بری از کلاس View یا یکی از کلاس های مشتق آن (subclass)، شما می توانید view دلخواه خود را ایجاد نمایید.

برای ترسیم view می توانید متد ()onDraw را بکار ببرید. در این متد یک آبجکت Canvas به عنوان ورودی دریافت می کنید. آبجکت نام برده به شما امکان می دهد تا عملیات ترسیم همچون کشیدن خط، دایره، درج متن یا bitmap را در سطح آن انجام دهید. در صورتی که view مجددا ترسیم گردد، شما می توانید متد ()invalidate را فراخوانی کنید که خود سبب فراخوانی متد ()onDraw این view می شود.

نکته: در صورت تعریف view های اختصاصی، حتما کلاس ViewConfiguration را بررسی نمایید چرا که این کلاس تعدادی ثابت (constant) درون ساخته دارد که شما می توانید برای تعریف view ها مورد استفاده قرار دهید.

برای ترسیم Views توسعه دهندگان اغلب از 2D Canvas API استفاده می کنند.

5-23-2-اندازه گیری view ها

Layout manager متد ()onMeasure از view مورد نظر را صدا می زند. View سپس پارامترهای layout را از layout manager دریافت می کند. layout manager وظیفه ی تعیین اندازه ی تمامی view های داخل خود را بر عهده دارد.

View می بایست متد setMeasuredDimension (int, int) را با نتیجه مورد نظر فراخوانی نماید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

3-23-5-تعریف layout manager اختصاصی

برای تعریف layout manager اختصاصی خود می توانید اعضا و توابع کلاس ViewGroup را به ارث ببرید. یا انجام کار فوق این امکان برای شما فراهم می شود تا layout manger های کارامد و بهینه تری را پیاده سازی نموده یا افکت های دیداری را خلق کنید که در محیط (platform) اندروید وجود ندارد.

یک layout manager سفارشی، به تبع پیاده سازی توابع ()onMeasure و ()onLayout را بازنویسی (override) نموده و عملیات محاسبه ی (اندازه ی) المان های فرزند خود را نیز به صورت اختصاصی انجام می دهد. برای مثال ممکن است استفاده از ویژگی سنگین و زمان بر layout_weight از کلاس LinearLayout را کنار بگذارد.

نکته: به منظور محاسبه ی اندازه ی المان محصور (فرزند) کافی است متد ()measureChildWithMargins از کلاس ViewGroup را فراخوانی نمایید.

توصیه می شود تمامی پارامترهای اضافی layout را در یک کلاس داخلی درون نمونه ی پیاده سازی شده ی خود از ViewGroup قرار دهید. برای مثال کلاس LayoutParams به عنوان یک ویژگی در کلاس ViewGroup ویژگی های command، layout و parameters را درون خود کپسوله کرده است و کلاس LinearLayout نیز علاوه بر این ویژگی ها، ویژگی layout_weight را به این ویژگی اضافه نموده است.

Life Cycle-24-5

1-24-5-event ها و توابع مربوط به مدیریت چرخه ی حیات window

یک view زمانی نمایش داده می شود که به layout hierarchy متصل باشد. layout hierarchy نیز خود به تبع به window وصل می باشد. هر view تعداد زیادی تابع hook مدیریت چرخه ی حیات دارد.

متد ()onAttachedToWindow زمانی فراخوانی می شود که پنجره در حافظه بارگذاری شده و در دسترس باشد.

متد onDetachedFromWindow() زمانی فراخوانی می شود که view از میزبان (parent) خود جدا شده باشد (و البته میزبان نیز خود به یک window وصل باشد). این اتفاق، برای مثال، زمانی رخ می دهد که activity (به عنوان مثال با صدا خورده شدن متد finished()) یا view بازیافت شده باشند.

متد onAttachedToWindow() زمانی فراخوانی می شود که یک پنجره در دسترس باشد.

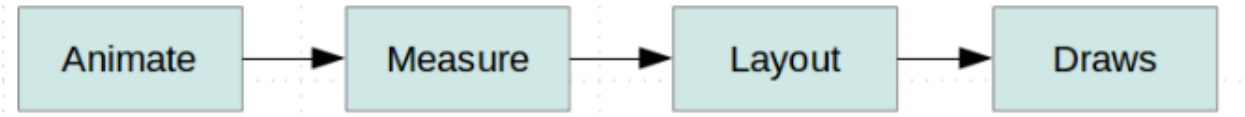
متد onDetachedFromWindow() از والد View پس از حذف استفاده می نماید(حتی اگر به یک پنجره دیگر پیوست باشد) . این رویداد نیز برای Activity به صورت باز پس گیری (حتی وقتی متد finished() فراخوانی شده باشد) روی می دهد.

متد onDetachedFromWindow() را می توان جهت متوقف کردن انیمیشن ها و پاک سازی و آزاد نمودن منابع مورد استفاده ی view فراخوانی نمود.

1-24-5-Event های مربوط به life cycle به صورت ترتیبی/پیمایشی (Traversal life cycle event)

رخدادهای چرخه ی حیات به ترتیب عبارت اند از:

1. Animate
2. Measure
3. Layout
4. Draw



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

لازم است view ها با نحوه ی اندازه گیری و چیدمان (layout) خود آشنا باشند. متد requestLayout() به view اعلان می کند که خود را اندازه گرفته و موقعیت دهی (layout) نمایند. از آنجایی که این عملیات ممکن است layout دیگر view ها را نیز متاثر کند، متد requestLayout() پدر (parent) نیز فراخوانی می شود.

نکته: زمانی که چندین layout را داخل هم قرار داده و آن ها را زیاد تودرتو می نمایید، این امر سبب به اجرا در آمدن الگوریتم فراخوانی بازگشتی می شود. حال زمانی که عمق ساختار درختی یا سلسله مراتب زیاد بوده و در این حین لازم باشد سلسله مراتب مجددا محاسبه شود، عملیات اندازه گیری و موقعیت دهی ناگذیر سنگین و طولانی خواهد بود.

متد onMeasure() اندازه ی view و المان های محصور آن (view های فرزند) را مشخص می کند. سپس با فراخوانی setMeasureDimension() داخل بدنه ی خود و قبل از اجرای دستور return، اندازه ی آن ها را تنظیم می نماید.

متد onLayout() تمامی view ها را بر اساس نتیجه یا خروجی متد onMeasure() موقعیت دهی می نماید. این فراخوانی معمولا تنها یکبار انجام می شود در حالی که onMeasure() می تواند چندین بار صدا خورده شود.

2-24-5-چرخه ی حیات activity

View ها به event های چرخه ی حیات activity ها دسترسی ندارند. اگر لازم باشد که view ها از رخداد این event ها باخبر شوند، در آن صورت می بایست یک interface در view ایجاد نمایید (interface مربوطه را در بدنه ی کلاس view پیاده سازی کرده) و متدهای مربوط به مدیریت چرخه ی حیات activity را از طریق آن فراخونی نمایید.

25-5-تعریف attribute های بیشتر برای view های اختصاصی

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

شما می توانید attribute های اضافی بر سازمان برای view های ترکیبی و اختصاصی خود تعریف نمایید. برای این منظور ابتدا بایستی یک فایل به نام attrs.xml در پوشه ی res/values خود ایجاد نمایید. در زیر مثالی را می بینید که برای view جدیدی به نام ColorOptionsView تعدادی attribute جدید تعریف شده است.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <declare-styleable name="ColorOptionsView">
      <attr name="titleText" format="string" localization="suggested" />
      <attr name="valueColor" format="color" />
   </declare-styleable>
</resources>
```

جهت استفاده از attribute های نام برده در فایل layout، می بایست آن ها را در بخش header فایل XML اعلان نمایید. در کد زیر این کار توسط دستور xmlns:custom صورت گرفته است. مقدار attribute هایی که زیر این دستور درج می شوند، متعاقبا به view مورد نظر نیز اعمال می شوند.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
<!-- define new name space for your attributes -->
   xmlns:custom="http://schemas.android.com/apk/res/com.vogella.android.view.compoundview"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
>
<!-- Assume that this is your new component. It uses your new attributes -->
      <com.vogella.android.view.compoundview.ColorOptionsView
         android:layout_width="match_parent"
         android:layout_height="?android:attr/listPreferredItemHeight"
         custom:titleText="Background color"
         custom:valueColor="@android:color/holo_green_light"
          />
</LinearLayout>
```

مثال زیر نشان می دهد چگونه کامپوننت های شما می توانند به این attribute ها دسترسی داشته باشند.

```
package com.vogella.android.view.compoundview;
import android.content.Context;
import android.content.res.TypedArray;
import android.util.AttributeSet;
```

```
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.View;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;
public class ColorOptionsView extends View {
	private View mValue;
	private ImageView mImage;
	public ColorOptionsView(Context context, AttributeSet attrs) {
		super(context, attrs);
		TypedArray a = context.obtainStyledAttributes(attrs,
				R.styleable.Options, 0, 0);
		String titleText = a.getString(R.styleable.Options_titleText);
		int valueColor = a.getColor(R.styleable.Options_valueColor,
android.R.color.holo_blue_light);
		a.recycle();
		// more stuff
	}
}
```

تمرین: ساخت و پیاده سازی یک view ترکیبی

ایجاد پروژه

یک پروژه ی جدید اندروید با داده ها و مقادیر زیر ایجاد نمایید.

جدول پروژه ی جدید اندرویدی(1,1)	
Property	Value
Testing (تست)	Table width
Application Name (اسم اپلیکیشن)	Compound view example
Project Name (اسم پروژه)	com.vogella.android.customview.compoundview
Package name (اسم پوشه ی حاوی کلاس ها)	com.vogella.android.customview.compoundview
API (Minimum, Target, Compile with) پایین ترین ویرایش اندروید که برنامه بایستی بر روی آن قابل اجرا باشد، ویرایشی که برنامه برای آن	Latest

طراحی شده، ورژنی که برنامه با آن کامپایل می شود	
Template (قالب آماده و پیاده سازی پروژه)	Empty Activity
Activity	MainActivity
Layout	activity_main

1-25-5-تعریف و استفاده از attribute های جدید

یک فایل حامل attribute های جدید به نام attr.xml را داخل پوشه ی res/values ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <declare-styleable name="Options">
      <attr name="titleText" format="string" localization="suggested" />
      <attr name="valueColor" format="color" />
   </declare-styleable>
</resources>
```

محتوای فایل layout ای که توسط کلاس activity برای نمایش در UI فراخوانی می شود را به صورت زیر ویرایش نمایید.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   xmlns:custom="http://schemas.android.com/apk/res/com.vogella.android.view.compoundview"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   android:showDividers="middle"
   android:divider="?android:attr/listDivider"
   tools:context=".MainActivity" >
      <com.vogella.android.view.compoundview.ColorOptionsView
         android:id="@+id/view1"
         android:layout_width="match_parent"
android:layout_height="?android:attr/listPreferredItemHeight"
android:background="?android:selectableItemBackground"
         android:onClick="onClicked"
         custom:titleText="Background color"
         custom:valueColor="@android:color/holo_green_light"
          />
      <com.vogella.android.view.compoundview.ColorOptionsView
         android:id="@+id/view2"
         android:layout_width="match_parent"
  android:layout_height="?android:attr/listPreferredItemHeight"
android:background="?android:selectableItemBackground"
```

```
        android:onClick="onClicked"
        custom:titleText="Foreground color"
        custom:valueColor="@android:color/holo_orange_dark"
         />
</LinearLayout>
```

2-25-5-ساخت View ترکیبی

فایل layout زیر به نام view_color_options را برای view اختصاصی خود ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<merge xmlns:android="http://schemas.android.com/apk/res/android" >
   <TextView
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="1"
      android:layout_centerVertical="true"
        android:layout_marginLeft="16dp"
      android:textSize="18sp"
      />
      <View
        android:layout_width="26dp"
        android:layout_height="26dp"
        android:layout_centerVertical="true"
        android:layout_marginLeft="16dp"
        android:layout_marginRight="16dp"
        />
   <ImageView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
        android:layout_marginRight="16dp"
      android:layout_centerVertical="true"
      android:visibility="gone"
      />
</merge>
```

View اختصاصی خود را به صورت زیر پیاده سازی نمایید.

```
package com.vogella.android.customview.compoundview;
import com.vogella.android.view.compoundview.R;
import android.content.Context;
import android.content.res.TypedArray;
import android.util.AttributeSet;
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.View;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;
public class ColorOptionsView extends LinearLayout {
      private View mValue;
```

```
        private ImageView mImage;
        public ColorOptionsView(Context context, AttributeSet attrs) {
                super(context, attrs);
                TypedArray a = context.obtainStyledAttributes(attrs,
                                R.styleable.ColorOptionsView, 0, 0);
                String titleText = a.getString(R.styleable.ColorOptionsView_titleText);
                int valueColor = a.getColor(R.styleable.ColorOptionsView_valueColor,
android.R.color.holo_blue_light);
                a.recycle();
                setOrientation(LinearLayout.HORIZONTAL);
                setGravity(Gravity.CENTER_VERTICAL);
                LayoutInflater inflater = (LayoutInflater) context
.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
                inflater.inflate(R.layout.view_color_options, this, true);
                TextView title = (TextView) getChildAt(0);
                title.setText(titleText);
                mValue = getChildAt(1);
                mValue.setBackgroundColor(valueColor);
                mImage = (ImageView) getChildAt(2);
        }
        public ColorOptionsView(Context context) {
                this(context, null);
        }
        public void setValueColor(int color) {
                mValue.setBackgroundColor(color);
        }
        public void setImageVisible(boolean visible) {
                mImage.setVisibility(visible ? View.VISIBLE : View.GONE);
        }
}
```

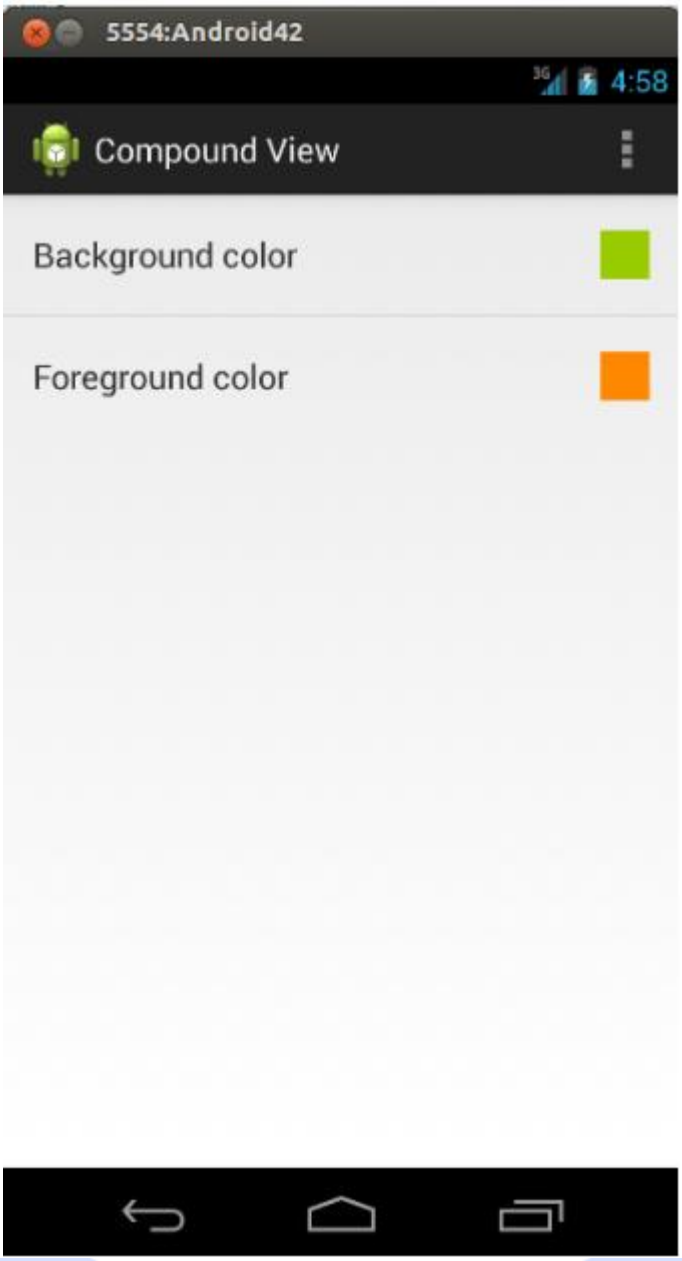
3-25-5-تنظیم و ویرایش activity

بدنه ی activity را به صورت زیر ویرایش نموده و سپس اپلیکیشن خود را اجرا نمایید.

```
package com.vogella.android.customview.compoundview;
import com.vogella.android.view.compoundview.R;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.View;
import android.widget.Toast;
public class MainActivity extends Activity {
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
        }
```

```
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }
    public void onClicked(View view) {
        String text = view.getId() == R.id.view1 ? "Background" : "Foreground";
        Toast.makeText(this, text, Toast.LENGTH_SHORT).show();
    }
}
```

اپلیکیشن در زمان اجرا می بایست ظاهری مشابه زیر داشته باشد.



Canvas API-26-5 (توابع مرتبط با کلاس Canvas)

1-26-5-شرح مفهوم Canvas API

Canvas API به شما اجازه می دهد تا افکت های دیداری و گرافیکی منحصر بفردی خلق نمایید.

کلاس Canvas تعدادی تابع جهت ترسیم اشکال در اختیار دارد. برای انجام عملیات ترسیم، توسعه دهنده به 4 مولفه ی ابتدایی احتیاج دارد: 1. یک bitmap جهت میزبانی پیکسل ها 2. یک Canvas

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

برای نگه داشتن توابع draw (جهت ترسیم و نوشتن در bitmap) 3. یک کلاس جهت ترسیم مانند Rect، Path، text، Bitmap 4. یک کلاس paint به منظور تعریف رنگ ها و style های المان های گرافیکی.

در واقع شما بر روی سطح Bitmap اشکال را ترسیم می کنید، توابع ترسیم اشکال را از کلاس Canvas وام گرفته و در نهایت با استفاده از کلاس Paint رنگ و استایل اشکال مورد نظر بر روی سطح canvas را تعیین می نمایید.

1-26-5-کلاس Canvas

آبجکت Canvas آن فایل تصویری پیکسلی یا bitmap ای که اشکال بر روی آن ترسیم می شوند را در برمی گیرد. علاوه بر آن توابعی را ویژه ی تنظیم رنگ (drawARGB())، ترسیم عکس پیکسلی (drawBitmap())، درج متن (drawText())، ترسیم مستطیل با گوشه های گرد (drawRoundRect()) و غیره ... در اختیار برنامه نویس قرار می دهد.

2-26-5-کلاس Paint

برای تنظیم رنگ، فونت و غیره ... بر روی آبجکت Canvas به یک آبجکت از جنس کلاس Paint نیاز دارید.

کلاس Paint در واقع این قابلیت را در اختیار شما قرار می دهد تا رنگ، فونت و برخی از افکت ها را برای انجام عملیات ترسیم مشخص نمایید.

متد setStyle() به شما اجازه می دهد تا به سیستم اعلان نمایید آیا تنها outline و خط پیرامون (Paint.Style.STROKE) باید ترسیم شود یا تنها بخش رنگ شده (Paint.Style.FILL) و یا هر دو بخش مزبور.

برای تنظیم کانال آلفا Paint کافی است متد setAlpha() را فراخوانی نمایید.

جهت تنظیم رنگ های بیشتر برای آبجکت Paint () می توانید از Shaders استفاده نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Shader-5-26-3

آبجکت shader این امکان را فراهم می کند تا برای آبجکت Paint محتوایی که باید ترسیم شوند را مشخص نمایید. برای مثال، می توانید با استفاده از BitmapShader که کلاس مشتق از Shader هست، یک عکس پیکسلی (Bitmap) تعریف نمایید و سپس در آن شکل هندسی دلخواه را ترسیم نمایید. بدین وسیله شما می توانید به عنوان نمونه، تصویری با گوشه های گرد بکشید. کافی است یک BitmapShader برای آبجکت Paint تعریف نموده و با فراخوانی تابع drawRoundRect()، شکل مستطیل را با گوشه های گرد ترسیم نمایید.

محیط اندروید (platform) تعدادی کلاس مشتق از Shader جهت ترسیم و پیاده سازی شیب رنگ ارائه می دهد. این کلاس ها عبارتند از: LinearGradient، RadialGradient و SweepGradient.

به منظور استفاده از Shader لازم است آن را از طریق متد setShader() به آبجکت Paint انتساب دهید.

در صورتی که ناحیه ی پر شده بزرگ تر از Shaders هست، آنگاه می توانید از طریق Shader tile model مشخص کنید که باقی ناحیه ی مربوطه چگونه پر شود. به طور مثال، ثابت Shader.TileMode.CLAMP به سیستم دستور می دهد که رنگ مربوط به گوشه ها (edge corner) بایستی برای پر کردن فضای اضافی مورد استفاده قرار گیرد. در حالی که ثابت Shader.TileMode.MIRROR اعلان می کند که تصویر مورد نظر بایستی قرینه سازی شده و Shader.TileMode.REPEAT نیز به اندروید اعلان می کند که عکسی را تکرار کند.

4-26-5-ذخیره ی دائمی و ماندگارسازی داده های view

اغلب view های متعارف اندروید قادرند داده های مربوط به وضعیت خود را نگه دارند. این داده ها سپس توسط سیستم به صورت دائمی ذخیره می شوند. سیستم اندروید جهت ذخیره ی داده های مربوط به وضعیت view متد onSaveInstanceState() را صدا می زند و جهت ذخیره ی دائمی و بازگردانی این داده ها متد onRestoreInstanceState(Parcable) را فراخوانی می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

برای ماندگار سازی داده ها، مرسوم است که کلاس View.BasedSavedState را داخل view به صورت یک کلاس درونی static پیاده سازی نمایید.

متد View.BasedSavedState یک قرارداد شخصی برای نگه داری داده ها می باشد.

اندروید یک view را بر اساس ID آن در فایل layout میزبان پیدا کرده و سپس آبجکت Bundle را (که حاوی داده های مربوط به وضعیت view می باشد) به view ارسال می کند. View با داده های کپسوله شده در این آبجکت وضعیت خود را بازگردانی می نماید.

شما می بایست اطلاعات UI را دقیقا در همان وضعیتی که کاربر آن را ترک کرده، نگه دارید و بعد مجددا آن را بازگردانی نمایید. برای مثال موقعیت نوار پیمایش یا انتخاب کاربر را به حالت قبلی آن بازگردانید.

بخش هفتم :

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

استفاده از Resource Selector در اندروید – انتخاب منابع متناسب با هر دستگاه به وسیله ی گزینشگر منابع (resource 532ualifier)

مبحث حاضر به شرح resource selector و کاربرد آن در اندروید پرداخته و سپس توضیح می دهد چگونه برنامه را طراحی نمایید که بر روی انواع نمایشگر با عرض و تراکم پیکسلی متفاوت قابل اجرا بوده و از زبان های مختلف پشتیبانی کند.

5-27-تنظیمات و پیکربندی های مختلف دستگاه های اندروید

1-27-5-طراحی برنامه برای انواع دستگاه های اندروید (با نمایشگر و اندازه های مختلف)

دستگاه های اندرویدی از نظر اندازه، تراکم پیکسلی نمایشگر و تنظیمات زبان مورد استفاده متفاوت هستند. برای سازگاری برنامه با دستگاه های مختلف، اندروید منابع مناسب با دستگاه میزبان را انتخاب نموده و اندازه بندی کامپوننت های رابط کاربری (UI) را خود بر حسب واحد dip (فرمت اندازه بندی نسبی) و به صورت اتوماتیک انجام می دهد.

2-27-5-Resource qualifier (تعریف کننده و گزینشگر منابع)

توسعه دهنده با بهره گیری از resource qualifier قادر خواهد برنامه را طوری طراحی کند که با دستگاه های مختلف سازگار باشد و در واقع برای هر دستگاه منابع مناسب (وضعیت نمایش، کیفیت تصویر و زبان مرتبط) را انتخاب نماید. برای ارائه ی فایل های منبع متناسب با تنظیمات هر دستگاه که توسط گزینشگر qualifier معرفی شده و قابل دستیابی می باشد، لازم است یک subfolder در پوشه ی res با ترکیب این qualifier و اسم فایل ایجاد نمایید. به عنوان مثال اسم فایل layout را layout-qualifier انتخاب کنید.

نکته: سیستم اندروید تنظیمات و پیکربندی دستگاه جاری را ارزیابی نموده، سپس بر اساس نتیجه ی ارزیابی فایل مناسب را به صورت خودکار انتخاب می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Resource qualifier–5–27–3 های مهم

استفاده از وضعیت نمایش/orientation به عنوان یک resource qualifier (گزینش منابع بر اساس وضعیت نمایش)

فرض کنید که می خواهید برای activity خود در نمای افقی (landscape mode) از یک فایل layout به نام activity_main.xml استفاده نمایید. در چنین شرایطی، ابتدا پوشه res/layout-land را ایجاد نموده، سپس فایل جدید را با همین نام (activity_main.xml) داخل این پوشه جایگذاری نمایید.

Version qualifier–5–27–4 (گزینش منابع بر اساس ورژن اندروید)

برای انتخاب منابع مناسب با دستگاه جاری، توسعه دهنده می تواند از ورژن کتابخانه های اندروید (API level) بهره بگیرد که مبتنی بر qualifier یا تعریف کننده ی v[minimum API level]- می باشد. از این طریق شما می توانید بر اساس ویرایش اندروید theme و style متفاوت ارائه نمایید.

به عنوان نمونه برنامه نویس می تواند با تکیه بر این qualifier که منابع را بر اساس ویرایش اندروید گزینش می کند، style مربوطه را برای برنامه ارائه نماید و تجربه ی کاربری بهینه را به ارمغان بیاورد.

Width&Height–5–27–5 به عنوان گزینشگر منبع

یکی دیگر از آیتم هایی که توسعه دهندگان برای گزینش منابع مناسب با هر دستگاه مورد استفاده قرار می دهند، smallest available width و available width می باشد. smallest available width عبارت است از حداقل طول قابل استفاده ی صفحه برای نمایش کامپوننت های UI و available width عبارت است از پهنای واقعی بر اساس وضعیت نمایش دستگاه (نمای افقی/عمودی).

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

می توان با استفاده از گزینشگر width، برای مثال، بر اساس پهنای موجود نمایشگر دستگاه، فایل layout مناسب را انتخاب نمود. این نوع انتخاب منبع مبتنی بر qualifier (گزینشگر یا تعریف کننده) sw[Number]dp- یا w[Number]dp- می باشد. [Number] مخفف تعداد پیکسل ها مستقل از دستگاه (dip) می باشد. به عنوان مثال، نمایشگر یک تبلت 7 اینچی معمولا 600dp می باشد و شما می توانید از طریق گزینشگر res/layout-sw600dp/ فایل layout سازگار با نمایشگر دستگاه میزبان را انتخاب نمایید.

سایر qualifier ها جهت گزینش منابع

سایر resource qualifier ها برای گزینش منابع را در آدرس زیر پیدا خواهید کرد.

http://developer.android.com/guide/topics/resources/providing-resources.html#AlternativeResources

String-28-5 ها و محتوای متنی (ترجمه ی نوشته های برنامه)

می توانید با استفاده از resource selector ها، پوشه ی values را که می تواند حامل مقادیر رشته ای باشند، انتخاب نمایید. برای این منظور می توانید از پوشه ی values-qualifier استفاده کنید.

به عنوان مثال، جهت ارائه ی محتوای متنی به زبان انگلیسی یا آلمانی، می بایست از پوشه ی values-en و values-de استفاده نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Plural–5–28–1 ها

اندروید از فایل های محتوایی به نام Plurals پشتیبانی می کند. Plural ها منابعی در قالب XML هستند که به شما این امکان را می دهد تا محتوای متنی مورد نیاز را بر اساس کمیت (مقدار خصیصه ی quantity) انتخاب نمایید. برای استفاده از فایل مقادیر مختلف همچون "zero"، "one"، "two"، "many"، "few" و "other" را داخل XML به quantity انتساب می دهید، سپس در کد با استفاده از متد getQuantityString() مقدار صحیح را از XML واکشی می کنید. می توانید رشته ها را فرمت دهی کنید. اما در صورت فرمت دهی String ها، برای بازیابی مقدار صحیح، لازم است فایل plural و number را به عنوان پارامتر به متد نام برده ارسال کنید. چنانچه از Object ها برای فرمت دهی استفاده می کنید، در آن صورت کافی است آن ها را به عنوان پارامتر اضافی به متد getQuantityString() ارسال نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <plurals
     name="tutorials">
     <item quantity="zero">no Tutorial </item>
     <item quantity="one">one Tutorial </item>
     <item quantity="other">%d  Tutorials</item>
  </plurals>
</resources>
```

تکه کد زیر یک فایل plural تعریف می کند. این فایل بایستی در پوشه ی "res/values" جایگذاری شده و در این مثال نام آن را "plurals" می باشد.

```
// number is defined somewhere before this
// number =....
// get the Resources
Resources res = getResources();
// get the
String quantityString = res.getQuantityString(R.plurals.tutorials,
   number, number);
// do something with it...
```

لازم به ذکر است که انتخاب بر اساس ضرورت نحوی و گرامری صورت گرفته است. رشته ی zero در انگلیسی کاملا نادیده گرفته می شود حتی اگر quantity برابر 0 باشد چرا که 0 به استثنای عدد 1

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

هیچ تفاوتی از نظر گرامری با 2 یا سایر اعداد ندارد ("no tutorial"، "one tutorial"، "two tutorials" و غیره ...)

2-28-5-استفاده از google translate

می توانید String (محتوای متنی) خود را به صورت دستی ترجمه کنید یا از ترجمه ی ماشینی استفاده نمایید.

اگرچه ترجمه ی ماشینی چندان مناسب نیست اما برخی مواقع می تواند تنها گزینه پیشرو باشد به خصوص اگر بودجه ی اپلیکیشن مورد نظر محدود باشد. نرم افزار Google Translate به صورت رایگان ترجمه را برای شما انجام می دهد که طبیعتا به خاطر ماشینی بودن آن از کیفیت پایین برخوردار است، اما در صورت تمایل می توانید در ازای پرداخت مبلغی، ترجمه ی ماشینی را توسط کارشناس ویرایش شده تحویل بگیرید.

نکته: نرم افزار نام برده در ابتدا پیشنهاد ترجمه ی با کیفیت و پولی را به شما می دهد. پس از رد کردن این پیشنهاد می توانید از سرویس ترجمه ی ماشینی بهره بگیرید.

3-28-5-مدیریت مبحث کیفیت تصویر و تراکم پیکسلی نمایشگر های مختلف

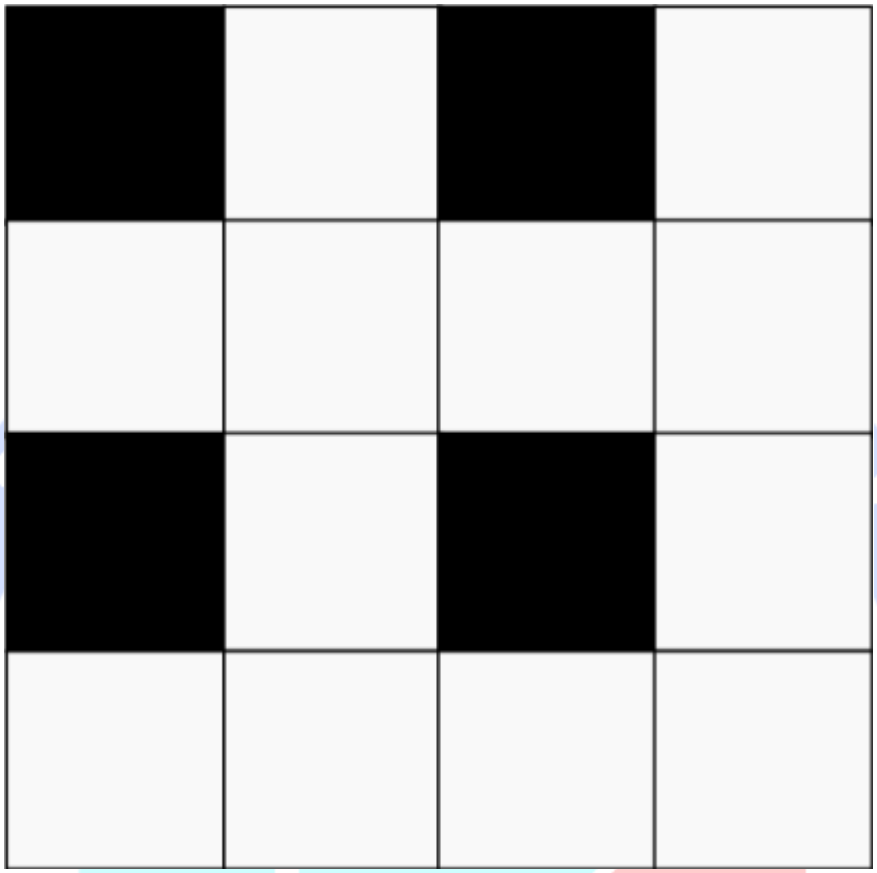
نمایشگر دستگاه های اندرویدی از نظر کیفیت تصویر (resolution) و تراکم پیکسلی متفاوت هستند.

یک کامپوننت UI با اندازه ی یکسان می تواند از تراکم پیکسلی متفاوتی برخوردار بوده و به عبارت دیگر تعداد پیکسل های تشکیل دهنده ی آن بیشتر باشد که نتیجتا کیفیت بهتر را به دنبال دارد.

به عنوان مثال، اگر تعداد پیکسل های یک کامپوننت رابط کاربری را خود به طور مستقیم تعیین نمایید، ظاهر آن در دستگاهی با تراکم پیکسلی پایین به صورت زیر خواهد بود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

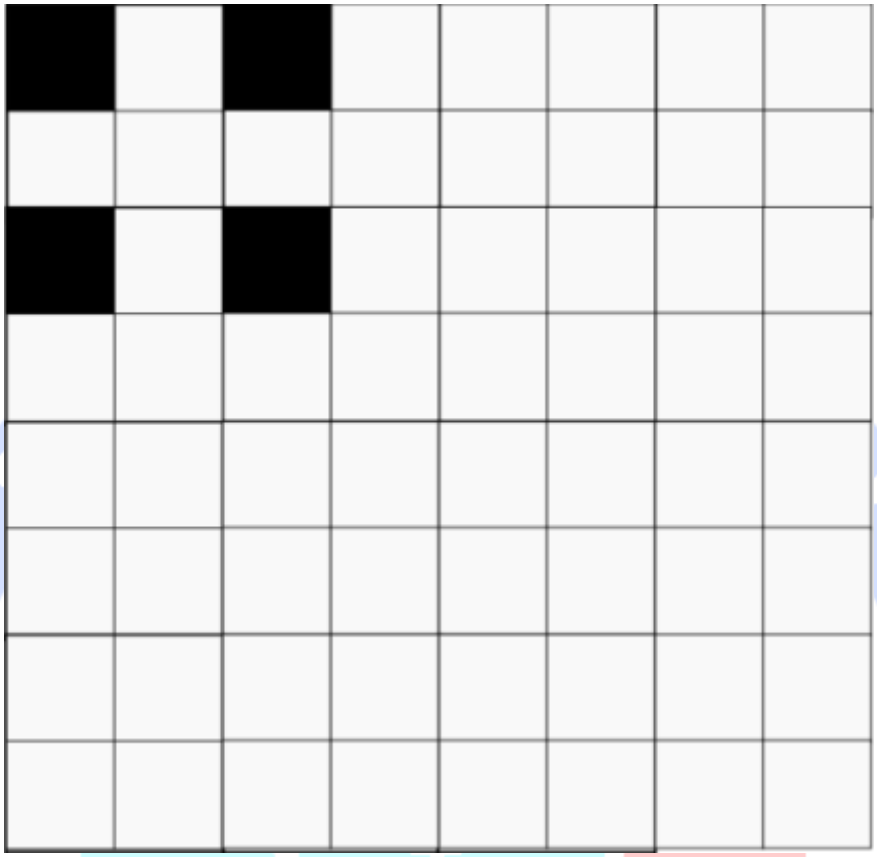
88146323 - 88446780 - 88146330



حال چنانچه تصویر فوق را با همین تعداد پیکسل در دستگاهی با تراکم پیکسلی بالاتر مشاهده کنید، خواهید دید که ظاهر و رابط کاربری اپلیکیشن ناخوانا و غیر قابل استفاده خواهد بود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



به همین علت، شما بایستی منابع و محتوای گرافیکی لازم و مناسب (همچون آیکون اجرای برنامه/launcher و action bar) را در وضوح مختلف ارائه نمایید. این کار را می توانید با استفاده از pixel density به عنوان گزینش گر منابع مورد نیاز به انجام برسانید.

4-28-5-استفاده از density به عنوان گزینش گر منابع (انتخاب محتوا بر اساس چگالی پیکسلی)

می توانید pixel density را مبنای گزینش منابع مورد نیاز اپلیکیشن خود قرار دهید. اندروید برای این منظور گزینه های زیر را در اختیار توسعه دهنده قرار می دهد. حداقل (baseline) است160dpi. چنانچه دستگاه اندروید مورد نظر 320dpi داشته باشد، در آن صورت Drawable از drawable-hdpi انتخاب می شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Table 1. Density resource selector

گزینشگر بر اساس پیکسل	معادل
ldpi	160 dpi x 0.75
mdpi	160 dpi
hdip	1.5 x 160 dpi = 240 dpi
xhdpi	2 x 160 dpi = 320 dpi
xxhdpi	3 x 160 dpi = 480 dpi
xxxhdpi	

5-28-5-ارائه ی آیکون در اندازه های مختلف

شما بایستی launcher یا آیکون اجرا، آیکون مربوط به action bar و آیکونی که در نوار اعلانات (notification bar) به نمایش در می آید را در پنج اندازه ی مهم زیر ارائه نمایید.

Table 2. Android icons size

Icons	mdpi	hdpi	xhdpi	xxhdpi	xxxhdpi
Launcher icon	48 px	72 px	96 px	144 px	192 px
Action bar icon	32 px	48 px	64 px	96 px	128 px
Notification icon	24 px	36 px	48 px	72 px	96 px

لازم به ذکر است که آیکون اجرای اپلیکیشن (launcher) را بایستی برای ارائه در Google Play در اندازه ی 512x512 px نیز در نظر بگیرید.

5-29-تعیین اندازه ی کامپوننت های UI در فایل های layout

5-29-1-اندازه بندی در ابعاد ثابت (fixed) یا نسبی (relative)

اندروید به شما این امکان را می دهد تا اندازه ی کامپوننت های رابط کاربری را در ابعاد ثابت یا نسبی داخل فایل های layout مشخص نمایید. در صورت تعیین اندازه ی المان های رابط کاربری در اندازه ی ثابت، برای مثال پیکسل، ظاهر اپلیکیشن شما ممکن است در یک دستگاه مناسب باشد اما در دستگاهی که دارای تراکم پیکسلی بالاتر است، دکمه به نسبت نمایشگر میزبان بسیار کوچک و بی تناسب جلوه کند.

در ساخت اپلیکیشن های اندرویدی توصیه می شود همیشه ابعاد کامپوننت های UI خود را به صورت نسبی (relative) مشخص نمایید.

5-29-2-استفاده از واحد dp برای اندازه بندی و تعیین ابعاد المان های UI به صورت نسبی

واحد اندازه بندی که بایستی در تعیین ابعاد المان های رابط کاربری مورد استفاده قرار دهید، dp می باشد.

توجه: dp حالت اختصار dip یا device independent pixel به معنی تعیین تعداد پیکسل مستقل از چگالی دستگاه میزبان می باشد.

منظور از dp یک پیکسل در دستگاه اندروید با چگالی 160dpi می باشد. تراکم پیکسلی مزبور در واقع به چگالی اولین دستگاه تولیده شده ی اندروید (G1) اشاره دارد که از آن تحت عنوان mdpi (medium dots per inch) نیز یاد می شود.

اگر اندازه ی دستگاه را بر حسب dp مشخص نمایید، در آن صورت سیستم اندروید المان های رابط کاربری شما را به صورت خودکار متناسب با دستگاه میزبان بزرگ/کوچک و مقیاس دهی می کند.

در یک دستگاه mdpi (دستگاهی با چگالی یا تراکم پیکسلی متوسط)، 1dp معادل یک پیکسل است. این در حالی است که 1dp در دستگاهی با ldpi (چگالی پایین) کوچکتر (حدودا 120dpi)، در دستگاهی با چگالی بالا (high density) بزرگتر (حدودا 240dpi) می باشد. از این رو یک dp حدودا همان میزان فضا را در تمامی دستگاه ها اشغال می کند.

می توانید اندازه و ابعاد المان های رابط کاربری خود را در فایل های layout بر حسب dp تعیین نمایید.

3-29-5-اندازه بندی نوشته ها بر حسب واحد sp

چنانچه قرار است واحد اندازه بندی با تنظیمات انتخابی کاربر در خصوص نوشته ها مقیاس دهی و تنظیم شود، لازم است از واحد اندازه گیری sp استفاده نمایید. این واحد اندازه گیری شبیه به dp عمل می کند، با این تفاوت که بر اساس تنظیمات دلخواه کاربر نیز قابل مقیاس بندی شده و با آن هماهنگ می باشد (به عبارت دیگر با تنظیمات اندازه ی فونتی که کاربر تعیین می کند منطبق می باشد. از این رو خروجی با چگالی نمایشگر میزبان و انتخاب کاربر همگام خواهد بود).

چنانچه کاربر تصمیم بگیرد که اندازه ی فونت را در تنظیمات افزایش دهد، به دنبال آن تمامی view هایی که بر حسب sp تنظیم شده اند، به همان ترتیب مقیاس دهی می شوند.

30-5-تعیین اندازه ی کامپوننت های UI در کد برنامه (source code)

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

API و مجموعه توابع اندروید اغلب از توسعه دهنده درخواست می کند که اندازه ی المان ها را بر حسب پیکسل مشخص کند و واحد dp را به عنوان ورودی از شما نمی پذیرد. در چنین شرایطی لازم است واحد dp را به پیکسل تبدیل نمایید.

برای نیل به این هدف می توانید از متد زیر استفاده نمایید. این متد ابعاد المان را بر حسب dp به عنوان ورودی دریافت می کند سپس آن را داخل بدنه ی خود به پیکسل تبدیل نموده و در خروجی برمی گرداند.

```
public int convertToPixelFromDp(int dpInput) {
    // get the screen's density scale
    final float scale = getResources().getDisplayMetrics().density;
    // convert the dps to pixels, based on density scale
    return (int) (dpInput * scale + 0.5f);
}
```

جهت بدست آوردن چگالی پیکسلی دستگاه جاری می توانید متد زیر را مورد استفاده قرار دهید.

```
getResources().getConfiguration().densityDpi;
```

بخش هشتم :

پیاده سازی انیمیشن در اپلیکیشن های اندرویدی/آموزش Animation API

مبحث حاضر به شرح نحوه ی پیاده سازی انیمیشن در اندروید (استفاده از property Animation API) می پردازد. پروژه های این بخش در محیط کاری Android Studio نوشته شده و مبتنی بر ویرایش 5.1 اندروید می باشد.

5-31-انیمیشن سازی در اندروید

5-31-1-استفاده از انیمیشن در اندروید

ویرایش 3.0 اندروید ابزار Properties Animation API را برای اولین بار رونمایی کرد که به توسعه دهنده این امکان را می دهد تا مقادیر property های یک آبجکت را از یک نقطه ی زمانی تا نقطه ی زمانی دیگر ویرایش نموده و جلوه های دیداری زیبایی را خلق کند.

با بهره گیری از API مذکور قادر خواهید برای property های دلخواه آبجکت یک مقدار آغاز و یک مقدار پایان تعیین کنید. این دو مقدار یک بازه ی زمانی مشخص می کنند که طی آن ظاهر المان تغییر کرده یا از مکانی در صفحه به مکان دیگری حرکت می کند.

این API صرفا محدود به Views نبوده و به راحتی برای سایر آبجکت های جاوا قابل استفاده می باشد.

5-31-2-کلاس های Animator و AnimatorListener

کلاس پدر (superclass) توابع انیمیشن سازی اندروید (Animation API) کلاس Animator می باشد. ObjectAnimator کلاسی است که توسعه دهنده برای ویرایش attribute های یک آبجکت مورد استفاده قرار می دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

همچنین می توانید یک نمونه از کلاس AnimatorListener را به کلاس Animator اضافه نمایید. کلاس نام برده یک گوش فرخوان/listener است و این قابلیت را دارد که در طی مراحل مختلف یک انیمیشن فراخوانی شود.

شما می توانید با استفاده از این listener به اتفاقات معینی گوش داده و قبل یا بعد از رخداد اتفاق مورد نظر، عملیات معینی را به اجرا بگذارید. به عنوان مثال یک view را از ViewGroup حذف نمایید یا ظاهر یک المان UI را در بازه ی زمانی معینی تغییر دهید.

3-31-5-کلاس ViewPropertyAnimator

کلاس ViewPropertyAnimator در ویرایش 3.1 اندروید ارائه شده و دسترسی به انیمیشن های متعارف که بر روی view ها اجرا می شوند را به مراتب آسان تر ساخت.

متد ()animate بر روی view فراخوانی شده و نمونه ای از ViewPropertyAnimator را در خروجی بر می گرداند. این آبجکت به شما امکان اجرای چندین انیمیشن را به طور همزمان می دهد. آبجکت ViewPropertyAnimator از fluent API برای پشتیبانی از کدهای خوانا، نوشتن روان و پشت سرهم دستورات استفاده کرده و علاوه بر آن به شما این اجازه را می دهد تا مدت زمان اجرای انیمیشن را مشخص نمایید (Fluent API عبارت است از پیاده سازی interface شی گرا به صورتی که قابلیت خوانایی کد به طور چشم گیری افزایش می یابد و طوری که برنامه نویس بتواند دستورات را به صورت روان و پشت سرهم یکجا بنویسد).

هدف از بکار بردن ViewPropertyAnimator ارائه ی یک API ساده برای اجرا و دسترسی آسان به انیمیشن های پرکاربرد اندروید می باشد.

کد زیر نحوه ی استفاده از این متد را به نمایش می گذارد.

```
// Using hardware layer
myView.animate().translationX(400).withLayer();
```

به منظور افزایش کارایی، شما می توانید به ViewPropertyAnimator این اجازه را بدهید تا از یک لایه ی سخت افزاری نیز استفاده کند.

```
// Using hardware layer
myView.animate().translationX(400).withLayer();
```

در صورت نیاز می توانید به طور مسقیم اینترفیس Runnable را پیاده سازی کنید که در ابتدا و انتهای انیمشین اجرا می شود.

```
// StartAction
myView.animate().translationX(100).withStartAction(new Runnable(){
    public void run(){
        viewer.setTranslationX(100-myView.getWidth());
        // do something
    }
});
// EndAction
myView.animate().alpha(0).withStartAction(new Runnable(){
    public void run(){
        // rRemove the view from the parent layout
        parent.removeView(myView);
    }
});
```

متد ()setInterpolator به شما اجازه می دهد تا آبجکتی از جنس TimeInterpolator تعریف کرده و تغییر مقدار را در طول زمان مشخصی انجام دهید. حالت استاندارد linear می باشد.

محیط اندروید (platform) تعدادی حالت پیش فرض و استاندارد همچون AccelerateDecelerateInterpolator تعیین می کند که در آن انیمیشن با سرعت پایین آغاز شده، در اواسط تند می شود و بار دیگر در پایان با سرعت پایین خاتمه می یابد.

از طریق متد setEvaluator می توانید آبجکتی از نوع TypeEvaluator تنظیم کنید که به شما این امکان را می دهد تا برای property type دلخواه انیمیشن تعریف کنید. این کار با ارائه ی evaluator

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

های اختصاصی برای نوع هایی انجام می شود که اندروید آن ها را نمی شناسد و به صورت پیش فرض توسط سیستم انیمیشن سازی اندروید استفاده نمی شود.

Layout animations–5–31–4

کلاس LayoutTransition این قابلیت را به توسعه دهنده می دهد تا بر روی layout container (ظرف و میزبان layout) انیمیشن تنظیم کند. بدین وسیله هر تغییر در view hierarchy این container به صورت انیمیشن اجرا خواهد شد.

```
package com.example.android.layoutanimation;
import android.animation.LayoutTransition;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
public class MainActivity extends Activity {
    private ViewGroup viewGroup;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        LayoutTransition l = new LayoutTransition();
        l.enableTransitionType(LayoutTransition.CHANGING);
        viewGroup = (ViewGroup) findViewById(R.id.container);
        viewGroup.setLayoutTransition(l);
    }
    public void onClick(View view) {
        viewGroup.addView(new Button(this));
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }
}
```

5–31–5–اعمال انیمیشن بر روی انتقال بین activity ها (پیاده سازی انیمیشن بر روی transition بین دو activity)

انیمیشن را می توان علاوه بر روی Views، بر روی انتقال بین activity ها نیز اعمال نمود.

کلاس ActivityOptions به شما اجازه می دهد تا انیمیشن های پیش فرض اندروید و در صورت نیاز انیمیشن های اختصاصی خود را پیاده سازی نمایید.

```
public void onClick(View view) {
    Intent intent = new Intent(this, SecondActivity.class);
    ActivityOptions options = ActivityOptions.makeScaleUpAnimation(view, 0,
            0, view.getWidth(), view.getHeight());
    startActivity(intent, options.toBundle());
}
```

آموزش View Animation

این بخش به شرح نحوه ی استفاده از Properties animation API می پردازد.

یک پروژه ی و activity جدید اندروید به ترتیب به نام های com.vogella.android.animation.views و AnimationExampleActivity ایجاد نمایید. فایل layout می بایست main.xml نام گذاری شده باشد. محتوای آن را به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:id="@+id/layout"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
  <LinearLayout
    android:id="@+id/test"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" >
    <Button
      android:id="@+id/Button01"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:onClick="startAnimation"
      android:text="Rotate" />
    <Button
      android:id="@+id/Button04"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:onClick="startAnimation"
      android:text="Group" >
    </Button>
    <Button
      android:id="@+id/Button03"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:onClick="startAnimation"
      android:text="Fade" />
```

```
    <Button
      android:id="@+id/Button02"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:onClick="startAnimation"
      android:text="Animate" />
  </LinearLayout>
  <ImageView
    android:id="@+id/imageView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:src="@drawable/icon" />
  <TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/imageView1"
    android:layout_alignRight="@+id/imageView1"
    android:layout_marginBottom="30dp"
    android:text="Large Text"      android:textAppearance="?android:attr/textAppearanceLarge" />
</RelativeLayout>
```

فایل XML زیر را برای تعریف منو ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
  <item
    android:id="@+id/item1"
    android:showAsAction="ifRoom"
    android:title="Game">
  </item>
</menu>
```

بدنه ی activity خود را به صورت زیر ویرایش نمایید.

```
package com.vogella.android.animation.views;
import android.animation.AnimatorSet;
import android.animation.ObjectAnimator;
import android.app.Activity;
import android.content.Intent;
import android.graphics.Paint;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.ImageView;
```

```
import android.widget.TextView;
public class AnimationExampleActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
    public void startAnimation(View view) {
        float dest = 0;
        ImageView aniView = (ImageView) findViewById(R.id.imageView1);
        switch (view.getId()) {
        case R.id.Button01:
            dest = 360;
            if (aniView.getRotation() == 360) {
                System.out.println(aniView.getAlpha());
                dest = 0;
            }
            ObjectAnimator animation1 = ObjectAnimator.ofFloat(aniView,
                    "rotation", dest);
            animation1.setDuration(2000);
            animation1.start();
            // Show how to load an animation from XML
            // Animation animation1 = AnimationUtils.loadAnimation(this,
            // R.anim.myanimation);
            // animation1.setAnimationListener(this);
            // animatedView1.startAnimation(animation1);
            break;
        case R.id.Button02:
            // shows how to define a animation via code
            // also use an Interpolator (BounceInterpolator)
            Paint paint = new Paint();
            TextView aniTextView = (TextView) findViewById(R.id.textView1);
            float measureTextCenter = paint.measureText(aniTextView.getText()
                    .toString());
            dest = 0 - measureTextCenter;
            if (aniTextView.getX() < 0) {
                dest = 0;
            }
            ObjectAnimator animation2 = ObjectAnimator.ofFloat(aniTextView,
                    "x", dest);
            animation2.setDuration(2000);
            animation2.start();
            break;
        case R.id.Button03:
            // demonstrate fading and adding an AnimationListener
            dest = 1;
            if (aniView.getAlpha() > 0) {
                dest = 0;
            }
            ObjectAnimator animation3 = ObjectAnimator.ofFloat(aniView,
```

```
                    "alpha", dest);
            animation3.setDuration(2000);
            animation3.start();
            break;
        case R.id.Button04:
            ObjectAnimator fadeOut = ObjectAnimator.ofFloat(aniView, "alpha",
                    0f);
            fadeOut.setDuration(2000);
            ObjectAnimator mover = ObjectAnimator.ofFloat(aniView,
                    "translationX", -500f, 0f);
            mover.setDuration(2000);
            ObjectAnimator fadeIn = ObjectAnimator.ofFloat(aniView, "alpha",
                    0f, 1f);
            fadeIn.setDuration(2000);
            AnimatorSet animatorSet = new AnimatorSet();
            animatorSet.play(mover).with(fadeIn).after(fadeOut);
            animatorSet.start();
            break;
        default:
            break;
        }
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.mymenu, menu);
        return super.onCreateOptionsMenu(menu);
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        Intent intent = new Intent(this, HitActivity.class);
        startActivity(intent);
        return true;
    }
}
```

یک activity دیگر به نام HitActivity ایجاد نمایید.

```
package com.vogella.android.animation.views;
import java.util.Random;
import android.animation.Animator;
import android.animation.AnimatorListenerAdapter;
import android.animation.AnimatorSet;
import android.animation.ObjectAnimator;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
public class HitActivity extends Activity {
    private ObjectAnimator animation1;
    private ObjectAnimator animation2;
    private Button button;
```

```
    private Random randon;
    private int width;
    private int height;
    private AnimatorSet set;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.target);
        width = getWindowManager().getDefaultDisplay().getWidth();
        height = getWindowManager().getDefaultDisplay().getHeight();
        randon = new Random();
        set = createAnimation();
        set.start();
        set.addListener(new AnimatorListenerAdapter() {
            @Override
            public void onAnimationEnd(Animator animation) {
                int nextX = randon.nextInt(width);
                int nextY = randon.nextInt(height);
                animation1 = ObjectAnimator.ofFloat(button, "x", button.getX(),
                        nextX);
                animation1.setDuration(1400);
                animation2 = ObjectAnimator.ofFloat(button, "y", button.getY(),
                        nextY);
                animation2.setDuration(1400);
                set.playTogether(animation1, animation2);
                set.start();
            }
        });
    }
    public void onClick(View view) {
        String string = button.getText().toString();
        int hitTarget = Integer.valueOf(string) + 1;
        button.setText(String.valueOf(hitTarget));
    }
    private AnimatorSet createAnimation() {
        int nextX = randon.nextInt(width);
        int nextY = randon.nextInt(height);
        button = (Button) findViewById(R.id.button1);
        animation1 = ObjectAnimator.ofFloat(button, "x", nextX);
        animation1.setDuration(1400);
        animation2 = ObjectAnimator.ofFloat(button, "y", nextY);
        animation2.setDuration(1400);
        AnimatorSet set = new AnimatorSet();
        set.playTogether(animation1, animation2);
        return set;
    }
}
```

زمانی که این مثال را اجرا کرده و بر روی دکمه های مختلف آن کلیک می کنید، انیمیشن آغاز می شود. از طریق ActionBar می توانید به activity دوم پیمایش کنید.

5-32-اعمال انیمیشن بر روی انتقال بین activity ها در اندروید با shared view ها

ویرایش 5.0 اندروید این امکان را فراهم آورد تا بر روی فعل انتقال بین دو activity انیمیشن اعمال نموده و نیز view های تعریف کرد که بین دو activity مشترک هستند. پس از تعریف یک بخش مشترک، view قدیمی طی یک انیمیشن از activity اول در جایگاه و اندازه ی view جدید در activity دوم قرار می گیرد.

به منظور تست این قابلیت، یک پروژه ی جدید ایجاد نموده و پکیج com.vogella.android.activityanimationwithsharedviews را در بالای فایل پروژه (به عنوان top level package) درج نمایید.

دو activity با دو فایل layout متفاوت تعریف نمایید که هر دو دربردارنده ی یک المان ImageView بوده و خاصیت android:transitionName (property) را داشته باشند.

activity_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingBottom="@dimen/activity_vertical_margin"
  android:paddingLeft="@dimen/activity_horizontal_margin"
  android:paddingRight="@dimen/activity_horizontal_margin"
  android:paddingTop="@dimen/activity_vertical_margin"
  tools:context=".MainActivity">
  <ImageView
    android:id="@+id/sharedimage"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:scaleType="centerCrop"
```

```
    android:src="@drawable/ic_sharedimage"
    />
</LinearLayout>
```

activity_second.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingBottom="@dimen/activity_vertical_margin"
  android:paddingLeft="@dimen/activity_horizontal_margin"
  android:paddingRight="@dimen/activity_horizontal_margin"
  android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.vogella.android.activityanimationwithsharedviews.SecondActivity">
  <ImageView
    android:id="@+id/sharedimage"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:src="@drawable/ic_sharedimage"
    android:layout_alignParentBottom="true"
    android:layout_alignParentEnd="true" />
  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/hello_world"
    android:id="@+id/textView" />
  <Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="New Button"
    android:id="@+id/button"
    android:transitionName="sharedImage"
    android:layout_below="@+id/textView"
    android:layout_alignParentStart="true"
    android:layout_marginTop="54dp" />
</RelativeLayout>
```

کد activity خود را به صورت زیر ویرایش نمایید.

```
package com.vogella.android.activityanimationwithsharedviews;
import android.app.Activity;
import android.app.ActivityOptions;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
public class MainActivity extends Activity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final ImageView sharedImage = (ImageView) findViewById(R.id.sharedimage);
        sharedImage.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                //This is where the magic happens.
                    // makeSceneTransitionAnimation takes a context, view,
                    // a name for the target view.
                ActivityOptions options =
                        ActivityOptions.
                        makeSceneTransitionAnimation(MainActivity.this, sharedImage, "sharedImage");
                Intent intent = new Intent(MainActivity.this, SecondActivity.class);
                startActivity(intent, options.toBundle());
            }
        });
    }
}
package com.vogella.android.activityanimationwithsharedviews;
import android.app.Activity;
import android.os.Bundle;
public class SecondActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
    }
}
```

پس از اجرای اپلیکیشن و کلیک بر روی المان image view، می بینید که المان مزبور با انیمیشن به view ای که خاصیت android:transitionName را دارد (کنترل دکمه) منتقل می شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فصل ششم

سرویس Broadcast Receiver و Notification Manager در اندروید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش اول :

سرویس های اندروید/Service در اندروید

مبحث جاری نحوه ی استفاده از سرویس های پیش فرض اندروید، پیاده سازی و استفاده از سرویس های اختصاصی را شرح می دهد. این آموزش مبتنی بر ویرایش 6.0 اندروید می باشد.

6-1-سرویس های اندروید

6-1-1-سرویس چیست

Service یک کامپوننت نرم افزاری است که در پس زمینه اجرا شده و مستقیما با کاربر تعامل ندارد. از آنجایی که سرویس فاقد است، طبیعتا به چرخه ی حیات یک activity نیز متصل نمی باشد.

سرویس ها اغلب برای انجام عملیات تکراری و طولانی مورد استفاده قرار می گیرند. از جمله ی این عملیات می توان به دانلود از اینترنت، بررسی و جستجو برای داده های جدید، پردازش اطلاعات، بروز آوری content provider ها و موارد مشابه اشاره کرد.

سرویس ها دارای اولویت سطح بالاتری نسبت به activity های غیرفعال/غیر قابل مشاهده در UI هستند و از این رو احتمال اینکه اندروید آن ها را به صورت خودکار خاتمه دهد بسیار پایین است.

اندروید به شما این امکان را می دهد تا سرویس ها را طوری تنظیم کنید که اگر به هر دلیلی مجبور به حذف این سرویس ها از حافظه شد، به مجرد قرار گرفتن منابع کافی در اختیار سیستم، قادر باشد آن ها را مجددا راه اندازی کند.

می توان به سرویس ها اولویت یکسان و برابر با ACTIVITY های حاضر در پیش زمینه (FORGROUND) اختصاص داد. در این سناریو لازم است یک notification قابل مشاهده و فعال در UI برای سرویس های مربوطه لحاظ نمایید. این روش بیشتر برای سرویس هایی بکار می رود که یک فایل ویدیویی یا موسیقی را پخش می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

6-1-2-سرویس ها و پردازش پس زمینه ای (background processing)

به صورت پیش فرض، سرویس در همان فرایندی اجرا می شود که thread اصلی اپلیکیشن در آن حال اجرا است. به همین جهت توسعه دهنده بایستی از پردازش ناهمزمان در سرویس استفاده نموده و task هایی که هزینه بر و سنگین هستند را در پس زمینه راه اندازی کند. یکی از الگوهایی که مکررا برای پیاده سازی سرویس بکار می رود، اجرای یک Thread جدید در سرویس جهت انجام پردازش در پس زمینه و خاتمه دادن سرویس به هنگام اتمام پردازش می باشد.

سرویس هایی که در بستر فرایند خود اپلیکیشن اجرا می شوند معمولا تحت عنوان service های محلی یا local شناخته می شوند.

6-1-3-سرویس های خود محیط اندروید (platform) و سرویس های اختصاصی

محیط اندروید سرویس های آماده و از پیش تعریف شده ای را درنظر گرفته و راه اندازی می کند که تمامی اپلیکیشن های اندرویدی، در صورت برخوردار بودن از مجوزهای لازم قادر به استفاده از آن ها استفاده می باشند. سرویس های سیستم را کلاسی به نام Manager در اختیار اپلیکیشن ها قرار می دهد.کافی است برای دسترسی به آن متد ()getSystemService را فراخوانی نمایید.

کلاس Context تعدادی ثابت فراهم می کند که شما با استفاده از آن ها می توانید سرویس های نام برده را فراخوانی کنید.

اپلیکیشن اندروید می تواند علاوه بر سرویس های پیش فرض سیستم اندروید، سرویس های اختصاصی تعریف نموده و از آن ها در کنار سرویس های سیستم استفاده نماید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

توسعه دهنده قادر است با پیاده سازی سرویس های اختصاصی خود اپلیکیشن های پاسخگو و تعاملی (responsive) طراحی نماید. شما می توانید داده های اپلیکیشن را به وسیله های سرویس واکشی نموده و زمانی که اپلیکیشن راه اندازی شد، داده های جدید در اختیار کاربر قرار دهید.

6-1-4-راه اندازی و تعریف سرویس های اختصاصی

سرویس های اختصاصی اغلب توسط کامپوننت های دیگر راه اندازی می شوند، به عبارت دیگر سایر اجزا نرم افزاری اپلیکیشن های اندرویدی نظیر activity ها، broadcast receiver ها و سرویس های دیگر هستند که سرویس های اختصاصی را راه اندازی می کنند.

6-1-5-سرویس های پیش زمینه (foreground)

سرویس پیش زمینه سرویسی است که از نظر اولویت و اهمیت با یک activity فعال و قابل مشاهده در UI یکسان است و به همین جهت حتی اگر سیستم اندروید با کمبود حافظه مواجه باشد باز هم اجازه ی حذف از آن ها حافظه را ندارد. سرویس foreground می بایست در نوار نشان دهنده ی وضعیت کلی سیستم (status bar) یک اطلاعیه یا notification در زیر بخش عنوان "Ongoing" مختص به خود داشته باشد. این بدین معنی است که تا زمان حذف سرویس از foreground یا حافظه، notification قابل dismiss و حذف از status bar نخواهد بود.

```
Notification notification = new Notification(R.drawable.icon, getText(R.string.ticker_text),
        System.currentTimeMillis());
Intent notificationIntent = new Intent(this, ExampleActivity.class);
PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, notificationIntent, 0);
notification.setLatestEventInfo(this, getText(R.string.notification_title),
        getText(R.string.notification_message), pendingIntent);
startForeground(ONGOING_NOTIFICATION_ID, notification);
```

6-2-تعریف سرویس های اختصاصی

1-2-6-پیاده سازی و اعلان

برای پیاده سازی یک سرویس اختصاصی، ابتدا لازم است آن را با استفاده از تگ service در لایه ی XML (فایل تنظیمات AndroidManifest.xml) تعریف نموده و سپس در کدهای جاوا (کلاسی که سرویس را پیاده سازی می کند) از کلاس Service یا یکی از کلاس های مشتق آن ارث بری نمایید.

کد زیر نحوه ی تعریف سرویس در لایه ی xml و پیاده سازی آن در کلاس های جاوا را نمایش می دهد.

```
<service
      android:name="MyService"
      android:icon="@drawable/icon"
      android:label="@string/service_name"
      >
</service>
public class MyService extends Service {
 @Override
 public int onStartCommand(Intent intent, int flags, int startId) {
       //TODO do something useful
       return Service.START_NOT_STICKY;
 }
 @Override
 public IBinder onBind(Intent intent) {
     //TODO for communication return IBinder implementation
   return null;
 }
}
```

راه اندازی یک سرویس

دیگر کامپوننت های نرم افزاری اندروید (activity ،receiver ،service) می توانند با فراخوانی تابع startService(intent)، یک سرویس اندروید را راه اندازی کنند.

```
// use this to start and trigger a service
Intent i= new Intent(context, MyService.class);
// potentially add data to the intent
i.putExtra("KEY1", "Value to be used by the service");
context.startService(i);
```

روش دیگر راه اندازی سرویس، فراخوانی متد ()bindSerivce می باشد. از این طریق شما می توانید مستقیما با service مورد نظر تعامل داشته باشید.

در آینده بیشتر درباره ی روش دوم توضیح خواهیم داد.

2-2-6-فرایند راه اندازی و اجرای سرویس

چنانچه پس از فراخوانی متد startService(intent)، سرویس همچنان راه اندازی نشده باشد، آنگاه آبجکت service ایجاد شده و متد ()onCreate از کلاس سرویس فراخوانی می شود.

پس از اجرای سرویس، متد onStartCommand(intent) در سرویس فراخوانی می شود. سپس آبجکت Intent را به عنوان پارامتر از تابع فراخوانده شده ی startService(intent) دریافت می کند.

در واقع سیستم اندروید با فراخوانی متد ()startService، تابع onStartCommand(intent) را زمانی صدا می زند که کامپوننت نرم افزاری دیگری همچون activity نیاز به اجرای سرویس مورد نظر داشته باشد. لازم به ذکر است که اگر برنامه نویس این متد را پیاده سازی کند، آنگاه مسئولیت متوقف نمودن سرویس را بایستی خود با صدا زدن توابع ()stopService، ()stopSelf بر عهده بگیرد.

اگر startService(intent) صدا خورده شود، درحالی که سرویس همچنان در حال اجر است، در آن صورت متد ()onStartCommand نیز فراخوانی می شود. از این جهت سرویس بایستی این قابلیت و آمادگی را داشته باشد که متد ()onStartCommand بارها صدا خورده شود.

نکته: در صورتی که این متد دوبار در کد فراخوانی شود، چه اتفاق رخ خواهد داد؟ آیا لازم است نگران همگام سازی فرخوانی های ()onStartCommand باشید؟ در پاسخ باید گفت خیر. این متد را سیستم اندروید در thread اصلی (UI) صدا می زند، از این جهت امکان فراخوانی همزمان آن در دو thread متفاوت وجود ندارد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

سرویس تنها یکبار راه اندازی می شود، فارغ تعداد دفعاتی که متد startService() فراخوانی می شود.

3-2-6-رفتار بازآغازی و restart سرویس

سرویس داخل بدنه ی متد onStartCommand() خود یک مقدار int را برمی گرداند (return). این دستور رفتار سرویس را در صورتی که محیط اندروید (platform) مجبور به خاتمه دادن سرویس و حذف آن از حافظه شود را مشخص می نماید. برای این منظور می توانید از ثوابت زیر استفاده نمایید. این ثوابت به همراه شرح کاربرد در جدول زیر ذکر شده اند.

Table 1. Restart options

شرح	ثابت
اگر سیستم سرویس را از حافظه حذف کرد، سرویس مجددا راه اندازی می شود. آبجکت intent ارسالی به متد onStartCommand حامل مقدار null خواهد بود. این گزینه اغلب برای سرویس هایی بکار می روند که خود اطلاعات مربوط به وضعیتشان را مدیریت کرده و وابستگی به داده های موجود در Intent ندارند.	Service.START_STICKY
سرویس راه اندازی نشده است. برای سرویس هایی مورد استفاده	Service.START_NOT_STICKY

Table 1. Restart options	
شرح	ثابت
قرار می گیرد در فواصل زمانی معین راه اندازی می شوند. سرویس تنها هنگامی راه اندازی می شود که runtime تعدادی فراخوانی ()startService را از زمان حذف سرویس از حافظه، به صورت معلق و آماده ی اجرا داشته باشد.	
عملکردی مشابه Service.START_STICKY دارد با این تفاوت که آبجکت intent اصلی مجددا به متد onStartCommand ارسال می شود.	Service.START_REDELIVER_INTENT

نکته: می توانید با فراخوانی متد ()Intent.getFlags به همراه هر یک از پارامترهای (1. START_FLAG_REDELIVERY = در صورتی که سرویس با Service.START_REDELIVER_INTENT راه اندازی شده باشد 2. START_FLAG_RETRY = چنانچه سرویس با Service.START_REDELIVER_INTENT صدا خورده شده باشد 3. START_FLAG_RETRY = اگر سرویس با Service.START_STICKY فراخوانده شده باشد) بررسی کنید آیا سرویس مجددا راه اندازی شده است یا خیر.

4-2-6-متوقف کردن یک سرویس

برای متوقف کردن یک سرویس کافی است متد ()stopService را صدا بزنید. فارغ از تعداد دفعاتی که متد ()startService را برای راه اندازی خود سرویس فراخوانی کردید، یکبار صدا زدن تابع ()stopService سرویس مورد نظر را کاملا متوقف می سازد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

یک سرویس می تواند خود را با فراخوانی متد stopSelf() متوقف کند. این تابع معمولا زمانی صدا خورده می شود که سرویس کارش کاملا تمام شده باشد.

استفاده از IntentServices (ارث بری از کلاس IntentServices)

می توانید از کلاس IntentService برای پیاده سازی سرویس استفاده نمایید.

IntentService برای اجرای تسک های خاصی در background بکار می رود. نمونه ی پیاده سازی شده از این کلاس پس از اتمام وظایفش، خود را به صورت اتوماتیک متوقف ساخته و از حافظه حذف می نماید. یکی از موارد استفاده ی این کلاس در بارگیری منابع و محتوای خاص از اینترنت است.

کلاس IntentService متد onHandleIntent() را ارائه می دهد. این متد را سیستم اندروید به صورت ناهمزمان (asynch) فراخوانی می نماید.

6-3-متصل کردن دوطرفه ی سرویس ها (service binding)

6-3-1-وصل شدن از activity ها به سرویس ها

چنانچه activity لازم دارد به طور مستقیم با service تبادل داده و تعامل داشته باشد، ابتدا متد bindService() را برای راه اندازی service صدا می زند. این متد یک آبجکت ServiceConnection به عنوان ورودی می گیرد که به هنگام آغاز سرویس و اتمام اجرای متد onBind() فراخوانی می شود. متد ذکر شده یک آبجکت IBinder به ServiceConnection برمی گرداند.

در واقع سیستم با فراخوانی bindService() متد onBind() را زمانی صدا می زند که کامپوننت دیگری درخواست اتصال به این سرویس را داشته باشد. در صورتی که توسعه دهنده این متد را پیاده سازی کرده باشد، آنگاه می بایست یک interface نیز پیاده سازی کرده و آبجکت IBinder را به عنوان خروجی برگرداند که کلاینت برای ارتباط با سرور از آن استفاده می کند. لازم به توضیح است که پیاده سازی این متد ضروری می باشد، اما چنانچه نمی خواهید اجازه ی اتصال دو طرفه (Bind) را بدهید، می توانید null را بازگردانی کنید.

Activity با استفاده از آبجکت IBinder با سرویس اطلاعات رد و بدل می کنند.

زمانی که فرایند اتصال به پایان می رسد، متد onStartCommand() در سرویس با آبجکت Intent فراخوانی شده که متد bindService() از آن استفاده می کند.

6-3-2-اتصال به سرویس های محلی

اگر سرویس در فرایندی یکسان با activity اجرا شود (سرویس و activity هر دو در یک فرایند اجرا شوند)، در آن صورت می توان سرویس را به activity برگرداند. این امر زمینه ای برای activity فراهم می آورد تا متدهای سرویس را مستقیما فراخوانی کند. این روش را به صورت کاربردی در <exercise_bindlocalservice" /> می بینید که پیاده سازی شده است.

6-3-3-اتصال به سرویس با استفاده از IPC

اگر سرویس در فرایند ویژه ی خودش اجرا شود، آنگاه بایستی با استفاده از IPC با سرویس اطلاعات تبادل نمایید.

6-4-اجرای سرویس ها در فرایندهای مجزا

6-4-1-اجرای یک سرویس در فرایند مختص به خود

می توانید سرویس خود را طوری تنظیم نمایید که در فرایند جداگانه و مختص به خود اجرا شود. برای این منظور مقدار android:process را در داخل تگ service، برابر ":process_description" قرار می دهید.

```
<service
    android:name="WordService"
    android:process=":my_process"
    android:icon="@drawable/icon"
```

```
    android:label="@string/service_name"
    >
</service>
```

علامت دونقطه رو به روی : = android:process به سیستم Android اعلان می کند که Service مورد نظر مختص اپلیکیشن میزبان و تعریف کننده ی آن است. در صورتی که از دو نقطه استفاده نشده باشد، آنگاه Service در بستر یک فرایند سراسری اجرا شده و سایر اپلیکیشن های اندرویدی به آن دسترسی خواهند داشت.

اجرای یک سرویس در فرایند خودش سبب مسدود شدن اپلیکیشن نمی شود، حتی اگر سرویس مورد نظر عملیات طولانی مدت در thread اصلی خود اجرا کند. اما از طرفی، از آنجایی که سرویس ها در فرایند خود اجرا می شوند، شما بایستی برای اتصال و تبادل داده با سرویس از سایر بخش ها از IPC بهره بگیرید.

حتی اگر سرویس در فرایند خود اجرا شود، برای دسترسی به اینترنت، برنامه نویس ملزم به استفاده از پردازش ناهمزمان خواهد بود چرا که اندروید تحت هیچ عنوانی اجازه ی دسترسی به اینترنت را در thread اصلی یک فرایند نمی دهد.

2-4-6-چه زمان می بایست یک سرویس را در فرایند مجزا اجرا کرد؟

اگر یک سرویس را در فرایند جداگانه و مختص به خود اجرا کنید، در آن صورت یک فضا با آدرس ویژه در حافظه به آن تخصیص یافته و garbage collector دستگاه مجازی (AVD) در این فرایند، در فرایند کلی اپلیکیشن دخالتی نخواهد داشت.

اپلیکیشن به ندرت نیاز پیدا می کند که سرویس را در فرایند ویژه و مجزا راه اندازی کند. اجرای سرویس ها در بستر فرایندهای مجزا، سبب می شود پیاده سازی ارتباط و تبادل داده بین سایر کامپوننت های نرم افزاری اندروید و سرویس به تبع دشوار شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

نکته: اگر می خواهید دسترسی به یک سرویس را در اختیار اپلیکیشن دیگری قرار دهید، در آن صورت سرویس بایستی در فرایند مختص به خود اجرا شود.

6-4-تبادل داده و ارتباط با سرویس ها

6-4-1-روش های مختلف برای برقراری ارتباط با سرویس ها

راه های مختلفی برای تبادل داده و تعامل بین activity و سرویس وجود دارد. مطالب زیر روش های ممکن برای نیل به این هدف را نام برده و روش پیشنهادی خود را در اختیار شما قرار می دهد.

6-4-2-استفاده از داده های کپسوله شده در intent

در یک سناریوی ساده، نیازی به هیچ تعامل مستقیم (بین سرویس و activity) وجود ندارد. سرویس داده های کپسوله شده در intent را از کامپوننت آغاز کننده (فراخواننده ی سرویس) دریافت نموده و عملیات لازم را به انجام می رساند. لازم به ذکر است که برای این منظور notification ضروری نیست. در واقع در شرایطی که سرویس محتوای یک content provider را با داده های جدید بروز آوری می کند، خود کامپوننت نرم افزاری مزبور activity را از این رخداد با خبر کرده و هیچ اقدام یا مرحله ی دیگری در سرویس لازم نیست. این روش هم برای سرویس های محلی و هم برای سرویس هایی که در فرایند مختص به خود اجرا می شوند، قابل پیاده سازی و استفاده خواهد بود.

6-4-3-استفاده از receiver

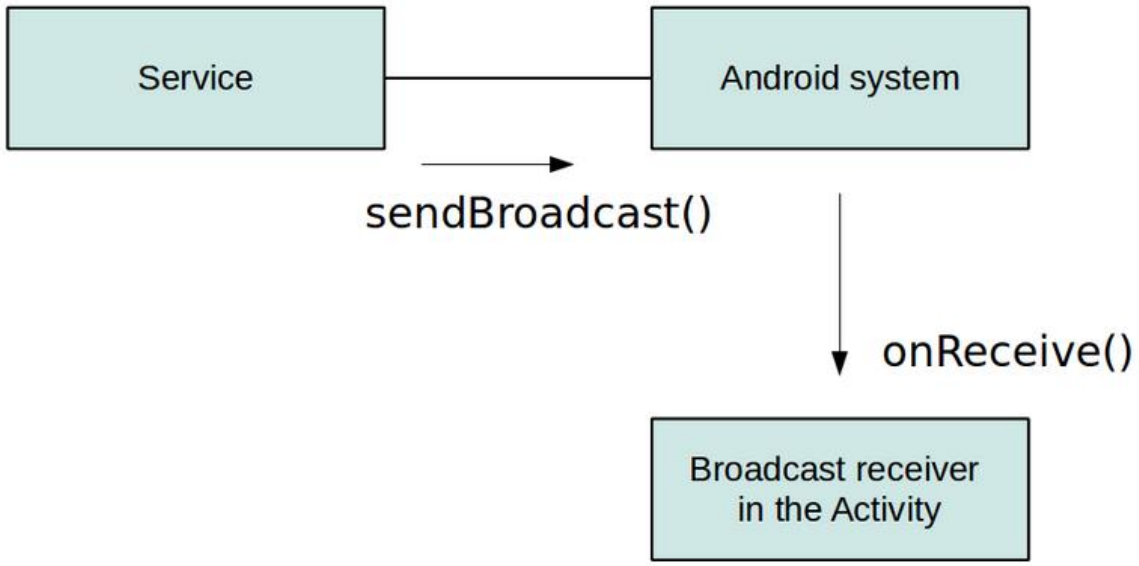
می توان برای تعامل و ارتباط بین activity و سرویس ها از broadcast ها و receiver هایی که به این broadcast ها گوش می دهند، بهره گرفت. به عنوان مثال، activity شما می تواند یک broadcast receiver برای گوش دادن به event ای معین ثبت کند و سرویس مورد نظر اتفاق افتادن event های مربوطه را به بیرون (کاپوننت های دیگر) اعلان نماید. این روش بسیار معمول

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بوده و اغلب زمانی استفاده می شود که سرویس می بایست پس از به انجام رساندن پردازش به activity این اتفاق را اعلان کند.

این جریان ارتباطی در تصویر زیر به تصویر کشیده شده است.



روش نام برده برای سرویس های محلی و سرویس هایی که در فرایند میزبان و مختص خود اجرا می شوند، قابل استفاده می باشد.

4-4-6-اتصال activity به سرویس محلی

چنانچه activity و service هر دو در فرایندی یکسان اجرا شوند، در این شرایط activity قادر خواهد بود به سرویس مستقیما به صورت دو طرفه وصل شود. روش حاضر از میان گزینه هایی که تاکنون عنوان شده، بهینه ترین بوده و برای زمانی که activity نیاز دارد با سرویس با سرعت بالا تبادل داده داشته باشد بسیار مناسب می باشد.

ناگفته نماند که این روش تنها برای سرویس های محلی (سرویس هایی که) قابل پیاده سازی و استفاده می باشد.

6-4-5- Handler و ResultReceiver یا Messenger

چنانچه سرویس با activity تعامل دو طرفه داشته باشد (اطلاعاتی را به activity برگرداند)، در آن صورت می تواند از طریق داده های کپسوله شده در intent که از activity دریافت می کند، آبجکتی از جنس Messenger دریافت نماید. در صورتی که Messenger به Handler در activity متصل باشد، آنگاه service قادر خواهد بود آبجکت هایی از جنس Message را به activity ارسال نماید.

Messenger در واقع اینترفیس parcelable را پیاده سازی می کند، بدین معنی که می توان آن را به فرایند دیگر ارسال کرده و با استفاده از این آبجکت Message هایی را به Handler در activity ارسال نمود.

Messenger همچنین متدی به نام ()getBinder را ارائه می دهد. این متد قابلیت ارسال آبجکتی از جنس Messenger به activity را فراهم می نماید. activity نیز متعاقبا قادر خواهد بود Message های (نمونه هایی از کلاس Message که حاوی توصیف و آبجکت های داده ای دلخواه می باشد) متعددی را به سرویس مورد نظر ارسال کند.

این روش برای سرویس های محلی که در فرایند خود اجرا می شوند، قابل استفاده می باشد.

6-4-6- اتصال به سرویس در فرایند دیگر با استفاده از AIDL

به منظور تبادل داده و اتصال (bind) به سرویسی که در فرایند دیگری در حال اجرا است، برنامه نویس بایستی از IPC (ارتباط میان پردازشی) کمک بگیرد. برای نیل به این هدف، ابتدا لازم است یک فایل AIDL ایجاد کند که تقریبا مشابه interface های جاوا می باشد با این تفاوت که پسوند آن aidl. بوده و تنها اجازه ی ارث بری و بسط دیگر فایل های AIDL را دارد.

توصیه می شود از این روش زمانی استفاده نمایید که لازم باشد به سرویسی که در فرایند دیگری در حال اجرا است متصل شوید. برای مثال زمانی که اپلیکیشن های دیگر درخواست استفاده از سرویس مورد نظر را داشته باشند باید از این رویکرد استفاده کرد.

تمرین: استفاده از Service ها و تبادل داده با سرویس

در مثال زیر یک activity با کنترل دکمه پیاده سازی می کنید که از سرویس برای دانلود فایل از اینترنت استفاده می کند. زمانی که کاربر بر روی دکمه در activity کلیک می کند، دانلود اطلاعات از اینترنت آغاز می گردد. پس از به پایان رسیدن دانلود، سرویس از طریق broadcast receiver به activity این اتفاق را اعلان می کند.

در مثال کاربردی حاضر از کلاس IntentService استفاده خواهید نمود چرا که این کلاس پردازش های پس زمینه ای را به صورت خودکار مدیریت می نماید.

یک پروژه و activity جدید به ترتیب به نام های com.vogella.android.service.receiver و MainActivity ایجاد نمایید.

سپس کلاس زیر را برای پیاده سازی سرویس تعریف نمایید.

```
package com.vogella.android.service.receiver;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import android.app.Activity;
import android.app.IntentService;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.os.Message;
import android.os.Messenger;
import android.util.Log;
public class DownloadService extends IntentService {
    private int result = Activity.RESULT_CANCELED;
```

```
public static final String URL = "urlpath";
public static final String FILENAME = "filename";
public static final String FILEPATH = "filepath";
public static final String RESULT = "result";
public static final String NOTIFICATION = "com.vogella.android.service.receiver";
public DownloadService() {
	super("DownloadService");
}
// will be called asynchronously by Android
@Override
protected void onHandleIntent(Intent intent) {
	String urlPath = intent.getStringExtra(URL);
	String fileName = intent.getStringExtra(FILENAME);
	File output = new File(Environment.getExternalStorageDirectory(),
			fileName);
	if (output.exists()) {
		output.delete();
	}
	InputStream stream = null;
	FileOutputStream fos = null;
	try {
		URL url = new URL(urlPath);
		stream = url.openConnection().getInputStream();
		InputStreamReader reader = new InputStreamReader(stream);
		fos = new FileOutputStream(output.getPath());
		int next = -1;
		while ((next = reader.read()) != -1) {
			fos.write(next);
		}
		// successfully finished
		result = Activity.RESULT_OK;
	} catch (Exception e) {
		e.printStackTrace();
	} finally {
		if (stream != null) {
			try {
				stream.close();
			} catch (IOException e) {
				e.printStackTrace();
			}
		}
		if (fos != null) {
			try {
				fos.close();
			} catch (IOException e) {
				e.printStackTrace();
			}
		}
	}
	publishResults(output.getAbsolutePath(), result);
}
private void publishResults(String outputPath, int result) {
```

```
            Intent intent = new Intent(NOTIFICATION);
            intent.putExtra(FILEPATH, outputPath);
            intent.putExtra(RESULT, result);
            sendBroadcast(intent);
        }
}
```

این کلاس را در لایه ی XML، فایل تنظیمات اپلیکیشن AndroidManifest.xml اضافه نمایید. حال زمان آن رسیده تا مجوزهای لازم را برای درج داده (write) در حافظه ی خارجی و دسترسی به اینترنت را نیز اعطا نمایید. پس از وارد کردن تغییرات فوق، محتوای AndroidManifest.xml می بایست ظاهری مشابه زیر داشته باشد.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.vogella.android.service.receiver"
   android:versionCode="1"
   android:versionName="1.0" >
   <uses-sdk
      android:minSdkVersion="17"
      android:targetSdkVersion="18" />
   <uses-permission android:name="android.permission.INTERNET" />
   <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
   <application
      android:allowBackup="true"
      android:icon="@drawable/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >
      <activity
         android:name="com.vogella.android.service.receiver.MainActivity"
         android:label="@string/app_name" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
      <service android:name="com.vogella.android.service.receiver.DownloadService" >
      </service>
   </application>
</manifest>
```

فایل layout مربوط به activity خود را به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
```

```
    <Button
      android:id="@+id/button1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:onClick="onClick"
      android:text="Download" />
    <LinearLayout
      android:layout_width="wrap_content"
      android:layout_height="wrap_content" >
      <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Status: " />
      <TextView
        android:id="@+id/status"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Not started" />
    </LinearLayout>
</LinearLayout>
```

کلاس MainActivity خود را به صورت زیر ویرایش نمایید.

```
package com.vogella.android.service.receiver;
import android.app.Activity;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
import android.widget.Toast;
public class MainActivity extends Activity {
      private TextView textView;
      private BroadcastReceiver receiver = new BroadcastReceiver() {
            @Override
            public void onReceive(Context context, Intent intent) {
                  Bundle bundle = intent.getExtras();
                  if (bundle != null) {
                        String string = bundle.getString(DownloadService.FILEPATH);
                        int resultCode = bundle.getInt(DownloadService.RESULT);
                        if (resultCode == RESULT_OK) {
                              Toast.makeText(MainActivity.this,
                                          "Download complete. Download URI: " + string,
                                          Toast.LENGTH_LONG).show();
                              textView.setText("Download done");
                        } else {
                              Toast.makeText(MainActivity.this, "Download failed",
                                          Toast.LENGTH_LONG).show();
                              textView.setText("Download failed");
```

```
                }
            }
        }
    };
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        textView = (TextView) findViewById(R.id.status);
    }
    @Override
    protected void onResume() {
        super.onResume();
        registerReceiver(receiver, new IntentFilter(
                DownloadService.NOTIFICATION));
    }
    @Override
    protected void onPause() {
        super.onPause();
        unregisterReceiver(receiver);
    }
    public void onClick(View view) {
        Intent intent = new Intent(this, DownloadService.class);
        // add infos for the service which file to download and where to store
        intent.putExtra(DownloadService.FILENAME, "index.html");
        intent.putExtra(DownloadService.URL,
                "http://www.vogella.com/index.html");
        startService(intent);
        textView.setText("Service started");
    }
}
```

حال زمانی که برنامه ی فوق را اجرا کرده و بر روی دکمه کلیک می کنید، سرویس بلافاصله اقدام به بارگیری اطلاعات لازم از اینترنت می نماید. با اتمام دانلود، رابط کاربری بروز رسانی شده و یک پیغام Toast با اسم فایل به نمایش در می آید.

تنظیمات را طوری تغییر دهید که سرویس در فرایند ویژه ی خود اجرا شود. بار دیگر اپلیکیشن را تست کرده و اطمینان حاصل نمایید که همچنان به درستی عمل می کند چرا که با وارد نمودن تغییرات جدید، broadcast receiver ها از فرایندهای مختلف broadcast ها را دریافت می کنند.

تمرین: پیاده سازی و استفاده از سرویس محلی (local service)

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

تمرین حاضر نحوه ی اتصال از یک activity به سرویس محلی (سرویسی که با activity در یک فرایند اجرا می شود) را به صورت عملی نمایش می دهد.

پس از اینکه دستگاه بوت می شود، سرویس راه اندازی شده و داده هایی را در فواصل زمانی مشخص از اینترنت دانلود می نماید. حال Activity، خود را به سرویس متصل کرده تا به داده هایی که اخیرا از اینترنت بارگیری شده دسترسی داشته باشد.

یک پروژه و activity جدید به ترتیب به نام های de.vogella.android.ownservice.local و MainActivity تعریف نمایید.

کلاسی به نام LocalWordService با بدنه ی زیر ایجاد نمایید.

```
package de.vogella.android.ownservice.local;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import android.app.Service;
import android.content.Intent;
import android.os.Binder;
import android.os.IBinder;
public class LocalWordService extends Service {
    private final IBinder mBinder = new MyBinder();
    private ArrayList<String> list = new ArrayList<String>();
    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        Random random = new Random();
        if (random.nextBoolean()) {
            list.add("Linux");
        }
        if (random.nextBoolean()) {
            list.add("Android");
        }
        if (random.nextBoolean()) {
            list.add("iPhone");
        }
        if (random.nextBoolean()) {
            list.add("Windows7");
        }
        if (list.size() >= 20) {
            list.remove(0);
        }
        return Service.START_NOT_STICKY;
    }
```

```
    @Override
    public IBinder onBind(Intent arg0) {
        return mBinder;
    }
    public class MyBinder extends Binder {
        LocalWordService getService() {
            return LocalWordService.this;
        }
    }
    public List<String> getWordList() {
        return list;
    }
}
```

دو کلاس با پیاده سازی زیر تعریف نمایید. این دو کلاس در لایه ی XML متناظر به عنوان BroadcastReceivers معرفی و ثبت خواهند شد.

```
package de.vogella.android.ownservice.local;
import java.util.Calendar;
import android.app.AlarmManager;
import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
public class MyScheduleReceiver extends BroadcastReceiver {
    // restart service every 30 seconds
    private static final long REPEAT_TIME = 1000 * 30;
    @Override
    public void onReceive(Context context, Intent intent) {
        AlarmManager service = (AlarmManager) context
                .getSystemService(Context.ALARM_SERVICE);
        Intent i = new Intent(context, MyStartServiceReceiver.class);
        PendingIntent pending = PendingIntent.getBroadcast(context, 0, i,
                PendingIntent.FLAG_CANCEL_CURRENT);
        Calendar cal = Calendar.getInstance();
        // start 30 seconds after boot completed
        cal.add(Calendar.SECOND, 30);
        // fetch every 30 seconds
        // InexactRepeating allows Android to optimize the energy consumption
        service.setInexactRepeating(AlarmManager.RTC_WAKEUP,
                cal.getTimeInMillis(), REPEAT_TIME, pending);
        // service.setRepeating(AlarmManager.RTC_WAKEUP, cal.getTimeInMillis(),
        // REPEAT_TIME, pending);
    }
}
package de.vogella.android.ownservice.local;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
```

```
public class MyStartServiceReceiver extends BroadcastReceiver {
      @Override
      public void onReceive(Context context, Intent intent) {
            Intent service = new Intent(context, LocalWordService.class);
            context.startService(service);
      }
}
```

تمامی کامپوننت های نرم افزاری را بایستی در فایل تنظیمات اپلیکیشن (AndroidManifest.xml) معرفی نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="de.vogella.android.ownservice.local"
   android:versionCode="1"
   android:versionName="1.0" >
   <uses-sdk android:minSdkVersion="10" />
   <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
   <application
      android:icon="@drawable/icon"
      android:label="@string/app_name" >
      <activity
         android:name=".MainActivity"
         android:label="@string/app_name" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
      <service
         android:name=".LocalWordService"
         android:icon="@drawable/icon"
         android:label="@string/service_name" >
      </service>
      <receiver android:name="MyScheduleReceiver" >
         <intent-filter>
            <action android:name="android.intent.action.BOOT_COMPLETED" />
         </intent-filter>
      </receiver>
      <receiver android:name="MyStartServiceReceiver" >
      </receiver>
   </application>
</manifest>
```

محتوای فایل layout مربوط به activity را به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="fill_parent"
```

```
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello" />
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="onClick"
        android:text="Update" >
    </Button>
    <ListView
        android:id="@id/android:list"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
    </ListView>
</LinearLayout>
```

کلاس activity خود را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.ownservice.local;
import java.util.ArrayList;
import java.util.List;
import android.app.ListActivity;
import android.content.ComponentName;
import android.content.Context;
import android.content.Intent;
import android.content.ServiceConnection;
import android.os.Bundle;
import android.os.IBinder;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Toast;
public class MainActivity extends ListActivity {
      private LocalWordService s;
      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main);
            wordList = new ArrayList<String>();
            adapter = new ArrayAdapter<String>(this,
                        android.R.layout.simple_list_item_1, android.R.id.text1,
                        wordList);
            setListAdapter(adapter);
      }
      @Override
      protected void onResume() {
            super.onResume();
            Intent intent= new Intent(this, LocalWordService.class);
```

```
        bindService(intent, mConnection,
                Context.BIND_AUTO_CREATE);
    }
    @Override
    protected void onPause() {
        super.onPause();
        unbindService(mConnection);
    }
    private ServiceConnection mConnection = new ServiceConnection() {
        public void onServiceConnected(ComponentName className,
                IBinder binder) {
            LocalWordService.MyBinder b = (LocalWordService.MyBinder) binder;
            s = b.getService();
            Toast.makeText(MainActivity.this, "Connected", Toast.LENGTH_SHORT)
                    .show();
        }
        public void onServiceDisconnected(ComponentName className) {
            s = null;
        }
    };
    private ArrayAdapter<String> adapter;
    private List<String> wordList;
    public void onClick(View view) {
        if (s != null) {
            Toast.makeText(this, "Number of elements" + s.getWordList().size(),
                    Toast.LENGTH_SHORT).show();
            wordList.clear();
            wordList.addAll(s.getWordList());
            adapter.notifyDataSetChanged();
        }
    }
}
```

اپلیکیشن را اجرا نمایید. پس از کلیک بر روی دکمه، داده ها بار دیگر از سرویس واکشی شده و لیست آیتم ها/ListView با داده های جدید بروز آوری می شوند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش دوم :

زمان بندی تسک ها در اندروید با استفاده از توابع کتابخانه ای JobScheduler

این آموزش نحوه ی زمان بندی تسک ها در اندروید را با استفاده از توابع JobScheduler (API) تشریح می کند.

6-5-زمان بندی تسک ها

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

6-5-1-نحوه ی پیاده سازی

فرض کنید تسکی در اپلیکیشن اندرویدی خود دارید که بایستی بارها اجرا شود. در چنین شرایطی لازم است این را هم در نظر داشته باشید که سیستم اندروید ممکن است با توجه به شرایط activity ها و service ها را برای آزاد سازی منابع از حافظه حذف نماید. از این جهت نمی توانید برای زمان بندی تسک ها از کلاس های ساده ی محیط (platform) جاوا همچون TimerTask استفاده نمایید.

6-5-2-روش ها و ابزار مختلف زمان بندی تسک ها

سیستم اندروید دو ابزار برای زمان بندی اجرای تسک ها به شرح زیر ارائه می دهد:

- AlarmManager (روش قدیمی)
- JobScheduler API

اپلیکیشن های تحت موبایلی که امروزه برای محیط اندروید ساخته می شوند، برای زمان بندی و مدیریت اجرای تسک ها می بایست از توابع کتابخانه ای JobScheduler (API) استفاده نمایند. اپلیکیشن های تحت موبایل شما می توانند job ها را برای اجرا در زمان معین زمان بندی کرده، به سیستم اجازه دهند تا اجرای آن ها را بر اساس میزان حافظه، باتری و وضعیت اتصال مدیریت کند.

6-6-زمان بندی background task ها با استفاده از JobScheduler

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-6-6-توابع کتابخانه ای job scheduler

به طور کلی Job scheduler یک برنامه است که برای کنترل و مدیریت غیرحضوری برنامه‌هایی که در background سیستم اجرا می‌شوند (غالبا برای پردازش دسته‌ای)، مورد استفاده قرار می گیرد.

ویرایش 5.0 اندروید (API 21) توابع کتابخانه ای به نام job scheduler را در قالب کلاس JobScheduler در اختیار برنامه نویس قرار می دهد. API نام برده این امکان را فراهم می آورد تا هنگامی که منابع در اختیار سیستم به اندازه ی کافی رسید، job ها را به صورت دسته ای (batch) در پس زمینه به اجرا بگذارد. به بیان دیگر، توسعه دهنده می تواند با استفاده از این API تمامی کارهایی که از نظر زمان اجرا اهمیت چندانی برای کاربر ندارند را زمان بندی کند.

2-6-6-مزایای استفاده از JobScheduler

JobScheduler بر خلاف SyncAdapter اختصاصی یا alarm manager این قابلیت را دارد که job ها را برای پردازش دسته جمعی زمان بندی کند (آن ها را زمان بندی کرده و به هنگام رسیدن موعد تمامی کارها را دسته جمعی پردازش کند). در واقع JobScheduler با این کار (دسته بندی job ها برای پردازش دسته جمعی) میزان مصرف باتری را کاهش داده و کارایی کلی را بالا می برد.

یکی از مزایای استفاده از JobManager این است که مدیریت بارگذاری محتوا در وب را آسان ساخته و مسئله ی ناپایایی (unreliability) اینترنت را بر طرف می سازد. علاوه بر آن در صورت restart/بازآغازی، اپلیکیشن را طوری مدیریت می کند که خللی به وجود نیاید. برخی از موارد استفاده ی این JobScheduler را در زیر مشاهده می کنید:

- تسک هایی که به هنگام متصل بودن دستگاه به منبع تغذیه بایستی اجرا شوند.
- تسک هایی که برای اجرا نیاز به دسترسی به اینترنت یا Wi-Fi دارند.
- تسک هایی که ضروری نیستند یا مستقیما با کاربر تعامل ندارند.

- تسک هایی که زمان اجرای آن ها چندان اهمیتی ندارد و از این جهت می توان آن ها را گونه ای برنامه ریزی کرد که به صورت دسته جمعی و در زمان مشخصی در آینده اجرا شوند.

3-6-6-نحوه ی ایجاد یک Job

آبجکت JobInfo حامل اطلاعات ارسالی به JobScheduler بوده و کاملا پارامترهای مورد نیاز برای زمان بندی کارهایی که بر روی اپلیکیشن فراخواننده بایستی انجام شود را در خود کپسوله می کند.

به عبارت دیگر آبجکت JobInfo دربردارنده ی یک واحد کاری (unit of work) می باشد. این آبجکت مشخصات زمان بندی و کارهایی که باید انجام شود را تعیین می کند.

Job scheduler به شما این امکان را می دهد تا وضعیت دستگاه را بررسی کنید، برای مثال اینکه دستگاه بیکار است یا دسترسی به اینترنت در زمان مورد نظر امکان پذیر می باشد یا خیر و سپس بر اساس آن کارهای زمان بندی شده را انجام دهید.

جهت تعیین تنظیمات کارهای زمان بندی شده (محتوای کپسوله شده در آبجکت JobInfo) کافی است از کلاس JobInfo.Builder استفاده نمایید. می توانید تسک ها را طوری زمان بندی کنید که تحت شرایط خاصی اجرا می شود. این شرایط به شرح زیر هستند:

- زمانی که دستگاه در حال شارژ می باشد.
- زمانی که دستگاه بیکار است.
- دستگاه به اینترنت unmetered و نامحدود دسترسی دارد.
- زمانی که عملیاتی باید قبل از تاریخ نهایی و مشخصی انجام شود.
- زمانی که کاری باید طی زمان مشخصی انجام شود برای مثال طی یک ساعت آینده.
- زمانی که عملیات باید پس از یک تاخیر جزئی به اجرا در آید، برای مثال 10 دقیقه صبر کرده و پس از آن کار زمان بندی شده را انجام دهد.

این محدودیت ها را می توان ترکیب کرد. برای مثال، می توانید یک job را به صورتی زمان بندی کنید که هر 20 دقیقه یکبار و هنگامی که دستگاه به اینترنت نامحدود وصل است اجرا شود.

deadline یک تاریخ معین است که به عنوان محدودیت (constraint) زمانی اعمال شده و عملیات نهایتا تا زمان سر رسید آن باید انجام شود.

به منظور پیاده سازی یک job، می بایست کلاس JobService را به ارث برده (extend) و سپس توابع onStartJob و onStopJob را پیاده سازی نمایید. چنانچه job مورد نظر به هر دلیلی ناموفق بود، لازم است مقدار بولی true را از onStopJob برگردانید تا job ناموفق دوباره انجام شود. متد onStartJob در thread اصلی اجرا می شود، اگر پردازش ناهمزمان را در این متد آغاز کنید، در آن صورت می بایست مقدار بولی true و در غیر صورت false را برگردانید.

تمرین: زمان بندی background job ها با استفاده از JobScheduler

هدف تمرین

در تمرین حاضر، یک نمونه اپلیکیشن ساده می نویسید که job ها را با فراخوانی توابع کتابخانه ای JobScheduler (API) زمان بندی می کند.

ایجاد پروژه و فایل layout

یک پروژه ی جدید اندروید به نام com.vogella.android.jobscheduler بر اساس ورژن/API 21 کتابخانه های اندروید ایجاد نمایید.

حال فایل layout جدید ایجاد کنید که حداقل یک کنترل دکمه در آن تعریف شده باشد. سپس متد onClick را به property مربوطه (onClick در لایه ی XML) تخصیص دهید.

4-6-6-ایجاد JobService

یک سرویس با پیاده سازی زیر ایجاد نمایید.

```
package com.vogella.android.jobscheduler;
import java.util.LinkedList;
import android.app.job.JobInfo;
import android.app.job.JobParameters;
import android.app.job.JobScheduler;
import android.app.job.JobService;
```

```
import android.content.Context;
import android.content.Intent;
import android.os.Message;
import android.os.Messenger;
import android.os.RemoteException;
import android.util.Log;
/**
 * JobService to be scheduled by the JobScheduler.
 * Requests scheduled with the JobScheduler call the "onStartJob" method
 */
public class TestJobService extends JobService {
	private static final String TAG = "SyncService";
	@Override
	public boolean onStartJob(JobParameters params) {
		// fake work
		Log.i(TAG, "on start job: " + params.getJobId());
		return true;
	}
	@Override
	public boolean onStopJob(JobParameters params) {
		return true;
	}
}
```

نکته: لازم است سرویس را با استفاده از تگ service در لایه ی XML (فایل تنظیمات manifest) تعریف نمایید.

5-6-6-ساخت activity آزمایشی

پیاده سازی کلاس activity خود را به صورت زیر ویرایش نمایید.

```
package com.vogella.android.jobscheduler;
import android.app.Activity;
import android.app.job.JobInfo;
import android.app.job.JobScheduler;
import android.content.ComponentName;
import android.content.Context;
import android.content.res.Resources;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.RadioButton;
public class MainActivity extends Activity {
	TestJobService testService;
	private static int kJobId = 0;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
```

```
            setContentView(R.layout.activity_main);
        }
        public void onClick(View v) {
            ComponentName mServiceComponent = new ComponentName(this, TestJobService.class);
            JobInfo.Builder builder = new JobInfo.Builder(kJobId++, serviceComponent);
            builder.setMinimumLatency(5 * 1000); // wait at least
            builder.setOverrideDeadline(50 * 1000); // maximum delay
           builder.setRequiredNetworkType(JobInfo.NETWORK_TYPE_UNMETERED); // require unmetered
network
            builder.setRequiresDeviceIdle(true); // device should be idle
            builder.setRequiresCharging(false); // we don't care if the device is charging or not
            JobScheduler jobScheduler = getApplication().getSystemService(Context.JOB_SCHEDULER_SERVICE);
            jobScheduler.schedule(builder.build());
        }
        public void cancelAllJobs(View v) {
            JobScheduler tm = (JobScheduler) getSystemService(Context.JOB_SCHEDULER_SERVICE);
            tm.cancelAll();
        }
}
```

تمرین: فراخوانی JobScheduler از داخل یک receiver/راه اندازی و اجرای یک job -> service از طریق Receiver

هدف از تمرین

در این تمرین به صورت کاملا کاربردی خواهید آموخت چگونه می توان یک job را که داخل receiver تعریف شده، با اتفاق افتادن رخداد خاصی (که receiver به آن گوش می دهد) فعال نمایید.

6-6-6-ایجاد receiver

Receiver را به صورت زیر پیاده سازی کنید.

```
package com.example.android.rssreader;
import android.app.job.JobInfo;
import android.app.job.JobScheduler;
import android.content.BroadcastReceiver;
import android.content.ComponentName;
import android.content.Context;
import android.content.Intent;
public class BootReceiver extends BroadcastReceiver {
   @Override
   public void onReceive(Context context, Intent intent) {
```

```
        ComponentName serviceComponent = new ComponentName(context, MyJobService.class);
        JobInfo.Builder builder = new JobInfo.Builder(0, serviceComponent);
        builder.setMinimumLatency(5 * 1000); // wait at least
        builder.setOverrideDeadline(50 * 1000); // maximum delay
        builder.setRequiredNetworkType(JobInfo.NETWORK_TYPE_UNMETERED); // require unmetered network
        builder.setRequiresDeviceIdle(true); // device should be idle
        builder.setRequiresCharging(false); // we don't care if the device is charging or not
        JobScheduler jobScheduler = context.getSystemService(JobScheduler.class);
        jobScheduler.schedule(builder.build());
    }
}
```

7-6-6-ایجاد Job

```
package com.example.android.rssreader;
import android.app.job.JobParameters;
import android.app.job.JobService;
import android.content.Intent;
/**
 * Created by vogella on 30.06.16.
 */
public class MyJobService extends JobService {
    @Override
    public boolean onStartJob(JobParameters params) {
        String url = params.getExtras().getString(RssApplication.URL);
        Intent i = new Intent(this, RssDownloadService.class); // starts the RssDownloadService service
        i.putExtra(RssApplication.URL, url); // some extra data for the service
        startService(i);
        return false; // true if we're not done yet
    }
    @Override
    public boolean onStopJob(JobParameters params) {
        // true if we'd like to be rescheduled
        return true;
    }
}
```

می توانید receiver (گوش دهنده به رخدادها) را برای event خاصی همچون رخداد boot و بالا آمدن سیستم ثبت و معرفی نمایید. event مذکور receiver مربوطه را از اتفاق مورد نظر با خبر نموده که آن خود job ای که سرویس اختصاصی را اجرا می کند، فعال می نماید.

بخش سوم :

آموزش استفاده از Broadcast receiver (گوش فراخوان برای رخدادها) در اندروید

این آموزش به شرح نحوه ی ایجاد و استفاده از کامپوننت نرم افزاری broadcast receiver در اندروید را شرح می دهد.

Broadcast receiver-7-6

broadcast receiver شرح مفهوم–6–7–1

Broadcast receiver یک کامپوننت نرم افزاری در اندروید است که به شما این امکان را می دهد تا به event های مختلف گوش فرا دهید. به عبارت دیگر broadcast receiver یک نوع گوش فراخوان به رخدادها است.

زمانی که رخداد یا event مورد نظر رخ می دهد، runtime سیستم اندروید تمامی receiver هایی که برای آن رخداد ثبت شده و به آن گوش می دهند را با خبر می کند.

برای مثال می توان به اپلیکیشن هایی اشاره کرد که به رخداد سیستمی ACTION_BOOT_COMPLETED گوش داده و زمانی که سیستم اندروید پروسه ی boot را به اتمام می رساند، اپلیکیشن را از این اتفاق با خبر می سازد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

6-7-2-پیاده سازی receiver

Receiver را بایستی در لایه ی XML، درون فایل تنظیمات AndroidManifest.xml تعریف نمایید.

یک receiver را می توان به صورت فوق تعریف کرد که روش static نام دارد یا می توان آن را به صورت dynamic در کلاس های جاوا با فراخوانی متد ()Context.registerReceiver اعلان نمایید.

برای تعریف receiver لازم است از کلاس BroadcastReceiver ارث بری نمایید.

اگر event ای که broadcast receiver برای آن ثبت شده رخ دهد، سیستم اندروید متد ()onReceive را از receiver را صدا می زند.

6-7-3-Lifecycle یا چرخه ی حیات کامپوننت broadcast receiver

پس از اینکه اجرای متد ()onReceive از کلاس receiver به پایان رسید، سیستم اندروید این اجازه را دارد که receiver را برای استفاده ی مجدد در آینده (recycle) بازیافت نماید.

6-7-4-پردازش ناهمزمان (async processing)

پیش از API 11 توسعه دهندگان امکان اجرای هیچ گونه عملیات ناهمزمانی را در متد ()onReceive نداشتند چرا که به محض به اتمام رسیدن اجرای متد ()onReceive سیستم اندروید می توانست آن کامپوننت را بازیافت نماید. چنانچه عملیات طولانی برای اجرا وجود داشته باشد، در این شرایط بهتر است بجای broadcast receiver، یک کامپوننت سرویس را فراخوانی نمایید.

از ورژن 11 کتابخانه های اندروید، شما می توانید متد ()goAsync را برای اجرای عملیات ناهمزمان فراخوانی نمایید. این متد در خروجی آبجکتی از جنس PendingResult برمی گرداند. لازم به ذکر است که تا زمانی که متد ()PendingResult.finish را بر روی این آبجکت فراخوانی نکرده اید، سیستم اندروید receiver را همچنان فعال در نظر می گیرد. از این طریق شما می توانید در receiver، عملیات ناهمزمان راه اندازی نمایید. بلافاصله پس از اینکه thread پردازش خود را به

پایان می رساند، تسک (میزبان) متد finish() را صدا زده و به سیستم اندروید اعلان می کند که کامپوننت مورد بحث اکنون قابل بازیافت (استفاده ی مجدد و recycle) می باشد.

6-7-5-محدودیت هایی در تعریف و استفاده از broadcast receiver

از ویرایش 3.1 به بعد اندروید، چنانچه اپلیکیشن مورد نظر را کاربر راه اندازی نکرده یا اینکه آن را به صورت صریح از طریق منوی اندروید (Manage ‣ Application) خاتمه داده باشد، آنگاه سیستم به صورت پیش فرض تمامی receiver ها را از دریافت intent ها محروم خواهد نمود.

این یک امکان امنیتی است که به واسطه ی آن کاربر اطمینان دارد تنها اپلیکیشن هایی که خود به صورت صریح اجرا کرده قادر به دریافت broadcast intent ها خواهند بود.

نکته: این بدین معنی نیست که کاربر بایستی اپلیکیشن را مجددا پس از reboot اجرا کند. بلکه سیستم اندروید به یاد دارد که کاربر قبلا آن را یکبار اجرا کرده است. بنابراین چنانچه کاربر اپلیکیشن را اجرا کرده باشد(آن را به صورت صریح نبسته باشد) سیستم اندروید پس از reboot به خاطر می آورد که برنامه ی مورد بحث یکبار قبلا اجرا شده و خود اقدام به اجرای مجدد آن می نماید.

6-7-6-ارسال broadcast به اپلیکیشن به منظور تست

دستورات زیر را در پنجره ی فرمان adb (adb command line tool) درج نمایید. اسم کلاس و پکیج که در دستور زیر مشاهده می کنید، می بایست قبلا در لایه ی XML، داخل فایل manifest تعریف شده باشد. شما بایستی intent و داده های کپسوله در آن را به کامپوننت مورد نظر ارسال نمایید. حال زمانی که broadcast سراسری به نام ACTION_BOOT_COMPLETED را به کامپوننت نرم افزاری مد نظر ارسال می کنید، چندین اتفاق متعدد در سیستم اندروید رخ می دهد.

```
# trigger a broadcast and deliver it to a component
adb shell am activity/service/broadcast -a ACTION -c CATEGORY -n NAME
```

```
# for example (this goes into one line)
adb shell am broadcast -a
android.intent.action.BOOT_COMPLETED -c android.intent.category.HOME -n
package_name/class_name
```

7-7-6-شرح مفهوم Pending Intent

Pending intent یک آبجکت متشکل از درخواست عملیات دلخواه (راه اندازی یک activity، اجرای سرویس یا ارسال broadcast) اطلاعات و جزئیات عملیات (کپسوله شده در یک intent) و در نهایت یک context می باشد. نمونه ی مشتق شده از کلاس PendingIntent را می توان به سایر اپلیکیشن های اندرویدی واگذار کرد تا حتی پس از بسته شدن برنامه ی میزبان یا ایجاد کننده ی pending intent، بتواند عملیاتی را به انجام برساند. زمانی که توسعه دهنده pending intent را به اپلیکیشن دیگری می فرستد، در عمل به آن برنامه اجازه می دهد تا مجوزهای اپلیکیشن ایجاد کننده ی pending intent را جهت انجام عملیات خاص داشته باشد. این بر خلاف intent های متعارف است که اپلیکیشن های دریافت کننده ی آن می بایست برای انجام عملیات از مجوزهای خود استفاده کنند.

به عبارت دیگر pending intent یک token است که برنامه نویس به اپلیکیشن دیگری مانند notification manager، alarm manger یا هر اپلیکیشن دیگری می دهد. با ارسال این نوع intent به اپلیکیشن دیگر، در واقع توسعه دهنده به اپلیکیشن مقصد این اجازه را می دهد تا مجوزهای اپلیکیشن فرستنده را جهت اجرای کد و عملیات از پیش تعریف شده داشته باشد.

برای اجرای یک broadcast از طریق pending intent، ابتدا متد getBroadcast() را فراخوانی نموده و آبجکتی از کلاس PendingIntent را دریافت نمایید. حال جهت اجرا activity از طریق pending intent، متد PendingIntent.getActivity() را صدا زده و activity مورد نظر را به عنوان خروجی از این متد دریافت می کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

6-8-Broadcast ها و رخدادهای سیستمی

کلاس Intent بسیاری از رخدادهای سیستمی را در قالب فیلدهای ایستا و ثابت (static و final) به صورت آماده (از قبل تعریف شده) در خود کپسوله می کند. سایر کلاس های سیستم اندروید نیز رخدادهای سیستمی دیگری را تعریف می کنند. به عنوان مثال می توان به TelephoneManager برای تغییر وضعیت گوشی اشاره کرد.

جدول زیر تعدادی رخداد سیستمی بسیار مهم را همراه با شرح کاربرد ذکر می کند.

جدول 1- رخدادهای سیستمی	
شرح کاربرد	Event
Boot سیستم با پایان رسیده است. برای اجرای نیاز به مجوز android.permission.RECEIVE_BOOT_COMPLETED دارد.	Intent.ACTION_BOOT_COMPLETED
دستگاه به منبع تغذیه وصل شده است.	Intent.ACTION_POWER_CONNECTED
دستگاه دیگر به منبع تغذیه وصل نیست.	Intent.ACTION_POWER_DISCONNECTED
زمانی اجرا می شود که باتری دستگاه رو به اتمام باشد. معمولا برای کاستن از تعداد activity های اپلیکیشنی که میزان مصرف باتری آن بالا است، مورد استفاده قرار می گیرد.	Intent.ACTION_BATTERY_LOW
باتری دستگاه بار دیگر در وضعیت مناسبی قرار دارد.	Intent.ACTION_BATTERY_OKAY

6-9-اجرا و راه اندازی سرویس به صورت خودکار از Receiver

یکی از رفتارهایی که زیاد مورد نیاز بوده و پیاده سازی می شود، فراخوانی و اجرای یک سرویس به صورت خودکار پس از reboot سیستم است. این کار می تواند برای مثال جهت همگاهنگ سازی داده ها صورت گیرد. برای این منظور ابتدا یک receiver (گوش فراخوان به رخداد) برای

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

android.intent.action.BOOT_COMPLETED تعریف می کنید. لازم است مجوز android.permission.RECEIVE_BOOT_COMPLETED را نیز در فایل تنظیمات برای اجرای عملیات مزبور به اپلیکیشن اعطا نمایید.

مثال زیر نحوه ی تعریف رخداد BOOT_COMPLETED در فایل تنظیمات (Manifest) را به نمایش می گذارد.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="de.vogella.android.ownservice.local"
   android:versionCode="1"
   android:versionName="1.0" >
   <uses-sdk android:minSdkVersion="10" />
   <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
   <application
      android:icon="@drawable/icon"
      android:label="@string/app_name" >
      <activity
         android:name=".ServiceConsumerActivity"
         android:label="@string/app_name" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
      <receiver android:name="MyScheduleReceiver" >
         <intent-filter>
            <action android:name="android.intent.action.BOOT_COMPLETED" />
         </intent-filter>
      </receiver>
      <receiver android:name="MyStartServiceReceiver" >
      </receiver>
   </application>
</manifest>
```

به دنبال تعریف رخداد فوق در لایه ی XML، کامپوننت receiver می بایست سرویس را به وسیله ی کد زیر به صورت خودکار و با بالا آمدن سیستم راه اندازی کند.

```
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
public class MyReceiver extends BroadcastReceiver {
      @Override
      public void onReceive(Context context, Intent intent) {
```

```
        // assumes WordService is a registered service
        Intent intent = new Intent(context, WordService.class);
        context.startService(intent);
    }
}
```

توجه: چنانچه اپلیکیشن در کارت SD (حافظه ی خارجی) مستقر باشد، در آن صورت پس از رخداد android.intent.action.BOOT_COMPLETED، اپلیکیشن مورد نظر دیگر در دسترس نخواهد بود. در این شرایط می بایست به رخداد android.intent.action.ACTION_EXTERNAL_APPLICATIONS_AVAILABLE گوش داده و Receiver را داخل فایل تنظیمات اپلیکیشن برای آن ثبت نمایید.

نکته: به خاطر داشته باشید از ورژن API 11 کتابخانه های اندروید، برای اینکه اپلیکیشن بتواند رخداد android.intent.action.BOOT_COMPLETED events. را دریافت کند، کاربر می بایست حداقل یکبار قبلا اپلیکیشن را اجرا کرده باشد.

تمرین: ثبت یک RECEIVER جهت گوش فراخوانی به تماس های دریافتی

هدف از تمرین

در تمرین حاضر یک broadcast receiver تعریف می کنید که به رخدادهای مربوط به وضعیت دستگاه تلفن گوش می دهد. اگر دستگاه یک تماس دریافتی داشته باشد، در آن صورت receiver از رخداد آگاه شده و یک پیغام مربوطه ثبت (log) می نماید.

ساخت پروژه

یک پروژه ی جدید به نام de.vogella.android.receiver.phone ایجاد نموده و سپس یک activity جدید تعریف نمایید.

نکته: به یاد داشته باشید که receiver تنها به شرطی که کاربر قبلا آن را یکبار اجرا کرده باشد، صدا خورده می شود. برای این منظور لازم است یک activity تعریف نمایید.

6-9-1-پیاده سازی receiver برای گوش فراخوانی به رخدادهای مربوطه به وضعیت تلفن و تماس ها

کلاس MyPhoneReceiver را ایجاد نمایید.

```
package de.vogella.android.receiver.phone;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.telephony.TelephonyManager;
import android.util.Log;
public class MyPhoneReceiver extends BroadcastReceiver {
      @Override
      public void onReceive(Context context, Intent intent) {
            Bundle extras = intent.getExtras();
            if (extras != null) {
                  String state = extras.getString(TelephonyManager.EXTRA_STATE);
                  Log.w("MY_DEBUG_TAG", state);
                  if (state.equals(TelephonyManager.EXTRA_STATE_RINGING)) {
                        String phoneNumber = extras
                                    .getString(TelephonyManager.EXTRA_INCOMING_NUMBER);
                        Log.w("MY_DEBUG_TAG", phoneNumber);
                  }
            }
      }
}
```

6-9-2-اعطا مجوز گوش فراخوانی به تغییرات مربوط به وضعیت در receiver

مجوز android.permission.READ_PHONE_STATE را به فایل تنظیمات (manifest) اضافه نموده تا اپلیکیشن قابلیت گوش فراخوانی به تغییرات در وضعیت (از طریق receiver) را داشته باشد. سپس receiver مورد نظر را در لایه ی XML (داخل فایل Manifest) تعریف نمایید. فایل manifest پس از وارد نمودن تغییرات فوق بایستی ظاهری مشابه زیر داشته باشد.

تست اپلیکیشن

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

اپلیکیشن خود را نصب نموده و سپس از طریق ADV (android device monitor) یک تماس تلفنی به دستگاه را شبیه سازی نمایید. مطمئن شوید که receiver فراخوانی شده و یک پیغام در LogCat view ثبت می شود.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="de.vogella.android.receiver.phone"
   android:versionCode="1"
   android:versionName="1.0" >
    <uses-sdk android:minSdkVersion="15" />
   <uses-permission android:name="android.permission.READ_PHONE_STATE" >
   </uses-permission>
   <application
      android:icon="@drawable/icon"
      android:label="@string/app_name" >
        <activity
         android:name=".MainActivity"
         android:label="@string/title_activity_main" >
          <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
          </intent-filter>
      </activity>
      <receiver android:name="MyPhoneReceiver" >
        <intent-filter>
          <action android:name="android.intent.action.PHONE_STATE" >
          </action>
        </intent-filter>
      </receiver>
   </application>
</manifest>
```

تمرین: receiver و سرویس های سیستم

هدف از تمرین

در این تمرین یک receiver را از طریق سرویس درون ساخته ی محیط اندروید Android alert manager زمان بندی خواهید کرد. پس از اینکه receiver فراخوانی می شود، این کامپوننت نرم افزاری با استفاده از Android vibrator manager و یک پیغام Toast (popup message) کاربر را از تغییرات مربوطه آگاه می سازد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

پیاده سازی پروژه

یک پروژه و activity جدید به ترتیب به نام های de.vogella.android.alarm و AlarmActivity ایجاد نمایید.

سپس فایل یک layout با محتوای زیر ایجاد کنید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <EditText
      android:id="@+id/time"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:hint="Number of seconds"
      android:inputType="numberDecimal" >
   </EditText>
   <Button
      android:id="@+id/ok"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:onClick="startAlert"
      android:text="Start Counter" >
   </Button>
</LinearLayout>
```

کلاس broadcast receiver را به صورت زیر پیاده سازی نمایید. این کلاس سرویس vibrator را صدا می زند.

```
package de.vogella.android.alarm;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Vibrator;
import android.widget.Toast;
public class MyBroadcastReceiver extends BroadcastReceiver {
      @Override
      public void onReceive(Context context, Intent intent) {
            Toast.makeText(context, "Don't panik but your time is up!!!!.",
                  Toast.LENGTH_LONG).show();
```

```
        // Vibrate the mobile phone
        Vibrator vibrator = (Vibrator) context.getSystemService(Context.VIBRATOR_SERVICE);
        vibrator.vibrate(2000);
    }
}
```

این کلاس را در لایه ی XML، داخل فایل AndroidManifest.xml اعلان نموده و اجازه ی vibrate برای دستگاه را به اپلیکیشن اعطا نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="de.vogella.android.alarm"
  android:versionCode="1"
  android:versionName="1.0" >
  <uses-sdk android:minSdkVersion="15" />
  <uses-permission android:name="android.permission.VIBRATE" >
  </uses-permission>
  <application
    android:icon="@drawable/icon"
    android:label="@string/app_name" >
    <activity
      android:name=".AlarmActivity"
      android:label="@string/app_name" >
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
    <receiver android:name="MyBroadcastReceiver" >
    </receiver>
  </application>
</manifest>
```

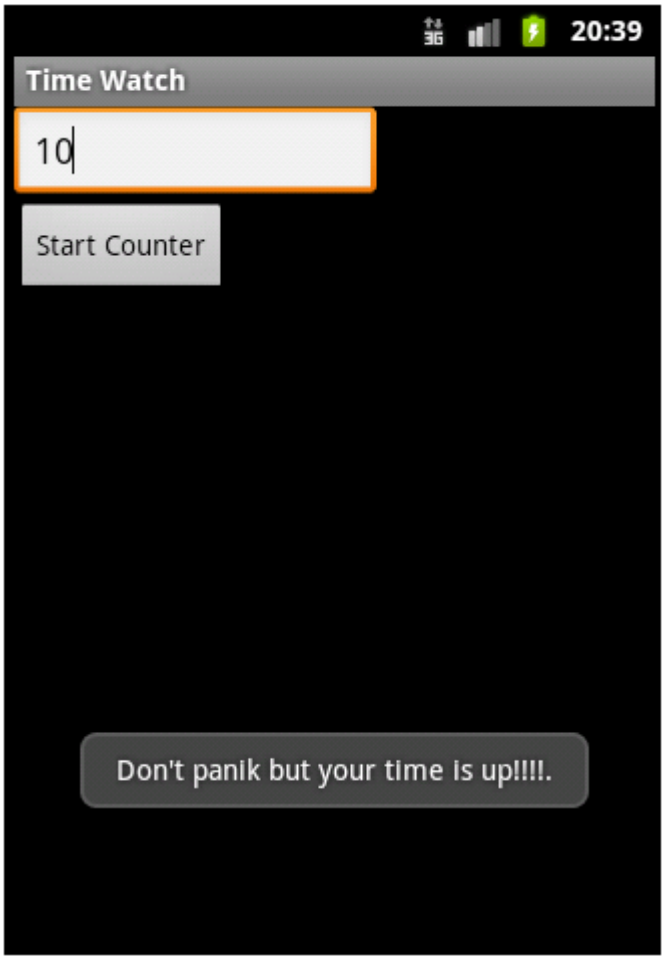
بدنه ی کلاس AlarmActivity خود را به صورت زیر ویرایش نمایید. این activity داخل خود یک آبجکت intent تعریف نموده و به وسیله ی آن receiver را فراخوانی می کند، سپس اطلاعات این intent را به سرویس alarm manager معرفی/ارسال می نماید.

```
package de.vogella.android.alarm;
import android.app.Activity;
import android.app.AlarmManager;
import android.app.PendingIntent;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
```

```
public class AlarmActivity extends Activity {
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
	}
	public void startAlert(View view) {
		EditText text = (EditText) findViewById(R.id.time);
		int i = Integer.parseInt(text.getText().toString());
		Intent intent = new Intent(this, MyBroadcastReceiver.class);
		PendingIntent pendingIntent = PendingIntent.getBroadcast(
				this.getApplicationContext(), 234324243, intent, 0);
		AlarmManager alarmManager = (AlarmManager) getSystemService(ALARM_SERVICE);
		alarmManager.set(AlarmManager.RTC_WAKEUP, System.currentTimeMillis()
				+ (i * 1000), pendingIntent);
		Toast.makeText(this, "Alarm set in " + i + " seconds",
				Toast.LENGTH_LONG).show();
	}
}
```

تست اپلیکیشن

اپلیکیشن خود را روی دستگاه تست نمایید. پس از تنظیم زمان، alarm را فراخوانی کنید. اکنون با گذشت تعداد ثانیه های تعیین شده، یک پیغام Toast به نمایش در می آید. لازم به ذکر است که قابلیت vibration در شبیه ساز اندروید کار نمی کند.



10-6-تعریف broadcast receiver به صورت dynamic (در زمان اجرای برنامه و به وسیله ی کدهای جاوا)

1-10-6-Receiver هایی که به صورت dynamic اعلان شده

همان طور که قبلا گفته شد می توانید receiver را در فایل manifest اندروید ثبت نمایید که از آن تحت عنوان ثبت به صورت static یاد می شود. در این بخش ایجاد و حذف receiver در زمان

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

اجرای برنامه به وسیله ی متدهای Context.registerReceiver() و Context.unregisterReceiver() را شرح خواهیم داد.

توجه: لازم است receiver ای که به صورت dynamic تعریف کرده اید را با فراخوانی متد Context.unregisterReceiver() حذف (unregister) نمایید. در صورت عدم فراخوانی متد مذکور، سیستم اندروید پیغام خطای leaked broadcast receiver را صادر می کند. بنابراین اگر receiver را داخل بدنه ی متد onResume() کلاس activity خود اعلان کرده باشید، بایستی آن را داخل onPause() حذف (unregister) نمایید.

2-10-6-استفاده از package manager جهت غیرفعال سازی receiver های static

می توانید با استفاده از کلاس PackageManager به راحتی receiver های تعریف شده در لایه ی XML (فایل AndroidManifest.xml) را فعال/غیرفعال نمایید.

```
ComponentName receiver = new ComponentName(context, myReceiver.class);
PackageManager pm = context.getPackageManager();
pm.setComponentEnabledSetting(receiver,
    PackageManager.COMPONENT_ENABLED_STATE_ENABLED,
    PackageManager.DONT_KILL_APP);
```

3-10-6-Intent های ماندگار (Sticky broadcast intent)

یک intent معمولا پس از اینکه ارسال شده و سیستم اطلاعات درون آن را پردازش کرد، دیگر در دسترس نبوده و از حافظه پاک خواهد شد. چنانچه مایلید آبجکت مزبور پس از پردازش توسط سیستم همچنان باقی ماند و به اصطلاح ماندگار باشد، کافی است متد sendStickyBroadcast(Intent) را فراخوانی نمایید. در صورت فراخوانی این متد، intent ماندگار بوده و پس از اتمام broadcast، در حافظه باقی می ماند.

سیستم اندروید برای حفظ برخی از اطلاعات سیستمی از sticky broadcast/intent استفاده می کند. برای مثال می توان به وضعیت باتری اشاره کرد که سیستم آن را ارسال نموده و پس از ارسال تا مدتی باقی می ماند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

مثال:

```
// Register for the battery changed event
IntentFilter filter = new IntentFilter(Intent.ACTION_BATTERY_CHANGED);
/ Intent is sticky so using null as receiver works fine
// return value contains the status
Intent batteryStatus = this.registerReceiver(null, filter);
// Are we charging / charged?
int status = batteryStatus.getIntExtra(BatteryManager.EXTRA_STATUS, -1);
boolean isCharging = status == BatteryManager.BATTERY_STATUS_CHARGING
      || status == BatteryManager.BATTERY_STATUS_FULL;
boolean isFull = status == BatteryManager.BATTERY_STATUS_FULL;
// How are we charging?
int chargePlug = batteryStatus.getIntExtra(BatteryManager.EXTRA_PLUGGED, -1);
boolean usbCharge = chargePlug == BatteryManager.BATTERY_PLUGGED_USB;
boolean acCharge = chargePlug == BatteryManager.BATTERY_PLUGGED_AC;
```

می توانید داده ها را از مقدار بازگشتی متد registerReceiver(BroadcastReceiver, IntentFilter) بازیابی نمایید. این روش برای BroadcastReceiver که حامل مقدار null می باشد نیز قابل استفاده است.

در دیگر جنبه ها، این متد رفتاری مشابه متد sendBroadcast(Intent) دارد.

استفاده از sticky broadcast ها لازمه ی تنظیم مجوزهای خاصی در لایه ی XML می باشد.

بخش چهارم :

استفاده از notification manager در اندروید/پیاده سازی notification

این مبحث به نحوه ی استفاده از notification manager در اندروید می پردازد. پروژه ی آموزش حاضر در محیط برنامه نویسی Android studio نوشته شده و مبتنی بر ویرایش 5.0 سیستم عامل اندروید می باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Notification manager-11-6

1-11-6-شرح مفهوم notification manager

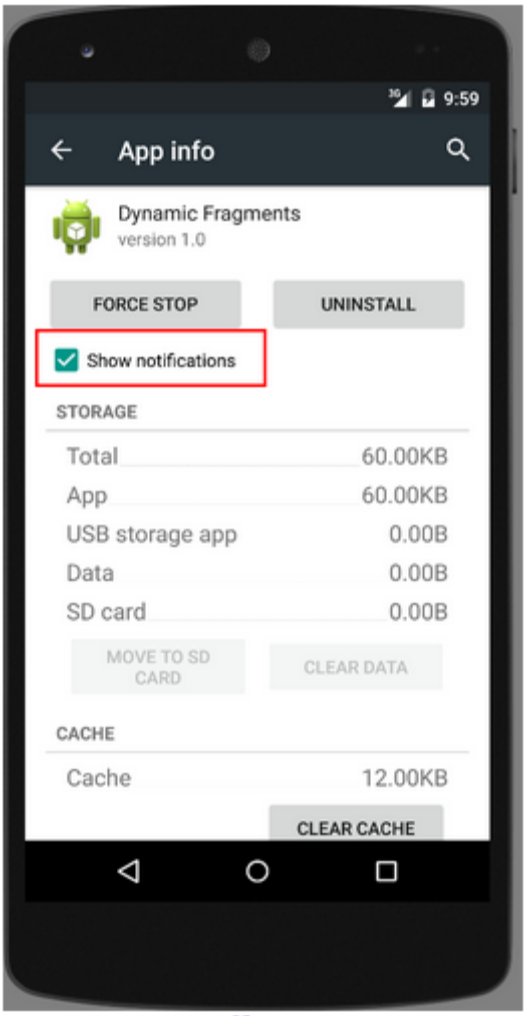
در سیستم عامل اندروید توسعه دهنده قادر است با استفاده از notification هشدارهای سیستمی یا مربوط به اپلیکیشن را به کاربر نمایش دهد.

اندروید به شما این امکان را می دهد تا notification ها را داخل titlebar اپلیکیشن خود نمایش دهید. کاربر می تواند بر روی notification تپ کرده، آن را باز نماید. سپس با انتخاب notification، یک activity و صفحه ی جدید را به اجرا در بیاورد.

از آنجایی که notification ها کیفیت تجربه ی کاربری را پایین آورده و کمی آزار دهنده هستند، کاربر این اجازه را دارد که notification های اپلیکیشن مربوطه را غیرفعال نماید. برای نیل به این هدف، کافی است کاربر به بخش تنظیمات/Settings اپلیکیشن در دستگاه اندروید مراجعه کند. پس از انتخاب گزینه Apps در Settings، کاربر اپلیکیشنی که می خواهد notification های آن را غیر فعال کند، انتخاب نموده و تیک چک باکس show notifications اپلیکیشن مرتبط را برمی دارد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



2-11-6-تنظیم و مقدار دهی notification ها

برای ساخت و نمایش notification در اندروید از کلاس Notification استفاده می شود.

نحوه ی ساخت notification در اندروید: به منظور مقداردهی و تنظیم notification از کلاس NotificationManager استفاده می شود. این کلاس با فراخوانی متد getSystemService() از Context قابل بازیابی می باشد.

```
NotificationManager notificationManager = (NotificationManager)
    getSystemService(NOTIFICATION_SERVICE);
```

Notification.Builder یک interface جهت ساخت و مقداردهی آبجکت های لازم از کلاس Notification در اختیار توسعه دهنده قرار می دهد.

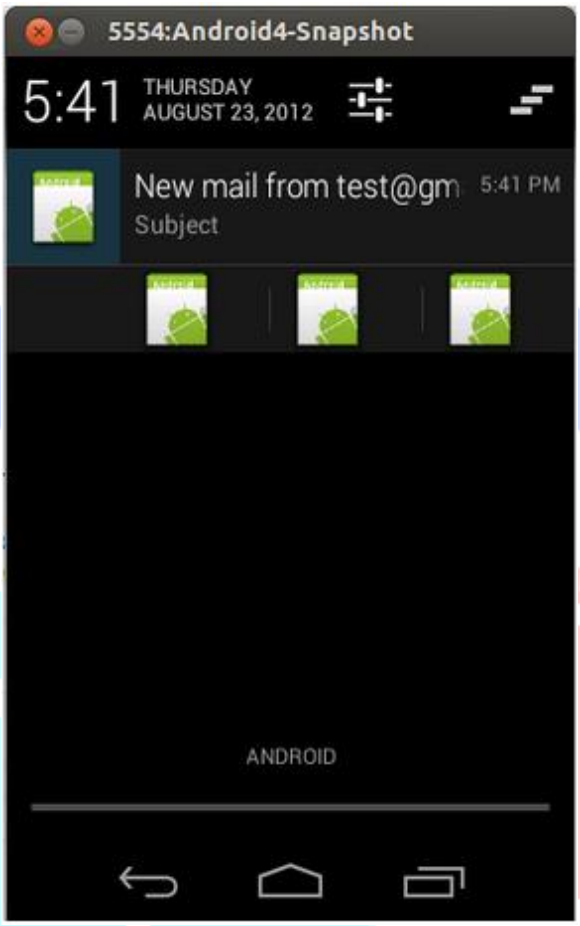
می توانید با استفاده از آبجکت PendingIntent، عملیاتی که پس از انتخاب notification باید اجرا شود را تعیین نمایید.

Notification.Builder به شما این امکان را می دهد که تا 3 کنترل دکمه با رفتار (action) دلخواه به notification اضافه نمایید.

```
// prepare intent which is triggered if the
// notification is selected
Intent intent = new Intent(this, NotificationReceiver.class);
// use System.currentTimeMillis() to have a unique ID for the pending intent
PendingIntent pIntent = PendingIntent.getActivity(this, (int) System.currentTimeMillis(), intent, 0);
// build notification
// the addAction re-use the same intent to keep the example short
Notification n  = new Notification.Builder(this)
                    .setContentTitle("New mail from " + "test@gmail.com")
                    .setContentText("Subject")
                    .setSmallIcon(R.drawable.icon)
                    .setContentIntent(pIntent)
                    .setAutoCancel(true)
                    .addAction(R.drawable.icon, "Call", pIntent)
                    .addAction(R.drawable.icon, "More", pIntent)
                    .addAction(R.drawable.icon, "And more", pIntent).build();
NotificationManager notificationManager =
    (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
notificationManager.notify(0, n);
```

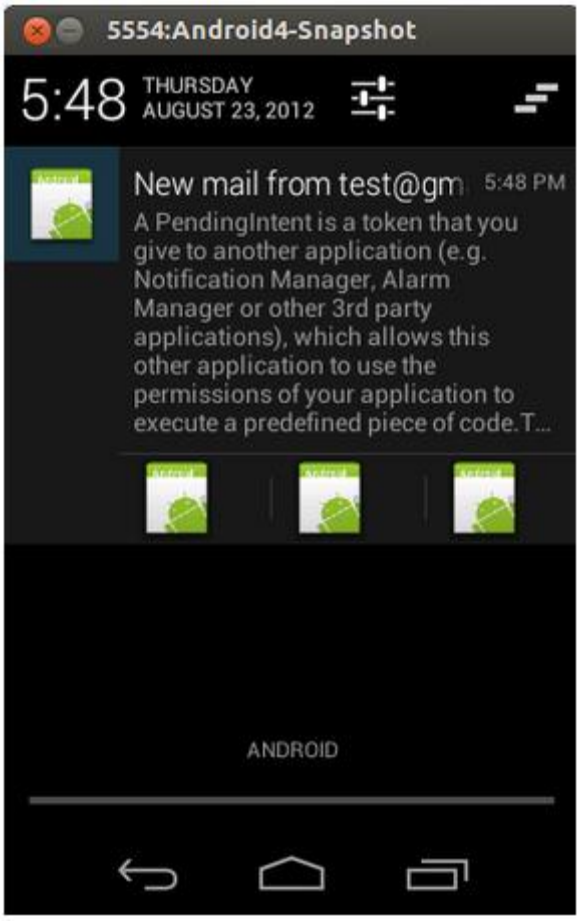

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



ویرایش 4.1 اندروید به کاربران این امکان را می دهد تا notification ها را باز کرده و جزئیات بیشتری را مشاهده نمایند. در واقع از ورژن ذکر شده به بعد علاوه بر نمای (view) معمولی notification، نمای بزرگی نیز تعبیه شده که به هنگام باز شدن notification فراخوانی می شود. برای طراحی ظاهر notification، سه style قابل گزینش می باشد: 1. big picture style 2. big text style 3. Inbox style. کد زیر استفاده از متد ()BigTextStyle را به نمایش می گذارد که تا 256 dp از کل نمایشگر را به notification اختصاص می دهد.

```
String longText = "...";
Notification noti = new Notification.Builder(this).
.....
.setStyle(new Notification.BigTextStyle().bigText(longText))
```



3-11-6-لغو کردن notification ها

کاربر می تواند تمامی notification ها را لغو کند. چنانچه توسعه دهنده notification را بر روی auto-cancel تنظیم کرده باشد، آنگاه به هنگام انتخاب کاربر، notification از نمایشگر حذف می شود.

می توانید ID یا شناسه ی منحصربفرد Notification را به عنوان پارامتر به متد cancel() پاس داده و آن را بر روی آبجکت NotificationManager فراخوانی نمایید. با فراخوانی متد cancelAll()، تمامی notification هایی که قبلا صادر کرده بودید، حذف می شوند.

تمرین: NotificationManager

یک پروژه و activity جدید به ترتیب به نام های de.vogella.android.notificationmanager و CreateNotificationActivity ایجاد نمایید. activity برای تنظیم ظاهر خود بایستی فایل main.xml را فراخوانی کند.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <Button
      android:id="@+id/button1"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:onClick="createNotification"
      android:text="Create Notification" >
   </Button>
</LinearLayout>
```

فایل result.xml را با محتوای زیر ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent" >
   <TextView
      android:id="@+id/textView1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="This is the result activity opened from the notification" >
   </TextView>
</LinearLayout>
```

اکنون کلاس activity دیگری به نام NotificationReceiverActivity ایجاد کرده و بدنه ی آن را به صورت زیر تنظیم نمایید. به یاد داشته باشید که activity مربوطه را حتما داخل فایل تنظیمات AndroidManfest.mf تعریف کنید.

```
package de.vogella.android.notificationmanager;
import android.app.Activity;
import android.os.Bundle;
public class NotificationReceiverActivity extends Activity {
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.result);
```

```
    }
}
```

پیاده سازی کلاس CreateNotificationActivity را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.notificationmanager;
import android.app.Activity;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
public class CreateNotificationActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
    public void createNotification(View view) {
        // Prepare intent which is triggered if the
        // notification is selected
        Intent intent = new Intent(this, NotificationReceiverActivity.class);
        PendingIntent pIntent = PendingIntent.getActivity(this, (int) System.currentTimeMillis(), intent, 0);
        // Build notification
        // Actions are just fake
        Notification noti = new Notification.Builder(this)
                .setContentTitle("New mail from " + "test@gmail.com")
                .setContentText("Subject").setSmallIcon(R.drawable.icon)
                .setContentIntent(pIntent)
                .addAction(R.drawable.icon, "Call", pIntent)
                .addAction(R.drawable.icon, "More", pIntent)
                .addAction(R.drawable.icon, "And more", pIntent).build();
        NotificationManager notificationManager = (NotificationManager)
getSystemService(NOTIFICATION_SERVICE);
        // hide the notification after its selected
        noti.flags |= Notification.FLAG_AUTO_CANCEL;
        notificationManager.notify(0, noti);
    }
}
```

اپلیکیشن خود را اجرا کرده و کنترل دکمه را فشار دهید. یک notification جدید ایجاد شده که با انتخاب آن، activity و پنجره ی جدید در نمایشگر باز می شود.

بخش پنجم :

Backup گیری از داده ها در اندروید/تهیه ی نسخه ی پشتیبان از داده ها در اندروید

آموزش حاضر به شرح توابع کتابخانه ای Android data backup API پرداخته و نحوه ی استفاده از آن برای بازگردانی داده های مربوط به تنظیمات و Configuration اپلیکیشن را توضیح می دهد.

Backup-12-6 در اندروید

1-12-6-هدف از تهیه ی نسخه ی پشتیبان از داده ها

سرویس backup گیری اندروید به توسعه دهنده این امکان را می دهد تا داده های ماندگار اپلیکیشن را در حافظه ی راه دور cloud ذخیره نماید. بدین وسیله زمانی که کاربر دستگاه اندروید خود را عوض می کند یا اپلیکیشنی را مجددا نصب می نماید و یا factory reset گوشی را فعال می کند، داده های اپلیکیشن و اطلاعات مربوط به تنظیمات همگی به راحتی قابل بازگردانی خواهد بود.

البته تمامی دستگاه های اندروید لزوما قابلیت data backup را ندارند. همچنین حافظه ی cloud ممکن است از جانب تولید کننده ی دستگاه مورد نظر به صورت اختصاصی تنظیم شده باشد. از این رو تیم توسعه ی محیط اندروید به هیچ وجه امنیت و سلامت کامل داده ها را به هنگام backup و بازگردانی تضمین نمی کند.

چنانچه سرویس در دستگاه مد نظر موجود نباشد، متعاقبا سرویس backup فراخوانی نمی شود. از این رو بهتر است ابتدا سرویس را پیاده سازی نموده و سپس آن را بر روی دستگاه دلخواه اجرا نمایید.

برای استفاده از سرویس backup گیری، ابتدا لازم است اسم پکیج اپلیکیشن خود را در آدرس اینترنتی https://developer.android.com/google/backup/signup.html ثبت نمایید. پس از

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

معرفی کردن اسم پکیج اپلیکیشن خود، این سایت یک کلید backup در اختیار شما قرار می دهد که برای تهیه ی نسخه ی پشتیبان از داده های اپلیکیشن مورد نظر در آینده از آن استفاده خواهید نمود.

2-12-6-تهیه ی نسخه ی پشتیبان از shared preferences (داده های کوچک همچون اطلاعات مربوط به تنظیمات اپلیکیشن) و فایل ها

برای ذخیره ی دائمی داده ها و بازگردانی آن ها، ابتدا یک کلاس تعریف کنید که توابع و فیلدهای کلاس BackupAgent را به ارث می برد.

ساده ترین راه برای پیاده سازی backup استفاده از SharedPreferencesBackupHelper ویژه ی backup گیری از داده های کوچک همچون اطلاعات مربوط به تنظیمات اپلیکیشن و استفاده از FileBackupHelper ویژه ی تهیه ی نسخه ی پشتیبان از داده های مستقر در حافظه ی داخلی سیستم می باشد. دو کلاس کمکی (helper) نام برده به صورت خودکار تمامی فایل های ثبت شده را ذخیره کرده و بازگردانی می نماید.

3-12-6-تهیه ی Backup کلی

به منظور تهیه ی backup پیچیده و کلی تر، می توانید خود کلاس BackupAgent و متدهای save/ restore آن را به صورت مستقیم پیاده سازی نمایید. برای مطالعه ی جزئیات بیشتر می توانید به آدرس http://developer.android.com/guide/topics/data/backup.html مراجعه نمایید.

تمرین: پیاده سازی backup گیری از داده های مربوط به تنظیمات اپلیکیشن

پیاده سازی پروژه

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

اپلیکیشنی که کد آن را در زیر مشاهده می کنید از پکیج com.vogella.android.databackup در بالای فایل XML استفاده می کند. این پکیج در گوگل سرویس ثبت شده است.

Backup agenda و registration key در فایل تنظیمات اپلیکیشن AndroidManifest.xml به صورت زیر درج می شود.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.vogella.android.databackup"
   android:versionCode="1"
   android:versionName="1.0" >
   <uses-sdk
      android:minSdkVersion="17"
      android:targetSdkVersion="17" />
   <application
      android:allowBackup="true"
      android:backupAgent="MyBackupAgent"
      android:icon="@drawable/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >
      <activity
         android:name="com.vogella.android.databackup.MainActivity"
         android:label="@string/app_name" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
      <meta-data
         android:name="com.google.android.backup.api_key"
        android:value="AEdPqrEAAAAI4SfiyaQncNamIUH0NboU3tzOjXGztXLv2LZkEw" />
   </application>
</manifest>
```

Backup agent مربوطه را می توان به صورت زیر پیاده سازی کرد.

```
package com.vogella.android.databackup;
import android.app.backup.BackupAgentHelper;
import android.app.backup.SharedPreferencesBackupHelper;
public class MyBackupAgent extends BackupAgentHelper {
```

```
    // The name of the SharedPreferences file
    static final String PREFS = "myprefs";
    // A key to uniquely identify the set of backup data
    static final String PREFS_BACKUP_KEY = "myprefs";
    @Override
    public void onCreate() {
        SharedPreferencesBackupHelper helper = new SharedPreferencesBackupHelper(
                this, PREFS);
        addHelper(PREFS_BACKUP_KEY, helper);
    }
}
```

4-12-6-راه اندازی و فعال کردن پروسه های backup و restore (بازگردانی داده ها)

Backup manager پیاده سازی شده، به صورت اتوماتیک و بدون دخالت و تعامل کاربر shared preferences و تنظیمات دلخواه ثبت شده کاربر را به طور دائمی ذخیره نموده و بازگردانی می نماید. کافی است shared preferences را تحت فایلی که اپلیکیشن خود را با نام آن ثبت کردید، تغییر داده و ذخیره نمایید.

چنانچه مایلید عملیات ذخیره و بازگردانی داده ها را از طریق پنجره ی فرمان adb shell command تست نمایید، در آن صورت می توانید دستورات زیر را به کار ببرید.

```
# Trigger a backup, usage:
# adb shell bmgr backup <package>
# schedule backup
adb shell bmgr backup com.vogella.android.databackup
# ensure scheduled backup run
adb shell bmgr run
# to restore you backup use bmgr restore
adb shell bmgr restore com.vogella.android.databackup
```

بخش ششم :

استفاده ی بهینه از حافظه و افزایش کارایی در اندروید

آموزش حاضر به شرح نحوه ی استفاده ی بهینه از حافظه و افزایش کارایی اپلیکیشن می پردازد. کدهای این بخش داخل محیط برنامه نویسی Android Studio تست و اجرا شده است.

13-6-رهنمودهایی جهت طراحی و ارائه ی اپلیکیشن های کارامد

13-6-1-چرا باید در استفاده از منابع اندروید مراقب بود؟

دستگاه های اندروید نسبت به کامپیوترهای رومیزی یا لپ تاپ قدرت و توانایی کمتری دارند. به همین دلیل بایستی در استفاده از حافظه دقت بیشتری داشته باشید. به ویژه از ویرایش 5.0 اندروید قبل تر، توسعه دهنده باید مراقب باشد تا حد امکان garbage collector دستگاه مجازی جاوا (JVM) را درگیر نکند چرا که این امر سبب freeze و هنگ کردن runtime اندروید به مدت 200 میلی ثانیه می شود. در چنین شرایطی چنانچه کاربر، برای مثال، با نوار اسکرول در حال پیمایش به پایین لیست باشد، با توجه به تاخیر قابل توجهی که garbage collector سبب می شود، تجربه ی کاربری به شدت پایین می آید.

13-6-2-اجتناب از تخصیص و ایجاد آبجکت غیر ضروری

سعی کنید تا حد امکان از ایجاد آبجکت های غیر ضروری، به خصوص در جاهای هزینه بر خودداری نمایید. توصیه می کنیم آبجکت های موجود را بازیافت نموده و مجددا از آن ها استفاده کنید. ایجاد آبجکت های اضافی و غیر ضروری سبب راه اندازی مکرر garbage collector می شود. می توان با اجتناب از ساخت آبجکت های غیر ضروری به راحتی از این رخداد جلوگیری نمود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

برای مثال بهتر است از ایجاد آبجکت در حلقه ها یا در بدنه ی متد onDraw() داخل آبجکت view اختصاصی خود خودداری نمایید.

3-13-6-استفاده از data structure های کارامد

اندروید چندین نمونه با پیاده سازی های مختلف از کلاس های Sparse*Array ارائه می دهد. به کد زیر توجه کنید.

```
Map<Integer, String> map = new HashMap<Integer, String>();
```

استفاده از این کد سبب ساخت آبجکت های غیر ضروری از نوع Integer می شود.

محیط اندروید جهت نگاشت مقادیر به دیگر آبجکت ها، data structure های کارمدتری را در اختیار توسعه دهنده قرار می دهد. در صورت امکان توصیه ی ما بر این است که از این ساختارها استفاده نمایید چرا که این ساختارها از ساخت آبجکت اضافی جلویگری می کنند. از جمله می توان به HashMap اشاره کرد.

همان طور که در بالا نیز گفته شد، ساخت آبجکت عملیات سنگین و هزینه بری بوده و تا حد امکان می بایست از رخداد غیر ضروری آن جلوگیری نمود. زیرا به دنبال کاهش تعداد دفعات ایجاد آبجکت، garbage collector نیز دفعات کمتری جهت مدیریت حافظه فراخوانده می شود.

جدول زیر مثال هایی از کاربرد SparseArrays را نمایش می دهد.

Data structure هایی که از نظر مصرف حافظه بهینه هستند	
Data structure	شرح
SparseArray<E>	مقادیری از نوع عدد صحیح را به آبجکت ها نگاشت کرده و از ساخت آبجکت های integer جلوگیری می کند.
SparseBooleanArray	مقادیر از نوع integer را به بولی ها (boolean) نگاشت می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

SparseIntArray	مقادیر integer را به integer ها نگاشت می کند.

به منظور بهینه سازی مثال کاربردی فوق، بهتر است از data structure زیر استفاده نمایید.

```
SparseArray<String> map = new SparseArray<String>();
map.put(1, "Hello");
```

14-6-مدیریت bitmap

Bitmap ها (عکس های پیکسلی) ، چنانچه در اندازه ی کامل و حقیقی بارگذاری شوند، مقدار زیادی از حافظه را اشغال خواهند کرد. توصیه می شود bitmap ها را فقط در اندازه ی مورد نیاز بارگذاری کنید.

فرض کنید اپلیکیشنی طراحی کردید که تصویری در ابعاد 100x100 dp را نمایش می دهد، برای این اپلیکیشن می بایست تصویر مربوطه را دقیقا در همین اندازه داخل حافظه بارگذاری نمایید.

یک روش معمول این است که با ارسال یک flag به BitmapFactory، bitmap دلخواه را بدون اینکه در حافظه بارگذاری شود، اول اندازه گیری نمایید.

```
// instruct BitmapFactory to only the bounds and type of the image
BitmapFactory.Options options = new BitmapFactory.Options();
options.inJustDecodeBounds = true;
BitmapFactory.decodeResource(getResources(), R.id.myimage, options);
// get width and height
int imageHeight = options.outHeight;
int imageWidth = options.outWidth;
// type of the image
String imageType = options.outMimeType;
```

پس از آن می توانید نسخه ی درست اندازه بندی شده ی تصویر مورد نظر را در حافظه بارگذاری نمایید. اندروید تصاویر را به توان 2 می رساند. می توانید با استفاده از متد زیر scale factor (ضریب مقیاس بندی) را بر مبنای 2 محاسبه نمایید.

```
public static Bitmap decodeBitmapWithGiveSizeFromResource(Resources res, int resId,
      int reqWidth, int reqHeight) {
   // First decode with inJustDecodeBounds=true to check dimensions
   final BitmapFactory.Options options = new BitmapFactory.Options();
   options.inJustDecodeBounds = true;
   BitmapFactory.decodeResource(res, resId, options);
   // Calculate inSampleSize
   options.inSampleSize = calculateInSampleSize(options, reqWidth, reqHeight);
   // Decode bitmap with inSampleSize set
   options.inJustDecodeBounds = false;
   return BitmapFactory.decodeResource(res, resId, options);}
public static int calculateInSampleSize(
         BitmapFactory.Options options, int reqWidth, int reqHeight) {
   // Raw height and width of image
   final int height = options.outHeight;
   final int width = options.outWidth;
   int inSampleSize = 1;
   if (height > reqHeight || width > reqWidth) {
      final int halfHeight = height / 2;
      final int halfWidth = width / 2;
      // Calculate the largest inSampleSize value that is a power of 2 and keeps both
      // height and width larger than the requested height and width.
      while ((halfHeight / inSampleSize) > reqHeight
            && (halfWidth / inSampleSize) > reqWidth) {
         inSampleSize *= 2;}}
   return inSampleSize;}
```

با استفاده از این متد می توان تصویر مورد نظر را همان طور که در مثال زیر نمایش داده شده، به view تخصیص داد.

```
viewWidth = imageView.getWidth();
viewHeight = imageView.getHeight();

imageView.
imageView.setImageBitmap(
   decodeSampledBitmapFromResource(getResources(), R.id.myimage, viewWidth, viewHeight));
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

6-15-استفاده از Cache

1-15-6-استفاده از حافظه ی نهان

همان طور که می دانید فرایند ساخت آبجکت بسیار سنگین و هزینه بر است. Cache این امکان را فراهم می کند تا آبجکت هایی که قبلا در حافظه موجود هستند را بازیافت نموده (مجددا مورد استفاده قرار داد) و بدین وسیله از تخصیص یافتن مجدد منابع به ساخت آبجکت (جدید) خودداری گردد. به عبارتی، زمانی که شما آبجکتی را در حافظه بارگذاری می کنید، می توانید این حافظه را به عنوان cache و حافظه ی موقت برای میزبانی آبجکت ساخته شده در نظر بگیرید. به عنوان مثال زمانی را در نظر بگیرید که تعدادی فایل تصویری را از اینترنت بارگیری می کنید. در این لحظه، جهت نمایش فایل های ذکر شده در یک لیست، می بایست آن ها را به طور موقت در حافظه نگه دارید. امر مانع از این می شود که عکس ها بارها از اینترنت دانلود شوند.

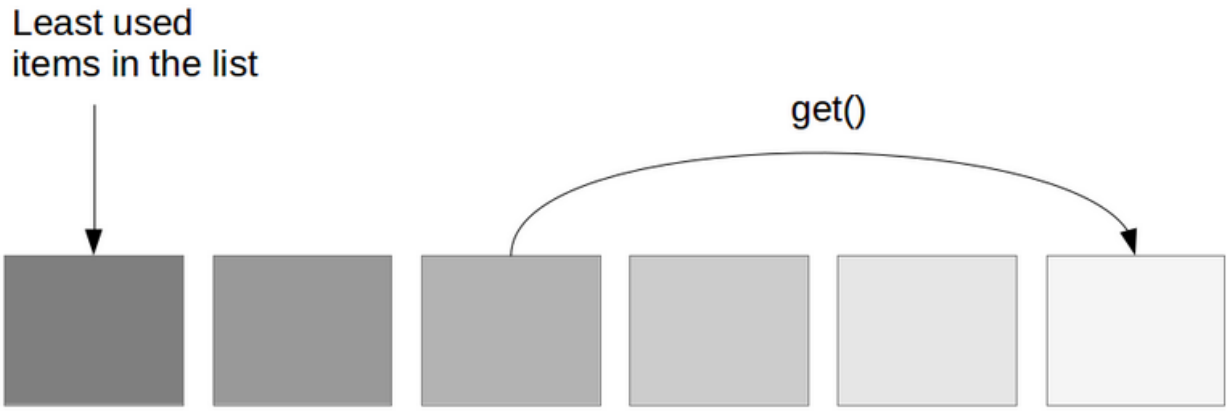
شما به ناگذیر بایستی برخی از آبجکت های مستقر در حافظه را بازیافت نمایید چرا که در غیر این صورت با کمبود حافظه مواجه خواهید شد. یک روش پیشنهادی این است که آبجکت هایی که طولانی مدت در اپلیکیشن مورد استفاده قرار نگرفته اند را بازیافت نمایید.

محیط (platform) اندروید از API 12 (ورژن 12 کتابخانه های اندروید) کلاسی به نام LruCache را ارائه می نماید (لازم به ذکر است که این قابلیت در کتابخانه های پشتیبانی از ورژن های قبلی اندوریدsupport-v4 library نیز قابل دسترسی می باشد). کلاس مزبور امکان پیاده سازی یک cache جهت نگهداری آبجکت هایی که مدت طولانی مورد استفاده قرار نگرفته اند را فراهم می سازد. به عبارت دیگر این کلاس یک LRU cache پیاده سازی می کند. در این cache تمامی اعضایی که مدت ها از آخرین بار استفاده از آن ها می گذرد، قرار می گیرند (آبجکت هایی که کمترین استفاده ی اخیر را داشته اند در این حافظه ی موقت جای می گیرند). این cache ظرفیت مشخصی دارد و هنگامی که این ظرفیت پر می شود، تمامی آیتم هایی که طولانی مدت مورد استفاده قرار نگرفته اند بر حسب نیاز از این cache حذف می شوند. این قابلیت تصویر زیر نمایش داده شده است.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

LRU Cache



Calling get() for an item, moves it to the top of the cache

مثال زیر یک نمونه از پیاده سازی کلاس LruCache را برای ذخیره ی موقتی فایل های تصویری فراهم می سازد.

```
public class ImageCache extends LruCache<String, Bitmap> {
  public ImageCache( int maxSize ) {
    super( maxSize );
  }
  @Override
  protected int sizeOf( String key, Bitmap value ) {
    return value.getByteCount();
  }
  @Override
  protected void entryRemoved( boolean evicted, String key, Bitmap oldValue, Bitmap newValue ) {
    oldValue.recycle();
  }
}
```

همان طور که در نمونه کد زیر مشاهده می کنید، استفاده از آن بسیار ساده می باشد.

```
LruCache<String, Bitmap> bitmapCache = new LruCache<String, Bitmap>();
```

جهت اطلاع از ظرفیت حقیقی cache، کافی است اندازه و ظرفیت کل حافظه ی قابل استفاده در دستگاه میزبان را بررسی نمایید. برای این منظور می توانید از کلاس MemoryClass استفاده نمایید. کد زیر استفاده از آن را به نمایش می گذارد.

```
int memClass = ( ( ActivityManager) activity.getSystemService( Context.ACTIVITY_SERVICE )
).getMemoryClass();
int cacheSize = 1024 * 1024 * memClass / 8;
LruCache cache = new LruCache<String, Bitmap>( cacheSize );
```

2-15-6-پاک سازی cache

از API 14 این امکان برای شما وجود دارد که متد ()onTrimMemory را در کامپوننت های نرم افزاری اندورید بازنویسی (override) نمایید. این متد را اندروید فراخوانی نموده و به شما این اجازه را می دهد تا زمانی که سیستم منابعی را جهت پردازش های پیش زمینه ای (foreground process) لازم داشت، محتوای حافظه را پاک سازی و آزاد نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فصل هفتم

Dependency Injection

، آزمایش ابزارهای کمکی

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش اول :

تست بخش های مجزای اپلیکیشن های اندرویدی به وسیله ی ابزار unit testing / آزمایش کامپوننت های اپلیکیشن اندروید با استفاده از ابزار jUnit

آموزش حاضر به شرح unit testing (اجرای تست نرم افزاری بر روی بخش های مختلف پروژه به صورت مجزا) با استفاده از ابزار JUnit 4.x می پردازد. این آموزش نحوه ی ایجاد تست های jUnit را شرح داده و سپس استفاده از محیط برنامه نویسی Eclipse را در راستای نوشتن تست های نرم افزاری لازم جهت کسب اطمینان از عملکرد صحیح بخش های مختلف اپلیکیشن را توضیح می دهد.

7-1-هدف از نوشتن تست های نرم افزاری چیست؟

7-1-1-شرح مفهوم تست های نرم افزاری

تست اپلیکیشن (software test) یک قطعه ی نرم افزاری (کد) هست که خود یک قطعه ی نرم افزاری دیگر را (جهت آزمایش و اطمینان از کارکرد صحیح و اینکه آیا قطعه ی نوشته شده هدف مورد نظر را برآورده می سازد) اجرا می کند. این تست نرم افزاری بررسی می کند آیا قطعه کد نوشته شده منجر به وضعیت مورد انتظار می شود (که به آن state testing گویند) و یا مراحل لازم (سلسه رخداد ها) به ترتیب رخ می دهند یا خیر (که به آن behavior testing گویند).

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

7-1-2-چرا استفاده از تست ها توصیه می شود؟

Unit test به توسعه دهنده کمک می کند تا منطق یک تکه از برنامه را آزمایش کرده و بدین وسیله مطمئن شود رفتار مورد انتظار را از خود نشان می دهد/هدف و نیاز مدنظر را برآورده می سازد یا خیر.

در واقع با نوشتن و اجرای unit test بر روی بخش های نرم افزاری به صورت مجزا این امکان فراهم می شود تا کاستی ها و خطاهایی که ممکن است با اضافه شدن رفتار / اعمال تغییرات جدید به برنامه در آن رخنه کرده باشند، توسط توسعه دهنده به سرعت کشف شده و برطرف گردد. نوشتن تست برای بخش های متعددی از اپلیکیشن به برنامه نویس کمک می کند تا بعده ها بدون اینکه مجبور به نوشتن تست های دستی (manual test) شود، بتواند قابلیت های نوین زیادی با عملکرد صحیح به اپلیکیشن خود اضافه نماید.

7-1-3-فریم ورک های تست گیری (testing frameworks) برای Java

فریم ورک های تست گیری متعددی برای Java نوشته شده و در دسترس می باشد. از جمله ی پرکاربرد و محبوب ترین آن ها می توان به JUnit و TestNG اشاره کرد.

مقاله ی حاضر بر روی نحوه ی استفاده از ابزار JUnit متمرکز می شود.

7-2-واژه ها و مفاهیم مرتبط با تست گیری

7-2-1-کد (یا اپلیکیشن) مورد تست / Code Under Test

کدی که تست بر روی آن اجرا می شود در اصطلاح Code Under Test (کد مورد آزمایش) خوانده می شود. حال اگر واحد مورد تست شما یک آپلیکیشن باشد، در آن صورت application under test اطلاق خواهد شد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-2-7-Test fixture (مقادیر ثابت تست)

Test fixture عبارت است از state ثابت (اطلاعات مربوط به وضعیت) چند آبجکت که به عنوان مقادیر پایه (baseline) برای اجرای تست بر روی بخش های مجزا نرم افزار مورد استفاده قرار می گیرد. مقصود اصلی استفاده از test fixture این است که محیط ثابت و مشخصی برای تست وجود داشته باشد به طوری که نتایج / خروجی حاصل تکرارپذیر باشد. می توان از آن به عنوان پیش شرط های تست نیز نام برد. به عبارت دیگر، هرچه که بایستی از قبل آماده باشد تا یک تست اجرا شده و نتیجه ی مورد نظر را به عنوان خروجی انتظار داشت، text fixture خوانده می شود.

به عنوان مثال، text fixture می تواند یک رشته ی ثابت باشد که به عنوان پارامتر ورودی ثابت به یک متد پاس داده می شود. در نهایت تست مشخص می کند آیا متد مورد آزمایش نتایج مورد انتظار را تولید می کند / عملکرد صحیح را ارائه می دهد یا خیر.

3-2-7-شرح مفهوم Unit test و تست بخش های مختلف نرم افزار (کد) به صورت مجزا

Unit test یک قطعه کد است که توسعه دهنده برای تست قابلیتی (کسب اطمینان از عملکرد صحیح کد) از برنامه آن را نوشته و انتظار دارد که نتیجه ی (رفتار یا وضعیت) دلخواه را تولید کند.

آن درصدی از کل کد که مورد آزمایش قرار می گیرد، در اصطلاح test coverage اطلاق می گردد.

Unit test معمولا بخش یا قطعات (مستقل) کوچکی از برنامه را در سطح کلاس یا متد مورد آزمایش قرار می دهند. لازم به ذکر است که dependency ها و کتابخانه های خارجی را می بایست از unit test ها حذف کرد. این کار را می توان با جایگزین کردن این کتابخانه ها با پیاده سازی و بدنه ی تست یا آبجکت ساختگی که از پیش توسط framework ساخته شده، انجام داد.

گفتنی است که Unit test ها کارایی چندانی برای تست رابط های کاربری پیچیده و تعامل داده بین کامپوننت های نرم افزاری ندارند. برای این منظور بهتر است integration test طراحی نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Integration test–7–2–4 (تست اپلیکیشن در context و بستر خودش)

Integration test عبارت است از آزمایش اپلیکیشن در context و بستر کلی خودش به طوری که برای تست نیازمند بخش های دیگر اپلیکیشن باشد (به عبارت دیگر آن قطعه هایی از کد که بر روی بخش های دیگر تاثیر جانبی/side effect دارند). Integration test بر روی رابطه و بخش های مشترک چند کامپوننت نرم افزاری تمرکز می کند. این نوع تست همچنین اطمینان حاصل می کنند که کل سیستم به درستی و مورد انتظار رفتار می کند و در این حین نیاز برای نوشتن و اجرای تست های دقیق دستی را کاهش می دهند.

این نوع تست ها به شما امکان می دهند تا پیشنهادات و بازخورد کاربران از اپلیکیشن را در مجموعه تست هایی (test suite) که برای آزمایش و کسب اطمینان از عملکرد صحیح اپلیکیشن نوشته اید، لحاظ نمایید. از این جهت، خروجی تست مربوطه مشابه تعامل مورد انتظار بین کاربر و اپلیکیشن خواهد بود.

Performance test–7–2–5 (تست های بررسی کارایی)

Performance test یا تست های ارزیابی کارایی به طور مکرر برای بررسی میزان کارایی و مستند سازی قدرت کامپوننت های نرم افزاری مورد استفاده قرار می گیرند. تست ذکر شده را توسعه دهنده برای این می نویسد که میزان کارایی کد مورد آزمایش را در شرایطی که بار کاری سنگین هست بررسی نموده و از بالا بودن بازده در هر شرایطی اطمینان حاصل کند.

Behavior testing و State testing–7–2–6

تستی که بررسی می کند آیا متدهایی با پارامترهای ورودی مورد نظر فراخوانی شده اند یا خیر، interaction test (behavior test) خوانده می شوند. Behavior test نتیجه ی متد فراخوانی شده را اعتبار سنجی نمی کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

State test (تست وضعیت) ، بر خلاف behavior test، در بررسی و آزمایش رفتار اپلیکیشن مورد تست خلاصه می شود.

چنانچه الگوریتم یا قابلیت های سیستمی را مورد آزمایش قرار داده اید، در آن صورت تست شما بایستی بر روی state متمرکز شود، نه تعامل بین کامپوننت ها.

در این تست ها غالبا از آبجکت های ساختگی (mock) و stub (کد جایگزین) کلاس های مرتبط برای استخراج، حذف تعامل یا وابستگی با دیگر کلاس ها استفاده می شود. پس از آن شما می توانید (اطلاعات مربوط به) وضعیت یا رفتار را بر اساس نیاز خود تست نمایید.

7-3-سازماندهی تست

7-3-1-تست در کجا بایستی نوشته شود

Unit test های معمولی اغلب در پروژه ی مجزا یا پوشه ی اصلی (source folder) جداگانه جایگذاری می شوند. این کار سبب می شود کد تست شده از کد اصلی جدا نگه داشته شود.

7-3-2-کدام بخش نرم افزار بایستی تست شود

اینکه دقیقا می بایست چه بخش هایی تست شود، یک موضوع بسیار بحث برانگیز است. برخی از توسعه دهندگان بر این عقیده هستند که تمامی دستورات داخل کد می بایست مورد بررسی قرار گرفته و تست شود.

به هر حال شما بایستی برای بخش های پیچیده و حساس نرم افزار خود تست بنویسید. اگر ویژگی و قابلیت های جدید به اپلیکیشن خود اضافه نمایید، آنگاه یک مجموعه تست قدرتمند و کارامد می تواند اپلیکیشن شما را در برابر ضعف و خطاهایی که ممکن است بعدها به داخل اپلیکیشن و کد جاری راه پیدا کنند محافظت خواهد کرد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در طراحی تست نرم افزاری می توان بخش های جزئی از کد را نادیده گرفت. برای مثال، نوشتن تست برای متدهای getter و setter که صرفا مقادیری را به فیلدهایی تخصیص می دهند، تقریبا بیهوده است. در واقع نوشتن تست برای تمامی دستورات داخل کد مشخصا بسیار زمان بر بوده و تقریبا امری بی فایده می باشد چرا که شما اپلیکیشن را در بستر JVM تست می کنید. JVM به خودی خود test case هایی از پیش تعریف شده و مشخص دارد.

در صورت طراحی تست برای code base هایی که قبلا تستی برای آن ها نوشته نشده، بهتر است ابتدا برای آن بخش هایی از کد تست تهیه طراحی نمایید که دفعات رخداد خطا در گذشته در آن ها زیاد باشد. سپس می توانید بر روی بخش های اساسی کد (اپلیکیشن) متمرکز شوید.

7-4-استفاده از JUnit

7-4-1- JUnit framework(فریم ورک انجام unit test بر روی پروژه های جاوا)

JUnit 4.x در واقع یک فریم ورک اجرای unit test بر روی پروژه های جاوایی است که با استفاده از annotation ها، متدهایی که رفتارها و قابلیت های تست را به صورت جداگانه آزمایش می کنند با علائم یا دستورهای خاص نشانه گذاری و معرفی می نماید. برای کسب اطلاعات بیشتر در زمینه ی این فریم ورک تست گیری می توانید به آدرس http://junit.org/ و https://github.com/junit-team/junit مراجعه نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

7-4-2-نحوه ی طراحی یک تست در فریم ورک تست گیری JUnit؟

JUnit test در حقیقت یک متد است که داخل کلاس تعریف شده و منحصرا برای منظور تست مورد استفاده قرار می گیرد. کلاسی که میزبان متد مذکور می باشد در اصطلاح test class نامیده می شود. به منظور نوشتن یک تست بر مبنای JUnit 4، متد مورد نظر را با دستور @org.junit.Test داخل کلاس میزبان نشانه گذاری (annotate) می کنید.

این متد کد مورد آزمایش را اجرا می کند. می توانید از متد assert که خود JUnit ارائه می دهد استفاده نمایید و یا از متدی که فریم ورک assert دیگری فراهم می نماید، بهره بگیرید. متد assert در اصل برای بررسی نتیجه ی واقعی با نتایج مورد انتظار مورد استفاده قرار می گیرد. متدهایی که به این صورت استفاده می شوند در اصطلاح تست گیری، assert ها یا دستورات assert خوانده می شوند.

لازم است در دستورات assert پیغام های معنی دار قرار دهید چرا که در صورت مواجه شدن با خطا کاربر سریع تر می تواند مشکل را شناسایی کرده و برطرف نماید. این ویژگی می تواند به ویژه برای فردی که کد را می خواند اما در نوشتن آن شرکت نداشته، مفید باشد.

7-4-3-نمونه ای از JUnit test

کد زیر یک نمونه از پیاده سازی تست JUnit را نمایش می دهد. این تست فرض را بر این می گذارد که کلاس MyClass وجود داشته و متدی به نام multiply(int, init) را دربردارد.

```
import static org.junit.Assert.assertEquals;
import org.junit.Test;
public class MyTests {
	@Test
	public void multiplicationOfZeroIntegersShouldReturnZero() {
		MyClass tester = new MyClass(); // MyClass is tested
		// assert statements
		assertEquals("10 x 0 must be 0", 0, tester.multiply(10, 0));
		assertEquals("0 x 10 must be 0", 0, tester.multiply(0, 10));
		assertEquals("0 x 0 must be 0", 0, tester.multiply(0, 0));
	}
}
```

4-4-7-قرارداد/روش های نام گذاری تست های مبتنی بر JUnit

قراردادهای نام گذاری متعددی برای تخصیص اسم به تست های طراحی شده بر اساس فریم ورک JUnit وجود دارد. یک راه حل پر طرفدار افزودن پسوند "-test" به اسم کلاس های تست گیری (test case) و جایگذاری آن ها در پکیج جدید "test" می باشد.

به طور کلی، اسم تست باید درباره ی کاربرد و مورد استفاده ی آن توضیح دهد.در صورتی که اسم تخصیص داده شده به تست، مورد کاربرد آن را روشن بیان نماید، دیگر نیازی به خواندن پیاده سازی و محتوای کد نخواهد بود.

یک قرارداد ممکن و پرطرفدار استفاده از کلمه ی "should" در اسم متد تست گیری می باشد. برای مثال، می توان به "ordersShouldBeCreated" یا "menuShouldGetActive" اشاره کرد. با این نوع انتخاب اسم برای متد، مشخص می شود که متد مورد نظر دقیقا چه رفتاری را تست کرده و انتظار چه نتیجه ای را دارد.

4-4-7-قراردادهای نام گذاری JUnit برای Maven

در صورت استفاده از سیستم کامپایل Maven (build system)، بهتر است از "Test" بجای "Tests" به عنوان پسوند استفاده نمایید. سیستم کامپایل Maven (به واسطه ی افزونه ی surefire خود) به صورت اتوماتیک و درون ساخته این کلاس ها را داخل test scope خود دربردارد.

4-4-7-مجموعه تست های JUnit (JUnit test suites)

در صورتی که چندین کلاس تست گیری برای اپلیکیشن نوشته باشید، بد نیست تمامی آن ها را در یک قالب واحد به نام test suite یا مجموعه تست بگنجانید. شایان ذکر است که اجرای test suite به مثابه ی اجرای تمامی کلاس های تست گیری، به ترتیب از قبل مشخص شده است که در مجموعه تست تعریف شده اند.

Test suite یک مجموعه ی بزرگ از تست ها است که خود می تواند test suite های دیگر را دربرگیرد.

نمونه کد زیر نحوه ی استفاده از test suite را نمایش می دهد. test suite حاضر دو کلاس تست گیری MyClassTest و MySecondClassTest را شامل می شود. در صورت تمایل می توانید به این کلاس های تست گیری، کلاس دیگری اضافه نمایید. کافی است کلاس مورد نظر را به دستور @Suite.SuiteClasses الحاق کنید.

```
package com.vogella.junit.first;
import org.junit.runner.RunWith;
import org.junit.runners.Suite;
import org.junit.runners.Suite.SuiteClasses;
@RunWith(Suite.class)
@SuiteClasses({
        MyClassTest.class,
        MySecondClassTest.class })
public class AllTests {

}
```

7-4-7-اجرای تست از طریق خط فرمان/command line

می توانید تست مورد نظر را خارج از محیط برنامه نویسی (IDE) و به وسیله ی کدهای استاندارد جاوا راه اندازی کنید. برنامه نویس قادر خواهد بود با ترکیب سیستم های کامپایلی (build system) نظیر Apache Maven یا Gradle با یک Continuous Integration Server (همچون Jenkins) تست ها را به صورت خودکار و در فواصل زمانی منظم بر روی بخش هایی از نرم افزار به اجرا در آورد.

کلاس org.junit.runner.JunitCore متدی به نام runClasses() را ارائه می دهد. این متد به توسعه دهنده امکان می دهد تا همزمان یک یا چند کلاس تست گیری (test class) را اجرا کند. کلاس مزبور در خروجی آبجکتی از جنس org.junit.runner.Result برمی گرداند. این آبجکت را می توان جهت بازیابی اطلاعات مربوط به تست ها مورد استفاده قرار داد.

کلاس زیر نمایش می دهد چگونه می توان MyClassTest را اجرا نمود. این کلاس، کلاس تست گیری (test class) شما را اجرا کرده و خطاهای ممکن را در console درج می نماید.

```
package de.vogella.junit.first;
import org.junit.runner.JUnitCore;
import org.junit.runner.Result;
import org.junit.runner.notification.Failure;
public class MyTestRunner {
  public static void main(String[] args) {
    Result result = JUnitCore.runClasses(MyClassTest.class);
    for (Failure failure : result.getFailures()) {
      System.out.println(failure.toString());
    }
  }
}
```

کلاس حاضر را می توان مانند سایر برنامه های java در خط فرمان اجرا نمود. کافی است فایل JAR کتابخانه ی JUnit را به classpath اضافه نمایید.

7-5-ساختارهای پایه ای فریم ورک JUnit

7_5_1_Annotation های JUnit

JUnit 4.x با استفاده از annotation های خود، متدهای تست گیری را علامت گذاری کرده و امکان تنظیم اختصاصی آن ها را برای توسعه دهنده فراهم می آورد. جدول زیر لیستی از annotation های مهم در فریم ورک JUnit را به نمایش می گذارد.

JUnit Annotations	
Annotation	شرح
@Test public void method()	Annotation حاضر نشانگر این است که متد مورد نظر، یک متد تست گیری / آزمایشی است.
@Test (expected = Exception.class)	در صورتی که متد حاضر، exception نام برده را صادر نکرد، با شکست مواجه می شود.
@Test(timeout=100)	در صورتی که اجرای متد بیش از 100 میلی ثانیه به طول انجامید، با شکست مواجه می شود.
@Before public void method()	این متد قبل از هر تست اجرا می شود. در واقع با استفاده از این متد محیط برای تست گیری آماده می شود(عملیات ابتدایی همچون خواندن داده های ورودی، مقداردهی اولیه کلاس انجام می پذیرد).

@After public void method()	بعد از هر تست اجرا می شود. از این متد برای پاک سازی محیط تست گیری (همچون حذف داده های موقتی، بازگردانی تنظیمات پیش فرض) استفاده می شود. بعلاوه قادر است با حذف ساختارهای هزینه بر و سنگین در حافظه، میزان استفاده از آن را کاهش دهد.
@BeforeClass public static void method()	این متد یکبار قبل از اجرای تمامی تست ها اجرا می شود. متد مذکور برای اجرای عملیات زمان بر، برای مثال، اتصال به دیتابیس مورد استفاده قرار می گیرد. توابعی که با این annotation نشانه گذاری شده باشند، می بایست برای کار (سازگاری) با JUnit به صورت static تعریف شده باشند.
@AfterClass public static void method()	این متد یکبار، پس از اینکه تمام تست ها انجام شده و به پایان رسیدند، اجرا می شود. با فراخوانی این متد، عملیات پاک سازی همچون قطع اتصال به دیتابیس صورت می گیرد. متدهایی که با این annotation نشانه گذاری شده اند، می بایست جهت سازگاری با JUint به صورت static (با کلیدواژه ی static) تعریف شده باشند.
@Ignore or @Ignore("Why disabled")	با درج این annotation بالای متد، متد تست گیری نادیده گرفته شده یا غیرفعال می شود. این به خصوص برای زمانی که کدهای اصلی تغییر یافته ولی test case هنوز متناسب با آن ویرایش و تنظیم نشده، یا اگر مدت اجرای تست بیش از حد طولانی شده باشد، مفید واقع می شود. توصیه می شود پارامتر اختیاری را نیز لحاظ نموده و مقداردهی نمایید (چرا تست غیرفعال شده است).

2-5-7-دستورات Assert

JUnit کلاسی به نام Assert`class را شامل می شود. این کلاس توابع static ای را ارائه می دهد که شرایط خاصی را چک می کنند. دستورات assert معمولا با assert` آغاز می شوند. متد assertion، خروجی واقعی تست را با مقدار مورد انتظار مقایسه می کند و چنانچه مقایسه با شکست مواجه شد، یک AssertionException صادر می کند.

جدول زیر این توابع را به صورت اجمالی همراه با شرح عملکرد هریک به نمایش می گذارد. پارامترهای ذکر شده در []، اختیاری بوده و از جنس String هستند.

توابع assert

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

دستور	شرح
fail(message)	تست با شکست مواجه شده و در خروجی پیغامی اختیاری نمایش داده می شود. توسعه دهنده ممکن است از این annotation برای غیرفعال کردن بخشی از کد یا مانع از جرا شدن آن، استفاده کند.
assertTrue([message,] boolean condition)	بررسی می کند آیا شرط بولی صحیح است یا خیر.
assertFalse([message,] boolean condition)	بررسی می کند آیا نتیجه ی شرط بولی غلط است یا خیر.
assertEquals([message,] expected, actual)	تست می کند آیا دو مقدار با هم برابر هستند یا خیر. توجه: درباره ی آرایه باید گفت که صرفا reference آن بررسی می شود و محتوای آرایه ها مورد توجه قرار نمی گیرد.
assertEquals([message,] expected, actual, tolerance)	تست می کند آیا مقادیر float یا double با هم مطابقت دارند یا خیر. پارامتر tolerance تعداد ارقام اعشاری است که باید یکسان باشند.
assertNull([message,] object)	بررسی می کند آیا آبجکت مورد تست null هست یا خیر.
assertNotNull([message,] object)	بررسی کرده و اطمینان حاصل می نماید که آبجکت null نباشد.
assertSame([message,] expected, actual)	اطمینان حاصل می کند که هر دو متغیر به یک آبجکت اشاره داشته باشند.
assertNotSame([message,] expected, actual)	بررسی می کند که دو متغیر، هر یک به آبجکت جداگانه اشاره داشته باشند.

3-5-7-ترتیب اجرای تست

فریم ورک اجرای تست JUnit فرض را بر این می گذارد که تمامی متدهای آزمایشی (test method) با ترتیبی غیر مشخص و دلخواه اجرا شوند. در واقع تستی که به صورت کارامد نوشته شده باشد، نباید ترتیب اجرای خاصی داشته باشد. تست هایی که برای آزمایش بخش های مختلف برنامه نوشته می شود، نباید به سایر تست ها وابستگی داشته باشد.

از JUnit 4.11 به بعد، به صورت پیش فرض توصیه می شود از ترتیب deterministic و نه predictable برای اجرای تست بر روی نرم افزار استفاده نمایید.

می توانید با استفاده از یک annotation مشخص کنید که متدهای تست گیری (test method) بر اساس اسم متد مرتب شوند (در واقع بر اساس ترتیب حروف الفبا). به منظور فعال سازی این ویژگی، بایستی کلاس تست گیری (test class) خود را با علامت گذاری (annotate) @FixMethodOrder(MethodSorters.NAME_ASCENDING) نمایید. همچنین می توانید با استفاده از پارامتر MethodSorters.DEFAULT در annotation، به صورت صریح، حالت پیش فرض را تنظیم نمایید. بعلاوه می توانید از MethodSorters.JVM که از تنظیمات پیش فرض JVM استفاده می کند، بهره بگیرید که ممکن است از یک اجرا به اجرای دیگر متفاوت باشد.

4-5-7-غیرفعال سازی تست ها

@Ignore این امکان را فراهم می آورد که یک تست را به صورت static غیرفعال کرده و در اصطلاح آن را در تست نادیده بگیرید. یا می توانید با استفاده از پارامترهای Assume.assumeFalse یا Assume.assumeTrue یک شرط برای تست تعریف نمایید. در صورتی که شرط نتیجه ی true بدهد و برقرار باشد، پارامتر Assume.assumeFalse تست مربوطه را نامعتبر (invalid) معرفی و نشانه گذاری می کند. چنانچه شرط نتیجه ی false بدهد، پارامتر Assume.assumeTrue پس از ارزیابی تست، آن را نامعتبر معرفی و علامت گذاری می کند. به طور مثال، دستور زیر یک تست را در سیستم عامل Linux را غیرفعال می کند:

```
Assume.assumeFalse(System.getProperty("os.name").contains("Linux"));
```

6-7-پشتیبانی محیط برنامه نویسی Eclipse از JUnit

1-6-7-ایجاد تست های JUnit

در صورت تمایل می توانید تست های JUnit را خود به صورت دستی بنویسید، اما محیط کاری Eclipse این امکان را برای شما فراهم می آورد تا تست های JUnit را به صورت ویزادی و طبق یک روال از پیش تعیین شده تهیه نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

برای مثال، می توان از ایجاد تست JUnit یا یک کلاس تست گیری جهت آزمایش عملکرد کلاس از پیش نوشته شده نام برد. برای این منظور بر روی کلاس جدید راست کلیک کرده، کلاس مورد نظر را از Package Explorer_ view انتخاب نمایید. سپس بار دیگر بر روی آن راست کلیک کرده و مسیر New ▸ JUnit Test Case را طی کنید.

در صورت تمایل می توانید این کار را به صورت ویزاردی و با استفاده از JUnits Wizards انجام دهید. برای دستیابی به این راهنمای نصب ویزاردی، کافی است مسیر رو به رو را دنبال کنید: File ▸ New ▸ Other... ▸ Java ▸ JUnit.

2-6-7-اجرای تست های JUnit

محیط کاری Eclipse به توسعه دهنده این امکان را می دهد تا تست ها را به صورت تعاملی اجرا کند.

به منظور اجرای یک تست، کلاس تست گیری (test class) را انتخاب نموده، سپس بر روی آن راست کلیک کنید. حال گزینه ی Run-as ▸ JUnit Test را انتخاب کنید.

این کار JUnit را راه اندازی کرده و تمامی متدهای تست گیری (test method) را داخل کلاس اجرا می کند.

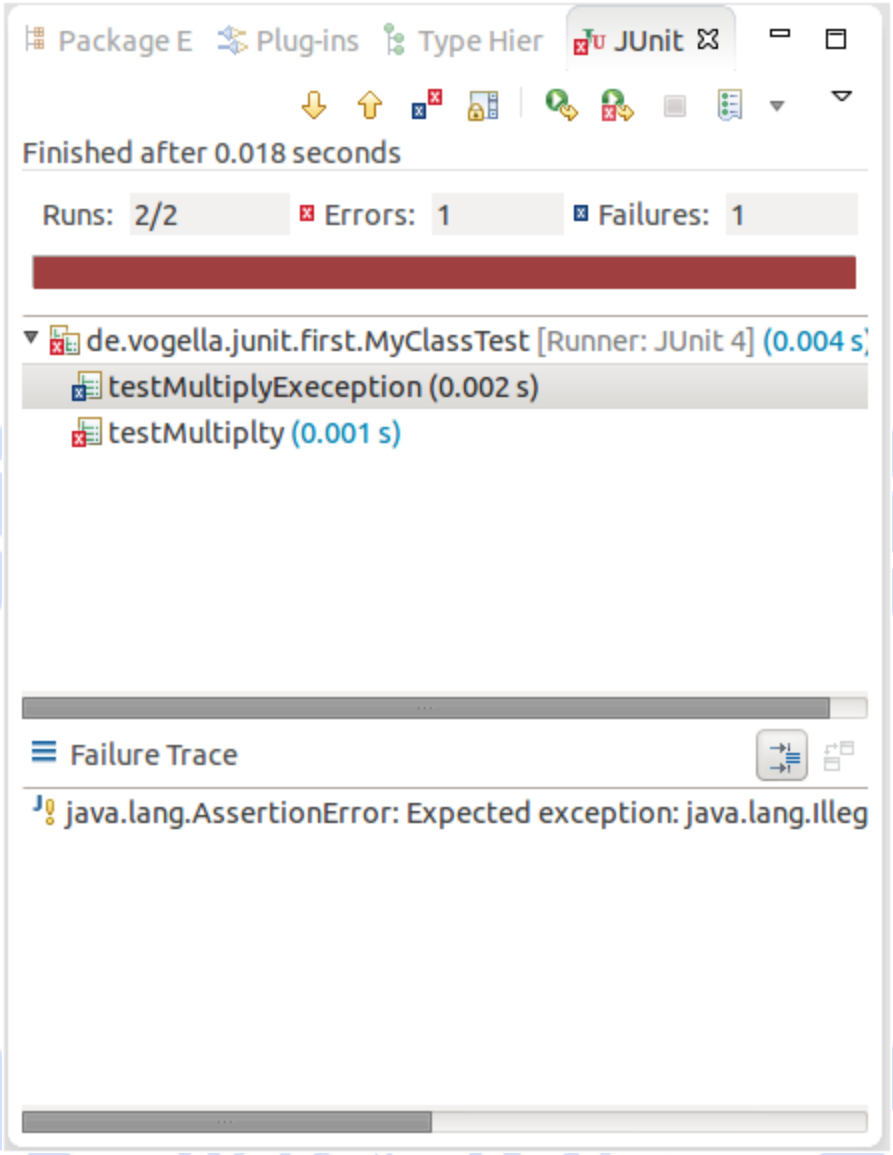
Eclipse به برنامه نویس اجازه می دهد تا با فشردن کلیدهای Alt+Shift+X به صورت همزمان، تست مورد نظر را در کلاس انتخابی اجرا کند.

برای اینکه فقط و فقط تست انتخابی اجرا شود، اشاره گر موس را بر روی اسم متد تست گیری قرار داده و سپس کلیدهای فوق را فشار دهید.

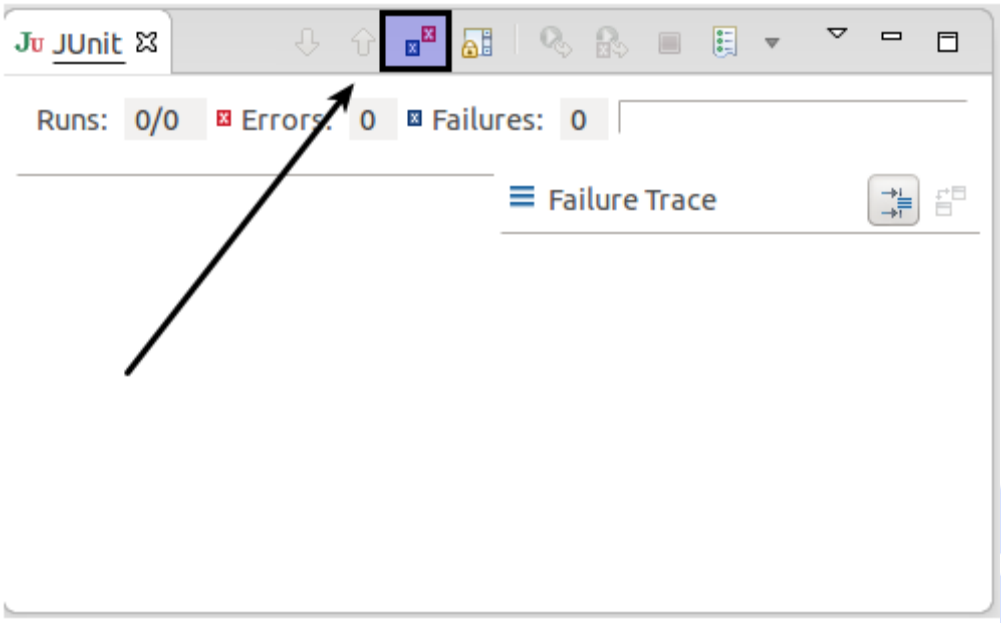
به منظور مشاهده ی نتیجه ی تست JUnit، محیط کاری Eclipse یک ابزار به نام JUnit view را ارائه می دهد. می توانید unit test های دلخواه خود را به صورت انفرادی در view مذکور انتخاب نموده و پس از راست کلیک بر روی آن و انتخاب گزینه ی Run، آن ها را مجددا اجرا نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

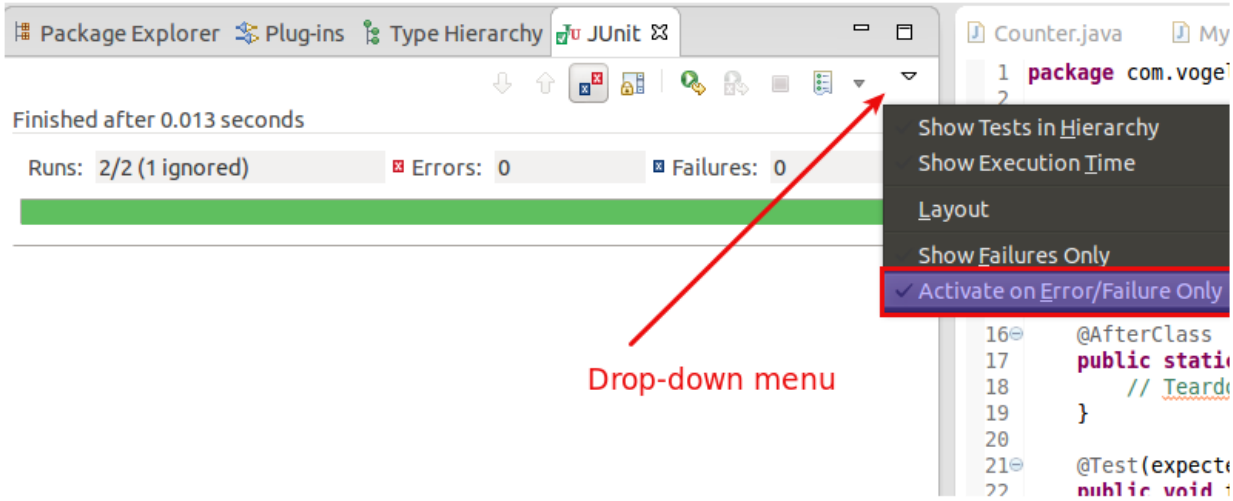
88146323 - 88446780 - 88146330



به صورت پیش فرض، این view تمامی تست های موجود را به نمایش می گذارد. می توانید این view را طوری تنظیم نمایید که تنها تست های ناموفق (failed) را نمایش دهد.



همچنین می توانید view را طوری تنظیم کنید که تنها در صورت برخورد با یک تست ناموفق فعال شود.



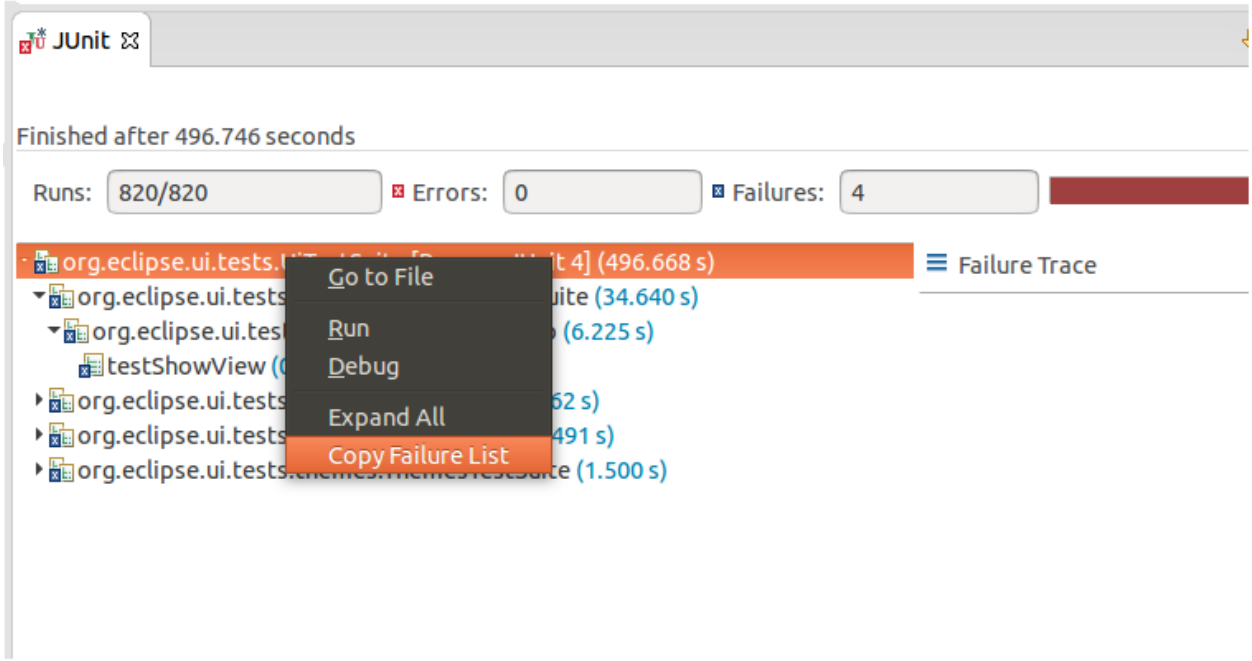
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

توجه: محیط برنامه نویسی Eclipse تنظیماتی مرتبط با اجرای تست ها (run configurations) ایجاد می کند. جهت مشاهده و ویرایش آن ها می توانید به این صورت اقدام نمایید: Run ▸ Run Configurations... menu.

3-6-7-استخراج و بازیابی تست های ناموفق و stacktrace ها

جهت بازیابی لیست مربوط به تست ناموفق، بر روی نتیجه ی تست راست کلیک کرده و Copy Failure List را انتخاب نمایید. این کار سبب می شود تست های ناموفق و stack trace ها در clipboard کپی و درج شوند (stack trace عبارت است فرایند ردیابی اجرای کدهای برنامه در بخشی از حافظه به نام stack).



4-6-7-امکان static import در فریم ورک JUnit

Static import یک امکان است که به فیلدهایی که داخل کلاس به صورت public static تعریف شده اند، این اجازه را می دهد تا بدون ذکر اسم کلاس میزبان، بکار برده شوند.

دستورات assert فریم ورک مذکور، برای اینکه قابلیت نوشتن دستورات تست گیری (test statement) مختصر و بهینه را برای توسعه دهنده فراهم کنند، معمولا به صورت public static

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

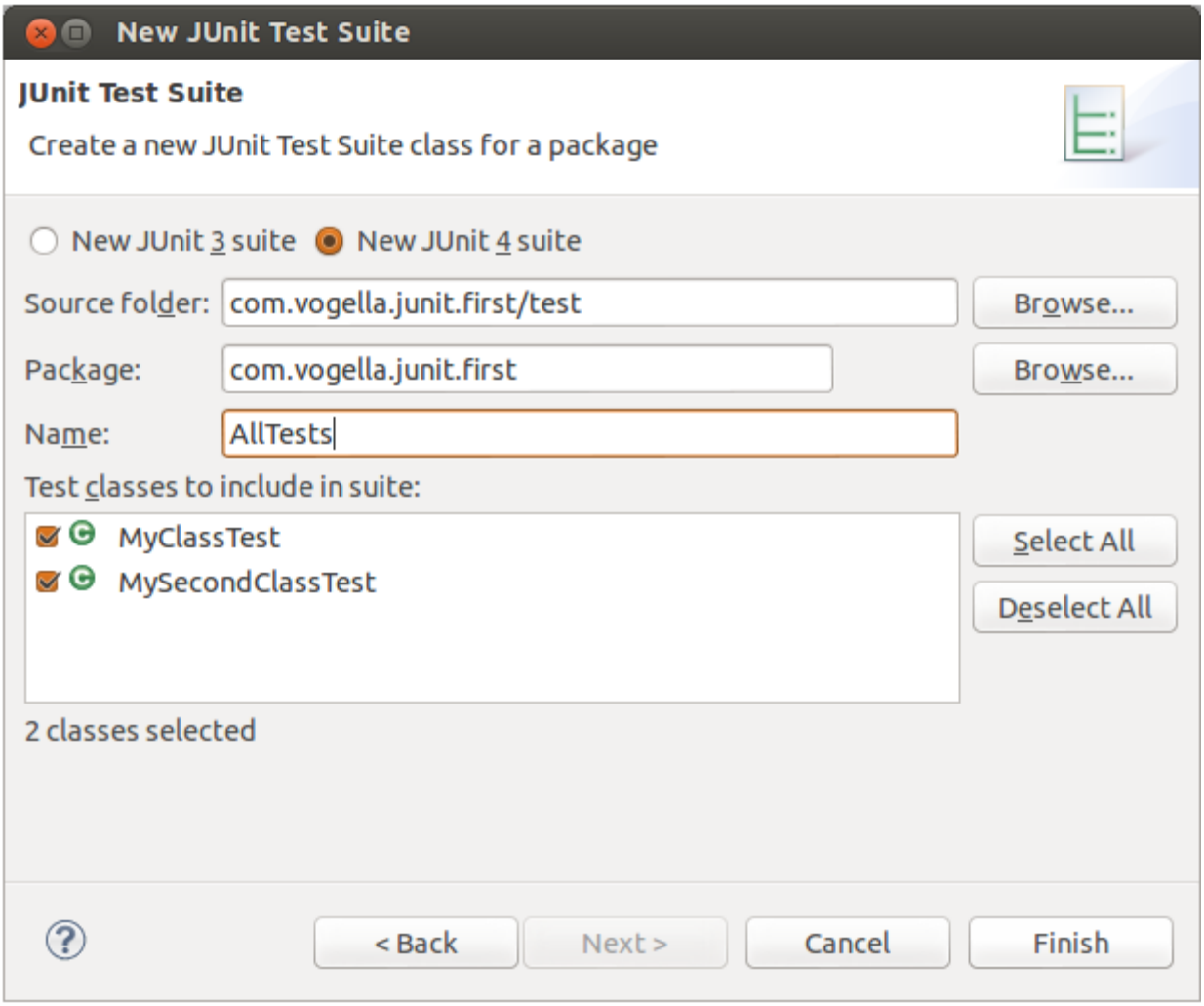
88146323 - 88446780 - 88146330

تعریف می شوند. تکه کد زیر یک دستور assert را با دو حالت ممکن (با امکان static imports و یکی بدون static import) به نمایش می گذارد.

```
// without static imports you have to write the following statement
Assert.assertEquals("10 x 5 must be 50", 50, tester.multiply(10, 5));

// alternatively define assertEquals as static import
import static org.junit.Assert.assertEquals;

// more code

// use assertEquals directly because of the static import
assertEquals("10 x 5 must be 50", 50, tester.multiply(10, 5));
```

5-6-7-برنامه ی راهنما/Wizard برای ساختن مجموعه تست (test suite)

می توانید در محیط کاری Eclipse و با امکاناتی که در اختیار شما قرار می دهد، به راحتی یک test suite (مجموعه تست) ایجاد نمایید. برای نیل به این هدف، ابتدا کلاس های آزمایشی و تست گیری (test class) که می بایست در مجموعه (suite) گنجانده شود را از طریق کادر Package Explorer گزینش نموده، در این کادر بر روی آن ها راست کلیک نمایید و سپس مسیر رو به رو را طی کنید: New ▸ Other... ▸ JUnit ▸ JUnit Test Suite.



6-6-7-تست exception و خطاها

@Test (expected = Exception.class) دارای قابلیتی محدودی است چرا که تنها قادر است یک exception (خطا) را تست کند. به منظور تست کلی exception ها، می توانید از الگوی تست گیری (test pattern) زیر استفاده نمایید.

```
try {
  mustThrowException();
  fail();
} catch (Exception e) {
  // expected
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
  // could also check for message of exception, etc.
}
```

7-6-تست نویسی برای افزونه ها (JUnit Plug-in Test)

توسعه دهنده می تواند با استفاده از تست های JUnit Plug-in برای افزونه های نرم افزاری خود نیز تست طراحی کند. این تست ها در بستر test runner ویژه راه اندازی شده که به یک نمونه ی مستقل VM برای اجرا در محیط Eclipse احتیاج دارد. در واقع متدهای تست گیری محصور در کلاس (test class)، داخل آن نمونه مجزا (instance) اجرا می شوند.

7-7-نصب JUnit

7-7-1-استفاده از JUnit با سیستم کامپایل Gradle

به منظور استفاده از JUnit در فایل Gradle build، لازم است کتابخانه (dependency) testCompile را به فایل build خود اضافه نمایید.

```
apply plugin: 'java'
dependencies {
  testCompile 'junit:junit:4.12'
}
```

7-7-2-استفاده از JUnit با سیستم کامپایل Maven

به منظور استفاده از JUnit در فایل Maven build، می بایست کتابخانه (dependency) زیر را به فایل pom خود اضافه نمایید.

```
<dependency>
   <groupId>junit</groupId>
   <artifactId>junit</artifactId>
   <version>4.12</version>
</dependency>
```

3-7-7-استفاده از JUnit درون ساخته ی محیط کاری Eclipse

محیط برنامه نویسی Eclipse با یک ورژن خاص از JUnit در اختیار برنامه نویسان قرار می گیرد. اگر از Eclipse برای ساخت و توسعه ی اپلیکیشن خود بهره می گیرید، در آن صورت لزومی ندارد افزونه یا محتوای جدیدی از اینترنت دانلود نمایید.

4-7-7-دانلود کتابخانه ی JUnit

اگر می خواهید کتابخانه ی JUnit مورد استفاده را به صورت صریح کنترل کنید، در آن صورت کافی است فایل JUnit4.x.jar را از وب سایت زیر دانلود کنید. محتوای دانلود شده، junit-4.*.jar را دربردارد که همان کتابخانه ی JUnit است. این کتابخانه را به پروژه ی Java خود و classpath مورد نظر اضافه نمایید.

http://junit.org/

8-7-تنظیم محیط برنامه نویسی Eclipse برای استفاده از امکان static import کتابخانه ی JUnits

محیط کاری Eclipse به خودی خود همیشه قابلیت ایجاد دستورات static import مربوطه را ندارد. شما می توانید این محیط برنامه نویسی را طوری تنظیم نمایید که با استفاده از امکان code completion خود توابع معمول و پرکاربرد JUnit را وارد متن برنامه کرده و دستورات static import را به صورت خودکار اضافه نماید. برای دستیابی به این هدف، ابتدا پنجره ی Preferences را از طریق Window ▸ Preferences باز نمایید و سپس مسیر رو به رو را طی کنید: Java ▸ Editor ▸ Content Assist ▸ Favorites.

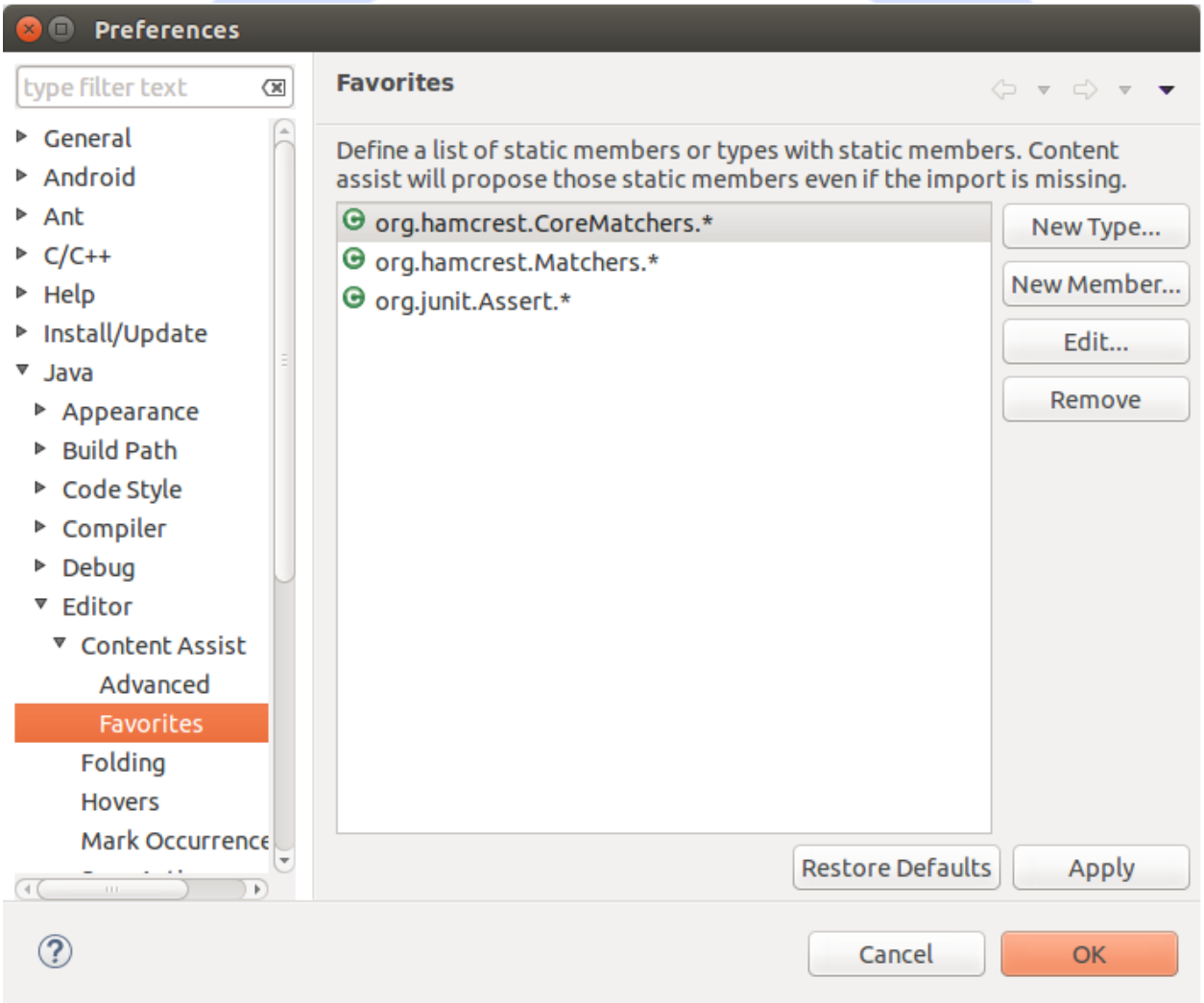
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

با استفاده از دکمه ی New Type دستورات زیر را به آن اضافه نمایید:

- org.junit.Assert
- org.hamcrest.CoreMatchers
- org.hamcrest.Matchers

با این کار متدهای assertTrue، assertFalse و assertEquals را مستقیما در Content Assists قابل دسترسی و آماده ی استفاده می نمایید.

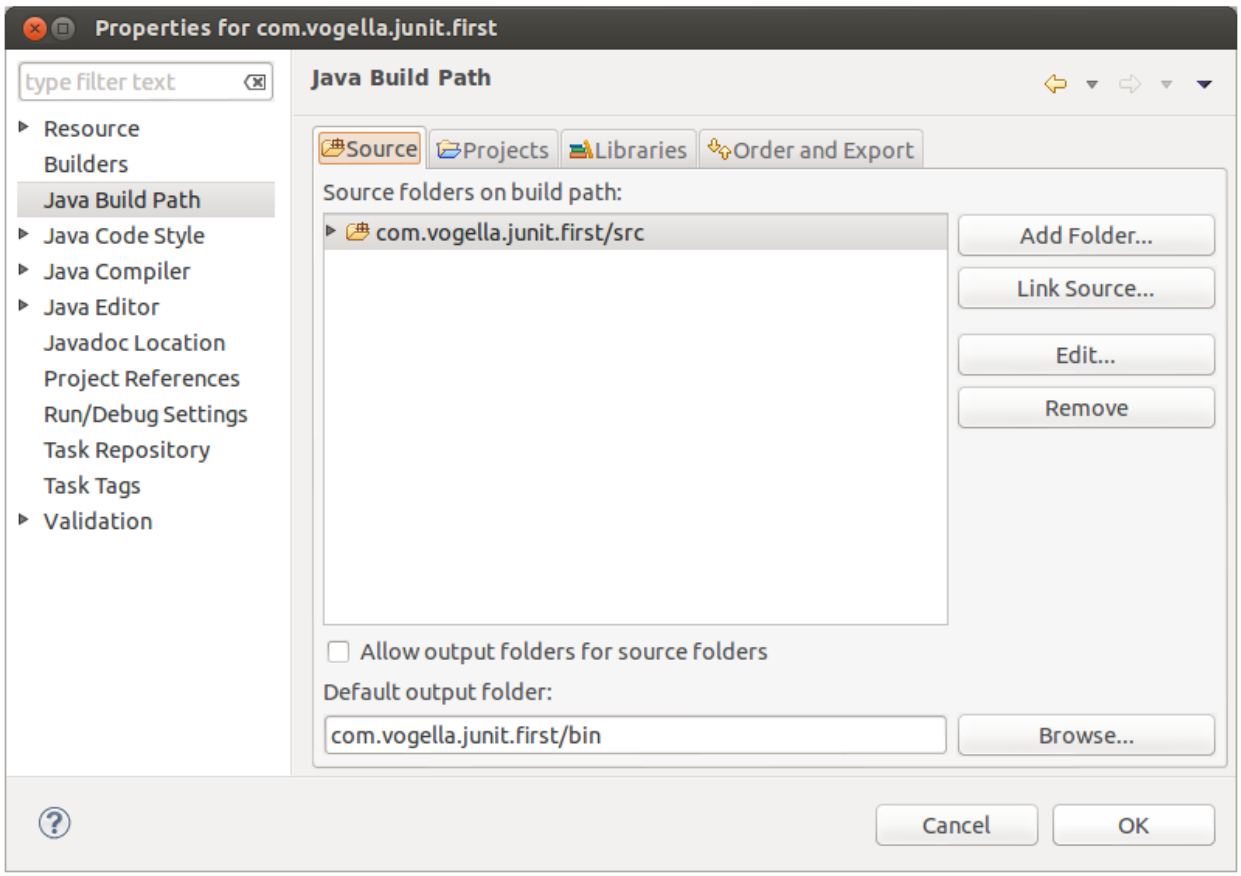


اکنون شما می توانید با استفاده از Content Assists (که با فشردن کلیدهای Ctrl+Space به طور همزمان به سرعت در اختیار شما قرار می گیرد) به راحتی متد و import را اضافه نمایید.

تمرین: استفاده از JUnit

آماده سازی پروژه

یک پروژه ی جدید به نام com.vogella.junit.first ایجاد نمایید. سپس یک پوشه ی جدید به نام test در پروژه ی مزبور ایجاد کنید. برای این منظور بر روی پروژه ی مورد نظر راست کلیک کرده، Properties را انتخاب نمایید و این مسیر را طی کنید: Java ▸ Build Path. حال بر روی تب Source کلیک نموده و آن را انتخاب کنید.

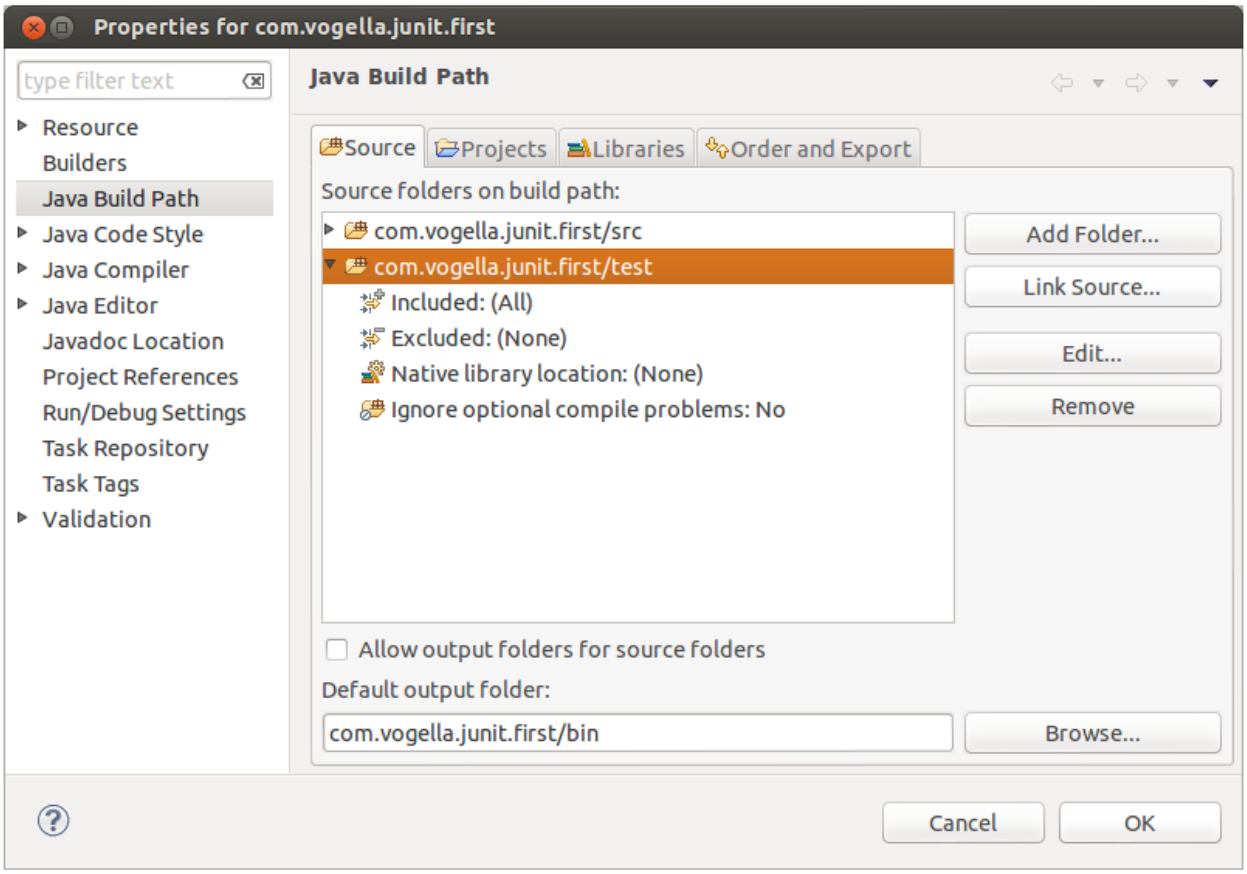


دکمه ی Add Folder را کلیک کنید. پس از آن، دکمه ی Create New Folder را فشار دهید. اسم پوشه را test انتخاب نمایید.

نتیجه در تصویر زیر به نمایش گذاشته شده است.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



توجه: در صورت تمایل می توانید با راست کلیک بر روی پروژه و انتخاب New ▸ Source Folder، یک پوشه ی اصلی جدید نیز اضافه نمایید.

1-8-7-ساخت یک کلاس Java

در پوشه ی src، پکیج com.vogella.junit.first را ایجاد کرده و سپس کلاس زیر را به آن اضافه نمایید.

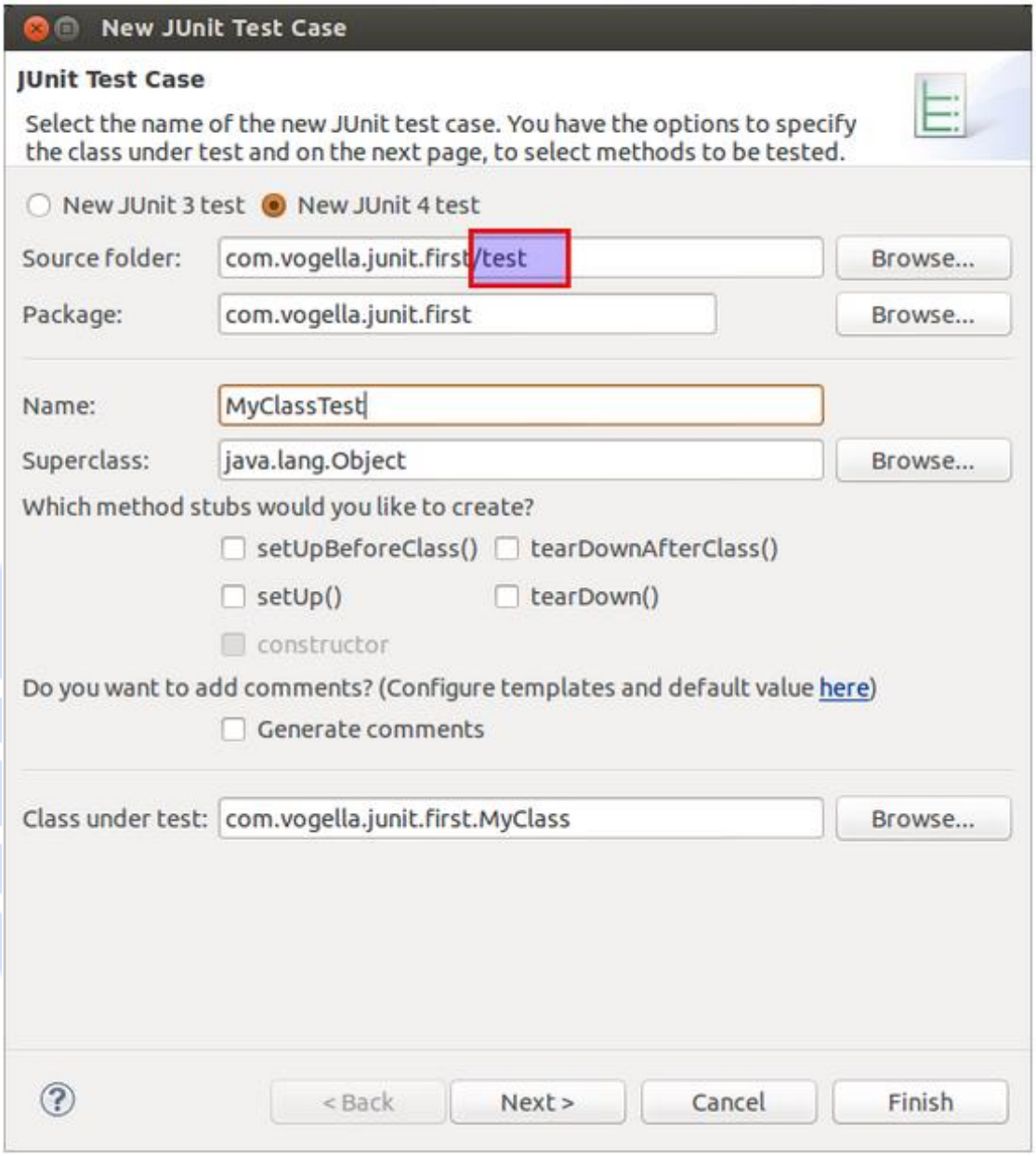
```
package com.vogella.junit.first;
public class MyClass {
  public int multiply(int x, int y) {
    // the following is just an example
    if (x > 999) {
      throw new IllegalArgumentException("X should be less than 1000");
    }
    return x / y;
```

```
  }
}
```

2-8-7-ساخت و طراحی یک تست JUnit

بر روی کلاس جدید، داخل کادر Package Explorer، راست کلیک نموده و سپس این مسیر را طی نمایید: New ▸ JUnit Test Case.

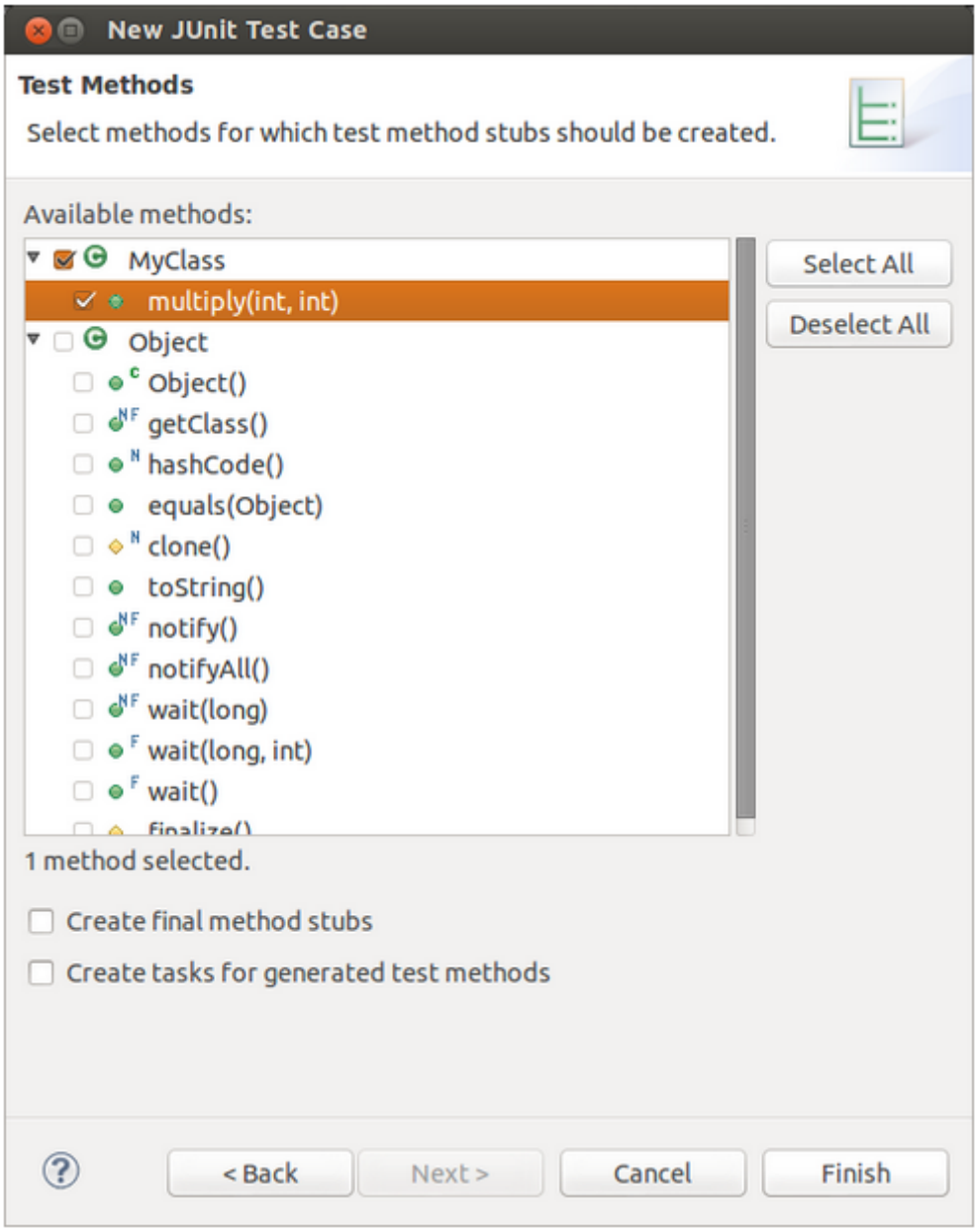
در برنامه ی راهنمای نصب (ویزارد) که در زیر مشاهده می کنید، حتما دکمه ی رادیویی New JUnit 4 test را فعال کرده و سپس پوشه ی source را بر روی test تنظیم نمایید تا کلاس تست در این پوشه ایجاد شود.



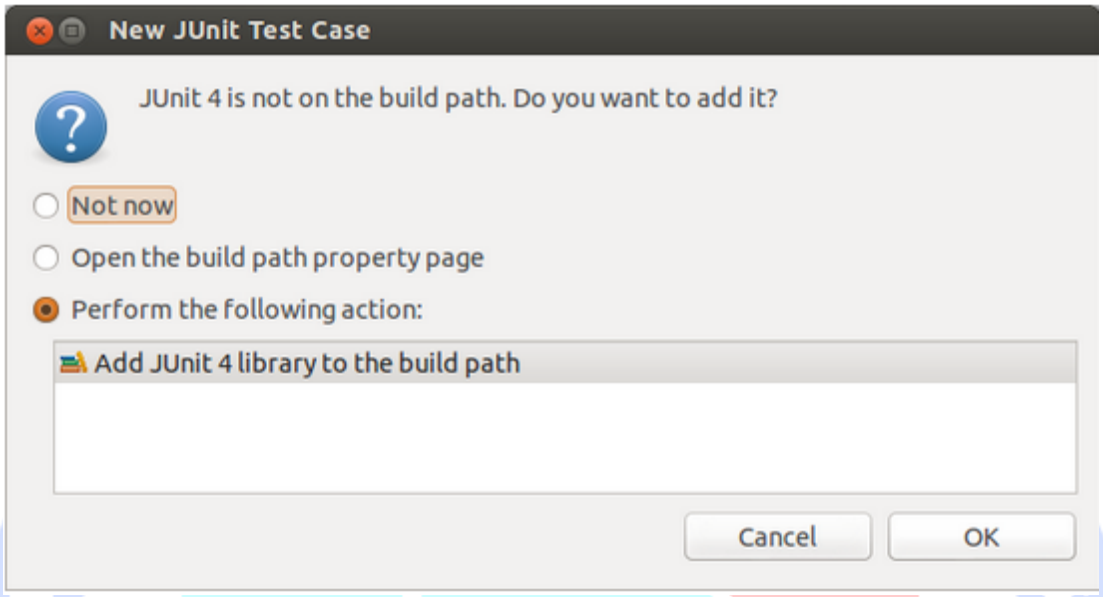
بر روی دکمه ی Next کلیک کرده و متدهایی که می خواهید تست شوند را انتخاب نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



اگر کتابخانه ی JUnit در classpath پروژه ی جاری قید نشده بود، در آن صورت محیط کاری Eclipse از شما می خواهد که حتما آن را اضافه نمایید. با این کار شما در واقع JUnit و امکانات آن را به پروژه ی خود اضافه می نمایید.



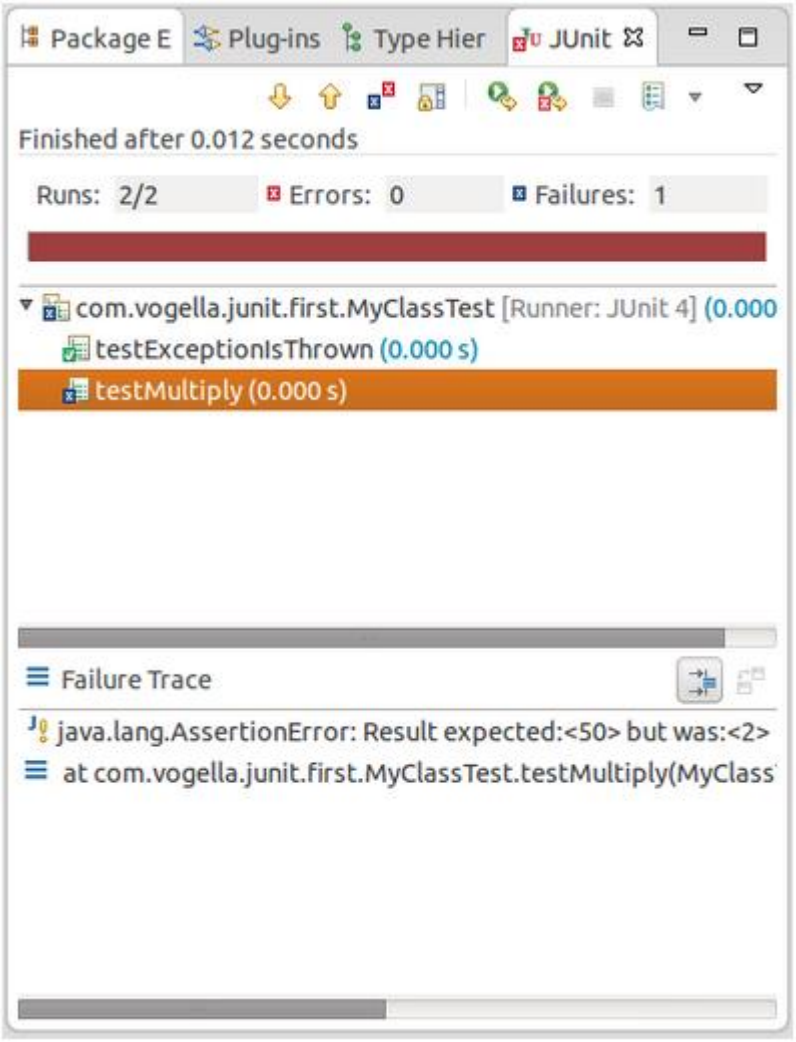
حال تست مورد نظر را با کد و محتوای زیر ایجاد نمایید.

```
package com.vogella.junit.first;
import static org.junit.Assert.assertEquals;
import org.junit.AfterClass;
import org.junit.BeforeClass;
import org.junit.Test;
public class MyClassTest {
  @Test(expected = IllegalArgumentException.class)
  public void testExceptionIsThrown() {
    MyClass tester = new MyClass();
    tester.multiply(1000, 5);
  }
  @Test
  public void testMultiply() {
    MyClass tester = new MyClass();
    assertEquals("10 x 5 must be 50", 50, tester.multiply(10, 5));
  }
}
```

2-8-7-اجرای تست در محیط برنامه نویسی Eclipse

بر روی کلاس تست گیری (test class) جدید راست کلیک کرده و سپس گزینه ی Run-As ▸ JUnit Test را انتخاب نمایید.

نتیجه ی این تست ها در کادر JUnit (view) به نمایش گذاشته می شود. در مثال حاضر، طبق انتظار یک تست بایستی موفقیت آمیز باشد و اما دیگری در خروجی یک خطا صادر کند. خطای نام برده با نوار قرمز نمایش داده می شود.



تست حاضر به این خاطر ناموفق است که کلاس (ماشین حساب) در انجام عملیات مورد انتظار (ضرب) با مشکل مواجه می شود. در واقع به جای انجام عملیات ضرب، دو عدد را بر هم تقسیم می کند. حال اشکال فنی برنامه را برطرف نموده و بار دیگر تست را اجرا کنید تا نوار سبز در خروجی تست به نمایش گذاشته شود.

7-9-تنظیمات و امکانات پیشرفته ی JUnit

1-9-7-اجرای تست های دارای پارامتر مشخص (Parameterized test)

JUnit این امکان را برای شما فراهم می آورد تا پارامترهای دلخواه را در کلاس های تست گیری مورد استفاده قرار دهید. کلاس مورد نظر می تواند شامل یک متد تست گیری بوده و این متد هر بار با پارامترهای ارائه شده ی مختلف اجرا شود.

بایستی کلاسی که قرار است پارامترهای مختلف به آن ارسال شوند را با دستور @RunWith(Parameterized.class) علامت گذاری نمایید.

چنین کلاسی (کلاس تست گیری) باید یک متد static در خود داشته که با دستور @Parameters نشانه گذاری شده باشد. این متد یک مجموعه آرایه (array collection) تولید کرده و در خروجی بازمی گرداند. تک تک آیتم ها در این آرایه به عنوان پارامتر ورودی به متد تست گیری (test method) حاضر در این کلاس ارسال می شوند.

می توانید فیلدهای public را با دستور @Parameter نشانه گذاری نمایید و بدین سیله مقادیر تستی (مقادیر یا پارامترهای آزمایشی) را داخل تست تزریق نمایید.

کد زیر نمونه ای از یک تست که پارامترهایی از بیرون به آن تزریق می شوند را به نمایش می گذارد. تست حاضر متد multiply() از کلاس MyClass را که یک کلاس دورنی و nested داخل کلاس میزبان می باشد را با ارسال پارامتر به آن، به منظور تست، آزمایش می نماید.

```
package testing;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.junit.runners.Parameterized;
import org.junit.runners.Parameterized.Parameters;
import java.util.Arrays;
import java.util.Collection;
import static org.junit.Assert.assertEquals;
import static org.junit.runners.Parameterized.*;
@RunWith(Parameterized.class)
public class ParameterizedTestFields {
   // fields used together with @Parameter must be public
   @Parameter
   public int m1;
   @Parameter (value = 1)
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    public int m2;
    // creates the test data
    @Parameters
    public static Collection<Object[]> data() {
       Object[][] data = new Object[][] { { 1 , 2 }, { 5, 3 }, { 121, 4 } };
       return Arrays.asList(data);
    }
    @Test
    public void testMultiplyException() {
       MyClass tester = new MyClass();
       assertEquals("Result", m1 * m2, tester.multiply(m1, m2));
    }
    // class to be tested
    class MyClass {
       public int multiply(int i, int j) {
          return i *j;
       }
    }
}
```

یا می توانید به جای استفاده از دستور Parameter@، تابع سازنده (constructor) را بکار ببرید. در این تابع مقادیر مورد استفاده در هر تست نگهداری می شود. لازم به ذکر است که تعداد المان های موجود در آرایه که متد مورد نظر (متدی که با دستور فوق نشانه گذاری شده باشد) ارائه می دهد، بایستی با تعداد پارامترهای حاضر در تابع سازنده ی کلاس برابر باشد. در واقع به ازای هر پارامتر یک کلاس مستقل ایجاد شده و سپس مقادیر آزمایشی به واسطه ی تابع سازنده (constructor) به کلاس مورد نظر فرستاده می شوند.

```
package de.vogella.junit.first;
import static org.junit.Assert.assertEquals;
import java.util.Arrays;
import java.util.Collection;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.junit.runners.Parameterized;
import org.junit.runners.Parameterized.Parameters;
@RunWith(Parameterized.class)
public class ParameterizedTestUsingConstructor {
   private int m1;
   private int m2;
   public ParameterizedTestUsingConstructor(int p1, int p2) {
      m1 = p1;
      m2 = p2;
   }
   // creates the test data
   @Parameters
   public static Collection<Object[]> data() {
      Object[][] data = new Object[][] { { 1 , 2 }, { 5, 3 }, { 121, 4 } };
```

```
    return Arrays.asList(data);
  }
  @Test
  public void testMultiplyException() {
    MyClass tester = new MyClass();
    assertEquals("Result", m1 * m2, tester.multiply(m1, m2));
  }
  // class to be tested
  class MyClass {
    public int multiply(int i, int j) {
      return i *j;
    }
  }
}
```

حال اگر این کلاس تست گیری (test class) را اجرا کنید، متد حاضر در آن هر بار با پارامتر تعریف شده به صورت جداگانه اجرا می شود. به عنوان مثال، در نمونه ی فوق متد تست گیری مورد نظر سه بار اجرا می شود.

با مراجعه به آدرس اینترنتی https://github.com/Pragmatists/JUnitParams می توانید با روشی که نوشتن آن به مراتب آسان و معطف تر هست، آشنا شوید.

2-9-7-دستور @Rule (annotation)

Rule های کتابخانه ی JUnit به شما این امکان را می دهند تا رفتار و قابلیت های مورد نظر را به تک تک تست های داخل کلاس تست گیری (test class) اختصاص داده و اعمال کنید. می توانید فیلدهایی که از جنس TestRule هستند را با @Rule نشانه گذاری نمایید. همچنین می توانید آبجکت هایی ایجاد کنید که قابلیت تنظیم و استفاده از آن ها در متدهای تست گیری وجود داشته باشد. با این کار به میزان انعطاف پذیری تست شما افزوده می شود. به طور مثال، می توانید مشخص کنید که به هنگام اجرای کد آزمایشی، انتظار دارید که کدام پیغام خطا (exception message) نمایش داده شود.

```
package de.vogella.junit.first;
import org.junit.Rule;
import org.junit.Test;
import org.junit.rules.ExpectedException;
public class RuleExceptionTesterExample {
  @Rule
  public ExpectedException exception = ExpectedException.none();
```

```
  @Test
  public void throwsIllegalArgumentExceptionIfIconIsNull() {
    exception.expect(IllegalArgumentException.class);
    exception.expectMessage("Negative value not allowed");
    ClassToBeTested t = new ClassToBeTested();
    t.methodToBeTest(-1);
  }
}
```

JUnit از قبل تعداد زیادی از rule ها را پیاده سازی نموده و آن ها را به صورت آماده در اختیار توسعه دهنده قرار می دهند. به طور مثال، کلاس TemporaryFolder این امکان را به شما می دهد تا فایل ها و پوشه هایی تنظیم کنید که خود به خود پس از هر بار اجرای تست حذف می شوند. نمونه ی زیر استفاده از پیاده سازی TemporaryFolder را به نمایش می گذارد.

```
package de.vogella.junit.first;
import static org.junit.Assert.assertTrue;
import java.io.File;
import java.io.IOException;
import org.junit.Rule;
import org.junit.Test;
import org.junit.rules.TemporaryFolder;
public class RuleTester {
  @Rule
  public TemporaryFolder folder = new TemporaryFolder();
  @Test
  public void testUsingTempFolder() throws IOException {
    File createdFolder = folder.newFolder("newfolder");
    File createdFile = folder.newFile("myfilefile.txt");
    assertTrue(createdFile.exists());
  }
}
```


3-9-7-طراحی و تنظیم rule های اختصاصی برای JUnit

جهت نوشتن rule های اختصاصی و دلخواه خود، شما می بایست اینترفیس TestRule را پیاده سازی نمایید. این اینترفیس، متد apply(Statement, Description) را در اختیار توسعه دهنده قرار می دهد که خروجی آن نمونه ای از Statement می باشد. پارامتر Statement نشانگر تست های موجود در بستر JUnit runtime است و Statement#evaluate() نیز تابعی است که این

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

تست ها را به هنگام runtime ، اجرا می کند. پارامتر Description، توصیف گر تست های فردی بوده و به شما اجازه می دهد تا از طریق reflection اطلاعات مرتبط با تست ها را بخوانید.

نمونه ی زیر یک مثال ساده از افزودن دستور log به اپلیکیشن اندرویدی، قبل و بعد از اجرای تست می باشد.

```
package testing.android.vogella.com.asynctask;
import android.util.Log;
import org.junit.rules.TestRule;
import org.junit.runner.Description;
import org.junit.runners.model.Statement;
public class MyCustomRule implements TestRule {
   private Statement base;
   private Description description;
   @Override
   public Statement apply(Statement base, Description description) {
      this.base = base;
      this.description = description;
      return new MyStatement(base);
   }
   public class MyStatement extends Statement {
      private final Statement base;
      public MyStatement(Statement base) {
         this.base = base;
      }
      @Override
      public void evaluate() throws Throwable {
         System.
         Log.w("MyCustomRule",description.getMethodName() + "Started" );
         try {
            base.evaluate();
         } finally {
            Log.w("MyCustomRule",description.getMethodName() + "Finished");
         }
      }
   }
}
```

به منظور استفاده از این rule، کافی است یک فیلد که با @Rule نشانه گذاری شده باشد به کلاس تست گیری (test class) خود اضافه نمایید.

```
@Rule
public MyCustomRule myRule = new MyCustomRule();
```

4-9-7-Category ها و دسته بندی تست ها

می توان تست ها را دسته بندی نموده و یا آن ها را بر اساس اسم annotation ها، داخل دسته ی مورد نظر گنجاند یا از دسته ای حذف کرد. مثال زیر بر اساس نکات و رهنمودهای منتشر شده ی ویرایش JUnit 4.8 می باشد.

```
public interface FastTests { /* category marker */
}
public interface SlowTests { /* category marker */
}
public class A {
	@Test
	public void a() {
		fail();
	}
	@Category(SlowTests.class)
	@Test
	public void b() {
	}
}
@Category({ SlowTests.class, FastTests.class })
public class B {
	@Test
	public void c() {
	}
}
@RunWith(Categories.class)
@IncludeCategory(SlowTests.class)
@SuiteClasses({ A.class, B.class })
// Note that Categories is a kind of Suite
public class SlowTestSuite {
	// Will run A.b and B.c, but not A.a
}
@RunWith(Categories.class)
@IncludeCategory(SlowTests.class)
@ExcludeCategory(FastTests.class)
@SuiteClasses({ A.class, B.class })
// Note that Categories is a kind of Suite
public class SlowTestSuite {
	// Will run A.b, but not A.a or B.c
}
```

7-10-ایجاد آبجکت های ساختگی یا شبیه سازی رفتار آبجکت/Mocking

برای اجرای تست بر روی بخش های مختلف پروژه ی نرم افزاری که همان unit testing نیز خوانده می شود، گاه لازم است از object mocking استفاده نمایید. در این سناریو آبجکت واقعی با یک آبجکت ساختگی که رفتار و عملکرد آبجکت مورد نظر را شبیه سازی می کند، در طول اجرای تست جایگزین می شود.

فریم ورک های متعددی برای mocking (جهت تست نرم افزار) وجود دارد. به منظور کسب اطلاعات بیشتر در مورد فریم ورک های شبیه سازی و تست نرم افزار می توانید به آموزش Mockito تحت آدرس http://www.vogella.com/tutorials/Mockito/article.html یا EasyMock تحت آدرس http://www.vogella.com/tutorials/EasyMock/article.html مراجعه فرمایید.

بخش دوم :

ساخت و طراحی تست های Unit و Instrumentation جهت تست پروژه های اندرویدی

مبحث حاضر به شما آموزش می دهد چگونه می توانید برای اپلیکیشن های اندرویدی خود instrumentation & unit test طراحی نموده و از عملکرد صحیح بخش های مختلف پروژه خود اطمینان حاصل نمایید. سپس برای شما شرح می دهد چگونه این تست ها را در محیط کاری Android Studio و با استفاده از سیستم کامپایل Gradle اجرا نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

برای یادگیری این بخش لازم است با برنامه نویسی اندروید آشنایی کافی داشته باشید.

10-7-مقدمه ای بر تست اپلیکیشن های اندرویدی

7-10-1-تست پروژه های اندرویدی

اپلیکیشن های اندرویدی بر روی دستگاه هایی اجرا می شوند که دارای حافظه و باتری محدودی بوده و پردازنده ی ضعیفی دارند. رفتار اپلیکیشن همچنین بر عوامل خارجی همچون اتصال به اینترنت، استفاده ی کلی از سیستم و غیره ... بستگی دارد. با توجه به آنچه گفته شد، اشکال زدایی، تست و بهینه سازی اپلیکیشن های اندرویدی بسیار مهم می باشد. انتخاب بخش هایی که باید تست شوند نیز حائز اهمیت بوده، به شما کمک می کند تا اپلیکیشن اندرویدی خود را بهبود ببخشید و در آینده عملیات تعمیر و نگهداشت آن را آسان می سازد.

از آنجایی که امکان تست اپلیکیشن های اندرویدی بر روی تمامی دستگاه ها (با تنظیمات و سخت افزار مختلف) وجود ندارد، توصیه می شود تست های نرم افزاری خود را بر روی دستگاه هایی با پیکربندی و سخت افزار معمول اجرا نمایید. لازم است اپلیکیشن خود را حداقل یکبار بر روی دستگاهی با پایین ترین قدرت سخت افزاری تست نمایید. هرچند بد نیست که اپلیکیشن مورد نظر را بر روی دستگاه هایی که بالاترین قدرت سخت افزاری را دارند نیز تست نمایید. برای مثال اپلیکیشن خود را بر دستگاه هایی که چگالی پیکسلی و وضوح تصویری بالایی دارند، تست کرده و مطمئن شوید که برنامه ی تحت موبایل شما با این پیکربندی نیز کاملا سازگار است.

Unit testing اپلیکیشن های اندرویدی می تواند به گروه های زیر تقسیم شود:

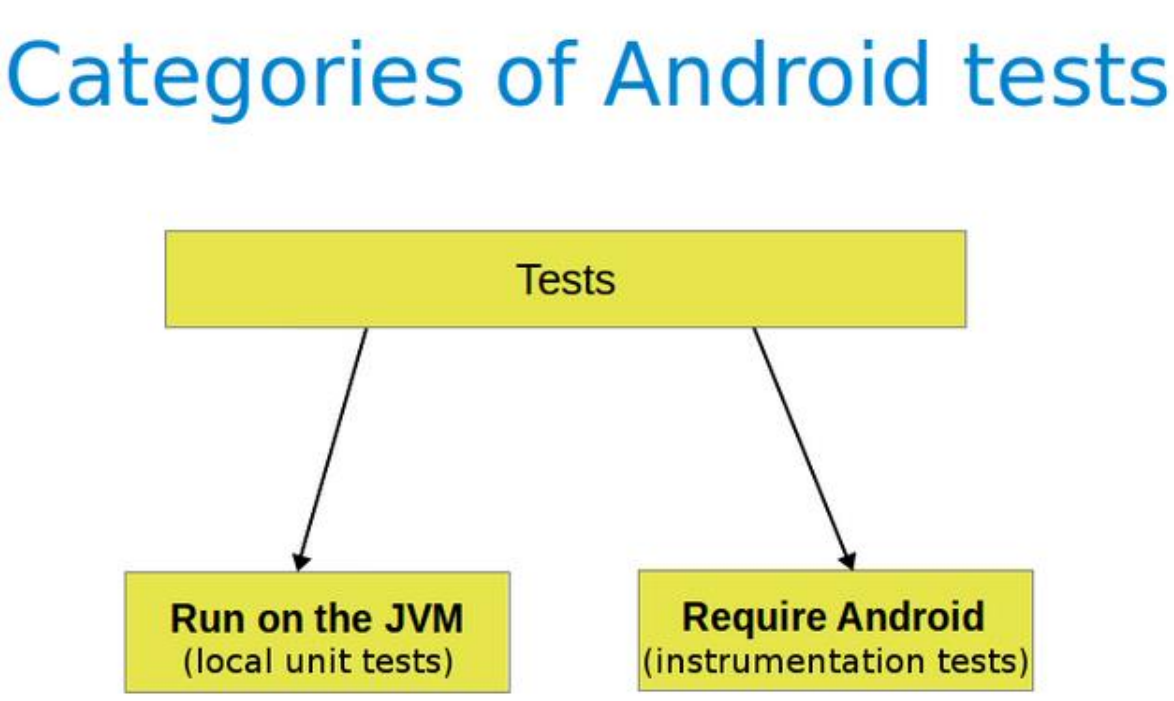
- Local unit test (تست در بستر دستگاه مجازی جاوا) – تست هایی که می توان آن ها را در بستر JVM اجرا کرد در اصطلاح تست های local خوانده می شوند. تا حد امکان بهتر است از تست های local برای تست نرم افزار خود بهره بگیرید. تست های local به این علت که بر روی JVM اجرا می شوند، از لحاظ زمانی به خصوص نسبت به تست

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

هایی که بر روی دستگاه واقعی اندروید اجرا می شوند، سریع تر به انجام می رسند.

- Instrumented unit test (تست بر روی دستگاه واقعی اندروید) – تست هایی که بر روی یک سیستم واقعی اندروید انجام می شود. اگر می خواهید کدی را امتحان کنید که به API و توابع کتابخانه ای اندروید احتیاج دارد، در آن صورت لازم است این تست ها را بر روی دستگاه اندرویدی اجرا کنید. متاسفانه این امر سبب می شود زمان اجرای تست طولانی تر شود.

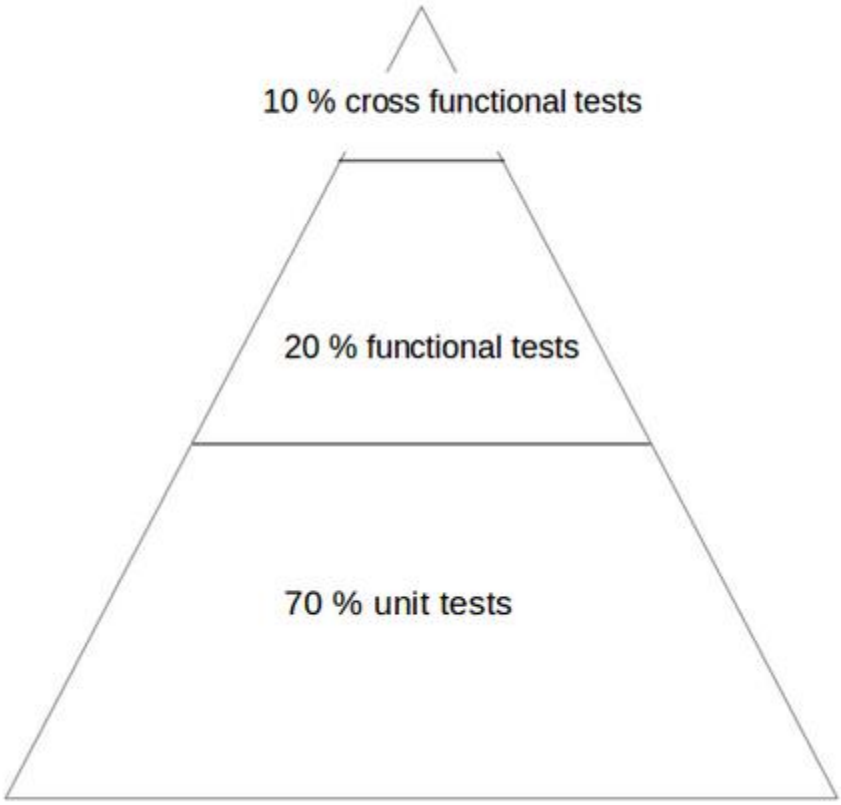


2-10-7-بخش هایی که در تست نرم افزاری، تمرکز بر روی آن قرار می گیرد

توصیه می شود در تست نرم افزار تمرکز خود را بیشتر بر روی آزمایش و کسب اطمینان از عملکرد صحیح منطق (logic) اپلیکیشن قرار دهید.

توصیه می شود تست اپلیکیشن خود را بر اساس قاعده ی زیر طراحی نمایید:

1. 70-80 % از کل تست را به unit test اختصاص دهید تا بدین وسیله stability و ثبات code base تضمین شود.
2. 20-30 % از تست را به صورت functional طراحی نموده و از کارکرد صحیح کد اطمینان کامل حاصل نمایید.
3. و در صورتی که اپلیکیشن شما با کامپوننت هایی از سایر اپلیکیشن ها زیاد تعامل دارد، بخشی از کل تست را به صورت cross functional تعبیه نمایید.



در تست اپلیکیشن باید به این امر نیز توجه داشته باشید که آیا تنها خود اپلیکیشن مورد آزمایش قرار می گیرد یا رابطه ی (integration) اپلیکیشن با اپلیکیشن های دیگر نیز مورد بررسی قرار گرفته و آزمایش می شود. در صورتی که تنها خود اپلیکیشن مورد آزمایش است، می توانید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

از فریم ورک های اجرای تست استفاده نمایید که تنها به دانش جزئی از اپلیکیشن ها برای مثال view ID ها احتیاج است.

3-7-10-پیش شرط های تست گیری (testing preconditions)

بد نیست در تست اپلیکیشن های اندرویدی خود متدی به نام testPreconditions() داشته باشید که پیش شرط تمامی تست ها را بررسی می کند. چنانچه این متد در انجام وظیفه ی خود موفق نباشد، بلافاصله پی می برید که فرض یا پیش شرط های دیگر تست ها نیز نقض شده است.

4-7-10-ATSL و تست اپلیکیشن با ابزار JUnit 4

Android Testing Support Library (ATSL) به شما اجازه می دهد تا اپلیکیشن خود را با یک test runner سازگار با JUnit 4 مورد آزمایش قرار دهید. می توانید unit test های اپلیکیشن خود را بر روی JVM و به صورت محلی اجرا نمایید یا آن را در بستر دستگاه واقعی اندروید (Android runtime) راه اندازی کنید. علاوه بر آن، Google یک فریم ورک اجرای تست بر روی UI به نام Espresso تعبیه نموده که به شما امکان توسعه ی تست های رابط کاربری را می دهد.

برای فراهم آوردن امکان تست نرم افزار در محیط دستگاه مجازی جاوا (JVM)، افزونه ی Gradle یک نسخه ی خاص از فایل android.jar را ارائه می نماید (این فایل تحت عنوان Android mockable jar نیز شناخته می شود). فایل نام برده در اختیار کل unit test قرار گرفته تا تمامی فیلدها، متدها و کلاس ها در دسترس و قابل استفاده باشند. لازم به ذکر است که فراخوانی هر یک از توابع mockable JAR، به صورت پیش فرض، منجر به صدور خطا (exception) می شود.

بنابراین اگر کلاس های شما توابع کتابخانه ای اندروید (Android API) را فراخوانی نمی کنند، شما می توانید از فریم ورک تست گیری JUnit (یا هر فریم ورک دیگری که برای تست اپلیکیشن های جاوا ارائه شده) بدون محدودیت استفاده نمایید. چنانچه به توابع کتابخانه ای اندروید (android API) وابستگی (dependency) دارید (مجبورید از کتابخانه ها و کلاس های Android API

استفاده نمایید)، در آن صورت این وابستگی ها در کد شما بایستی (برای unit test ها) از طریق فریم ورک های mocking (فریم ورک های ساختگی همچون Mockito) جایگزین و جبران گردد. برای مشاهده ی اطلاعات بیشتر جهت فعال سازی مقادیر بازگشتی برای متدهای ساختگی و شبیه سازی شده از فایل mockable.jar اندروید، به این آدرس مراجعه نمایید: Activating default return values for mocked methods in android.jar.

Android Testing Support Library این قابلیت را در اختیار توسعه دهنده قرار می دهد تا تست های نرم افزاری برای اپلیکیشن های خود طراحی کند. کتابخانه ی مذکور AndroidJUnitRunner، فریم ورک تست گیری Espresso و UI Automator را شامل می شود.

AndroidJUnitRunner امکان ساخت و اجرای تست های JUnit 4 را فراهم می کند، در حالی که چارچوب تست گیری Espresso برای طراحی تست ویژه ی لایه ی UI نرم افزار مناسب می باشد. با استفاده از UI Automator نیز برنامه نویس قادر خواهد بود تست های cross functional برای اپلیکیشن اندرویدی طراحی نماید.

AndroidJunitRunner توابع کتابخانه ای لازم برای اجرای تست نرم افزاری بر روی دستگاه حقیقی اندروید (Instrumentation API) را در قالب کلاس InstrumentationRegistry در اختیار توسعه دهنده قرار می دهد.

- InstrumentationRegistry.getInstrumentation() تابعی است که در خروجی instrumentation فعلی و در حال اجرا را برمی گرداند.
- InstrumentationRegistry.getContext() تابعی است که در خروجی Context پکیج instrumentation جاری را بازگردانی می نماید.
- InstrumentationRegistery.getTargetContext() متدی است که Context اپلیکیشن مقصد را بازگردانی می نماید.
- InstrumentationRegistry.getArguments()، یک کپی از مجموعه یا Bundle آرگومان هایی که به این instrumentation ارسال شده را در خروجی بازمی گرداند. این تابع زمانی

به کار شما می آید که بخواهید به آرگومان های command line ارسال شده به instrumentation برای تست خود استفاده نمایید.

برای دسترسی و مدیریت lifecycle نیز می توانید از کلاس ActivityLifecycleMonitorRegistry استفاده نمایید.

7-11-ساختار پروژه ی اندرویدی و ایجاد پوشه ی تست

7-11-1-سازماندهی پروژه ی اندرویدی برای تست

برای سازمان دهی پروژه های تست اپلیکیشن، توسعه دهندگان معمولا از قرار داد خاصی پیروی می کنند. جهت سازمان دهی کد اپلکیشن، در این پروژه های اندرویدی توصیه می شود که از ساختار عنوان شده در زیر استفاده نمایید. این ساختار را ویزارد پروژه ی اندروید نیز به صورت جداگانه ارائه می دهد.

اگر از این قرارداد پیروی کنید، بی شک سیستم کامپایل اندروید (Android build system) قادر خواهد بود تست شما را به صورت خودکار بر روی دستگاه مورد نظر (JVM یا دستگاه واقعی اندروید) اجرا کند.

7-11-2-برطرف کردن خطای "error duplicate files in path"

در صورت برخورد با این خطا:"error duplicate files in path. Path in archive: LICENSE.txt" ، می توانید با افزودن قطعه کد زیر به فایل app/gradle.build ، به راحتی خطای مزبور را برطرف نمایید.

```
android {
   packagingOptions {
   exclude 'LICENSE.txt'
   }
```

}

7-12-اجرای Unit test بر روی JVM

7-12-1-اجرای Unit test بر روی نرم افزار در بستر Android runtime

Android برای تست هایی که (به صورت local) در محیط JVM یا در بستر دستگاه واقعی اندروید (Android runtime) اجرا می شوند، واژه ی unit tests را بکار می برد.

Unit test عبارت است از تستی که عملکرد و قابلیت یک کامپوننت نرم افزاری را به صورت جداگانه مورد آزمایش قرار می دهد.

اپلیکیشنی را در نظر بگیرید که در آن دکمه ای داخل یک activity، سبب فعال سازی و اجرای activity دیگری می شود. unit test ای که برای اپلیکیشن نوشته می شود، بررسی می کند آیا intent مورد نیاز (برای راه اندازی activity دوم) صادر می شود یا خیر. Unit test کاری با اینکه آیا activity دوم در نتیجه ی فراخوانده شدن intent اجرا می شود یا خیر، ندارد.

Unit test ها بر اساس ورژن ویرایش شده از کتابخانه ی اندروید android.jar اجرا می شوند. در این ویرایش تمامی modifier های final حذف شده اند. این امر به شما امکان می دهد تا از کتابخانه های شبیه سازی و ایجاد آبجکت های ساختگی (mocking library) همچون Mockito بهره بگیرید.

7-12-2-محل جایگذاری unit test ها در پروژه ی اندرویدی

همان طور که قبلا ذکر شد، unit test های پروژه ی اندروید باید در پوشه ی app/src/test قرار داده شوند.

3-12-7-کتابخانه ها/dependency های الزامی در فایل Gradle build

برای استفاده و اجرای تست های JUnit بر روی کامپوننت های نرم افزاری مربوط به اپلیکیشن خود، بایستی آن ها را در قالب dependency (کتابخانه) به فایل Gradle build خود اضافه نمایید.

```
dependencies {
    // Unit testing dependencies
    testCompile 'junit:junit:4.12'
    // Set this dependency if you want to use the Hamcrest matcher library
    testCompile 'org.hamcrest:hamcrest-library:1.3'
    // more stuff, e.g., Mockito
}
```

4-12-7-اجرای unit test ها از طریق سیستم کامپایل Gradle

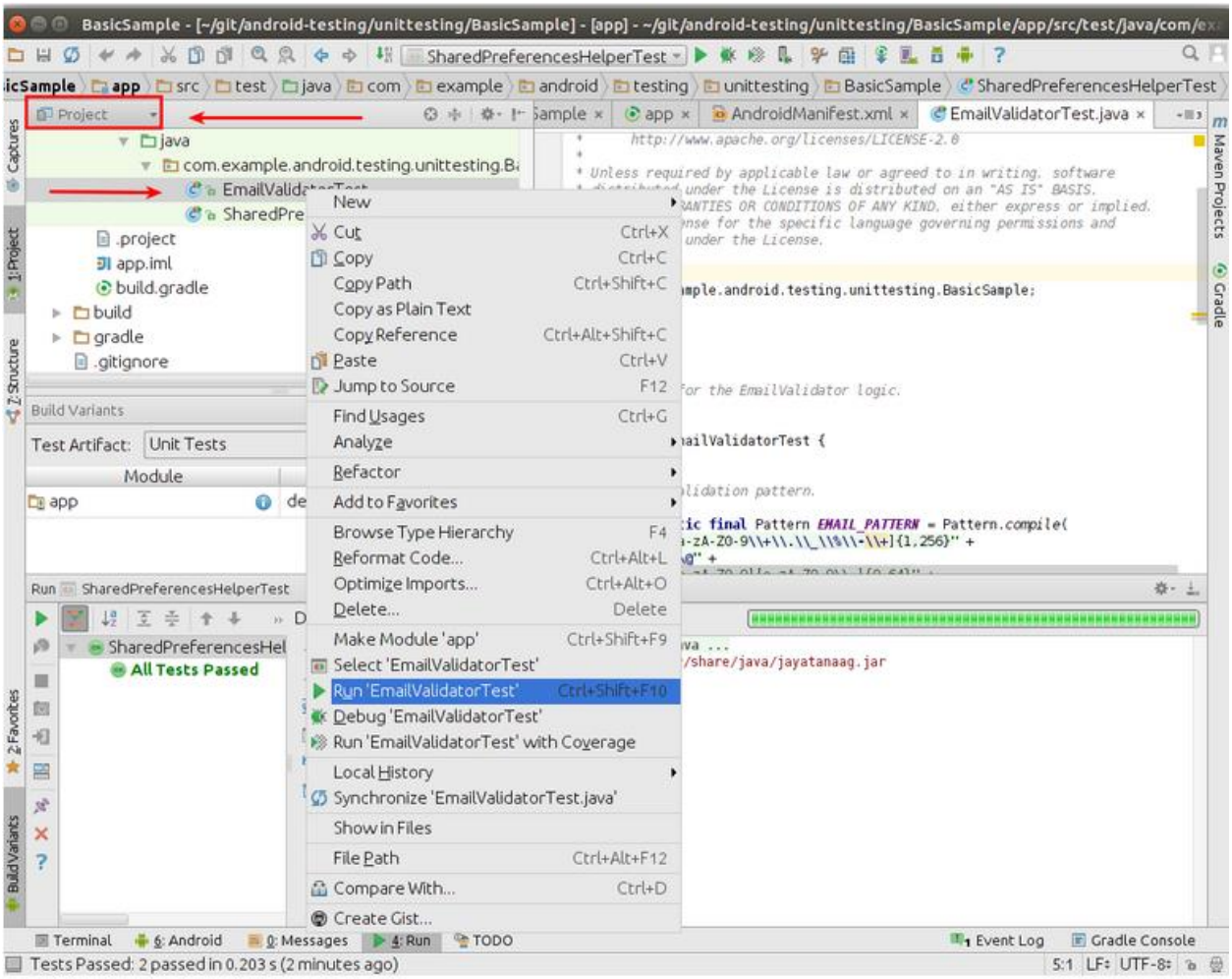
Unit test های خود را با فراخوانی دستور gradlew test اجرا نمایید.

5-12-7-اجرای unit test ها از محیط کاری Android Studio

به منظور اجرای یک unit test، بر روی کلاس تست گیری (test class) در پنجره ی Project راست کلیک کرده و سپس گزینه ی Run را انتخاب نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



6-12-7-محل قرارگیری نتایج و گزارش های مربوط به تست (test reports)

گزارش ها و نتایج مربوط به تست در پوشه ی app/build/reports/tests/debug/`directory ایجاد می شوند. فایل index.html` اطلاعاتی درباره ی هر یک از صفحات تست فراهم آورده و به آن ها جهت مشاهده ی اطلاعات بیشتر لینک می دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

7-12-7-فعال سازی مقادیر بازگشتی پیش فرض متدهای ساختگی (mocked methods) در فایل android.jar

به طور پیش فرض، فراخوانی هر یک از توابع موجود در فایل android.jar سبب رخداد خطا (exception) می شود. این وضعیت پیش فرض باعث می شود که unit test ها تنها کد شما را تست کنند و به هیچ یک از رفتار خاص محیط یا بستر اجرای اندروید (Android platform) وابسته نباشند. در صورت نیاز به تنظیم اختصاصی یک قابلیت یا رفتار خاص، کافی است از فریم ورک های شبیه ساز (mocking framework) جهت جایگزینی این متدها و فراخوانی آن ها استفاده نمایید.

همچنین می توانید به سیستم کامپایل Gradle دستور بدید که به عنوان خروجی متدهای فراخوانی شده، مقادیر پیش فرض مورد نظر را بازگردانی نمایند. برای این منظور کافی است تنظیمات زیر را در فایل Gradle build اپلیکیشن خود لحاظ نمایید.

```
android {
  // ...
  testOptions {
    unitTests.returnDefaultValues = true
  }
}
```

تمرین: طراحی unit test جهت تست اپلیکیشن

مرحله ی آماده سازی: ساخت پروژه ی اندروید

پروژه ی اندرویدی خود را بر اساس برنامه ای که در مباحث قبلی ساختید (Android temperature converter)، طراحی نمایید.

تمرین: ساخت unit test

هدف از انجام این تمرین

طی تمرین حاضر قادر خواهید بود یک تست JUnit4 برای پروژه ی اندرویدی خود بنویسید.

8-12-7-افزودن dependency / کتابخانه ی JUnit

Dependency یا کتابخانه ی Junit را در فایل app/build.gradle جایگذاری نمایید. اگر پوشه ی test در پروژه ی شما موجود نبود، در آن صورت بایستی طبق فرایندی که قبلا توضیح داده شد (نحوه ی ساخت پوشه ی تست در محیط کاری (Android Studio) را طی نموده و پوشه ی مورد نیاز را ایجاد نمایید.

```
dependencies {
    // Unit testing dependencies
    testCompile 'junit:junit:4.12'
}
```

ساخت و طراحی تست

در پوشه ی app/src/test، دو متد تست گیری (test method) زیر را داخل کلاس ConverterUtil تعریف نمایید.

```
package com.vogella.android.temperature.test;
import static org.junit.Assert.*;
import org.junit.After;
import org.junit.Before;
import org.junit.Test;
import com.vogella.android.temperature.ConverterUtil;
public class ConverterUtilTest {
    @Test
    public void testConvertFahrenheitToCelsius() {
        float actual = ConverterUtil.convertCelsiusToFahrenheit(100);
        // expected value is 212
        float expected = 212;
        // use this method because float is not precise
        assertEquals("Conversion from celsius to fahrenheit failed", expected,
                actual, 0.001);
    }
    @Test
    public void testConvertCelsiusToFahrenheit() {
        float actual = ConverterUtil.convertFahrenheitToCelsius(212);
        // expected value is 100
        float expected = 100;
```

```
        // use this method because float is not precise
        assertEquals("Conversion from celsius to fahrenheit failed", expected,
                actual, 0.001);
    }
}
```

9-12-7-اجرای unit test بر روی پروژه

بد نیست با تست unit test هایی که برای بخش های مختلف پروژه ی خود نوشته اید، اطمینان حاصل نمایید که تست ها به درستی پیاده سازی شده اند. تست های شما بایستی طبق انتظار به درستی اجرا شوند.

7-13-طراحی instrumentation test برای اجرای تست بر روی اپلیکیشن در بستر دستگاه حقیقی اندروید

1-13-7-Instrumentation test

API یا توابع کتابخانه ای تست گیری اندروید hook هایی را برای وصل شدن به کامپوننت های نرم افزاری و lifecycle اپلیکیشن اندروید فراهم می آورند (hook مکان یا interface ای است که در قالب کد پکیج شده ارائه شده و به برنامه نویس اجازه می دهد تا کدهایی با تنظیمات اختصاصی و دلخواه وارد متن برنامه کند. به طور مثال، توسعه دهنده می تواند کدی وارد برنامه کند که وظیفه ی آن بررسی این است که هر چند وقت یکبار یک مسیر منطقی داخل اپلیکیشن مورد نظر طی می شود).

این hook ها در اصطلاح instrumentation API خوانده شده و به تست های شما این اجازه را می دهد تا اداره ی lifecycle و event های مرتبط با تعامل کاربر را بدست بگیرند.

در شرایط معمولی، اپلیکیشن نمی تواند event های مربوط به lifecycle را کنترل کند و در این برهه کاربر مدیریت جریان یا روند اپلیکیشن را بدست می گیرد. برای مثال، زمانی که سیستم اندروید کامپوننت activity اپلیکیشن را اجرا می کند، در واقع متد ()onCreate فراخوانده می شود. یا هنگامی که کاربر دکمه ای را در نمایشگر فشار می دهد، کد مربوطه از اپلیکیشن فراخوانده و اجرا می شود. توسعه دهنده می تواند با بهره گیری از توابع کتابخانه ای Instrumentation، رخدادهایی از این دست را با اجرای کد آزمایشی (test code) خود اداره کند. سپس می تواند متد ()finish را صدا زده، activity را مجددا اجرا کند و بدین وسیله مطمئن شود که اطلاعات مربوط به وضعیت (instance state) activity مزبور پس از راه اندازی مجدد بازگردانی می شوند یا خیر.

Unit test هایی که زیرمجموعه ی instrumentation هستند، همان طور که پیش تر نیز ذکر شد، بجای JVM (دستگاه مجازی جاوا)، بر روی دستگاه های حقیقی اندروید یا در بستر نرم افزار شبیه ساز (emulator) اجرا می شوند. این تست ها به دستگاه واقعی اندروید و منابع آن دسترسی داشته و برای اجرای تست بر روی کامپوننت های مجزای نرم افزار و انجام unit test بسیار ساده می باشند. همان طور که می دانید، تست قابلیت های اپلیکیشن به واسطه ی unit test با بهره گیری از فریم ورک های شبیه سازی (mocking frameworks) به راحتی امکان پذیر نمی باشد. به عنوان مثال می توان به تستی اشاره کرد که کد و پیاده سازی یک Parcelable را آزمایش کرده و سعی می کند از عملکرد صحیح آن اطمینان حاصل نماید.

کلاس تست گیری که مبتنی بر instrumentation می باشد، این امکان را به شما می دهد تا event های اصلی (event های مربوط به نمایشگر/touch event) را به اپلیکیشن مورد تست ارسال نمایید.

با وجود فریم ورک های تست گیری همچون Espresso که قابلیت آزمایش UI را فراهم می آورند، توسعه دهنده به ندرت مجبور می شود تا مستقیما از instrumentation API استفاده نماید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-13-7-سیستم اندروید چگونه تست ها را اجرا می کند

Test runner (کلاس تست گیری) پایه برای اجرای تست های نرم افزاری بر روی اپلیکیشن های اندرویدی، InstrumentationTestRunner است. این کلاس تمامی متدهای تست گیری را اجرا کرده و در حافظه بارگذاری می نماید. سپس به واسطه ی instrumentation API با سیستم اندروید ارتباط برقرار می کند. زمانی که تستی را جهت آزمایش اپلیکیشن اندرویدی خود اجرا می کنید، سیستم اندروید تمامی فرایندهای مربوط اپلیکیشن مورد آزمایش (که در حال تست شدن هستند) را از حافظه پاک کرده و متعاقبا یک نمونه (instance) جدید از آن را در حافظه بارگذاری می کند. لازم به توضیح است که کلاس مزبور اپلیکیشن را راه اندازی نمی کند، بلکه این امر، وظیفه ی متدهای تست گیری می باشد. به عبارت دیگر این متد تست گیری است که اداره ی lifecycle کامپوننت های نرم افزاری را بدست می گیرد.

کلاس تست گیری مورد نظر، به هنگام راه اندازی و مقداردهی اولیه (initialization)، همچنین متد onCreate() مربوط به اپلیکیشن و activity مورد تست را صدا می زند.

3-13-7-شبیه سازی رفتار آبجکت ها در اندروید (ایجاد آبجکت های ساختگی جهت تست)

توسعه دهنده برای اجرای تست های instrumentation بر روی نرم افزار اندرویدی خود، همچنین می تواند از فریم ورک شبیه سازی (mocking framework) Mockito استفاده نماید. با بهره گیری از این فریم ورک توسعه قادر خواهد بود آن بخش هایی از سیستم اندروید را که تست آن ها ضرورتی ندارد، با بخش هایی از این فریم ورک جایگزین نماید. ویرایش های قبلی چارچوب نرم افزاری اندروید (Android framework) کلاس های شبیه سازی / mocking classes را ویژه ی تست بخش های اپلیکیشن ارائه می دادند. اما امروزه با وجود Mockito این کلاس ها دیگر کاربردی ندارند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

4-13-7-محل قرارگیری تست های instrumentation

همان طور که قبلا توضیح داده شد (در بخش سازماندهی پروژه و تست ها)، unit test ها بایستی در پوشه ی app/src/androidTest/java قرار گیرد.

5-13-7-تعریف dependency ها و testInstrumentationRunner داخل فایلGradle build

جهت استفاده از JUnit برای اپلیکیشن های اندرویدی خود، می بایست dependency (کتابخانه ی مورد نیاز) را به فایل Gradle build اضافه نمایید. علاوه بر آن لازم است android.support.test.runner.AndroidJUnitRunner را نیز در قالب کلاس testInstrumentationRunner داخل فایل build تعریف نمایید.

```
defaultConfig {
    ..... more stuff
    testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
  }
dependencies {
  // Unit testing dependencies
  androidTestCompile 'junit:junit:4.12'
  // Set this dependency if you want to use the Hamcrest matcher library
  androidTestCompile 'org.hamcrest:hamcrest-library:1.3'
  // more stuff, e.g., Mockito
}
```

6-13-7-استفاده از @RunWith(AndroidJUnit4.class)

توصیه می شود تست را با دستور @RunWith(AndroidJUnit4.class) نیز نشانه گذاری (annotate) نمایید. AndroidJUnit4 از JUnit4 ارث بری می کند (extend می کند). بنابراین اگر از syntax/ساختار نگارشی خالص junit4 و ActivityTestRule استفاده نمایید، دیگر لزومی ندارد تست را با annotation فوق نشانه گذاری نمایید. اما در صورتی که لازم باشد تست های فریم ورک Espresso را همراه با ActivityTestRule و JUnit4 اجرا نمایید، استفاده از آن الزامی خواهد بود.

7-13-7-اجرای unit test از طریق سیستم کامپایل Gradle

Unit test را با فراخوانی دستور gradlew connectedCheck اجرا نمایید.

7-13-8-اجرای unit test از داخل محیط برنامه نویسی Android Studio

بر روی کلاس تست گیری در پنجره ی Project راست کلیک کرده و سپس گزینه ی Run را انتخاب نمایید.

7-13-9-محل جایگذاری گزارش های تست

گزارش های تست در پوشه ی /app/build/reports/androidTests/connected جایگذاری می شود.

فایل index.html اطلاعات مختصری درباره ی تست ارائه داده و به تمامی صفحات آن لینک می دهند.

7-13-10-نحوه ی جایگزین کردن بخش های اپلیکیشن با instrumentation test ها

می توانید برای اجرای تست های مبتنی بر instrumentation و جایگزین کردن کلاس اپلیکیشن (با تست های instrumentation)، کلاس AndroidJUnitRunner و newApplication را بازنویسی (override) نمایید.

```
package com.vogella.android.daggerjunitmockito;
import android.app.Application;
public class MyMockApplication extends Application {
   @Override
   public void onCreate() {
     // do something important for your tests here
   }
}
```

Test runner:

```
package com.vogella.android.daggerjunitmockito;
import android.app.Application;
import android.content.Context;
import android.support.test.runner.AndroidJUnitRunner;
public class MockTestRunner extends AndroidJUnitRunner {
  @Override
  public Application newApplication(ClassLoader cl, String className, Context context)
      throws InstantiationException, IllegalAccessException, ClassNotFoundException {
    return super.newApplication(cl, MyMockApplication.class.getName(), context);
  }
}
```

همچنین لازم است این test runner را در فایل build.gradle ثبت (تعریف) نمایید.

```
android {
    /// more
      testInstrumentationRunner "com.vogella.android.daggerjunitmockito.MockTestRunner"
    }
    /// more
}
```

تمرین: شبیه سازی و کسب اطمینان از عملکرد صحیح قابلیت دسترسی به فایل (Mocking file access)

هدف از این تمرین

لازم نیست این کلاس را در اپلیکیشن اندرویدی خود بکار ببرید و در اینجا تنها به منظور نمایش نحوه ی استفاده از Mockito در unit test بکار برده شده است.

11-13-7-ایجاد کلاسی جهت تست گیری

در اپلیکیشن اندرویدی آماده ی خود و یا پروژه ای با پکیج com.vogella.android.testing.mockitocontextmock، کلاس زیر را پیاده سازی نمایید.

```
public class Util {
      public static void writeConfiguration(Context ctx ) {
            try (FileOutputStream openFileOutput =
                  ctx.openFileOutput( "config.txt", Context.MODE_PRIVATE);) {
                openFileOutput.write("This is a test1.".getBytes());
                openFileOutput.write("This is a test2.".getBytes());
            } catch (Exception e) {
```

```
                // not handled
            }
        }
    }
```

12-13-7-ساخت unit test جدید

با استفاده از فریم ورک شبیه سازی (mocking framework) Mockito، یک unit test جدید بنویسید که موارد زیر را بررسی می کند:

1. openFileOutput دقیقا یکبار فراخوانی شود.
2. متد ()write حداقل دوبار صدا زده شود.

```
package com.vogella.android.testing.mockitocontextmock;
import android.content.Context;
import org.junit.Before;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.mockito.Mock;
import org.mockito.MockitoAnnotations;
import java.io.FileOutputStream;
import static org.junit.Assert.fail;
import static org.mockito.Matchers.any;
import static org.mockito.Matchers.anyInt;
import static org.mockito.Matchers.anyString;
import static org.mockito.Mockito.atLeast;
import static org.mockito.Mockito.times;
import static org.mockito.Mockito.verify;
import static org.mockito.Mockito.when;
public class TextContextOutputStream {
@Mock
Context context;
@Mock
FileOutputStream fileOutputStream;
@Before
public void init(){
   MockitoAnnotations.initMocks(this);
}
   @Test
public void writeShouldWriteTwiceToFileSystem() {
   try {
      when(context.openFileOutput(anyString(), anyInt())).thenReturn(fileOutputStream);
      Util.writeConfiguration(context);
```

```
        verify(context, times(1)).openFileOutput(anyString(), anyInt());
        verify(fileOutputStream, atLeast(2)).write(any(byte[].class));
    } catch (Exception e) {
        e.printStackTrace();
        fail();
    }
}
}
```

14-7-اطلاعات بیشتر در خصوص اجرای تست بر روی اپلیکیشن های اندرویدی

14-7-1-کلاس های assert

توابع کتابخانه ای اجرای تست (Android testing API) بر روی اپلیکیشن های اندرویدی، علاوه بر JUnit Assert، دو کلاس دیگر به نام های MoreAsserts و ViewAsserts ارائه می دهد.

14-7-2-Test group ها (گروه بندی تست ها)

دستورات @SmallTest، @MediumTest` و @LargeTest`annotations` امکان گروه بندی تست ها را برای توسعه دهنده فراهم می کنند. به عبارتی روشن تر این قابلیت توسعه دهنده را قادر می سازد تا تست ها را بر اساس ویژگی خاصی گروه بندی نماید. برای مثال تنها آن دسته از تست هایی را بر روی نرم افزار اجرا کند که زمان اجرای آن ها چندان طولانی نیست یا تست هایی که زمان اجرای آن ها طولانی هستند را بر روی سرویس دهنده ی continuous integration به اجرا بگذارند (Continuous integration به یک سری روش گفته می شود که طی آن سلامت نرم افزار به طور دایم تضمین می شود، به طوری که خورده تغییرات افرادِ درگیر در پروژه باعث رخداد خطا در کل پروژه نشده و به درستی مثل تکه های پازل در کنار هم قرار گیرند).

برای اینکه تنها تست های انتخابی اجرا شوند، شما می توانید کلاس InstrumentationTestRunner را به واسطه ی افزونه ی Gradle (plug-in) تنظیم نمایید. کد

زیر نمونه ای از فایل build.gradle را نمایش می دهد که با دستور @SmallTests علامت گذاری شده و به همین خاطر تنها تست هایی که زمان اجرای طولانی ندارند را اجرا می کند.

```
android {
  //....
  defaultConfig {
  //....
    testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    testInstrumentationRunnerArgument "size", "small"
  }
}
import android.test.suitebuilder.annotation.MediumTest;
import android.test.suitebuilder.annotation.SmallTest;
import org.junit.Test;
public class ExampleTest {
    @Test
    @SmallTest
    public void validateSecondActivity() {
       // Do something not so long...
    }
    @Test
    @MediumTest
    public void validateSecondActivityAgain() {
       // Do something which takes more time....
    }
}
```

3-14-7-فیلتر کردن تست

می توانید تست های نرم افرازی خود را با annotation ها نشانه گذاری کرده و از این طریق تست های مزبور را فیلتر نمایید (برای مثال شرایطی که تست در آن اجرا می شود را دقیق مشخص کنید). در واقع کلاس اجرای تست های نرم افزاری اندروید (android test runner) به شما امکان می دهد این تست ها را به وسیله ی annotation های زیر، محدود ساخته و صریحا مشخص کنید که تست با چه شرایطی بایستی اجرا شود.

Annotation های فیلتر و محدود سازی تست	
Annotation	شرح کاربرد
@RequiresDevice	تعیین می نماید که تست تنها بایستی بر روی دستگاه های واقعی اندروید اجرا شود. چنانچه محیط اجرای تست شبیه ساز باشد، تست به واسطه ی این annotation اجرا نخواهد شد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

@SdkSupress	@SDKSupress(minSdkVersion=18)

4-14-7-دستور @FlakyTest

می توانید با استفاده از دستور @FlakyTest به سیستم اندروید اعلان نمایید که تست را در صورت عدم موفقیت، بار دیگر اجرا کند. Annotation ذکر شده، یک attribute به نام tolerance دارد که با مقدار دهی آن می توانید تعیین نمایید که تست پس از چندبار ناموفق بودن در اجرا، failed در نظر گرفته شده و دیگر تکرار نشود.

تمرین: تست lifecycle

هدف از این تمرین

در این تمرین، استفاده از فریم ورک تست گیری نرم افزار در بستر دستگاه حقیقی اندروید (instrumentation test) به صورت عملی نمایش داده شده است. البته برای چنین تستی، بهتر است از فریم ورک های سطح بالاتر همچون Espresso استفاده نمایید.

ایجاد پروژه و تست آن

یک پروژه و activity جدید به ترتیب به نام های com.vogella.android.test.simpleactivity و MainActivity تعریف نمایید.

حال activity دیگری به نام SecondActivity تعریف نموده و به پروژه ی نام برده اضافه نمایید. این activity بایستی از یک layout استفاده نموده و حداقل یک آبجکت TextView دربرداشته باشد. id این آبجکت بایستی بر روی "resultText" تنظیم شده و text آن با "Started" مقداردهی شده باشد.

یک فیلد EditText به فایل layout کلاس MainActivity اضافه نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

یک آبجکت دکمه به فایل layout ای که مورد استفاده ی کلاس MainActivity می باشد، اضافه نمایید. زمانی که دکمه کلیک می شود، دومین activity بایستی اجرا شود.

فیلد متنی EditText را با استفاده از "text" به عنوان کلید آن، داخل آبجکت intent قرار دهید. پس از آن رشته ی http://www.vogella.com را به عنوان extra (اطلاعات اضافی که بین دو activity مبادله

می شود) با انتخاب "URL" به عنوان کلید، (به وسیله ی دستور intent.putExtra("URL", "http://www.vogella.com");) داخل آبجکت intent قرار دهید.

در زیر نمونه کدی از پیاده سازی MainActivity را مشاهده می کنید.

```
package testing.android.vogella.com.simpleactivity;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
public class MainActivity extends Activity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
  }
  public void onClick(View view) {
    Intent intent = new Intent(this, SecondActivity.class);
    intent.putExtra("URL", "http://www.vogella.com");
    startActivity(intent);
  }
}
```

5-14-7-ساخت کلاس و پروژه ی تست

یک instrumentation test دیگر مشابه زیر پیاده سازی نمایید.

```
package com.vogella.android.test.simpleactivity.test;
import android.app.Activity;
import android.app.Instrumentation;
import android.app.Instrumentation.ActivityMonitor;
import android.test.ActivityInstrumentationTestCase2;
import android.test.TouchUtils;
import android.test.UiThreadTest;
import android.test.ViewAsserts;
```

```
import android.view.KeyEvent;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import com.vogella.android.test.simpleactivity.R;
import com.vogella.android.test.simpleactivity.MainActivity;
import com.vogella.android.test.simpleactivity.SecondActivity;
public class SecondActivityFunctionalTest extends
        ActivityInstrumentationTestCase2<SecondActivity> {
    private static final String NEW_TEXT = "new text";
    public SecondActivityFunctionalTest() {
        super(SecondActivity.class);
    }
    public void testSetText() throws Exception {
        SecondActivity activity = getActivity();
        // search for the textView
        final TextView textView = (TextView) activity
                .findViewById(R.id.resultText);
        // set text
        getActivity().runOnUiThread(new Runnable() {
            @Override
            public void run() {
                textView.setText(NEW_TEXT);
            }
        });
        getInstrumentation().waitForIdleSync();
        assertEquals("Text incorrect", NEW_TEXT, textView.getText().toString());
    }
    @UiThreadTest
    public void testSetTextWithAnnotation() throws Exception {
        SecondActivity activity = getActivity();
        // search for the textView
        final TextView textView = (TextView) activity
                .findViewById(R.id.resultText);
        textView.setText(NEW_TEXT);
        assertEquals("Text incorrect", NEW_TEXT, textView.getText().toString());
    }
}
```

15-7-استفاده از ابزار Monkey جهت ایجاد و ارسال شبه event به دستگاه

1-15-7-توضیحی درباره ی ابزار monkey

Monkey یک ابزار command-line است که بر روی محیط شبیه ساز یا دستگاه حقیقی اندروید اجرا شده و این قابلیت را به شما می دهد تا علاوه بر مجموعه ای از event هایی که توسط کاربر فرخوانی می شوند، تعدادی از event هایی که در سطح سیستم رخ می دهند را نیز به صورت

شبه تصادفی ایجاد و به سیستم ارسال نمایید. در واقع شما با این ابزار قادر خواهید بود اپلیکیشن های در حال توسعه ی خود را به صورت تصادفی اما تکرار پذیر، تحت شرایط سنگین آزمایش نموده و مطمئن شوید که اپلیکیشن مورد نظر در صورت وجود بار کاری زیاد نیز دارای قابلیت دسترسی و مدیریت خطای بالایی می باشد (stress-test).

شما می توانید ابزار مزبور را طوری تنظیم کنید که تنها برای پکیج خاصی اجرا شده و صرفا اپلیکیشن مد نظر را تست کند.

2-15-7-نحوه ی استفاده از ابزار Monkey

دستور زیر 2000 رخداد تصادفی به اپلیکیشنی که اسم پکیج de.vogella.android.test.target را در بالای فایل خود دارند، ارسال می کند.

```
adb shell monkey -p de.vogella.android.test.target -v 2000
```

لازم به ذکر است که Monkey در برخی شرایط مشکلاتی را برای سرویس دهنده ی adb ایجاد می کند. در چنین موقعیتی می توانید با اجرای دو دستور زیر، سرویس دهنده ی adb را مجددا راه اندازی نموده و مشکل را برطرف کنید.

```
adb kill-server
adb start-server
```

برای اینکه مطمئن شوید توالی یا مجموعه event های تولید شده همیشه یکسان می باشد، باید پارامتر [seed] s- را به دستور اضافه نمایید.

16-7-تست آبجکت Application

در اندروید، منطق اپلیکیشن، داده ها و تنظیمات مربوط به کل برنامه داخل آبجکتی به نام application جای گرفته است. لازم است این آبجکت را تست کرده و از عملکرد صحیح آن اطمینان حاصل نمایید.

می توانید یک تست JUnit 4 برای آزمایش عملکرد اپلیکیشن نوشته و آن را بر روی JVM اجرا نمایید. در این سناریو بایستی تمامی dependency و کتابخانه ی های مربوط به آبجکت اپلیکیشن را شبیه سازی (mock) نمایید.

برای تست آبجکت application در runtime اندروید، می بایست از کلاس ApplicationTestCase استفاده نمایید. در آینده انتظار می رود که شرکت گوگل یک rule جدید برای تست آبجکت یاد شده، در کتابخانه ی JUnit4 تعبیه نماید، اما در زمان حاضر چنین امکانی برای استفاده ی توسعه دهنده فراهم نیست.

test runner / کلاس اجرای تست محیط Android (InstrumentationTestRunner) در مرحله ی مقداردهی اولیه، به صورت خودکار یک نمونه از کلاس application می سازد. بنابراین در صورت استفاده از پردازش ناهمزمان در بدنه ی متد ()onCreate، لازم است حتما این نکته را به خاطر داشته باشید.

تمرین: تست اپلیکیشن

ایجاد پروژه

یک اپلیکیشن جدید اندروید با اسم پکیج com.vogella.android.testing.applicationtest بر اساس قالب آماده ی Empty Activity (template) ایجاد نمایید.

آبجکت application را با پیاده سازی زیر به برنامه ی خود اضافه نمایید.

```
package com.vogella.android.testing.applicationtest;
import android.app.Application;
import java.util.ArrayList;
import java.util.List;
public class MyApplication extends Application {
   public static final List<String> list = new ArrayList<String>();
}
```

لازم است اپلیکیشن را در لایه ی XML، داخل فایل manifest نیز اعلان نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.vogella.android.testing.applicationtest" >
   <application
      android:name=".MyApplication"
      android:allowBackup="true"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >
      <activity
         android:name=".MainActivity"
         android:label="@string/app_name" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
   </application>
</manifest>
```

1-16-7-تعریف unit test برای تست آبجکت application

داخل پوشه ی app/src/test یک unit test جدید تعریف نمایید. تست را طوری بنویسید که انتظار داشته باشد (assert) فیلد MyApplication.list تهی نبوده و اندازه ی (size) اولیه ی آن 0 باشد.

2-16-7-طراحی instrumentation test برای آبجکت application

یک unit test جدید بر اساس فریم ورک تست گیری JUnit 3 به صورت زیر تعبیه نمایید.

```
package com.vogella.android.testing.applicationtest;
import android.content.pm.PackageInfo;
import android.test.ApplicationTestCase;
import android.test.MoreAsserts;
public class ApplicationTest extends ApplicationTestCase<MyApplication> {
   private MyApplication application;
   public ApplicationTest() {
      super(MyApplication.class);
   }
   protected void setUp() throws Exception {
      super.setUp();
      createApplication();
      application = getApplication();
   }
   public void testCorrectVersion() throws Exception {
      PackageInfo info = application.getPackageManager().getPackageInfo(application.getPackageName(), 0);
      assertNotNull(info);
      MoreAsserts.assertMatchesRegex("\\d\\.\\d", info.versionName);
   }
}
```

7-17-تست سایر کامپوننت های نرم افزاری اندروید

1-17-7-تست سرویس

جهت تست یک سرویس، بایستی از کلاس ServiceTestRule که از Testing Support Library اندروید مشتق می شود، استفاده نمایید. در ویرایش های قبلی به منظور آزمایش کامپوننت سرویس از ServiceTestCase استفاده می شد که با ارائه ی کلاس نام برده، استفاده از آن منسوخ شد.

این rule یک مکانیزم ساده در اختیار توسعه دهنده قرار می دهد که سرویس را قبل از اجرای تست راه اندازی نموده و با اتمام آن، کامپوننت مورد نظر را متوقف می سازد. در واقع این rule تضمین می کند که سرویس به هنگام اجرا به طور کامل متصل شده باشد. می توانید سرویس را به واسطه ی یکی از توابع کمکی (helper method) فراخوانی نمایید. زمانی که تست پایان می یابد، سرویس به صورت خودکار متوقف شده و به همراه آن تمامی متدهایی که با @After نشانه گذاری شده اند نیز خاتمه می یابند.

نکته: این rule همراه با IntentService قابل استفاده نمی باشد چرا که به مجرد اتمام اجرای متد onHandleIntent، به طور خودکار از حافظه پاک می گردد.

کد زیر یک سرویس را تست می کند.

```
@RunWith(AndroidJUnit4.class)
@MediumTest
public class MyServiceTest {
   @Rule
   public final ServiceTestRule mServiceRule = new ServiceTestRule();
   // test for a service which is started with startService
   @Test
   public void testWithStartedService() {
      mServiceRule.
      startService(new Intent(InstrumentationRegistry.getTargetContext(),
                  MyService.class));
      // test code
   }
   @Test
 // test for a service which is started with bindService
   public void testWithBoundService() {
```

```
        IBinder binder = mServiceRule.
                bindService(new Intent(InstrumentationRegistry.getTargetContext(),
                        MyService.class));
        MyService service = ((MyService.LocalBinder) binder).getService();
        assertTrue("True wasn't returned", service.doSomethingToReturnTrue());
    }
}
```

```
adb kill-server
adb start-server
```

2-17-7-تست کامپوننت نرم افزاری Content provider

به منظور تست content provider، توسعه دهنده می تواند از کلاس ProviderTestCase2 استفاده کند. این کلاس به صورت خودکار از کامپوننت (content provider) مورد تست، نمونه سازی کرده و آبجکت IsolatedContext را در آن درج می نماید. گرچه این context یا بستر از سیستم اندروید مجزا است، اما همچنان اجازه ی دسترسی به فایل و دیتابیس را فراهم می کند. در واقع استفاده از آبجکت IsolatedContext این اطمینان را به وجود می آورد که provider مورد تست، تاثیری بر روی دستگاه واقعی اندروید نمی گذارد.

کلاس نام برده همچنین با ارائه ی متدی به نام getMockContentResolver() امکان دسترسی به کلاس MockContentResolver را فراهم می سازد.

شما بایستی تمامی عملیات مربوط به provider را آزموده و همچنین بررسی نمایید که در صورت فراخوانی provider با URI یا projection غیرمجاز چه اتفاقی رخ می دهد.

3-17-7-تست Loader

جهت تست کارکرد یک loader، توسعه دهنده می بایست از کلاس `LoaderTestCase` استفاده نماید. البته در آینده ی نزدیک انتظار می رود که شرکت گوگل با معرفی یک rule جدید در کتابخانه ی JUnit4، کلاس نام برده را منسوخ نماید.

Annotation-18-7 های متفرقه یا پشتیبان

علاوه بر annotation های نام برده، شرکت گوگل یک پکیج حاوی annotation های جدید در اختیار توسعه دهنده قرار می دهد. پکیج مزبور تعدادی metadata annotation (دستوراتی که اطلاعاتی را درباره ی عملکرد کد شرح می دهند) ارائه می دهد که با استفاده از آن ها در متن برنامه، توسعه دهنده قادر است از رخداد خطا در عملکرد اپلیکیشن جلوگیری نماید.

جهت استفاده از این annotation ها، کافی است کتابخانه ی (dependency) زیر را به فایل Gradle build خود اضافه نمایید.

```
dependencies {
    compile 'com.android.support:support-annotations:22.2.0'
}
```

جهت دریافت اطلاعات بیشتر درباره ی annotation ها می توانید به آدرس http://tools.android.com/tech-docs/support-annotations مراجعه نمایید.

Annotation_7_18_1 های مربوط به Nullness

با درج دستور @Nullable در بالای کد توسعه دهنده به سیستم اعلان می کند که یک پارامتر یا مقدار بازگشتی می تواند null باشد. به طور مشابه، دستور @NonNull در بالای کد مربوطه نشانگر این است که یک پارامتر (یا مقدار بازگشتی) نمی تواند و نباید null باشد.

Annotation_7_18_2 های مربوطه به thread

در صورتی که یک متد تنها از thread خاصی قابل فراخوانی باشد، می توانید آن را با یکی از annotation های زیر نشانه گذاری نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

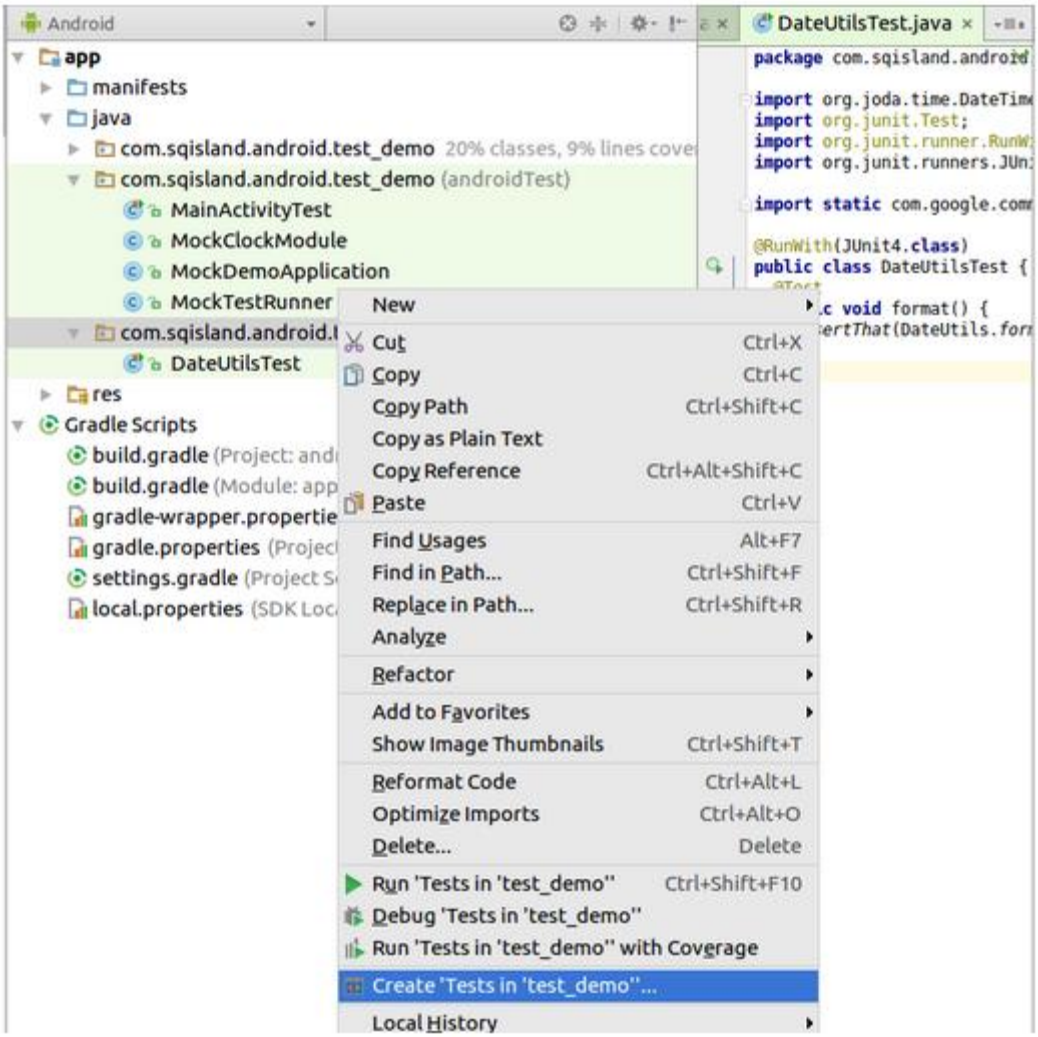
- @UiThread
- @MainThread
- @WorkerThread
- @BinderThread

مثال:

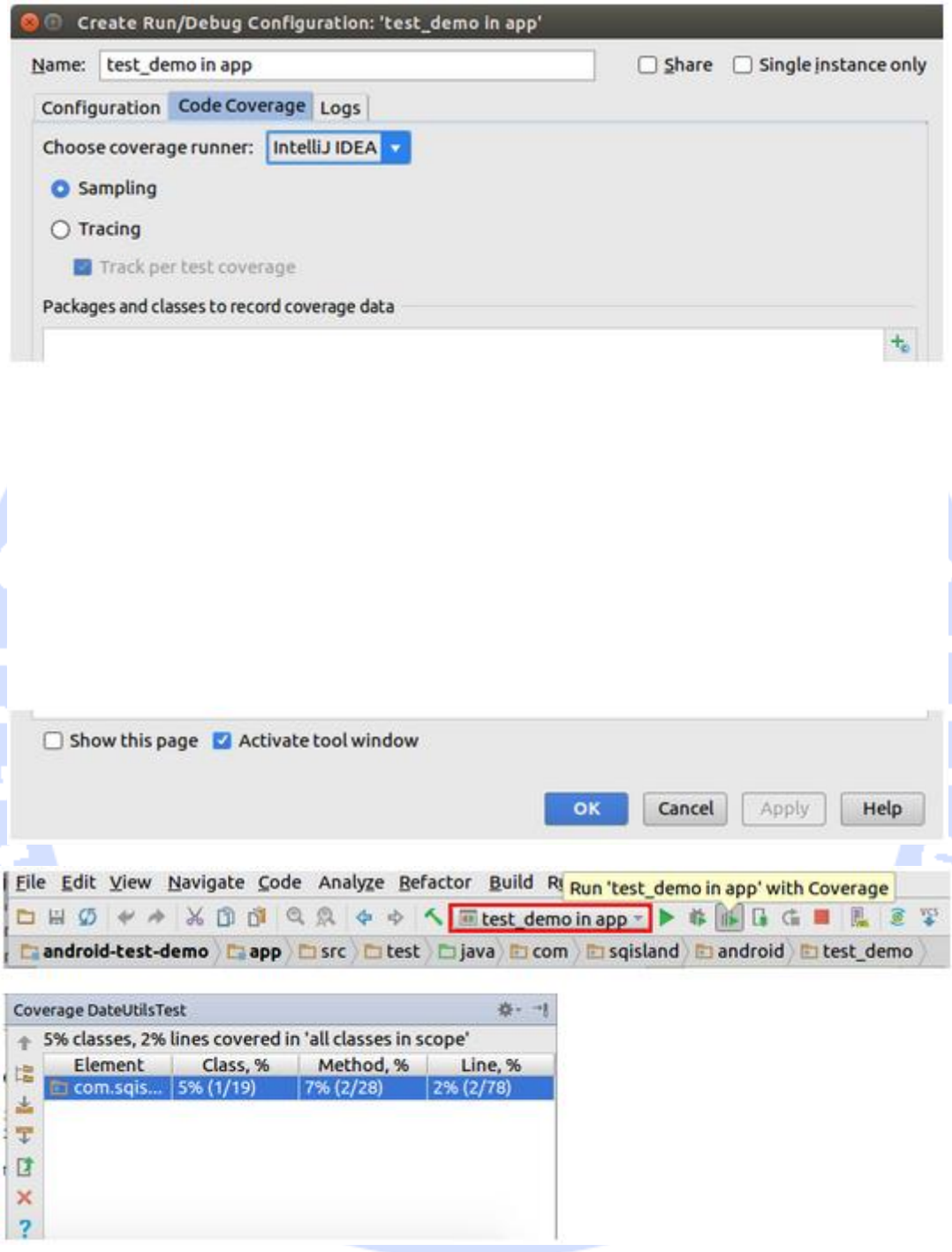
```
@WorkerThread
protected abstract Result doInBackground(Params... params);
@MainThread
  protected void onProgressUpdate(Progress... values) {
  }
```

3-18-7-تهیه ی گزارش از مقدار کد تست شده از کل پروژه (Code coverage report)

Code coverage report بیانگر این است چه میزان از کل کد اپلیکیشن شما توسط unit test مورد آزمایش قرار گرفته است. به منظور تهیه ی چنین گزارشی، می توانید تنظیمات جداگانه ای برای اجرا (launch config) آن تعبیه نمایید. برای نیل به این هدف، پکیج مورد نظر خود را انتخاب نموده و پس از باز نمودن منو، آیتم [Create Tests in...] را انتخاب نمایید.



حال می توانید تنظیمات runtime (config) جدید برای code coverage یا میزان کد مورد آزمایش خود ایجاد نمایید. پس از اجرا، یک گزارش تهیه شده و به نمایش گذاشته می شود.

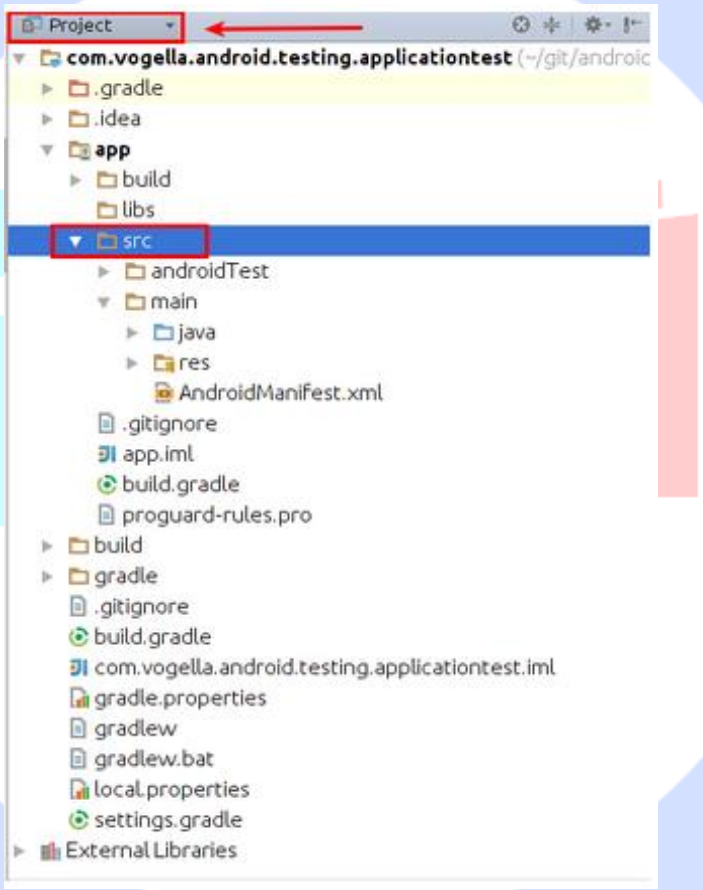


19-7-ساخت پوشه ی تست در محیط کاری Android Studio

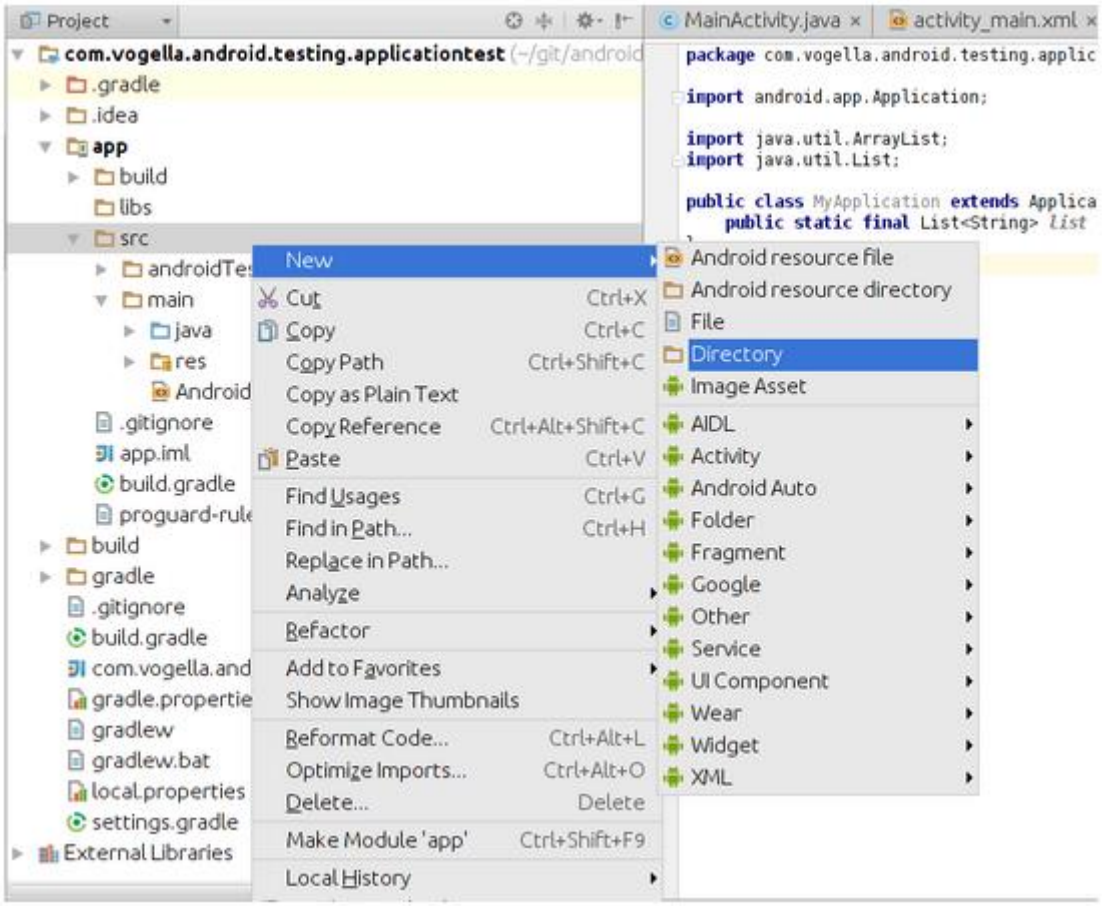
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

ویرایش های جدید محیط برنامه نویسی Android Studio، یک پوشه ی تست (test folder) به قالب آماده ی پروژه ی پیش فرض (project template) خود اضافه کرده است. در صورت استفاده از قالبی که قابلیت ایجاد پوشه ی تست را به صورت درون ساخته ندارد، بایستی این پوشه را خود به صورت دستی ایجاد نمایید. جهت ایجاد پوشه ی تست در محیط کاری مذکور، کادر Project را باز نمایید. در این view می توانید ساختار درختی و directory structure پروژه ی خود را مشاهده نمایید.

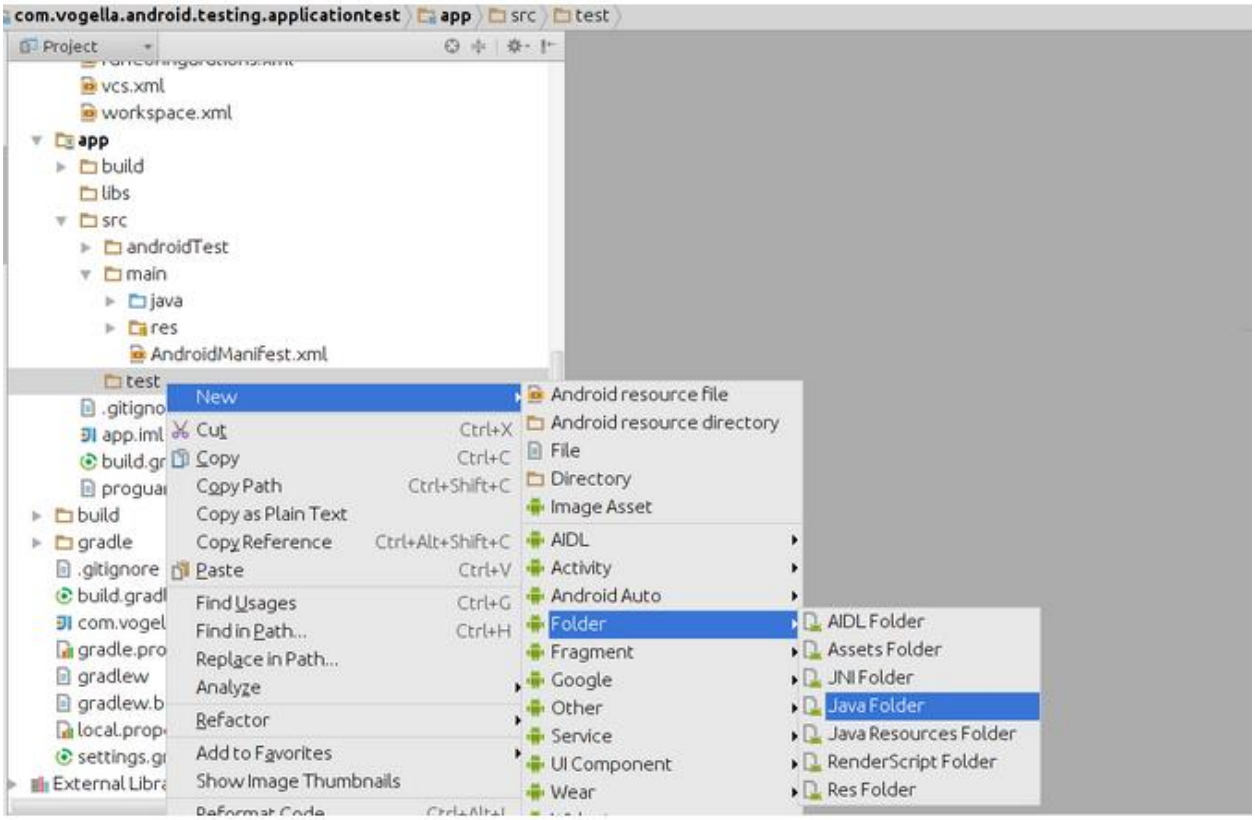


Src را انتخاب نموده و پس از باز کردن Context menu یک پوشه ی تست (test folder) جدید ایجاد نمایید.



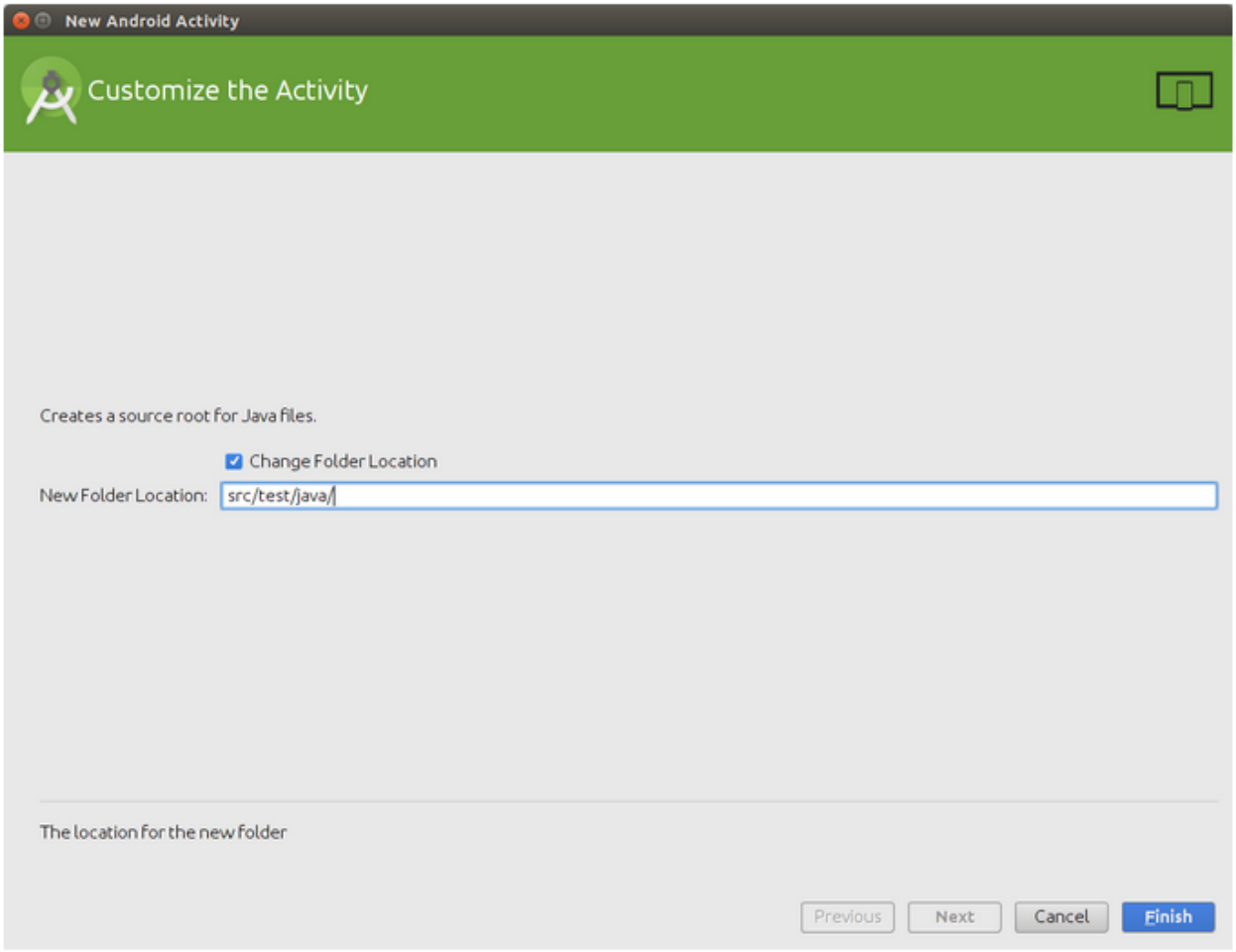
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



در صورتی که همچنان کامل انجام نشده باشد، کتابخانه ی JUnit (dependency) را به فایل Gradle build اضافه نمایید.

```
dependencies {
    // Unit testing dependencies
    testCompile 'junit:junit:4.12'
    // Set this dependency if you want to use the Hamcrest matcher library
    testCompile 'org.hamcrest:hamcrest-library:1.3'
    // more stuff, e.g., Mockito
}
```

توجه: لازم به ذکر است که در نتیجه ی ساخت پوشه ی Java، ممکن است پوشه ی جدید تست به عنوان source file به فایل gradle.build اضافه گردد. چنانچه دستور زیر در فایل

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

app/build.gradle موجود باشد، در آن صورت می بایست کد ذکر شده را کاملا از فایل حذف نمایید. تست نباید به هیچ وجه به عنوان یک source folder معمولی در نظر گرفته شود.

```
sourceSets { main { java.srcDirs = ['src/main/java', 'src/test/java/'] } }
```

پس از آن کافی است unit test خود را به این ساختار درختی پوشه ها یا folder structure اضافه نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش سوم :

اجرای unit test بر روی اپلیکیشن با استفاده از فریم ورک تست گیری Mockito

آموزش حاضر نحوه ی تست نویسی برای نرم افزارهای اندرویدی را با استفاده از فریم ورک Mockito شرح می دهد.

7-20-تست بخش های مجزای نرم افزار با استفاده از آبجکت های ساختگی (mock objects)

1-20-7-هدف از نوشتن unit test و چالش هایی که در تست با آن ها مواجه می شوید

Unit test، همان طور که از اسم آن پیدا است، بایستی یک اپلیکیشن را در سطح کلاس های مجزا آزمایش نماید، به بیان دیگر هر یک از کلاس های تشکیل دهنده ی پروژه را به صورت جداگانه مورد تست قرار داده و از عملکرد صحیح تمامی آن ها اطمینان حاصل کند. برای نیل به این هدف در unit test، لازم است side effect (عوارض) یا وابستگی به سایر کلاس ها و بخش های سیستم حذف شود. بدیهی است که برای حذف این وابستگی ها، می بایست dependency به دیگر کلاس ها را حذف نمایید.

این کار با استفاده از جایگزین برای dependency های واقعی پروژه امکان پذیر می باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

2-20-7-طبقه بندی کلاس های تست گیری مختلف

یک dummy object/آبجکت خام بین بخش های مختلف پروژه پاس داده می شود اما هیچگاه توابع داخل آن به معنای واقعی فراخوانی نمی شوند (dummy object: آبجکتی که API و توابع کتابخانه ای یک آبجکت را عینا کپی نماید. در واقع این آبجکت ها به توسعه دهنده اجازه می دهند تا آبجکت هایی که کلاس های پروژه به آن نیاز دارند را، بدون اینکه پیاده سازی و رفتار خاصی برای آن تعریف کرده باشند، شبیه سازی نماید). با استفاده از این آبجکت می توان، به عنوان مثال، لیست پارامترهای یک متد را با مقادیر پر کرد.

Fake object ها / توابع ساختگی دارای پیاده سازی و قابلیت های واقعی اما معمولا بسیار ساده هستند. به عنوان مثال، بجای استفاده از دیتابیس واقعی از یک دیتابیس مقیم در حافظه (in-memory) استفاده می کنند.

پیش از توضیح مفهوم stub class، بد نیست به شرح مفهوم test stub بپردازیم. Test stub یک قطعه برنامه است که قابلیت های ضروری (که کامپوننت مورد تست به آن ها وابسته هست) از کامپوننت مورد تست را شبیه سازی می کند. test stub ها در جواب توابع صدا زده شده در طول تست، پاسخ های معین ارائه می دهند.

به عبارت دیگر، stub ها برنامه هایی هستند که به منزله ی یک جایگزین موقتی برای کامپوننت فراخوانی شده ایفای نقش کرده و همان خروجی نرم افزار یا برنامه ی واقعی را تولید می کنند.

Stub class (کلاس جایگزین با پیاده سازی نیمه) ، عبارت است از implementation ناتمام از کل یک کلاس یا interface که نمونه ای از آن در طول تست گیری مورد استفاده قرار می گیرد. stub ها ممکن است اطلاعاتی درباره ی فراخوانی های رخ داده نیز ثبت نمایند.

حال به شرح مفهوم mock object (آبجکت ساختگی) می پردازیم. Mock object ابجکت های ساختگی هستند که رفتار آبجکت های واقعی را به صورت کنترل شده و مشخص شبیه سازی و به عبارتی تقلید می کنند. برنامه نویس اغلب با استفاده از آبجکت ساختگی، سعی دارد تا رفتار آبجکت (واقعی) دیگری را شبیه سازی نماید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

به تعریف دیگر، mock object، عبارت است از پیاده سازی ساختگی برای یک interface یا کلاس که در آن خروجی فراخوانی برخی توابع تعریف می شود.

می توان test double ها را به آبجکت های مورد تست ارسال کرد (test double یک واژه ی عمومی برای تمامی توابع و آبجکت های ساختگی است که به منظور تست نرم افزار و کنترل کیفیت آن مورد استفاده قرار می گیرند). تست های شما می توانند بررسی کنند آیا کلاس مورد نظر در طول اجرای تست طبق انتظار پاسخ می دهند یا خیر.

به طور مثال شما در طول تست می توانید بررسی کنید، آیا توابع مورد نظر در سطح آبجکت به درستی فراخوانی می شوند یا خیر. این قابلیت به شما امکان می دهد تا اطمینان حاصل نمایید به هنگام تست گیری، منحصرا کلاس تست شده و اینکه تست های شما وابستگی به بخش های دیگر برنامه نداشته باشند (یا در نتیجه ی اجرای تست سایر بخش های پروژه تاثیر نپذیرند و عوارض جانبی درکار نباشد).

توسعه دهنده این امکان را دارد که آبجکت های ساختگی (mock) را به صورت اختصاصی تنظیم کند. گفتنی است که آبجکت های ساختگی به کد و تنظیمات کمتری احتیاج داشته و بایستی در اجرای تست بر روی نرم افزار بر سایر روش ها ارجحیت داشته باشند.

3-20-7-نحوه ی ساخت mock object

می توانید آبجکت های ساختگی را خود به صورت دستی (از طریق کد) ایجاد کنید یا اینکه از یک فریم ورک شبیه سازی (mocking framework) جهت تقلید قابلیت ها و رفتارهای کلاس های واقعی اپلیکیشن استفاده نمایید. Mocking framework ها به شما این امکان را می دهند تا آبجکت های ساختگی ایجاد نموده و برای آن ها رفتاری مشابه رفتار کلاس مورد تست تعریف نمایید.

مثالی شناخته شده و معمول از mock object، یک data provider ساده هست. در مرحله ی production یا استفاده ی واقعی از اپلیکیشن، برای اتصال به data source، پیاده سازی و کد

لازم مورد استفاده قرار می گیرد. این در حالی است که برای تست، یک آبجکت ساختگی منبع داده ای را شبیه سازی نموده و اطمینان حاصل می نماید که شرایط تست همواره ثابت هستند.

این آبجکت های ساختگی را می توان در اختیار کلاس مورد تست قرار داد. به همین جهت، کلاسی که تست می شود نباید وابستگی (dependency) زیادی به داده های خارجی داشته باشد.

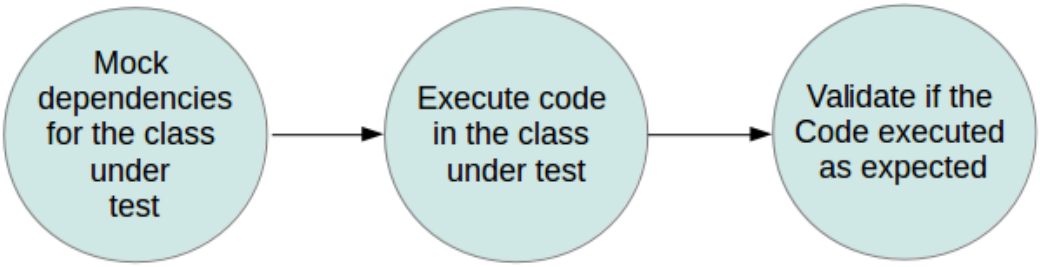
فریم ورک های شبیه سازی (Mocking framework) در کل این امکان را فراهم می آورند تا تعامل مورد انتظار با آبجکت ساختگی (mock) را به راحتی تست نمایید. به عنوان مثال، شما می توانید بررسی کنید که تنها متدهای مد نظر بر روی آبجکت ساختگی فراخوانی شده اند.

4-20-7-استفاده از Mockito برای شبیه سازی و ایجاد آبجکت ساختگی

Mockito یک فریم ورک پرطرفدار شبیه سازی و ایجاد آبجکت ساختگی برای تست نرم افزار است که گاه همراه با JUnit نیز مورد استفاده قرار می گیرد. این فریم ورک توسعه دهنده را قادر می سازد آبجکت های ساختگی ایجاد کرده و رفتار یا پیاده سازی آن ها را بر اساس نیاز تنظیم کند. در واقع توسعه دهنده با بهره گیری از Mockito می تواند تست نویسی برای کلاس هایی که به سایر کلاس ها و کتابخانه ها وابسته هستند یا به عبارتی دیگر dependency به داده های خارجی دارند را بسیار آسان نماید.

در استفاده از Mockito در تست های نرم افزاری خود، اغلب اقداماتی نظیر عنوان شده در زیر را انجام خواهید داد:

- تمامی Dependency ها به خارج از کلاس مورد تست (کلاس هایی که کلاس مورد تست با آن ها تعامل دارد) را شبیه سازی نموده و سپس آبجکت های ساختگی خود را داخل کد مورد تست درج می نمایید.
- کد مورد تست را اجرا نمایید.
- اطمینان حاصل نمایید که کد مورد نظر به درستی اجرا می شود.



7-21-افزودن Mockito در قالب dependency به پروژه

7-21-1-استفاده از سیستم کامپایل Gradle

برای کامپایل پروژه با استفاده از Gradle، لازم است ابتدا کتابخانه (dependency) زیر را به فایل Gradle build اضافه نمایید.

```
repositories { jcenter() }
dependencies { testCompile "org.mockito:mockito-core:2.0.57-beta" }
```

7-21-2-استفاده از سیستم کامپایل Maven

کاربران سیستم Maven می توانند خود یک dependency تعریف کنند. عبارت های g:"org.mockito"، a:"mockito-core" را در وب سایت http://search.maven.org جستجو نموده و فایل pom مورد نیاز را پیدا نمایید.

7-21-3-استفاده از محیط کاری Eclipse

محیط برنامه نویسی Eclipse با هر دو سیستم Gradle و Maven سازگار می باشد. لازم به ذکر است که آخرین ویرایش فریم ورک Mockito امکان دانلود ابزار هر دو سیستم را نمی دهد. بنابراین توصیه می شود در این محیط کاری از مجموعه ابزار یکی از دو سیستم مزبور استفاده نمایید.

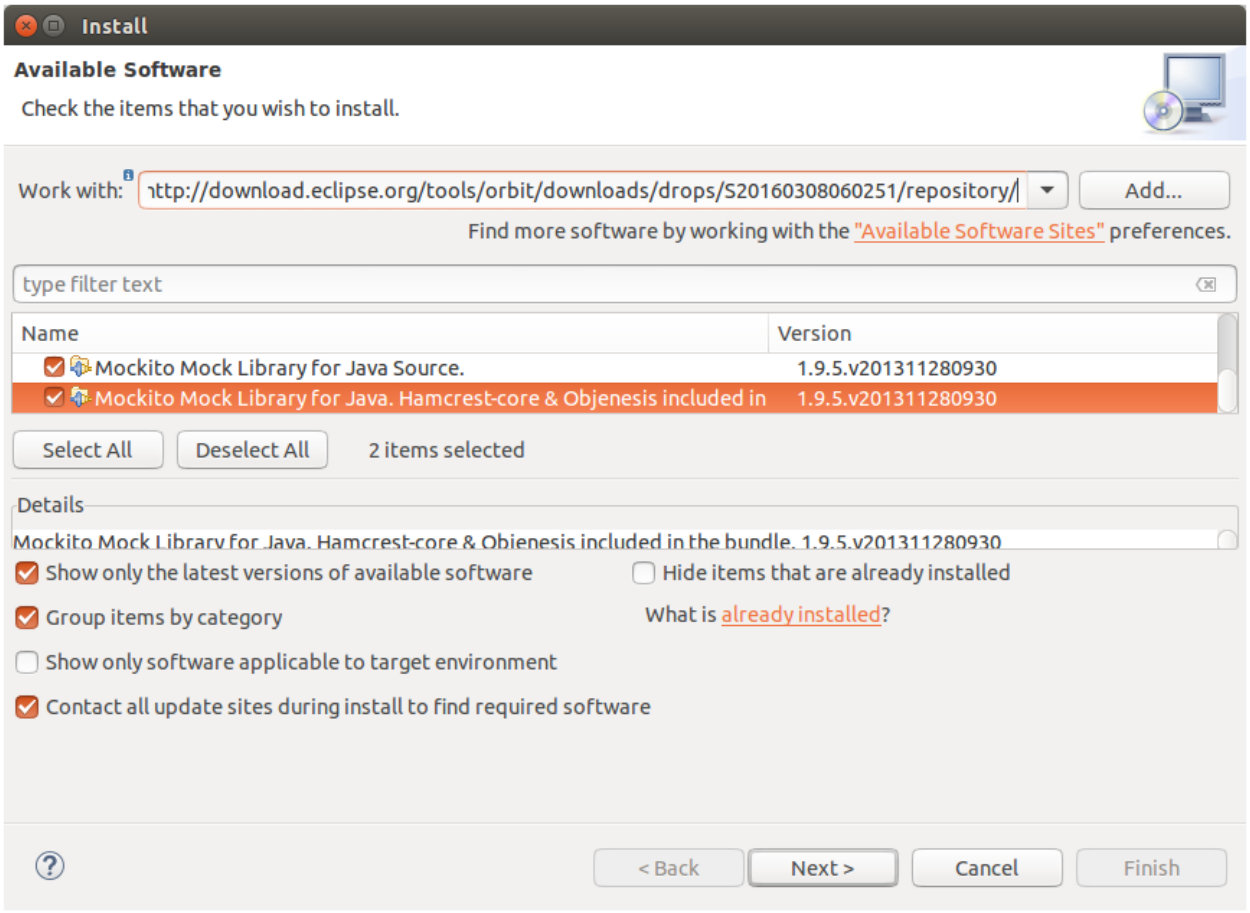
4-21-7-افزونه نویسی برای Eclipse یا OSGi (چارچوب توسعه و نصب کتابخانه ها و کامپوننت های نرم افزاری)

در اپلیکیشن های RCP یا برنامه های تحت وب سمت کلاینت مبتنی بر Eclipse، کتابخانه ها و dependency های مورد نیاز غالبا از مخزن پروژه ی p2 repository گرفته می شوند. Repository های Orbit هم منبع خوبی برای کتابخانه های کمکی (third party) هستند که به راحتی در اپلیکیشن ها یا افزونه های مبتنی بر Eclipse می توانید از آن ها استفاده نمایید.

برای دسترسی به repository های Orbit می توانید به آدرس http://download.eclipse.org/tools/orbit/downloads مراجعه نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



7-22-استفاده از توابع کتابخانه ای Mockito API / Mockito

7-22-1-دستورات Static import

اگر به دستور org.mockito.Mockito.*;، یک static import اضافه نمایید، در آن صورت قادر خواهید بود به متدهای Mockitos مانند ()mock به طور مستقیم دسترسی داشته باشید. static import ها به شما این امکان را می دهند تا اعضای static کلاس همچون متدها و فیلدها را بدون اینکه اسم کلاس را به طور صریح ذکر کنید، فراخوانی و استفاده نمایید.

2-22-7-اعلان و تنظیم آبجکت های ساختگی / mock objects با فریم ورک Mockito

همان طور که قبلا نیز ذکر شد، Mockito یک فریم ورک است که امکان تعریف آبجکت های ساختگی (mock object) را فراهم می آورد. برای این منظور، فریم ورک نام برده دو روش زیر را ارائه می دهد:

1. فراخوانی تابع ()mock
2. استفاده از دستور @Mock

در صورت استفاده از روش دوم، شما بایستی خود آبجکت های ساختگی را مقداردهی اولیه نمایید. جهت مقداردهی اولیه این آبجکت لازم است از MockitoRule استفاده کنید. MockitoRule متد static ای به نام MockitoAnnotations.initMocks(this) را فراخوانی می کند که فیلدهای نشانه گذاری شده (دارای annotation) را مقداردهی می نماید. در صورت تمایل می توانید از دستور @RunWith(MockitoJUnitRunner.class) استفاده کرده و همین نتیجه را بدست بیاورید.

کد زیر استفاده از دستور @Mock و MockitoRule را به صورت عملی نمایش می دهد.

```
import static org.mockito.Mockito.*;
public class MockitoTest  {
      @Mock                                                                  (1
      MyDatabase databaseMock;
      @Rule public MockitoRule mockitoRule = MockitoJUnit.rule();            (2
      @Test
      public void testQuery()  {
            ClassToTest t  = new ClassToTest(databaseMock);                  (3
            boolean check = t.query("* from t");
                                                                  (4
            assertTrue(check);                                               (5
            verify(databaseMock).query("* from t");                          (6
      }
}
```

1. به Mockito اعلان می کند که از نمونه ی databaseMock شبیه سازی یا mock کند.
2. به Mockito دستور می دهد تا بر مبنای دستور @Mock، آبجکت های ساختگی یا mock ایجاد کند.

3. با استفاده از آبجکت ساختگی از کلاس مورد تست نمونه سازی (instantiate) می کند.
4. بخشی از کد کلاس مورد تست را اجرا می کند.
5. انتظار دارد (assert می کند) که متد فراخوانده شده در خروجی مقدار بولی true را باز گرداند.
6. بررسی کرده و اطمینان حاصل می کند که متد query بر روی mock یا نمونه ی ساختگی از MyDatabase فراخوانی می شود.

3-22-7-تنظیم آبجکت های ساختگی / mock ها

می توانید به طور دقیق مشخص کنید که یک متد پس از فراخوانی، چه مقداری را در خروجی بازگرداند و این کار را به واسطه ی تکنیک نوشتن روان، خوانا و پشت سرهم دستورات یا همان fluent API که فریم ورک Mockito در اختیار شما قرار می دهد، انجام دهید.

دو متد when(....).thenReturn(....) که به صورت زنجیره ای و پشت سرهم فراخوانی می شوند این امکان را به شما می دهد تا یک شرط معین تعریف کرده و مقدار بازگشتی آن را نیز مشخص نمایید. اگر بیش از یک مقدار مشخص کنید، این مقادیر دقیقا به همان ترتیبی که تعریف شده اند، برگردانده می شوند. mock ها ممکن است بر اساس پارامترهای ارسالی به متد، مقادیر متفاوتی را بازگردانی نمایند. همچنین می توانید با استفاده از توابع anyString یا anyInt تعیین کنید که مستقل از مقدار ورودی، مقدار خاص به عنوان خروجی از متد بازگردانده شود.

```
import static org.mockito.Mockito.*;
import static org.junit.Assert.*;
@Test
public void test1() {
	// create mock
	MyClass test = Mockito.mock(MyClass.class);
	// define return value for method getUniqueId()
	when(test.getUniqueId()).thenReturn(43);
	// use mock in test....
	assertEquals(test.getUniqueId(), 43);
}
// Demonstrates the return of multiple values
@Test
public void testMoreThanOneReturnValue() {
	Iterator i= mock(Iterator.class);
	when(i.next()).thenReturn("Mockito").thenReturn("rocks");
	String result=i.next()+" "+i.next();
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
        //assert
        assertEquals("Mockito rocks", result);
}
// this test demonstrates how to return values based on the input
@Test
public void testReturnValueDependentOnMethodParameter() {
        Comparable c= mock(Comparable.class);
        when(c.compareTo("Mockito")).thenReturn(1);
        when(c.compareTo("Eclipse")).thenReturn(2);
        //assert
        assertEquals(1,c.compareTo("Mockito"));
}
// this test demonstrates how to return values independent of the input value
@Test
public void testReturnValueInDependentOnMethodParameter() {
        Comparable c= mock(Comparable.class);
        when(c.compareTo(anyInt())).thenReturn(-1);
        //assert
        assertEquals(-1 ,c.compareTo(9));
}
// return a value based on the type of the provide parameter
@Test
public void testReturnValueInDependentOnMethodParameter() {
        Comparable c= mock(Comparable.class);
        when(c.compareTo(isA(Todo.class))).thenReturn(0);
        //assert
        Todo todo = new Todo(5);
        assertEquals(todo ,c.compareTo(new Todo(1)));
}
```

دو تابع doReturn(...).when(...).methodCall نیز که به صورت پشت سرهم (زنجیره ای/chained) صدا زده می شوند مشابه نمونه ی قبلی عمل می کنند، اما بیشتر برای متدهایی که خروجی ندارند (void هستند)، کارا هستند. می توان با استفاده از doThrow برای متدهایی که خروجی ندارند (void بر می گردانند)، یک خطا (exception) صادر کرد. این کاربرد در تکه کد زیر به نمایش گذاشته می شود.

```
import static org.mockito.Mockito.*;
import static org.junit.Assert.*;
// this test demonstrates how use doThrow
@Test(expected=IOException.class)
public void testForIOException() {
        // create an configure mock
        OutputStream mockStream = mock(OutputStream.class);
        doThrow(new IOException()).when(mockStream).close();
        // use mock
        OutputStreamWriter streamWriter= new OutputStreamWriter(mockStream);
        streamWriter.close();
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
}
```

4-22-7-متد verify() (بررسی صحت فراخوانی یک متد با پارامترهای مورد نظر)

Mockito تمامی توابع فراخوانده شده و پارامترهای ارسالی به آبجکت ساختگی (mock) را رصد کرده و اطلاعات مربوط به آن ها را ثبت می کند. شما می توانید با استفاده از متد verify() بر روی آبجکت ساختگی مطمئن شوید که شرایط و نیاز های تعیین شده برآورده شده اند. به طور مثال، می توانید بررسی کرده و اطمینان حاصل نمایید که یک متد با پارامترهای مورد نیاز فراخوانی شده اند یا خیر. این تست گیری در اصطلاح behavior testing خوانده می شود. behavior testing با نتیجه ی متد فراخوانی شده کاری ندارد، بلکه می خواهد مطمئن شود آیا متد معینی با پارامترهای مورد نظر فراخوانی می شود یا خیر.

```
import static org.mockito.Mockito.*;
@Test
public void testVerify()  {
    // create and configure mock
    MyClass test = Mockito.mock(MyClass.class);
    when(test.getUniqueId()).thenReturn(43);
    // call method testing on the mock with parameter 12
    test.testing(12);
    test.getUniqueId();
    test.getUniqueId();
    // now check if method testing was called with the parameter 12
    verify(test).testing(Matchers.eq(12));
    // was the method called twice?
    verify(test, times(2)).getUniqueId();
    // other alternatives for verifiying the number of method calls for a method
    verify(mock, never()).someMethod("never called");
    verify(mock, atLeastOnce()).someMethod("called at least once");
    verify(mock, atLeast(2)).someMethod("called at least twice");
    verify(mock, times(5)).someMethod("called five times");
    verify(mock, atMost(3)).someMethod("called at most 3 times");
}
```

5-22-7-قرار دادن آبجکت های جاوا در Spy (رصد و شنود آبجکت های جاوا و توابع فراخوانده شده بر روی آن با spy)

می توانید با درج دستور Spy@ یا فراخوانی متد spy()، آبجکت های (واقعی) جاوا را رصد نمایید. بدین معنی که ابتدا آبجکت را داخل دو دستور ذکر شده قرار دهید، سپس تمامی توابع متعارف آبجکت را صدا زده و در این میان هر اتفاق یا تعاملی که رخ می دهد را دقیقا ردگیری (شنود و دنبال) کنید. در واقع با این کار تمامی توابع، به جز آن هایی که خود به صورت صریح مستثنی نمایید، بر روی این آبجکت صدا زده می شوند.

```
import static org.mockito.Mockito.*;
// Lets mock a LinkedList
List list = new LinkedList();
List spy = spy(list);
//You have to use doReturn() for stubbing
doReturn("foo").when(spy).get(0);
// this would not work
// real method is called so spy.get(0)
// throws IndexOutOfBoundsException (list is still empty)
when(spy.get(0)).thenReturn("foo");
```

متد verifyNoMoreInteractions() به شما این امکان را می دهد تا مطمئن شوید هیچ تابع دیگری (به غیر از آنچه خود به صورت صریح مشخص کرده اید) فراخوانی نشده است.

6-22-7-پیاده سازی الگوی dependency injection در فریم ورک Mockito با استفاده از دستور @InjectMocks

فریم ورک Mockito با ارائه ی دستوری به نام @InjectMocks به شما این امکان را می دهد تا بر اساس نوع dependency مورد نظر، آن را از طریق تابع سازنده (constructor)، متد یا فیلد تزریق نمایید (الگوی dependency injection را به این صورت پیاده سازی کنید).

```
public class ArticleManager {
   private User user;
   private ArticleDatabase database;
   ArticleManager(User user) {
```

```
    this.user = user;
  }
  void setDatabase(ArticleDatabase database) { }
}
```

این کلاس را می توان از طریق Mockito ساخته و کلاس های مورد نیاز یا dependency های آن را با استفاده از آبجکت های ساختگی (mocks) به داخل آن تزریق کرد.

```
@RunWith(MockitoJUnitRunner.class)
public class ArticleManagerTest  {
    @Mock ArticleCalculator calculator;
    @Mock ArticleDatabase database;
    @Most User user;
    @Spy private UserProvider userProvider = new ConsumerUserProvider();
    @InjectMocks private ArticleManager manager;                                    (1
    @Test public void shouldDoSomething() {
        // assume that ArticleManager has a method called initialize which calls a method
        // addListener with an instance of ArticleListener
        manager.initialize();
      // validate that addListener was called
        verify(database).addListener(any(ArticleListener.class));
    }
}
```

یک نمونه از کلاس ArticleManager ساخته و آبجکت های ساختگی (mock ها) را به داخل آن تزریق می کند.

7-22-7-دسترسی به آرگومان های ارسالی به توابع (Capturing arguments)

کلاس ArgumentCaptor به شما امکان می دهد تا به آرگومان های پاس داده به متدها در طول پروسه ی بررسی صحت فراخوانی متد، دسترسی داشته باشید. بدین وسیله شما قادر خواهید بود به آرگومان های توابع فراخوانی شده دسترسی داشته و سپس آن ها را در تست های خود مورد استفاده قرار دهید.

```
import static org.hamcrest.Matchers.hasItem;
import static org.junit.Assert.assertThat;
import static org.mockito.Mockito.mock;
import static org.mockito.Mockito.verify;
import java.util.Arrays;
import java.util.List;
import org.junit.Rule;
import org.junit.Test;
import org.mockito.ArgumentCaptor;
import org.mockito.Captor;
```

```
import org.mockito.junit.MockitoJUnit;
import org.mockito.junit.MockitoRule;
public class MockitoTests {
      @Rule public MockitoRule rule = MockitoJUnit.rule();
      @Captor
   private ArgumentCaptor<List<String>> captor;
      @Test
   public final void shouldContainCertainListItem() {
            List<String> asList = Arrays.asList("someElement_test", "someElement");
      final List<String> mockedList = mock(List.class);
      mockedList.addAll(asList);
      verify(mockedList).addAll(captor.capture());
      final List<String> capturedArgument = captor.<List<String>>getValue();
      assertThat(capturedArgument, hasItem("someElement"));
   }
}
```

8-22-7-محدودیت های فریم ورک Mockito

فریم ورک Mockito نیز مانند سایر چارچوب های کاری محدودیت هایی دارد. این فریم ورک قادر به تست ساختارهای زیر نمی باشد:

- کلاس هایی که به صورت final تعریف شده باشند و امکان ارث بری از آن ها وجود نداشته باشد
- کلاس های anonymous یا کلاس های فاقد اسم
- انواع داده ای اولیه و primitive همچون string

23-7-استفاده از فریم ورک Mockito در اندروید

Mockito را می توان به طور مستقیم نیز در unit test های اندروید مورد استفاده قرار داد. کافی است dependency مربوطه را به فایل Gradle build اضافه نمایید. جهت استفاده از این فریم ورک در تست های instrumentation اندروید، لازم است dexmaker و dexmaker-mockito را نیز به عنوان dependency در فایل Gradle build اضافه نمایید.

```
dependencies {
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    testCompile 'junit:junit:4.12'
    // required if you want to use Mockito for unit tests
    testCompile 'org.mockito:mockito-core:1.+'
    // required if you want to use Mockito for Android instrumentation tests
    androidTestCompile 'org.mockito:mockito-core:1.+'
    androidTestCompile "com.google.dexmaker:dexmaker:1.2"
    androidTestCompile "com.google.dexmaker:dexmaker-mockito:1.2"
}
```

تمرین: نوشتن تست مبتنی بر instrumentation با استفاده از فریم ورک Mockito

ایجاد اپلیکیشن های آزمایشی (مورد تست) در اندروید

یک اپلیکیشن جدید اندروید با اسم پکیج com.vogella.android.testing.mockito.contextmock تعریف نمایید. حال یک متد static داخل اعلان نموده و در بدنه ی آن یک آبجکت intent با پارامترهای زیر ایجاد نمایید.

```
public static Intent createQuery(Context context, String query, String value) {
    // Reuse MainActivity for simplification
    Intent i = new Intent(context, MainActivity.class);
    i.putExtra("QUERY", query);
    i.putExtra("VALUE", value);
    return i;
}
```

1-23-7-اضافه کردن dependency مورد نیاز (Mockito) به فایل app/build.gradle

```
dependencies {
    // the following is required to use Mockito and JUnit for your
    // instrumentation unit tests on the JVM
        androidTestCompile 'junit:junit:4.12'
    androidTestCompile 'org.mockito:mockito-core:2.0.57-beta'
    androidTestCompile 'com.android.support.test:runner:0.3'
```

```
    androidTestCompile "com.google.dexmaker:dexmaker:1.2"
    androidTestCompile "com.google.dexmaker:dexmaker-mockito:1.2"
    // the following is required to use Mockito and JUnit for your unit
    // tests on the JVM
    testCompile 'junit:junit:4.12'
    testCompile 'org.mockito:mockito-core:1.+'
}
```

2-23-7-ساخت یک unit test جدید

اکنون یک unit test جدید با استفاده از Mockito ایجاد نموده و بررسی کنید آیا intent با داده های extra مرتبط فراخوانی می شود یا خیر.

برای این منظور آبجکت Context را با استفاده از توابع Mockito به صورت زیر شبیه سازی نمایید.

```
package com.vogella.android.testing.mockitocontextmock;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.mockito.Mockito;
import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertNotNull;
public class TextIntentCreation {
  @Test
  public void testIntentShouldBeCreated() {
    Context context = Mockito.mock(Context.class);
    Intent intent = MainActivity.createQuery(context, "query", "value");
    assertNotNull(intent);
    Bundle extras = intent.getExtras();
    assertNotNull(extras);
    assertEquals("query", extras.getString("QUERY"));
    assertEquals("value", extras.getString("VALUE"));
  }
}
```

تمرین: ساخت اُبجکت های ساختگی (mock) با استفاده از توابع Mockito

هدف از تمرین

در تمرین حاضر، ابتدا API ای تعبیه نمایید که قابل تست و شبیه سازی باشند. سپس با استفاده از توابع Mockito آن ها را تست نمایید.

3-23-7-ایجاد یک نمونه Twitter API

ابتدا یک کلاس TwitterClient پیاده سازی کنید که از نمونه ای از اینترفیس ITweet استفاده کرده و توابع آن را فراخوانی می کند. دسترسی به نمونه های این interface ملال آور بوده و نیازمند استفاده از یک سرویس هزینه بر و پیچیده می باشد. علاوه بر این، سرویس مذکور بایستی برای دسترسی، اول فراخوانی و اجرا شود.

```
public interface ITweet {
    String getMessage();
}

public class TwitterClient {
    public void sendTweet(ITweet tweet) {
        String message = tweet.getMessage();
        // send the message to Twitter
    }
}
```

4-23-7-شبیه سازی از نمونه ی ITweet

برای اینکه لازم نباشد جهت دسترسی به نمونه هایی از اینترفیس ITweet، یک سرویس هزینه بر و پیچیده راه اندازی نمایید، توصیه می شود نمونه ی interface مزبور را به واسطه ی فریم ورک Mockito شبیه سازی کنید.

```
@Test
public void testSendingTweet() {
    TwitterClient twitterClient = new TwitterClient();
    ITweet iTweet = mock(ITweet.class);
    when(iTweet.getMessage()).thenReturn("Using mockito is great");
    twitterClient.sendTweet(iTweet);
}
```

اکنون کلاس TwitterClient می تواند توابع اینترفیس ساختگی و شبیه سازی شده ی ITweet را پیاده سازی نماید. در نتیجه ی استفاده از توابع این interface شبیه سازی شده (فراخوانی متد getMessage() بر روی نمونه ی ساختگی ITweet)، پیغام "Using Mockito is great" به عنوان خروجی متد در اختیار شما قرار می گیرد.

7-23-5-بررسی صحت فراخوانی متد

اطمینان حاصل نمایید که متد getMessage() حداقل یکبار فراخوانی می شود.

```
@Test
public void testSendingTweet() {
	TwitterClient twitterClient = new TwitterClient();
	ITweet iTweet = mock(ITweet.class);
	when(iTweet.getMessage()).thenReturn("Using mockito is great");
	twitterClient.sendTweet(iTweet);
	verify(iTweet, atLeastOnce()).getMessage();
}
```

7-23-6-مرحله ی اعتبار سنجی تست

تست را اجرا کرده و مطمئن شوید که با موفقیت انجام می شود.

7-24-استفاده از PowerMock به همراه Mockito

7-24-1-استفاده از PowerMock برای شبیه سازی متدهای static

فریم ورک Mockito قادر به شبیه سازی متدهای static نیست. برای شبیه سازی رفتار متدهای static ، بایستی از PowerMock استفاده نمایید. PowerMock یک کلاس به نام "PowerMockito" ارائه می دهد که برای ساخت class/object/mock، بررسی صحت اجرا(اعتبارسنجی) و برآورده ساختن انتظارات این کلاس را فراخوانی می کند.

برای بررسی صحت برآورده شدن انتظارات می توانید همچنان از خود Mockito (توابع times() ، anyInt() استفاده نمایید.

```
import java.net.InetAddress;
import java.net.UnknownHostException;
public final class NetworkReader {
```

```
    public static String getLocalHostname() {
       String hostname = "";
       try {
          InetAddress addr = InetAddress.getLocalHost();
          // Get hostname
          hostname = addr.getHostName();
       } catch ( UnknownHostException e ) {
       }
       return hostname;
    }
}
```

برای نوشتن تستی که قابلیت های NetworkReader را برای شما شبیه سازی می کند، می توانید از تکه کد زیر استفاده نمایید.

```
import org.junit.runner.RunWith;
import org.powermock.core.classloader.annotations.PrepareForTest;
@RunWith( PowerMockRunner.class )
@PrepareForTest( NetworkReader.class )
public class MyTest {
// Find the tests here
 @Test
public void testSomething() {
   mockStatic( NetworkUtil.class );
   when( NetworkReader.getLocalHostname() ).andReturn( "localhost" );
   // now test the class which uses NetworkReader
}
```

25-7-استفاده از یک wrapper بجای Powermock

همچنین شما می توانید تابع static دلخواه خود را داخل یک wrapper گنجانده (یک تابع که تابع دیگری را فراخوانی می کند) و سپس آن را توسط Mockito شبیه سازی نمایید.

```
class FooWraper {
  void someMethod() {
    Foo.someStaticMethod()
  }
}
```

بخش چهارم :

استفاده از فریم ورک Hamcrest جهت اجرای تست نرم افزاری

این مبحث به اجرای تست نرم افزاری بر روی اپلیکیشن به وسیله ی فریم ورک Hamcrest می پردازد.

7-26-هدف از Hamcrest matcher framework

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Hamcrest یک فریم ورک و مجموعه ای از کتابخانه های اجرای تست های نرم افزاری هست. فریم ورک مزبور به شما این امکان را می دهد تا به واسطه ی کلاس های matcher از پیش موجود، شرایط خاصی را بررسی کرده و از صحت آن ها اطمینان حاصل نمایید. علاوه بر آن، شما می توانید matcher های خود را با پیاده سازی اختصاصی تعبیه نمایید.

Hamcrest نسل سوم از فریم ورک matcher هست. اولین نسل از این فریم ورک از assert(logical statement) استفاده می کرد که کاستی قابل توجه آن عدم خوانایی بالا بود. نسل دوم توابع خاصی برای assertion و انتظار دریافت خروجی مورد نظر همچون assertEquals() ارائه دادند. این رویکرد سبب اضافه شدن توابع assert متعدد دیگری به Hamcrest که نسل سوم از این فریم ورک می باشد، شد. Hamcrest از متدی به نام assertThat و یک عبارت matcher برای بررسی اینکه تست موفق آمیز بود یا خیر استفاده می کند.

Hamcrest سعی دارد تا تست ها را تا حد امکان خوانا، مختصر و کاربر پسند تعبیه نماید. به طور مثال، متد is یک تابع میزبان یا wrapper برای equalTo(value) می باشد.

```
import static org.hamcrest.MatcherAssert.assertThat;
import static org.hamcrest.Matchers.is;
import static org.hamcrest.Matchers.equalTo;
boolean a;
boolean b;
// all statements test the same
assertThat(a, equalTo(b));
assertThat(a, is(equalTo(b)));
assertThat(a, is(b));
```

تکه کدهای زیر دستورات assert خالص JUnit4 را با matcher های Hamcrest مقایسه می کنند.

```
// JUnit 4 for equals check
assertEquals(expected, actual);
// Hamcrest for equals check
assertThat(actual, is(equalTo(expected)));
// JUnit 4 for not equals check
assertFalse(expected.equals(actual));
// Hamcrest for not equals check
assertThat(actual, is(not(equalTo(expected))));
```

می توانید matcher ها را با استفاده از anyof یا allof به صورت زنجیره ای و پشت سرهم فراخوانی نمایید.

```
assertThat("test", anyOf(is("testing"), containsString("est")));
```

در کل، پیغام های خطایی که Hamcrest تولید می کند، بسیار خوانا می باشند. علاوه بر آن در استفاده از Hamcrest، ویژگی type safety زبان جاوا به خوبی رعایت شده و matcher ها که از انواع داده ای generic استفاده می کنند دیگر با مشکل بر نخواهند خورد (کد type-safe اجازه ی دسترسی به بخش های نامربوط حافظه را نمی دهد. به عبارت دیگر تنها به آن بخش هایی از حافظه که امکان پذیر است اجازه ی دسترسی را می دهد).

27-7-استفاده از matcher ها در Hamcrest

27-7-1-اعلان dependency های مربوطه برای Gradle

جهت استفاده از Hamcrest matcher ها در پروژه ای که برای کامپایل از سیستم Gradle استفاده می کند، کافی است کتابخانه / dependency های زیر را به آن اضافه نمایید.

```
dependencies {
   // Unit testing dependencies
   testCompile 'junit:junit:4.12'
   // Set this dependency if you want to use Hamcrest matching
   testCompile 'org.hamcrest:hamcrest-library:1.3'
}
```

27-7-2-اعلان dependency های لازم برای سیستم Maven

جهت استفاده از Hamcrest در پروژه ای که برای کامپایل از سیستم Maven استفاده می کند، لازم است dependency زیر را به فایل pom خود اضافه نمایید.

```
<dependency>
<groupId>org.hamcrest</groupId>
<artifactId>hamcrest-library</artifactId>
<version>1.3</version>
```

```
<scope>test</scope>
</dependency>
```

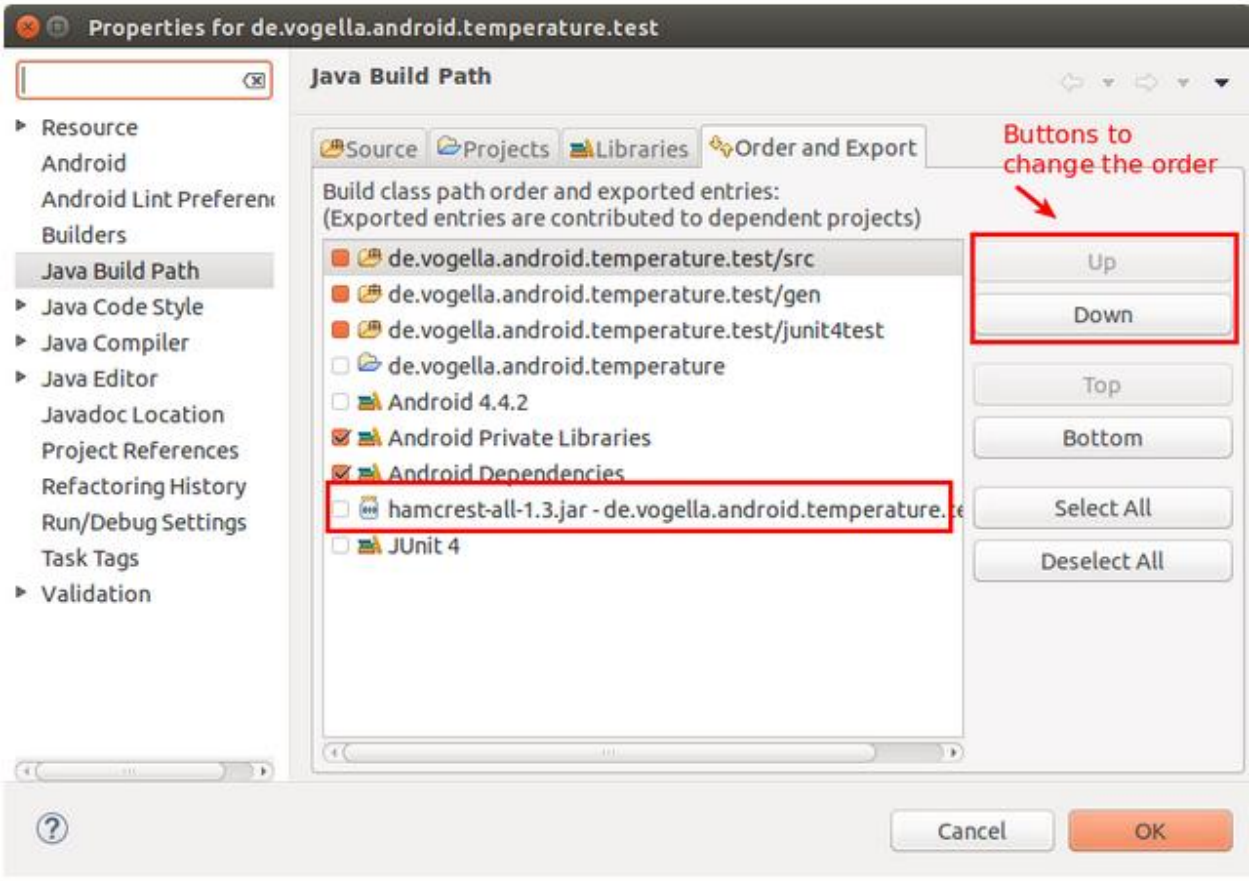
3-27-7-اضافه کردن Hamcrest به طور مستقیم به classpath پروژه در محیط کاری Eclipse

ویرایشی از JUnit که در محیط برنامه نویسی Eclipse نصب می شود، صرفا matcher های اصلی Hamcrest را شامل می شود. جهت استفاده از تمامی matcher های موجود، آخرین نسخه ی hamcrest-all-*.jar را از آدرس https://code.google.com/p/hamcrest/downloads/list دنلود نمایید و سپس آن را به classpath پروژه ی خود اضافه کنید.

نکته: در صورت برخورد با خطای "java.lang.SecurityException: class "org.hamcrest.Matchers"'s signer information does not match signer information of other classes in the same package"، اطمینان حاصل نمایید که فایل hamcrest jar قبل از کتابخانه ی Junit در build path قرار گرفته است. جهت ویرایش ترتیب قرارگیری می توانید به بخش project properties در محیط کاری Eclipse تحت آدرس Java Build Path در تب Order and Export مراجعه نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



28-7-استفاده کاربردی از Hamcrest

مثال

کاربرد عملی از matcher های Hamcrest در کد زیر به نمایش گذاشته شده است.

```
assertThat(Long.valueOf(1), instanceOf(Integer.class));
```

1-28-7-استفاده ازmatcher های Hamcrest برای list ها

استفاده از matcher های Hamcrest برای نوع داده ای list در تکه کد زیر به نمایش گذاشته شده است.

```
import org.junit.Test;
import java.util.Arrays;
import java.util.List;
import static org.hamcrest.MatcherAssert.assertThat;
import static org.hamcrest.Matchers.contains;
import static org.hamcrest.Matchers.containsInAnyOrder;
import static org.hamcrest.Matchers.greaterThan;
import static org.hamcrest.collection.IsCollectionWithSize.hasSize;
import static org.hamcrest.core.Every.everyItem;
public class HamcrestListMatcherExamples {
   @Test
   public void listShouldInitiallyBeEmpty() {
      List<Integer> list = Arrays.asList(5, 2, 4);
      assertThat(list, hasSize(3));
      // ensure the order is correct
      assertThat(list, contains(5, 2, 4));
      assertThat(list, containsInAnyOrder(2, 4, 5));
      assertThat(list, everyItem(greaterThan(1)));
   }
}
// Check that a list of objects has a property race and
// that the value is not ORC
assertThat(fellowship, everyItem(hasProperty("race", is(not((ORC))))));
```

2-28-7-مرور کلی بر matcher های Hamcrest

مهم ترین matcher های فریم ورک مزبور در زیر شرح داده شده اند:

- allof – چنانچه تمامی matcher ها با یکدیگر مطابقت داشته باشند، همخوانی رخ می دهد.
- anyof – چنانچه هر کدام از matcher ها مطابقت داشته باشد، همخوانی رخ می دهد.
- not – در صورتی که matcher گنجانده شده در wrapper مطابقت نداشته باشد و بالعکس، همخوانی رخ می دهد.
- equalTo – با استفاده از متد equals بررسی می کند آیا آبجکت ها با هم برابر هستند یا خیر.
- is – یک پیشوند برای equalTo است که صرفا جهت بهبود خوانایی کد به این متد اضافه می شود.
- hasToString - Object.toString را تست می کند.

- instanceOf، isCompatibleType – نوع را تست می کند.
- notNullValue، nullValue – بررسی می کند آیا مقدار null است یا خیر.
- sameInstance – هویت آبجکت را بررسی می کند. در واقع بررسی می کند آیا این آبجکت از همان نمونه است یا خیر.
- hasEntry، hasKey، hasValue – بررسی می کند آیا نوع داده ای map آیتمی داخل خود دارد یا خیر.
- hasItem، hasItems – بررسی می کند آیا یک collection داخل خود آیتم هایی را دارد یا تهی است.
- hasItemInArray – بررسی می کند آیا یک آرایه تهی است یا حاوی آیتم و مقداری می باشد.
- closeTo – بررسی می کند آیا مقادیر ممیز شناور نزدیک به مقدار مورد نظر هستند یا خیر.
- greaterThan، greaterThanOrEqualTo، lessThan، lessThanOrEqualTo
- equalToIgnoringWhiteSpace – بررسی می کند آیا رشته ها با هم برابر هستند یا خیر. لازم به ذکر است که در تست فضای خالی نادیده گرفته می شود.
- containsString، endsWith، startsWith – بررسی می کند آیا رشته ها با هم مطابقت دارند یا خیر.

29-7-ساخت Hamcrest matcher اختصاصی خود

توسعه دهنده می تواند اعضای کلاسی به نام TypeSafeMatcher را به ارث برده (extend) و بدین وسیله Hamcrest matcher اختصاصی خود را بنویسد. مثال زیر یک نمونه از پیاده سازی matcher را نمایش می دهد که وظیفه ی آن بررسی تطابق بین یک String با عبارت باقاعده ی (regular expression) مورد نظر می باشد.

```
import org.hamcrest.Description;
import org.hamcrest.TypeSafeMatcher;
public class RegexMatcher extends TypeSafeMatcher<String> {
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    private final String regex;
    public RegexMatcher(final String regex) {
        this.regex = regex;
    }
    @Override
    public void describeTo(final Description description) {
        description.appendText("matches regular expression=`" + regex + "`");
                                                                                    }
    @Override
    public boolean matchesSafely(final String string) {
        return string.matches(regex);
    }
     // matcher method you can call on this matcher class
  public static RegexMatcher matchesRegex(final String regex) {
    return new RegexMatcher(regex);
  }
}
```

نمونه کد زیر نحوه ی استفاده و پیاده سازی آن را نمایش می دهد.

```
package com.vogella.android.testing.applicationtest;
import org.junit.Test;
import static org.hamcrest.MatcherAssert.assertThat;
public class TestCustomMatcher {
  @Test
  public void testRegularExpressionMatcher() throws Exception {
    String s ="aaabbbaaaa";
    assertThat(s, RegexMatcher.matchesRegex("a*b*a*"));
  }
}
```

بخش پنجم :

نوشتن تست نرم افزاری برای اپلیکیشن با فریم ورک AssertJ

در مبحث حاضر نحوه ی نوشتن تست های نرم افزاری مختصر، خوانا و کارامد برای اپلیکیشن های اندرویدی خود با استفاده از فریم ورک AssertJ را خواهید آموخت.

7-30-مقدمه ای بر AssertJ

AssertJ نیز یک کتابخانه اجرای تست نرم افزاری همچون Hamcrest است که با ارائه ی تعداد زیاری دستور assert و interface کارا به توسعه دهنده این اجازه را می دهد تا تست های نگهداشت پذیر و کارامد برای پروژه ی اندرویدی خود بنویسند. در واقع AssertJ یک شاخه از کتابخانه ای از دستورات assert به نام Fest است که امروزه دیگر به صورت حرفه ای پشتیبانی نمی شود.

کتابخانه ی مزبور بیشتر جهت آسان سازی نوشتن دستورات assert در تست های نرم افزاری و بهبود خوانایی آن ها تعبیه شد. این کتابخانه با برخورداری از interface کارامد، به توسعه دهنده امکان می دهد تا دستورات assert را به صورت کارا و روان (پشت سرهم) جهت تست بخش های نرم افزار بنویسد.

متد پایه و اصلی کتابخانه ی نام برده assertThat بوده و دستورات مربوط به assert (بررسی صحت اجرای دستور و برآورده شدن انتظارات) بلافاصله پس از این متد درج می شوند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

7-31-استفاده ی کاربردی از AssertJ

Gradle_7_32

جهت استفاده از AssertJ همراه با سیستم کامپایل Gradle در پروژه، لازم است کتابخانه (dependency) زیر را به فایل Gradle اضافه نمایید.

```
testCompile 'org.assertj:assertj-core:2.0.0'
```

Maven_7_31_2

به منظور استفاده از کتابخانه ی مورد نظر در پروژه ای که با سیستم Maven کامپایل می شود، کافی است dependency (کتابخانه ی) زیر را به فایل pom خود اضافه نمایید.

```
<dependency>
  <groupId>org.assertj</groupId>
  <artifactId>assertj-core</artifactId>
  <!-- use 3.0.0 for Java 8 projects -->
  <version>2.0.0</version>
  <scope>test</scope>
</dependency>
```

7_31_3_تنظیمات مرتبط با محیط کاری Eclipse

جهت استفاده از متد assertThat از کتابخانه ی AssertJ، پس از طی کردن مسیر رو به رو: Window ▸ Preferences ▸ Java ▸ Editor ▸ Content assist ▸ Favorites ▸ New Type ، دستور org.assertj.core.api.Assertions را درج نمایید.

پس از درج دستور فوق بایستی عبارت org.assertj.core.api.Assertions.* را در کد مشاهده نمایید.

7_31_4_تنظیمات مرتبط با محیط کاری IntelliJ

برای استفاده از محیط برنامه نویسی IntelliJ نیاز به تنظیمات خاص نیست. کافی است واژه ی assertThat را تایپ نموده و سپس با فشردن کلیدهای (Ctrl-Space)، قابلیت تکمیل خودکار کد (code completion) محیط کاری میزبان را دو بار فراخوانی نمایید.

مثالی کاربردی از AssertJ

نمونه کد زیر از صفحه ی اصلی سایت AssertJ اقتباس شده است.

```
// unique entry point to get access to all assertThat methods and utility methods (e.g. entry)
import static org.assertj.core.api.Assertions.*;
// common assertions
assertThat(frodo.getName()).isEqualTo("Frodo");
assertThat(frodo).isNotEqualTo(sauron)
                 .isIn(fellowshipOfTheRing);
// String specific assertions
assertThat(frodo.getName()).startsWith("Fro")
                           .endsWith("do")
                           .isEqualToIgnoringCase("frodo");
// collection specific assertions
assertThat(fellowshipOfTheRing).hasSize(9)
                               .contains(frodo, sam)
                               .doesNotContain(sauron);
// using extracting magical feature to check fellowshipOfTheRing characters name :)
assertThat(fellowshipOfTheRing)
     .extracting("name")
     .contains("Boromir", "Gandalf", "Frodo", "Legolas")
     .doesNotContain("Sauron", "Elrond");
// map specific assertions, ringBearers initialized with the elves rings and the one ring bearers.
assertThat(ringBearers).hasSize(4)
                       .contains(entry(oneRing, frodo), entry(nenya, galadriel))
                       .doesNotContainEntry(oneRing, aragorn);
// and many more assertions : dates, file, numbers, exceptions ...
```


بخش ششم :

تست لایه ی UI اپلیکیشن های اندرویدی به واسطه ی فریم ورک Espresso

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آموزش حاضر به شرح نحوه ی توسعه ی تست های نرم افزاری برای لایه ی رابط کاربری اپلیکیشن های اندرویدی با استفاده از فریم ورک تست گیری Espresso می پردازد. جهت درک کامل مفاهیم این مبحث می بایست با اصول برنامه نویسی با زبان Java و چارچوب کاری Android آشنا باشید.

32-7-فریم ورک تست گیری Espresso و تست لایه ی UI اپلیکیشن

Espresso یک فریم ورک جهت آسان سازی تعبیه تست های نرم افزاری قابل اطمینان برای لایه ی رابط کاربری پروژه های اندرویدی می باشد. شرکت گوگل این فریم ورک را برای اولین بار در اکتبر سال 2013 معرفی کرد.

Espresso از ویرایش 2.0 بخشی از Android Support Repository محسوب می شود.

این فریم ورک تست گیری به صورت خودکار عملیات تست گیری شما را با لایه ی UI اپلیکیشن هماهنگ می سازد.

Espresso اطمینان حاصل می کند که قبل از به اجرا در آمدن تست، activity مورد نظر حتما راه اندازی شده باشد. از دیگر ویژگی های جالب توجه فریم ورک مذکور این است که منتظر می ماند تا تمامی background activity ها پایان یابند و سپس خود به اجرا در می آید.

Espresso در اصل برای تست قابلیت های یک اپلیکیشن (اطمینان از عملکرد صحیح اپلیکیشن در سطح UI یک اپلیکیشن) در آن واحد طراحی شده، اما می توان از آن برای تست تعامل اپلیکیشن ها با هم نیز بهره گرفت. چنانچه قصد تست تعامل بین اپلیکیشن ها را دارید (نه صرفا بررسی صحت عملیاتی که در سطح UI یک اپلیکیشن رخ می دهد)، لازم است از تکنیک تست گیری black box استفاده نمایید (تکنیک تست گیری black box به روشی در تست نرم‌افزار اشاره دارد که در آن فرض می‌شود اطلاعاتی در مورد جزئیات داخلی عملکرد اپلیکیشن مورد نظر وجود ندارد و تمرکز تست‌ها منحصرا بر روی خروجی‌های مختلف در مقابل ورودی‌های متفاوت است).

Espresso در کل از سه کامپوننت نرم افزاری زیر تشکیل شده است:

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

- ViewMatchers - به شما امکان می دهد تا view مورد نظر را در نمودار درختی view ها / view hierarchy جاری پیدا کنید.
- ViewActions - اجازه می دهد تا عملیاتی را بر روی view ها انجام دهید.
- ViewAssertions - به توسعه دهنده این امکان را می دهد تا جهت اطمینان از صحت اطلاعات مربوط به وضعیت view state / view مورد نظر (برآورده شدن انتظار خاصی در رابطه با وضعیت view)، view را تست کند.

یک تست ساده ی Espresso

ساختار کلی تست های Espresso به صورت زیر می باشد:

```
onView(ViewMatcher)
.perform(ViewAction)
  .check(ViewAssertion);
```

این تست : 1) view را پیدا می کند 2) عملیاتی بر روی view اجرا می نماید 3) با بررسی وضعیت view پی می برد آیا اطلاعات و وضعیت view با آنچه انتظار می رفت مطابقت دارد یا خیر.

کد زیر استفاده ی کاربردی از Espresso را به نمایش می دهد.

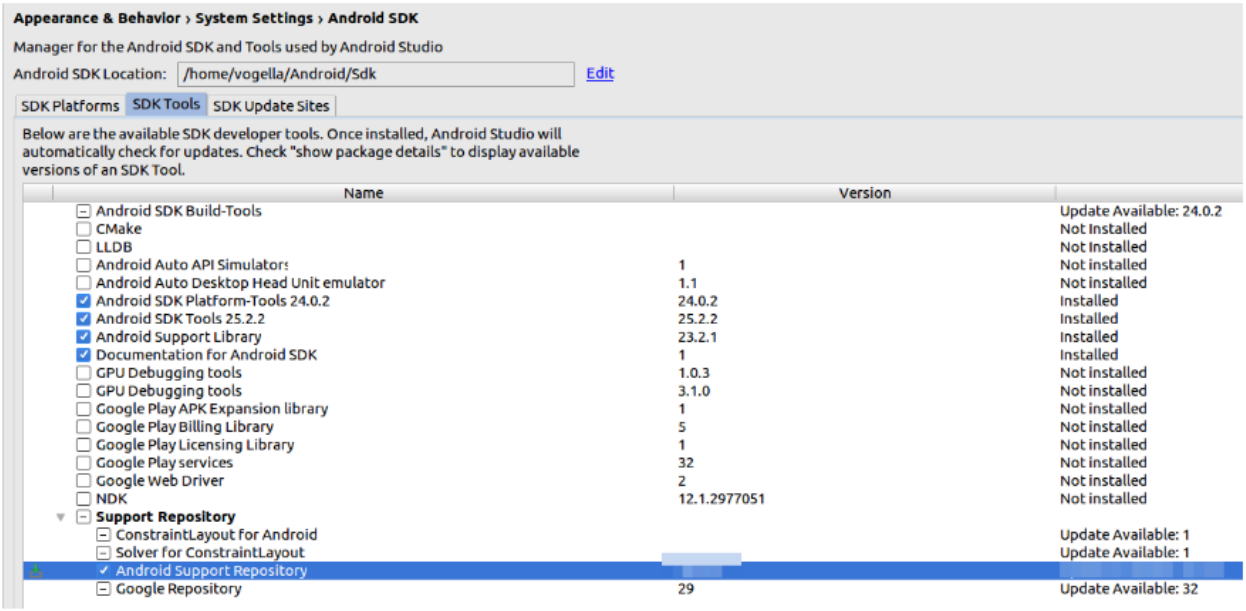
```
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.assertion.ViewAssertions.matches;
import static android.support.test.espresso.matcher.ViewMatchers.isDisplayed;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
// image more code here...
// test statement
onView(withId(R.id.my_view))          // withId(R.id.my_view) is a ViewMatcher
      .perform(click())               // click() is a ViewAction
      .check(matches(isDisplayed())); // matches(isDisplayed()) is a ViewAssertion
// new test
onView(withId(R.id.greet_button))
.perform(click())
.check(matches(not(isEnabled())));
```

چنانچه Espresso با فراخوانی ViewMatcher قادر به یافتن view مورد نظر نبود، در آن صورت کل ساختار درختی view را در پیغام خطا می گنجاند. این امر برای تجزیه و تحلیل و برطرف نمودن مشکل مفید می باشد.

7-33-نصب و استفاده از Espresso

1-33-7-نصب

ابتدا Android Support Repository را از طریق Android SDK manager نصب نمایید.



2-33-7-تنظیم فایل Gradle build برای استفاده از توابع Espresso

جهت استفاده از امکانات Espresso در تست های نرم افزاری خود، کتابخانه ی زیر را به فایل Gradle build اپلیکیشن اضافه نمایید.

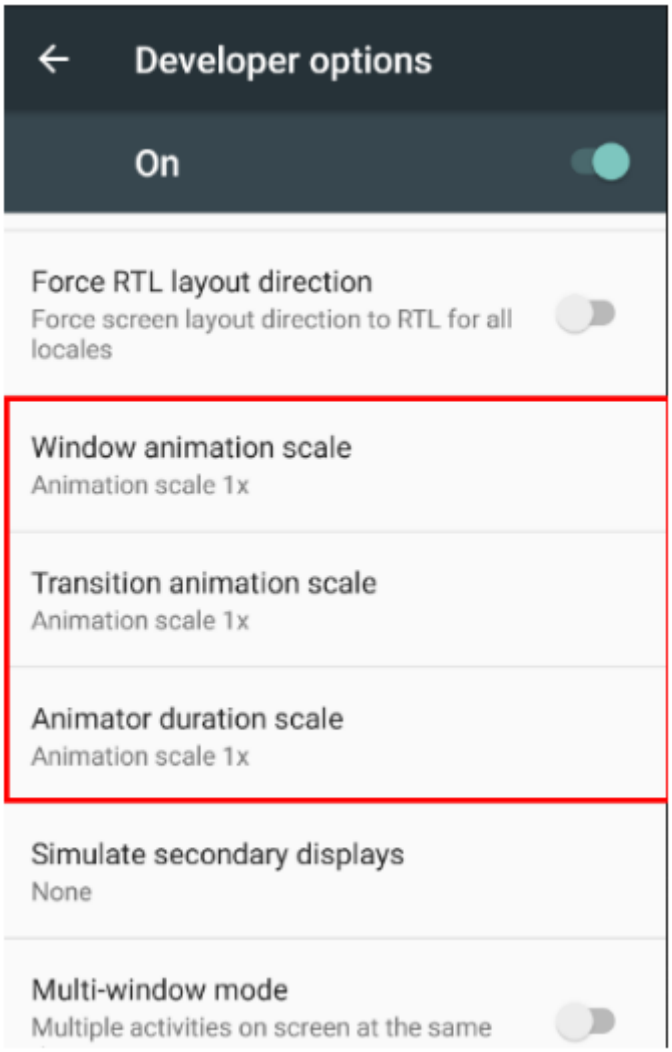
```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    // Android runner and rules support
    androidTestCompile 'com.android.support.test:runner:0.5'
    androidTestCompile 'com.android.support.test:rules:0.5'
    // Espresso support
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    // add this for intent mocking support
    androidTestCompile 'com.android.support.test.espresso:espresso-intents:2.2.2'
    // add this for webview testing support
    androidTestCompile 'com.android.support.test.espresso:espresso-web:2.2.2'
}
```

لازم است android.support.test.runner.AndroidJUnitRunner را به عنوان مقدار پارامتر testInstrumentationRunner داخل فایل build اپلیکیشن خود ذکر نمایید. شاید لازم باشد با توجه به کتابخانه ی مورد استفاده، مقدار LICENSE.txt را از packagingOptions حذف نمایید. کد زیر مثالی در این زمینه را نشان می دهد.

```
apply plugin: 'com.android.application'
android {
    compileSdkVersion 22
    buildToolsVersion '22.0.1'
    defaultConfig {
        applicationId "com.example.android.testing.espresso.BasicSample"
        minSdkVersion 10
        targetSdkVersion 22
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    packagingOptions {
        exclude 'LICENSE.txt'
    }
    lintOptions {
        abortOnError false
    }
}
dependencies {
    // as before.....
}
```

3-33-7-تنظیمات دستگاه (Device settings)

اگر می خواهید از Espresso جهت تست لایه ی UI اپلیکیشن استفاده کنید، در آن صورت توصیه می شود انیمیشن های سیستمی را غیرفعال نموده، بدین وسیله از رخداد خطا یا از کار افتادگی ناگهانی نرم افزار جلوگیری کنید و نیز مطمئن شوید که خروجی تست ثابت و تکرار پذیر می باشد.



تمرین: استفاده ی کاربردی از Espresso جهت تست پروژه

ایجاد پروژه ی و تست آن توسط Espresso

یک پروژه ی جدید اندرویدی به نام Espresso First با اسم پکیج com.vogella.android.espressofirst ایجاد نمایید. این پروژه را بر اساس قالب آماده ی Blank Template ایجاد نمایید.

محتوای فایل layout پروژه ی خود، activity_main.xml را به صورت زیر ویرایش نمایید.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   tools:context=".MainActivity">
   <EditText
      android:id="@+id/inputField"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content" />
   <Button
      android:id="@+id/changeText"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="New Button" android:onClick="onClick"/>
   <Button
      android:id="@+id/switchActivity"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Change Text" android:onClick="onClick"/>
</LinearLayout>
```

یک فایل جدید به نام activity_second.xml ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical" android:layout_width="match_parent"
   android:layout_height="match_parent">
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/textAppearanceLarge"
      android:text="Large Text"
      android:id="@+id/resultView" />
</LinearLayout>
```

یک activity جدید با محتوای زیر ایجاد نمایید.

```
package com.vogella.android.espressofirst;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
public class SecondActivity extends Activity{
   @Override
   protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
        TextView viewById = (TextView) findViewById(R.id.resultView);
        Bundle inputData = getIntent().getExtras();
        String input = inputData.getString("input");
        viewById.setText(input);
    }
}
```

سپس بدنه ی کلاس MainActivity خود را نیز به صورت زیر ویرایش نمایید.

```
package com.vogella.android.espressofirst;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
public class MainActivity extends Activity {
    EditText editText;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        editText = (EditText) findViewById(R.id.inputField);
    }
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.changeText:
                editText.setText("Lalala");
                break;
            case R.id.switchActivity:
                Intent intent = new Intent(this, SecondActivity.class);
                intent.putExtra("input", editText.getText().toString());
                startActivity(intent);
                break;
        }
    }
}
```

4-33-7-تنظیم و ویرایش فایل app build.gradle

تنظیمات را طبق توضیحات مرتبط با تنظیم فایل Gradle build برای استفاده از Espresso در مقاله ی حاضر انجام دهید.

5-33-7-تست نویسی برای پروژه بر اساس فریم ورک Espresso

```
package com.vogella.android.espressofirst;
import android.support.test.rule.ActivityTestRule;
import android.support.test.runner.AndroidJUnit4;
```

```
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.action.ViewActions.closeSoftKeyboard;
import static android.support.test.espresso.action.ViewActions.typeText;
import static android.support.test.espresso.assertion.ViewAssertions.matches;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
import static android.support.test.espresso.matcher.ViewMatchers.withText;
@RunWith(AndroidJUnit4.class)
public class MainActivityEspressoTest {
   @Rule
   public ActivityTestRule<MainActivity> mActivityRule =
         new ActivityTestRule<>(MainActivity.class);
   @Test
   public void ensureTextChangesWork() {
      // Type text and then press the button.
      onView(withId(R.id.inputField))
            .perform(typeText("HELLO"), closeSoftKeyboard());
      onView(withId(R.id.changeText)).perform(click());
      // Check that the text was changed.
      onView(withId(R.id.inputField)).check(matches(withText("Lalala")));
   }
   @Test
   public void changeText_newActivity() {
      // Type text and then press the button.
      onView(withId(R.id.inputField)).perform(typeText("NewText"),
            closeSoftKeyboard());
      onView(withId(R.id.switchActivity)).perform(click());
      // This view is in a different Activity, no need to tell Espresso.
      onView(withId(R.id.resultView)).check(matches(withText("NewText")));
   }
}
```

6-33-7-ساخت و تنظیم intent) start intent اجرا و راه اندازی activity دوم)

پس از ارسال مقدار false به عنوان پارامتر سوم به متد ActivityTestRule، می توانید به ساخت و تنظیم آبجکت intent برای فراخوانی و اجرای activity دوم بپردازید. این عملیات در نمونه کد زیر به نمایش گذاشته شده است.

```
package com.vogella.android.testing.espressosamples;
```

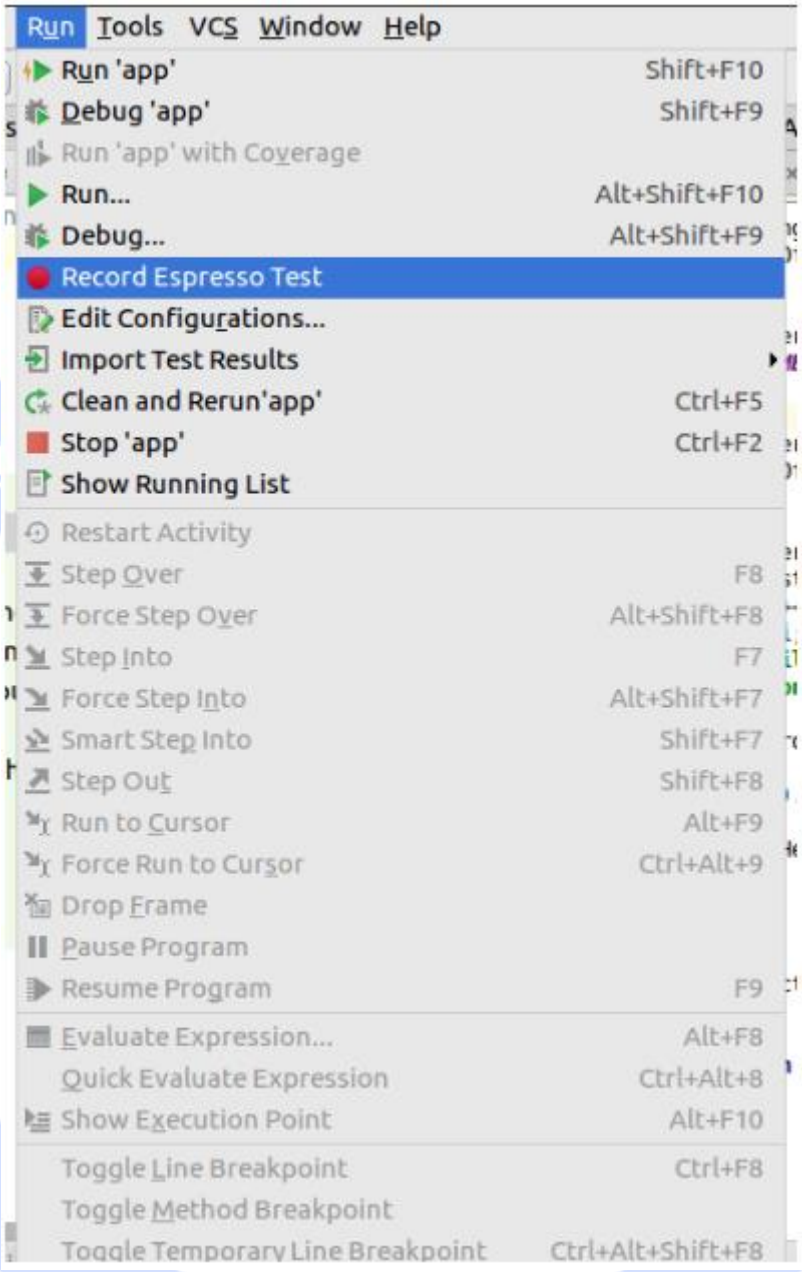
```
import android.content.Intent;
import android.support.test.rule.ActivityTestRule;
import android.support.test.runner.AndroidJUnit4;
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.assertion.ViewAssertions.matches;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
import static android.support.test.espresso.matcher.ViewMatchers.withText;
@RunWith(AndroidJUnit4.class)
public class SecondActivityTest {
   @Rule
   // third parameter is set to false which means the activity is not started automatically
   public ActivityTestRule<SecondActivity> rule =
         new ActivityTestRule(SecondActivity.class, true, false);
   @Test
   public void demonstrateIntentPrep() {
      Intent intent = new Intent();
      intent.putExtra("EXTRA", "Test");
      rule.launchActivity(intent);
 onView(withId(R.id.display)).check(matches(withText("Test")));
   }
}
```

7-33-7-ضبط تعامل با UI اپلیکیشن و جهت اجرای تست Espresso (Espresso UI recorder)

محیط برنامه نویسی Android Studio در منوی Run خود آیتمی به نام Record Espresso Test دارد (جهت دسترسی به آن این مسیر را طی نمایید: Run ▸ Record Espresso Test) که با فعال شدن، تعامل کاربر با UI اپلیکیشن را ضبط کرده و از روی آن یک تست Espresso ایجاد می کند.

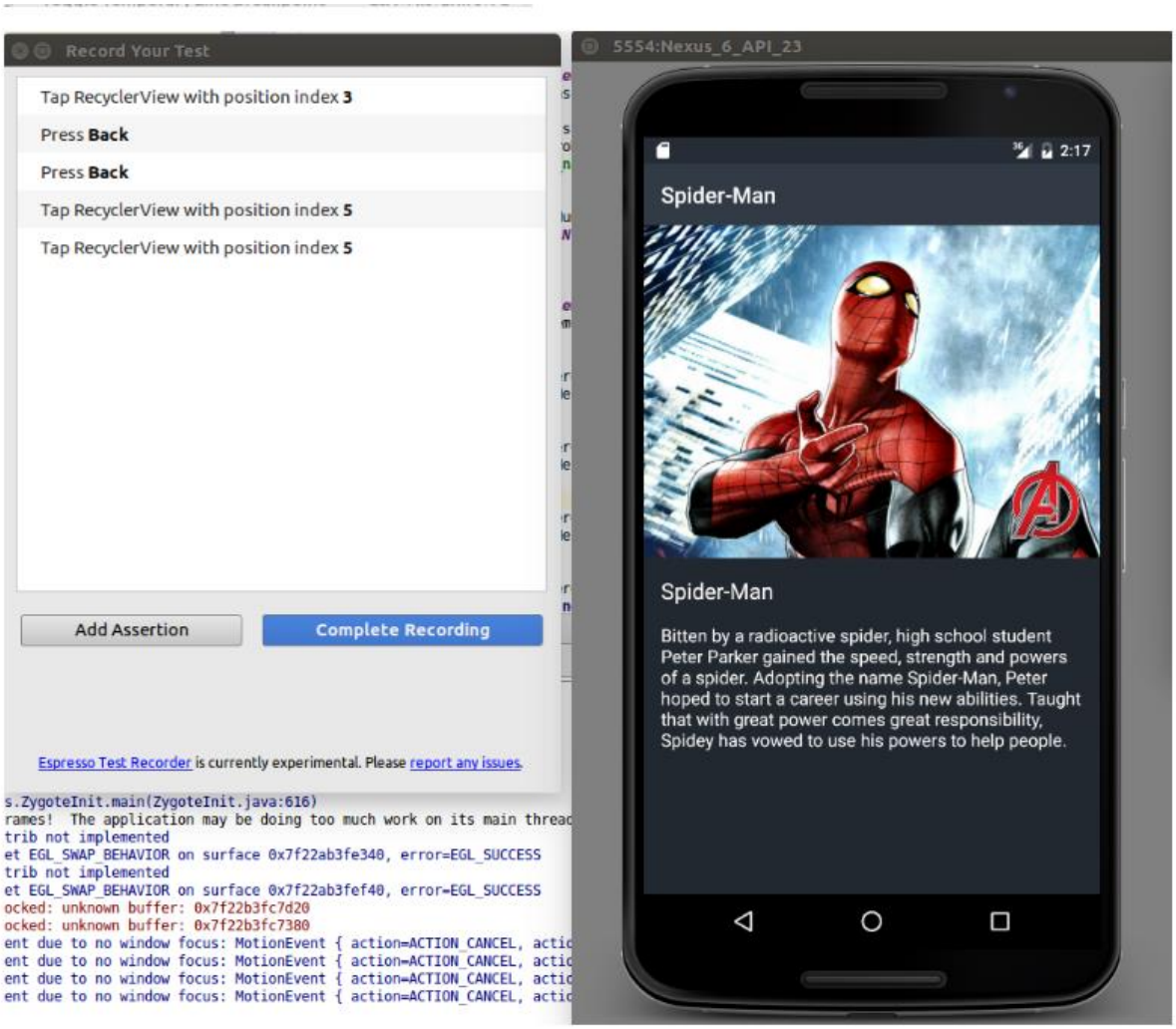
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



8-33-7-تنظیم activity مورد تست

می توانید به آبجکت activity مورد تست دسترسی داشته و توابع دلخواه را بر روی آن فراخوانی نمایید. فرض کنید می خواهید یک متد را به صورت زیر بر روی نمونه ای از کلاس activity خود صدا بزنید.

```
public class MainActivity extends Activity {
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
   }
   public void configureMainActivity(String Uri) {
      // do something with this
```

```
    }
}
```

می توانید متد ConfigureMainActivity را به صورت زیر بر روی نمونه ای از کلاس activity خود (MainActivity) فراخوانی نمایید.

```
package com.vogella.android.myapplication;
import android.support.test.rule.ActivityTestRule;
import android.support.test.runner.AndroidJUnit4;
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
@RunWith(AndroidJUnit4.class)
public class ExampleInstrumentedTest {
    @Rule
    public ActivityTestRule<MainActivity> mActivityRule =
            new ActivityTestRule<MainActivity>(MainActivity.class);
    @Test
    public void useAppContext() throws Exception {
        MainActivity activity = mActivityRule.getActivity();
        activity.configureMainActivity("http://www.vogella.com");
        // do more
    }
}
```

نکته: همچنین می توانید متدهای مورد نظر را در ActivityTestRule بازنویسی (override) نمایید. برای مثال، بدنه ی توابع beforeActivityLaunched و afterActivityLaunched را بازنویسی نمایید.

می توانید به activity جاری نیز دسترسی داشته باشید.

```
@Test
public void navigate() {
        Activity instance = getActivityInstance();
        onView(withText("Next")).perform(click());
        Activity activity = getActivityInstance();
        boolean b = (activity instanceof  SecondActivity);
        assertTrue(b);
        // do more
    }
    public Activity getActivityInstance() {                                    (1
        final Activity[] activity = new Activity[1];
        InstrumentationRegistry.getInstrumentation().runOnMainSync(new Runnable( ) {
            public void run() {
                Activity currentActivity = null;
                Collection resumedActivities =
ActivityLifecycleMonitorRegistry.getInstance().getActivitiesInStage(RESUMED);
```

```
            if (resumedActivities.iterator().hasNext()){
                currentActivity = (Activity) resumedActivities.iterator().next();
                activity[0] = currentActivity;
            }
        }
    });
    return activity[0];
}
```

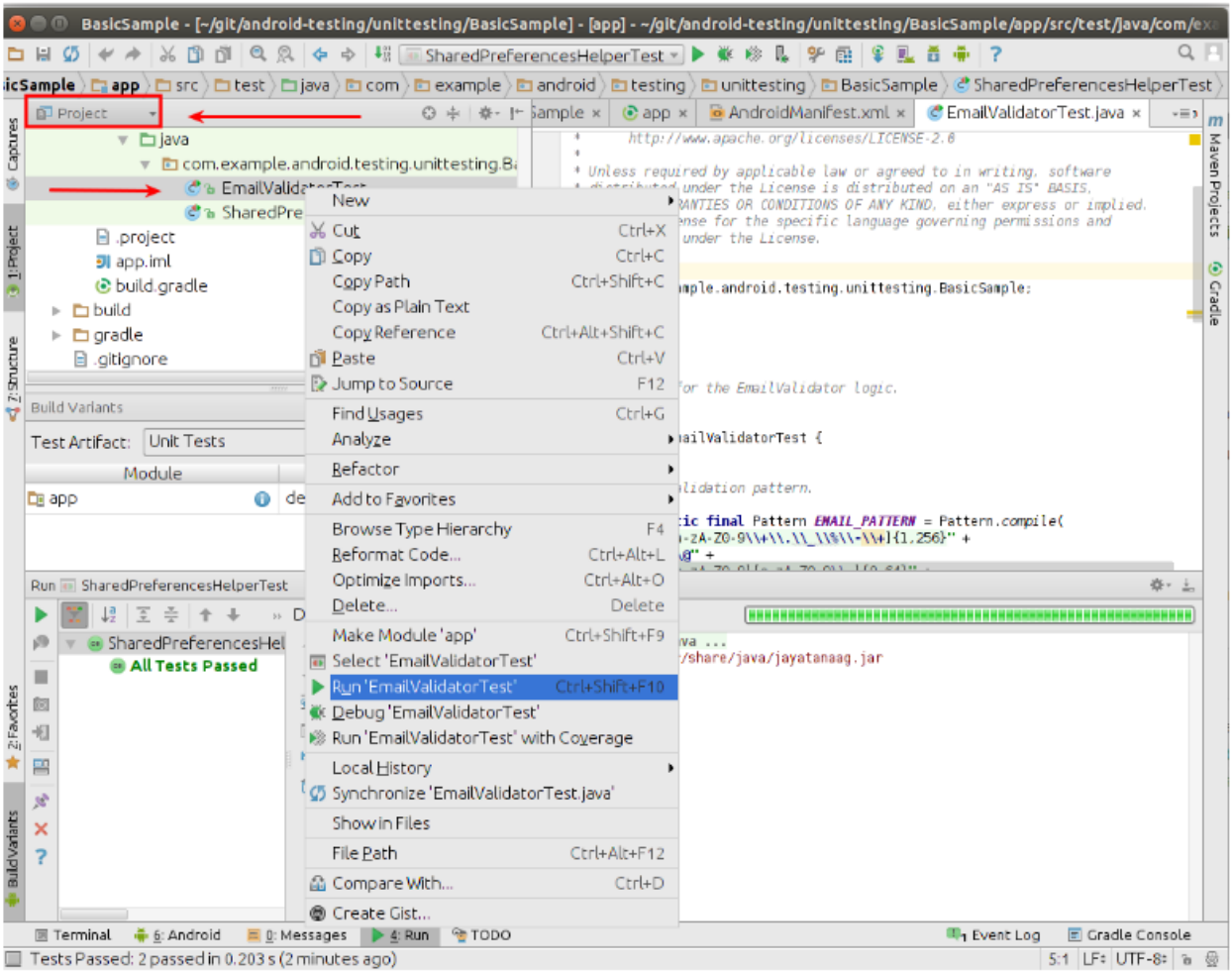
امکان دسترسی به activity جاری را فراهم می سازد.

توجه: لازم به ذکر است که دستور ActivityLifecycleMonitorRegistry جزء توابع کتابخانه ای / API نیست و به همین جهت امکان تغییر آن در آینده وجود دارد.

7-34-اجرای تست های Espresso

1-34-7-اجرای تست در محیط کاری Android Studio

بر روی تست راست کلیک کرده و سپس گزینه ی Run را انتخاب نمایید.

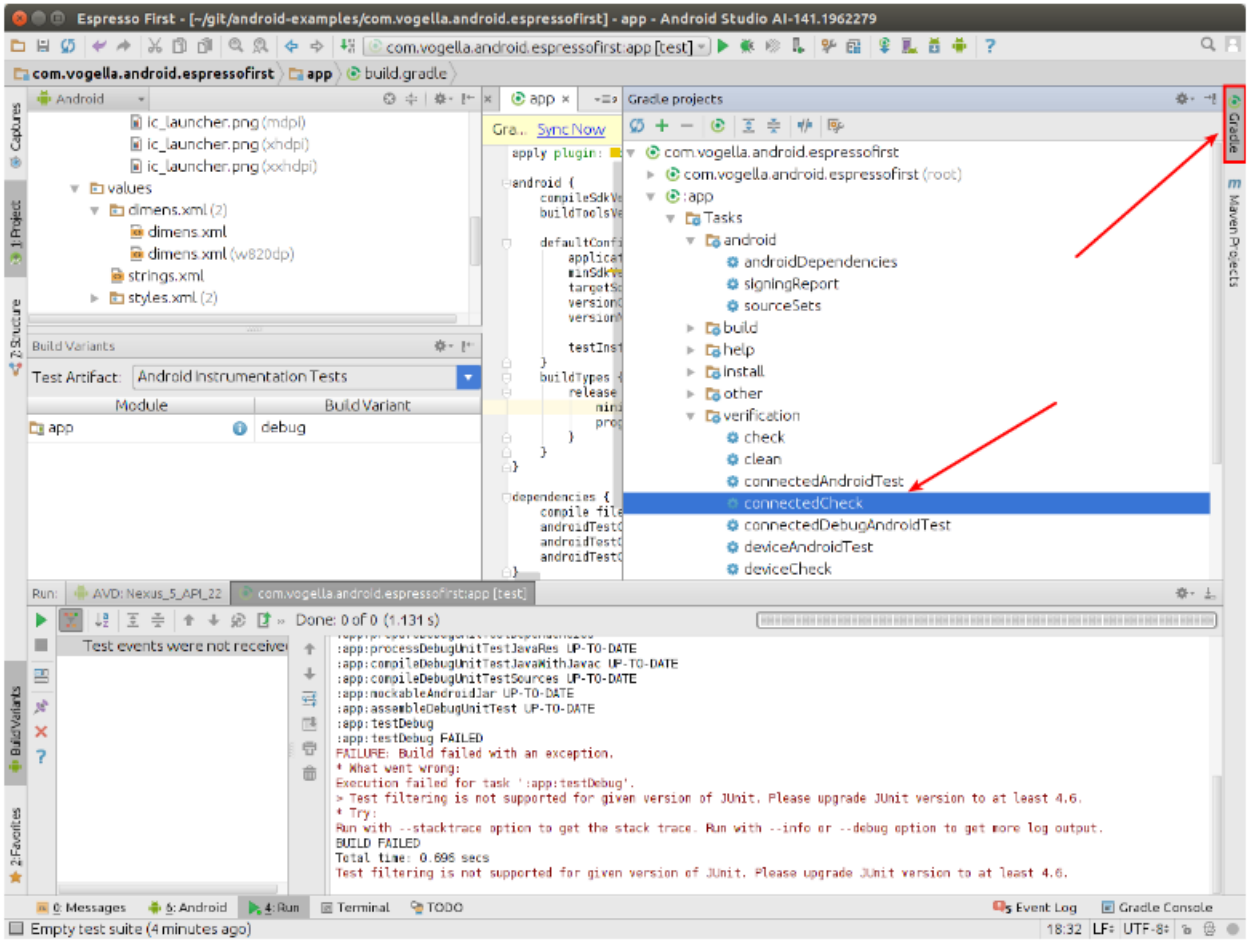


2-34-7-استفاده از سیستم کامپایل Gradle

بر روی آیکون کوچک Gradle در نوار کناری سمت راست محیط کلیک نموده و سپس گزینه ی connectedCheck را انتخاب نمایید تا تست به طور مستقیم از Gradle اجرا شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



3-34-7-بررسی صحت نمایش پیغام toast

در زیر مثالی را مشاهده می کنید که در آن بررسی می شود آیا پس از کلیک بر روی آیتمی از لیست یک پیغام toast در نمایشگر ارائه می شود یا خیر.

```
package com.vogella.android.test.juntexamples;
import android.support.test.rule.ActivityTestRule;
import android.support.test.runner.AndroidJUnit4;
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
import static android.support.test.espresso.Espresso.onData;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.assertion.ViewAssertions.matches;
import static android.support.test.espresso.matcher.RootMatchers.withDecorView;
import static android.support.test.espresso.matcher.ViewMatchers.isDisplayed;
```

```
import static android.support.test.espresso.matcher.ViewMatchers.withText;
import static org.hamcrest.CoreMatchers.containsString;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.CoreMatchers.startsWith;
import static org.hamcrest.Matchers.greaterThan;
import static org.hamcrest.Matchers.hasToString;
import static org.hamcrest.Matchers.instanceOf;
import static org.hamcrest.Matchers.not;
import static org.hamcrest.Matchers.notNullValue;
import static org.junit.Assert.assertThat;
@RunWith(AndroidJUnit4.class)
public class MainActivityTestList {
   @Rule
  public ActivityTestRule<MainActivity> rule  = new  ActivityTestRule<>(MainActivity.class);
   @Test
   public void ensureListViewIsPresent() throws Exception {
      onData(hasToString(containsString("Frodo"))).perform(click());
      onView(withText(startsWith("Clicked:"))).
      inRoot(withDecorView(
            not(is(rule.getActivity().
            getWindow().getDecorView())))).
            check(matches(isDisplayed()));
   }
}
```

35-7-شبیه سازی آبجکت های Intent با توابع کتابخانه ای Espresso

Espresso این اجازه را به توسعه دهنده می دهد تا اّبجکت های Intent را شبیه سازی / mock کند (mock = تعریف یک آبجکت intent ساختگی که رفتار آبجکت واقعی intent را شبیه سازی می کند). به واسطه ی این امکان توسعه دهنده قادر خواهد بود بررسی کند آیا کلاس activity آبجکت های intent مد نظر را صادر می کند یا خیر و نیز اینکه پس از دریافت خروجی مرتبط intent ، واکنش داخواه را نشان می دهد یا خیر.

فریم ورک تست گیری Espresso، این intent های ساختگی را در قالب کتابخانه ای به نام com.android.support.test.espresso:espresso-intents در اختیار توسعه دهنده قرار می دهد. لازم است تنظیمات لازم را نیز در فایل Gradle build اعمال نمایید (جهت دریافت اطلاعات کامل به بخش مربوطه در مقاله ی حاضر مراجعه فرمایید).

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

اگر می خواهید از intent های ساختگی Espresso در تست های نرم افزاری خود استفاده کنید، لازم است بجای ActivityTestRule ، از IntentTestRuleاستفاده نمایید.

```
package testing.android.vogella.com.simpleactivity;
import android.support.test.espresso.intent.rule.IntentsTestRule;
import android.support.test.runner.AndroidJUnit4;
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.intent.Intents.intended;
import static android.support.test.espresso.intent.matcher.IntentMatchers.toPackage;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
@RunWith(AndroidJUnit4.class)
public class TestIntent {
   @Rule
   public IntentsTestRule<MainActivity> mActivityRule =
         new IntentsTestRule<>(MainActivity.class);
   @Test
   public void triggerIntentTest() {
      onView(withId(R.id.button)).perform(click());
      intended(toPackage("testing.android.vogella.com.simpleactivity"));
   }
}
```

7-36-نوشتن تست برای بررسی عملکرد intent

1-36-7-ایجاد پروژه ی آزمایشی (project under test)

یک پروژه ی جدید اندروید با اسم پکیج testing.android.vogella.com.simpleactivity بر اساس قالب آماده ی Empty Activity ایجاد نمایید.

حال با طی نمودن این مسیر: File ▸ New ▸ Activity ▸ Empty Activity، یک activity جدید به نام SecondActivity به پروژه ی خود اضافه نمایید. این activity بایستی جهت تنظیم ظاهر خود یک فایل layout با حداقل یک المان TextView را فراخوانی نماید. شناسه (id) المان رابط کاربری TextView باید "resultText" و مقدار متنی آن بایستی بر روی "Started" تنظیم شده باشد.

یک فیلد EditText به فایل layout که ظاهر کلاس MainActivity را در UI تنظیم می کند، اضافه نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

اکنون یک دکمه به فایل layout مورد استفاده ی MainActivity اضافه نمایید. پس از کلیک بر روی دکمه، activity دوم بایستی فراخوانی و اجرا شود.

فیلد EditText را به واسطه ی فراخوانی دستور intent.putExtra();، به عنوان کلید داخل آبجکت Intent قرار دهید. سپس String/مقدار رشته ای http://www.vogella.com را نیز با توجه به کلید "URL" و به وسیله ی پل ارتباطی extra (با ارسال آن به عنوان پارامتر اول به دستور intent.putExtra()) داخل آبّجکت intent قرار دهید.

در زیر پیاده سازی نمونه از کلاس MainActivity را مشاهده می کنید.

```
package testing.android.vogella.com.simpleactivity;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
public class MainActivity extends Activity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
  }
  public void onClick(View view) {
    Intent intent = new Intent(this, SecondActivity.class);
    intent.putExtra("URL", "http://www.vogella.com");
    startActivity(intent);
  }
}
```

نوشتن تست

حال یک تست مبتنی بر فریم ورک Espresso بنویسید که موارد زیر را آزمایش کرده و از عملکرد صحیح اطمینان حاصل می نماید:

- بررسی کند آیا کلاس MainActivity یک المان دکمه با اسم یا شناسه ی R.id.button را دارد یا خیر.
- مطمئن شود مقدار نمایش داده شده بر روی دکمه رشته ی "Start new activity" می باشد.

- بررسی صحت فراخوانی متد getActivity.onClick() و اینکه آیا intent صحیح فراخوانی شده است یا خیر. این intent بایستی دستور String extra (hasExtra("URL", "http://www.vogella.com") را در خود محصور داشته باشد.

2-36-7-بررسی صحت تست (اعتبارسنجی)

کد آزمایشی شما بایستی محتوایی مشابه زیر داشته باشد.

```
package testing.android.vogella.com.simpleactivity;
import android.support.test.espresso.intent.rule.IntentsTestRule;
import android.support.test.runner.AndroidJUnit4;
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.assertion.ViewAssertions.matches;
import static android.support.test.espresso.intent.Intents.intended;
import static android.support.test.espresso.intent.matcher.IntentMatchers.hasExtra;
import static android.support.test.espresso.intent.matcher.IntentMatchers.toPackage;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
import static android.support.test.espresso.matcher.ViewMatchers.withText;
import static org.hamcrest.core.IsNull.notNullValue;
@RunWith(AndroidJUnit4.class)
public class TestIntent {
   @Rule
   public IntentsTestRule<MainActivity> mActivityRule = new IntentsTestRule<>(MainActivity.class);
   @Test
   public void triggerIntentTest() {
      // check that the button is there
      onView(withId(R.id.button)).check(matches(notNullValue() ));
      onView(withId(R.id.button)).check(matches(withText("Start new activity")));
      onView(withId(R.id.button)).perform(click());
      intended(toPackage("testing.android.vogella.com.simpleactivity"));
      intended(hasExtra("URL", "http://www.vogella.com"));
   }
}
```

تمرین: طراحی تست های functional (تست بخش های برنامه به صورت ایزوله) برای بررسی عملکرد صحیح activity

هدف از این تمرین

در تمرین حاضر، یک activity دیگر را فراخوانی کرده و اطمینان حاصل می کنید که activity مورد نظر به درستی اجرا می شود.

3-36-7-طراحی تست های functional برای activity ها

```
package testing.android.vogella.com.simpleactivity;
import android.support.test.rule.ActivityTestRule;
import android.support.test.runner.AndroidJUnit4;
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.assertion.ViewAssertions.matches;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
import static android.support.test.espresso.matcher.ViewMatchers.withText;
@RunWith(AndroidJUnit4.class)
public class TestSecondActivityIsStarted {
  @Rule
  public ActivityTestRule<MainActivity> mActivityRule = new ActivityTestRule<>(MainActivity.class);
  @Test
  public void validateSecondActivity() {
    // check that the button is there
    onView(withId(R.id.button)).perform(click());
    onView(withId(R.id.resultText))
        .check(matches(withText(("Started"))));
  }
}
```

برای بررسی صحت اطمینان ویرایش مستقیم view، لازم است کلاس تست گیری زیر را برای آزمایش عملکرد کلاس SecondActivity ایجاد نمایید.

```
package testing.android.vogella.com.simpleactivity;
import android.support.test.rule.ActivityTestRule;
import android.support.test.runner.AndroidJUnit4;
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.Espresso.pressBack;
```

```
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.assertion.ViewAssertions.matches;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
import static android.support.test.espresso.matcher.ViewMatchers.withText;
@RunWith(AndroidJUnit4.class)
public class SecondActivityFunctionalTest {
   @Rule
   public ActivityTestRule<MainActivity> mActivityRule = new ActivityTestRule<>(MainActivity.class);
   @Test
   public void validateSecondActivity() {
      // check that the button is there
      onView(withId(R.id.button)).perform(click());
      onView(withId(R.id.resultText))
            .check(matches(withText(("Started"))));
      pressBack();
      onView(withId(R.id.button))
            .check(matches(withText(("Start new activity"))));
   }
}
```

7-37-آزمایش عملکرد کد ناهزمان با استفاده از فریم ورک تست گیری Espresso

تست عملکرد کد ناهمزمان بدون کمک فریم ورک، امری بسیار پیچیده و ملال آور است. قبل از عرضه ی این فریم ورک، توسعه دهنده مجبور می شد مدت زمان مشخصی را جهت تست کد ناهمزمان صبر نماید و یا به واسطه ی استفاده از نمونه ی کلاس CountDownLatch در test code (کد آزمایشی)، از پردازش ناهمزمان اعلان نماید که پروسه به پایان رسیده است. از آنجایی که Espresso به صورت خودکار بر روی thread pool (انباره ای که thread ها در آن قرار دارند و تسک هایی که اغلب در قالب یک queue یا صف سازمان دهی شده اند را انجام می دهند) مربوط به AsynchronousTask نظارت دارد، این کار بسیار آسان می شود (کلاس AsyncTask یک کلاس abstract در چارچوب نرام افزاری اندروید است که تسک های زمان بر را در پس زمینه اجرا کرده و نتیجه را بدون اینکه بر روی thread اصلی تاثیر داشته باشد، در آن (UI thread) به صورت بهینه نمایش دهد). علاوه بر آن، Espresso صفی که event های مربوط به UI اپلیکیشن در آن قرار گرفته اند را رصد کرده و سپس تنها زمانی اجرای تست را از سرمی گیرد که هیچ تسکی در حال اجرا نباشد.

در صورت استفاده از منابع متفرقه همچون IntentService، بایستی اینترفیس IdlingResource را پیاده سازی نمایید. پیاده سازی interface مذکور بایستی بر این resource نظارت داشته و آن را برای فریم ورک Espresso معرفی نماید (IntentService یک کلاس پایه است که برای پیاده سازی کامپوننت سرویس بایستی از آن ارث بری نمایید و وظیفه ی آن مدیریت درخواست های (ارائه شده در قالب آبجکت های intent) ناهمزمان می باشد).

(IdlingResource نشانگر یکی از منابع اپلیکیشن مورد تست است که امکانی را فراهم می کند تا همزمان با اجرای تست، عملیات پس زمینه ای ناهمزمان اجرا شود. برای مثال می توان به intent resource اشاره کرد که کلیک بر روی یک دکمه را پردازش می کند. به صورت پیش فرض، فریم ورک Espresso تمامی عملیات مربوط به view را علاوه بر AsyncTasks با UI Thread هماهنگ می سازد. اما، برای منابع hand-made هیچ راهی برای انجام این کار در اختیار ندارد. در چنین سناریوی، طراح تست می تواند منابع اختصاصی را به Espresso معرفی کند. پس از آن، Espresso صبر می کند تا resource مربوطه بی کار شود و بعد عملیات view را اجرا می نماید).

```
package com.vogella.android.espressointentservice;
import android.app.ActivityManager;
import android.content.Context;
import android.support.test.espresso.IdlingResource;
import java.util.List;
public class IntentServiceIdlingResource implements IdlingResource {
  ResourceCallback resourceCallback;
  private Context context;
  public IntentServiceIdlingResource(Context context) {
    this.context = context;
  }
  @Override
  public String getName() {
    return IntentServiceIdlingResource.class.getName();
  }
  @Override
  public void registerIdleTransitionCallback(
      ResourceCallback resourceCallback) {
    this.resourceCallback = resourceCallback;
  }
  @Override
  public boolean isIdleNow() {
    boolean idle = !isIntentServiceRunning();
    if (idle && resourceCallback != null) {
      resourceCallback.onTransitionToIdle();
```

```
        }
        return idle;
    }
    private boolean isIntentServiceRunning() {
        ActivityManager manager =
                (ActivityManager) context.getSystemService(Context.ACTIVITY_SERVICE);
        // Get all running services
        List<ActivityManager.RunningServiceInfo> runningServices =
                manager.getRunningServices(Integer.MAX_VALUE);
        // check if our is running
        for (ActivityManager.RunningServiceInfo info : runningServices) {
            if (MyIntentService.class.getName().equals(
                    info.service.getClassName())) {
                return true;
            }
        }
        return false;
    }
}
package com.vogella.android.espressointentservice;
import android.app.Instrumentation;
import android.content.Context;
import android.support.test.InstrumentationRegistry;
import android.support.test.espresso.Espresso;
import android.support.test.rule.ActivityTestRule;
import android.support.test.runner.AndroidJUnit4;
import org.junit.After;
import org.junit.Before;
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.assertion.ViewAssertions.matches;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
import static android.support.test.espresso.matcher.ViewMatchers.withText;
import static org.hamcrest.Matchers.notNullValue;
@RunWith(AndroidJUnit4.class)
public class IntegrationTest {
    @Rule
    public ActivityTestRule rule = new ActivityTestRule(MainActivity.class);
    IntentServiceIdlingResource idlingResource;
    @Before
    public void before() {
        Instrumentation instrumentation
                = InstrumentationRegistry.getInstrumentation();
        Context ctx = instrumentation.getTargetContext();
        idlingResource = new IntentServiceIdlingResource(ctx);
        Espresso.registerIdlingResources(idlingResource);
    }
    @After
    public void after() {
```

```
        Espresso.unregisterIdlingResources(idlingResource);
    }
    @Test
    public void runSequence() {
        // this triggers our intent service, as we registered
        // Espresso for it, Espresso wait for it to finish
        onView(withId(R.id.action_settings)).perform(click());
        onView(withText("Broadcast")).check(matches(notNullValue()));
    }
}
```

تمرین: تست کاربردی کد ناهمزمان با استفاده از Espresso

یک پروژه جدید با اسم پکیج testing.android.vogella.com.asynctask ایجاد نمایید. پروژه حاضر بایستی این امکان را فراهم کند تا با کلیک بر روی دکمه، یک نمونه از AsyncTask جهت مدیریت عملیات در پس زمینه به طور ناهمگام، فراخوانی شود.

نمونه ای از پیاده سازی activity:

```
package testing.android.vogella.com.asynctask;
import android.app.Activity;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void onClick(View view) {
        TextView textView = (TextView) findViewById(R.id.text);
        textView.setText("Running");
        myTask.execute("test");
    }
    final AsyncTask<String, Void, String> myTask = new AsyncTask<String, Void, String>() {
        @Override
        protected String doInBackground(String... arg0) {
            return "Long running stuff";
        }
        @Override
        protected void onPostExecute(String result) {
            TextView textView = (TextView) findViewById(R.id.text);
            textView.setText("Done");
```

```
    }
  };
}
```

کد پیاده سازی شده برای تست:

```
package testing.android.vogella.com.asynctask;
import android.content.Context;
import android.support.test.InstrumentationRegistry;
import android.support.test.rule.ActivityTestRule;
import android.support.test.runner.AndroidJUnit4;
import org.junit.Rule;
import org.junit.Test;
import org.junit.rules.TestName;
import org.junit.runner.RunWith;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.assertion.ViewAssertions.matches;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
import static android.support.test.espresso.matcher.ViewMatchers.withText;
@RunWith(AndroidJUnit4.class)
public class EspressoTest {
  @Rule
  public ActivityTestRule<MainActivity> mActivityRule =
      new ActivityTestRule<>(MainActivity.class);
  @Test
  public void buttonShouldUpdateText(){
    onView(withId(R.id.update)).perform(click());
    onView(withId(R.id.text)).check(matches(withText("Done")));
  }
}
```

تمرین: تعریف matcher اختصاصی برای Espresso

(matcher = یک موتور است که رشته ای از حروف را بر اساس الگو یا pattern تفسیر شده، بررسی کرده و سعی بر یافتن موارد منطبق دارد. matcher در واقع با فراخوانی متد matcher از pattern مورد نظر، ایجاد می شود.)

اندروید با ارائه ی کلاسی به نام BoundedMatcher به شما امکان می دهد تا view matcher های مورد نظر (matcher هایی بنویسید که با view مورد نظر منطبق باشد) را ویژه ی view type های (جهت انطباق با view ای از نوع مورد نظر همچون textbox یا button) معین اعلان نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

حال یک matcher بنویسید که صحت عملکرد متن های راهنمای (text hint) فیلد EditText را بررسی می کند.

```
public static Matcher<View> withItemHint(String itemHintText) {
  checkArgument(!(itemHintText.equals(null)));
  return withItemHint(is(itemHintText));
}
public static Matcher<View> withItemHint(final Matcher<String> matcherText) {
  // use preconditions to fail fast when a test is creating an invalid matcher.
  checkNotNull(matcherText);
  return new BoundedMatcher<View, EditText>(EditText.class) {
    @Override
    public void describeTo(Description description) {
      description.appendText("with item hint: " + matcherText);
    }
    @Override
    protected boolean matchesSafely(EditText editTextField) {
      return matcherText.matches(editTextField.getHint().toString());
    }
  };
}
```

می توان از طریق کد زیر آن را انجام داد:

```
import static com.your.package.test.Matchers.withItemHint;
...
onView(withItemHint("test")).check(matches(isDisplayed()));
```

بخش هفتم :

تست تعامل بین چندین کامپوننت نرم افزاری لایه ی UI اپلیکیشن با استفاده از فریم ورک UI Automator به روش black-box

7-38-تست تعامل بین چندین کامپوننت نرم افزاری اپلیکیشن با استفاده از فریم ورک UI Automator

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آموزش حاضر به شما می آموزد چگونه حتی برای اپلیکیشن هایی که source code آن ها را در دست ندارید، تست هایی به روش black-box طراحی کنید.

7-39-تست تعامل بین کامپوننت های نرم افزاری اپلیکیشن به روش blackbox

1-39-7-استفاده از UI Automator جهت تست تعامل بین کامپوننت های اپلیکیشن (تست کل اپلیکیشن)

در مبحث حاضر تمامی کامپوننت های رابط کاربری اپلیکیشن به روش black box (functional) مورد تست قرار می گیرد.

SDK (مجموعه ابزار توسعه و طراحی اپلیکیشن) اندروید یک کتابخانه ی مبتنی بر جاوا به نام uiautomator دارد که به وسیله ی آن توسعه دهنده قادر خواهد بود تست نرم افزاری برای کامپوننت های UI اپلیکیشن نوشته و نیز با ارائه ی یک موتور امکان اجرای این تست ها را فراهم می آورد. لازم به ذکر است که برای استفاده از هر دو ابزار مزبور، بایستی ویرایش 4.3 اندروید (ورژن 18 توابع کتابخانه ای اندروید/API level) بر روی دستگاهی که تست بر روی آن اجرا می شود، نصب باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

کتابخانه ی uiautomator با ارائه ی کلاس های UiDevice، UiSelector و کلاس UiObject که بر اساس کلاس UiSelector ساخته می شود، به ترتیب قابلیت های زیر را در اختیار توسعه دهنده قرار می دهد:

1. تبادل داده و تعامل با دستگاه میزبان
2. پیدا کردن المان ها در نمایشگر
3. ساخت آبجکت های تشگیل دهنده ی ظاهر برنامه و لایه ی UI

دو کلاس UiCollection و UiScrollable نیز به ترتیب این اجازه را می دهند تا چندین المان UI را همزمان انتخاب کرده و جهت دسترسی به المان مورد نظر در view پیمایش نماید.

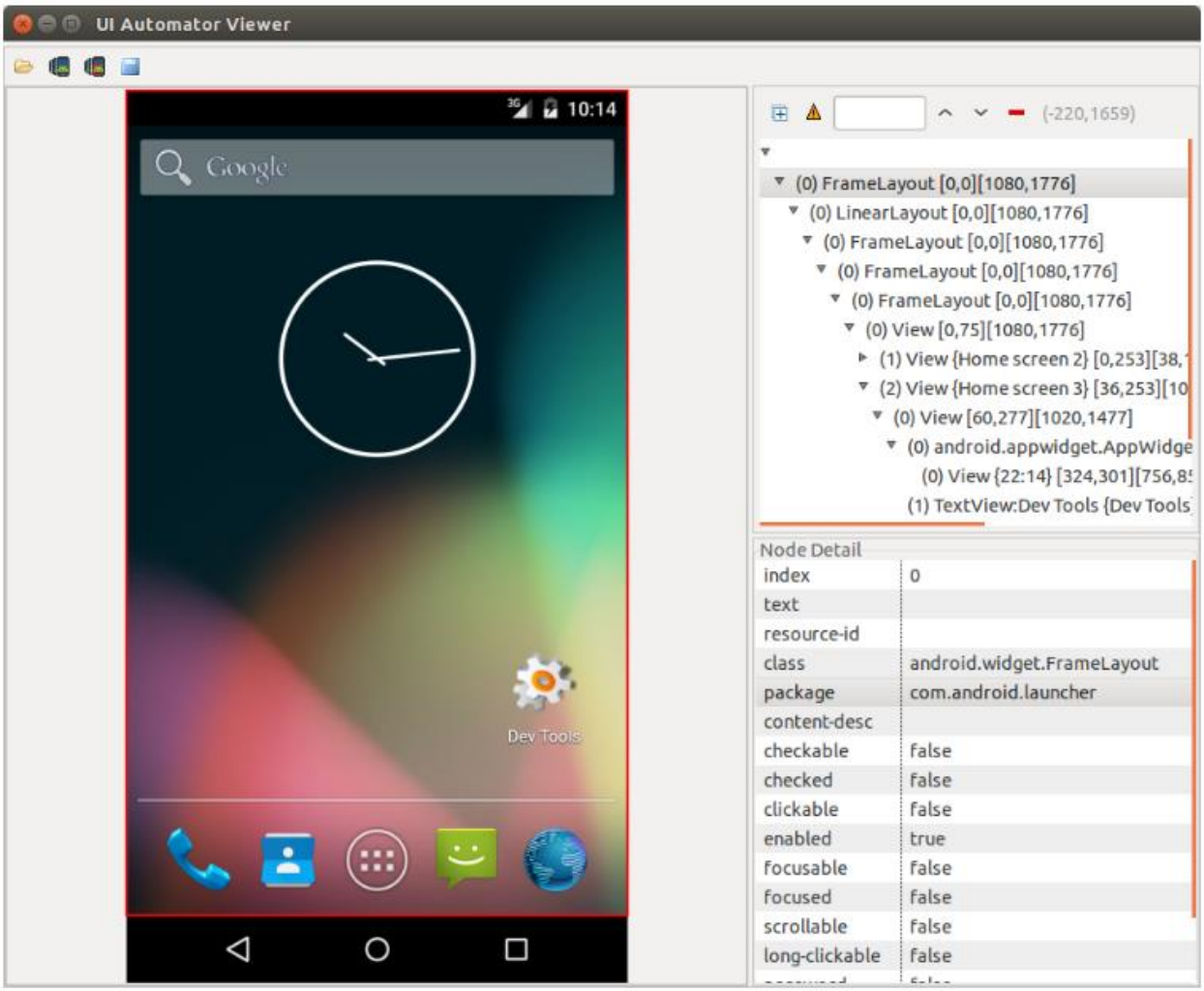
2-39-7-دسترسی به اطلاعات مربوط به view با ابزار uiautomatorviewer

چارچوب نرم افزاری اندروید ابزاری به نام uiautomatorviewer را ارائه می دهد که به وسیله ی آن شما قادر خواهید بود لایه ی UI یک اپلیکیشن را به راحتی تجزیه و تحلیل کنید. شما می توانید با استفاده از این ابزار اندیس، متن یا attribute اپلیکیشن را پیدا کنید. در واقع این ابزار به شما امکان می دهد تا به نمودار درختی (layout hierarchy) المان های رابط کاربری دسترسی داشته و property های مربوط به هریک از کنترل ها (المان های رابط کاربری) را مشاهده نمایید.

به واسطه ی این ابزار حتی افرادی که در نوشتن اپلیکیشن شرکت نداشته یا برنامه نویس نیستند نیز می توانند به راحتی برنامه را تحلیل کرده و برای آن تست طراحی کنند. این را در تصویر زیر مشاهده می کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



به منظور اجرای ابزار uiautomatorviewer، پس از ارائه ی آدرس /android-sdk/tools در خط فرمان، دستور uiautomatorviewer را درج نمایید.

دسترسی و استفاده از UI Automator برای تست اپلیکیشن

انجام تنظیمات لازم در پروژه

برای استفاده از ابزار UI Automator جهت اجرای تست بر روی اپلیکیشن، کتابخانه (dependency) زیر را به فایل build.gradle پروژه اضافه نموده و AndroidJUnitRunner را به عنوان مقدار به

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

پارامتر testInstrumentationRunner پاس دهید. در زیر این مثال را به صورت کاربردی مشاهده می کنید.

```
apply plugin: 'com.android.application'
android {
    compileSdkVersion 19
    buildToolsVersion "23.0.0"
    defaultConfig {
        applicationId "com.vogella.android.test.uiautomator"
        minSdkVersion 19
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}
dependencies {
    androidTestCompile 'com.android.support.test:runner:0.3'
    androidTestCompile 'junit:junit:4.12'
    testCompile 'junit:junit:4.12'
    androidTestCompile 'org.hamcrest:hamcrest-library:1.3'
    androidTestCompile 'com.android.support.test:runner:0.3'
    androidTestCompile 'com.android.support.test:rules:0.3'
    androidTestCompile 'com.android.support.test.uiautomator:uiautomator-v18:2.1.1'
}
```

توصیه می شود جهت جلوگیری از بروز اختلال، انیمیشن را بر روی دستگاه اندرویدی که تست در بستر آن اجرا می شود، غیر فعال نمایید.

3-39-7-محل قرارگیری تست ها

تست های مبتنی بر UI Automator می بایست به صورت پیش فرض در پوشه ی app/src/androidTest جایگذاری شود.

4-39-7-نحوه ی نوشتن تست

تست های UI Automator می بایست با دستور @RunWith(AndroidJUnit4.class) نشانه گذاری شده و با استفاده از نمونه ی کلاس ActivityTestRule خود را تنظیم نمایند.

5-39-7-ساخت پروژه و تنظیم فایل Gradle build

یک پروژه ی جدید که اسم پوشه ی بالایی آن (top level project name) com.example.android.testing.uiautomator می باشد، ایجاد نمایید.

فایل Gradle build را به صورت زیر ویرایش نمایید.

```
apply plugin: 'com.android.application'
android {
   compileSdkVersion 19
   buildToolsVersion "23.0.0"
   defaultConfig {
      applicationId "com.vogella.android.test.uiautomator"
      minSdkVersion 19
      targetSdkVersion 23
      versionCode 1
      versionName "1.0"
      testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
   }
   buildTypes {
      release {
         minifyEnabled false
         proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
      }
   }
}
dependencies {
   androidTestCompile 'com.android.support.test:runner:0.3'
   androidTestCompile 'junit:junit:4.12'
   testCompile 'junit:junit:4.12'
   androidTestCompile 'org.hamcrest:hamcrest-library:1.3'
   androidTestCompile 'com.android.support.test:runner:0.3'
```

```
    androidTestCompile 'com.android.support.test:rules:0.3'
    androidTestCompile 'com.android.support.test.uiautomator:uiautomator-v18:2.1.1'
}
```

ایجاد تست

تستی با پیاده سازی زیر را در پوشه ی androidTest_ تعریف نمایید.

```
package com.vogella.android.test.uiautomator;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.content.pm.ResolveInfo;
import android.support.test.InstrumentationRegistry;
import android.support.test.rule.ActivityTestRule;
import android.support.test.runner.AndroidJUnit4;
import android.support.test.uiautomator.By;
import android.support.test.uiautomator.UiDevice;
import android.support.test.uiautomator.UiObject;
import android.support.test.uiautomator.UiObjectNotFoundException;
import android.support.test.uiautomator.UiScrollable;
import android.support.test.uiautomator.UiSelector;
import android.support.test.uiautomator.Until;
import org.hamcrest.Matchers;
import org.junit.Before;
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
import static org.hamcrest.Matchers.*;
import static org.junit.Assert.*;
@RunWith(AndroidJUnit4.class)
public class MyUiAutomatorTest {
    @Rule
    public ActivityTestRule<MainActivity> mActivityRule
            = new ActivityTestRule<MainActivity>(MainActivity.class);
    private UiDevice mDevice;
    @Before
    public void setUp() {
        // Initialize UiDevice instance
        mDevice = UiDevice.getInstance(InstrumentationRegistry.getInstrumentation());
        // Start from the home screen
        mDevice.pressHome();
        mDevice.wait(Until.hasObject(By.pkg(getLauncherPackageName()).depth(0)), 1000);
    }
    @Test
    public void checkSettings() throws UiObjectNotFoundException {
        // Simulate a short press on the HOME button.
        mDevice.pressHome();
        // We're now in the home screen. Next, we want to simulate
        // a user bringing up the All Apps screen.
        // If you use the uiautomatorviewer tool to capture a snapshot
```

```
        // of the Home screen, notice that the All Apps button's
        // content-description property has the value "Apps". We can
        // use this property to create a UiSelector to find the button.
        UiObject allAppsButton = mDevice.findObject(new UiSelector().description("Apps"));
        // Simulate a click to bring up the All Apps screen.
        allAppsButton.clickAndWaitForNewWindow();
        // In the All Apps screen, the Settings app is located in
        // the Apps tab. To simulate the user bringing up the Apps tab,
        // we create a UiSelector to find a tab with the text
        // label "Apps".
        UiObject appsTab = mDevice.findObject(new UiSelector().text("Apps"));
        // Simulate a click to enter the Apps tab.
        appsTab.click();
        // Next, in the apps tabs, we can simulate a user swiping until
        // they come to the Settings app icon. Since the container view
        // is scrollable, we can use a UiScrollable object.
        UiScrollable appViews = new UiScrollable(
                new UiSelector().scrollable(true));
        // Set the swiping mode to horizontal (the default is vertical)
        appViews.setAsHorizontalList();
        // create a UiSelector to find the Settings app and simulate
        // a user click to launch the app.
        UiObject settingsApp = appViews
                .getChildByText(new UiSelector()
                        .className(android.widget.TextView.class.getName()),
                        "Settings");
        settingsApp.clickAndWaitForNewWindow();
        // Validate that the package name is the expected one
        UiObject settingsValidation = new UiObject(
                new UiSelector()
                        .packageName("com.android.settings"));
        assertThat(settingsValidation.exists(), equalTo(true));
    }
    private String getLauncherPackageName() {
        // Create launcher Intent
        final Intent intent = new Intent(Intent.ACTION_MAIN);
        intent.addCategory(Intent.CATEGORY_HOME);
        // Use PackageManager to get the launcher package name
        PackageManager pm = InstrumentationRegistry.getContext().getPackageManager();
        ResolveInfo resolveInfo = pm.resolveActivity(intent, PackageManager.MATCH_DEFAULT_ONLY);
        return resolveInfo.activityInfo.packageName;
    }
}
```

اجرا و بررسی صحت اجرای موفقیت آمیز تست

تست را اجرا کرده و از عملکرد صحیح آن اطمینان حاصل نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش هشتم :

تست لایه ی UI اپلیکیشن/رابط کاربری با استفاده از فریم ورک تست گیری Robotium

آموزش حاضر نحوه ی تست لایه ی UI اپلیکیشن/اجرای تست بر روی لایه ی UI اپلیکیشن اندرویدی را شرح می دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Robotium-7-40

Robotium یک افزونه برای فریم ورک تست گیری اندروید است و صرفا برای آسان سازی نوشتن تست های نرم افزاری برای UI اپلیکیشن طراحی شده است. تست های Robotium اعضای کلاس ActivityInstrumentationTestCase2 را به ارث برده و به شما این امکان را می دهد تا test case هایی (مورد آزمایش) بنویسید که چندین activity را شامل می شود.

نکته: تست هایی که بر اساس Robotium طراحی می شوند منحصرا با لایه UI اپلیکیشن تعامل دارند. به عبارت دیگر این نوع تست ها فرض می گیرند که اپلیکیشن مورد آزمایش یک جعبه ی سیاه است که محتویات و کد داخل آن اطلاعاتی ندارد.

Robotium، همان طور که قبلا ذکر شد، نوعی افزونه برای فریم ورک تست گیری اندروید می باشد (چارچوب طراحی تست نرم افزاری اندروید را بسط می دهد) که نوشتن تست برای اپلیکیشن های اندرویدی را آسان می سازد.

کلاس اصلی فریم ورک مزبور برای طراحی تست، Solo می باشد. این کلاس به مجرد ساخت test case و تست activity مقداردهی اولیه می شود.

Robotium نصب-7-41

جهت تست Robotium در پروژه ی اندرویدی خود، لازم است کتابخانه (dependency) مورد نیاز آخرین ویرایش Robotium را به فایل build اضافه نمایید.

```
dependencies {
   // Unit testing dependencies
   androidTestCompile 'com.jayway.android.robotium:robotium:5.4.12'
}
```

در زمان تنظیم مقاله ی حاضر، 5.4.1 جدیدترین ویرایش Rbotium می باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

7-42-نمونه ای از پیاده سازی تست های Robotium

کد زیر نحوه ی پیاده سازی و استفاده از فریم ورک Robotium در تست یک activity (تست لایه ی رابط کاربری اپلیکیشن) را نمایش می دهد.

```
package de.vogella.android.test.target.test;
import junit.framework.Assert;
import android.test.ActivityInstrumentationTestCase2;
import com.robotium.solo.Solo;
import de.vogella.android.test.target.SimpleActivity;
import de.vogella.android.test.target.SimpleListActivity;
public class SimpleActivityTest extends
        ActivityInstrumentationTestCase2<SimpleActivity> {
    private Solo solo;
    public SimpleActivityTest() {
        super(SimpleActivity.class);
    }
    public void setUp() throws Exception {
        solo = new Solo(getInstrumentation(), getActivity());
    }
    @Override
    public void tearDown() throws Exception {
        solo.finishOpenedActivities();
    }
}
```

در زیر نمونه ی دیگری از کاربرد متد تست گیری Robotium را مشاهده می کنید که در آن پیاده سازی یک لیست مورد آزمایش قرار می گیرد.

```
// check that we have the right activity
solo.assertCurrentActivity("wrong activity", SimpleActivity.class);
// Click a button which will start a new Activity
// Here we use the ID of the string to find the right button
solo.clickOnButton(solo.getString(R.string.button1));
// Validate that the Activity is the correct one
solo.assertCurrentActivity("wrong activity", SimpleListActivity.class);
solo.clickInList(1);
// searchForText has a timeout of 5 seconds
assertTrue(solo.waitForText("Android")); // Assertion
solo.clickInList(2);
assertTrue(solo.waitForText("iPhone")); // Assertion
solo.clickInList(3);
assertTrue(solo.waitForText("Blackberry")); // Assertion
```

```
solo.goBack();
solo.clickOnButton("Button2");
solo.clickOnButton("Button3");
// open the menu
solo.sendKey(Solo.MENU);
solo.clickOnText("Preferences");
solo.clickOnText("User");
solo.clearEditText(0);
Assert.assertTrue(solo.searchText(""));
solo.enterText(0, "http//:www.vogella.com");
Assert.assertTrue(solo.searchText("http//:www.vogella.com"));
solo.goBack();
```

43-7-توابع کتابخانه ای / Robotium API

Solo توابعی برای فراخوانی کامپوننت های رابط کاربری (کامپوننت های UI اپلیکیشن اندرویدی) در اختیار توسعه قرار می دهد. در جدول زیر تعدادی از این توابع را همراه با شرح کاربرد مشاهده می نمایید.

متد	شرح
getView(int id)	بر اساس (پارامتر) ID ارائه شده، view مد نظر را در activity جاری پیدا کرده و به عنوان خروجی برمی گرداند.
assertCurrentActivity(text, Activity.class)	انتظار دارد که activity جاری با دومین پارامتر ارسالی به تابع، برابر باشد.
getCurrentActivity() .getFragmentManager() .findFragmentById()	Fragment مورد نظر را پیدا کرده و در خروجی برمی گرداند.
waitForText(text)	به مدت 5 ثانیه برای مقدار متنی مورد نظر (ارائه شده به عنوان پارامتر) در UI اپلیکیشن جاری جستجو می کند (منتظر وارد شدن متن در صفحه می شود).
clickOnButton(text)	بر روی کنترل دکمه ی ای که مقدار متنی "text" را دارد کلیک می کند.
sendKey(Solo.MENU);	Sends the menu key event.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

	Event کلید منو را ارسال می کند.
clickOnText(text)	در رابط کاربری اپلیکیشن جاری، مقدار متنی که به عنوان پارامتر به تابع ارسال شده را در خروجی برمی گرداند.
enterText()	یک متن ساده در رابط کاربری جاری وارد می کند.
searchText(text)	در رابط کاربری جاری، متنی که به عنوان پارامتر به تابع حاضر ارسال شده را جستجو کرده و در صورت یافتن، مقدار بولی true را بازمی گرداند.
searchButton(text)	کنترل دکمه ای که مقدار متنی (text) آن با مقدار پارامتر ارسالی منطبق می باشد را در رابط کاربری جاری پیدا می کند.
clickOnSearch()	امکان کلیک بر روی بخشی از نمایشگر را به شما می دهد.
goBack()	دکمه ی بازگشت را فشار می دهد.
setDatePicker()	یک تاریخ در کنترل DatePicker اندروید انتخاب می کند.
clickInList(x);	بر روی آیتم در لیست ساده/ListView کلیک می کند.
pressSpinnerItem(0,2);	یک آیتم در لیست کشویی/Spinner را فشار می دهد.
isCheckBoxChecked()	بررسی می کند آیا چک باکس تیک دار و فعال شده است یا خیر.
takeScreenshot()	یک عکس (screenshot) از وضعیت جاری تهیه کرده و داخل حافظه ی دستگاه میزبان در آدرس /sdcard/Robotium-Screenshots/ ذخیره می کند. لازم است جهت فراخوانی این تابع، مجوز theandroid.permission.WRITE_EXTERNAL_STORAGE را در فایل تنظیمات AndroidManifest.xml مربوط به اپلیکیشن مورد تست اعلان نمایید.
waitForActivity(SecondActivity.class, 2000)	به مدت 2 ثانیه منتظر activity ارسال شده به تابع به عنوان پارامتر می ماند (چنانچه در مدت معین شده activity یافت نشد، مقدار بولی false را بازمی گرداند).

می توانید با فراخوانی متد solo.setActivityOrientation(Solo.LANDSCAPE) وضعیت نمایش (orientation) صفحه ی جاری (activity) را تنظیم نمایید.

جهت تست رشته های ترجمه شده و بین المللی (internationalized strings)، می بایست با فراخوانی متد getString(id) به فایل resources از پروژه مورد تست دسترسی پیدا کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

مثال:

```
// Here we use the ID of the string to find the right button
solo.clickOnButton(solo.getString(de.vogella.android.test.target.R.string.button1));
```

اجرای تست

تست Robotium را به همان شیوه ای که محیط کاری Eclipse را راه اندازی می کنید، با کلیک راست بر روی test class (کلاس آزمایشی) و انتخاب آیتم Run-As ▸ Android JUnit Test از منو، اجرا نمایید.

در صورت تمایل می توانید تست های Robotium را با درج دستور زیر در پنجره ی فرمان، اجرا نمایید.

```
adb shell am instrument
  -w de.vogella.android.test.tester/android.test.InstrumentationTestRunner
```

تمرین: تست نویسی با توابع Robotium

تمرین: نوشتن یک تست واقعی با Robotium

یک پروژه ی جدید اندروید به نام com.vogella.android.test.robotium.target ایجاد نمایید. در صفحه یا activity اول سه کنترل دکمه قرار دهید. پس از کلیک بر روی دکمه، با توجه به کدی که نوشته اید، می بایست activity (صفحه) دیگری فراخوانی شده و یک لیست را برای کاربر به نمایش بگذارد.

1-43-7-ساخت پروژه ی آزمایشی و اضافه کردن کتابخانه ی Robotium به آن

یک پروژه ی جدید جهت تست به نام com.vogella.android.test.robotium.targetTest ایجاد نمایید. سپس داخل آن پوشه ی جدیدی به نام libs ایجاد کرده و فایل Robotium JAR را در آن جایگذاری نمایید.

نکته: پس از جایگذاری فایل JAR در پوشه ی نام برده، مجموعه ابزار Android (android tooling) محیط کاری Eclipse خود به صورت اتوماتیک آن را به build path پروژه ی شما اضافه می کنید. لازم به ذکر است که قرار دادن فایل JAR در پوشه ای با اسم متفاوت، غالبا سبب رخداد خطای ClassNotFoundExeception برای کلاس Solo می شود.

2-43-7-ایجاد پروژه ی آزمایشی و افزودن کتابخانه ی Robotium

کلاس آزمایشی (test class) زیر را پیاده سازی نمایید.

```
package de.vogella.android.test.target.test;
import junit.framework.Assert;
import android.test.ActivityInstrumentationTestCase2;
import com.robotium.solo.Solo;
import de.vogella.android.test.target.SimpleActivity;
import de.vogella.android.test.target.SimpleListActivity;
public class SimpleActivityTest extends
        ActivityInstrumentationTestCase2<SimpleActivity> {
    private Solo solo;
    public SimpleActivityTest() {
        super(SimpleActivity.class);
    }
    public void setUp() throws Exception {
        solo = new Solo(getInstrumentation(), getActivity());
    }
    @Override
    public void tearDown() throws Exception {
        solo.finishOpenedActivities();
    }
    public void testListItemClickShouldDisplayToast() throws Exception {
        // check that we have the right activity
```

```
        solo.assertCurrentActivity("wrong activity", SimpleActivity.class);
        // Click a button which will start a new Activity
        // Here we use the ID of the string to find the right button
        solo.clickOnButton(solo
                .getString(de.vogella.android.test.target.R.string.button1));
        // assert that the current activity is the SimpleListActivity.class
        solo.assertCurrentActivity("wrong activity", SimpleListActivity.class);
        solo.clickInList(1);
        // searchForText has a timeout of 5 seconds
        assertTrue(solo.waitForText("Android")); // Assertion
        solo.clickInList(2);
        assertTrue(solo.waitForText("iPhone")); // Assertion
        solo.clickInList(3);
        assertTrue(solo.waitForText("Blackberry")); // Assertion
        solo.goBack();
        solo.clickOnButton("Button2");
        solo.clickOnButton("Button3");
    }
    public void testListItemClickShouldDisplayToast() throws Exception {
        // open the menu
        solo.sendKey(Solo.MENU);
        solo.clickOnText("Preferences");
        solo.clickOnText("User");
        solo.clearEditText(0);
        Assert.assertTrue(solo.searchText(""));
        solo.enterText(0, "http//:www.vogella.com");
  Assert.assertTrue(solo.searchText("http//:www.vogella.com"));
        solo.goBack();
    }
}
```

برطرف کردن خطاهای اپلیکیشن

فرض کنید تست با تنظمیات و پیاده سازی شما از عملکرد صحیح برخوردار می باشد ولی اپلیکیشن نتیجه ی مورد انتظار را ارائه نمی دهد. اپلیکیشن را طوری ویرایش کنید که تست با موفقیت انجام شود.

بخش نهم :

استفاده از فریم ورک Dagger 2 جهت تزریق نیازمندی ها به پروژه (کلاس ها و کتابخانه های مورد نیاز)/ Dagger 2 & Dependency injection

مقاله ی حاضر به شرح نحوه ی استفاده از فریم ورک Dagger 2 جهت تزریق نیازمندی های پروژه به آن در اپلیکیشن های متعارف جاوا و اندروید می پردازد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

7-44-مقدمه ای بر مفهوم dependency injection >> استفاده از dependency injection در محیط و بستر اجرای Java

1-44-7-Dependency injection چیست؟

Dependency injection امروزه مفهومی است که در بسیاری از زبان های برنامه نویسی پیاده سازی می شود. مفهوم کلی تری که dependency injection بر آن پایه ریزی شده است، Inversion of Control می باشد. در مهندسی نرم‌افزار، وارونگی کنترل گونه‌ای از توسعه و طراحی اپلیکیشن را توصیف می‌کند که بخشهای اختصاصی نوشته شده یک برنامه رایانه‌ای جریان کنترل را از یک کتابخانه با قابلیت بازاستفاده عمومی دریافت می‌کند. معماری نرم‌افزاری ای که از این طرح به وجود می‌آید در مقایسه با برنامه نویسی سنتی رویه‌ای کنترل را وارونه می‌سازد: در برنامه نویسی سنتی کد اختصاصی که هدف برنامه را بیان می‌کند، برای مراقبت از وظایف عمومی کتابخانه‌های بازقابل استفاده را فرا می‌خواند اما با وراونه سازی کنترل این کدهای دارای قابلیت باز استفاده هستند که کدهای اختصاصی یا کدهای مربوط به وظیفه خاص را صدا می زنند. وارونگی کنترل برای افزایش ویژگی پودمانی/پیمانه ای (modularity) برنامه و قابلیت گسترش آن استفاده شده و در پارادایم برنامه نویسی شی گرا و دیگر الگوهای برنامه نویسی هم مورد استفاده قرار می گیرد.

Dependency Injection (فراهم نمودن نیازمندی های یک آبجکت) در برنامه‌نویسی شی گرا، الگوی توسعه است با قاعده ی اصلیِ جداکردنِ رفتار از تحلیلِ نیازمندی (Dependency Resolution) = روشی برای تجزیه‌کردنِ اجزا کاملا مستقلِ نرم‌افزاری. به صورت خلاصه Dependency injection، الگویی است جهت تزریق نیازمندی ها و آبجکت های مورد نیاز خارجی یک کلاس به آن، بجای استفاده مستقیم از آن نیازمندی ها (فراهم کردن آبجکت مورد نظر) در داخل کلاس. با توجه به این مفهوم، یک کلاس نباید نیازمندی ها (dependency) خود را به صورت static تنظیم کند، بلکه می بایست آن را از بیرون و توسط کلاس های دیگر تامین نماید.

چنانچه یک کلاس جاوا از نمونه ی کلاس دیگری استفاده کند، گفته می شود که به آن کلاس وابستگی دارد. در اصطلاح این امر class dependency (وابستگی یک کلاس به کلاس خارجی) خوانده می شود. به عنوان مثال، کلاسی که به قابلیت های سرویس logger (ثبت گزارش) احتیاج داشته باشد، در اصطلاح گفته می شود که کلاس اول به کلاس سرویس دهی (service class) احتیاج/dependency دارد یا کلاس اول قابلیت ها و نیازمندی های خود را از کلاس خارجی تامین می کند.

در حالت ایده ال، کلاس های Java بایستی تا حد امکان از دیگر کلاس های Java مستقل باشند. از این طریق، امکان استفاده ی مجدد از کلاس ها افزایش یافته و متعاقبا می توان این کلاس ها را به صورت ایزوله یا مجزا از یکدیگر به راحتی تست نموده و از عملکرد صحیح آن ها اطمینان حاصل کرد.

چنانچه کلاس Java به واسطه ی کلیدواژه ی new، نمونه ای از کلاس دیگر را ایجاد کرده و مورد استفاده قرار دهد، در آن صورت نمونه ی ایجاد شده از کلاس قابل استفاده و تست به طور مجزا از کلاس میزبان نخواهد بود. از این رخداد تحت عنوان hard dependency (وابستگی شدید) یاد می شود.

```
package com.example.e4.rcp.todo.parts;
import java.util.logging.Logger;
public class MyClass {
	private final static Logger logger;
	public MyClass(Logger logger) {
		this.logger = logger;
		// write an info log message
		logger.info("This is a log message.")
	}
}
```

نکته: لازم به توضیح است که کلاس حاضر، صرفا یک کلاس ساده ی جاوا بوده (هیچ ویژگی خاصی ندارد)، جز اینکه از ایجاد مسقیم آبجکت جلوگیری می کند.

کلاس framework که گاه dependency container نیز خوانده می شود، قادر است نیازمندهای این کلاس را به راحتی تحلیل و فراهم نماید (Dependency injection container یک کلاس است که می داند چگونه آبجکت ها و متعاقبا آبجکت های وابسته به آن را نمونه سازی و تنظیم نماید). با

این تجزیه و تحلیل، کلاس framework قادر خواهد بود یک نمونه از کلاس مورد نظر ایجاد کرده و سپس آبجکت های مورد نیاز را به واسطه ی java reflection، به داخل dependency های مشخص شده تزریق کند.

با این روش کلاس جاوا دیگر هیچ وابستگی شدید (hard dependency) ندارد، بدین معنی که دیگر برای تامین قابلیت ها و نیازمندی های خود به نمونه ای از کلاس دیگر احتیاج ندارد. این امکان به توسعه دهنده اجازه می دهد تا کلاس خود را، برای مثال با استفاده از آبجکت های ساختگی (mock)، به صورت ایزوله آزمایش نماید.

آبجکت های ساختگی (mock) آبجکت هایی هستند که قابلیت و رفتار آبجکت واقعی را شبیه سازی می کنند و در واقع سعی دارند تا مانند آبجکت مورد نظر رفتار کنند. اما این آبجکت های ساختگی یا mock برنامه ریزی نمی شوند بلکه طوری تنظیم شود که همیشه به یک شیوه ی از پیش تعیین شده و مشخص رفتار کنند. واژه ی mock یک کلمه ی انگلیسی است که معنی شبیه سازی و تقلید را در فارسی می دهد. بنابراین یک آبجکت mock، همان طور که قبلا نیز ذکر شد، رفتار یک آبجکت واقعی را شبیه سازی و تقلید می کند.

در صورت پیاده سازی الگو توسعه ی DI در پروژه، کلاس های جاوا به راحتی به صورت ایزوله قابل تست خواهد بود.

2-44-7-استفاده از annotation ها جهت توصیف و شرح نیازمندی ها / dependency های کلاس

روش های مختلفی برای شرح و مشخص کردن dependency های یک کلاس وجود دارد. پرکاربردترین روش معمولا در توصیف مسقیم dependency ها در کلاس میزبان با استفاده از annotation های Java خلاصه می شود (نشانه و علامت گذاری نیازمندها داخل کلاس با استفاده از annotation ها).

annotation های متعارف جاوا که برای شرح و نشانه گذاری dependency های یک کلاس مورد استفاده قرار می گیرند، همگی در JSR330 اعلان شده اند. JSR330 دو دستور @Inject و @Named را برای نشانه گذاری نیازمندی ها در کلاس، در اختیار توسعه دهنده قرار می دهد.

کد زیر یک کلاس را نمایش می دهد که با استفاده از annotation ها، نیازمندی ها و dependency های خود را مستقیما داخل کلاس میزبان اعلان می نماید.

```
// import statements left out
public class MyPart {
	@Inject private Logger logger;
	// inject class for database access
	@Inject private DatabaseAccessClass dao;
	@Inject
	public void createControls(Composite parent) {
		logger.info("UI will start to build");
		Label label = new Label(parent, SWT.NONE);
		label.setText("Eclipse 4");
		Text text = new Text(parent, SWT.NONE);
		text.setText(dao.getNumber());
	}
}
```

توجه: همان طور که در کد بالا مشاهده می کنید، کلاس حاضر از کلیدواژه یا عملگر new برای نمونه سازی و ساخت کامپوننت های رابط کاربری (UI) استفاده می کند. با این کار برنامه نویس اعلان می دارد که کامپوننت های رابط کاربری (اجزایی که با عملگر new نمونه سازی شده اند) قرار نیست در آینده تست شوند یا با آبجکت های ساختگی و شبیه ساز تست ها جایگزین گردند. به عبارت دیگر با پیاده سازی فوق، برنامه نویس تصمیم گرفته که به toolkit و مجموعه ابزار UI مربوطه وابستگی شدید (hard coupling) داشته باشد.

3-44-7-بر اساس JSR330، آبجکت ها به کدام بخش از کلاس تزریق یا پاس داده می شوند؟

الگوی DI را می توان از روش های زیر در پروژه پیاده سازی کرد:

- از طریق تابع سازنده ی (Constructor) کلاس که در اصطلاح construction injection خوانده می شود.

- از طریق فیلد که در اصطلاح field injection نام دارد.
- از طریق پارامترهای یک متد که از آن تحت عنوان method injection یاد می شود.

می توان از dependency injection برای فیلدها و متدهای static و غیر static نیز استفاده کرد. لازم به ذکر است که توسعه دهندگان خبره سعی می کنند از پیاده سازی DI بر روی متدها و فیلدهای static خودداری کنند چرا که علاوه بر به همراه داشتن محدودیت هایی، اشکال زدایی و فرایند debug را بسیار سخت می کند.

- فیلدهای static پس از اینکه اولین آبجکت کلاس از طریق پیاده سازی DI ساخته شد، تزریق می شوند، بدین معنی که داخل تابع سازنده (constructor) دسترسی به آن ها امکان پذیر نخواهد بود.
- فیلدهای static را نمی توان به عنوان final علامت گذاری کرد چرا که در این صورت، کامپایلر یا خود اپلیکیشن در زمان اجرای برنامه ایراد می گیرد.
- متدهای static تنها یکبار پس از اینکه اولین نمونه از کلاس ایجاد شد، فراخوانی می شوند.

4_44_7-ترتیبی که DI بر اساس آن در کلاس اجرا و پیاده سازی می شود

بر اساس JSR330 الگوی DI به ترتیب زیر در کلاس پیاده سازی می شود:

- تزریق نیازمندی ها از طریق تابع سازنده ی کلاس (Constructor injection).
- تزریق از طریق فیلد کلاس (field injection).
- تزریق با ارسال پارامتر به متد کلاس (method injection) .

JSR330، ترتیبی که متدها یا فیلدهای نشانه گذاری شده با @Inject بر آن اساس اجرا می شوند را تعیین نمی کند. به عبارت دیگر شما نمی توانید فرض را بر این بگذارید که متدها یا فیلدها به همان ترتیبی که در کلاس اعلان شده اند، فراخوانی می شوند.

توجه: از آنجایی که فیلدها و پارامترهای متد پس از فراخوانی تابع سازنده (constructor) به داخل کلاس تزریق می شوند، امکان استفاده از متغیرهای عضو تزریق شده از بیرون در تابع سازنده وجود ندارد.

7-45-فریم ورک های جاوا و dependency injection

می توانید با تعبیه constructor ها و توابع getter/setter کافی داخل کلاس های پروژه ی خود، بدون استفاده از framework، الگو توسعه ی DI را پیاده سازی نمایید.

یک فریم ورک که برای پیاده سازی الگو DI طراحی شده است، صرفا مقداردهی اولیه کلاس ها با آبجکت های مناسب را ساده تر می سازد.

دو فریم ورک پرطرفدار جهت پیاده سازی DI عبارتند از Spring و Google Guice.

7-46-Dependency injection با فریم ورک Dagger2

7_46_1-Dagger2 چیست؟

همان طور که قبلا گفته شد، به این خاطر که ایزوله سازی آبجکت ها و فراهم کردن نیازمندی هایشان (dependency ها) به سادگی صورت پذیرد، می‌توان از فریمورک‌های مختلفی استفاده کرد که Dagger قطعا یکی از پرطرفدارترین و کاراترین آن ها است. Dagger یک فریم ورک جهت پیاده

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

سازی الگو توسعه ی DI در پروژه محسوب می شود که مبتنی بر Java Specification Request 330 (JSR) طراحی شده، از ابزار code generation استفاده می کند و مبتنی بر annotation ها می باشد. کد تولید شده خوانا بوده و اشکال زدایی آن بسیار آسان می باشد.

Dagger2 از دو annotation زیر استفاده می کند:

- @Module و @Provides: کلاس و متدهایی تعریف می کند که نیازمندی های (dependency) کلاس ها را در اختیارشان قرار می دهد.
- @Inject: dependency هایی را فراخوانی یا درخواست می کند. می توان از این دستور برای نشانه گذاری Constructor، یک فیلد یا متد از کلاس استفاده کرد.
- @Component: ماژول های انتخابی را فعال ساخته و برای پیاده سازی DI در پروژه استفاده می شود.

در فریم ورک Dagger اجازه ی استفاده از فیلدهایی که به صورت private تعریف شده اند برای field injection وجود ندارد چرا که Dagger 2 از generated code (و نه reflection) برای دسترسی به فیلدها استفاده می کند.

2-46-7-اعلان dependency provider ها (تعریف ارائه دهندگان آبجکت های موردنیاز و نیازمندی های کلاس)

منظور از واژه ی dependency injection context، تعداد آبجکت هایی است که قابلیت تزریق و ارسال آن ها به عنوان پارامتر به کلاس ها وجود دارد.

در فریم ورک Dagger2، کلاس هایی که با دستور @Module نشانه گذاری شده اند، وظیفه ی ارائه ی آبجکت هایی را دارند که قرار است به عنوان dependency و پارامتر در اختیار کلاس

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

درخواست کننده قرار گیرد. این کلاس های میزبان می توانند در خود توابعی داشته باشند که با دستور Proivdes@ نشانه گذاری شده و در خروجی آبجکت هایی را برمی گردانند که می توان از آن ها برای رفع نیازمندی های کلاس بهره گرفت.

متدهایی که با دستور Provides@ نشانه گذاری شده باشند، این قابلیت را دارند که کلاس های مورد نیاز یا dependency ها را از طریق پارامترهای ارسالی به متد فراهم کنند. این آبجکت های مورد نیاز یا dependency ها را با فریم ورک Dagger2 در اختیار کلاس درخواست کننده قرار می دهیم.

3-46-7-تعریف آبجکت های مورد نیاز یا dependency ها (Object consumer)

جهت اعلان یک dependency کافی است ابتدا دستور Inject@ را بالای کد مربوطه درج نمایید. پس از اینکه تابع سازنده (constructor) را با دستور مزبور نشانه گذاری کردید، Dagger2 قادر خواهد بود که با استفاده از نمونه ی این آبجکت (instance) نیازمندی ها یا به اصطلاح آبجکت های مورد نیاز کلاس را برآورده سازد. علت استفاده از روش مذکور برای پیاده سازی DI این است که از نوشتن چندین متد نشانه گذاری شده با Provides@ و طولانی سازی کد خودداری شود.

4-46-7-برقراری ارتباط بین Consumer ها (کلاس های درخواست کننده یا مصرف کننده) و provider های (ارائه دهندگان) آبجکت های مورد نیاز

دستور Component@ در سطح یک interface (الگوی پیاده سازی) مورد استفاده قرار می گیرد. فریم ورک Dagger2 با استفاده از این interface، کد لازم را تولید می نماید. ساختار نحوی برای پیاده سازی کلاس مورد نیاز (generated class) این است که Dagger به عنوان پیشوند درج شده و بلافاصله پس از آن اسم interface مربوطه قرار می گیرد. کلاسی که در نتیجه ی استفاده از الگوی فوق تولید می شود یک متد create در اختیار توسعه دهده قرار می دهد که به واسطه ی

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آن می توان آبجکت ها را بر اساس configuration ارائه شده، تنظیم کرد. متدهایی که در سطح interface اعلان شده اند می توانند به این آبجکت های تعریف شده دسترسی داشته باشند.

اینترفیس @Component ارتباط بین ارائه دهنده ی آبجکت ها و نیازمندی های کلاس (module ها) و آبجکت هایی که نیاز به dependency را اعلان می کنند، برقرار می نماید. جدول زیر توضیح مختصر و مفیدی پیرامون annotation های Dagger ارائه می نماید.

دستور/annotation	شرح کاربرد
@Module	در سطح کلاس هایی تعریف می شود که حامل متدهای نشانه گذاری شده با دستور @Provides می باشند.
@Provides	می تواند بر روی متدهای موجود در کلاس نشانه گذاری شده با دستور @Module مورد استفاده قرار گیرد. همچنین برای متدهایی کاربرد دارد که آبجکت های موردنیاز کلاس را برای پیاده سازی الگو توسعه ی DI در اختیار آن قرار می دهند.
@Singleton	با درج این دستور یک نمونه ی واحد از آبجکت ارائه شده، ساخته و به اشتراک گذاشته می شود.
@Component	در سطح یک interface مورد استفاده قرار می گیرد. این interface را فریم ورک Dagger2 برای تولید کدی که با استفاده از ماژول، نیازمندی ها و آبجکت های مورد درخواست کلاس را فراهم می کند، مورد استفاده قرار می گیرد.

Scope annotation-7-46-5

می توانید با استفاده از دستور @Singleton اعلان نمایید که از آبجکت مورد نظر یک نمونه بیشتر ایجاد نخواهد شد.

6-46-7-برخورد ویژه ی Dagger با فیلدها (field injection)

Dagger 2 خود به صورت اتوماتیک فیلدها را به داخل کلاس تزریق نمی کند. همچنین از تزریق فیلدهایی که با سطح دسترسی private تعریف شده اند، عاجز است. اگر می خواهید از روش

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

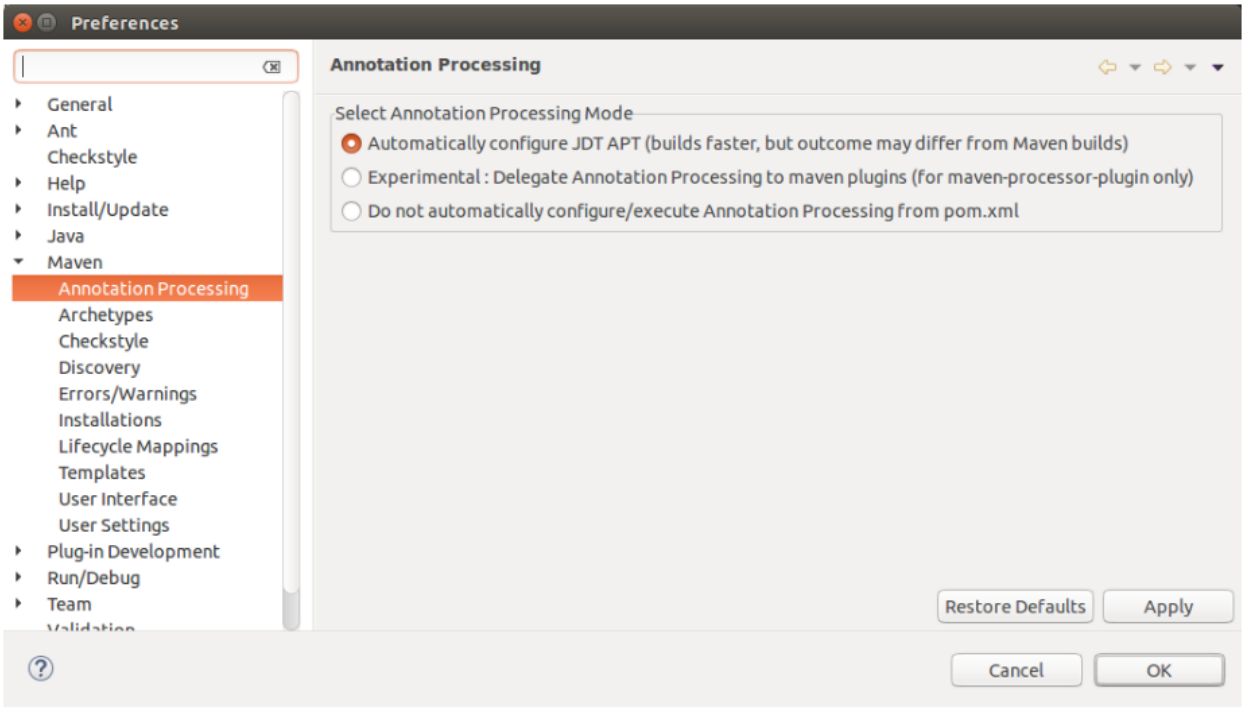
88146323 - 88446780 - 88146330

field injection برای برطرف کردن نیازمندها استفاده نمایید، در آن صورت می بایست یک متد در سطح interface@Component خود اعلان نموده که نمونه ای که قرار است فیلد را به داخل آن تزریق کنید به عنوان پارامتر بپذیرد.

7-47-7-استفاده از Dagger 2 در محیط کاری Eclipse همراه با سیستم کامپایل Maven

جهت استفاده از محیط کاری Eclipse و سیستم کامپایل Maven با فریم ورک DI به نام Dagger 2، لازم است ابزار Maven و افزونه ی apt را در محیط برنامه نویسی مزبور نصب کنید تا بدین وسیله Maven قابلیت تنظیم و آماده سازی پردازش گر annotation ها (annotation processer) را داشته باشد. برای فراهم آوردن قابلیت پشتیبانی Eclipse از سیستم Maven، بایستی به سایت آپدیت و بروز رسانی ویرایش مربوطه از Eclipse مراجعه نموده و پس از بارگیری m2e-apt از آدرس http://download.jboss.org/jbosstools/updates/m2e-extensions/m2e-apt آن را در محیط کاری مورد نظر نصب نمایید.

پس از نصب افزونه ی مربوطه، لازم است با طی کردن این مسیر داخل preferences (تنظیمات Eclipse): Window ▸ Preferences ▸ Maven ▸ Annotation Processing، پردازش مناسب (قابلیت پردازش annotation) را فعال سازی نمایید. حال گزینه ی Automatically configure JDT APT را انتخاب کنید.



تمرین کاربردی: پیاده سازی الگو توسعه ی DI از طریق فریم ورک Dagger2

در تمرین حاضر شما می بایست از فریم ورک Dagger 2 در یک برنامه ی متعارف جاوا که با سیستم کامپایل مدیریت Maven می شود، جهت پیاده سازی الگو توسعه ی DI استفاده نمایید.

این پروژه ی آزمایشی را می توانید در هر کدام از محیط های برنامه نویسی Java که مایلید پیاده سازی کنید. در تمرین جاری ما از محیط کاری Eclipse که ابزار Maven بر روی آن نصب شده، استفاده می کنیم. همچنین در صورتی که IDE نام برده را در اختیار ندارید، می توانید پروژه را از طریق یک text editor معمولی و با فراخوانی دستورات Maven در پنجره ی فرمان (command line) توسعه داده و کامپایل نمایید.

چنانچه قصد استفاده از محیط کاری Eclipse را دارید، در آن صورت لازم است ابزار لازم Maven را نیز دانلود و نصب نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

7-46-7-ساخت پروژه ی آزمایشی و استفاده از فریم ورک Dagger 2

یک پروژه ی جدید Maven با اسم پکیج com.vogella.java.dagger2 ایجاد نمایید.

7-46-8-تعریف یا تنظیم و ویرایش فایل Maven build

محتوای فایل pom.xml را به صورت زیر ویرایش نمایید.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.vogella.java.dagger2</groupId>
  <artifactId>com.vogella.java.dagger2</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>com.vogella.java.dagger2</name>
  <dependencies>
    <dependency>
      <groupId>com.google.dagger</groupId>
      <artifactId>dagger</artifactId>
      <version>2.4</version>
    </dependency>
    <dependency>
      <groupId>com.google.dagger</groupId>
      <artifactId>dagger-compiler</artifactId>
      <version>2.4</version>
      <optional>true</optional>
    </dependency>
  </dependencies>
  <build>
    <plugins>
      <plugin>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.3</version>
        <configuration>
          <source>1.8</source>
          <target>1.8</target>
        </configuration>
      </plugin>
    </plugins>
  </build>
</project>
```

9-46-7-تعریف و استفاده از کلاس هایی که به آبجکت هایی از بیرون نیاز دارند (درخواست dependency دارند)

دو کلاس زیر را تعریف کنید که برای عملکرد خود نیاز به آبجکت هایی از بیرون دارند (dependency هایی را از بیرون درخواست یا فراخوانی می کنند).

```
package com.vogella.java.dagger2;
import javax.inject.Inject;
public class User {
	private String firstName;
	private String lastName;
	public User(String firstName, String lastName) {
		this.firstName = firstName;
		this.lastName = lastName;
	}
	@Override
	public String toString() {
		return "User [firstName=" + firstName + ", lastName=" + lastName + "]";
	}
}
package com.vogella.java.dagger2;
import javax.inject.Inject;
public class BackendService {
	@Inject
	public User user;
	private String serverUrl;
	@Inject
	public BackendService(@Named("serverUrl") String serverUrl) {
		this.serverUrl = serverUrl;
	}
	public boolean callServer() {
		if (user !=null && serverUrl!=null && serverUrl.length()>0) {
			System.out.println("User: " + user + " ServerUrl: "  + serverUrl);
			return true;
		}
		return false;
	}
}
```

10-46-7-تعریف کامپوننت های مورد نیاز (دستور @Component)

این کامپوننت ها هستند که اعلان می کنند کدام module یا کامپوننت های نرم افزاری هستند که dependency ها و آبجکت های مورد نیاز را فراهم می کنند. Dagger 2 با استفاده از این interface (@Component) کد کلاس accessor را ایجاد می کند. این کلاس نیز متدهای آماده و تعریف شده در بدنه ی interface را برای پیاده سازی ارائه می دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
package com.vogella.java.dagger2.component;
import javax.inject.Singleton;
import com.vogella.java.dagger2.BackendService;
import com.vogella.java.dagger2.modules.BackEndServiceModule;
import com.vogella.java.dagger2.modules.UserModule;
import dagger.Component;
@Singleton
@Component(modules = { UserModule.class, BackEndServiceModule.class })
public interface MyComponent {
   // provide the dependency for dependent components
   // (not needed for single-component setups)
      BackendService provideBackendService();
      // allow to inject into our Main class
      // method name not important
      void inject(Main main);
}
```

متدی که امکان تزریق و مقداردهی فیلدها در BackendServer را می دهد.

11-46-7-استفاده از کد تولید شده توسط Dagger

کلاس آزمایشی زیر به صورت کاربردی از کدی که فریم ورک Dagger به صورت آماده در اختیار آن قرار می دهد استفاده می نماید.

```
package com.vogella.java.dagger2.main;
import com.vogella.java.dagger2.BackendService;
import com.vogella.java.dagger2.component.DaggerMyComponent;
import com.vogella.java.dagger2.component.MyComponent;
public class Main {
   @Inject
   BackendService backendService; //
   private MyComponent component;
   private Main() {
      component = DaggerMyComponent.builder().build();
      component.inject(this);
   }
   private void callServer() {
      boolean callServer = backendService.callServer();
      if (callServer) {
         System.out.println("Server call was successful. ");
      } else {
         System.out.println("Server call failed. ");
      }
   }
      public static void main(String[] args) {
      Main main = new Main();
      main.callServer();
```

```
    }
}
```

از آنجایی که فیلدها به صورت اتوماتیک تزریق نمی شوند، متد مربوطه فراخونی شده و فیلدهای مورد نظر را مقداردهی می نماید.

47-7 Dagger 2 و اندروید

47-7-1-پیاده سازی الگو توسعه ی DI در پروژه های اندرویدی

بسیاری از کامپوننت های نرم افزاری اندروید همچون activity ها را خود فریم ورک اندروید نمونه سازی می کند. به عبارتی دیگر کار توسط توسعه دهنده به صورت دستی انجام نمی گیرد. این امر فراهم کردن آبجکت های مورد نیاز کلاس (dependency) و کامپوننت های نرم افزاری اندروید از بیرون به وسیله ی توابع سازنده (constrcutor) را پیچیده و دشوار می کند.

47-7-2-استفاده ی کاربردی از Dagger 2 در اندروید

جهت فعال سازی Dagger 2 و امکان استفاده از آن در پروژه، لازم است فایل build.gradle را ویرایش نمایید.

```
buildscript {
   repositories {
      jcenter()
   }
   dependencies {
      classpath 'com.android.tools.build:gradle:2.2.0-alpha3'
      classpath 'com.neenbedankt.gradle.plugins:android-apt:1.8'
   }
}
allprojects {
   repositories {
      jcenter()
```

```
    }
}
task clean(type: Delete) {
    delete rootProject.buildDir
}
```

المان های زیر را به فایل app/build.gradle اضافه نمایید.

```
apply plugin: 'com.android.application'
apply plugin: 'com.neenbedankt.android-apt'
android {
    compileSdkVersion 22
    buildToolsVersion "23.0.0"
    defaultConfig {
        applicationId "com.vogella.android.dagger2simple"
        minSdkVersion 22
        targetSdkVersion 22
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}
ext {
    JUNIT_VERSION = '4.12'
    DAGGER_VERSION ='2.4'
}
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.google.dagger:dagger:2.4'
    apt "com.google.dagger:dagger-compiler:DAGGER_VERSION"
    provided 'javax.annotation:jsr250-api:1.0'
    compile 'javax.inject:javax.inject:1'
    testCompile "junit:junit:$JUNIT_VERSION"
    testApt "com.google.dagger:dagger-compiler:DAGGER_VERSION"
}
```

نکته: لازم به ذکر است که بایستی apt را برای هر scope ای که Dagger قرار است در آن مورد استفاده قرار بگیرد، دقیق تنظیم و مشخص نمایید. به طور مثال، برای تست اپلیکیشن های معمولی apt، برای تست هایی که بر روی دستگاه مجازی جاوا/JVM اجرا می شود testApt و در نهایت برای تست های اندرویدی نیز از testAndroidApt استفاده نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

تمرین کاربردی: پیاده سازی الگو توسعه ی DI با استفاده از فریم ورک Dagger 2 در پروژه های اندرویدی

هدف از تمرین

در این تمرین، Dagger 2 به صورت عملی در یک پروژه ی اندرویدی جهت پیاده سازی DI مورد استفاده قرار می گیرد. در برنامه ی ساده ی بخش حاضر، یک کلاس activity تعریف می شود که قابلیت احراز هویت (authenticate) و بررسی صحت گذرواژه/نام کاربری (credentials) را به کاربر می دهد. مقدار خروجی و نتیجه ی پیاده سازی در کادر متن (text field) به نمایش گذاشته می شود.

ساخت پروژه

یک پروژه ی جدید با اسم پکیج (top level package) com.vogella.android.dagger2simple ایجاد نمایید.

محتوای فایل layout ای که ظاهر کامپوننت activity و صفحه ی قابل مشاهده برای کاربر را مشخص می کند، به صورت زیر ویرایش نمایید.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   tools:context=".MainActivity">
   <TextView
      android:id="@+id/target"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/hello_world" />
</RelativeLayout>
```

3-47-7-افزودن dependency ها و کتابخانه های مورد نیاز سیستم کامپایل Gradle

آیتم های زیر را به فایل build.gradle پروژه اضافه نمایید.

```
buildscript {
    repositories {
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:2.2.0-alpha3'
        classpath 'com.neenbedankt.gradle.plugins:android-apt:1.8'
    }
}
allprojects {
    repositories {
        jcenter()
    }
}
task clean(type: Delete) {
    delete rootProject.buildDir
}
```

حال المان های زیر را به فایل app/build.gradle اضافه نمایید.

```
apply plugin: 'com.android.application'
apply plugin: 'com.neenbedankt.android-apt'
android {
    compileSdkVersion 22
    buildToolsVersion "23.0.0"
    defaultConfig {
        applicationId "com.vogella.android.dagger2simple"
        minSdkVersion 22
        targetSdkVersion 22
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}
ext {
    JUNIT_VERSION = '4.12'
    DAGGER_VERSION ='2.4'
}
dependencies {
```

```
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.google.dagger:dagger:2.4'
    apt "com.google.dagger:dagger-compiler:DAGGER_VERSION"
    provided 'javax.annotation:jsr250-api:1.0'
    compile 'javax.inject:javax.inject:1'
    testCompile "junit:junit:$JUNIT_VERSION"
    testApt "com.google.dagger:dagger-compiler:DAGGER_VERSION"
}
```

4-47-7-ترسیم یا اعلان نمودار dependency ها و نمایش وابستگی آبجکت های مورد نیاز به یکدیگر

کلاس های زیر را جهت ارائه کردن dependency ها و متصل کردن آن ها به یکدیگر (آبجکت های مورد نیاز جهت رفع نیازمندی ها) در پروژه ایجاد نمایید.

```
package com.vogella.android.dagger2simple;
public class NetworkApi {
    public boolean validateUser(String username, String password) {
        // imagine an actual network call here
        // for demo purpose return false in "real" life
        if (username == null || username.length() == 0) {
            return false;
        } else {
            return true;
        }
    }
}
```

```
package com.vogella.android.dagger2simple.modules;
import com.vogella.android.dagger2simple.NetworkApi;
import javax.inject.Singleton;
import dagger.Module;
import dagger.Provides;
@Module
public class NetworkApiModule {
    @Provides
    @Singleton
    public NetworkApi getNetwork(){
        return new NetworkApi();
    }
}
```

```
package com.vogella.android.dagger2simple.components;
import android.app.Activity;
import com.vogella.android.dagger2simple.MainActivity;
```

```
import com.vogella.android.dagger2simple.NetworkApi;
import com.vogella.android.dagger2simple.modules.NetworkApiModule;
import javax.inject.Singleton;
import dagger.Component;
@Singleton
@Component(modules = {NetworkApiModule.class})
public interface DiComponent {
   // to update the fields in your activities
   void inject(MainActivity activity);
}
```

5-47-7-ساخت آبجکت Application پروژه (Wiring up the application)

در بدنه ی کلاس Application پروژه ی خود، کد زیر را پیاده سازی کنید. این کلاس بستر (context) پیاده سازی DI را فراهم کرده و سپس با فراخوانی تابع getComponent دسترسی به آن را مهیا می کند.

```
package com.vogella.android.dagger2simple;
import android.app.Application;
import com.vogella.android.dagger2simple.components.DaggerDiComponent;
import com.vogella.android.dagger2simple.components.DiComponent;
public class MyApplication extends Application {
   DiComponent component;
   @Override
   public void onCreate() {
      super.onCreate();
      component = DaggerDiComponent.builder().build();
   }
   public DiComponent getComponent() {
      return component;
   }
}
```

البته لازم است این کلاس را در لایه ی XML (داخل فایل تنظیمات اپلیکیشن Manifest) نیز ثبت و معرفی نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.vogella.android.dagger2simple" >
   <application
      android:name="MyApplication"
      android:allowBackup="true"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >
```

```
      <activity
        android:name=".MainActivity"
        android:label="@string/app_name" >
        <intent-filter>
          <action android:name="android.intent.action.MAIN" />
          <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
      </activity>
    </application>
</manifest>
```

activity فعلی مقادیر تولید شده را از طریق متدی که در سطح کلاس @Component تعریف شده، تزریق می کند.

```
package com.vogella.android.dagger2simple;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TextView;
import android.widget.Toast;
import javax.inject.Inject;
public class MainActivity extends Activity {
  @Inject NetworkApi networkApi;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    ((MyApplication) getApplication()).getComponent().inject(this);
    boolean injected =  networkApi == null ? false : true;
    TextView userAvailable = (TextView) findViewById(R.id.target);
    userAvailable.setText("Dependency injection worked: " + String.valueOf(injected));
  }
}
```

6-47-7-تست اپلیکیشن و کسب اطمینان از عملکرد صحیح اپلیکیشن

اپلیکیشن را اجرا نمایید. UI اپلیکیشن می بایست صریحا اعلان کند که dependency injection با موفقیت انجام شد.در غیر این صورت، نگاهی به کلاس های تولید شده در پوشه ی app/build/generated/apt بیاندازید. این کلاس ها بایستی به طور خوانا سازماندهی شده و دقیقا مشخص کنند که چه خطایی رخ داده است.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

7-48-جایگزین کردن @Module با کلاس های اختصاصی خود در تست

می توانید در تست های نرم افزاری بجای کلاس های Dagger کلاس های اختصاصی خود را قرار دهید و این فریم ورک را طوری تنظیم کنید که از این کلاس ها استفاده کند. builder فریم ورک Dagger توابعی دارد که ماژول مورد نظر را ساخته و مقدار دهی می کند. این امکان برای شما وجود دارد که با استفاده از builder فریم ورک مزبور کلاس های Dagger را با کلاس دلخواه خود (کلاسی که باید در تست مورد استفاده قرار گیرد) جایگزین نمایید.

این امکان در Mockito نیز قابل استفاده می باشد. فرض کنید می خواهید کلاس زیر را تست نمایید.

```
public class ImportantClass {
   private MyRestService restService;
   private MyLogger logger;
   @Inject public MainService(MyRestService restService, MyLogger logger) {
      this.restService = restService;
      this.logger = logger;
   }
   public void perform() {
      String s = restService.getData();
      logger.write(s.toUpperCase());
   }
}
```

می توانید این کلاس را با Mockito تست نمایید.

```
public class ImportantClassTest {
public class MainServiceTest {
   @Rule public MockitoRule mockitoRule = MockitoJUnit.rule();
   @Mock MyRestService restService;
   @Mock MyLogger logger;
   @InjectMocks ImportantClass importClassToBeTested;
   @Test
   public void performShouldWriteOutput() {
      when(restService.getData()).thenReturn("abc");
      importClassToBeTested.perform();
      verify(logger).print("ABC");
   }
}
}
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش دهم :

ابزار تحلیل و سنجش کارایی اپلیکیشن های اندرویدی

آموزش حاضر به شرح ابزار موجود اندروید برای بررسی و سنجش کارایی اپلیکیشن های اندرویدی می پردازد.

7-49-مرور کلی

کارایی اپلیکیشن های اندرویدی در اجرای تمامی عملیات با سرعت هر چه تمام تر از اهمیت ویژه ای برخوردار است. مبحث جاری کلیه ی ابزار موجود اندروید در ره گیری و بهینه سازی اپلیکیشن را عنوان می کند.

7-50-StrictMode

توسعه دهنده می بایست تا حد امکان از اجرای عملیات طولانی، سنگین و هزینه بر در thread اصلی (UI thread) خودداری کند. از جمله ی این عملیات می توان به دسترسی به فایل و اینترنت اشاره کرد.

برای دستیابی به این هدف می توان از StrictMode بهره گرفت. این حالت عملیاتی (StrictMode) از ورژن 9 توابع کتابخانه ای اندروید (API ورژن 9) برای اولین بار در چارچوب نرم افزاری اندروید معرفی شده و به توسعه دهنده این امکان را می دهد تا برای اپلیکیشن سیاست ها و قوانین خاصی حاکم بر مدیریت thread ها به نام thread policy تعریف کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

به واسطه ی StrictMode توسعه دهده می تواند به سیستم اندروید دستور بدهد تا چنانچه اپلیکیشن عملیات سنگین و طولانی همچون دریافت ورودی/خروجی (I/O) را در thread اصلی به اجرا گذاشت، بلافاصله اپلیکیشن را متوقف سازد.

کد زیر نحوه ی استفاده از StrictMode را به نمایش می گذارد. به مجرد اینکه کامپوننت activity این تنظیمات و قوانین تنظیم شده را نقض کرد، سیستم اندروید آن را به طور ناگهانی متوقف می سازد.

```
package de.vogella.android.strictmode;
import java.io.BufferedWriter;
import java.io.OutputStreamWriter;
import android.app.Activity;
import android.os.Bundle;
import android.os.StrictMode;
public class TestStrictMode extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // Activate StrictMode
        StrictMode.setThreadPolicy(new StrictMode.ThreadPolicy.Builder()
                .detectAll()
                .detectDiskReads()
                .detectDiskWrites()
                .detectNetwork()
                 // alternatively .detectAll() for all detectable problems
                .penaltyLog()
                .penaltyDeath()
                .build());
        StrictMode.setVmPolicy(new StrictMode.VmPolicy.Builder()
                 .detectLeakedSqlLiteObjects()
                 .detectLeakedClosableObjects()
                // alternatively .detectAll() for all detectable problems
                .penaltyLog()
                .penaltyDeath()
                .build());
        // Test code
        setContentView(R.layout.main);
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        String eol = System.getProperty("line.separator");
        try {
            BufferedWriter writer =
                    new BufferedWriter(
                            new OutputStreamWriter(
                            openFileOutput("myfile",
                            MODE_WORLD_WRITEABLE)));
```

```
                writer.write("This is a test1." + eol);
                writer.write("This is a test2." + eol);
                writer.write("This is a test3." + eol);
                writer.write("This is a test4." + eol);
                writer.write("This is a test5." + eol);
                writer.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
}
```

StrictMode تنها در مرحله ی توسعه ی نرم افزار قابل استفاده می باشد. توصیه می شود از بکار بردن آن در یک اپلیکیشن توزیع و نصب شده که به صورت نهایی اجرا می شود، خودداری گردد.

بخش Developer Settings در Setting اپلیکیشن گوشی اندروید به شما این امکان را می دهد تا تنظیماتی را تعیین کنید که تحلیل و سنجش کارایی اپلیکیشن را آسان می سازد. به عنوان مثال شما می توانید اعلان کنید که محل برخورد انگشت با نمایشگر هایلایت شود.

برای مثال چنانچه ویرایش 4.2 اندروید بر روی دستگاه شما نصب است، برای دسترسی به بخش تنظیمات مربوط به توسعه دهنده، می بایست به بخش About در Settings مراجعه نموده و بر روی المان Build number هفت بار کلیک نمایید.

در صورتی که گوشی شما این گزینه ی تنظیم را نداشت، می توانید از برنامه ی شبیه ساز (emulator) استفاده نمایید.

در برخی موارد لازم است اپلیکیشن را مجددا راه اندازی کرده تا تنظیمات مورد نظر فعال شده و به درستی عمل کند.

1-50-7-مقدمه

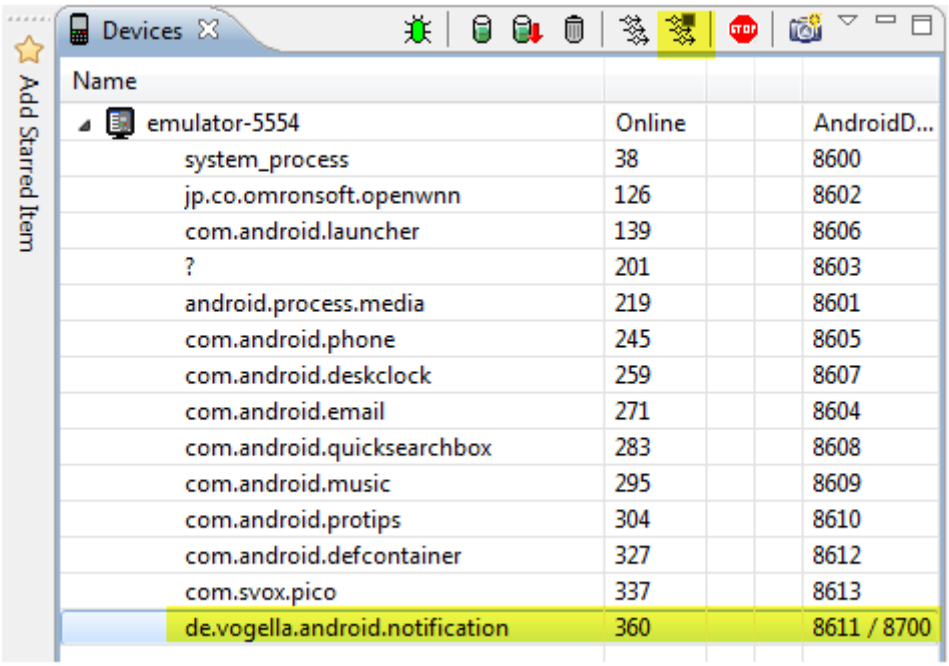
Traceview عبارت است از یک graphical viewer که به شما امکان می دهد تا گزارشاتی (log ها) که اپلیکیشن اندروید ضبط می کند را مشاهده نمایید. به واسطه ی TraceView شما می توانید کارایی اپلیکیشن را سنجیده و مشکلاتی که کارایی اپلیکیشن را پایین آورده و کیفیت تجربه ی کاربری را پایین می آورد، شناسایی و برطرف نمایید.

TraceView به عنوان یک ابزار مستقل در فولدر tools از پوشه ی نصب ابزار توسعه و ساخت اپلیکیشن های اندرویدی / Android SDK installation جایگذاری شده و قابل دسترسی می باشد. توسعه دهنده می تواند با نصب ADT (مجموعه ابزار توسعه ی نرم افزار اندرویدی) آن را در محیط برنامه نویسی Eclipse جاسازی کند.

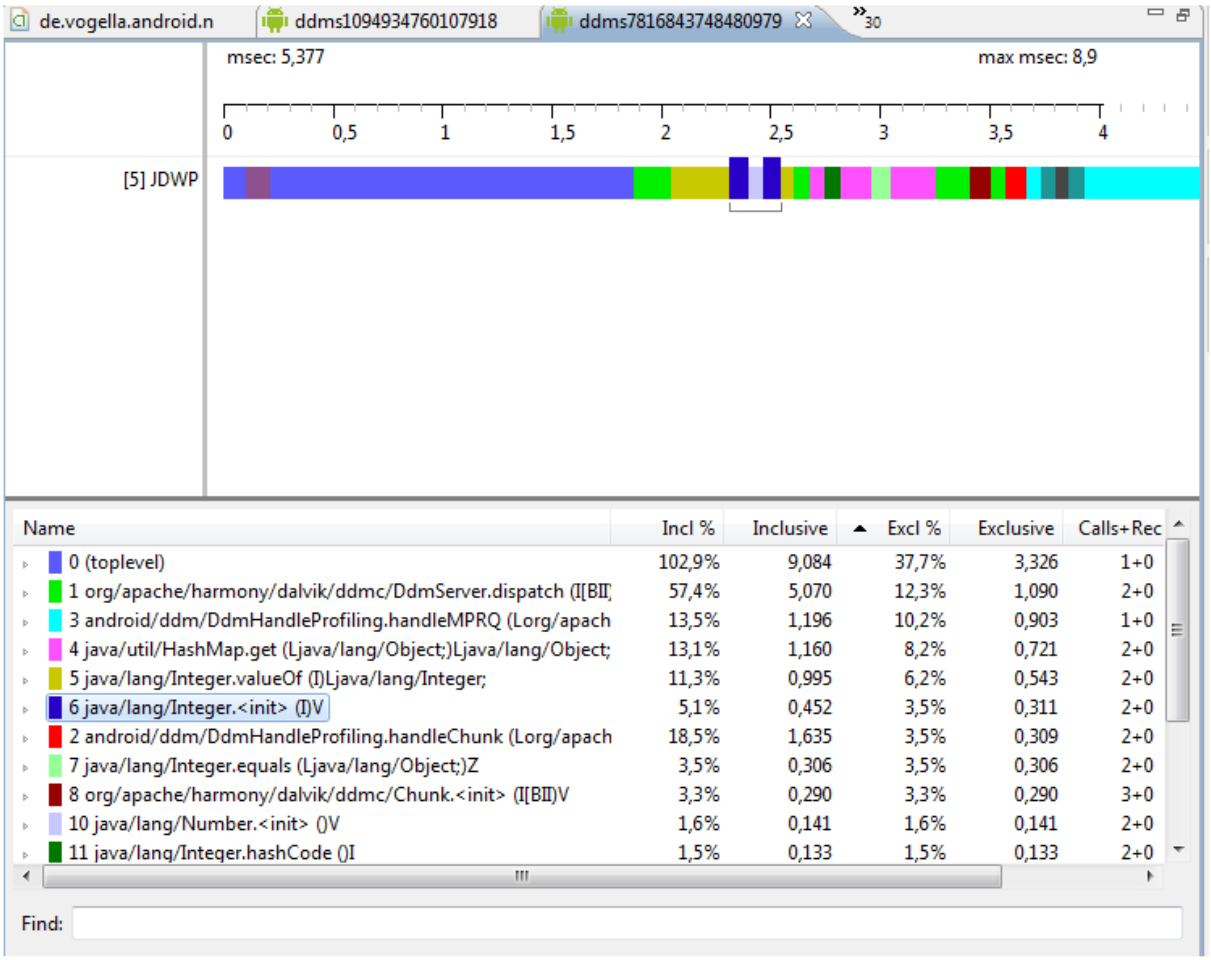
7-50-2-استفاده از ابزار TraceView در محیط کاری Android Studio

محیط برنامه نویسی به شما امکان می دهد تا از طریق ابزار Android Device Monitor روند اجرای اپلیکیشن را رصد و مشاهده (trace) نمایید. برای این منظور کافی است مسیر رو به روی را طی نمایید: Tools ▸ Android ▸ Android Device Monitor.

جهت فعال سازی قابلیت رد گیری روند اجرای اپلیکیشن، ابتدا فرایند اپلیکیشن خود را در کادر Devices انتخاب نموده و سپس دکمه ی Start Method Profiling را مطابق تصویر زیر کلیک کنید.

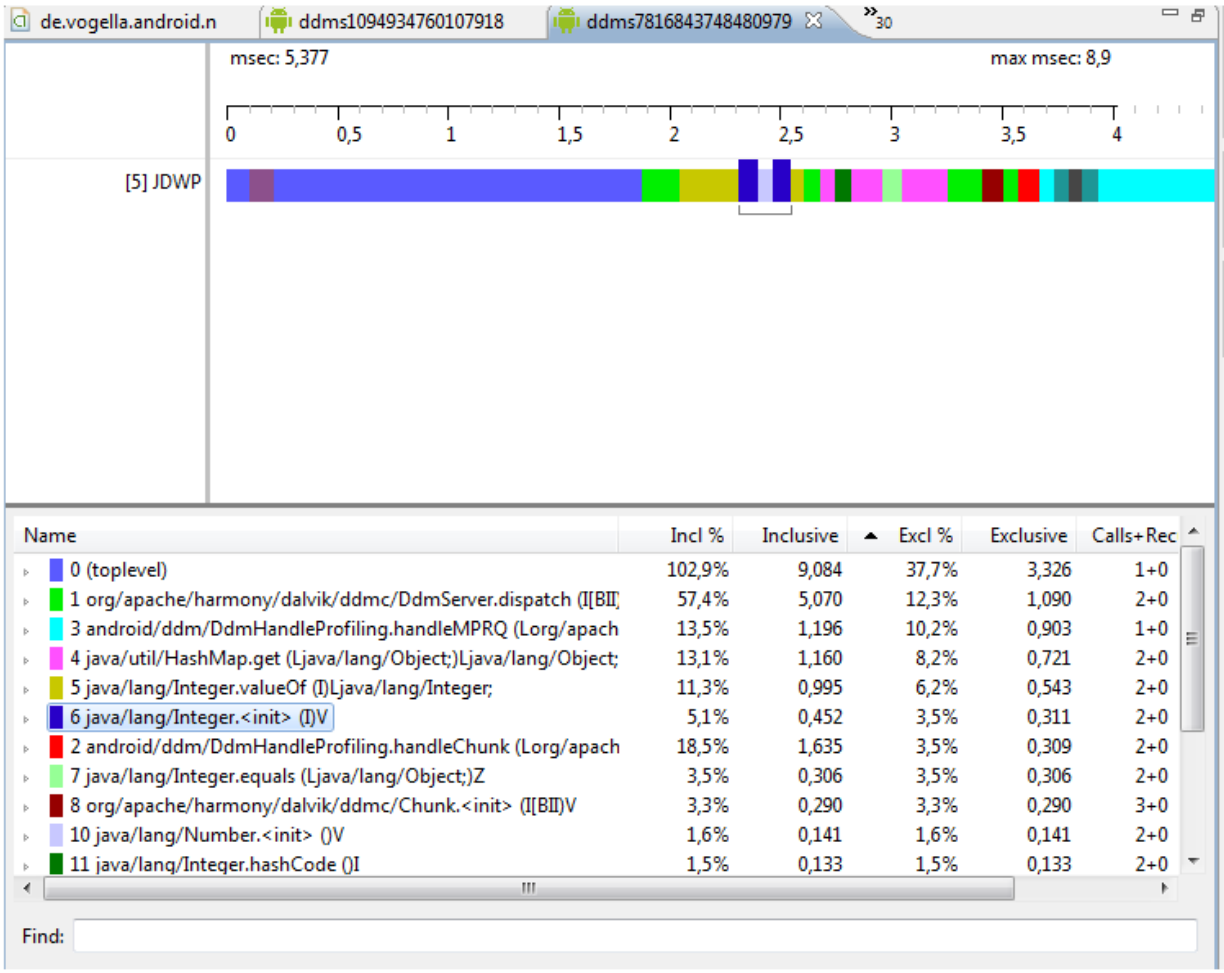


اپلیکیشن را اجرا نموده و مجددا دکمه ای که جهت فعال سازی ردگیری جریان اجرای اپلیکیشن (method profiling) کلیک کردید، کلیک نمایید. در پی این اقدام، یک ویرایشگر جدید باز شده، که نتایج ردگیری و ضبط شده از روند اجرای اپلیکیشن را به نمایش می گذارد.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



می توانید تصویر را بزرگنمایی کرده و جزئیات را دقیق تر مشاهده نمایید. برای کوچک نمایی، کافی است بر روی خط نمایشگر زمان دوبار کلیک کنید.

3-50-7-راه اندازی و فراخوانی ابزار TraceView از طریق خط دستور (command line)

برای اینکه پروسه ی رصد اجرای کد برنامه (trace) آغاز شود، کافی است تکه کد زیر را بین دو دستور زیر درج نمایید.

```
android.os.Debug.startMethodTracing("yourstring");
// ... your code is here
android.os.Debug.stopMethodTracing();
```

پارامتر "yourstring" به سیستم اندروید اعلان می کند که بایستی داده ها را تحت آدرس "/sdcard/yourstring.trace" ذخیره نماید. جهت جایگذاری داده ها در حافظه ی خارجی (sdcard)، اپلیکیشن شما می بایست مجوز WRITE_EXTERNAL_STORAGE را در لایه ی XML تعریف کرده باشد. پس از اجرای اپلیکیشن، می توانید نتایج را با اجرای دستور زیر در پنجره ی فرمان adb (command line tool)، در مسیر دلخواه کپی کنید.

```
adb pull /sdcard/yourstring.trace
traceview yourstring
```

با فراخوانی این دستور در پنجره ی فرمان adb، ابزار Traceview فعال شده و به شما اجازه می دهد تا داده های مربوط به کارایی اپلیکیشن خود را به صورت دیداری (visual) و قالب اطلاعات قابل مشاهده در نمایشگر بررسی و دنبال نمایید. در کادر DDMS یک دکمه ی trace و ردگیری جریان اپلیکیشن وجود دارد. با استفاده از این دکمه، جریان اجرای اپلیکیشن در حین اجرا و پیشرفت، رصد شده و لزومی به تنظیم مجوز خاصی در لایه ی XML نیست.

تمرین: استفاده ی کاربردی از Traceview

استفاده از ابزار Traceview در پروژه

یک اپلیکیشن اندرویدی جدید با نام پکیج (top level package) com.vogella.android.traceview ایجاد نمایید.

کلید زیر را به فایل values/strings.xml اضافه نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Traceview Example</string>
    <string name="action_settings">Settings</string>
    <string name="hello_world">Hello world!</string>
    <string name="number_template"><b>Random number: %1$s</b></string>
</resources>
```

یک فایل layout با نام rowlayout.xml و محتوای زیر ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="?android:attr/listPreferredItemHeight"
   android:orientation="horizontal" >
   <LinearLayout
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:orientation="vertical" >
      <LinearLayout
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:orientation="vertical" >
         <TextView
            android:id="@+id/textView1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Entry"
            android:textAppearance="?android:attr/textAppearanceListItem" />
         <TextView
            android:id="@+id/textView2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Stub"
            android:textAppearance="?android:attr/textAppearanceListItemSmall" />
      </LinearLayout>
   </LinearLayout>
   <ImageView
      android:id="@+id/imageView1"
      android:layout_width="10dp"
      android:layout_height="?android:attr/listPreferredItemHeight"
      android:background="@android:color/black" />
   <TextView
      android:id="@+id/textView3"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginTop="20dp"
      android:text="Medium Text"
      android:textAppearance="?android:attr/textAppearanceMedium" />
</LinearLayout>
```

یک کلاس adapter برای آبجکت ListView اپلیکیشن خود به صورت زیر پیاده سازی نمایید.

```
package com.vogella.android.traceview;
import java.util.Collections;
import java.util.List;
import android.content.Context;
import android.content.res.Resources;
import android.os.Debug;
import android.text.Html;
import android.view.LayoutInflater;
```

```
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;
public class MyArrayAdapter extends ArrayAdapter<String> {
	private List<String> values;
	private Context context;
	public MyArrayAdapter(Context context, List<String> values) {
		super(context, R.layout.rowlayout);
		this.context = context;
		this.values = values;
	}
	@Override
	public View getView(int position, View convertView, ViewGroup parent) {
		Debug.startMethodTracing("getViewOfTrace");
		// Ensure sorted values
		Collections.sort(values);
		LayoutInflater inflater = (LayoutInflater) context
				.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
		View view = inflater.inflate(R.layout.rowlayout, parent, false);
		Resources res = context.getResources();
		String text = String.format(res.getString(R.string.number_template),
				values.get(position));
		CharSequence styledText = Html.fromHtml(text);
		TextView textView = (TextView) view.findViewById(R.id.textView3);
		textView.setText(styledText);
		Debug.stopMethodTracing();
		return view;
	}
}
```

حال یک آبجکت ListView در کلاس activity (جهت ساخت یک لیست ساده) خود که 1000 رشته با مقدار تصادفی را در صفحه به نمایش می گذارد، پیاده سازی کنید.

```
package com.vogella.android.traceview;
import java.math.BigInteger;
import java.security.SecureRandom;
import java.util.ArrayList;
import java.util.List;
import android.app.ListActivity;
import android.os.Bundle;
public class MainActivity extends ListActivity {
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		List<String> list = createValues();
		MyArrayAdapter adapter = new MyArrayAdapter(this, list);
		setListAdapter(adapter);
	}
```

```
    private static List<String> createValues() {
        SecureRandom random = new SecureRandom();
        List<String> list = new ArrayList<String>();
        for (int i = 0; i < 1000; i++) {
            String string = new BigInteger(130, random).toString(32);
            list.add(string);
        }
        return list;
    }
}
```

مجوز لازم برای ذخیره و درج اطلاعات در حافظه ی خارجی دستگاه خود را در لایه ی XML اپلیکیشن اعلان نمایید.

4-50-7-رصد و سنجش کارایی اپلیکیشن (Trace)

اپلیکیشن خود را اجرا کنید . سپس از طریق adb به دستگاه مربوطه وصل شده و اطلاعات مربوط به رصد کارایی اپلیکیشن را در حافظه جایگذاری نمایید. حال خروجی مربوط به کارایی اپلیکیشن را تجزیه و تحلیل کنید.

5-50-7-برطرف کردن کاستی های مربوط به کارایی اپلیکیشن

اکنون زمان آن فرا رسیده تا کارایی اپلیکیشن را بر اساس اطلاعات و خروجی Traceview افزایش داده و کاستی های آن را در جهت بهبود تجربه ی کاربری برطرف نمایید.

- مقدار attribute ای که ظاهر متن را در متد getView() تعیین می کند (مقدار متن را بر روی bold یا برجسته تنظیم می کند)، با دستور android:textStyle="bold متناظر در لایه ی XML، داخل فایل LAYOUT جایگزین نمایید تا بدین وسیله از فراخوانی متد Html.fromHtml() جلوگیری شود.
- مرتب سازی را به مکان دیگری انتقال دهید.
- چنانچه مقدار convertView برابر null نبود، از آن بار دیگر در کلاس Adapter استفاده نمایید.

- با پیاده سازی الگو توسعه ی HolderPattern در آبجکت ListView، از فراخوانده شدن متد findViewById() جلوگیری نمایید.

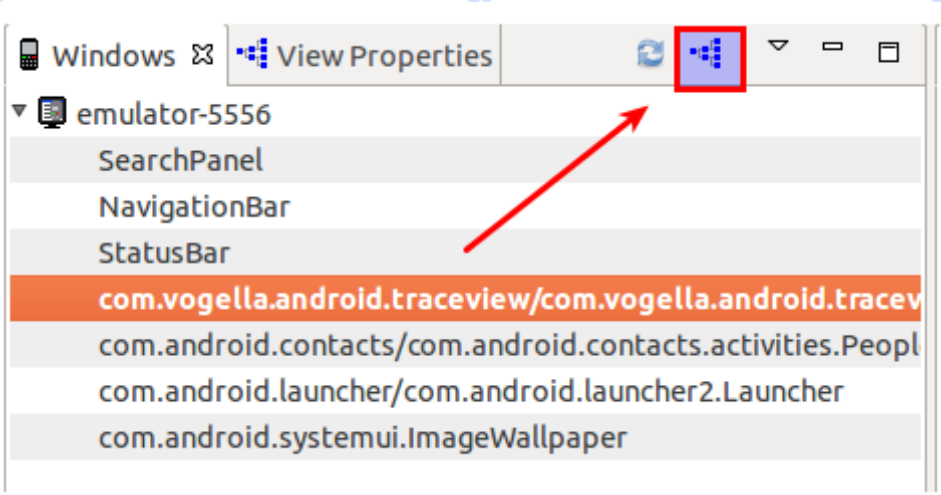
توجه: layout به طور بهینه طراحی نشده است. مثال فوق را در تمرین HierarchyView مجددا بکار خواهیم برد.

7-51-Hierarchy Viewer (ابزار نمایش درختی view ها)

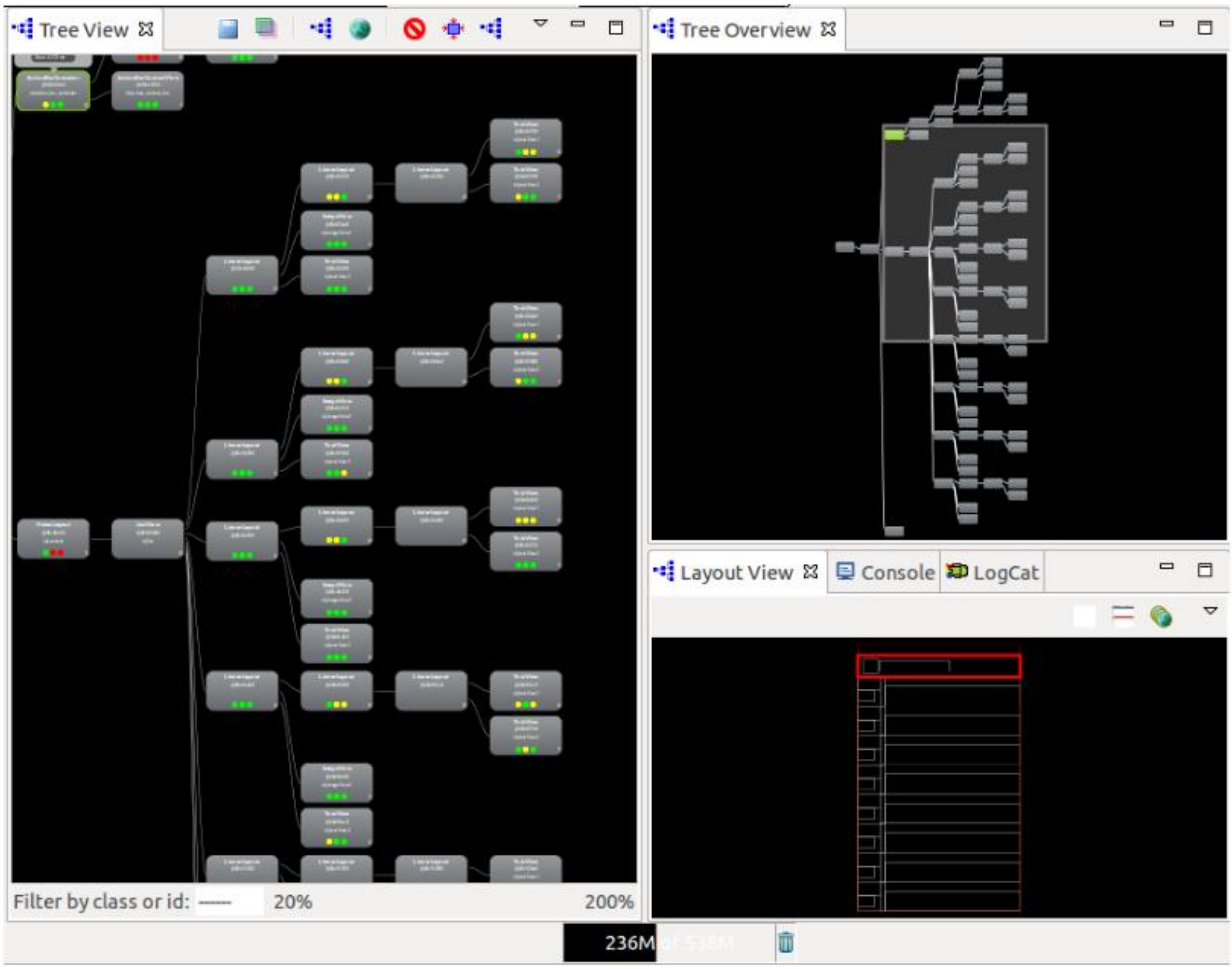
Hierarchy View به شما این امکان را می دهد تا نمودار درختی و سلسله مراتبی View اپلیکیشن اندروید خود را به صورت دیداری در محیط کاری اندروید ترسیم کرده و سپس لایه های غیرضروری و اضافی را در نمودار درختی مزبور شناسایی نمایید.

برای دسترسی به Hierarchy View، کافی است مسیر رو به رو را طی نمایید: Window ▸ Open Perspective ▸ Other... ▸ Hierarchy View.

در view یا کادر Widows می توانید فرایندی که می خواهید view hierarchy برای آن مورد تحلیل قرار گیرد را انتخاب نمایید.



layout ای که در حال حاضر فعال می باشد، مورد تحلیل قرار گرفته و به نمایش گذاشته می شود.



کادری که Tree View نام دارد، در هر یک از view ها سه دایره ی کوچک رنگی به نمایش می گذارد. اولین دایره، مدت زمانی که برای محاسبه ی اندازه ی view لازم است را بیان می کند. دومین دایره بیانگر زمان لازم برای ایجاد layout و طرح کلی می باشد و سومین دایره مدت زمان لازم برای ترسیم view را نشان می دهد. کارهای هزینه بر و طولانی با رنگ زرد یا قرمز برای برنامه نویس نمایش داده می شود.

تمرین: استفاده ی کاربردی از ابزار Hierarchy Viewer

در تمرین جاری، همچنان از مثال قبلی که در تمرین Traceview بکار برده شد، استفاده خواهیم کرد. Hierarchy View را باز کرده و لایه ی view را دقیق مورد بررسی قرار دهید.

گرچه هیچ یک از لایه های layout، با توجه به دایره ی رنگی داخل view ها، از نظر منابع مورد استفاده و زمان اجرا هزینه بر و طولانی نیستند، با این وجود لایه ها و view های غیرضروری در آن به چشم می خورند. حال لایه های غیرضروری را حذف نمایید.

7-52-بهینه سازی Layout

اکنون زمان آن فرا رسیده که فایل layout تنظیم کننده ظاهر و طرح کلی اپلیکیشن خود را بهینه نمایید. تصویر زیر با مثال نمایش می دهد چگونه با استفاده از Framelayout و دو المان رابط کاربری ImageView و TextView نتیجه ی ارائه شده را تولید نمایید. همان طور که در تصویر حاضر مشاهده می کنید، کامپوننت رابط کاربری TextView در layout جایگذاری شده و محتوای متنی آن به دو روش متفاوت سبک یا style دهی شده اند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

یک پروژه ی جدید به نام com.vogella.android.textview.spannablestring ایجاد نمایید.

حال دو attribute سبک دهی (styles) جدید به محتوای فایل styles.xml اضافه نمایید.

```
<resources xmlns:android="http://schemas.android.com/apk/res/android">
    <!--
        Base application theme, dependent on API level. This theme is replaced
        by AppBaseTheme from res/values-vXX/styles.xml on newer devices.
    -->
    <style name="AppBaseTheme" parent="android:Theme.Light">
        <!--
            Theme customizations available in newer API levels can go in
            res/values-vXX/styles.xml, while customizations related to
            backward-compatibility can go here.
        -->
    </style>
    <!-- Application theme. -->
    <style name="AppTheme" parent="AppBaseTheme">
        <!-- All customizations that are NOT specific to a API-level are here. -->
    </style>
    <style name="textHeader">
        <item name="android:padding">4dip</item>
        <item name="android:textAppearance">?android:attr/textAppearanceLarge</item>
```

```
    <item name="android:textColor">#000000</item>
    <item name="android:fontFamily">sans-serif-condensed</item>
  </style>
  <style name="textbody">
    <item name="android:padding">4dip</item>
    <item name="android:textAppearance">?android:attr/textAppearanceSmall</item>
    <item name="android:textSize">16sp</item>
    <item name="android:textColor">#c0c0c0</item>
  </style>
</resources>
```

یک فایل layout با محتوای زیر ایجاد نمایید.

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:id="@+id/FrameLayout1"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingBottom="@dimen/activity_vertical_margin"
  android:paddingLeft="@dimen/activity_horizontal_margin"
  android:paddingRight="@dimen/activity_horizontal_margin"
  android:paddingTop="@dimen/activity_vertical_margin"
  tools:context=".MainActivity" >
  <TextView
    android:id="@+id/input"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom"
    android:layout_margin="16dp"
    android:text="@string/hello_world"
    android:textSize="32sp" />
  <ImageView
    android:id="@+id/imageView1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:contentDescription="image"
    android:src="@drawable/vogella" />
</FrameLayout>
```

متن موجود در المان TextView خود را، همان طور که در کد زیر نمایش داده شده، با دو نمونه ی مجزا از TextAppearanceSpan سبک دهی و تنظیم (style) نمایید.

```
package com.vogella.android.textview.spannablestring;
import android.app.Activity;
import android.os.Bundle;
import android.text.Spannable;
import android.text.SpannableString;
import android.text.style.TextAppearanceSpan;
import android.view.Menu;
import android.widget.TextView;
public class MainActivity extends Activity {
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        TextView textView = (TextView) findViewById(R.id.input);
        String header = "This is the header";
        String description = "This is the description";
        Spannable styledText = new SpannableString(header + "\n" + description);
        TextAppearanceSpan span1 = new TextAppearanceSpan(this,
                R.style.textHeader);
        TextAppearanceSpan span2 = new TextAppearanceSpan(this,
                R.style.textbody);
        styledText.setSpan(span1, 0, header.length(),
                Spannable.SPAN_EXCLUSIVE_EXCLUSIVE);
        styledText.setSpan(span2, header.length() + 1, header.length() + 1
                + description.length(), Spannable.SPAN_EXCLUSIVE_EXCLUSIVE);
        textView.setText(styledText);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

در نهایت مشاهده خواهید کرد که layout جدید بسیار سریع تر از layout قبلی که مبتنی بر RelativeLayout می باشد، عمل می کند. لازم به توضیح است که در layout جاری از styling HTML خودداری شده چرا که استفاده از مفسر یا تبدلیگر HTML parser بسیار هزینه بر و سنگین می باشد.

7-53-تهیه ی روگرفت از حافظه (Memory Dumps)

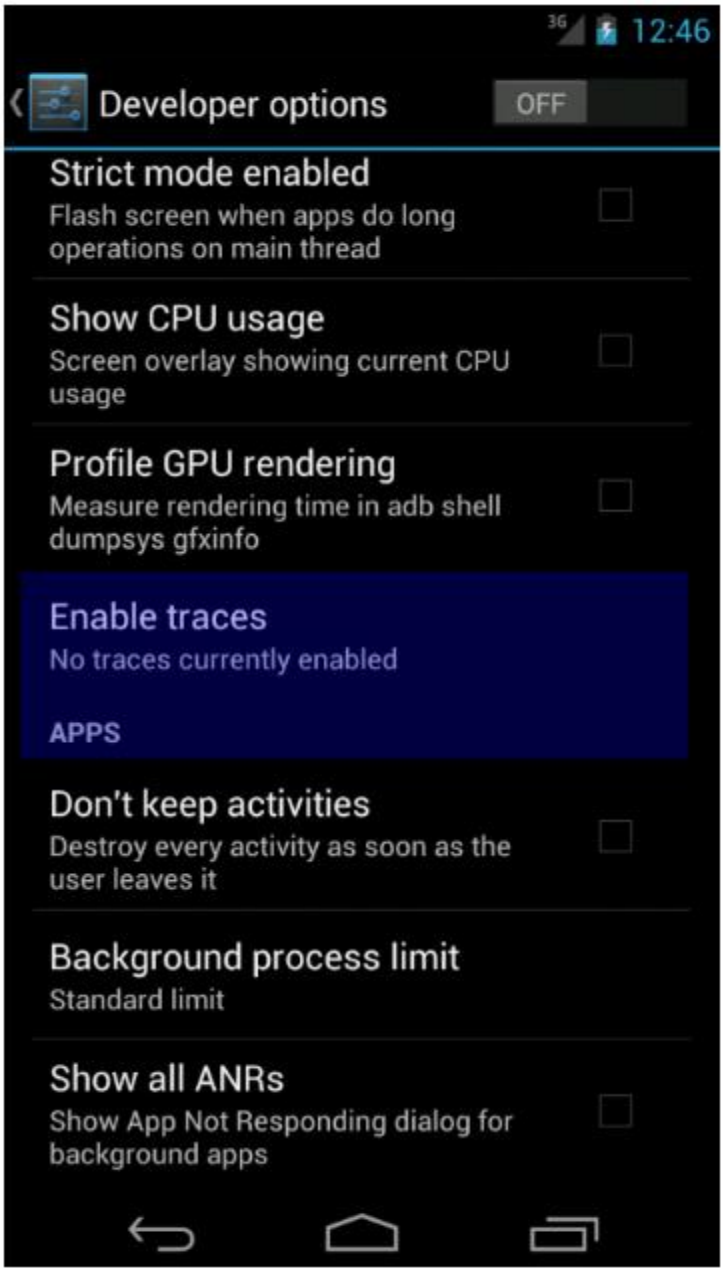
در صورت تمایل می توانید یک روگرفت یا تصویر لحظه ای از حافظه تهیه کرده و سپس آن را به واسطه ی ابزار Memory Analyzer محیط کاری Eclipse بررسی و تحلیل نمایید.

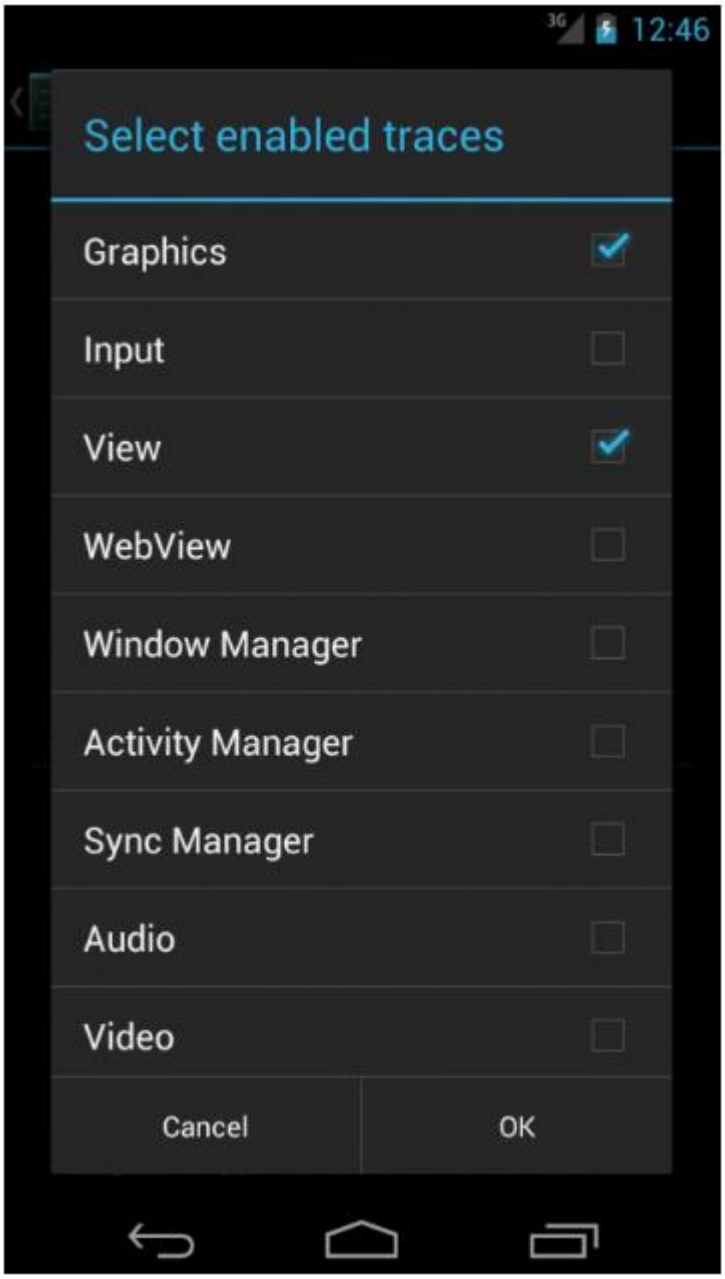
7-54-ابزار Systrace

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

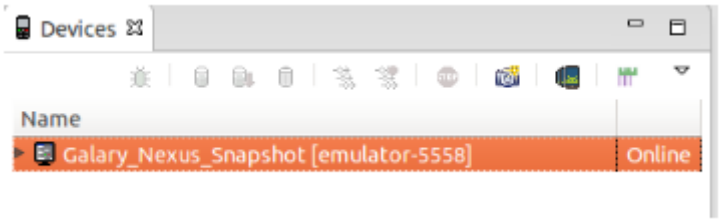
Systrace به شما این امکان را می دهد تا کارایی اپلیکیشن را خود به طور مسقیم در سطح هسته ی سیستم (kernel level) بررسی نمایید. جهت فعال سازی این ابزار، پس از انتخاب Developer options، المان Enable traces را کلیک نمایید. در کادر محاوره ی (dialog) بعدی، می توانید به طور صریح مشخص نمایید از کدام event ها، Graphics and View به عنوان مثال، بایستی گزارش تحلیلی و جزئی تهیه شود (profile شود).

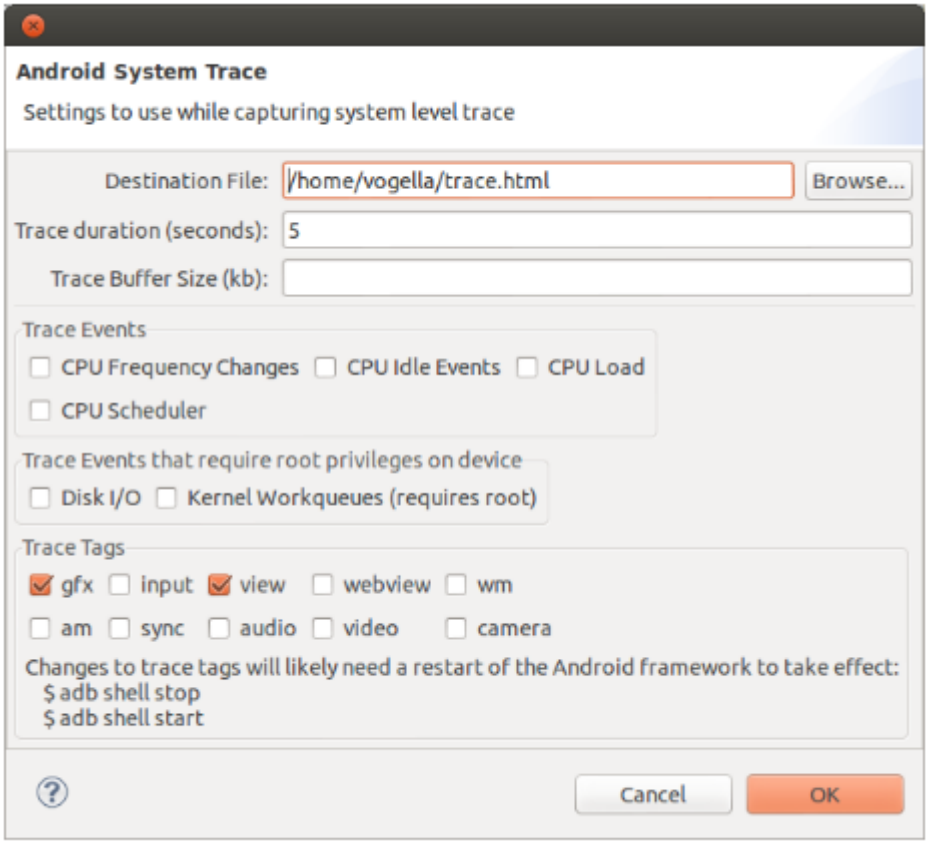




به منظور استفاده از ابزار systrace، یک پنجره ی فرمان (terminal) باز کرده، سپس systrace.py را از پوشه ی اصلی (directory) android_sdk_installdir/tools/systrace اجرا نمایید. ممکن است لازم باشد آن را بر روی یک فایل اجرایی (executable) تنظیم نمایید (در لینوکس chmod a+x systrace.py).

در صورت تمایل می توانید ابزار مزبور را از طریق DDMS مستقیما از محیط کاری Eclipse اجرا نمایید.





ابزار Systrace رخدادها را به مدت پنج ثانیه ضبط (capture) می کند. در نتیجه، ابزار نام برده یک فایل HTML ایجاد کرده و به شما این امکان را می دهد تا مشکلات احتمالی را تجزیه و تحلیل نمایید.

7-55-شبیه سازی چگالی یا تراکم پیکسلی (pixel density) از طریق خط دستور

شما می توانید با استفاده از خط دستور (command line)، چگالی پیکسلی و وضوح تصویر دستگاه های مختلف اندروید را شبیه سازی کنید. این امر به شما اجازه می دهد تا از یک دستگاه استفاده کرده و به وسیله ی آن، چگالی پیکسلی و وضوح تصویر دیگر دستگاه های اندروید را شبیه سازی نمایید.

```
// Set the display size
adb shell am display-size 600x800
// Set the display density
adb shell am display-density 80
```

7-56-قالب های آماده یا الگوهای پیاده سازی پروژه ی اندروید (Android template)

شما می توانید قالب های آماده دلخواه خود را برای ویزاد ساخت پروژه ی اندروید (Android project creation wizard) تعریف نمایید. جهت دستیابی به اطلاعات بیشتر در این زمینه می توانید به لینک های زیر مراجعه نمایید:

.https://plus.google.com/113735310430199015092/posts/XTKTamk4As8

قالب و الگوهای بیشتر کدنویسی (code template) تحت آدرس https://github.com/jgilfelt/android-adt-templates قابل دسترسی می باشد.

7-57-گزارش گیری و ثبت اطلاعات مربوط به کارایی موتور گرافیکی (ابزار Profile GPU rendering)

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در بخش Developer Options، داخل Settings یا در دستگاه واقعی اندروید خود، این امکان برای شما وجود دارد که ابزار Profile GPU rendering را فعال نموده و مدت زمان مورد نیاز برای ترسیم 128 فریم توسط سیستم را رصد و ثبت نمایید.

پس از فعال سازی این امکان و راه اندازی مجدد اپلیکیشن، می توانید با فراخوانی دستور زیر در پنجره ی فرمان، اطلاعات مربوطه را واکشی نمایید.

```
adb shell dumpsys gfxinfo your_package
```

برای مثال جهت سنجش frame rate (سرعت پخش فریم ها) اپلیکیشن با پیمایش به سمت پایین در یک لیست. در بیشتر موارد، لازم است با اپلیکیشن خود تعامل کنید تا مجبور شود خود را مجددا ترسیم کند.

در گزارش نهایی که در اختیار شما قرار می گیرد، بخشی به نام Profile data in ms را پیدا کنید.

7-58-تحلیل و بررسی وضعیت Overdraw (ترسیم یک آیتم بر روی آیتم دیگر)

Overdraw زمانی رخ می دهد که دو آیتم همپوشانی داشته باشند، بدین معنی یک آیتم بر روی آیتم دیگری رسم شود. به طور مثال، می توان به زمانی اشاره کرد که پس زمینه ی کامپوننت activity، یک Window باشد. حال اگر المان TextView به اپلیکیشن اضافه شود، این المان طبیعتا بر روی Window ترسیم شده و قرار می گیرد.

بنابراین وضعیت overdraw یک امر طبیعی بوده و رخداد آن مورد انتظار می باشد. با این حال توصیه می شود جهت بالا بردن کارایی اپلیکیشن و تجربه ی کاربری، از ترسیم آبجکت های متعدد بر روی هم و رخداد overdraw به صورت بیش از حد خودداری شود.

دلیل اصلی رخداد overdraw بی رویه می تواند view hierarchies و نمودار درختی بسیار پیچیده ی view ها باشد. به طور کلی وضعیت 2x overdraw (زمانی که یک پیکسل سه بار ترسیم می شود) یک امر طبیعی است اما بیشتر از آن می بایست اجتناب شود چرا که افت کارایی را به دنبال دارد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

می توانید از طریق Development Settings و فعال گردن گزینه ی Show GPU overdraw، وضعیت رخداده ی overdraw و view های ترسیم شده بر روی هم را به صورت دیداری در نمایشگر داشته باشید. این تنظیم بر اساس overdraw ها، رنگ به صفحه نمایش تخصیص می دهد. جدول زیر مجموعه رنگ بکار رفته را همراه با معنی هر یک شرح می دهد.

رنگ مربوطه	معنی رنگ
-	هیچ view ای بر روی view دیگر ترسیم نشده. به عبارت دیگر، وضعیت overdraw رخ نداده است.
آبی	1x overdraw = یک پیکسل دو بار رسم شده است.
سبز	2x
قرمز روشن	3x = یک پیکسل سه بار رسم شده و می تواند کمی مشکل زا باشد. اما با این وجود ،نواحی کوچک قرمز هنوز مشکل جدی ایجاد نمی کنند.
قرمز تیره	4x = یک پیکسل 5 بار یا حتی بیشتر ترسیم شده است. این رنگ نشانگر بروز مشکل می باشد.

پس از دریافت نسخه ی دیداری و بررسی بخش هایی که مشکل زا بودند، می توانید نمودار درختی View را با ابزار Hierarchy Viewer تحلیل نمایید.

7-59-ابزار رایگان و متن باز کارا برای تحلیل و بهینه سازی کارایی اپلیکیشن

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

1-59-7-استفاده از Leak Canary برای یافت و برطرف نمودن هدر رفت حافظه (memory leak)

Leak Canary یک ابزار متن باز است که می توان با اپلیکیشن اندروید ادغام نموده و به کمک آن هدر رفت حافظه را به صورت خودکار شناسایی کرد. استفاده از ابزار نام برده بسیار آسان است و شما می توانید آن را به عنوان کتابخانه (dependency) به فایل Gradle build اضافه نموده و سپس کتابخانه ی مزبور را در کلاس application خود مقداردهی اولیه (initialize) نمایید. با این کار یک lifecycle listener و گوش فراخوان به چرخه ی حیات activity ها اضافه شده و دقیقا بررسی می کند آیا متد destroy آن ها در زمان مناسب فراخوانی می شود یا خیر.

لازم به توضیح است که Canary leak به علت open source بودن به صورت مداوم ویرایش و بروز رسانی می شود. جهت اطلاع از نحوه ی بکارگیری این ابزار به آدرس https://github.com/square/leakcanary، صفحه ی Leak Canarys Github مراجعه نمایید.

2-59-7-استفاده از AndroidDevMetrics جهت مشاهده ی اطلاعات مربوط به کارایی اپلیکیشن

ابزار AndroidDevMetrics به توسعه دهنده این امکان را می دهد تا سرعت اجرای عملیات معمول همچون مقداردهی اولیه ی آبجکت یا فراخوانی توابع مدیریت چرخه ی حیات Activity (lifecycle) (از جمله onCreate()،onStart() و onResume()) را تحلیل نماید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

همان طور که درباره ی ابزار قبلی گفته شد، AndroidDevMetrics یک پروژه ی متن باز (Open Source) است. بدین معنی که ممکن است به صورت مداوم بروز رسانی شده و مورد ویرایش قرار گیرد. برای اطلاع از نحوه ی استفاده از این ابزار متن باز می توانید به آدرس https://github.com/frogermcs/androiddevmetrics - صفحه ی AndroidDevMetrics مراجعه نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فصل هشتم

ایجاد اپلیکیشن اندروید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش اول :

کامپایل / Build پروژه های اندرویدی با سیستم Gradle

آموزش حاضر نحوه ی استفاده از سیستم Gradle برای کامپایل پروژه های اندرویدی را شرح می دهد. در این مبحث با نحوه ی تنظیم نسخه های مختلف build (flavor) آشنا خواهید شد.

8-1-استفاده از Gradle برای کامپایل پروژه های اندرویدی

8-1-1-فایل های build اپلیکیشن های اندروید

فرایند build پروژه های اندرویدی توسط سیستم کامپایل Gradle مدیریت می شود. زمانی که پروژه ی جدیدی در محیط برنامه نویسی اندروید ایجاد می کنید، به دنبال آن build script (فایل build.gradle) نیز به طور خودکار تولید می شوند. در واقع محیط کاری Android Studio به صورت خودکار runtime و بستر اجرای Gradle را دربرمی گیرد، از این رو احتیاجی به نصب ابزار یا افزونه ی جداگانه وجود ندارد.

سیستم Gradle build طوری طراحی شده که از سناریوهای پیچیده مختلف برای ایجاد اپلیکیشن های اندرویدی پشتیبانی کند:

- Multi-distribution: وضعیتی که یک اپلیکیشن بایستی برای چندین سرویس گیرنده یا شرکت اختصاصی تنظیم شود.
- Multi-apk: تولید چندین فایل apk از یک اپلیکیشن جهت سازگاری با انواع دستگاه ها و استفاده ی مجدد از بخش هایی از کد برنامه در این فایل های apk.

همچنین می توانید از یک wrapper script که سیستم Gradle تولید می کند، استفاده نمایید (wrapper script: wrapper script یک اسکریپت است که چندین دستور و اسکریپت که قابلیت

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

اجرا در برنامه ی اصلی را ندارد، در بر می گیرد). این wrapper به شما امکان می دهد تا یک Gradle build را بدون اینکه لازم باشد هیچ ابزار و فایل دیگری نصب کنید، از خط دستور (command line) اجرا نمایید.

زمانی که بر روی دکمه ی run در محیط کاری Android Studio کلیک می کنید، تسک مربوطه ی Gradle راه اندازی شده و اپلیکیشن اجرا می شود.

نکته: می توانید به ورژن های مختلف افزونه ی Gradle تحت آدرس https://jcenter.bintray.com/com/android/tools/build/gradle/ دسترسی داشته باشید.

2-1-8-پروسه ای که کد برای تبدیل به اپلیکیشن اندروید طی می کند

کامپایلر جاوا فایل های حاوی کدهای جاوا (java source file) را به فایل هایی با پسوند class. تبدیل می کند (فایل هایی که دربردانده ی bytecode ها و کدهای زبان میانی بوده و بر روی JVM اجرا می شود). Android SDK یک ابزار به نام dx دارد که فایل های class. را به فایل هایی با پسوند dex. (فایل های اجرایی Dalvik) تبدیل می کند. تمامی فایل های کلاس اپلیکیشن در فایل اجرایی Dalvik جایگذاری می شوند. در طی این پروسه تبدیل، اطلاعات مازاد موجود در فایل های class. داخل فایل dex. بهینه می شوند. برای مثال، چناچه یک String در چندین فایل class. وجود داشته باشد، فایل dex. تنها یک reference یا اشاره گر به متغیر مزبور را نزد خود نگه می دارد.

فایل های dex. از نظر حجم بسیار سبک تر از فایل های class. متناظر هستند.

فایل dex. و سایر منابع و محتوا همچون image و فایل های XML همگی در قالب یک فایل apk. گنجانده می شوند. این وظیفه را ابزار پکیج بندی منابع و محتوای اپلیکیشن های اندرویدی به نام aapt (Android Asset Packaging Tool) بر عهده دارد.

فایل apk. خروجی، دربردارنده ی تمامی داده های لازم برای اجرای اپلیکیشن اندروید بوده و به راحتی از طریق ابزار adb قابل نصب بر روی دستگاه اندرویدی می باشد.

از ویرایش 5.0 به بعد، Android RunTime به عنوان ابزار بارگذاری و اجرا / runtime تمامی اپلیکیشن های اندرویدی ایفای نقش می کند. ART یک قابلیت است که اپلیکیشن را درست به مجرد درخواست کاربر لود کرده و اجرا می کند. بدین وسیله سرعت اجرای اپلیکیشن به مراتب نسبت به زمانی که از Dalvik استفاده می شد، بالا می رود. در واقع ART از تلفیقی از دو روش کامپایل Ahead of Time (ترجمه ی کد زودتر از درخواست کاربر) و Just-in-Time (ترجمه ی کد به محض درخواست کاربر) استفاده می کند. به هنگام نصب یک اپلیکیشن بر روی دستگاه اندرویدی، کد اپلیکیشن به زبان ماشین ترجمه می شود.

ابزار dex2oat فایل dex. که خروجی مجموعه ابزار اندروید می باشد را تحویل گرفته و آن را به یک فایل (EFL) با فرمت Executable and Linkable تبدیل می کند. این فایل حاوی کد dex، کد کامپایل شده برای سخت افزار و محیط جاوا / native code و meta-data می باشد.

راه اندازی Gradle از خط دستور (command line)

می توانید فایل Gradle build خود را از خط دستور اجرا کنید. برای این منظور، دستور زیر را از پوشه ی اصلی پروژه (main project directory) اجرا نمایید. لازم است Gradle بر روی دستگاه مربوطه نصب شده باشد و یا کد دربرگیرنده ی Gradle (Gradle wrapper gradlew) را برای اجرای build بکار ببرید. در صورت لزوم wrapper خود به صورت خودکار سیستم Gradle را دانلود می کند. در زیر لیست task های مهم Android Gradle را با شرح کاربرد مشاهده می شود:

شرح کاربرد	دستور
پروژه را build و کامپایل نموده و هر دو تسک assemble و check را اجرا می کند.	./gradlew build
پروژه را کامل از صفر کامپایل و build می کند.	./gradlew clean build
پروژه را از صفر build و کامپایل می کند.	./gradlew clean build
تست ها را اجرا می کند.	./gradlew test
تست های مبتنی بر instrumentation را اجرا می کند.	./gradlew connectedAndroidTest

جهت مشاهده ی تمامی task های موجود، دستور gradle wrapper را فراخوانی کنید.

```
gradle build
# alternatively speedup second grandle build by holding it in memory
# gradle build --daemon
```

با اجرای این دستور، یک پوشه ی build برای قرارگیری خروجی Gradle build ایجاد می شود. به صورت پیش فرض، Gradle build دو فایل با پسوند apk. در پوشه ی build/outputs/apk ایجاد می کند.

به منظور build و راه اندازی تست های نرم افزاری unit test خود بر روی دستگاه مجازی جاوا (JVM)، دستور زیر را فراخوانی کنید.

```
gradle test
```

حال جهت کامپایل، build و راه اندازی تست های مبتنی بر instrumentation خود بر روی دستگاه واقعی اندروید، دستور زیر را اجرا نمایید.

```
gradle connectedCheck
```

8-1-3-یا resource shrinking

سیستم کامپایل و build پروژه Gradle، قادر است با استفاده از روش بهینه سازی منابع (resource shrinking) در زمان کامپایل و build پروژه، کلاس ها و منابع غیرضروری جاوا را حذف نماید. بدین معنی که منابع بلااستفاده از پکیج اپلیکیشن به صورت خودکار حذف می شوند. علاوه بر آن، تمامی منابع غیرضروری از کتابخانه هایی که مورد استفاده ی پروژه هستند، حذف شده حجم نرم افزار به طور قابل توجهی کاهش می یابد.

به منظور فعال سازی قابلیت بیهنه سازی منابع (resource shrinking)، محتوای فایل build خود را به صورت زیر ویرایش کنید.

```
android {
   ...
   buildTypes {
      release {
         minifyEnabled true
         shrinkResources true
         proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
      }
   }
}
```

8-1-4-تعریف dependency ها و اعلان ورژن کتابخانه های لازم خارج از بدنه ی بستار dependencies (closure)

یک روش بهینه که استفاده از آن بسیار توصیه می شود، تعریف ورژن نیازمندی های کتابخانه ی خود (library dependencies) خارج از بدنه ی بستار dependencies (closure) می باشد. بدین وسیله نگهداشت و maintenance به غایت آسان تر می شود.

```
ext {
   // App dependencies
   junitVersion = '4.12'
   mockitoVersion = '1.10.19'
   powerMockito = '1.6.2'
   hamcrestVersion = '1.3'
}
dependencies {
   // Dependencies for local unit tests
   testCompile "junit:junit:$junitVersion"
   testCompile "org.mockito:mockito-all:$mockitoVersion"
   testCompile "org.hamcrest:hamcrest-all:$hamcrestVersion"
   testCompile "org.powermock:powermock-module-junit4:$powerMockito"
```

```
    testCompile "org.powermock:powermock-api-mockito:$ext.powerMockito"
}
```

توجه: پس از قرار دادن بستار ext داخل فایل build اصلی (root)، می توانید، برای مثال، با پارامتر 'rootProject.ext.junitVersion$' به راحتی به property ها و خواص آن دسترسی داشته باشید.

build-2-8 و کامپایل ورژن های (flavor) مختلف از اپلیکیشن اندرویدی خود

8-2-1- انواع build (build flavor)

به طور پیش فرض، اندروید دو نوع build دارد: debug (ورژن debug دربردارنده ی کد برای اشکال زدایی و مقداری گزارش یا logging) و release (ورژن آماده و بهینه سازی شده برای اجرا). برای این انواع build،

می توانید flavor های مختلف در Gradle build ایجاد نمایید (build دو ورژن دارد که هر دو یک نسخه از اپلیکیشن هستند، اما flavor برای سرویس گیرنده و مشتری های مختلف تولید می شوند مانند نسخه ی پولی و رایگان از یک اپلیکیشن).

سیستم Gradle build قادر است flavor های مختلفی از یک اپلیکیشن تولید و مدیریت کند. product flavor بیانگر یک ورژن اختصاصی از اپلیکیشن می باشد. این امر امکانی را فراهم می کند تا بخش هایی از codebase یا منابع مورد استفاده ی پروژه برای نسخه های مختلف اپلیکیشن مورد نظر متفاوت باشد.

برای مثال، می توانید برای انواع دستگاه ها همچون گوشی یا تبلت (device category)، ورژن های مختلف (build varaint) تولید نمایید. به عنوان یک مورد استفاده ی دیگر می توان به نسخه ی پولی و رایگان اپلیکیشن مورد نظر اشاره کرد. همچنین زمانی که قصد دارید به هنگام اجرای تست بر روی اپلیکیشن، منابع و کلاس های متفاوت را بکار ببرید.

2-2-8-تعریف انواع flavor در فایل Gradle build

می توانید با استفاده از بستار productFlavors در محتوای فایل app/build.gradle، نسخه های مختلف از اپلیکیشن نهایی خود اعلان نمایید.

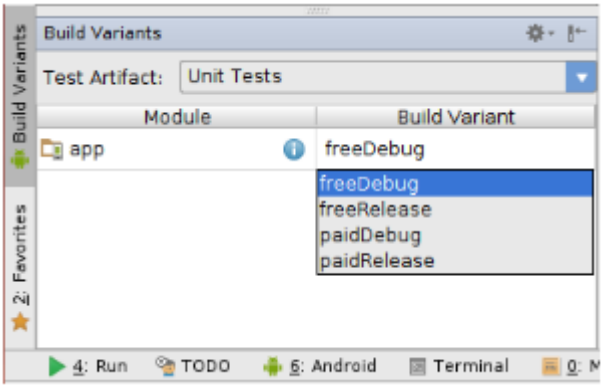
```
productFlavors {
   prod {
      applicationId = "com.vogella.android.gradlebuildflavors.prod"
      versionName = "1.0-paid"
   }
   mock {
      applicationId = "com.vogella.android.gradlebuildflavors.mock"
      versionName = "1.0-free"
   }
}
```

محتوای کل فایل build.gradle ممکن است ظاهری مشابه زیر داشته باشد.

```
apply plugin: 'com.android.application'
android {
   compileSdkVersion 22
   buildToolsVersion "22.0.1"
   defaultConfig {
      applicationId "com.exam.gradleexamples"
      minSdkVersion 19
      targetSdkVersion 22
      versionCode 1
      versionName "1.0"
   }
   buildTypes {
      release {
         minifyEnabled false
         proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
      }
   }
   productFlavors {
      prod {
         applicationId = "com.vogella.android.gradlebuildflavors.prod"
         versionName = "1.0-paid"
      }
      mock {
         applicationId = "com.vogella.android.gradlebuildflavors.mock"
         versionName = "1.0-free"
      }
   }
}
dependencies {
   compile fileTree(dir: 'libs', include: ['*.jar'])
   compile 'com.android.support:appcompat-v7:22.0.0'
```

```
    testCompile 'junit:junit:4.+'
}
```

پس از اعلان flavor و ورژن های مختلف از اپلیکیشن، می توانید داخل محیط کاری Android Studio، تمامی ورژن های مختلف از اپلیکیشن خود را در کادر Build Variants مشاهده و انتخاب نمایید.



8-2-3-ارائه ی منابع مختلف برای flavor ها و ورژن های مختلف اپلیکیشن

به منظور تعریف قابلیت و امکانات مختلف برای flavor مورد نظر، می بایست پوشه های مختلفی برای flavor های مشخص شده در پوشه ی app/src/ ایجاد نمایید.

منابع مختص به هر flavor، منابع کلی و اصلی را بازنویسی می کند. به طور مثال، زمانی یک آیکون متفاوت برای flavor معین تعیین کرده باشید، سیستم build و کامپایل Android، آن آیکونی که ویژه ی flavor مورد نظر تعریف شده را به عنوان آیکون اپلیکیشن انتخاب می کند.

8-2-4-تعریف source set های متفاوت برا flavor های مختلف از اپلیکیشن

دایرکتوری های مقیم در پوشه ی src/ در اصطلاح source sets شناخته می شود. هر flavor از نرم افزار می تواند source set اختصاصی خود را داشته باشد.

لازم به ذکر است که code file ها (فایل های حاوی کدهای جاوا) مانند resource ها و منابع پروژه جایگزین نمی شوند، بلکه با یکدیگر ادغام می گردند. به طور مثال، شما نمی توانید در یک flavor از اپلیکیشن خود اکتیویتی com.example.MainActivity را در پوشه ی app/main/java/ داشته

و در flavor دیگر، نسخه ی پیاده سازی شده ی دیگری از activity مزبور داشته باشید. در صورتی که دو نسخه ی پیاده سازی شده از activity داشته باشید، یک پیغام خطا در خصوص وجود کلاس های تکراری (duplicate class) صادر می شود.

با این وجود می توانید با جلوگیری از ایجاد کلاس در source folder اصلی پروژه و سپس ایجاد یک کلاس مجزا در هر flavor، برای هر flavor از اپلیکیشن خود نسخه ی پیاده سازی شده ی متفاوتی از یک کلاس (برای مثال activity) داشته باشید.

تمرین: ساخت اپلیکیشن های اندرویدی با flavor های مختلف

هدف اصلی از این تمرین

در تمرین جاری، یک اپلیکیشن اندرویدی با دو flavor مختلف به ترتیب به نام های prod و mock ایجاد می کنید.

نسخه ی mock منابع متفاوتی از نسخه ی prod مورد استفاده قرار می دهد. در نمونه ی اول، فایل strings.xml مستقر در folder/project بازنویسی (override) می شود. اینکه کدام ورژن build باشد، از طریق کادر Build Variants قابل تعریف است.

ساخت پروژه ی جدید اندروید

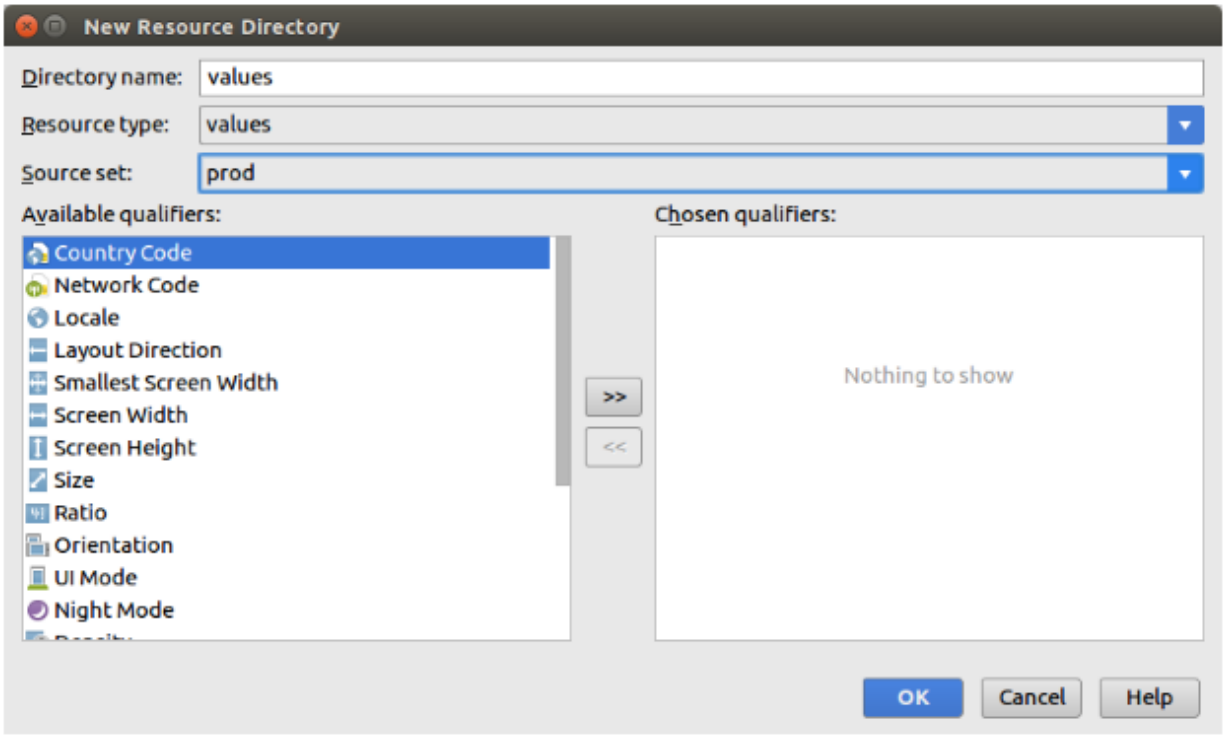
یک پروژه ی جدید بر اساس قالب آماده Empty Activity (template) و با اسم پکیج com.vogella.android.gradlebuildflavors ایجاد نمایید.

دو flavor جدید در فایل app/build.gradle به نام های "prod" و "mock" تعریف کنید.

```
apply plugin: 'com.android.application'
android {
   compileSdkVersion 22
   buildToolsVersion "22.0.1"
   defaultConfig {
      applicationId "com.exam.gradleexamples"
```

```
        minSdkVersion 19
        targetSdkVersion 22
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
    productFlavors {
        prod {
            applicationId = "com.vogella.android.gradlebuildflavors.prod"
            versionName = "1.0-paid"
        }
        mock {
            applicationId = "com.vogella.android.gradlebuildflavors.mock"
            versionName = "1.0-free"
        }
    }
}
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:22.0.0'
    testCompile 'junit:junit:4.+'
}
```

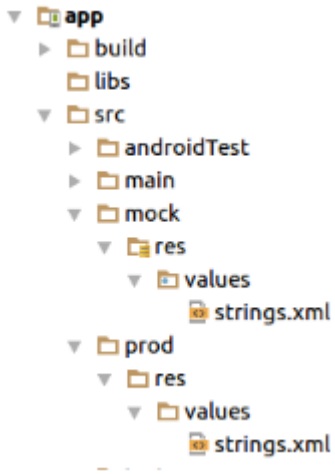
ساختار درختی پوشه ها (folder structure) برای دو flavor نام برده را ایجاد نمایید. اگر Android resource directory را انتخاب کنید، در آن صورت می توانید flavor مورد نظر را در پنجره ی محاوره ای زیر، همانند تصویر حاضر، انتخاب نمایید.



فایل strings.xml را از پوشه ی main به پوشه ی مناسب از flavor مربوطه کپی کرده و جایگذاری نمایید.

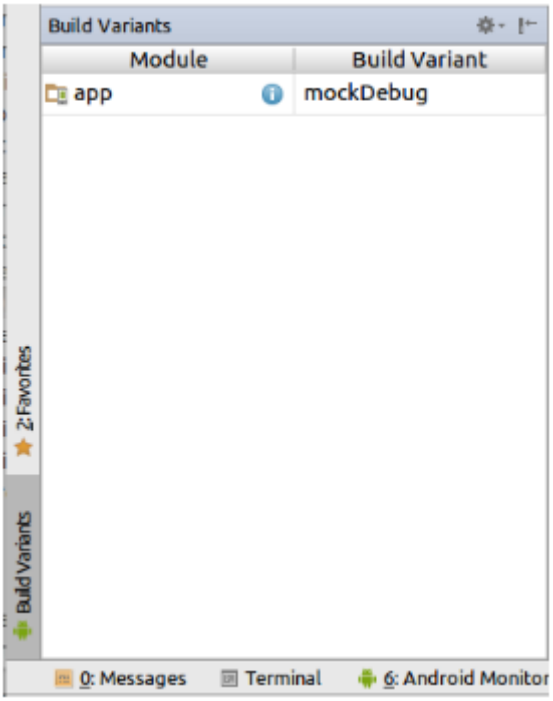
مقدار متغیر رشته ای hello_world از فایل strings.xml را برای هر flavor به ترتیب به مقدارهای Mock World! و Prod World! تغییر دهید.

```
<resources>
   <string name="app_name">Flavor</string>
   <string name="hello_world">Mock World! </string>
</resources>

<resources>
   <string name="app_name">Flavor</string>
   <string name="hello_world">Prod World! </string>
</resources>
```



5-2-8-اعتبارسنجی و تست پروژه

در کادر Build Variant محیط کاری Android Studio، گزینه ی mockDebug را انتخاب کرده و اپلیکیشن را جهت تست اجرا نمایید.



زمانی که اپلیکیشن را اجرا می کنید، می بایست رشته ی مربوطه (Mock World!) را از mock flavor مشاهده نمایید. اکنون prod flavor را انتخاب کرده و اجرا نمایید. مقدار رشته ای مربوطه (Prod Word!) بایستی در نمایشگر قابل مشاهده باشد.

8-2-6-کامپایل و build پروژه از طریق خط دستور Gradle

می توانید با درج دستور gradlew build/. در پنجره ی فرمان، تمامی flavor های اپلیکیشن خود را اجرا نمایید.

8-2-7-تست ورژن یا flavor های مختلف از یک اپلیکشن (gradle flavor های یک اپلیکیشن)

کلاسی به نام ShareIntentBuilder با پیاده سازی زیر ایجاد کنید. این کلاس به وسیله ی یک آبجکت intent که داده ها را به اشتراک می گذارد (share intent) و یک متد static، activity دیگری را راه اندازی می نماید.

```
import android.content.Context;
import android.content.Intent;
public class ShareIntentBuilder {
    public static void startSendActivity(Context context, String title, String body) {
        Intent intent = new Intent();
        intent.setAction(Intent.ACTION_SEND);
        intent.putExtra(Intent.EXTRA_TITLE, title);
        intent.putExtra(Intent.EXTRA_TEXT, body);
        intent.setType("text/plain");
        Intent chooserIntent = Intent.createChooser(intent, context.getResources().getText(R.string.share));
        chooserIntent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        context.startActivity(chooserIntent);
    }
```

8-2-8-پیاده سازی نسخه های مختلف از کلاس MainActivity در flavor مورد نظر از اپلیکیشن

اجازه دهید activity ای که این آبجکت intent را در سطح خود تعریف کرده و به عنوان پارامتر به بیرون ارسال می کند، در mock flavor جایگزین گردد. چنانچه flavor "mock" انتخاب شده بود، یک activity دیگر راه اندازی کنید که داده های ارسالی را نمایش می دهد. در صورتی که "prod" flavor انتخاب شده باشد، intent مشترک بین دو activity را ارسال کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

توجه: لازم به ذکر است که کلاس ها قابل بازنویسی نیستند. شما بایستی کلاس ها را در flavor های ویژه ی هر یک و نه در flavor اصلی ایجاد کنید.

8-3-تنظیم اختصاصی فایل Gradle build

8-3-1-ویرایش اسم فایل apk خروجی

```
apply plugin: 'com.android.application'
android {
    // more
  buildTypes {
    release {
      minifyEnabled false
      proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
    }
    applicationVariants.all { variant ->
      variant.outputs.each { output ->
        def file = output.outputFile
        def filename = file.name.replace("app", "lars")
        output.outputFile = new File(file.parent, filename)
      }
    }
// more
}
```

8-3-2-تعریف keystore مجزا برای debug build

می توانید در فایل build.gradle خود یک keystore مجزا تعریف نمایید (keystore: برای درج امضای خالق یک پروژه ی اندرویدی بر روی فایل APK خروجی از keystore استفاده می شود). برای مشاهده ی جزئیات بیشتر به آدرس http://tools.android.com/tech-docs/new-build-system/user-guide مراجعه فرمایید.

به طور مثال می توانید Keystore را ویژه ی نسخه ی debug اپلیکیشن مورد نظر به صورت زیر ویرایش نمایید.

```
android {
  signingConfigs {
    debug {
      storeFile file("your.keystore")
    }
  }
```

}

8-4-انتقال / migrate کردن یک پروژه ی خروجی گرفته شده از محیط Eclipse به Gradle

8–4–1–وارد کردن (import) یک پروژه ی خروجی گرفته شده از Eclipse در محیط کاری Android Studio

پروژه های اندرویدی به دو روش متفاوت (با دو نحوه ی تنظیم و پیکربندی) سازمان دهی می شوند: 1. یک روش پیکربندی قدیمی که تا سال 2013 توسط ابزار ساخت و توسعه ی اپلیکیشن های اندرویدی محیط کاری Eclipse (Eclipse ADT) استفاده می شد 2. دیگری از ساختار درختی جدید برای سازماندهی پروژه به نام Gradle build structure استفاده می کند. می توان سیستم Gradle را طوری تنظیم کرد که از هر دو فرمت ذکر شده برای پیکربندی پروژه استفاده کند.

پس از اینکه فایل Gradle مورد نیاز را به پروژه ی اندرویدی خروجی گرفته شده از Eclipse اضافه نمودید، می توانید پروژه ی خود را وارد محیط برنامه نویسی Android Studio کنید. برای این منظور کافی است داخل محیط مزبور، بر روی File از منوی اصلی کلیک کرده و گزینه ی Import Project را انتخاب نمایید. سپس پوشه ی پروژه و فایل Gradle build مربوطه را انتخاب کنید.

8–4–2–اضافه کردن فایل Gradle به پروژه ی اندرویدی که از محیط Eclipse خروجی گرفته شده است

برای نیل به این هدف، فایل build.gradle خود را به پوشه ی اصلی (root) پروژه اضافه نمایید.

```
buildscript {
   repositories {
      mavenCentral()
   }
   dependencies {
      classpath 'com.android.tools.build:gradle:2.2.0-beta3'
   }
}
apply plugin: 'com.android.application'
android {
    lintOptions {
        abortOnError false
    }
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

```
    compileSdkVersion 22
    buildToolsVersion "21.1.2"
            defaultConfig {
                targetSdkVersion 22
            }
    sourceSets {
        main {
            manifest.srcFile 'AndroidManifest.xml'
            java.srcDirs = ['src']
            resources.srcDirs = ['src']
            aidl.srcDirs = ['src']
            renderscript.srcDirs = ['src']
            res.srcDirs = ['res']
            assets.srcDirs = ['assets']
        }
        // Move the build types to build-types/<type>
        // For instance, build-types/debug/java, build-types/debug/AndroidManifest.xml, ...
        // This moves them out of them default location under src/<type>/... which would
        // conflict with src/ being used by the main source set.
        // Adding new build types or product flavors should be accompanied
        // by a similar customization.
        debug.setRoot('build-types/debug')
        release.setRoot('build-types/release')
    }
}
```


بخش دوم :

استفاده از سرور کامپایل و اجرای پروژه (build server) Jenkins جهت build و اجرای اپلیکیشن های اندرویدی

آموزش حاضر اطلاعاتی در خصوص نحوه ی استفاده از سرور کامپایل پروژه ی Jenkins جهت build پروژه های اندرویدی در اختیار شما قرار می دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

8-5-کامپایل و اجرای پروژه های اندرویدی با Jenkins

8-5-1-پیش نیازهای کامپایل پروژه (build job) با Jenkins

Jenkins قادر است اپلیکیشن های اندرویدی را به صورت خودکار کامپایل و تست نماید. جهت انجام پروسه ی کامپایل و اجرای پروژه و ایجاد یک build job بر روی Jenkins، بایستی تنظیمات و ابزار مورد نیاز build را به صورت آماده داشته باشید. ابزار ساخت و توسعه ی اپلیکیشن اندروید خود به صورت پیش فرض یک فایل Gradle build معتبر تولید می کند که برنامه نویس می تواند از در Jenkins استفاده کند.

8-5-2-نصب مجموعه ابزار ساخت و توسعه ی نرم افزار اندروید (Android SDK)

لازم است Android SDK از قبل بر روی سرویس دهنده نصب باشد. در صورتی که سرور کامپایل و اجرای پروژه یا build server برنامه ی display server یا هماهنگ سازی ورودی و خروجی کاربر با باقی سیستم عامل، سخت افزار را نداشته باشد، بایستی Android SDK را از طریق خط دستور (command line) اجرا نمایید. همچنین لازم است مکان قرارگیری Android SDK جهت استفاده ی Jenkins را مشخص نمایید.

```
# download the Android SDK via wget
# for the correct URL see http://developer.android.com/sdk/index.html
# and look in the section "SDK Tools Only"
wget <link from the above website>
# example
# wget http://dl.google.com/android/android-sdk_r24.2-linux.tgz
# unpack it
tar zxvf filename
# Add the new directory to your patch
# assumes you exported it to /opt/
# this must be set for the Jenkins user
export ANDROID_HOME="/opt/android-sdk-linux"
export PATH="$ANDROID_HOME/tools:$ANDROID_HOME/platform-tools:$PATH"
// run this as Jenkins user
```

```
android update sdk --no-ui
```

با استفاده از دستور android list targets می توانید ABI های نصب شده را مشاهده کنید (Application Binary Interface یا ABI یک رابط سطح-پایین بین اپلیکیشن و سیستم عامل یا یک نرم‌افزار دیگر ایجاد می‌نماید). در صورت عدم وجود ABI، لازم است آن را مانند زیر نصب نمایید.

```
android update sdk --all --filter sys-img-armeabi-v7a-android-23 --no-ui --force
```

8-5-3-نصب افزونه ها یا plug-in های Jenkins

جهت کامپایل و اجرای پروژه های اندرویدی (build) با Jenkins، لازم اســـت افزونه های زیر را نصب نمایید:

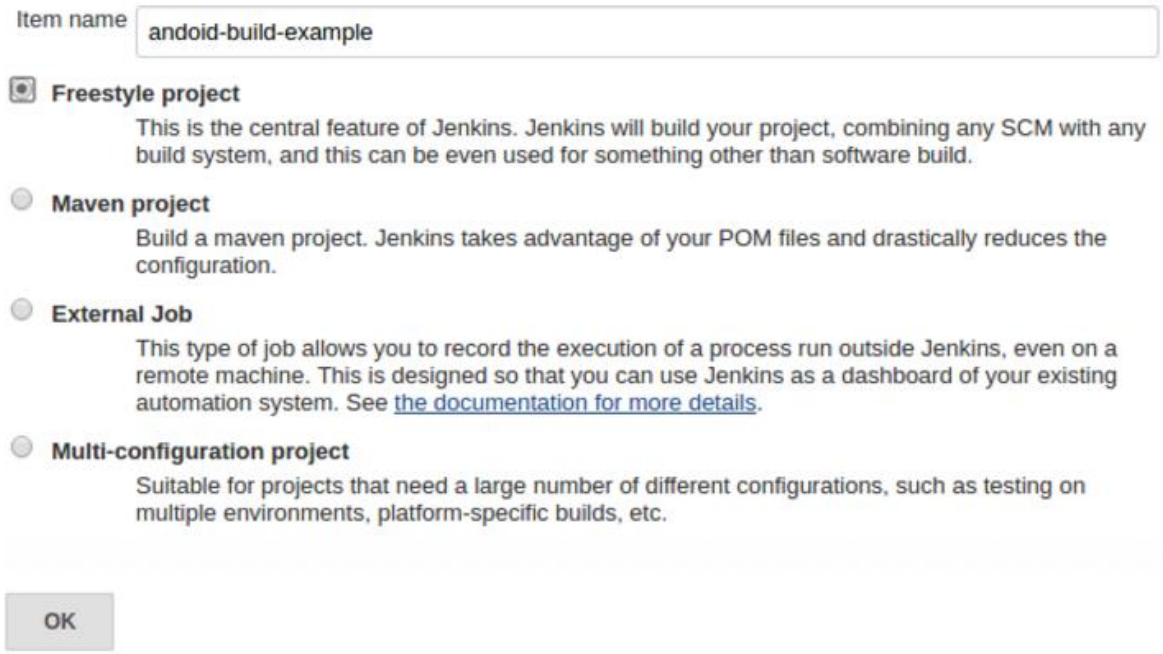
- افزونه ی Gradle.
- افزونه ی Git – در صورتی که Git برای منابع پروژه مورد استفاده قرار گرفته شده باشد.
- افزونه ی شبیه ساز اندروید (Android Emulator) – به توسعه دهنده امکان می دهد تا شـبیه سـاز اندروید (android emulator) را اجرا و فعال کرده و تا زمانی که شـبیه سـاز کاملا اجرا نشـده، پروسـه ی کامپایل و اجرای پروژه (build) را متوقف سـازد. برای تسـت اپلیکشن های اندروید مورد نیاز می باشد.

8-5-4-ایجاد build job برای اپلیکیشن های اندرویدی

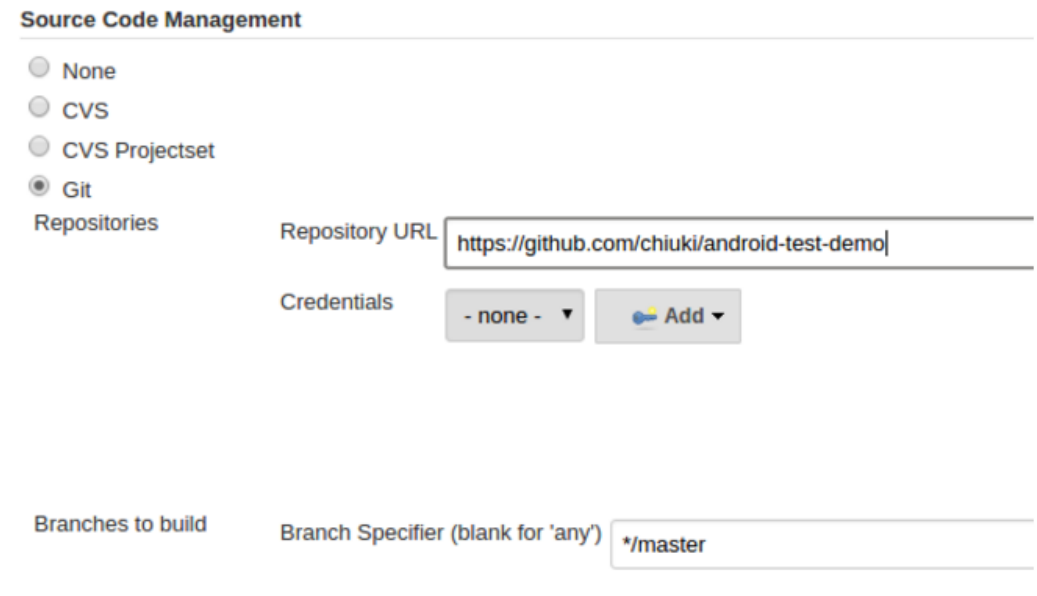
به منظور ایجاد build job های اندرویدی در Jenkins، پس از انتخاب New Job، اسم job مربوطه را وارد کنید. حال گزینه ی Build a free-style software project را انتخاب کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



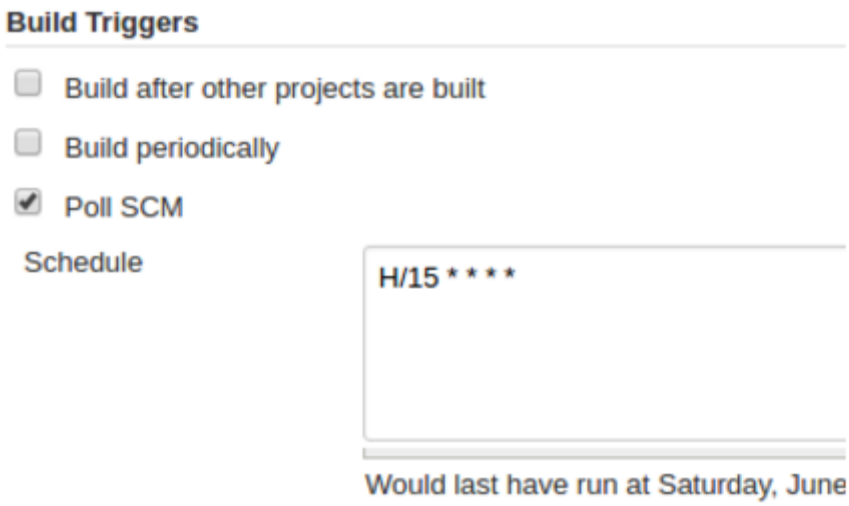
سپس می بایست مشخص کنید که منبع و source باید از کجا کپی (clone) شود.



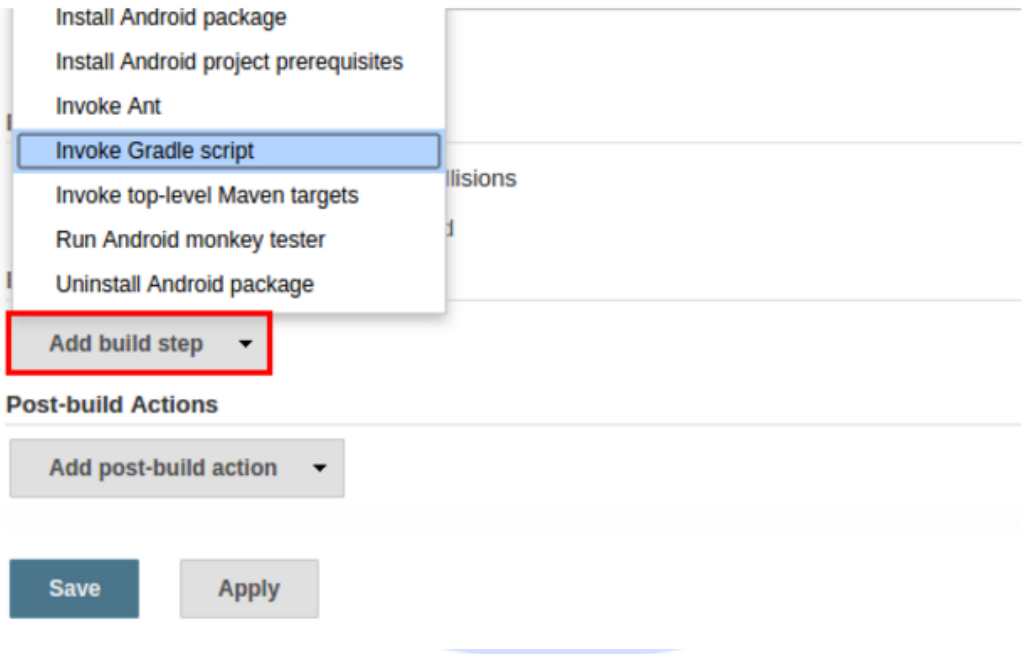
build trigger دلخواه را انتخاب کنید. در تصویر بالا، job و عملیات زمان بندی شده مورد نظر، هر 15 دقیقه یکبار repository انتخابی (Git) را چک می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

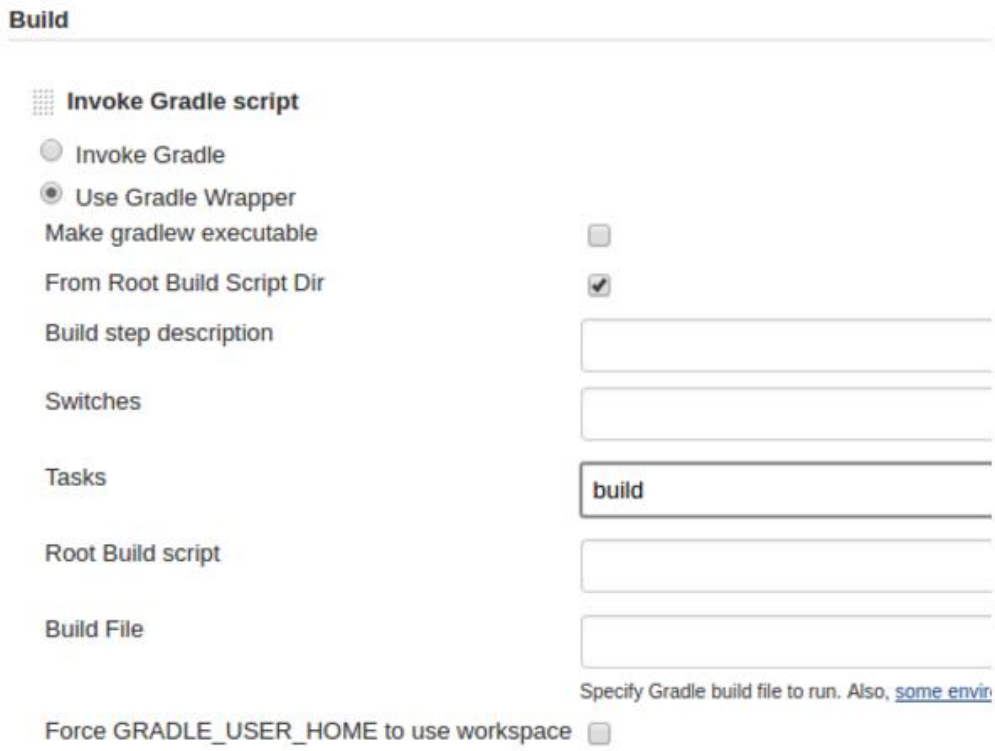


یک مرحله کامپایل و اجرا با سیستم Gradle (Gradle build step) به job ها و عملیات زمان بندی Jenkins اضافه نمایید.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

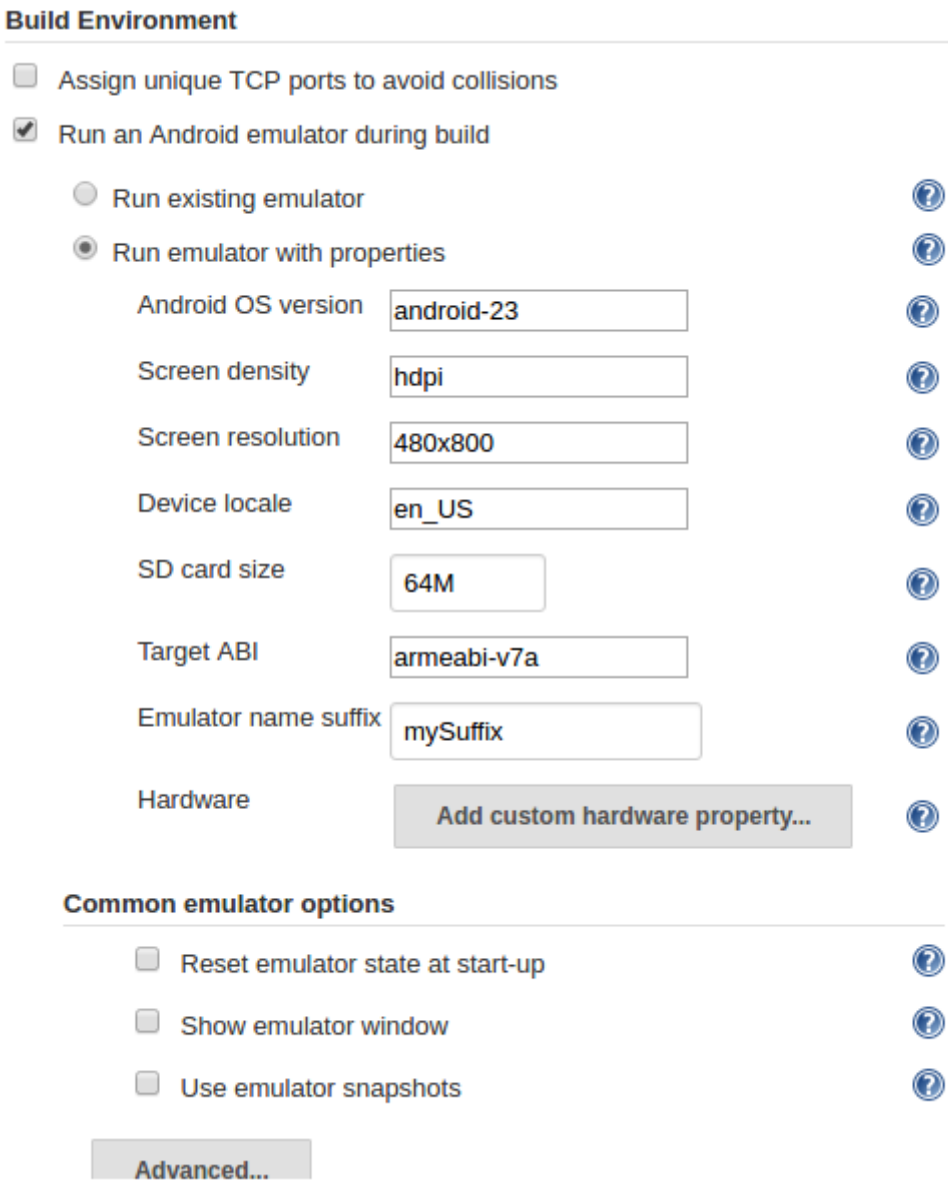


پس از طی نمودن مراحل فوق بایستی بتوانید Android build خود را اجرا نمایید.

5-5-8-اجرای تست بر روی دستگاه

جهت اجرای تست های instrumentation بر روی دستگاه دلخواه خود، باید یک دستگاه مجازی و شبیه ساز محیط (virtual device) اندروید راه اندازی نمایید.

دقت کنید که گزینه ی Show emulator window نبایستی انتخاب شده باشد چرا که عملیات زمان بندی شده یا job شما نباید به حضور یک برنامه ی display server وابسته باشد.



جهت اجرای تست و تست های instrumentation، بایستی از test connectedAndroidTest target استفاده کنید.

6-5-8-دیگر افزونه های مفید برای build پروژه های اندرویدی

- Android Emulator Plugin – از یک job دیگر با گزینه ی Build multi-configuration project پشتیبانی می کند. این گزینه به شما امکان می دهد تا همزمان یک پروژه را بر

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

روی چندین نسخه ی تنظیم شده از شبیه ساز تست نمایید. به طور مثال می توانید پروژه را با زبان ها، چگالی پیکسلی و وضوح تصویر مختلف تست نمایید.

معمولا دو job وجود دارد. یک job وظیفه ی build ساده یا کامپایل و اجرای تست بر روی پروژه را بر عهده دارد. دیگری یک job برای تست پروژه بر روی شبیه سازهایی با تنظیمات و پیکربندی متفاوت است.

افزونه ی پر کاربرد دیگری که بد نیست به آن اشاره شود، افزونه ی Lint است که به شما امکان می دهد تا ساختارهای مشکوک به داشتن خطا در کد پروژه را به واسطه ی Jenkins پیدا کرده و برطرف نمایید.

برای کسب اطلاعات بیشتر درباره ی این افزونه می توانید به آدرس https://wiki.jenkins-ci.org/display/JENKINS/Android+Lint+Plugin مراجعه نمایید.

بخش سوم :

استفاده از ابزار Android Debug Bridge (پل برقراری ارتباط و کنترل دستگاه یا شبیه ساز محیط اندرویدی و خطازدایی کد)

آموزش حاضر شرح می دهد چگونه با استفاده از ابزار ADB به دستگاه واقعی یا شبیه ساز محیط اندروید (AVD) متصل شده و آن را مدیریت کنید.

6-8-ابزار دسترسی، مدیریت و اشکال زدایی پروژه ی اندرویدی / ADB

Android Debug Bridge یا به اختصار ADB یک ابزار خط دستور جهت ارتباط و کنترل شبیه ساز (Emulator) یا دستگاه اندرویدی متصل به سیستم است و دستورات کاربردی متعددی نظیر نصب برنامه بر روی محیط شبیه ساز ، دسترسی به فایل های مستقر بر روی شبیه ساز و ... را برای برنامه نویس فراهم می کند.

می توانید با درج دستور adb در خط فرمان، به دستگاه اندرویدی خود دسترسی پیدا کنید.
دسترسی و اتصال به یک دستگاه مجازی (AVD) به راحتی امکان پذیر می باشد، اما جهت دسترسی کامل به دستگاه واقعی اندروید لازم است که آن دستگاه root شود. هر چند که دستگاه های root نشده نیز با adb قابل دسترسی هستند، اما در آن صورت توسعه دهنده صرفا در سطح یک کاربر عادی اجازه دسترسی خواهد داشت.

ابزار adb در پوشه ی [android-sdks]/platform-tools موجود می باشد. به منظور دسترسی کامل به این ابزار، بایستی آن را به متغیر path اضافه نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

adb به شما امکان می دهد تا دستوراتی را به دستگاه اندرویدی خود ارسال کرده، فایل هایی را در آن قرار داده یا از آن واکشی نمایید. همچنین امکان دسترسی از خط فرمان به سرور راه دور (shell access) و دستگاه اندروید را فراهم آورده و به توسعه دهنده اجازه می دهد تا اطلاعاتی نظیر میزان مصرف حافظه دستگاه را مشاهده نماید. مبحث زیر نحوه ی استفاده از دستورات مربوطه را برای شما شرح می دهد.

در صورتی که چندین دستگاه اندرویدی فعال و در حال اجرا داشته باشید، می توانید دستوراتی را به دستگاه ارسال کنید.

```
# Lists all devices
adb devices
#Result
List of devices attached
emulator-5554 attached
emulator-5555 attached
# Issue a command to a specific device
adb -s emulator-5554 shell
```

8-7-استفاده از adb

8-7-1-اجرای یک activity از خط دستور (command line)

کپی فایل ها

می توانید با درج دستور زیر در خط فرمان، یک فایل را از دستگاه کپی کرده یا در آن جایگذاری کنید.

```
adb shell am start -n yourpackagename/.activityname
```

8-8-حذف اپلیکیشن از دستگاه به وسیله ی دستورات adb

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

می توانید یک اپلیکیشن اندرویدی را با دستورات خط فرمان (از طریق shell) از دستگاه میزبان حذف نمایید. کافی است پوشه ی data/app (cd /data/app) را تغییر داده و سپس به راحتی اپلیکیشن اندرویدی را حذف نمایید.

در صورت تمایل می توانید یک اپلیکیشن را به وسیله ی دستورات adb، با ذکر اسم پکیج از دستگاه حذف نمایید.

```
adb uninstall <packagename>
```

8-9-اتصال به دستگاه با استفاده از Telnet

در صورت تمایل می توانید بجای استفاده از adb، به واسطه ی telnet به دستگاه اندرویدی دلخواه وصل شوید. این به شما امکان می دهد تا برخی از بسترها همچون تماس دریافتی را شبیه سازی کنید، وضعیت اتصال به شبکه را ویرایش نموده و وضعیت جغرافیایی خود را تنظیم کنید. به منظور اتصال به دستگاه شبیه سازی شده کافی است از "telnet localhost 5554" استفاده کنید. جهت خروج از فضای کاری console نیز کافی است دستور quit یا exit را اجرا نمایید.

به طور مثال، جهت تغییر تنظیمات گوشی، یا به منظور دریافت sms و تماس دریافتی، می توانید از دستورات زیر استفاده کنید.

```
# connects to device
telnet localhost 5554
# set the power level
power status full
power status charging
# make a call to the device
gsm call 012041293123
# send a sms to the device
sms send 12345 Will be home soon
# set the geo location
geo fix 48 51
```

8-10-دریافت اطلاعات سیستمی با ابزار خط دستور dumpsys

8-10-1-دستور adb dumpsys

دستور فوق به شما اجازه می دهد تا اطلاعات مربوط به سیستم اندروید و اپلیکیشن های در حال اجرا ضبط کرده و آن ها را رصد نمایید.

جهت بازیابی اطلاعات مربوطه به میزان مصرف حافظه، می توانید دستور زیر را فراخوانی کنید.

```
adb shell dumpsys meminfo <package.name>
```

8-10-2-مروری بر میزان مصرف حافظه با دستور dumpsys

دستور adb shell procrank تمامی اپلیکیشن های نصب شده بر روی دستگاه را بر اساس میزان مصرف حافظه لیست می کند. لازم به ذکر است که دستور نام برده بر روی دستگاه واقعی اندروید کار نمی کند. جهت واکشی لیست اپلیکیشن ها بر اساس میزان مصرف حافظه در دستگاه واقعی، بایستی دستور adb shell dumpsys meminfo را فراخوانی کنید.

8-10-3-واکشی اطلاعات درباره ی عملیات و تسک های زمان بندی شده

جهت کسب اطلاعات در خصوص اپلیکیشن، کافی است دستور adb shell dumpsys alarm را همراه با اسم پکیج پروژه در خط فرمان فراخوانی کنید. خروجی ممکن است ظاهری مشابه زیر داشته باشد:

```
RTC #6: Alarm{434a1234 type 1 com.example}
  type=1 whenElapsed=608198149 when=+12m13s122ms window=-1 repeatInterval=0 count=0
  operation=PendingIntent{430cf612: PendingIntentRecord{*43bbf887* com.vogella startService}}
```

همان طور که مشاهده می کنید، هشدار برای حدودا 12 دقیقه تنظیم شده است.

جهت کسب اطلاعات لازم در خصوص pending intent (معلق)، دستور adb shell dumpsys activity intents را اجرا کرده و ID مربوط به PendingIntentRecord (در این مثال 43bbf887) را به عنوان پارامتر ارسال کنید (PendingIntent تلفیقی از درخواست عملیات (راه اندازی یک activity دیگر، اجرا و فراخوانی یک سرویس یا ارسال یک broadcast، جزییات عملیات (در قالب intent و context) است. نمونه ایجاد شده می تواند به سایر اپلیکیشن ها ارسال گردد تا با کمک آنها حتی پس از خاتمه یافتن اپلیکیشن تولید کننده ی pendingintent و حذف

آن از حافظه، عملیاتی معینی انجام گیرد. همچنین آبجکت PendingIntent با برخورداری از flag هایی که عملکرد آنرا با توجه به اینکه نمونه ای از آن کلاس pendingintent موجود است یا نه، مدیریت می نماید):

```
  * PendingIntentRecord{43bbf887 com.vogella startService}
uid=10042 packageName=com.vogella type=startService flags=0x0
requestIntent=act=MY_ACTION cmp=com.vogella/.MyService (has extras)
```

تسک هایی که اطلاعات مربوط به مصرف باتری را واکشی می کنند

از ویرایش 5.0 اندروید، شما می توانید با فراخوانی دستور زیر در خط فرمان، اطلاعات مربوط به میزان مصرف باتری را واکشی نمایید:

```
adb shell dumpsys batterystats --charged <package-name>
```

فصل نهم
مکان یابی اندروید و آشنایی با Google Map

بخش اول :

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

توابع کتابخانه ای مکان یابی / Location API اندروید و ابزار fused location provider

آموزش حاضر نحوه ی استفاده از ابزار fused location provider را به شما آموزش می دهد.

توابع کتابخانه ای مکان یابی اندروید / Android Location API

بدست آوردن اطلاعات مربوط به مکان جغرافیایی دستگاه

امروزه اغلب دستگاه های مبتنی بر سیستم عامل اندروید این امکان را برای کاربران فراهم می کنند تا مکان جغرافیایی جاری دستگاه را بدست بیاورند. این امر از طریق ماژول GPS، wifi یا تکنیک مثلث سازی برج های مخابراتی امکان پذیر می باشد.

Google Play با استفاده از fused location provider آخرین مکان جغرافیایی دستگاه جاری را بازیابی می کند.

نصب

به منظور استفاده از location manager، سرویس Google play را در قالب dependency به فایل build.gradle اضافه نمایید.

```
dependencies {
    compile 'com.google.android.gms:play-services:9.2.0'
    compile 'com.google.android.gms:play-services-location:9.2.0'
}
```

همچنین بایستی مجوز مورد نیاز را در فایل تنظمیات اپلیکیشن اندرویدی، manifest، اعلان نمایید.

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
```

استفاده از LocationManager

اکنون می توانید به آخرین مکان جغرافیایی دستگاه دسترسی داشته باشید. fuse location provider توابع کتابخانه ای نوین و کارایی در اختیار توسعه دهنده قرار می دهد که کار مکان

یابی را انجام می دهد. در زیر مثالی را مشاهده می کنید که از این توابع برای واکشی اطلاعات مکان جاری دستگاه استفاده می کند.

بدست آوردن مختصات جغرافیایی به صورت دو طرفه (forward and reverse geocoding)

کلاس Geocoder به شما این امکان را می دهد تا مختصات جغرافیایی (عرض و طول جغرافیایی) یک مکان را بر اساس آدرس ارائه شده بدست آورده و بالعکس (بر اساس عرض و طول جغرافیایی ارائه شده، آدرس دقیق مثل اسم خیابان محل قرارگیری دستگاه را بدست آورید). از این پروسه تحت عنوان reverse geocoding یاد می شود. کلاس نام برده از یک سرویس آنلاین Google استفاده می کند.

امنیت

اگر می خواهید به حسگر GPS دسترسی داشته باشید، در آن صورت لازم است مجوز ACCESS_FINE_LOCATION را نیز اعلان نمایید. در غیر این صورت باید ACCESS_COARSE_LOCATION را در فایل تنظیمات اعلان کنید.

درخواست از کاربر برای فعال سازی حسگر GPS (امکان سخت افزاری GPS یا مکان یاب)

کاربر باید تصمیم بگیرد که امکان سخت افزاری GPS فعال شود یا خیر.

می توانید با فراخوانی متد isProviderEnabled() مطمئن شوید آیا LocationManager فعال و پیاده سازی شده است یا خیر (LocationManager کلاسی است که امکان دسترسی به سرویس های مکان یابی سیستم را فراهم می کند). در صورتی که فعال نشده باشد، می توانید به واسطه ی ارسال یک آبجکت Intent که مقدار خاصیت name از المان action آن (اسم action) در فایل xml، بر روی Settings.ACTION_LOCATION_SOURCE_SETTINGS تنظیم شده به کلاس android.provider.Settings، کاربر را به بخش تنظیمات مربوطه ارجاع دهید.

```
LocationManager service = (LocationManager) getSystemService(LOCATION_SERVICE);
boolean enabled = service
    .isProviderEnabled(LocationManager.GPS_PROVIDER);
```

```
// check if enabled and if not send user to the GSP settings
// Better solution would be to display a dialog and suggesting to
// go to the settings
if (!enabled) {
  Intent intent = new Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
      startActivity(intent);
}
```

معمولا به این صورت است که توسعه دهنده یک AlarmDialog در نمایشگر باز می کند که از کاربر می پرسد آیا می خواهد امکان سخت افزاری GPS را فعال کند یا اینکه اجرای اپلیکیشن مربوطه باید کلا لغو گردد.

لازم به ذکر است که شما نمی توانید به طور مستقیم امکان GPS را در کد برنامه فعال کنید بلکه کاربر باید آن را به صورت دستی فعال کند.

استفاده از امکان GPS و تنظیم موقعیت جاری

فعال سازی امکان GPS در محیط شبیه ساز

لازم است امکان GPS را بر روی دستگاه تست گیری خود فعال نمایید. اگر اپلیکیشن را بر روی محیط شبیه ساز تست نمایید، در حالی که امکان GPS فعال نشده باشد، کلاس LocationManager مقدار null را برمی گرداند.

activity یا صفحه ی Google Map، بایستی به صورت خودکار امکان سخت افزاری GPS را در شبیه ساز فعال کند. اما چنانچه می خواهید از location manager به طور مسقیم استفاده نمایید، بایستی این کار را خود انجام دهید. در حال حاضر مشکلاتی برای استفاده ی مستقیم از location manager وجود دارد.

Google Maps را در محیط شبیه ساز اجرا کرده و درخواست اطلاعات موقعیت جغرافیایی جاری را بدهید. این کار به شما اجازه می دهد تا GPS را فعال نمایید. مختصات جدید GPS را به شبیه ساز اندروید ارسال نمایید.

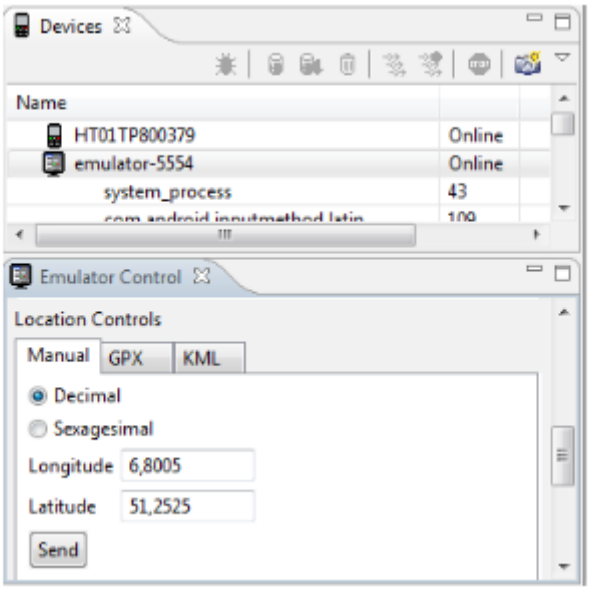
تنظیم موقعیت جغرافیایی

می توانید با استفاده از "DDMS" Perspective محیط کاری Eclipse، موقعیت جغرافیایی خود را به شبیه ساز یا دستگاه متصل ارسال نمایید. برای این منظور، ابتدا Perspective را باز کرده و سپس مسیر رو به رو را طی نمایید: Window ▸ Open Perspective ▸ Other... ▸ DDMS.

(perspective اسمی که به مجموعه ای از view ها و ناحیه ی ویرایشگر اشاره محیط کاری Eclipse اشاره دارد)

(DDMS که سرنام واژگان Dalvik debug monitor service است کاراترین ابزار اشکال زدایی برای پروژه های اندرویدی بوده و از قابلیت های آن می توان به گزارش گیری و نمایش اطلاعات file manager اشاره کرد. این ابزار قادر است به عنوان یک برنامه ی جدا راه اندازی شده و بر دستگاه میزبان نظارت داشته باشد.)

می توانید در Emulator Control، مختصات جغرافیایی را وارد کرده و دکمه ی Send را فشار دهید.



همچنین می توانید مختصات جغرافیایی را در شبیه ساز Android از طریق telnet تنظیم نمایید. پنجره ی فرمان یا کنسول را باز کرده و به دستگاه دلخواه متصل شوید. شماره ی port دستگاه شما در نوار عنوان (title area) شبیه ساز قابل مشاهده می باشد.

```
telnet localhost 5554
```

حال از طریق دستور زیر موقعیت جغرافیایی را مقداردهی و تنظیم نمایید.

```
geo fix 13.24 52.31
```

آموزش: استفاده از توابع کتابخانه ای Location API اندروید

ساخت پروژه

یک پروژه ی و activity جدید به ترتیب به نام de.vogella.android.locationapi.simple و ShowLocationActivity ایجاد نمایید.

گفتنی است که این مثال از Google Map استفاده نمی کند و بجای محیط شبیه ساز بر روی دستگاه واقعی اندروید اجرا می شود.

فایل layout خود را از <filename class="directory">res/layout_ به کد زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
  <LinearLayout
    android:id="@+id/linearLayout1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="40dip"
    android:orientation="horizontal" >
    <TextView
      android:id="@+id/TextView01"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginLeft="10dip"
      android:layout_marginRight="5dip"
      android:text="Latitude: "
      android:textSize="20dip" >
    </TextView>
    <TextView
      android:id="@+id/TextView02"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
```

```
        android:text="unknown"
        android:textSize="20dip" >
    </TextView>
  </LinearLayout>
  <LinearLayout
    android:id="@+id/linearLayout2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
    <TextView
      android:id="@+id/TextView03"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginLeft="10dip"
      android:layout_marginRight="5dip"
      android:text="Longitute: "
      android:textSize="20dip" >
    </TextView>
    <TextView
      android:id="@+id/TextView04"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="unknown"
      android:textSize="20dip" >
    </TextView>
  </LinearLayout>
</LinearLayout>
```

افزودن مجوزهای لازم

مجوزهای زیر را در فایل تنظیمات اپلیکیشن (AndroidManifest.xml) اعلان نمایید.

- INTERNET
- ACCESS_FINE_LOCATION
- ACCESS_COARSE_LOCATION

ویرایش کلاس Activity

محتوای کلاس ShowLocationActivity را به صورت زیر ویرایش نمایید. با اعمال تغییرات زیر از location manger کوئری گرفته شده و مقادیر مورد درخواست (خروجی) در activity به نمایش گذاشته می شوند.

```
package de.vogella.android.locationsapi.simple;
import android.app.Activity;
import android.content.Context;
import android.location.Criteria;
import android.location.Location;
```

```
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.TextView;
import android.widget.Toast;
public class ShowLocationActivity extends Activity implements LocationListener {
	private TextView latituteField;
	private TextView longitudeField;
	private LocationManager locationManager;
	private String provider;
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
		latituteField = (TextView) findViewById(R.id.TextView02);
		longitudeField = (TextView) findViewById(R.id.TextView04);
		// Get the location manager
		locationManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
		// Define the criteria how to select the locatioin provider -> use
		// default
		Criteria criteria = new Criteria();
		provider = locationManager.getBestProvider(criteria, false);
		Location location = locationManager.getLastKnownLocation(provider);
		// Initialize the location fields
		if (location != null) {
			System.out.println("Provider " + provider + " has been selected.");
			onLocationChanged(location);
		} else {
			latituteField.setText("Location not available");
			longitudeField.setText("Location not available");
		}
	}
	/* Request updates at startup */
	@Override
	protected void onResume() {
		super.onResume();
		locationManager.requestLocationUpdates(provider, 400, 1, this);
	}
	/* Remove the locationlistener updates when Activity is paused */
	@Override
	protected void onPause() {
		super.onPause();
		locationManager.removeUpdates(this);
	}
	@Override
	public void onLocationChanged(Location location) {
		int lat = (int) (location.getLatitude());
		int lng = (int) (location.getLongitude());
		latituteField.setText(String.valueOf(lat));
		longitudeField.setText(String.valueOf(lng));
	}
```

```
    @Override
    public void onStatusChanged(String provider, int status, Bundle extras) {
        // TODO Auto-generated method stub
    }
    @Override
    public void onProviderEnabled(String provider) {
        Toast.makeText(this, "Enabled new provider " + provider,
                Toast.LENGTH_SHORT).show();
    }
    @Override
    public void onProviderDisabled(String provider) {
        Toast.makeText(this, "Disabled provider " + provider,
                Toast.LENGTH_SHORT).show();
    }
}
```

اجرا و تست

اگر از شبیه ساز استفاده می کنید، مختصات جغرافیایی مورد نظر را به دستگاه ارسال کنید. با کلیک بر روی دکمه، مختصات جغرافیایی به نمایش گذاشته می شود.

بخش دوم :

توابع کتابخانه ای Google Maps / Google Maps v2 Android API v2

آموزش حاضر به شرح توابع کتابخانه ای Google Maps و نحوه ی استفاده از آن ها در پروژه های اندرویدی می پردازد. پروژه این مبحث در ویرایش 4.4 محیط کاری Eclipse، بر پایه ی ورژن 1.7 زبان جاوا و 4.4 اندروید نوشته شده اند.

Google Maps

MapsView

شرکت گوگل یک کتابخانه از طریق سرویس Google play در اختیار توسعه دهنده قرار می دهد که امکان استفاده ی آسان از Google Maps و پیاده سازی توابع آن در پروژه ی اندرویدی را به راحتی فراهم می آورد. مبحث حاضر از توابع کتابخانه ای Google Maps v2 استفاده می کند که نسبت به ورژن قبلی آن از قابلیت های بیشتری برخوردار است.

کتابخانه ی مزبور دو کلاس به ترتیب به نام های com.google.android.gms.maps.MapFragment و MapView ارائه می دهد که کامپوننت نقشه را برای کاربر نمایش می دهد.

برای استفاده از Google Maps لازم است اطلاعات اضافی به فایل تنظیمات اپلیکیشن AndroidManifest.xml اضافه نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.vogella.android.maps">
<!--
The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
Google Maps Android API v2, but you must specify either coarse or fine
location permissions for the 'MyLocation' functionality.
-->
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<application
android:allowBackup="true"
android:icon="@mipmap/ic_launcher"
android:label="@string/app_name"
android:supportsRtl="true"
android:theme="@style/AppTheme">
<!--
The API key for Google Maps-based APIs is defined as a string resource.
(See the file "res/values/google_maps_api.xml").
Note that the API key is linked to the encryption key used to sign the APK.
You need a different API key for each encryption key, including the release key that is used to
sign the APK for publishing.
You can define the keys for the debug and release targets in src/debug/ and src/release/.
-->
<meta-data
android:name="com.google.android.geo.API_KEY"
android:value="@string/google_maps_key" />
<activity
android:name=".MapsActivity"
android:label="@string/title_activity_maps">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

کلاس MapFragment

کلاس MapFragment از کلاس Fragment ارث بری کرده و توابعی جهت مدیریت چرخه ی حیات (life cycle) و سرویس های لازم برای نمایش ویجت (کامپوننت رابط کاربری) GoogleMap را فراهم می آورد. GoogleMap کلاسی است که وظیفه ی نمایش نقشه را بر عهده دارد. MapFragment برای دسترسی به این کلاس متد ()getMap را فراخوانی می کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

کلاس LatLng پل ارتباطی جهت تعامل با کلاس GoogleView خواهد بود.

درج marker (نشانه) بر روی نقشه

می توانید با استفاده از کلاس Marker، بر روی نقشه نشانه درج نمایید. این قابلیت بسیار منعطف بوده و قابل تنظیم می باشد.

کد زیر مثالی عملی را به نمایش می گذارد.

```
public class MainActivity extends Activity {
    static final LatLng HAMBURG = new LatLng(53.558, 9.927);
    static final LatLng KIEL = new LatLng(53.551, 9.993);
    private GoogleMap map;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        map = ((MapFragment) getFragmentManager().findFragmentById(R.id.map))
                .getMap();
        if (map!=null){
            Marker hamburg = map.addMarker(new MarkerOptions().position(HAMBURG)
                    .title("Hamburg"));
            Marker kiel = map.addMarker(new MarkerOptions()
                    .position(KIEL)
                    .title("Kiel")
                    .snippet("Kiel is cool")
                    .icon(BitmapDescriptorFactory
                            .fromResource(R.drawable.ic_launcher)));
        }
    }
```

می توانید بر روی نمونه ی ساخته شده از روی کلاس GoogleMap، توابع listener برای گوش دادن به marker های درج شده بر روی نقشه الصاق نمایید. کلاس OnMarkerClickListener یک متد به نام onMarkerClicked(Marker) در اختیار توسعه دهنده قرار می دهد که به مجرد کلیک کاربر بر روی marker فراخوانی می شود.

تنظیم اختصاصی GoogleMap

می توان GoogleMap را مطابق نیاز به صورت اختصاصی تنظیم کرد.

کد زیر از وبسایت اصلی گوگل اقتباس شده است.

```
static final LatLng HAMBURG = new LatLng(53.558, 9.927);
static final LatLng KIEL = new LatLng(53.551, 9.993);
private GoogleMap map;
... // Obtain the map from a MapFragment or MapView.
//Move the camera instantly to hamburg with a zoom of 15.
map.moveCamera(CameraUpdateFactory.newLatLngZoom(HAMBURG, 15));
// Zoom in, animating the camera.
map.animateCamera(CameraUpdateFactory.zoomTo(10), 2000, null);
```

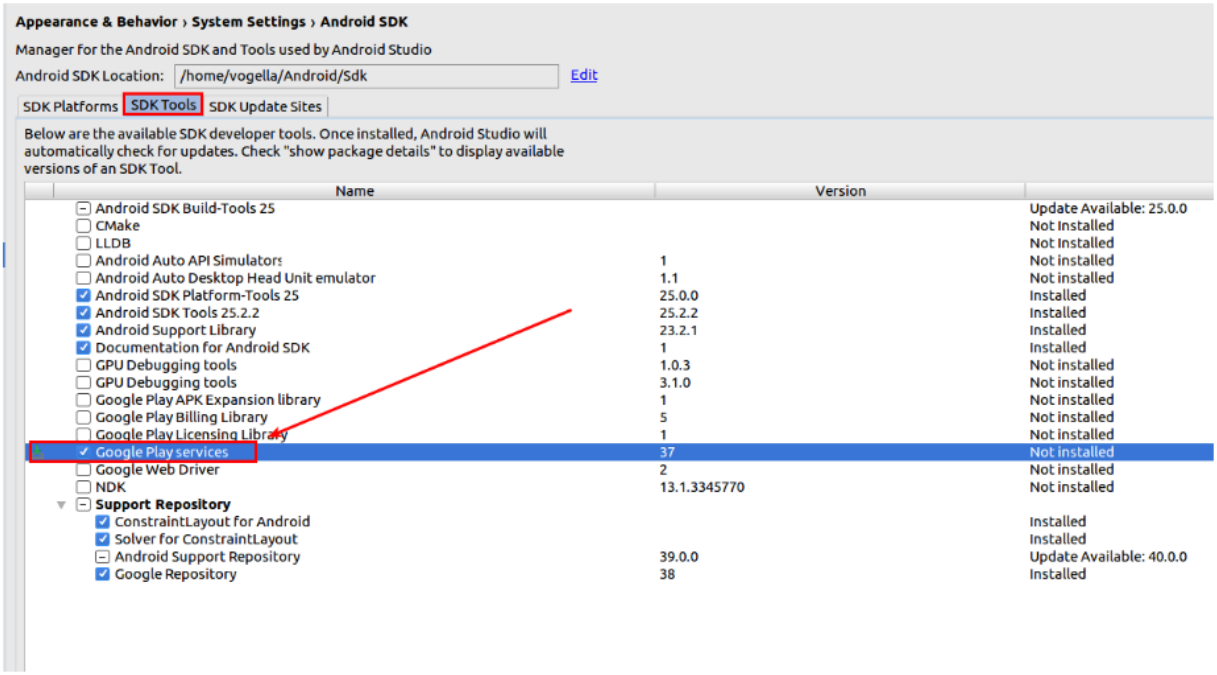
نرم افزار شبیه ساز محیط اندروید (emulator) و Google Maps

لازم است یک شبیه ساز مبتنی بر توابع کتابخانه ای Google ایجاد نمایید. با استفاده از این شبیه ساز می توانید بعده ها Google map و دیگر سرویس های ادغام شدنی نظیر Google Play Service را تست نمایید.

نصب Google Play service

دانلود Google Play Service

Android SDK Manager را باز کرده و Google Play Services را نصب نمایید.



اکنون می توانید dependency مورد نیاز را به فایل app/build.gradle اضافه نمایید.

```
compile 'com.google.android.gms:play-services:9.8.0'
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

ساخت و دریافت کلید شناسگر Google Map (key)

کنسول Google

جهت استفاده از Google Maps لازم است یک شناسه گر برنامه ی فراخواننده یا API key معتبر Google ایجاد نمایید. این کلید رایگان بوده و می توان از آن برای تمامی اپلیکیشن هایی که توابع کتابخانه ای Maps را فراخوانی می کنند، استفاده نمایید. در خصوص استفاده از این کلید محدودیتی وجود نداشته و قابلیت پشتیبانی از بی نهایت کاربر را دارد.

برای دریافت این کلید بایستی از طریق Google APIs Console اقدام نمایید. جهت دریافت کلید لازم است اسم پکیج و امضای دیجیتالی (signature key) اپلیکیشن را ارائه نمایید.

منظور از امضای دیجیتالی همان کلیدی است که اپلیکیشن اندروید خود را به هنگام خروجی گرفتن، توزیع و نصب (مرحله ی deployment) با آن امضا و به طور منحصر بفرد نشانه گذاری می کنید. در حین توسعه ی اپلیکیشن، سیستم کامپایل و اجرای پروژه های اندرویدی (build system)، به صورت خودکار یک debug key تولید کرده و از آن جهت اشکال زدایی اپلیکیشن و اینکه کاربر مجبور نباشد هر بار برای ترجمه و اجرای برنامه امضای الکترونیکی را وارد کند، استفاده می نماید.

ایجاد SHA-1 برای امضای دیجیتالی (Signature key)

debug key یا کلید اشکال زدایی و کامپایل پروژه تحت آدرس

userhome/.android/debug.keystore قابل دسترسی می باشد.

SHA-1 (الگوریتم درهم سازی ایمن): تابع درهم سازی در مبحث رمزنگاری است.

برای ساخت SHA-1 ویژه ی debug keystore (keystore برای اعتبارسنجی شناسه ی برنامه نویس و توسعه دهنده ی اپلیکیشن استفاده می شود.)، می توانید دستور keytool را از JDK نصب شده اجرا کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

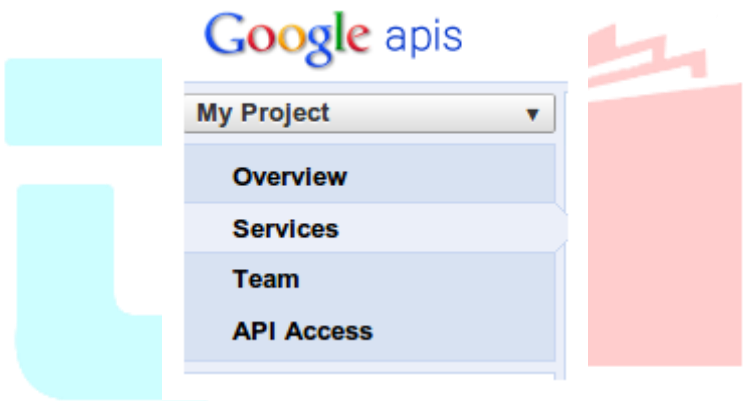
88146323 - 88446780 - 88146330

```
keytool -list -v -alias androiddebugkey \
-keystore <path_to_debug_keystore>debug.keystore \
-storepass android -keypass android
```

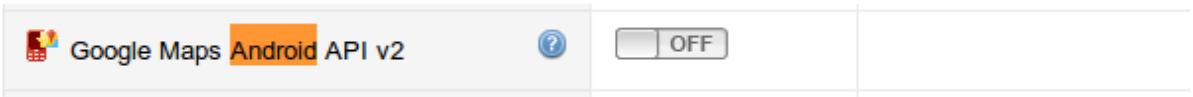
لازم است خروجی SHA-1 را برای استفاده در آینده کپی نمایید.

ثبت نام در Google APIs Console

شما بایستی در Google APIs Console ثبت و اعلان نمایید که می خواهید از توابع Google Maps برای پروژه ی اندرویدی خود استفاده نمایید. از منوی My Project المان Services را انتخاب نمایید.



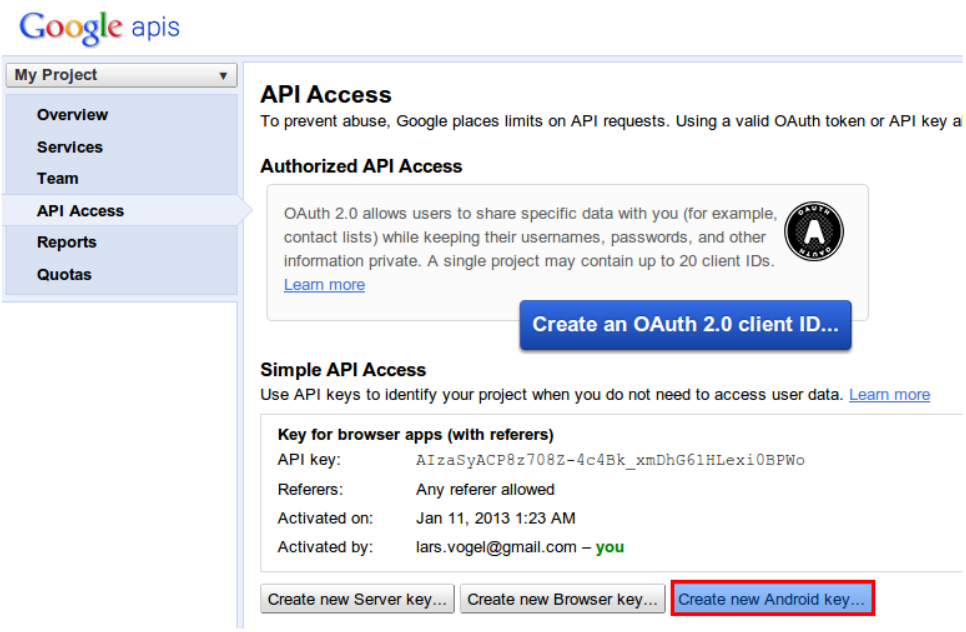
Google Maps Android API v2 را فعال نمایید.



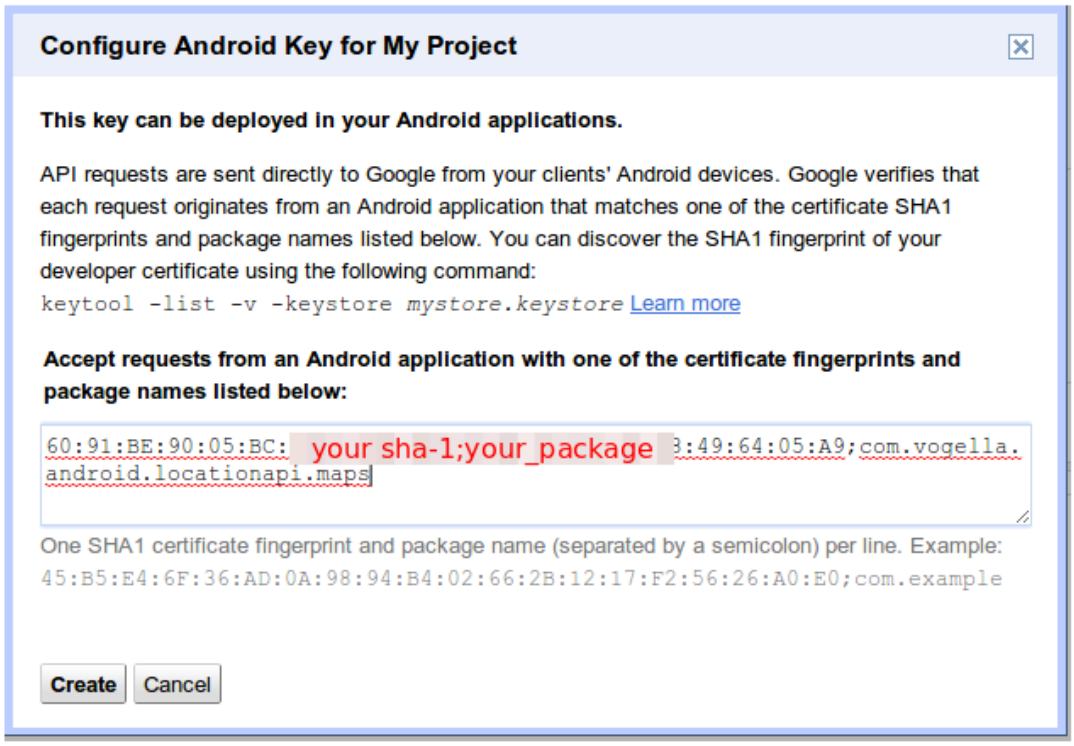
ساخت کلید و امضای دیجیتالی برای اپلیکیشن

شما بایستی بعده ها اپلیکیشن را از طریق اسم پکیج در کنسول همراه با SHA-1 fingerprint امضای دیجیتالی (signature key) ثبت نمایید (fingerprint: دنباله‌ای کوتاه از بایت‌ها می‌باشد که برای تایید اعتبار و یا جستجوی کلید عمومی/public key طولانی تر استفاده می‌شود. اثرانگشت‌ها با اعمال یک تابع درهم‌ساز رمزنگارانه/hash function بر روی یک کلید عمومی ایجاد می‌گردند. از آنجا که اثر انگشت‌ها نسبت به کلیدهای مربوطه کوتاه تر هستند، می‌توان از آن‌ها به منظور ساده سازی برخی وظایف مدیریت کلید استفاده نمود . رمزنگاری کلید عمومی

یا رمزنگاری نامتقارن روشی از رمزنگاری است که کلید مورد استفاده برای رمزگذاری با کلید مربوط برای رمزگشایی با هم متفاوت است.). برای این منظور پس از کلیک بر روی المان مربوطه، آیتم API Access را انتخاب نمایید. سپس بر روی المان ...Create new Android key کلیک کنید.



حال در کادر مربوطه SHA-1 fingerprint و اسم پکیج اپلیکیشن خود را وارد کرده و آن ها را با ویرگول از هم جدا نمایید. برای مثال همان طور که در تصویر زیر مشاهده می کنید، پکیج com.vogella.android.locationapi.maps در کادر مربوطه درج شده است.



آموزش: استفاده از Google Maps در پروژه

در این آموزش یک اپلیکیشن اندرویدی می نویسید که GoogleMap را از طریق یک fragment برای کاربر نمایش می دهد.

نصب Google Play Service

لازم است Google Play Service را نصب نمایید.

ایجاد پروژه ی اندرویدی

یک پروژه ی اندرویدی به نام com.vogella.android.maps بر اساس قالب آماده ی Google Maps Activity (template) ایجاد نمایید.

بررسی فایل تنظیمات پروژه

محتوای فایل manifest را چک کرده و مطمئن شوید که مجوزهای زیر توسط template نام برده در آن اعلان شده اند.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.vogella.android.maps">
  <!--
     The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
     Google Maps Android API v2, but you must specify either coarse or fine
     location permissions for the 'MyLocation' functionality.
  -->
  <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
  <application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <!--
       The API key for Google Maps-based APIs is defined as a string resource.
       (See the file "res/values/google_maps_api.xml").
       Note that the API key is linked to the encryption key used to sign the APK.
       You need a different API key for each encryption key, including the release key that is used to
       sign the APK for publishing.
       You can define the keys for the debug and release targets in src/debug/ and src/release/.
    -->
    <meta-data
      android:name="com.google.android.geo.API_KEY"
      android:value="@string/google_maps_key" />
    <activity
      android:name=".MapsActivity"
      android:label="@string/title_activity_maps">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>
</manifest>
```

template مورد نظر یک فایل به نام google_maps_api.xml ایجاد کرده است. این فایل حاوی دستور العمل هایی در خصوص نحوه ی ایجاد Google Maps API key می باشد.

پس از دریافت API key اپلیکیشن، این کلید را در فایل google_maps_api.xml وارد نمایید.

تنظیم فایل layout

در این مثال از کلاس MapFragment استفاده می کنیم. محتوای فایل layout خود را به صورت زیر ویرایش نمایید.

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
   android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.vogella.android.maps.MapsActivity"
    />
```

ویرایش Activity

پیاده سازی کلاس activity خود را به صورت زیر ویرایش نمایید.

```
package com.vogella.android.maps;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.MapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
public class MainActivity extends Activity {
      static final LatLng HAMBURG = new LatLng(53.558, 9.927);
      static final LatLng KIEL = new LatLng(53.551, 9.993);
      private GoogleMap map;
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
            map = ((MapFragment) getFragmentManager().findFragmentById(R.id.map))
                        .getMap();
      }
      @Override
      public boolean onCreateOptionsMenu(Menu menu) {
            getMenuInflater().inflate(R.menu.activity_main, menu);
            return true;
      }
}
```

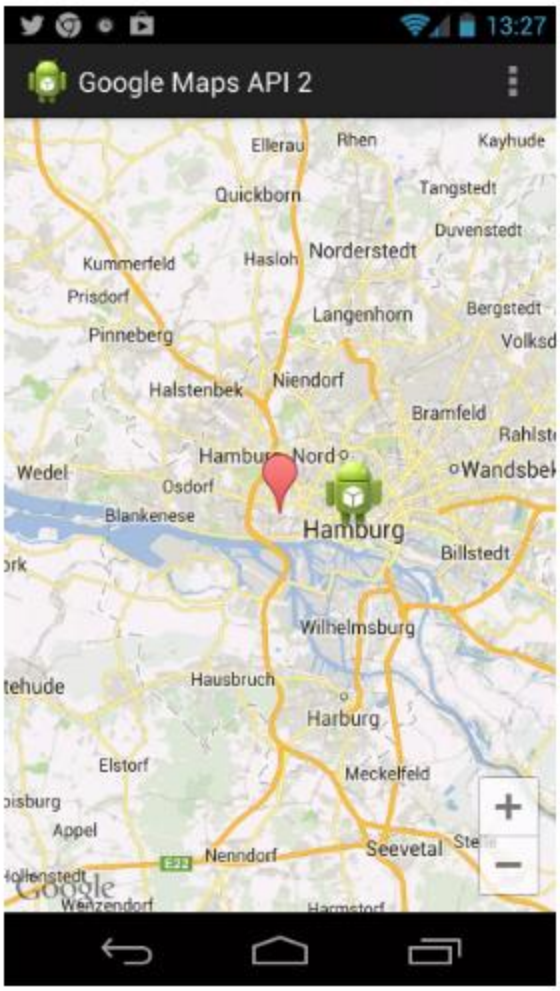
توجه: برای اجرای این مثال لازم است قابلیت پشتیبانی از multidex را فعال نمایید.

```
android {
```

```
    compileSdkVersion 25
    buildToolsVersion "24.0.0"
    defaultConfig {
        applicationId "com.vogella.android.maps"
        minSdkVersion 23
        targetSdkVersion 25
        versionCode 1
        versionName "1.0"
        multiDexEnabled true
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}
```

اجرا و تست اپلیکیشن

اپلیکیشن را اجرا کرده و تست نمایید. باتوجه به منطق برنامه باید بتوانید بر روی Map حرکت کرده و قسمت هایی از آن بزرگ نمایی/کوچک نمایی کنید.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فصل دهم

صفحه لمسی و سنسورهای اندروید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش اول :

توابع کتابخانه ای اندروید مربوط به امکان سخت افزاری دوربین / Camera API

این مبحث به شرح نحوه ی استفاده از امکان سخت افزاری Camera از طریق آبجکت Intent و توابع کتابخانه ای اندروید API می پردازد. پروژه ی این آموزش در ویرایش 4.2 محیط کاری Eclipse نوشته شده و مبتنی بر ورژن 1.6 جاوا و نسخه ی 4.2 سیستم عامل اندروید می باشد.

امکان سخت افزاری Camera

امروزه اغلب دستگاه های اندرویدی حداقل یک دوربین را دارند و برخی نیز یک دوربین در جلو و یک دوربین در عقب دستگاه دارند.

جهت استفاده از امکان سخت افزاری camera دستگاه اندرویدی، کافی است از اپلیکیشن آماده ی camera استفاده نمایید. بدین معنی که توسعه دهنده به واسطه ی یک آبجکت intent اپلیکیشن آماده ی Camera را اجرا کرده و سپس با خواندن داده های بازگشتی اپلیکیشن به نتیجه دست پیدا می کند.

روش دیگری که می توان مورد استفاده قرار داد، این است که camera یا امکان سخت افزاری دوربین را با فراخوانی توابع کتابخانه ای Camera API مستقیما در بطن پروژه جاسازی نمایید.

مبحث زیر مثال های کاربردی درباره ی پیاده سازی هر دو روش ارائه می دهد.

آموزش: اجرای اپلیکیشن Camera و گرفتن یک عکس با فراخوانی آبجکت Intent

یک پروژه و activity جدید به ترتیب به نام های de.vogella.android.imagepick و ImagePickActivity ایجاد نمایید.

یک المان ImageView که مقدار ID آن بر روی result تنظیم شده در لایه ی XML اپلیکیشن (فایل layout)، ایجاد نمایید. سپس یک المان button با ویژگی (property) onClick که به متد متناظر onClick در کلاس activity اشاره دارد، در این فایل اضافه نمایید.

حال کلاس ImagePickActivity را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.imagepick;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import android.app.Activity;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
public class ImagePickActivity extends Activity {
    private static final int REQUEST_CODE = 1;
    private Bitmap bitmap;
    private ImageView imageView;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        imageView = (ImageView) findViewById(R.id.result);
    }
    public void onClick(View View) {
        Intent intent = new Intent();
        intent.setType("image/*");
        intent.setAction(Intent.ACTION_GET_CONTENT);
        intent.addCategory(Intent.CATEGORY_OPENABLE);
        startActivityForResult(intent, REQUEST_CODE);
    }
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        InputStream stream = null;
        if (requestCode == REQUEST_CODE && resultCode == Activity.RESULT_OK)
            try {
                // recyle unused bitmaps
                if (bitmap != null) {
                    bitmap.recycle();
                }
                stream = getContentResolver().openInputStream(data.getData());
                bitmap = BitmapFactory.decodeStream(stream);
                imageView.setImageBitmap(bitmap);
            } catch (FileNotFoundException e) {
                e.printStackTrace();
```

```
            } finally ch (IOException e) {
                e{
                if (stream != null)
                    try {
                        stream.close();
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
            }
    }
}
```

آموزش: پیاده سازی توابع کتابخانه ای Camera API

در تمرین جاری، یک اپلیکیشن طراحی می کنید که به کاربر امکان می دهد با استفاده از دوربین جلوی دستگاه عکس تهیه کرده و بعد آن را بر روی حافظه ی خارجی (SD) ذخیره نماید.

توجه: در صورت استفاده از نرم افزار شبیه ساز محیط اندروید (Android emulator)، لازم است امکان سخت افزاری camera را تنظیم کرده و یک حافظه ی خارجی (SD card) به هنگام ایجاد دستگاه مجازی اندروید تعریف کرده باشید.

یک پروژه ی اندروید و activity جدید به ترتیب به نام های de.vogella.camera.api و MakePhotoActivity ایجاد نمایید.

لازم است مجوز android.permission.CAMERA را برای دسترسی به امکان سخت افزاری دوربین و android.permission.WRITE_EXTERNAL_STORAGE برای ذخیره ی اطلاعات بر روی حافظه ی خارجی به فایل AndroidManifest.xml اضافه نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="de.vogella.cameara.api"
  android:versionCode="1"
  android:versionName="1.0" >
  <uses-sdk android:minSdkVersion="15" />
  <uses-permission android:name="android.permission.CAMERA"/>
  <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
  <application
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name" >
    <activity
      android:name="de.vogella.camera.api.MakePhotoActivity"
      android:label="@string/app_name" >
```

```
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
  </application>
</manifest>
```

فایل main.xml مقیم در پوشه ی res/layout_ <filename class="directory"> را به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent" >
  <Button
    android:id="@+id/captureFront"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true"
    android:onClick="onClick"
    android:text="Make Photo" />
</RelativeLayout>
```

یک کلاس به نام PhotoHandler را به صورت زیر اعلان و پیاده سازی کنید که وظیفه ی آن جایگذاری فایل تصویری تهیه شده در حافظه ی خارجی (SD card) می باشد.

```
package de.vogella.camera.api;
import java.io.File;
import java.io.FileOutputStream;
import java.text.SimpleDateFormat;
import java.util.Date;
import android.content.Context;
import android.hardware.Camera;
import android.hardware.Camera.PictureCallback;
import android.os.Environment;
import android.util.Log;
import android.widget.Toast;
public class PhotoHandler implements PictureCallback {
    private final Context context;
    public PhotoHandler(Context context) {
        this.context = context;
    }
    @Override
    public void onPictureTaken(byte[] data, Camera camera) {
        File pictureFileDir = getDir();
        if (!pictureFileDir.exists() && !pictureFileDir.mkdirs()) {
            Log.d(MakePhotoActivity.DEBUG_TAG, "Can't create directory to save image.");
            Toast.makeText(context, "Can't create directory to save image.",
                    Toast.LENGTH_LONG).show();
```

```
            return;
        }
        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyymmddhhmmss");
        String date = dateFormat.format(new Date());
        String photoFile = "Picture_" + date + ".jpg";
        String filename = pictureFileDir.getPath() + File.separator + photoFile;
        File pictureFile = new File(filename);
        try {
            FileOutputStream fos = new FileOutputStream(pictureFile);
            fos.write(data);
            fos.close();
            Toast.makeText(context, "New Image saved:" + photoFile,
                    Toast.LENGTH_LONG).show();
        } catch (Exception error) {
            Log.d(MakePhotoActivity.DEBUG_TAG, "File" + filename + "not saved: "
                    + error.getMessage());
            Toast.makeText(context, "Image could not be saved.",
                    Toast.LENGTH_LONG).show();
        }
    }
    private File getDir() {
        File sdDir = Environment
          .getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES);
        return new File(sdDir, "CameraAPIDemo");
    }
}
```

محتوای کلاس MakePhotoActivity را به صورت زیر ویرایش نمایید. این کلاس فرض را بر این می گذارد که شما یک المان button متناظر در لایه ی xml، فایل layout خود تعریف کرده اید که ویژگی onClick آن به متد ()onClick در کلاس activity اشاره می کند.

```
package de.vogella.camera.api;
import android.app.Activity;
import android.content.pm.PackageManager;
import android.hardware.Camera;
import android.hardware.Camera.CameraInfo;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Toast;
import de.vogella.cameara.api.R;
public class MakePhotoActivity extends Activity {
    private final static String DEBUG_TAG = "MakePhotoActivity";
    private Camera camera;
    private int cameraId = 0;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        // do we have a camera?
```

```
        if (!getPackageManager()
                .hasSystemFeature(PackageManager.FEATURE_CAMERA)) {
            Toast.makeText(this, "No camera on this device", Toast.LENGTH_LONG)
                    .show();
        } else {
            cameraId = findFrontFacingCamera();
            if (cameraId < 0) {
                Toast.makeText(this, "No front facing camera found.",
                        Toast.LENGTH_LONG).show();
            } else {
                camera = Camera.open(cameraId);
            }
        }
    }
    public void onClick(View view) {
        camera.startPreview();
        camera.takePicture(null, null,
                new PhotoHandler(getApplicationContext()));
    }
    private int findFrontFacingCamera() {
        int cameraId = -1;
        // Search for the front facing camera
        int numberOfCameras = Camera.getNumberOfCameras();
        for (int i = 0; i < numberOfCameras; i++) {
            CameraInfo info = new CameraInfo();
            Camera.getCameraInfo(i, info);
            if (info.facing == CameraInfo.CAMERA_FACING_FRONT) {
                Log.d(DEBUG_TAG, "Camera found");
                cameraId = i;
                break;
            }
        }
        return cameraId;
    }
    @Override
    protected void onPause() {
        if (camera != null) {
            camera.release();
            camera = null;
        }
        super.onPause();
    }
}
```

بخش دوم :

آموزش استفاده از Android Sensor Manager

آموزش حاضر به شرح نحوه ی استفاده از کلاس SensorManager می پردازد. پروژه ی این مبحث در ویرایش 3.6 محیط کاری Eclispe نوشته شده و بر پایه ی ورژن 1.6 زبان جاوا و نسخه ی 2.3.3 اندروید (Gingerbread) می باشد.

پیاده سازی کلاس SensorManager و دسترسی به حسگرهای دستگاه اندروید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

برای دسترسی به حسگرهای دستگاه اندروید می بایست نمونه ای از کلاس SensorManager را پیاده سازی نمایید. از جمله ی این حسگرها می توان به شتاب سنج یا accelerometer اشاره کرد. متاسفانه، امکان تست شتاب سنج و شبیه سازی آن در Android emulator وجود ندارد.

برای دسترسی به کلاس SensorManager کافی است متد getSystemService(SENSOR_SERVICE) را فراخوانی نمایید. کلاس Sensor ثوابت متعددی (constants) برای دسترسی به حسگرهای مختلف فراهم می آورد.

- Sensor.TYPE_GYROSCOPE
- Sensor.TYPE_MAGNETIC_FIELD
- Sensor.TYPE_ORIENTATION
- Sensor.TYPE_ACCELEROMETER

می توانید با فراخوانی متد sensorManager.getDefaultSensor() به حسگر مربوطه دسترسی داشته باشید. این متد نوع حسگر (sensor type) و میزان زمان غیرفعال بودن تا فراخوانی حسگر را که به عنوان ثابت در سطح کلاس SensorManager تعریف شده، به عنوان پارامتر می پذیرد.

گوش فرادهنده به تغییرات حسگر (Sensor listener)

پس از دسترسی به حسگر مربوطه، لازم است یک SensorEventListener برای آن تعریف نمایید. این listener به تغییرات مرتبط با حسگر (sensor) گوش می دهد و به محض رخداد تغییر در اطلاعات حسگر مورد نظر، از آن با خبر می شود.

به منظور جلوگیری از اتلاف باتری، می توانید listener خود را در متد onResume() اعلان و ثبت نموده و داخل متد onPause() از حالت مزبور خارج نمایید (deregister کنید).

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آموزش استفاده ی کاربردی از حسگر Accelerometer

در این بخش یک اپلیکیشن خواهیم نوشت که در صورت تکان خوردن دستگاه، به صورت خودکار رنگ پس زمینه را تغییر می دهد. برای این منظور ابتدا یک پروژه ی اندرویدی و activityبه ترتیب به نام های de.vogella.android.sensor و SensorTestActivity ایجاد نمایید.

محتوای فایل layout خود را به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <TextView
      android:id="@+id/textView"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:text="Shake to get a toast and to switch color" />
</LinearLayout>
```

حال کد کلاس activity خود را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.sensor;
import android.app.Activity;
import android.graphics.Color;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.view.View;
import android.view.Window;
import android.view.WindowManager;
import android.widget.Toast;
public class SensorTestActivity extends Activity implements SensorEventListener {
      private SensorManager sensorManager;
      private boolean color = false;
      private View view;
      private long lastUpdate;
      /** Called when the activity is first created. */
      @Override
      public void onCreate(Bundle savedInstanceState) {
            requestWindowFeature(Window.FEATURE_NO_TITLE);
            getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
                        WindowManager.LayoutParams.FLAG_FULLSCREEN);
            super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.main);
        view = findViewById(R.id.textView);
        view.setBackgroundColor(Color.GREEN);
        sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
        lastUpdate = System.currentTimeMillis();
    }
    @Override
    public void onSensorChanged(SensorEvent event) {
        if (event.sensor.getType() == Sensor.TYPE_ACCELEROMETER) {
            getAccelerometer(event);
        }
    }
    private void getAccelerometer(SensorEvent event) {
        float[] values = event.values;
        // Movement
        float x = values[0];
        float y = values[1];
        float z = values[2];
        float accelationSquareRoot = (x * x + y * y + z * z)
                / (SensorManager.GRAVITY_EARTH * SensorManager.GRAVITY_EARTH);
        long actualTime = event.timestamp;
        if (accelationSquareRoot >= 2) //
        {
            if (actualTime - lastUpdate < 200) {
                return;
            }
            lastUpdate = actualTime;
            Toast.makeText(this, "Device was shuffed", Toast.LENGTH_SHORT)
                    .show();
            if (color) {
                view.setBackgroundColor(Color.GREEN);
            } else {
                view.setBackgroundColor(Color.RED);
            }
            color = !color;
        }
    }
    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {
    }
    @Override
    protected void onResume() {
        super.onResume();
        // register this class as a listener for the orientation and
        // accelerometer sensors
        sensorManager.registerListener(this,
                sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
                SensorManager.SENSOR_DELAY_NORMAL);
    }
    @Override
    protected void onPause() {
        // unregister listener
```

```
        super.onPause();
        sensorManager.unregisterListener(this);
    }
}
```

آموزش کاربردی: ساخت یک قطب نما

یک پروژه و activity جدید اندرویدی به ترتیب به نام های de.vogella.android.sensor.compass و MainActivity ایجاد نمایید.

اکنون یک کلاس View با پیاده سازی اختصاصی به صورت زیر ایجاد کنید.

```
package de.vogella.android.sensor.compass;
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.view.View;
public class MyCompassView extends View {
    private Paint paint;
    private float position = 0;
    public MyCompassView(Context context) {
        super(context);
        init();
    }
    private void init() {
        paint = new Paint();
        paint.setAntiAlias(true);
        paint.setStrokeWidth(2);
        paint.setTextSize(25);
        paint.setStyle(Paint.Style.STROKE);
        paint.setColor(Color.WHITE);
    }
    @Override
    protected void onDraw(Canvas canvas) {
        int xPoint = getMeasuredWidth() / 2;
        int yPoint = getMeasuredHeight() / 2;
        float radius = (float) (Math.max(xPoint, yPoint) * 0.6);
        canvas.drawCircle(xPoint, yPoint, radius, paint);
        canvas.drawRect(0, 0, getMeasuredWidth(), getMeasuredHeight(), paint);
        // 3.143 is a good approximation for the circle
        canvas.drawLine(
                xPoint,
                yPoint,
                (float) (xPoint + radius
                        * Math.sin((double) (-position) / 180 * 3.143)),
                (float) (yPoint - radius
                        * Math.cos((double) (-position) / 180 * 3.143)), paint);
```

```
            canvas.drawText(String.valueOf(position), xPoint, yPoint, paint);
        }
        public void updateData(float position) {
            this.position = position;
            invalidate();
        }
}
```

بدنه ی کلاس activity پروژه ی خود را به صورت زیر ویرایش کنید.

```
package de.vogella.android.sensor.compass;
import android.app.Activity;
import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.util.Log;
import android.widget.Toast;
public class MainActivity extends Activity {
    private static SensorManager sensorService;
    private MyCompassView compassView;
    private Sensor sensor;
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        compassView = new MyCompassView(this);
        setContentView(compassView);
        sensorService = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        sensor = sensorService.getDefaultSensor(Sensor.TYPE_ORIENTATION);
        if (sensor != null) {
            sensorService.registerListener(mySensorEventListener, sensor,
                    SensorManager.SENSOR_DELAY_NORMAL);
            Log.i("Compass MainActivity", "Registerered for ORIENTATION Sensor");
        } else {
            Log.e("Compass MainActivity", "Registerered for ORIENTATION Sensor");
            Toast.makeText(this, "ORIENTATION Sensor not found",
                    Toast.LENGTH_LONG).show();
            finish();
        }
    }
    private SensorEventListener mySensorEventListener = new SensorEventListener() {
        @Override
        public void onAccuracyChanged(Sensor sensor, int accuracy) {
        }
        @Override
        public void onSensorChanged(SensorEvent event) {
            // angle between the magnetic north direction
            // 0=North, 90=East, 180=South, 270=West
            float azimuth = event.values[0];
```

```
                compassView.updateData(azimuth);
            }
        };
        @Override
        protected void onDestroy() {
            super.onDestroy();
            if (sensor != null) {
                sensorService.unregisterListener(mySensorEventListener);
            }
        }
}
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش سوم :

آموزش استفاده ی کاربردی از قابلیت single touch و multi touch در اندروید / آموزش پیاده سازی و استفاده از توابع کتابخانه ای touch API در اپلیکیشن های اندرویدی

آموزش حاضر به شرح پیاده سازی و استفاده از توابع کتابخانه ای touch API و قابلیت لمس نمایشگر در اپلیکیشن های اندرویدی می پردازد.

رخدادهای مربوط به لمس نمایشگر (Android touch)

واکنش نشان دادن به event های مربوط به touch

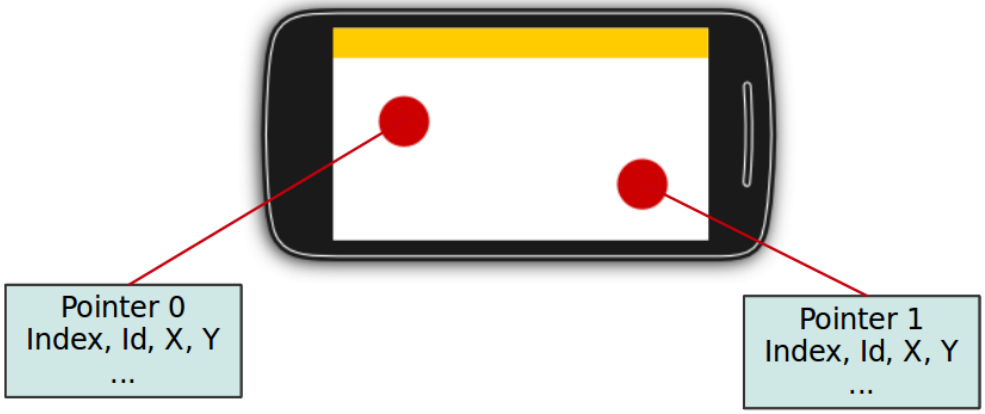
کلاس استاندارد و پرکاربرد View کتابخانه های اندروید به راحتی قادر است از رخدادهای touch پشتیبانی کرده و به آن ها پاسخ دهد.

می توانید به رخدادهایی که توسط لمس نمایشگر فعال می شوند، در activity و view های پروژه ی خود پاسخ دهید. سیستم اندروید این قابلیت را دارد تا همزمان از چندین pointer که نشانگر انگشت بر روی نمایشگر و تعامل کاربر با چند نقطه از صفحه نمایش دستگاه می باشد، پشتیبانی کند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Pointers



کلاس پایه ای (پدر) که برای پشتیبانی از قابلیت لمس نمایشگر، توسعه می بایست از آن ارث بری کند، کلاس MotionEvent می باشد. این کلاس به واسطه ی متد onTouchEvent() به عنوان پارامتر به آبجکت های Views فرستاده می شود.

به منظور فراهم آوردن قابلیت پاسخ دهی و واکنش نشان دادن به event های مربوط به touch (لمس نمایشگر)، کافی است متد onTouchEvent() را بازنویسی (override) نمایید.

کلاس MotionEvent تمامی اطلاعات مربوط به قابلیت touch نظیر تعداد pointer ها، مختصات X/Y، میزان اندازه و pressure هر pointer را در بردارد.

چنانچه event فعال شده توسط لمس نمایشگر را view مدیریت کند، آنگاه متد نام برده مقدار بولی true را در خروجی برمی گرداند. لازم به ذکر است که سیستم اندروید تلاش می کند تودرتوترین (deepest) کلاس view ای که event مربوط به touch را مدیریت کرده و مقدار true را برگردانده، پیدا کند.در صورتی که view خود داخل view دیگر (parent view یا view میزبان) جای گرفته باشد، آنگاه view میزبان می تواند با بازگردانی true از تابع onInterceptTouchEvent()، جلوی ارسال رخداد به view زیرمجموعه را بگیرد. در نتیجه ی این

عملیات، رخداد MotionEvent.Action_CANCEL به view ای که قبلا قرار بود رخداد اصلی را دریافت کند، ارسال شده و view میزبان خود مدیریت event را بر عهده می گیرد.

برای واکنش نشان دادن به رخدادهای لمس نمایشگر در سطح یک activity، کافی است OnTouchListener را برای View های مربوطه اعلان نمایید.

وضعیت لمس یک نقطه از نمایشگر (single touch)

چنانچه تنها یک انگشت در آن واحد با سطح نمایشگر برخورد داشته و سیستم اندروید تنها یک ورودی دریافت کرده باشد، در آن صورت توسعه دهنده می تواند با فراخوانی متد getX() و getY()، مقادیر و اطلاعات مربوط به موقعیت فعلی (مختصات) اولین انگشتی که با صفحه تعامل داشته را بازیابی کند.

با فراخوانی متد getAction() توسعه دهنده action و فعلی که اجرا شده را در خروجی دریافت می کند.

کلاس MotionEvent از مجموعه کتابخانه های اندروید تعدادی ثابت (constant) در اختیار برنامه نویس قرار می دهد که از طریق آن ها قادر خواهد بود اطلاعات مربوط به action رخ داده را مشخص نماید.

Event	شرح
MotionEvent.ACTION_DOWN	New touch started (انگشت با سطح نمایشگر برخورد کرده و در نتیجه touch یا لمس دیگری آغاز شده است)
MotionEvent.ACTION_MOVE	Finger is moving (انگشت کاربر در سطح نمایشگر در حال حرکت است)
MotionEvent.ACTION_UP	Finger went up (انگشت کاربر دیگر با نمایشگر تعامل ندارد)

Event	شرح
MotionEvent.ACTION_CANCEL	رخداد جاری لغو شده (به view زیرمجموعه یا فرزند ارسال نمی شود) و view دیگری مدیریت event مربوطه را برعهده می گیرد.
MotionEvent.ACTION_POINTER_DOWN	Pointer down (multi-touch) انگشت کاربر با سطح نمایشگر برخورد کرده و بر روی آن قرار دارد (وضعیت پشتیبانی از لمس همزمان چند نقطه از نمایشگر).
MotionEvent.ACTION_POINTER_UP	Pointer up (multi-touch) انگشت کاربر از روی سطح نمایشگر برداشته شده است (وضعیت پشتیبانی از لمس همزمان چند نقطه از نمایشگر).

قابلیت لمس چندین نقطه از صفحه (Multi touch)

قابلیت پیاده سازی و پشتیبانی از وضعیت لمس چندین نقطه از نمایشگر به طور همزمان در اپلیکیشن های اندرویدی از ویرایش 2.0 کتابخانه های اندروید اضافه شده و ورژن 2.2 به مراتب ارتقا یافت.

MotionEvent.ACTION_POINTER_DOWN و MotionEvent.ACTION_POINTER_UP زمانی ارسال می شوند که انگشت دوم با سطح نمایشگر تعامل برقرار کند.

دو رخداد MotionEvent.ACTION_DOWN و MotionEvent.ACTION_UP نیز به هنگام برخورد انگشت اول با سطح نمایشگر مورد استفاده قرار می گیرند.

متد getPointerCount() که در سطح کلاس MotionEvent موجود است، به شما این اجازه را می دهد تا تعداد pointer ها (انگشت یا شی دیگری که با نمایشگر ارتباط داشته) را بازیابی نمایید. تمامی رخدادهای مربوطه، به همراه موقعیت و مختصات pointer ها همگی در نمونه ی ایجاد شده از MotionEvent که به عنوان پارامتر متد onTouch() دریافت می کنید، موجود می باشد.

جهت رصد (track) و ردیابی اطلاعات مربوط به تمامی pointer ها، لازم است دو متد MotionEvent.getActionIndex() و MotionEvent.getActionMasked() را فراخوانی کنید و از طریق آن ها اندیس pointer و رخداد مربوط به touch که از برخورد این pointer فعال شده، شناسایی نمایید.

اندیس pointer ممکن است در طول زمان تغییر کند. به طور مثال می توان به زمانی اشاره کرد که یک انگشت از سطح نمایشگر برداشته می شود. نسخه ی پایدار از pointer همان pointer id است که با متد getPointerId(pointerIndex) از آبجکت MotionEvent قابل شناسایی و بازیابی می باشد.

کاربرد آن به صورت عملی در زیر نمایش داده شده است.

```
@Override
public boolean onTouchEvent(MotionEvent event) {
        // get pointer index from the event object
        int pointerIndex = event.getActionIndex();
        // get pointer ID
        int pointerId = event.getPointerId(pointerIndex);
        // get masked (not specific to a pointer) action
        int maskedAction = event.getActionMasked();
        switch (maskedAction) {
        case MotionEvent.ACTION_DOWN:
        case MotionEvent.ACTION_POINTER_DOWN: {
            // TODO use data
            break;
        }
        case MotionEvent.ACTION_MOVE: { // a pointer was moved
            // TODO use data
            break;
        }
        case MotionEvent.ACTION_UP:
        case MotionEvent.ACTION_POINTER_UP:
        case MotionEvent.ACTION_CANCEL: {
            // TODO use data
            break;
        }
        }
        invalidate();
        return true;
}
```

نکته: امکان تست قابلیت Multitouch در شبیه ساز وجود ندارد. برای آزمودن قابلیت لمس چندین نقطه از نمایشگر به طور همزمان، نیاز به یک دستگاه واقعی اندروید دارید (دستگاهی که ورودی را دریافت می کند باید یک دستگاه حقیقی اندروید باشد).

کلاس GestureDetectors

چارچوب نرم افزاری اندروید کلاسی به نام GestureDetector را ارائه می دهد که از اطلاعات و اعضای آبجکت (نمونه ی ساخته شده از کلاس) MotionEvents استفاده می نماید. این کلاس قادر است انواع حرکت ها (gesture) و رخدادها (event ها) را با استفاده از نمونه ی کلاس MotionEvents که به عنوان پارامتر ارائه می شود، شناسایی کند. تابع بازفراخوان GestureDetector.OnGestureListener (callback) کاربران را به محض رخداد مربوطه از motion event (رخدادی که بر اثر حرکت فعال می شود) مورد نظر مطلع می سازد.

به طور مثال کلاس ScaleGestureDetector این امکان را به شما می دهد تا حرکت از پیش تعیین شده ی کوچک و بزرگ کردن یک آبجکت با دو انگشت را تعریف نمایید.

تمرین: پیاده سازی view اختصاصی و مدیریت رخدادهای مربوط به لمس یک نقطه از نمایشگر (single touch event)

ترسیم با استفاده از قابلیت لمس نمایشگر

تمرین حاضر نحوه ی مدیریت رخدادهای مربوط به لمس یک نقطه از نمایشگر را داخل یک view اختصاصی نمایش می دهد.

ابتدا یک پروژه و activity جدید به ترتیب به نام های com.vogella.android.touch.single و SingleTouchActivity ایجاد نمایید. این activity از کلاس DialogFragment برای تنظیم رنگ قابل استفاده برای ترسیم در سطح نمایشگر بهره می گیرد.

حال یک فایل layout به نام fragment_colorpicker.xml با محتوای زیر ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical" android:layout_width="match_parent"
  android:layout_height="match_parent">
  <View
    android:layout_width="match_parent"
    android:layout_height="40dp"
    android:layout_margin="20dp"
    android:background="#ff0000"
    android:id="@+id/preview"
    />
  <SeekBar
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/red"
    android:progress="255"
    android:max="255"/>
  <SeekBar
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/green"
    android:max="255"/>
  <SeekBar
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/blue"
    android:max="255"/>
</LinearLayout>
```

کلاسی به نام TouchEventView تعریف کرده که یک آبجکت View با قابلیت پشتیبانی از لمس یک نقطه از نمایشگر در آن واحد را پیاده سازی می کند.

```
package com.vogella.android.touch.single;
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Path;
import android.util.AttributeSet;
import android.util.Log;
import android.view.GestureDetector;
import android.view.MotionEvent;
import android.view.VelocityTracker;
import android.view.View;
public class TouchEventView extends View {
  private Paint paint = new Paint();
  private Path path = new Path();
  Context context;
  GestureDetector gestureDetector;
  public TouchEventView(Context context, AttributeSet attrs) {
    super(context, attrs);
    gestureDetector = new GestureDetector(context, new GestureListener());
```

```
    this.context = context;
    paint.setAntiAlias(true);
    paint.setStrokeWidth(6f);
    paint.setColor(Color.BLACK);
    paint.setStyle(Paint.Style.STROKE);
    paint.setStrokeJoin(Paint.Join.ROUND);
  }
  public void setColor(int r, int g, int b) {
    int rgb = Color.rgb(r, g, b);
    paint.setColor(rgb);
  }
  private class GestureListener extends GestureDetector.SimpleOnGestureListener {
    // event when double tap occurs
    @Override
    public boolean onDoubleTap(MotionEvent e) {
      float x = e.getX();
      float y = e.getY();
      // clean drawing area on double tap
      path.reset();
      Log.d("Double Tap", "Tapped at: (" + x + "," + y + ")");
      return true;
    }
  }
  @Override
  protected void onDraw(Canvas canvas) {
    canvas.drawPath(path, paint);
  }
  @Override
  public boolean onTouchEvent(MotionEvent event) {
    float eventX = event.getX();
    float eventY = event.getY();
    switch (event.getAction()) {
    case MotionEvent.ACTION_DOWN:
      path.moveTo(eventX, eventY);
      return true;
    case MotionEvent.ACTION_MOVE:
      path.lineTo(eventX, eventY);
      break;
    case MotionEvent.ACTION_UP:
      break;
    default:
      return false;
    }
    // for demostraction purposes
    gestureDetector.onTouchEvent(event);
    // Schedules a repaint.
    invalidate();
    return true;
  }
}
```

محتوای فایل activity_main.xml را به صورت زیر اصلاح نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent">
  <com.vogella.android.touch.single.SingleTouchEventView
    android:layout_width="match_parent"
    android:background="@drawable/dwarf"
    android:layout_height="match_parent"
    android:id="@+id/drawingview">
  </com.vogella.android.touch.single.SingleTouchEventView>
</LinearLayout>
```

حال view مورد نظر را به کلاس activity خود اضافه نمایید.

```
package com.vogella.android.touch.single;
import android.app.Activity;
import android.os.Bundle;
public class SingleTouchActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(new SingleTouchEventView(this, null));
    }
}
```

اکنون زمانی که اپلیکیشن را اجرا می کنید، بایستی با توجه به کدی که نوشته اید قادر باشید با انگشت خود بر روی نمایشگر دستگاه اشکال مختلفی ترسیم کنید (در محیط شبیه ساز با استفاده از موس).

کد اپلیکیشن خود را طوری ویرایش کنید که از تعریف و محتوای یک فایل layout با فرمت XML برای تنظیم ظاهر خود استفاده کند.

نکته: به منظور اعلان view دلخواه خود در لایه ی XML اپیلکیشن، کد موجود در فایل layout، لازم است اسم و آدرس دقیق کلاس از جمله اطلاعات پکیج آن کلاس را ذکر نمایید.

افزودن قابلیت تنظیم پهنای خط ترسیم (line width)

یک منو به اپلیکیشن خود اضافه کنید (المان مربوطه را در لایه ی XML، داخل فایل layout به صورت زیر اعلان نمایید).

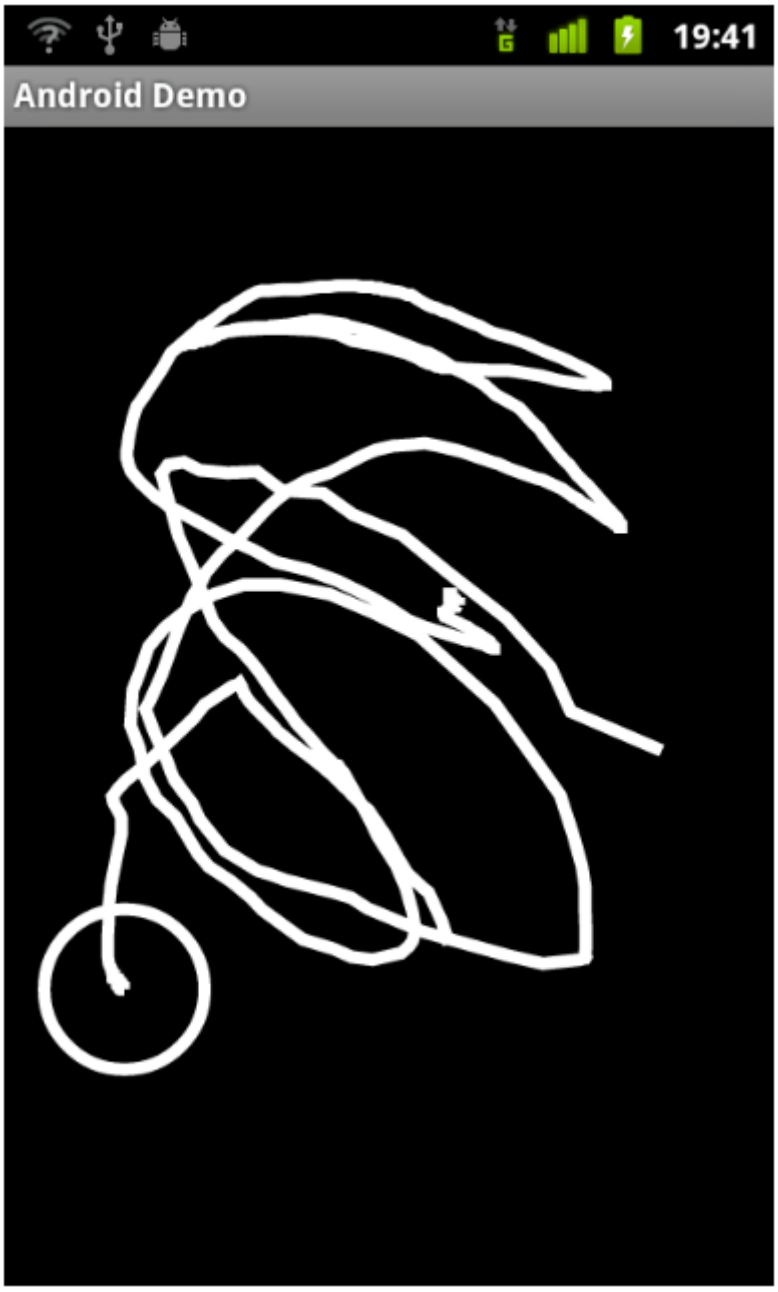
```
<?xml version="1.0" encoding="utf-8"?>
```

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:title="Pick pen" android:id="@+id/action_pickpen"/>
</menu>
```

رصد و ردیابی مختصات جاری انگشت کاربر بر روی نمایشگر

حال کدی به اپلیکیشن خود اضافه نمایید که موقعیت فعلی انگشت کاربر بر روی نمایشگر را با استفاده از یک دایره نشانه گذاری و دنبال کند. به منظور ترسیم دایره، می توانید متد addCircle(x, y, 50, Path.Direction.CW) را بر روی نمونه ی ساخته شده از کلاس Path فراخوانی کنید یا المان canvas را مستقیما بکار ببرید.

اطمینان حاصل کنید که تنها موقعیت جاری انگشت کاربر بر روی نمایشگر با استفاده از شکل دایره نشانه گذاری و هایلایت می شود. شکل مزبور بایستی به مجرد برخورد انگشت کاربر با سطح نمایشگر دستگاه پدیدار شده و زمانی که تماس انگشت با صفحه قطع شد، محو گردد. خروجی می بایست شمابه زیر باشد.



تمرین: پیاده سازی قابلیت پشتیبانی از لمس چندین نقطه از نمایشگر (Multitouch)

در تمرین جاری، توسعه دهنده می بایست یک view اعلان کند که از قابلیت multitouch پشتیبانی کرده و از این طریق امکان ردیابی و پاسخ دهی به چندین انگشت بر روی نمایشگر به

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

طور همزمان را فراهم آورد. لازم به ذکر است که تمرین پیاده سازی قابلیت لمس چند نقطه از نمایشگر در آن واحد در بستر شبیه ساز ممکن نبوده و توسعه دهنده می بایست برای این منظور حتما از دستگاه واقعی اندروید استفاده کند.

همان طور که قبلا گفته شد، نرم افزار emulator اندروید تنها اجازه ی شبیه سازی singletouch را (با اشاره گر موس) برای برنامه نویس فراهم می کند.

یک پروژه و activity جدید به ترتیب به نام های com.vogella.android.multitouch و MainActivity ایجاد نمایید.

ابتدا کلاسی به نام MultitouchView را به صورت زیر پیاده سازی کنید.

```
package com.vogella.android.multitouch;
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.PointF;
import android.util.AttributeSet;
import android.util.SparseArray;
import android.view.MotionEvent;
import android.view.View;
public class MultitouchView extends View {
    private static final int SIZE = 60;
    private SparseArray<PointF> mActivePointers;
    private Paint mPaint;
    private int[] colors = { Color.BLUE, Color.GREEN, Color.MAGENTA,
            Color.BLACK, Color.CYAN, Color.GRAY, Color.RED, Color.DKGRAY,
            Color.LTGRAY, Color.YELLOW };
    private Paint textPaint;
    public MultitouchView(Context context, AttributeSet attrs) {
        super(context, attrs);
        initView();
    }
    private void initView() {
        mActivePointers = new SparseArray<PointF>();
        mPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
        // set painter color to a color you like
        mPaint.setColor(Color.BLUE);
        mPaint.setStyle(Paint.Style.FILL_AND_STROKE);
        textPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
        textPaint.setTextSize(20);
    }
    @Override
    public boolean onTouchEvent(MotionEvent event) {
```

```
        // get pointer index from the event object
        int pointerIndex = event.getActionIndex();
        // get pointer ID
        int pointerId = event.getPointerId(pointerIndex);
        // get masked (not specific to a pointer) action
        int maskedAction = event.getActionMasked();
        switch (maskedAction) {
        case MotionEvent.ACTION_DOWN:
        case MotionEvent.ACTION_POINTER_DOWN: {
            // We have a new pointer. Lets add it to the list of pointers
            PointF f = new PointF();
            f.x = event.getX(pointerIndex);
            f.y = event.getY(pointerIndex);
            mActivePointers.put(pointerId, f);
            break;
        }
        case MotionEvent.ACTION_MOVE: { // a pointer was moved
            for (int size = event.getPointerCount(), i = 0; i < size; i++) {
                PointF point = mActivePointers.get(event.getPointerId(i));
                if (point != null) {
                    point.x = event.getX(i);
                    point.y = event.getY(i);
                }
            }
            break;
        }
        case MotionEvent.ACTION_UP:
        case MotionEvent.ACTION_POINTER_UP:
        case MotionEvent.ACTION_CANCEL: {
            mActivePointers.remove(pointerId);
            break;
        }
        }
        invalidate();
        return true;
    }
    @Override
    protected void onDraw(Canvas canvas) {
        super.onDraw(canvas);
        // draw all pointers
        for (int size = mActivePointers.size(), i = 0; i < size; i++) {
            PointF point = mActivePointers.valueAt(i);
            if (point != null)
                mPaint.setColor(colors[i % 9]);
            canvas.drawCircle(point.x, point.y, SIZE, mPaint);
        }
        canvas.drawText("Total pointers: " + mActivePointers.size(), 10, 40 , textPaint);
    }
}
```

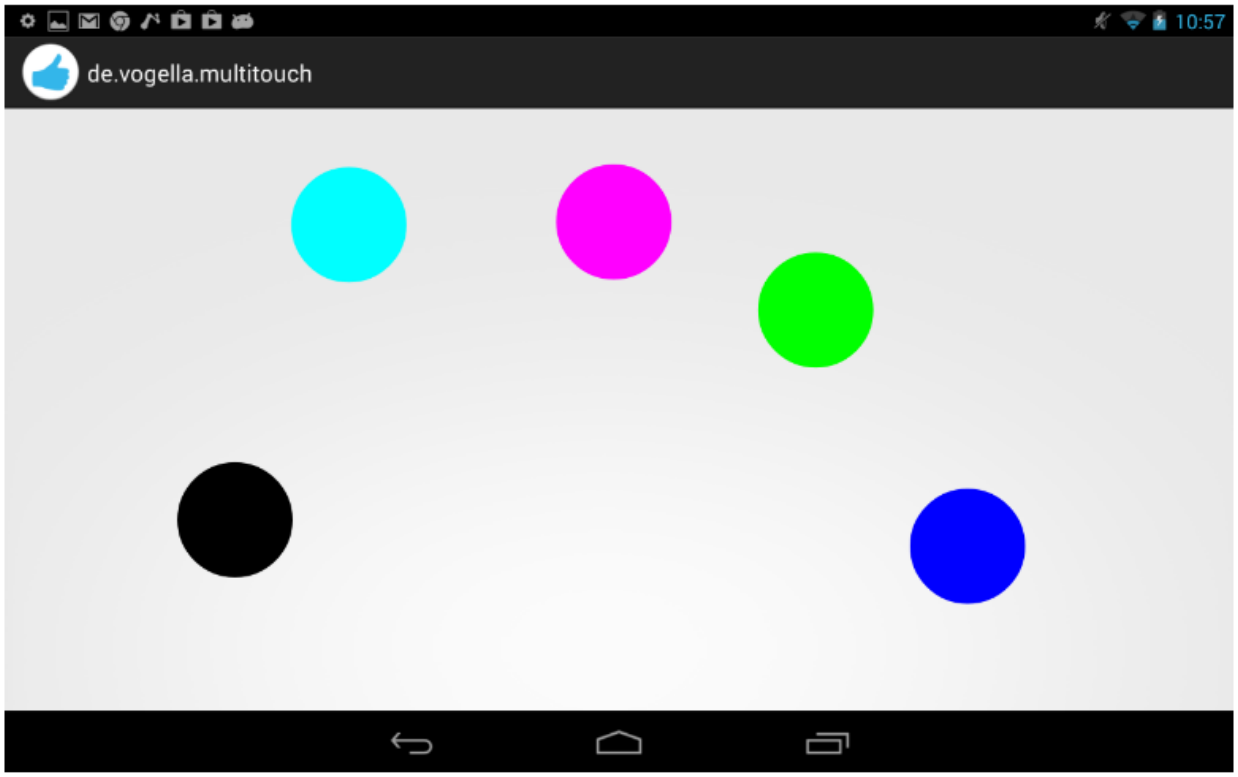
view زیر را به فایل layout متناظر activity خود اضافه نمایید.

```
<RelativeLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <com.vogella.android.multitouch.MultitouchView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      />
</RelativeLayout>
```

کلاس activity ای که قبلا ایجاد شده نیازی به تغییر ندارد.

```
package com.vogella.android.multitouch;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
public class MainActivity extends Activity {
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
      }
      @Override
      public boolean onCreateOptionsMenu(Menu menu) {
            getMenuInflater().inflate(R.menu.activity_main, menu);
            return true;
      }
}
```

اپلیکیشن در زمان اجرا به شما این امکان را می دهد تا با انگشت خود بر روی نمایشگر اشکال مختلف ترسیم نمایید. در هر دستگاهی، محدودیت از پیش تعیین شده ای برای حداکثر تعداد نقاط قابل پشتیبانی در آن واحد در نمایشگر وجود دارد. لازم است بررسی کنید دستگاه شما حداکثر چه تعداد pointer را به طور همزمان پشتیبانی می کند. در پایان، اپلیکیشن بایستی ظاهری مشابه زیر داشته باشد.



آموزش: پیاده سازی قابلیت بزرگ نمایی / کوچک نمایی و استفاده از multitouch

یک پروژه و activity اندروید به ترتیب به نام های de.vogella.android.touch.scaledetector و ScaleDetectorTestActivity ایجاد نمایید.

کلاس زیر را پیاده سازی نمایید.

```
package de.vogella.android.touch.scaledetector;
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.drawable.Drawable;
import android.view.MotionEvent;
import android.view.ScaleGestureDetector;
import android.view.View;
public class ImageViewWithZoom extends View {
	private Drawable image;
	private float scaleFactor = 1.0f;
	private ScaleGestureDetector scaleGestureDetector;
	public ImageViewWithZoom(Context context) {
		super(context);
```

```
            image = context.getResources().getDrawable(R.drawable.icon);
            setFocusable(true);
            image.setBounds(0, 0, image.getIntrinsicWidth(),
                    image.getIntrinsicHeight());
            scaleGestureDetector = new ScaleGestureDetector(context,
                    new ScaleListener());
        }
        @Override
        protected void onDraw(Canvas canvas) {
            super.onDraw(canvas);
            // Set the image bounderies
            canvas.save();
            canvas.scale(scaleFactor, scaleFactor);
            image.draw(canvas);
            canvas.restore();
        }
        @Override
        public boolean onTouchEvent(MotionEvent event) {
            scaleGestureDetector.onTouchEvent(event);
            invalidate();
            return true;
        }
        private class ScaleListener extends
                ScaleGestureDetector.SimpleOnScaleGestureListener {
            @Override
            public boolean onScale(ScaleGestureDetector detector) {
                scaleFactor *= detector.getScaleFactor();
                // don't let the object get too small or too large.
                scaleFactor = Math.max(0.1f, Math.min(scaleFactor, 5.0f));
                invalidate();
                return true;
            }
        }
}
```

این View را به activity خود اضافه نمایید.

```
package de.vogella.android.touch.scaledetector;
import android.app.Activity;
import android.os.Bundle;
public class ScaleDetectorTestActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(new ImageViewWithZoom(this));
    }
}
```

زمانی که اپلیکیشن را راه اندازی می کنید، بایستی بتوانید با استفاده از قابلیت multi-touch تصویر زیر را کوچک و بزرگ نمایید (pinch zoom).

بخش چهارم :

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آموزش کار با Gestures و مدیریت رخدادهای فعال شده از حرکات مختلف بر روی نمایشگر

مبحث حاضر نحوه ی استفاده از Gestures و GestureOverylayView در اندروید را به شما می آموزد. پروژه های این آموزش در نسخه ی 3.6 محیط کاری Eclipse و با استفاده از ویرایش 1.6 زبان جاوا نوشته شده و مبتنی بر ورژن 2.3 سیستم عامل اندروید (Gingerbread) می باشد.

کار با Gesture ها و پیاده سازی کلاس GestureOverlayView در اندروید

سیستم اندروید، برای لمس و حرکت دادن انگشت بر روی سطح نمایشگر، رخدادهای معینی نظیر بزرگ نمایی، فشار دادن طولانی و پیمایش با نوار اسکرول در صفحه را ارائه می دهد. به این رخدادها که در بر اثر لمس نمایشگر اتفاق افتاده و فعال می شوند در اصطلاح gestures گفته می شود.

برای استفاده از این امکان اندروید، اپلیکیشن شما بایستی یک نمونه از کلاس GestureOverlayView را پیاده سازی کند. سپس می توانید در این view، سایر کامپوننت های رابط کاربری و view های خود را جایگذاری نمایید.

gesture ها را منابع باینری تشکیل داده و در اختیار توسعه دهنده قرار می دهند. این منابع را می توان با استفاده از Android SDK تهیه کرد.

جهت بارگذاری این منابع باینری و gesture ها در activity یا صفحه قابل مشاهده برای کاربر لازم است متد GestureLib.fromRawResource() را فراخوانی کنید. پس از آنکه gesture توسط سیستم شناسایی شد، متد "onGesturePerformedListener" فراخوانده می شود. اما قبل از آن کلاس activity ای که می خواهد gesture را بارگذاری کند باید توابع اینترفیس "OnGesturePerformedListener" را پیاده سازی کرده و با فراخوانی متد

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

"addOnGesturePerformedListener()" بر روی آبجکت ساخته شده از روی GestureOverlayView، خود را معرفی نماید.

سیستم اندروید، آن دسته از gesture هایی را که شناخته با رنگ زرد و gesture هایی که برایش قابل شناسایی نیستند را با رنگ زرد کم رنگ نمایش می دهد. توسعه دهنده، در صورت تمایل، می تواند این قابلیت را غیر فعال کند. برای نیل به این هدف تنها کافی است که یکی از دو تابع setUncertainGestureColor(Color.TRANSPARENT) یا
setGestureColor(Color.TRANSPARENT) را بر روی (آبجکت یا نمونه از) GestureOverlayView فراخوانی نماید.

اگر از "GestureBuilder" برای ساخت gesture در شبیه ساز اندروید (Emulator) استفاده کنید، در آن صورت می توانید تعداد زیادی gesture با اسم یکسان ایجاد نمایید. از این طریق امکانی برای شما فراهم می شود تا gesture دلخواه و مد نظر خود را پیدا کرده و تعیین نمایید. اگر برای ویرایش 1.6 اندروید Android Emulator ایجاد کنید، این اپلیکیشن به صورت پیش فرض بر روی دستگاه شما نصب شده و آماده ی استفاده می باشد. لازم است یک دستگاه با قابلیت حافظه خارجی (sd card) ایجاد کنید چرا که در غیر این صورت امکان ذخیره ی gesture ها را نخواهید داشت. تمامی gesture ها در محیط شبیه ساز، داخل فایل gestures ذخیره می شوند.

شما می توانید تمامی gesture ها را به وسیله ی ابزار adb از شبیه ساز (emulator) به دستگاه محلی خود کپی نمایید. کافی است دستور زیر را در خط فرمان درج کنید:

```
./adb pull /sdcard/gestures ~/test
```

فایل gesture بایستی تحت آدرس "res/raw" در اپلیکیشن جایگذاری شود. پس از این کار می توانید از فایل مزبور به راحتی در آبجکت GestureOverlayView استفاده نمایید.

مثال

یک پروژه و activity جدید به ترتیب به نام های "de.vogella.android.gestures" و "GestureTest" ایجاد نمایید. سپس تعدادی gesture ایجاد کرده و آن ها را طبق توضیح فوق، در اپلیکیشن مورد نظر جایگذاری نمایید.

متاسفانه UI builder از GestureOverlayView پشتیبانی نمی کند. می توانید برای مشاهده ی متن خطای مربوطه (عدم امکان استفاده از GestureOverlayView در layout builder) به آدرس http://code.google.com/p/android/issues/detail?id=1368 مراجعه کنید. از این رو توسعه دهنده ناگذیر باید یک layout بدون gesture view ساخته و بعد این view را از طریق کدنویسی به GestureOverlayView اضافه کند.

فایل layout _main.xml_ را مانند زیر ایجاد کنید.

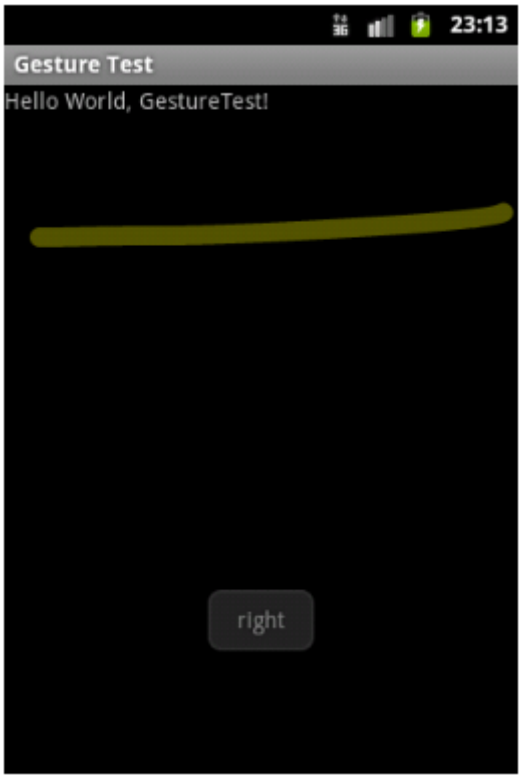
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:layout_width="match_parent"
   android:layout_height="match_parent" >
```

```
<TextView
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:text="@string/hello"/>
</LinearLayout>
```

کلاس activity زیر را پیاده سازی نمایید.

```
package de.vogella.android.gestures;
import java.util.ArrayList;
import android.app.Activity;
import android.gesture.Gesture;
import android.gesture.GestureLibraries;
import android.gesture.GestureLibrary;
import android.gesture.GestureOverlayView;
import android.gesture.GestureOverlayView.OnGesturePerformedListener;
import android.gesture.Prediction;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;
public class GestureTest extends Activity implements OnGesturePerformedListener {
    private GestureLibrary gestureLib;
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        GestureOverlayView gestureOverlayView = new GestureOverlayView(this);
        View inflate = getLayoutInflater().inflate(R.layout.main, null);
        gestureOverlayView.addView(inflate);
        gestureOverlayView.addOnGesturePerformedListener(this);
        gestureLib = GestureLibraries.fromRawResource(this, R.raw.gestures);
        if (!gestureLib.load()) {
            finish();
        }
        setContentView(gestureOverlayView);
    }
    @Override
    public void onGesturePerformed(GestureOverlayView overlay, Gesture gesture) {
        ArrayList<Prediction> predictions = gestureLib.recognize(gesture);
        for (Prediction prediction : predictions) {
            if (prediction.score > 1.0) {
                Toast.makeText(this, prediction.name, Toast.LENGTH_SHORT)
                        .show();
            }
        }
    }
}
```

اپلیکیشن را اجرا کنید. با توجه به منطق برنامه و کدی که برای اپلیکیشن خود نوشتید، باید بتوانید gesture های مختلفی ترسیم کرده و خروجی را در لحظه مشاهده نمایید. در اپلیشکن جاری تنها صرفا یک پیغام Toast نمایش داده می شود ولی شما می توانید هر عملیاتی که دوست دارید تعریف نمایید.



فصل یازدهم

پروژه های کتابخانه ای در اندروید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش اول :

آموزش استفاده از Support Library ها

آموزش حاضر نحوه ی استفاده از support library ها و همچنین دلایلی که چرا گاهی بهتر است از بکار بردن آن ها در پروژه پرهیز کرد را شرح می دهد.

استفاده از Support library های Google

شرح مفهوم Support Library

Support Libraryاین امکان را به توسعه دهنده می دهد تا قابلیت های نوین و جدید اضافه شده به Android SDK یا مجموعه ابزار ساخت و توسعه ی اپلیکیشن های اندروید را به همراه برنامه قرار دهد تا بدین وسیله برنامه قادر باشد از این امکانات در ویرایش های قدیمی اندروید نیز بهره بگیرد. به عبارت دیگر support library ها به منظور افزودن امکانات و قابلیت های ورژن های جدیدتر توابع و کتابخانه های اندروید به ویرایش های قدیمی تر تعبیه شدند.

پکیج Android Support Library در واقع مجموعه ای از code library ها (کتابخانه ای از کدهای آماده) هستند که نسخه های سازگار با ورژن های قدیمی تر از امکانات جدید کتابخانه های اندروید را به توابع کتابخانه ای و چارچوب نرم افزاری اندروید اضافه می کنند. علاوه بر آن ویژگی ها و widget هایی را برای استفاده در ورژن های قدیمی کتابخانه های اندروید فراهم می آورند که فقط و فقط از طریق API ها و توابع کتابخانه ای قابل دسترسی یا استفاده در پروژه هستند.

لازم به توضیح است که هر support library تنها برای ورژن (قدیمی) معینی از کتابخانه های اندروید (API level) قابل استفاده بوده و سازگار می باشد.

در کل دو نوع support library وجود دارد. اولین نوع قابلیت های جدید برای ورژن های قدیمی تر چارچوب نرم افزاری فراهم می کند (framework functionality جدید برای ویرایش های قدیمی تر کتابخانه های اندروید یا API release ارائه می دهد). نوع دوم support library، قابلیت های جدید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

برای تمامی دستگاه های مبتنی بر سیستم عامل اندروید فراهم می کند. حال این قابلیت یا هنوز به محیط (platform) اندروید اضافه نشده یا هیچگاه جزوی از آن نخواهد بود زیرا شرگت گوگل سعی بر این دارد که همواره امکانات و قابلیت های جدیدتر به آن اضافه کند و کتابخانه ی مزبور را سریع تر از توابع کتابخانه ای محیط اندروید (Android platform API) ارتقا دهد.

نصب Support library

support library به شما این امکان را می دهد تا از قابلیت های بیشتری در اپلیکیشن های اندرویدی خود استفاده کنید. حال این قابلیت ها ممکن است توسط ورژن های جدید کتابخانه های اندروید (API level) در اختیار توسعه دهنده قرار گیرد. علاوه بر آن امکانات دیگری نیز می تواند فراهم نماید که به صورت مجزا (unbundled) از چارچوب نرم افزاری اندروید، همچون ویجت RecyclerView که به منظور پیاده سازی هر چه بهینه تر لیست تعبیه شده، ارائه می شود.

اندروید ورژن های متعددی از support library ها ارائه می دهد که هر یک، ویژه ی نسخه ی خاصی از کتابخانه های اندروید (API level) قابل استفاده می باشد. به طور مثال، support library v7 برای آن دسته از دستگاه های اندروید که API level آن ها 7 باشد، قابل استفاده می باشد. نسخه های جدیدتر از support library برای عملکرد صحیح به نسخه های پایین تر نیز احتایج دارند. support library v7 به v4 احتیاج دارد.

Support library هایی که توسط گوگل ارائه می شوند

در زمان تنظیم مقاله ی حاضر، کامپوننت های متعددی توسط شرکت گوگل تعبیه شده که در زیر به آن ها اشاره می کنیم:

Support Library	شرح	Gradle dependency کتابخانه و نیازمندی های Gradle
v4 Support Library	ویژه ی ویرایش API) Android 1.6 level 4) و بالاتر تعبیه شده است.	این کتابخانه در آدرس android- sdk/extras/android/support/v7/appcompat/ directory موجود می باشد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Support Library	شرح	Gradle dependency کتابخانه و نیازمندی های Gradle
	دربردارنده ی قابلیت های فراوانی نظیر امکان پشتیبانی از fragment ها و فریم ورک Loader می باشد.	
v7 Support Libraries		مجموعه ای از کتابخانه ها که به Android 2.1 (کتابخانه های اندروید ورژن 7) و بالاتر برای عملکرد صحیح احتیاج دارند. علاوه بر آن لازم است v4 Support library در دسترس باشد.
v7 appcompat library	امکان پشتیبانی از Action Bar و پیاده سازی آن در بستر اپلیکیشن را فراهم می آورد. علاوه بر به توسعه دهنده امکان می دهد برای طراحی لایه ی UI اپلیکیشن از Material design استفاده کند.	در آدرس android- sdk/extras/android/support/v7/appcompat/ موجود می باشد.
v7 cardview library	با افزودن این نیازمندی به پروژه قادر خواهید بود که ویجت CardView را در اپلیکیشن خود پیاده سازی کنید.	android- sdk/extras/android/support/v7/cardview/

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Support Library	شرح	Gradle dependency کتابخانه و نیازمندی های Gradle
v7 recyclerview library	امکان استفاده از RecyclerView را پروژه فراهم می آورد.	compile "com.android.support:recyclerview-v7:24.0.0"
Design library to support material design	امکان استفاده از material design در طراحی رابط کاربری اپلیکیشن را فراهم می آورد.	compile 'com.android.support:design:24.0.0'

حذف support library از پروژه

چرا باید support library را حذف کرد؟

support library ها همواره در حال تغییر هستند و از این جهت تنظیم یک مقاله ی پایدار (با مطالبی که طولانی مدت بروز و مفید باشند) در شرح نحوه ی استفاده از این کتابخانه ها کاری بسیار دشوار است. علاوه بر آن الگو و قالب های آماده ی ساخت پروژه های اندرویدی در محیط کاری Android studio نیز همانند کتابخانه ی نام برده به سرعت بروز رسانی و ویرایش می شوند. این امر تست ویژگی جدید یا API های استاندارد را به مراتب دشوارتر می سازد.

تبدیل پروژه به یک پروژه ی استاندارد اندروید

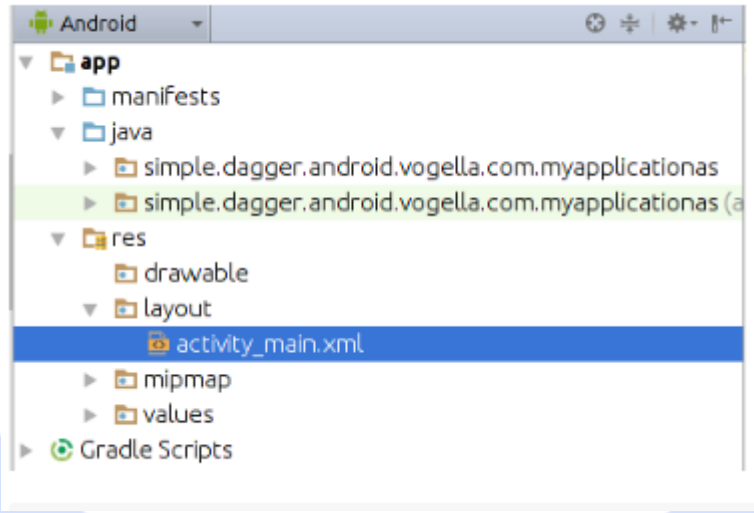
build dependency (کتابخانه ها و نیازمندی های build) در فایل Gradle build آسیبی به پروژه ی شما نمی رساند، اما توصیه می شود ارجاعات یا اشاره گر (reference) به support library / compatibility library را از کد و فایل های resource خود حذف نمایید.

برای این منظور، کد کلاس activity را طوری ویرایش کنید که منحصرا از کلاس پایه ی Activity ارث بری داشته باشد (نه از کلاس های activity دیگری نظیر AppCompatActivity). همچنین لازم

است کد را بهینه و تا حد ممکن مختصر تنظیم نمایید. در زیر نمونه ای پیاده سازی یک کلاس activity به نام MainActivity را مشاهده می کنید.

```
package com.vogella.android.myapplicationsdf;
import android.app.Activity;
import android.os.Bundle;
public class MainActivity extends Activity {
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
   }
}
```

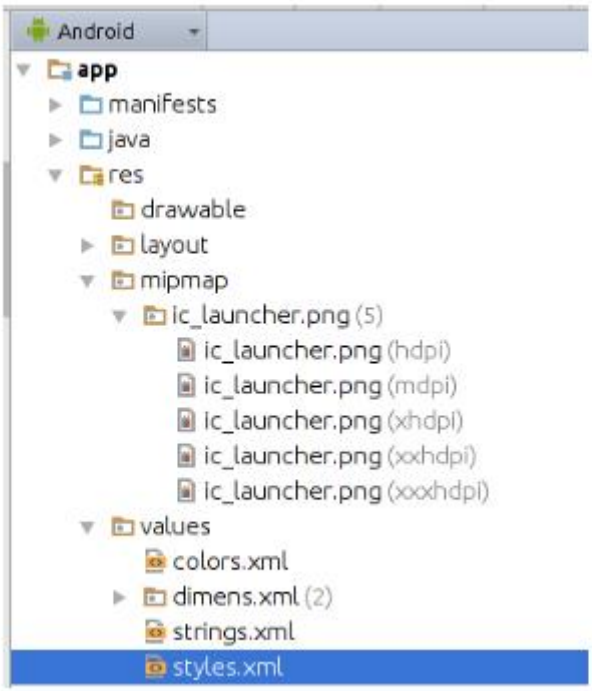
همچنین سبب می شود تنها المان های استاندارد از فایل layout متناظر استفاده کنند. در زیر نمونه ای از یک فایل layout استاندارد را مشاهده می کنید.




```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
```

```
    tools:context=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!" />
</RelativeLayout>
```

بجای compatibility theme) theme هایی که با ورژن های قدیمی اندروید نیز سازگاری دارند)، حتما از theme پیش فرض و درون ساخته ی اندروید استفاده نمایید. برای این منظور محتوای پوشه ی app/res/values/styles.xml را بررسی کرده و از material theme (یک style جدید ویژه ی طراحی UI ساده / material design برای اپلیکیشن که از ویرایش 5.0 اندروید معرفی شد) استفاده کنید.




```
<resources>
    <!-- Base application theme. -->
    <style name="AppTheme" parent="android:Theme.Material.Light.DarkActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style>
</resources>
```

بخش دوم :

نوشتن کتابخانه برای اپلیکیشن های اندرویدی

آموزش حاضر به شرح ایجاد و استفاده از پروژه های کتابخانه ای (library project) در اندروید می پردازد.

پروژه های کتابخانه ای اندروید و کتابخانه های جاوا

استفاده از کتابخانه های اندروید و جاوا

یک پروژه ی اندرویدی به راحتی می تواند از کدهای موجود در فایل های JAR (فایل های فشرده حاوی چندین کلاس جاوا به همراه metadata و منابع مورد نیاز که به کتابخانه های جاوایی معروف هستند) استفاده کند.

جدا از فایل های JAR، اندروید از یک فرمت توزیع نرم افزار (binary distribution) ویژه به نام Android ARchive(AAR) (کتابخانه هایی که جدا از کلاس های جاوا دربردارنده ی انواع فایل ها و منابع مبتنی بر XML می باشند) استفاده می کند. aar bundle. در واقع یک پروژه ی کتابخانه ای اندروید است با روش توزیع نرم افزار binary distribution (به صورت کامپایل شده) ارائه ی می شود.

یک فایل AAR تقریبا شبیه به یک فایل JAR است اما علاوه بر کلاس های نوشته شده با جاوا، حامل سایر منابع و محتوا و نیز byte-code کامپایل شده می باشد. فایل AAR را می توان در فرایند build و کامپایل اپلیکیشن درست مانند فایل JAR بکار برد.

توسعه دهنده می تواند ماژول های کتابخانه ای ایجاد کرده و از آن ها به عنوان dependency در پروژه های اندرویدی استفاده کند. این ماژول ها به شما امکان می دهد تا کد برنامه (source code) و منابعی همچون عکس و متن را ذخیره کرده و بین سایر پروژه های اندرویدی به اشتراک گذاشته و استفاده کنید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

نحوه ی استفاده از فایل JAR در اپلیکیشن اندرویدی

به منظور استفاده از یک فایل JAR (فایل کتابخانه ای جاوا) در پروژه، کافی است آن فایل را داخل پوشه ی libs در اپلیکیشن خود جایگذاری کنید. فایل های jar داخل این پوشه از طریق فایل build.gradle پیش فرض، در compile classpath قرار داده می شوند.

محدودیت هایی در استفاده از کتابخانه های java

در صورت نیاز به استفاده از کتابخانه ها لازم است این نکته را به خاطر داشته باشید که این کتابخانه ها تنها باید از API ها و توابع متعارف و شناخته شده در اندروید استفاده کنند. به طور مثال، کتابخانه های اندروید کتابخانه های لازم برای طراحی رابط کاربری اپلیکیشن همچون java.awt و javax.swing را ندارند. کتابخانه ای که کلاس های موجود در این کتابخانه ها را فراخوانی کند، قابل استفاده در اندروید نخواهد بود.

پروژه های کتابخانه ای قابل کامپایل و تبدیل به اپلیکیشن های کامل اندروید نبوده و به صورت مستقل از پروژه ی بزرگتر که از کلاس ها و توابع آن در خود استفاده کند، قابل اجرا نخواهند بود.

ماژول های اختصاصی کتابخانه ای اندروید (library modules)

ماژول های کتابخانه ای (library module)

زمانی که یک پروژه (application project) از ماژول کتابخانه (Android library module یا همان ماژول که بخشی مجزا از اپلیکیشن است) استفاده می کند، ابزار ساخت و توسعه ی اپلیکیشن های اندروید / ADT کد و منابع لازم را از پروژه ی کتابخانه ای در خروجی کامپایل و build شده ی پروژه می گنجاند. این بدین معنی است که کامپوننت ها، کد و منابع پروژه کتابخانه ای (library project) کاملا ترجمه شده و همگی در قالب فایل apk. اپلیکیشن کامپایل شده پکیج بندی می شوند.

از اینرو می توان به library module به چشم یک مصنوع زمان کامپایل نگاه کرد. ماژول کتابخانه ای می تواند شامل کلاس های جاوا، کامپوننت های نرم افزاری اندروید و منابع باشد. تنها نوع فایلی که در این کتابخانه ها جایی ندارد، asset ها هستند.

جهت ایجاد یک پروژه ی کتابخانه، در ویزارد ساخت پروژه های اندرویدی (Android project generation wizard)، گزینه ی Mark this project as library را انتخاب کنید.

یک پروژه ی کتابخانه ای بایستی تمامی کامپوننت های خود نظیر activity ها، سرویس و غیره ... را در لایه ی XML، داخل فایل تنظیمات AndroidManifest.xml اعلان کند. اپلیکیشنی که از این کتابخانه استفاده می کند نیز بایستی تمامی کامپوننت های نرم افزاری مورد استفاده را داخل فایل تنظیمات اپلیکیشن مربوطه اعلان نماید.

اولویت در انتخاب منابع (conflicting resources)

ابزار ساخت و توسعه اپلیکیشن های اندرویدی (Android development tool) منابع پروژه ی کتابخانه را با منابع پروژه ی اپلیکیشن ادغام می کند. حال چنانچه ID یا شناسه ی منابع پروژه چندین بار به صورت تکراری اعلان شده باشد، ADT تلاش می کند آن منبعی را از اپلیکیشن یا کتابخانه گزینش نماید که بالاترین اولویت را داشته و منابع اضافی را دور می ریزد.

ایجاد ماژول های کتابخانه ای اختصاصی در اندروید (Android library module)

استفاده از پروژه های کتابخانه ای (library project) به توسعه دهنده کمک می کند تا کد اپلیکیشن را به صورت بهینه سازمان دهی کند. جهت ایجاد یک ماژول کتابخانه ای جدید در محیط برنامه نویسی Android Studio، بر روی المان File از نوار بالایی محیط کلیک کرده و گزینه ی New Module را باز نمایید و سپس Android Library را انتخاب کنید.

تمرین: ساخت یک library module (کتابخانه ی اندروید)

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

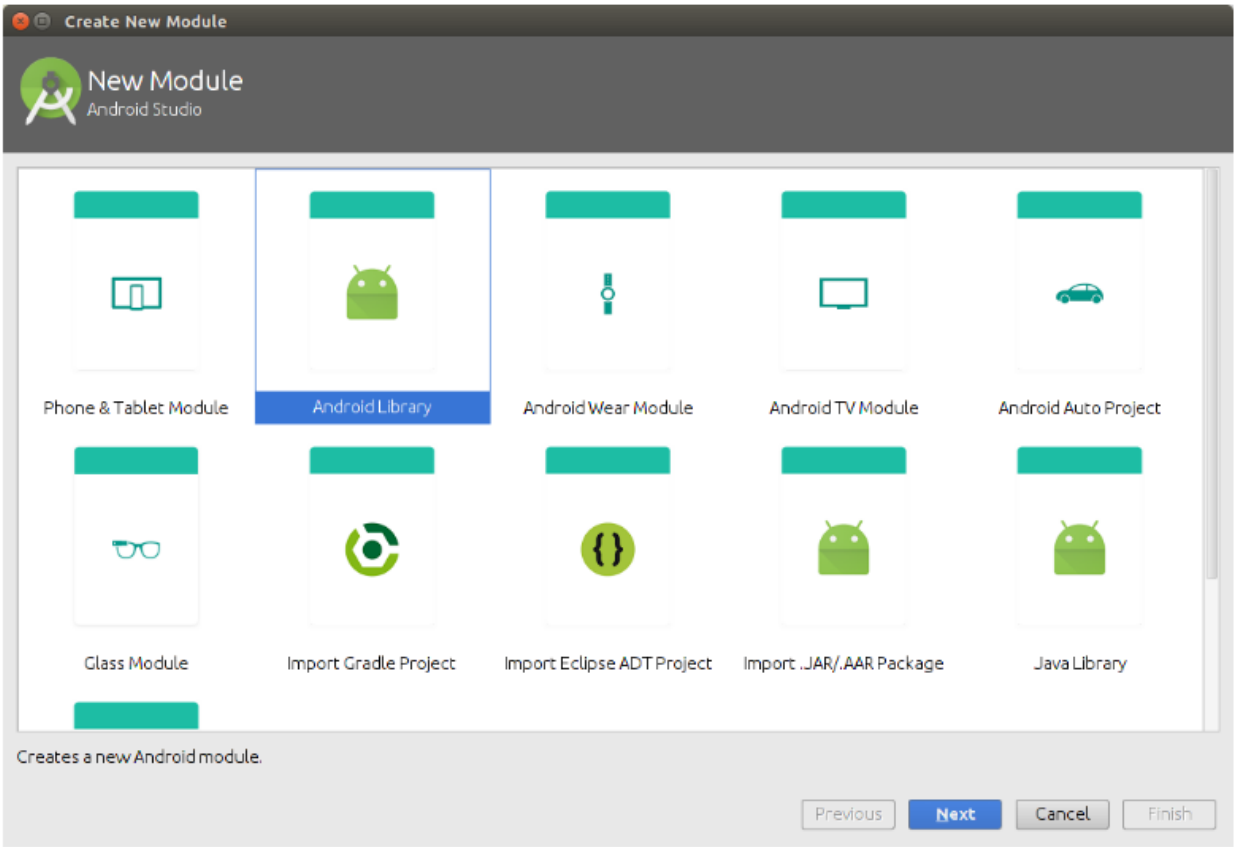
88146323 - 88446780 - 88146330

برای تمرین جاری لازم است پروژه ای که در مباحث قبلی ایجاد کرده و اسم پکیج (top level package) آن com.example.android.rssfeed می باشد را در اختیار داشته باشید (می توانید به مقاله ی آموزش fragment ها مراجعه نمایید). پروژه ی کتابخانه ای که ایجاد می کنیم علاوه بر data model (کلاس های ساخته شده از جداول دیتابیس)، یک تابع جهت بازیابی تعداد نمونه ها (instances) در بر خواهد داشت. کتابخانه ی مورد نظر اجازه ی دسترسی به داده ها و اطلاعات (ساختگی) RSS را می دهد. فایل RSS نیز صرفا یک فایل XML است که برای انتشار و بارگذاری اخبار و نوشته ها و پست های وبلاگ مورد استفاده قرار می گیرد.

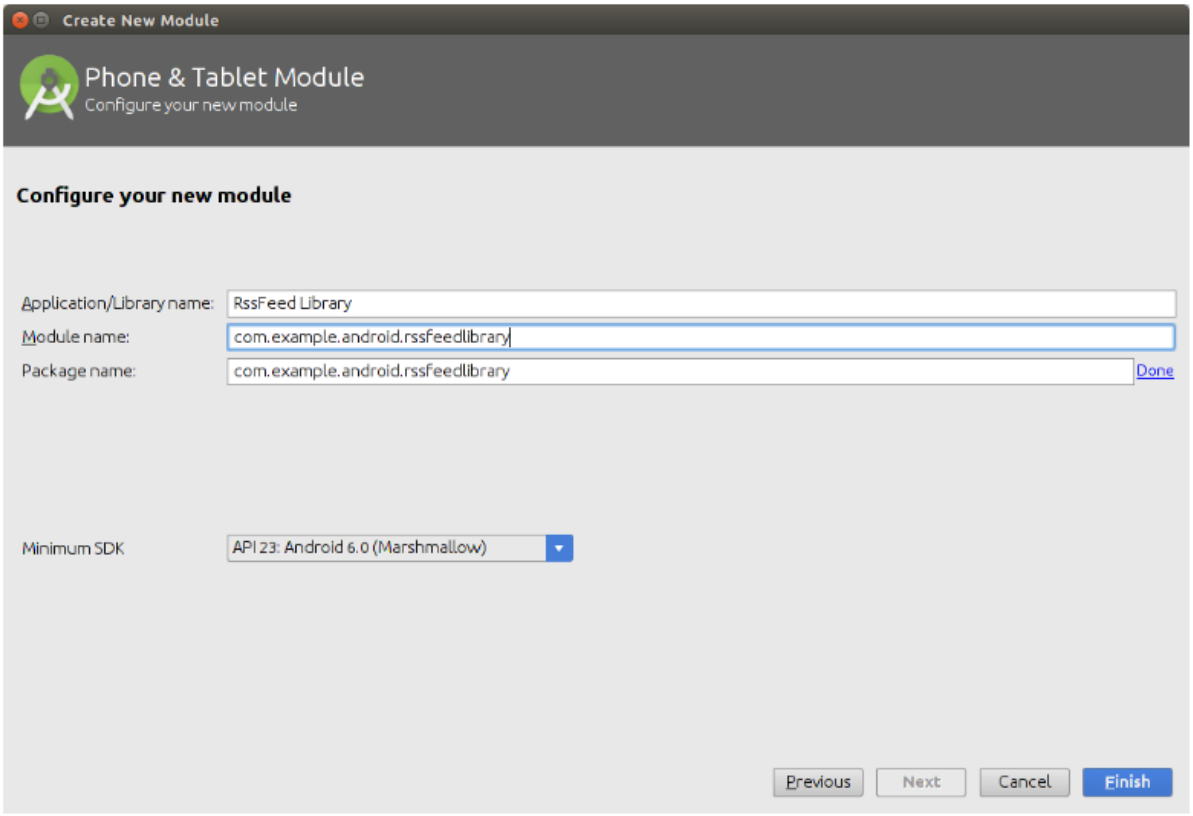
ایجاد یک کتابخانه ی جدید (library module)

در محیط برنامه نویسی Android Studio هر کتابخانه خود یک ماژول محسوب می شود. برای ایجاد یک ماژول یا کتابخانه ی جدید در محیط کاری نام برده، این مسیر را طی نمایید:

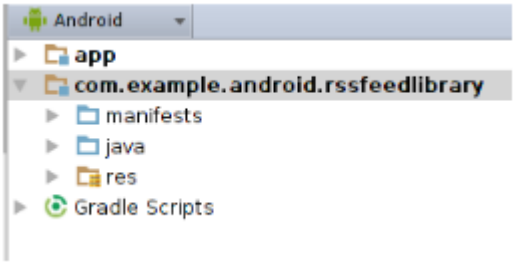
File ▸ New Module و سپس Android Library را انتخاب نمایید.



به عنوان اسم ماژول، com.example.android.rssfeedlibrary و به عنوان اسم کتابخانه Rssfeed Library را وارد نمایید.



زمانی که از شما درخواست می شود که قالب آماده یا template دلخواه را انتخاب نمایید، صریحا اعلان کنید که نمی خواهید activity ساخته شود. در نتیجه محیط کاری Android Studio یک ماژول دیگر برای شما به نمایش می گذارد.



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

ایجاد کلاس model (data model)

یک کلاس به نام RssItem جهت ذخیره ی داده های مربوط به محتوای RSS ایجاد نمایید.

حال توابع getter و setter، تابع سازنده (constructor) و یک متد ()toString در کلاس اعلان نمایید. خروجی بایستی ظاهری مشابه زیر داشته باشد:

```
package com.example.android.rssfeedlibrary;
public class RssItem {
	private String pubDate;
	private String description;
	private String link;
	private String title;
	public RssItem() {
	}
	public RssItem(String title, String link) {
		this.title = title;
		this.link = link;
	}
	public String getPubDate() {
		return pubDate;
	}
	public void setPubDate(String pubDate) {
		this.pubDate = pubDate;
	}
	public String getDescription() {
		return description;
	}
	public void setDescription(String description) {
		this.description = description;
	}
	public String getLink() {
		return link;
	}
	public void setLink(String link) {
		this.link = link;
	}
	public String getTitle() {
		return title;
	}
	public void setTitle(String title) {
		this.title = title;
	}
	@Override
	public String toString() {
		return "RssItem [title=" + title + "]";
	}
}
```

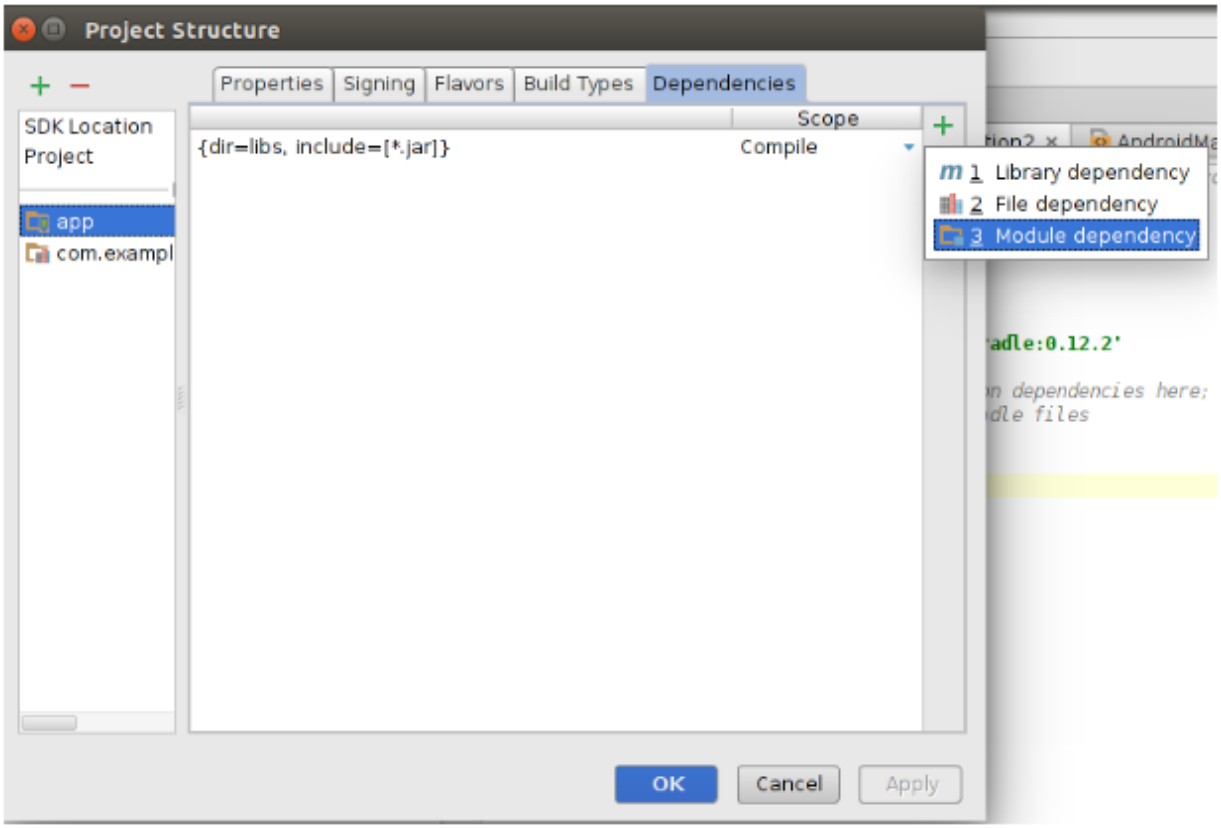
ایجاد کلاس های مورد نیاز (instances)

اکنون یک کلاس جدید به نام RssFeedProvider با یک متد static که لیستی از آبجکت های RssItem را در خروجی برمی گرداند، تعریف نمایید. این متد در حال حاضر تنها داده های آزمایشی (test data) را برمی گرداند.

```
package com.vogella.android.rssfeedlibrary;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
public class RssFeedProvider {
  public static List<RssItem> parse(String rssFeed) {
    List<RssItem> list = new ArrayList<>();
    Random r = new Random();
    // random number of item but at least 5
    Integer number = r.nextInt(10) + 5;
    for (int i = 0; i < number; i++) {
      // create sample data
      String s = String.valueOf(r.nextInt(1000));
      RssItem item = new RssItem("Summary " + s, "Description " + s);
      list.add(item);
    }
    return list;
  }
}
```

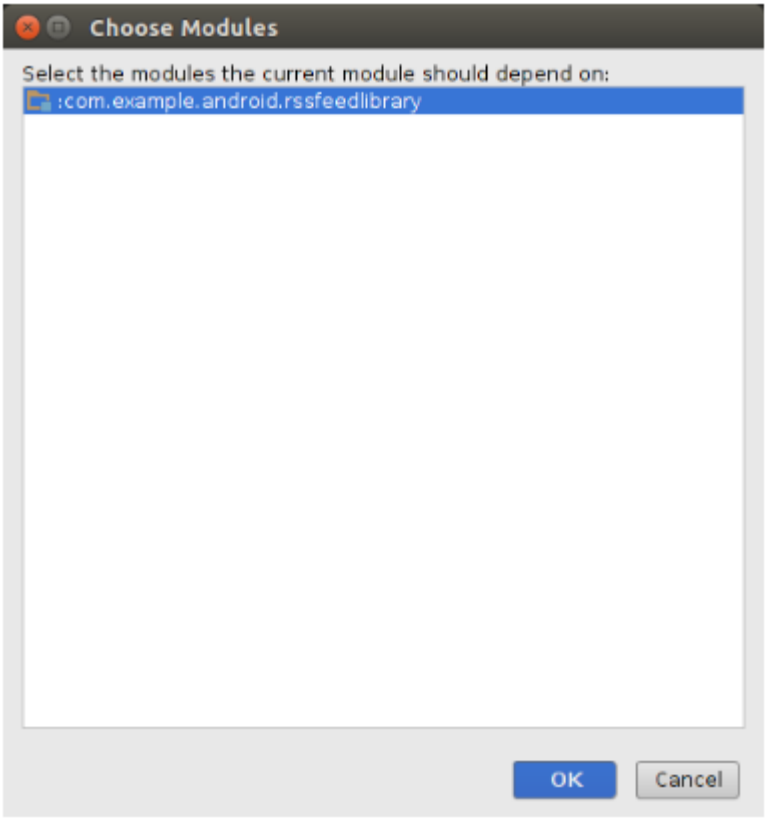
اضافه کردن کتابخانه به عنوان dependency به پروژه

به منظور استفاده از کتابخانه، کافی است آن را به عنوان dependency در بستر پروژه ی خود وارد کنید. برای این منظور مسیر رو به رو را طی نمایید: File ▸ Project Structure. پوشه ی app را انتخاب کنید. حال تب Dependencies را باز نموده و پس از کلیک بر روی علامت +، المان module dependencies را از فهرست انتخاب نمایید.



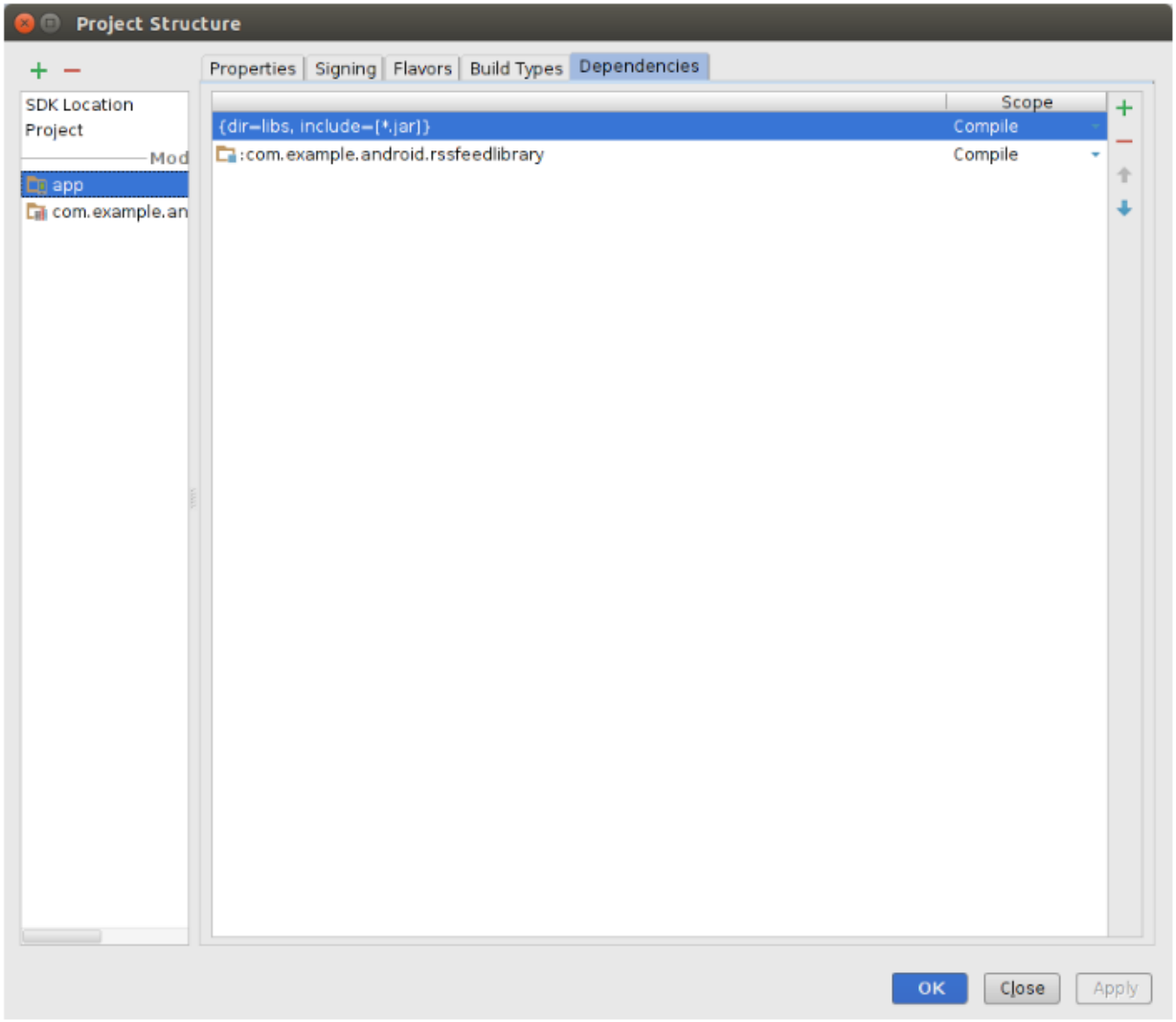
آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330



استفاده از پروژه کتابخانه جهت بروز رسانی داده های details fragment ها

متد updateDetail را جهت استفاده از کامپوننت نرم افزاری RssFeedProvider، در بدنه ی کلاس MyListFragment بروز رسانی نمایید. کدهای موجود در آن صرفا آزمایشی (test code) هستند.

```
public class MyListFragment extends Fragment {
..// everything as before
   // triggers update of the details fragment
   public void updateDetail(String uri) {  //
شده
      List<RssItem> list = RssFeedProvider
            .parse("http://www.vogella.com/article.rss");
      String itemListAsString = list.toString();
```

1) متد آپدیت

```
    listener.onRssItemSelected(itemListAsString);
}
```

تست اپلیکیشن و بررسی عملکرد صحیح آن

اپلیکیشن خود را اجرا کرده و مطمئن شوید که خروجی تابع ()toString (مقدار toString) از لیست RssItems در detail fragment نمایش داده می شود.

نکته: لازم به ذکر است که لیست، با توجه به منطق و کدی که در حال حاضر برای اپلیکیشن تنظیم شده، صرفا به صورت تصادفی ایجاد می شود.

بخش سوم :

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

معرفی کتابخانه های کارا و پرکاربرد اندروید / Android Library project / پروژه های کتابخانه ای اندروید و

مقاله ی حاضر تمامی کتابخانه های پرکاربرد را به همراه code repository های (انباری از کدهای آماده) کارا و محبوب اندروید معرفی می نماید.

پروژه های کتابخانه ای کارامد و پرکابرد اندروید

ویژگی متن باز و رایگان بودن (open source) اندروید سبب شده تا سایر توسعه دهندگان بتوانند به راحتی کتابخانه ها و ماژول های کارا و اضافی بر سازمان (جدا از خود چارچوب نرم افزاری اندروید) برای آن طراحی کرده و در اختیار همگان قرار دهند. مبحث جاری کتابخانه های open source قدرتمند و نمونه پروژه های مفید که برنامه نویس را در توسعه ی هر چه بهینه تر پروژه ی خود یاری می کند، فهرست می نماید.

لیست کتابخانه های پرکاربرد اندروید

- https://github.com/bumptech/glide - فریم ورک بارگذاری فایل های تصویری برای اندروید (Glide Image loading framework)
- http://www.vogella.com/tutorials/AndroidButterknife/article.html - فریم ورک نرم افزاری تزریق view و المان های
- رابط کاربری (view injection framework)
- https://github.com/square/otto - Otto - یک event bus که از جمله قابلیت های آن آسان سازی تعامل و تبادل داده بین کامپوننت های مختلف نرم افزاری اندروید می باشد.
- http://square.github.io/retrofit/ -Retrofit - یک کلاینت type-safe و نوع ایمن REST اندروید و جاوا
- http://code.google.com/p/achartengine/ - AChartEngine – موتور ترسیم chart و نمودار
- http://ormlite.com/ - جهت تولید دیتابیس برای اپلیکیشن های اندرویدی / ORM mapper
- http://greendao-orm.com/ GreenDAO - تعبیه شده توسط Markus Junginger، ORM Mapper

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

- Dagger2 - https://github.com/google/Dagger - فریم ورک تزریق نیازمندی و dependency injection برای پروژه های جاوا و اندروید
- http://code.google.com/p/android-scripting/ - این زمینه را برای زبان های اسکریپت نویسی فراهم می کند تا بر روی محیط اندروید اجرا شوند

Code repository و انباره نمونه کدها و پروژه ها

بهترین منبع برای (آشنایی با نحوه ی استفاده از توابع کتابخانه ای اندروید و API ها) نمونه کدهای اندروید، API demos می باشد.

Roman Nurik، یک توسعه دهنده در شرکت گوگل، نمونه کدهای خود را تحت آدرس زیر در اختیار علاقه مندان قرار می دهد: - http://code.google.com/p/romannurik-code/ . Roman Nurik's examples

Mark Murthy، مدرس دوره های برنامه نویسی اندروید، توسعه دهنده و نویسنده ی کتاب های برنامه نویسی، پروژه ها و نمونه کدهای پیشرفته ی خود را تحت آدرس https://github.com/commonsguy/cw-advandroid - Mark Murthy's advanced examples در اختیار توسعه دهندگان قرار می دهد.

بخش چهارم :

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

آموزش استفاده از کتابخانه ی Otto event bus در پروژه های اندرویدی

مبحث حاضر به شرح نحوه ی استفاده از Otto event bus در اپلیکیشن های اندروید می پردازد.

Otto یک پروژه ی متن باز و رایگان (open source) است که به کامپوننت ها اجازه می دهد تا مکانیزم event bus (واسط و مترجم پیغام بین فرستنده و گیرنده) را پیاده سازی کرده تا کامپوننت های بتوانند به event ها گوش داده (subscribe) و به آن ها پیغام ارسال کنند (publish).

Eventbus امکان برقراری ارتباط به سبک publish-subscribe را بین کامپوننت های نرم افزاری مختلف فراهم می سازد. بدین معنی که کامپوننت ها بدون اینکه لازم باشد به طور صریح به یکدیگر معرفی شده و اطلاعی از وجود هم داشته باشند، ارتباط برقرار کنند.

Otto یک شاخه از کتابخانه ی Guava event bus ساخت Google بوده و طوری بازطراحی شده که تا حد ممکن با Android سازگار باشد.

نصب کتابخانه

استفاده از این کتابخانه در جاوا یا اندروید بسیار ساده است، کافی است فایل JAR و تنها نیازمندی آن را از http://square.github.io/otto/ دانلود کرده و سپس آن را به متغیر classpath اپلیکیشن اضافه نمایید.

در صورت استفاده از Maven یا Gradle به عنوان سیستم کامپایل و build پروژه، شما می توانید به راحتی کتابخانه مورد نیاز (dependency) را به شناسه ی گروه (Group ID) com.squareup، شناسه ی artifact (otto) و ورژن 1.3.5 اضافه نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

چه زمانی باید از Otto استفاده کرد؟

Otto ابزار خوبی برای مهیا کردن امکان ارتباط بین activity و fragment های زیرمجموعه ی آن و یا تبادل داده بین activity و service می باشد.

چگونگی تنظیم Otto

توجه: آموزش حاضر فرض را بر این می گذارد که توسعه دهنده کتابخانه ی Otto را در یک اپلیکیشن اندرویدی مورد استفاده قرار می دهد، اگرچه امکان استفاده از آن در بستر پروژه های جاوایی نیز وجود دارد.

جهت استفاده از Otto، لازم است یک نمونه ی واحد (singleton instance) از کلاس Bus ایجاد کرده و سپس برای کامپوننت های نرم افزاری اندروید امکان دسترسی به آن را فراهم نمایید. عملیات مزبور در آبجکت Application پروژه ی اندرویدی شما انجام می شود.

```
public static Bus bus = new Bus(ThreadEnforcer.MAIN);
```

در مثال جاری، همان طور که مشاهده می کنید، مقدار ThreadEnforcer.MAIN به عنوان پارامتر به تابع سازنده ی کلاس Bus ارسال شده است. به واسطه ی این پارامتر، Otto، به سیستم اعلان می کند که event ها منحصرا باید از thread اصلی ارسال شوند. اگر مهم نیست که رخداد از کدام thread ارسال می شود، پارامتر ThreadEnforcer.ANY به عنوان آرگومان به تابع سازنده ی Bus پاس داده می شود.

نحوه ی register یا گوش دادن به event ها و unregister آن ها

جهت گوش دادن به رخدادها (event registration)، لازم است دستور @Subscribe را در بالای متد که با سطح دسترسی public تعریف شده و فقط یک پارامتر ورودی دارد، درج نمایید. پارامتر ارسالی به متد در واقع event key محسوب می شود، بدین معنی که چنانچه چنین نوع داده ای (data type) به واسطه ی event bus (واسط و مدیریت کننده ی رخدادها Otto) ارسال شود، در آن صورت متد مرتبط صدا زده می شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

event receiver ها (متدهایی که رخداد را دریافت می کنند) بایستی خود را به وسیله ی تابع register از کلاس Bus ثبت کرده و به رخداد مربوطه گوش فرا دهند.

```
// subscribe for string messages
@Subscribe
public void getMessage(String s) {
    Toast.makeText(this, s, Toast.LENGTH_LONG).show();
}
//subscribe for TestData messages
@Subscribe
public void getMessage(TestData data) {
    Toast.makeText(getActivity(), data.message, Toast.LENGTH_LONG).show();
}
//requires a registration e.g. in the onCreate method
bus.register(this);
```

جهت unregister کردن و قطع فرایند گوش دادن به event ها، کافی است متد ()unregister را فراخوانی نمایید.

نحوه ی ارسال event ها

برای ارسال رخداد نیازی به register در event bus نیست. کافی است متد ()post از کلاس Bus را فراخوانی نمایید (متد post را بر روی آبجکت ساخته شده از کلاس Bus صدا بزنید).

```
// post a string object
bus.post("Hello");
// example data to post
public class TestData {
    public String message;
}
// post this data
bus.post(new TestData().message="Hello from the activity");
```

چگونه کامپوننت های نرم افزاری جدید می توانند آخرین event را دریافت کنند؟

در صورتی که کامپوننت های جدید، نظیر یک fragment که به صورت dynamic و در زمان اجرا ایجاد شده، داده های مربوط به event را در طول فرایند ایجاد دریافت کنند، کامپوننت های نرم افزاری می توانند از طریق دستور @Produce خود را به عنوان producer یا تولید کننده ی داده های رخداد معرفی (register) کنند.

event receiver ها (کامپوننت هایی که می خواهند به event گوش داده و قرار است داده های مربوط به رخداد را دریافت کنند) بایستی متد register را از کلاس Bus فراخوانی نمایند.

```
@Produce
public String produceEvent() {
    return "Starting up";
}
```

تمرین: استفاده ی کاربردی از کتابخانه ی Otto event bus

مثال حاضر کتابخانه ی Otto را به صورت کاربردی در یک پروژه ی اندرویدی مورد استفاده قرار می دهد. هرچند این کتابخانه در اپلیکیشن های جاوا نیز قابل فراخوانی و استفاده می باشد.

یک پروژه ی اندرویدی به نام com.vogella.java.library.otto، بر اساس قالب آماده ی Empty Activity with Fragment (template) ایجاد نمایید.

چنانچه برای برنامه نویسی از محیط کاری Eclipse استفاده می کنید، در آن صورت فایل Otto JAR را دانلود کرده و سپس آن را به build path پروژه ی خود اضافه نمایید.

در صورت استفاده از Gradle، کافی است کتابخانه (dependency) مورد نیاز را به فایل build.gradle اضافه نمایید.

```
apply plugin: 'com.android.application'
android {
   compileSdkVersion 22
   buildToolsVersion "21.1.2"
   defaultConfig {
      applicationId "com.example.android.rssreader"
      minSdkVersion 22
      targetSdkVersion 22
      versionCode 1
      versionName "1.0"
   }
   buildTypes {
      release {
         minifyEnabled false
         proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
      }
   }
}
dependencies {
   compile fileTree(include: ['*.jar'], dir: 'libs')
   compile project(':com.example.android.rssfeedlibrary')
```

```
    compile 'com.squareup:otto:1.3.5'
    compile 'org.achartengine:achartengine:1.2.0'
}
```

محتوای فایل تنظیم ظاهر اپلیکیشن، activity_main.xml layout، بایستی ظاهری مشابه زیر داشته باشد.

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.vogella.android.otto.MainActivity"
    tools:ignore="MergeRootFrame" />
```

فایل تنظیم کننده ی ظاهر fragment، fragment_main.xml، بایستی محتوای مشابه زیر داشته باشد.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.vogella.android.otto.MainActivity$PlaceholderFragment" >
    <Button
        android:id="@+id/fragmentbutton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:text="Send event from fragment" />
</RelativeLayout>
```

پس از تعریف کردن fragment، آن را به صورت dynamic و داخل کلاس جاوایی به activity اضافه نمایید. سپس یک رخداد به ترتیب از fragment به activity و بالعکس ارسال کنید. در هر دو سناریو پس از دریافت رخداد یک پیغام Toast برای کاربر به نمایش بگذارید.

```
package com.vogella.android.otto;
import android.app.Activity;
import android.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
```

```
import android.widget.Toast;
import com.squareup.otto.Bus;
import com.squareup.otto.Produce;
import com.squareup.otto.Subscribe;
import com.squareup.otto.ThreadEnforcer;
import library.java.vogella.com.otto.R;
public class MainActivity extends Activity {
  public static Bus bus;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    if (savedInstanceState == null) {
      getFragmentManager().beginTransaction().add(R.id.container, new PlaceholderFragment()).commit();
    }
    bus = new Bus(ThreadEnforcer.MAIN);
    bus.register(this);
  }
  @Subscribe
  public void getMessage(String s) {
    Toast.makeText(this, s, Toast.LENGTH_LONG).show();
  }
  @Override
  public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
  }
  @Override
  public boolean onOptionsItemSelected(MenuItem item) {
    int id = item.getItemId();
    if (id == R.id.action_settings) {
      TestData t = new TestData();
      t.message="Hello from the activity";
      bus.post(t);
      return true;
    }
    return super.onOptionsItemSelected(item);
  }
  public class TestData {
    public String message;
  }
  /**
   * A placeholder fragment containing a simple view.
   */
  public static class PlaceholderFragment extends Fragment {
    public PlaceholderFragment() {
    }
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
      View rootView = inflater.inflate(R.layout.fragment_main, container, false);
      View button = rootView.findViewById(R.id.fragmentbutton);
      button.setOnClickListener(new View.OnClickListener() {
```

```
            @Override
            public void onClick(View v) {
                bus.post("Hello from the Fragment");
            }
        });
        bus.register(this);
        return rootView;
    }
    @Subscribe
    public void getMessage(MainActivity.TestData data) {
        Toast.makeText(getActivity(), data.message, Toast.LENGTH_LONG).show();
    }
  }
  @Produce
  public String produceEvent() {
    return "Starting up";
  }
}
```

توجه: مثال حاضر به طور عمد این گونه ساده طراحی شده است. در یک اپلیکیشن واقعی و تمام عیار اندروید، توسعه دهنده یک نمونه ی واحد (singleton) از کلاس Bus در آبجکت Application ایجاد کرده و fragment را داخل فایل مجزای خود قرار می دهد.

فصل دوازدهم

سر فصل های خاص

برنامه نویسی اندروید

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

بخش اول

آموزش مدیریت فایل های صوتی در اندروید / پیاده سازی توابع کتابخانه ی Media API جهت پخش و ضبط صدا

آموزش حاضر نحوه ی استفاده از توابع کتابخانه ای media API جهت پخش و ضبط صدا در محیط اندروید را شرح می دهد. پروژه ی این مبحث در محیط کاری Eclipse 3.7، با ویرایش 1.6 زبان java و بر اساس ورژن 4.0.3 سیستم عامل اندروید نوشته می شود.

مدیریت صدا در محیط اندروید (Android Sound)

پخش صدا

اندروید دو مجموعه توابع کتابخانه ای یا API برای پخش صدا در اختیار توسعه دهندگان قرار می دهد. اولین مجموعه از این توابع کلاس SoundPool و دیگری کلاس MediaPlayer می باشد.

SoundPool را می توان برای کلیپ های کوچک صوتی مورد استفاده قرار داد. این کلاس قادر است صداها را تکرار کرده و همچنین به طور همزمان چندین صدا را بخش کند. فایل های صوتی که توسط کلاس SoundPool پخش می شوند نباید از نظر حجم از مرز 1 مگابایت فراتر روند.

لازم به ذکر است که SoundPool فایل صوتی مربوطه را به صورت ناهمزمان (asynchronously) بارگذاری می کند. از ورژن 8 کتابخانه های اندروید (API 8) این امکان وجود دارد که از اتمام بارگذاری فایل و اطلاعات مورد نظر اطمینان حاصل نمایید. برای این منظور کافی است از OnLoadCompleteListener استفاده نمایید.

سیستم اندروید برای اهداف مختلف، از audio stream های (جریانی از بیت ها که حامل اطلاعات صوتی بوده و یک فایل صوتی را تشکیل می دهند) مختلف پشتیبانی می کند. می توان دکمه ی افزایش و کاهش میزان صدای دستگاه را طوری تنظیم کرد که برای مثال در حین

دریافت تماس، میزان صدای تماس گیرنده را کاهش یا افزایش دهد. جهت تنظیم دکمه برای کنترل media stream و اطلاعات صوتی (جریانی از بیت ها و اطلاعات که تشکیل دهنده ی فایل صوتی هستند)، کافی است audio type را در اپلیکیشن خود مقداری دهی و مشخص نمایید.

```
context.setVolumeControlStream(AudioManager.STREAM_MUSIC);
```

کلاس MediaPlayer گزینه ی بهتری جهت پخش موسیقی های طولانی تر و فایل های تصویری حجیم نظیر فیلم ها می باشد.

کلاس MediaRecorder

با استفاده از کلاس MediaRecorder می توان به راحتی اطلاعات مربوط به صدا و تصویر را ضبط کرد. جهت استفاده از کلاس نام برده لازم است دستگاه منبع (source device) و فرمت را مشخص نمایید.

افزودن Media جدید به Media library

این امکان برای شما وجود دارد که Media (فایل و داده های صوتی) جدید به media library (مجموعه فایل های صوتی) اندروید اضافه نمایید. برای نیل به این هدف کافی است یک آبجکت Intent ایجاد نموده و سپس از طریق آن media application (اپلیکیشن پخش و ضبط صدا) مستقر بر روی دستگاه، به آن اطلاع دهید که محتوای جدید در دسترس می باشد. کد زیر نحوه ی پیاده سازی آن را به نمایش می گذارد.

```
// add new file to your media library
ContentValues values = new ContentValues(4);
long current = System.currentTimeMillis();
values.put(MediaStore.Audio.Media.TITLE, "audio" + audiofile.getName());
values.put(MediaStore.Audio.Media.DATE_ADDED, (int) (current / 1000));
values.put(MediaStore.Audio.Media.MIME_TYPE, "audio/3gpp");
values.put(MediaStore.Audio.Media.DATA, audiofile.getAbsolutePath());
ContentResolver contentResolver = getContentResolver();
Uri base = MediaStore.Audio.Media.EXTERNAL_CONTENT_URI;
Uri newUri = contentResolver.insert(base, values);
// Notifiy the media application on the device
sendBroadcast(new Intent(Intent.ACTION_MEDIA_SCANNER_SCAN_FILE, newUri));
```

فرمت های مورد پشتیبانی

اندروید از فرمت های مختلف پشتیبانی می کند. جهت مشاهده ی فرمت های مورد پشتیبانی اندروید می توانید به آدرس http://developer.android.com/guide/appendix/media-formats.html مراجعه نمایید.

آموزش کاربردی: پخش صدا به وسیله ی کلاس SoundPool

در بخش حاضر یک اپلیکیشن خواهیم نوشت که به مجرد برخورد انگشت کاربر با سطح نمایشگر، یک آهنگ یا صدا را بخش خواهد کرد. ابتدا یک پروژه و activity جدید اندروید به ترتیب به نام های "de.vogella.android.soundpool" و PlaySound ایجاد نمایید.

محتوای فایل main.xml که ظاهر اپلیکیشن را تعریف می کند، به صورت زیر ویرایش نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
   <TextView
     android:id="@+id/textView1"
     android:layout_width="match_parent"
     android:layout_height="match_parent"
     android:text="Click on the screen to start playing" >
   </TextView>
</LinearLayout>
```

فایل صوتی مورد نظر را دانلود نمایید. در صورت تمایل می توانید فایل صوتی آزمایشی خود را از آدرس http://hamsterrepublic.com/ohrrpgce/Free_Sound_Effects.html بارگیری کنید. این سایت تعداد زیادی فایل صوتی کم حجم به صورت رایگان در اختیار شما قرار می دهد. فایل صوتی دریافت شده را داخل پوشه ی "res/raw" تحت نام "sound1.ogg" ذخیره نمایید. با انتخاب این پسوند برای نام فایل شما به سیستم اعلان می کنید که یک فایل با فرمت ogg از اینترنت بارگیری کرده اید.

کلاس activity خود را به صورت زیر پیاده سازی نمایید.

```
package de.vogella.android.soundpool;
import android.app.Activity;
```

```
import android.media.AudioManager;
import android.media.SoundPool;
import android.media.SoundPool.OnLoadCompleteListener;
import android.os.Bundle;
import android.util.Log;
import android.view.MotionEvent;
import android.view.View;
import android.view.View.OnTouchListener;
public class PlaySound extends Activity implements OnTouchListener {
	private SoundPool soundPool;
	private int soundID;
	boolean loaded = false;
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
		View view = findViewById(R.id.textView1);
		view.setOnTouchListener(this);
		// Set the hardware buttons to control the music
		this.setVolumeControlStream(AudioManager.STREAM_MUSIC);
		// Load the sound
		soundPool = new SoundPool(10, AudioManager.STREAM_MUSIC, 0);
		soundPool.setOnLoadCompleteListener(new OnLoadCompleteListener() {
			@Override
			public void onLoadComplete(SoundPool soundPool, int sampleId,
					int status) {
				loaded = true;
			}
		});
		soundID = soundPool.load(this, R.raw.sound1, 1);

	}
	@Override
	public boolean onTouch(View v, MotionEvent event) {
		if (event.getAction() == MotionEvent.ACTION_DOWN) {
			// Getting the user sound settings
			AudioManager audioManager = (AudioManager) getSystemService(AUDIO_SERVICE);
			float actualVolume = (float) audioManager
					.getStreamVolume(AudioManager.STREAM_MUSIC);
			float maxVolume = (float) audioManager
					.getStreamMaxVolume(AudioManager.STREAM_MUSIC);
			float volume = actualVolume / maxVolume;
			// Is the sound loaded already?
			if (loaded) {
				soundPool.play(soundID, volume, volume, 1, 0, 1f);
				Log.e("Test", "Played sound");
			}
		}
		return false;
	}
}
```

حال زمانی که کاربر سطح نمایشگر، activity اپلیکیشن، را لمس می کند، فایل صوتی پخش خواهد شد. لازم به ذکر است که صدا بر اساس تنظیمات صوتی جاری پخش خواهد شد.

آموزش کاربردی: ضبط صدا (media) با استفاده از کلاس MediaRecorder

در بخش آموزشی حاضر یک اپلیکیشن طراحی می کنیم که به مجرد برخورد انگشت کاربر با سطح نمایشگر، یک فایل صوتی را پخش خواهد می کند. یک پروژه و activity اندروید به ترتیب به نام های "de.vogella.android.media.soundrecording" و RecordSound ایجاد نمایید.

مجوز درج اطلاعات در حافظه خارجی (SD) و ضبط اطلاعات صوتی را در فایل تنظیمات اپلیکیشن AndroidManifest.xml اعلان نمایید.

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
 <uses-permission android:name="android.permission.RECORD_AUDIO" />
```

محتوای فایل main.xml را به صورت زیر اصلاح نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="horizontal" >
  <Button
    android:id="@+id/start"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Start Recording"
    android:onClick="startRecording" />
  <Button
    android:id="@+id/stop"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Stop Recording"
    android:enabled="false"
     android:onClick="stopRecording"
    />
</LinearLayout>
```

کد کلاس activity اپلیکیشن را نیز به صورت زیر ویرایش نمایید.

```
package de.vogella.android.media.soundrecording;
import java.io.File;
import java.io.IOException;
import android.app.Activity;
import android.content.ContentResolver;
import android.content.ContentValues;
import android.content.Intent;
```

```
import android.media.MediaRecorder;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.provider.MediaStore;
import android.util.Log;
import android.view.View;
import android.widget.Toast;
public class SoundRecordingActivity extends Activity {
    MediaRecorder recorder;
    File audiofile = null;
    private static final String TAG = "SoundRecordingActivity";
    private View startButton;
    private View stopButton;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        startButton = findViewById(R.id.start);
        stopButton = findViewById(R.id.stop);
    }
    public void startRecording(View view) throws IOException {
        startButton.setEnabled(false);
        stopButton.setEnabled(true);
        File sampleDir = Environment.getExternalStorageDirectory();
        try {
            audiofile = File.createTempFile("sound", ".3gp", sampleDir);
        } catch (IOException e) {
            Log.e(TAG, "sdcard access error");
            return;
        }
        recorder = new MediaRecorder();
        recorder.setAudioSource(MediaRecorder.AudioSource.MIC);
        recorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
        recorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);
        recorder.setOutputFile(audiofile.getAbsolutePath());
        recorder.prepare();
        recorder.start();
    }
    public void stopRecording(View view) {
        startButton.setEnabled(true);
        stopButton.setEnabled(false);
        recorder.stop();
        recorder.release();
        addRecordingToMediaLibrary();
    }
    protected void addRecordingToMediaLibrary() {
        ContentValues values = new ContentValues(4);
        long current = System.currentTimeMillis();
        values.put(MediaStore.Audio.Media.TITLE, "audio" + audiofile.getName());
        values.put(MediaStore.Audio.Media.DATE_ADDED, (int) (current / 1000));
        values.put(MediaStore.Audio.Media.MIME_TYPE, "audio/3gpp");
```

```
            values.put(MediaStore.Audio.Media.DATA, audiofile.getAbsolutePath());
            ContentResolver contentResolver = getContentResolver();
            Uri base = MediaStore.Audio.Media.EXTERNAL_CONTENT_URI;
            Uri newUri = contentResolver.insert(base, values);
            sendBroadcast(new Intent(Intent.ACTION_MEDIA_SCANNER_SCAN_FILE, newUri));
            Toast.makeText(this, "Added File " + newUri, Toast.LENGTH_LONG).show();
        }
}
```

حال زمانی که کاربر دکمه ی "Start Recording" را فشار می دهد، دستگاه شروع به ضبط صدا می نماید. با فشردن دکمه ی stop recording، می بینید که فرایند ضبط صدا متوقف شده و اطلاعات ذخیره شده متعاقبا به media library (مجموعه فایل های صوتی) شما اضافه می گردد.

بخش دوم

نکات آموزشی در خصوص دریافت و کامپایل کدهای برنامه اندرویدی (source code)

آموزش حاضر به شرح نحوه ی دریافت و ترجمه ی کدهای اپلیکیشن های اندروید و نیز دانلود source code مجموعه ابزار ساخت و توسعه ی اپلیکیشن های اندرویدی (ADT) که افزونه ها و plug in های محیط کاری Eclipse هستند، می پردازد. این مبحث بر اساس Ubuntu می باشد.

دریافت کد برنامه (Android source code)

جهت دسترسی به کد برنامه ی های اندروید شما به دو ابزار Git و repo دارید.

نصب ابزار مورد نیاز

ابتدا لازم است ابزار خط فرمان Git (command line) را نصب نمایید. برای این منظور می بایست به آموزش مقدمه ای بر Git و نصب آن مراجعه نمایید.

علاوه بر ابزار نام برده، لازم است ابزار دیگری به نام repo را نصب نمایید. برای این منظور کافی است دستورات زیر را اجرا نمایید.

```
# assumes that you have a local directory called bin
# in your home folder
# download the tool from Google
curl http://commondatastorage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
# make it executable
chmod a+x ~/bin/repo
```

کپی کردن کد اصلی اپلیکیشن (source code cloning)

پس از نصب ابزار repo، شما می توانید کد اپلیکیشن اندرویدی را با فراخوانی دستور زیر دریافت نمایید.

```
// To get the current master
repo init -u git://android.git.kernel.org/platform/manifest.git ; repo sync ;
```

دستور "git branch -a" به شما تمامی ویرایش های اندروید از جمله froyo و gingerbread را نمایش می دهد. می توانید با درج و اجرای دستور "git checkout branch_name" تمامی ورژن های اندروید را مشاهده و بررسی نمایید.

جهت ترجمه و کامپایل کد دستور زیر را فراخوانی کنید.

```
make
```

ابزار ساخت و توسعه ی اپلیکیشن های اندرویدی (ADT)

ابزار ADT

پروژه ی ADT ابزار و تجهیزات نرم افزاری ویژه ی دو محیط برنامه نویسی Eclipse و IntellJ فراهم می کند که به واسطه ی آن توسعه دهنده می تواند پروژه و برنامه های اندرویدی دلخواه خود را بنویسد.

نکته: می توانید برای مشاهده ی اطلاعات و مستندات آموزشی مربوطه به آدرس http://tools.android.com/ مراجعه نمایید.

کمک به توسعه ی پروژه ADT

فرایند کمک به توسعه ی ADT بر اساس Gerrit می باشد، اما از ابزار دیگری به نام repo استفاده می کند. پروسه ی کمک به توسعه ی پروژه ی مزبور در آدرس http://tools.android.com/contributing شرح داده است.

نحوه ی ساخت ابزار

برای آموزش در خصوص نحوه ی ساخت تجهیزات و ابزار توسعه ADT ویژه ی محیط برنامه نویسی Eclipse می توانید به آدرس زیر مراجعه نمایید:

http://tools.android.com/build/eclipse

بخش سوم

آموزش ساختار داخلی و معماری سیستم اندروید (Android Architecture)

مقاله ی پیشرو ساختار درونی اندروید را مورد بررسی قرار داده و توضیحاتی در خصوص معماری آن در اختیار شما قرار می دهد.

برنامه نویسی و توسعه اپلیکیشن برای Android

سیستم عامل اندروید

اندروید یک سیستم عامل مبتنی بر نسخه ی اصلاح و بروز رسانی شده ی Linux 2.6 و توابع برنامه نویسی و اینترفیس Java می باشد. جهت اجرای بهینه ی اندروید بر روی دستگاه موبایل، تیم توسعه دهندگان اندروید تعدای کتابخانه و درایور ویرایش شده به آن اضافه کرده اند.

در واقع علاوه بر خود دستگاه مجازی جاوا (Java virtual machine) یا همان DVM، ابزار مختلفی دیگری نظیر کامپایلر، خطایاب کد (debugger) و نرم افزار شبیه ساز (Emulator) در اختیار توسعه دهنده قرار داده شده است.

Android در اصل ساخت تیم نرم افزاری به نام Open Handset Alliance بوده و هم اکنون توسط شرکت گوگل رهبری می شود.

اندروید از یک دستگاه مجازی ویژه به نام Dalvik استفاده می کند. این دستگاه از bytecode های (فرمت کامپایل شده برنامه ی جاوایی) به خصوصی استفاده می کند، از این جهت شما نمی توانید انتظار اجرای bytecode متعارف جاوا را بر روی محیط اندروید داشته باشید (bytecode مجموعه ای از دستورات است که برای اجرای موثر توسط مفسر، طراحی شده است. بر خلاف کد منبع که برای انسان قابل فهم است، bytecode که کد قابل حمل نیز خوانده می شود از یک سری کدهای عددی فشرده، آدرس های عددی و ثابت ها تشکیل شده است).

سیستم اندروید ابزاری به نام dx در اختیار توسعه دهنده قرار می دهد که فایل های تشکیل شده از کلاس های جاوا (java class file) را به فایل هایی با پسوند dex (Dalvik Executable یا فایل های اجرایی dalvik) تبدیل می کند. برنامه ی aapt (ابزار بسته بندی فایل های asset و محتوا در اندروید) اپلیکیشن های اندروید و فایل های آن ها را داخل فایلی میزبانی به نام Android Package با پسوند apk. می گنجاند. جهت تسهیل پروسه ی توسعه ی پروژه و نرم افزارهای اندرویدی، شرکت گوگل ابزاری تخصصی به نام ADT (مجموعه ابزار ساخت و توسعه اپلیکیشن های اندرویدی) ویژه ی محیط کاری Eclipse ارائه می دهد. ابزار نام برده فرایند تبدیل از کلاس های جاوایی به فایل های dex را به صورت خودکار انجام داده و خود فایل apk را در زمان آماده سازی و نصب پروژه (deployment) ایجاد می نماید.

در ادامه ی مبحث حاضر به شرح سیستم runtime و بستر اجرای اندروید خواهیم پرداخت.

سیستم داخلی اندروید (internals)

در زمان اجرا و راه اندازی سیستم اندروید، هسته یا kernel لینوکس ابتدا فرایند "init" را فراخوانی می کند. init دو فایل "/init.rc" و "init.device.rc" را به ترتیب می خواند. فایل دوم بر اساس نوع دستگاه متفاوت بوده و به بیانی دقیق تر با توجه به هر دستگاه (device specific) ویژه و اختصاصی می باشد. فایل نام برده در محیط دستگاه مجازی تحت عنوان "init.goldfish.rc" شناخته می شود.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

فایل init.rc، به وسیله ی ابزاری به نام "/system/bin/app_process"، فرایند "Zygote" (یک کامپوننت نرم افزاری و سرویسی از سیستم عامل اندروید که پدر تمامی فرایندهای اپلیکیشن های اندرویدی می باشد) را اجرا می کند. فرایند مزبور کلاس های اصلی جاوا را در حافظه بارگذاری نموده و پروسه ی پردازش اولیه ی آن ها را آغاز می کند. این کلاس ها برای اپلیکیشن اندرویدی مورد نظر بارها قابل استفاده بوده (reusable) و همین امر نیز اجرای آن را به مراتب تسریع می بخشد. پس از اتمام عملیات اولیه، فرایند ذکر شده (zygote) به socket گوش فرا داده و منتظر دریافت درخواست ها می شود.

در اندروید درایور ویژه ای به نام Binder وجود دارد که زمینه ی ارتباطات بین فرایندی بهینه را فراهم می آورد به گونه ای که ارجاعات و اشاره گرهایی به آبجکت ها بین فرایندها رد و بدل می شوند و خود آبجکت ها در Shared Memory نگهداری می شوند. در نتیجه ی این امر، از آنجایی که اکنون داده ها و اطلاعات کمتری می بایست بین فرایندها رد و بدل شود، ارتباط بهینه سازی می گردد.

برخلاف سایر سیستم های مبتنی بر لینوکس، اندروید فضای جایگزین و اضافی (swap space) در اختیار ندارد، به همین جهت میزان حافظه ی قابل استفاده برای سیستم، محدود به همان حافظه ی مستقر بر روی دستگاه می باشد.

اندروید از کتابخانه ی ویژه ی C به نام "Bionic" به جای Glibc بهره می گیرد. این کتابخانه با Glibc سازگار نبوده و به حافظه کمتری احتیاج دارد. در واقع Bionic دربردارنده ی یک نوع خاص از پیاده سازی thread می باشد که میزان حافظه ی مورد استفاده ی هر thread را بهینه ساخته و زمان اجرای thread های جدید را کاهش می دهد.

از ویرایش 2.3 به بعد، سیستم اندروید از Ext4، سیستم فایل متعارف Linux، استفاده می کند. پیش از ویرایش 2.3، سیستم فایل مورد استفاده ی اندروید YAFFS بود. برخی از ارائه دهندگان اپلیکیشن و خدمات نرم افزاری (vendor) سیسم فایل اختصاصی و دلخواه خود را جایگزین سیستم فایل متعارف اندروید می کنند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

تمامی اپلیکیشن ها user application data اختصاصی خود را دارند و به صورت پیش فرض تنها از خود اپلیکیشن می توان به این فایل ها دسترسی داشت. به منظور به اشتراک گذاری داده ها بین اپلیکیشن ها، شما می توانید اطلاعات را در فایل های ذخیره کنید که برای اپلیکیشن های دیگر قابل دسترسی می باشد. همچنین به عنوان روشی جایگزین، اپلیکیشن ها می توانند content provider اختصاصی خود را تعریف کرده و جهت تامین داده های مورد نیاز خود از این کامپوننت نرم افزاری استفاده کنند. اپلیکیشن هایی که با کلید یا گواهی نامه (certificate) یکسان امضای دیجیتالی شده باشند، می توانند با اعلان المان "android:sharedUserId" در فایل تنظیمات "AndroidManifest.xml"، یک شناسه ی مشترک داشته باشند. اپلیکیشن هایی که شناسه ی مشترک دارند، همگی در بستر فرایند یکسان اجرا شده و شناسه ی کاربری واحدی دارند. از اینرو می توانند به اطلاعات و فایل های هم دسترسی داشته باشند.

فایل apk. اپلیکیشنی که بر روی دستگاه نصب شده در پوشه ی "data/app" جایگذاری می شود. شما می توانید با استفاده از file manager یا دستور adb pull / push، اپلیکیشن را بارگیری کرده یا آن را بر روی دستگاه نصب کنید. فایل نصبی (.apk) اپلیکیشن های پولی یا اپلیکیشن هایی از فروشگاه های مجازی که قابلیت "Copy Protection" آن ها فعال شده، در پوشه ی "data/app-private" قرار دارند و به طور پیش فرض برای کاربر قابل دسترسی نمی باشند.

به منظور محافظت هر چه بیشتر اپلیکیشن در مقابل کپی برداری غیرمجاز، می توانید از کتابخانه ی "Licence Verification Library" استفاده نمایید. این کتابخانه بررسی می کند آیا اپلیکیشن پولی مورد نظر از Android Market به طور قانونی خریداری شده یا خیر.

در حال حاضر حداکثر حجم اپلکیشن های اندرویدی بین 16 تا 24 مگابایت متغیر می باشد. اینکه حجم واقعی اپلیکیشن چقدر است، منحصرا در زمان کامپایل اپلیکیشن مشخص می شود.

بخش چهارم

آموزش C2DM (سرویس اطلاع رسانی Cloud to device messaging)

آموزش حاضر نحوه ی ارسال اطلاعات (push) از یک server (سرویس دهنده) به دستگاه مبتنی بر Google را شرح می دهد. پروژه ها و مثال های این مبحث در محیط کاری Eclipse 3.7، با استفاده از زبان Java نوشته شده و مبتنی بر Android 2.3 می باشد.

Cloud to device messaging

مقایسه ی poll و push

امروزه اغلب اپلیکیشن های تحت موبایل اطلاعاتی را از اینترنت دریافت می کنند. یک روش برای بروز رسانی اپلیکیشن و دریافت داده های جدید این است که اپلیکیشن در فواصل زمانی معین از سرویس دهنده برای داده های جدید پرس و جو کند که در اصطلاح به آن polling می گویند. چنانچه داده های جدیدی برای دانلود موجود نباشد، این روش صرفا پهنای باند اضافی و باتری دستگاه را مصرف کرده است.

روش دیگری که می توان مورد بررسی قرار داد این است که هر زمان داده های جدیدی موجود بود، سرویس دهنده با اپلیکیشن تحت موبایل ارتباط برقرار کرده و آن را از وجود داده های جدید آگاه کند که در اصطلاح به آن Pushing می گویند. در شرایطی که قرار نیست به طور مداوم داده های جدید اضافه شود، توصیه می شود که از push استفاده نمایید.

سرویس ارسال اطلاعات از server به اپلیکیشن های اندرویدی / سرویس Cloud

شما می توانید به واسطه ی Google play service (سرویس ارائه دهنده ی خدمات اندروید) اطلاعیه هایی را به اپلیکیشن خود ارسال کنید. برای این منظور، لازم است Google Cloud Messaging for Android را از طریق Google API Console فعال نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

در سرویس C2DM سه طرف شرکت دارند: 1. سرویس دهنده ی اپلیکیشن (app server) که پیغام ها و اطلاعاتی را به دستگاه اندروید ارسال (push) می کند 2. سرورهای Google C2DM 3. اپلیکیشن تحت موبایل اندرویدی. برنامه ی مستقر در سرویس دهنده ی اپلیکیشن می تواند به هر زبانی نوشته شده باشد (Java، PHP، Python و ...).

زمانی که سرویس دهنده ی اپلیکیشن (app server) لازم بداند که پیغام هایی را بایستی به اپلیکیشن تحت موبایل اندرویدی ارسال (push) کند، این پیغام را از طریق متد HTTP POST به سرویس دهنده ی C2DM گوگل ارسال می نماید.

سرورهای C2DM پیغام را به دستگاه مربوطه آدرس دهی و ارسال می کنند (route). چنانچه دستگاه مورد نظر آنلاین نبود، در آن صورت پیغام زمانی تحویل داده می شود که دستگاه به اینترنت وصل و در دسترس قرار گیرد. پس از اینکه پیغام تحویل داده شد (دستگاه آن را دریافت کرد)، یک Broadcast intent ایجاد می شود. اپلیکیشن تحت موبایل از قبل ویژه ی این Broadcast، یک Intent Receiver تعریف کرده است. بنابراین اپلیکیشن مربوطه اجرا شده و پیغام را از طریق Intent Receiver مزبور پردازش می کند.

پیغام های C2DM معمولا از مرز 1024 بایت تجاوز نمی کنند و منحصرا برای این تعبیه شده اند که دستگاه مربوطه را از وجود داده های جدید آگاه کنند. به عبارت دیگر این پیغام با ارسالشان را از وجود اطلاعات جدید آگاه ساخته و هیچ گونه داده ای را به خودی خود به دستگاه منتقل نمی کنند. workflow و روند کار به این صورت هست که سرویس دهنده های C2DM گوگل، اپلیکیشن اندرویدی را از موجود بودن داده های جدید جهت بروز رسانی برنامه مطلع می سازند. پس از آن، اپلیکیشن مستقر بر روی دستگاه موبایل، داده ها را از سرویس دهنده ی دیگری واکشی و دریافت می کند.

دستگاه های اندرویدی همواره ارتباط خود را با سرویس دهنده ی Google Play برقرار نگه می دارند. C2DM از اتصال موجود به سرویس دهنده های گوگل استفاده می کند. این اتصال بهینه سازی شده تا مصرف پهنای باند و باتری تا حد امکان کاهش یابد.

در حال حاضر C2DM هنوز در مرحله ی تست بتا به سر می برد و برای استفاده از آن، شما می بایست درخواست و ثبت نام نمایید. C2DM به هر ارسال کننده اجازه می دهد تا سقف 200,000 پیغام در روز به صورت رایگان ارسال کند.

ابزار لازم برای تست کاربردی C2DM

سرویس C2MD از ویرایش 2.2 اندروید قابل بهره برداری بوده و جهت استفاده از آن لازم است اپلیکیشن Android Play بر روی دستگاه مورد نظر نصب شده باشد.

برای استفاده از C2DM در شبیه ساز محیط اندروید، بایستی از یک دستگاه که ورژن 8 کتابخانه های اندروید (API 8) یا بالاتر بر روی آن نصب است استفاده نموده و نیز از طریق بخش Settings، یک حساب کاربری Google در شبیه ساز ایجاد نمایید.

مجوزهای لازم

جهت استفاده از C2DM در اپلیکیشن خود، بایستی مجوزهای زیر را اعلان نمایید:

- com.google.android.c2dm.permission.RECEIVE
- android.permission.INTERNET

علاوه بر دو مجوز مزبور، اپلیکیشن شما بایستی مجوز "applicationPackage ".permission.C2D_MESSAGE" را با "android:protectionLevel" که با "signature" مقداردهی شده تنظیم نمایید تا بدین وسیله سایر اپلیکیشن ها نتوانند به پیغام گوش داده و آن را دریافت کنند. در واقع اپلیکیشن های دیگر برای اینکه بتوانند به این پیغام گوش داده و آن را دریافت کنند، می بایست با گواهی نامه و امضای دیجیتالی (certificate) که اپلیکیشن اصلی دریافت کننده ی پیغام با آن امضا شده، امضا شده باشند. به عبارت دیگر "android:protectionLevel="signature سبب می شود تنها اپلیکیشن هایی که با امضای دیجیتالی یکسان امضا شده اند، بتوانند پیغام را دریافت کنند.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

Intent Receiver (اعلان intent receiver جهت دریافت intent های مربوطه)

اپلیکیشن شما می بایست یک intent receiver جهت دریافت دو intent اعلان نماید:

- com.google.android.c2dm.intent.REGISTRATION
- com.google.android.c2dm.intent.RECEIVE

receiver ای که ویژه ی "com.google.android.c2dm.intent.RECEIVE" اعلان شده، زمانی صدا زده می شود که پیغام جدیدی دریافت شده باشد، در حالی که receiver اعلان شده برای "com.google.android.c2dm.intent.REGISTRATION" زمانی فراخوانی می شود که کد ثبت (registration code) اپلیکیشن مربوطه دریافت شده باشد.

مراحل پیاده سازی

ثبت و معرفی app server (سرویس دهنده ی اپلیکیشن)

سرویس دهنده ی اپلیکیشن می بایست خود را برای سرورهای C2DM معرفی و احراز هویت (authenticate) نماید. از طریق ایمیل و گذرواژه، یک شناسه ی احراز هویت یا امنیتی (authentication token) که با متد درخواست اطلاعات HTTP POST به سرورهای C2DM تعیین شده و در اختیار اپلیکیشن قرار می گیرد. حال شناسه ی امنیتی یا همان token در اپلیکیشن سمت سرویس دهنده (app server) ذخیره شده و زمانی که اپلیکیشن می خواهد پیغامی را ارسال کند، خود را به واسطه ی این شناسه به سرورهای C2DM می شناساند.

برای مثال، جهت دریافت شناسه ی امنیتی که با ارائه ی آدرس و گذرواژه ی ایمیل معتبر و ثبت شده، در اختیار اپلیکیشن قرار می گیرد، می توانید به صورت زیر اقدام نمایید (کد و کلاس زیر را پیاده سازی نمایید):

```
package de.vogella.java.c2dm.server.util;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
```

```
public class AuthenticationUtil {
	private AuthenticationUtil() {
		// Util class cannot get instanziated
	}
	public static String getToken(String email, String password)
			throws IOException {
		// create the post data
		// Requires a field with the email and the password
		StringBuilder builder = new StringBuilder();
		builder.append("Email=").append(email);
		builder.append("&Passwd=").append(password);
		builder.append("&accountType=GOOGLE");
		builder.append("&source=MyLittleExample");
		builder.append("&service=ac2dm");
		// Setup the Http Post
		byte[] data = builder.toString().getBytes();
		URL url = new URL("https://www.google.com/accounts/ClientLogin");
		HttpURLConnection con = (HttpURLConnection) url.openConnection();
		con.setUseCaches(false);
		con.setDoOutput(true);
		con.setRequestMethod("POST");
		con.setRequestProperty("Content-Type",
				"application/x-www-form-urlencoded");
		con.setRequestProperty("Content-Length", Integer.toString(data.length));
		// Issue the HTTP POST request
		OutputStream output = con.getOutputStream();
		output.write(data);
		output.close();
		// read the response
		BufferedReader reader = new BufferedReader(new InputStreamReader(
				con.getInputStream()));
		String line = null;
		String auth_key = null;
		while ((line = reader.readLine()) != null) {
			if (line.startsWith("Auth=")) {
				auth_key = line.substring(5);
			}
		}
		// Finally get the authentication token
		// To something useful with it
		return auth_key;
	}
}
```

شناسه ی امنیتی (token) در فواصل زمانی معین بروز رسانی می شود.

مثال بالا با زبان جاوا نوشته شده است. این امکان نیز وجود دارد که شناسه ی مورد نیاز را با استفاده از سایر ابزار http یا زبان های برنامه نویسی تهیه کرد. به طور مثال، شما می توانید با بهره گیری از ابزار خط فرمان (command line tool) و ابزار curl، سرور را شبیه سازی نمایید.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

دریافت شناسه ی ثبت (registration ID) اپلیکیشن تحت موبایل

جهت معرفی اپلیکیشن های اندرویدی و تحت موبایل خود به سرورهای C2DM (به منظور دریافت پیغام و خدمات بروز رسانی از سرور توسعه دهنده ی اپلیکیشن)، لازم است "com.google.android.c2dm.intent.REGISTER" را اعلان (fire) نمایید. با این کار سرویس فراخوانی (trigger) شده که شناسه ی ثبت (registration id) را به سرورهای C2DM ارسال می نماید.

آبجکت intent حامل اطلاعات اضافی به نام extra است. "sender" به عنوان کلید (پارامتر ورودی اول متد putExtra) و آدرس ایمیل که با آن ثبت انجام شده به عنوان پارامتر ورودی دوم پذیرفته می شود. بعلاوه، آبجکت intent مزبور بایستی دربردارنده ی PendingIntent با اطلاعات اضافی در قالب extra که به عنوان پارامتر اول به متد putExtra ارسال شده، باشد. PendingIntent به سیستم اندروید اطلاعاتی پیرامون اپلیکیشن جاری ارائه می دهد. مقدار sender آدرس ایمیلی است که تحت آن سرویس پیغام رسانی C2MD خود را ثبت یا اعلان نمودید. مقدار رشته ای "youruser@gmail.com" را که آدرس ایمیل شما می باشد، جایگزین پارامتر دوم متد putExtra نمایید.

```
public void register(View view) {
    Intent intent = new Intent("com.google.android.c2dm.intent.REGISTER");
    intent.putExtra("app",PendingIntent.getBroadcast(this, 0, new Intent(), 0));
    intent.putExtra("sender", "youruser@gmail.com");
    startService(intent);
}
```

سرویس به صورت ناهمزمان خود را به Google معرفی کرده و اینتنت "com.google.android.c2dm.intent.REGISTRATION" را به محض موفقیت آمیز بودن ثبت (registration) ارسال می کند. اپلیکیشن شما بایستی یک Broadcast receiver ویژه ی این intent ثبت کند. برای این منظور لازم است مجوز مورد نیاز را بر اساس پکیج خود اعلان نمایید چرا که سیستم اندروید این امر را از داخل بررسی می کند.

```
package de.vogella.android.c2dm.simpleclient;
import android.content.BroadcastReceiver;
import android.content.Context;
```

```
import android.content.Intent;
import android.util.Log;
public class C2DMRegistrationReceiver extends BroadcastReceiver {
	@Override
	public void onReceive(Context context, Intent intent) {
		String action = intent.getAction();
		Log.w("C2DM", "Registration Receiver called");
		if ("com.google.android.c2dm.intent.REGISTRATION".equals(action)) {
			Log.w("C2DM", "Received registration ID");
			final String registrationId = intent
					.getStringExtra("registration_id");
			String error = intent.getStringExtra("error");
			Log.d("C2DM", "dmControl: registrationId = " + registrationId
					+ ", error = " + error);
			// TODO Send this to my application server
		}
	}
}
```

محتوای فایل تنظیمات اپلیکیشن "AndroidManifest.xml" مشابه زیر می باشد. به خاطر داشته باشید که اگر از پکیج دیگری استفاده می کنید، بایستی کد و محتوای فایل را نیز متعاقبا تنظیم نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="de.vogella.android.c2dm.simpleclient"
  android:versionCode="1"
  android:versionName="1.0" >
  <uses-sdk android:minSdkVersion="8" />
  <permission
    android:name="de.vogella.android.c2dm.simpleclient.permission.C2D_MESSAGE"
    android:protectionLevel="signature" />
  <uses-permission
    android:name="de.vogella.android.c2dm.simpleclient.permission.C2D_MESSAGE" />
  <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
  <uses-permission android:name="android.permission.INTERNET" />
  <application
    android:icon="@drawable/icon"
    android:label="@string/app_name" >
    <activity
      android:label="@string/app_name"
      android:name=".C2DMClientActivity" >
      <intent-filter >
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
    <receiver
      android:name=".C2DMRegistrationReceiver"
      android:permission="com.google.android.c2dm.permission.SEND" >
```

```
        <intent-filter >
          <action android:name="com.google.android.c2dm.intent.REGISTRATION" >
          </action>
          <category android:name="de.vogella.android.c2dm.simpleclient" />
        </intent-filter>
      </receiver>
    </application>
</manifest>
```

اینتنت "com.google.android.c2dm.intent.REGISTRATION" دربردارنده ی یک شناسه ی ثبت (registration ID) می باشد.هر registration ID نشانگر و نماینده یک دستگاه منحصربفرد و معین می باشد. مانند اینکه هر گوشی اندروید یک کد ثبت و شناسه ی (registration code) اختصاصی خود را دارد.

ممکن است C2DM شناسه ی ثبت را در فواصل زمانی معین بروز رسانی کند اما تا آن زمان، اپلیکیشن شما می بایست این ID را برای استفاده ی آینده نزد خود ذخیره نگه دارد.

پس از اینکه اپلیکیشن اندرویدی شناسه ی ثبت (registration ID) را دریافت کرد، می بایست این اطلاعات را به سرویس دهنده ی اپلیکشن (App server) ارسال کند. این سرویس دهنده با استفاده از شناسه ی ثبت، یک پیغام از طریق سرورهای C2DM به دستگاهی که اپلیکیشن اندرویدی بر روی آن نصب است ارسال می کند.

به عنوان مثال، کد زیر deviceId و registrarionId را به یک سرویس دهنده ارسال می کند.

```
// Better do this in an asynchronous thread
public void sendRegistrationIdToServer(String deviceId, String registrationId) {
      Log.d("C2DM", "Sending registration ID to my application server");
      HttpClient client = new DefaultHttpClient();
      HttpPost post = new HttpPost("http://your_url/register");
      try {
            List<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(1);
            // Get the deviceID
            nameValuePairs.add(new BasicNameValuePair("deviceid", deviceId));
            nameValuePairs.add(new BasicNameValuePair("registrationid", registrationId));
            post.setEntity(new UrlEncodedFormEntity(nameValuePairs));
            HttpResponse response = client.execute(post);
            BufferedReader rd =
```

```
                new BufferedReader(new InputStreamReader(response.getEntity().getContent()));
            String line = "";
            while ((line = rd.readLine()) != null) {
                Log.e("HttpResponse", line);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
}
```

سرویس دهنده registration ID ها را به طور دائمی در خود ذخیره می کند.

ثبت و اعلان یک Receiver ویژه ی پیغام های ارسالی از سرورهای C2DM

درست مشابه ثبت registration receiver، بایستی یک message receiver جهت دریافت پیغام های ارسالی از C2MD معرفی نمایید. این می تواند همان registration receiver یا receiver مجزا باشد. مثال زیر ثبت یک message receiver مجزا را نشان می دهد.

```
package de.vogella.android.c2dm.simpleclient;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
public class C2DMMessageReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        Log.w("C2DM", "Message Receiver called");
        if ("com.google.android.c2dm.intent.RECEIVE".equals(action)) {
            Log.w("C2DM", "Received message");
            final String payload = intent.getStringExtra("payload");
            Log.d("C2DM", "dmControl: payload = " + payload);
            // Send this to my application server
        }
    }
}
```

همچنین لازم است که message receiver را به صورت زیر در فایل تنظیمات اپلیکیشن AndroidManifest.xml اعلان نمایید.

```
<receiver android:name=".C2DMMessageReceiver"
    android:permission="com.google.android.c2dm.permission.SEND">
    <intent-filter>
        <action android:name="com.google.android.c2dm.intent.RECEIVE"></action>
        <category android:name="de.vogella.android.c2dm.simpleclient" />
    </intent-filter>
</receiver>
```

ارسال پیغام

اکنون application server و اپلیکیشن تحت موبایل اندرویدی شما آماده ی استفاده از C2DM و تبادل اطلاعات با یکدیگر هستند. سرور، شناسه ی امنیتی/احراز هویت (authentication token) و شناسه ی ثبت (registration ID) اپلیکیشن سرویس گیرنده را در اختیار دارد و اپلیکیشن نیز با اعلان Broadcast Receiver آماده ی این است که پیغام های مربوطه را دریافت کند.

به منظور ارسال پیغام به دستگاه مورد نظر (سرویس گیرنده ی نهایی)، App server یک درخواست HTTP POST به سرورهای C2MD گوگل ارسال می کند. این درخواست یا متد HTTP GET دربردارنده ی registration ID دستگاه مورد نظر و شناسه ی احراز هویت (برای اینکه به گوگل اعلان کند که سرور اجازه ی ارسال پیغام به اپیلکیشن تحت موبایل را دارد) می باشد.

```
package de.vogella.java.c2dm.server.util;
import java.io.IOException;
import java.io.OutputStream;
import java.net.URL;
import java.net.URLEncoder;
import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLSession;
public class MessageUtil {
	private final static String AUTH = "authentication";
	private static final String UPDATE_CLIENT_AUTH = "Update-Client-Auth";
	public static final String PARAM_REGISTRATION_ID = "registration_id";
	public static final String PARAM_DELAY_WHILE_IDLE = "delay_while_idle";
	public static final String PARAM_COLLAPSE_KEY = "collapse_key";
	private static final String UTF8 = "UTF-8";
	public static int sendMessage(String auth_token, String registrationId,
			String message) throws IOException {
		StringBuilder postDataBuilder = new StringBuilder();
		postDataBuilder.append(PARAM_REGISTRATION_ID).append("=")
				.append(registrationId);
		postDataBuilder.append("&").append(PARAM_COLLAPSE_KEY).append("=")
				.append("0");
		postDataBuilder.append("&").append("data.payload").append("=")
				.append(URLEncoder.encode(message, UTF8));
		byte[] postData = postDataBuilder.toString().getBytes(UTF8);
		// Hit the dm URL.
		URL url = new URL("https://android.clients.google.com/c2dm/send");
		HttpsURLConnection
				.setDefaultHostnameVerifier(new CustomizedHostnameVerifier());
```

```
        HttpsURLConnection conn = (HttpsURLConnection) url.openConnection();
        conn.setDoOutput(true);
        conn.setUseCaches(false);
        conn.setRequestMethod("POST");
        conn.setRequestProperty("Content-Type",
                "application/x-www-form-urlencoded;charset=UTF-8");
        conn.setRequestProperty("Content-Length",
                Integer.toString(postData.length));
        conn.setRequestProperty("Authorization", "GoogleLogin auth="
                + auth_token);
        OutputStream out = conn.getOutputStream();
        out.write(postData);
        out.close();
        int responseCode = conn.getResponseCode();
        return responseCode;
    }
    private static class CustomizedHostnameVerifier implements HostnameVerifier {
        public boolean verify(String hostname, SSLSession session) {
            return true;
        }
    }
}
```

زمانی که سرویس دهنده (app server) پیغام را به سرور C2DM ارسال می کند، این سرور پیغام را در صف و حالت انتظار نگه می دارد تا دستگاه آنلاین شود. پیغام در قالب یک broadcast به دستگاه مربوطه ارسال می شود. اپلیکیشن شما بایستی از قبل برای این broadcast event ثبت نام کرده (به آن گوش فرا داده) تا آن را به هنگام ارسال، دریافت نماید.

پیغام دریافتی به broadcast receiver ای که به "com.google.android.c2dm.intent.RECEIVE" گوش فرا داده و ویژه ی آن ثبت شده، فرستاده می شود. داده های مورد نظر به راحتی با فراخوانی متد getExtras() بر روی نمونه ی ساخته شده از کلاس Intent قابل بازیابی می باشد. کلیدهای موجود عبارتند از "payload"، "from"، "collapse_key". داده های اصلی در پارامتر "payload" قرار دارند. receiver می تواند این داده ها را استخراج کرده و به آن واکنش مناسب را نشان دهد.

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330

ثبت نام و درخواست برای استفاده از سرویس C2DM

در حال حاضر C2DM در مرحله ی آزمایش بتا به سر می برد. برای استفاده از آن نیاز به مجوز دسترسی و ثبت نام دارید. برای ثبت نام و دسترسی به این سرویس می توانید به code.google.com/android/c2dm/signup.html مراجعه نمایید.

اگر می خواهید مثال زیر را بر روی محیط شبیه ساز اجرا و تست نمایید، در آن صورت لازم است ورژن 8 کتابخانه های اندروید (API 8) یا بالاتر را نصب کرده و در اختیار داشته باشید. همچنین لازم است یک حساب کاربری گوگل (Google user) در محیط شبیه ساز ایجاد نمایید. برای این منظور کافی است مسیر رو به رو را طی کنید: Settings ▸ Accounts Sync.

چنانچه قصد دارید که مثال حاضر را بر روی دستگاه واقعی اندروید تست نمایید، در آن صورت لازم است Android Market را بر روی دستگاه مربوطه نصب کنید.

آموزش کاربردی: طراحی و ساخت اپلیکیشن با قابلیت تعامل با سرورهای C2DM (C2DM enabled)

ایجاد پروژه و فایل layout

پروژه و activity جدید دیگری به ترتیب به نام های "de.vogella.android.c2dm.simpleclient" و "C2DMClientActivity" ایجاد نمایید. حال یک فایل main.xml با محتوای زیر ایجاد نمایید.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
  <Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="register"
    android:text="Register" >
  </Button>
  <Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="showRegistrationId"
    android:text="Show" >
```

```
  </Button>
</LinearLayout>
```

سپس فایل "activity_result.xml" را با محتوای زیر ایجاد نمایید. این فایل را در activity های که صفحه ی نتایج را نمایش می دهند فراخوانی خواهیم کرد.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
  <TextView
    android:id="@+id/result"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="center"
    android:text="No info."
    android:textAppearance="?android:attr/textAppearanceLarge" >
  </TextView>
</LinearLayout>
```

ایجاد receiver ها و activity های مورد نیاز

دو کلاس زیر به نام های C2DMRegistrationReceiver و "C2DMMessageReceiver" را ایجاد نمایید. این دو کلاس بعده ها به عنوان receiver یا گوش فرادهنده به (دریافت کننده ی) registration intent و message intent معرفی شده و مورد استفاده قرار خواهند گرفت.

```
package de.vogella.android.c2dm.simpleclient;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.List;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.SharedPreferences.Editor;
import android.preference.PreferenceManager;
```

```
import android.provider.Settings.Secure;
import android.util.Log;
public class C2DMRegistrationReceiver extends BroadcastReceiver {
@Override
public void onReceive(Context context, Intent intent) {
String action = intent.getAction();
Log.w("C2DM", "Registration Receiver called");
if ("com.google.android.c2dm.intent.REGISTRATION".equals(action)) {
Log.w("C2DM", "Received registration ID");
final String registrationId = intent
.getStringExtra("registration_id");
String error = intent.getStringExtra("error");
Log.d("C2DM", "dmControl: registrationId = " + registrationId
+ ", error = " + error);
String deviceId = Secure.getString(context.getContentResolver(),
Secure.ANDROID_ID);
createNotification(context, registrationId);
sendRegistrationIdToServer(deviceId, registrationId);
// Also save it in the preference to be able to show it later
saveRegistrationId(context, registrationId);
}
}
private void saveRegistrationId(Context context, String registrationId) {
SharedPreferences prefs = PreferenceManager
.getDefaultSharedPreferences(context);
Editor edit = prefs.edit();
edit.putString(C2DMClientActivity.AUTH, registrationId);
edit.commit();
}
public void createNotification(Context context, String registrationId) {
NotificationManager notificationManager = (NotificationManager) context
.getSystemService(Context.NOTIFICATION_SERVICE);
Notification notification = new Notification(R.drawable.icon,
"Registration successful", System.currentTimeMillis());
// hide the notification after its selected
notification.flags |= Notification.FLAG_AUTO_CANCEL;
Intent intent = new Intent(context, RegistrationResultActivity.class);
intent.putExtra("registration_id", registrationId);
PendingIntent pendingIntent = PendingIntent.getActivity(context, 0,
intent, 0);
notification.setLatestEventInfo(context, "Registration",
"Successfully registered", pendingIntent);
notificationManager.notify(0, notification);
}
// incorrect usage as the receiver may be canceled at any time
// do this in an service and in an own thread
public void sendRegistrationIdToServer(String deviceId,
String registrationId) {
Log.d("C2DM", "Sending registration ID to my application server");
HttpClient client = new DefaultHttpClient();
HttpPost post = new HttpPost("http://vogellac2dm.appspot.com/register");
try {
```

```
List<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(1);
                // Get the deviceID
nameValuePairs.add(new BasicNameValuePair("deviceid", deviceId));
nameValuePairs.add(new BasicNameValuePair("registrationid",
                                                registrationId));
post.setEntity(new UrlEncodedFormEntity(nameValuePairs));
HttpResponse response = client.execute(post);
BufferedReader rd = new BufferedReader(new InputStreamReader(
                response.getEntity().getContent()));
String line = "";
while ((line = rd.readLine()) != null) {
Log.e("HttpResponse", line);
}
} catch (IOException e) {
e.printStackTrace();
}
}
}

package de.vogella.android.c2dm.simpleclient;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
public class C2DMMessageReceiver extends BroadcastReceiver {
@Override
public void onReceive(Context context, Intent intent) {
String action = intent.getAction();
Log.w("C2DM", "Message Receiver called");
if ("com.google.android.c2dm.intent.RECEIVE".equals(action)) {
Log.w("C2DM", "Received message");
final String payload = intent.getStringExtra("payload");
Log.d("C2DM", "dmControl: payload = " + payload);
// TODO Send this to my application server to get the real data
// Lets make something visible to show that we received the message
createNotification(context, payload);
}
}
public void createNotification(Context context, String payload) {
NotificationManager notificationManager = (NotificationManager) context
.getSystemService(Context.NOTIFICATION_SERVICE);
Notification notification = new Notification(R.drawable.icon,
"Message received", System.currentTimeMillis());
// hide the notification after its selected
notification.flags |= Notification.FLAG_AUTO_CANCEL;
Intent intent = new Intent(context, MessageReceivedActivity.class);
intent.putExtra("payload", payload);
PendingIntent pendingIntent = PendingIntent.getActivity(context, 0,
intent, PendingIntent.FLAG_CANCEL_CURRENT);
```

```
notification.setLatestEventInfo(context, "Message",
"New message received", pendingIntent);
notificationManager.notify(0, notification);

}

}
```

حال دو کلاس activity زیر که نتایج را در صفحه برای کاربر به نمایش می گذارند، ایجاد و پیاده سازی نمایید.

```
package de.vogella.android.c2dm.simpleclient;
import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;
public class RegistrationResultActivity extends Activity {
@Override
protected void onCreate(Bundle savedInstanceState) {
setContentView(R.layout.activity_result);
Bundle extras = getIntent().getExtras();
if (extras != null) {
String registrationId = extras.getString("registration_id");
if (registrationId != null && registrationId.length() > 0) {
TextView view = (TextView) findViewById(R.id.result);
view.setText(registrationId);
}
}
super.onCreate(savedInstanceState);
}
}
```

```
package de.vogella.android.c2dm.simpleclient;
import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;
public class MessageReceivedActivity extends Activity {
@Override
protected void onCreate(Bundle savedInstanceState) {
setContentView(R.layout.activity_result);
Bundle extras = getIntent().getExtras();
if (extras != null) {
String message = extras.getString("payload");
if (message != null && message.length() > 0) {
TextView view = (TextView) findViewById(R.id.result);
view.setText(message);
}
}
super.onCreate(savedInstanceState);
}
}
```

فایل تنظیمات اپلیکیشن AndroidManifest.xml را به صورت زیر اعلان نمایید. این فایل intent receiver ها، activity ها را تعریف کرده و مجوزهای لازم را درخواست می کند.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="de.vogella.android.c2dm.simpleclient"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk android:minSdkVersion="8" />
    <permission
android:name="de.vogella.android.c2dm.simpleclient.permission.C2D_MESSAGE"
    android:protectionLevel="signature" />
    <uses-permission
android:name="de.vogella.android.c2dm.simpleclient.permission.C2D_MESSAGE" />
<uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <application
    android:icon="@drawable/icon"
    android:label="@string/app_name" >
        <activity
    android:label="@string/app_name"
    android:name=".C2DMClientActivity" >
            <intent-filter >
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <receiver
    android:name=".C2DMRegistrationReceiver"
    android:permission="com.google.android.c2dm.permission.SEND" >
            <intent-filter >
<action android:name="com.google.android.c2dm.intent.REGISTRATION" >
            </action>
<category android:name="de.vogella.android.c2dm.simpleclient" />
            </intent-filter>
        </receiver>
        <receiver
    android:name=".C2DMMessageReceiver"
    android:permission="com.google.android.c2dm.permission.SEND" >
            <intent-filter >
<action android:name="com.google.android.c2dm.intent.RECEIVE" >
            </action>
<category android:name="de.vogella.android.c2dm.simpleclient" />
            </intent-filter>
        </receiver>
<activity android:name="RegistrationResultActivity" >
        </activity>
<activity android:name="MessageReceivedActivity" >
        </activity>
    </application>
```

```
</manifest>
```

محتوای کلاس "C2DMClientActivity" را به صورت زیر ویرایش نمایید.

```
package de.vogella.android.c2dm.simpleclient;
import android.app.Activity;
import android.app.PendingIntent;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.preference.PreferenceManager;
import android.util.Log;
import android.view.View;
import android.widget.Toast;
public class C2DMClientActivity extends Activity {
public final static String AUTH = "authentication";
// Example Activity to trigger a request for a registration ID to the Google
// server
@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
}
public void register(View view) {
Log.w("C2DM", "start registration process");
Intent intent = new Intent("com.google.android.c2dm.intent.REGISTER");
intent.putExtra("app",
PendingIntent.getBroadcast(this, 0, new Intent(), 0));
// Sender currently not used
intent.putExtra("sender", "nonsenses@gmail.com");
startService(intent);
}
public void showRegistrationId(View view) {
SharedPreferences prefs = PreferenceManager
.getDefaultSharedPreferences(this);
String string = prefs.getString(AUTH, "n/a");
Toast.makeText(this, string, Toast.LENGTH_LONG).show();
Log.d("C2DM RegId", string);
}
}
```

متدهای موجود در activity به واسطه ی خاصیت (property) onclick به المان های button در لایه ی XML متصل هستند. اولین دکمه در activity، یک درخواست برای registration ID و دومین دکمه کلید ثبت یا registration key ذخیره شده را نمایش می دهد. هر دو دکمه همچنین registration id را در Logcat (logcat view) چاپ می کنند.

Registration id را از Logcat View کپی نمایید چرا که بعده ها می بایست آن را در پیاده سازی سرور مجددا مورد استفاده قرار دهید.

ثبت اپلیکیشن برای دریافت پیغام (Registration)

اپلیکیشن خود را اجرا کرده، حساب کاربری ثبت شده را نزد خود نگاه داشته و سپس دکمه ی مورد نظر را فشار دهید. حال registration ID را در LogCat پیدا کنید.

در صورت مشاهده ی پیغام زیر مطمئن شوید که از یک دستگاه مبتنی بر Google استفاده نموده و یک حساب کاربری Google در دستگاه مربوطه ایجاد کرده اید:

```
Unable to start service Intent
    {act=com.google.android.c2dm.intent.REGISTER ... }: not found
```

آموزش کاربردی: استفاده از ابزار خط فرمان curl جهت شبیه سازی قابلیت های سرور

سیستم عامل های مبتنی بر Linux این امکان را به شما می دهنده تا سرویس مورد نظر را به راحتی از طریق خط فرمان (command line) تست نمایید. می توان با استفاده از curl در خط فرمان، کلید احراز هویت (authentication key) را درخواست نمود. از پاسخ/خروجی، آن بخشی که پس از "Auth=" قرار دارد را بازیابی نمایید.

```
curl https://www.google.com/accounts/ClientLogin -d
    Email=your_user -d "Passwd=your_password" -d accountType=GOOGLE
    -d source=Google-cURL-Example -d service=ac2dm
```

این بخش (قطعه ی پس از auth) و کد ثبت (registration code) را می توان جهت ارسال یک پیغام به دستگاه مربوطه مورد استفاده قرار داد.

```
curl --header "Authorization: GoogleLogin auth=your_authenticationid"
    "https://android.apis.google.com/c2dm/send" -d registration_id=your_registration
    -d "data.payload=payload" -d collapse_key=0
```

آموزش کاربردی: ایجاد server app

همان طور که قبلا تشریح شد، app server می بایست از طریق پروتکل HTTPS یک Authentication key (کلیدی جهت احراز هویت) بازیابی کند. پس از آن app server قادر خواهد بود که پیغام هایی را از طریق پروتکل HTTP به دستگاه مربوطه ارسال نماید. app server جهت شناساندن خود به سرور C2MD می بایست کلید احراز هویت (Auth key) و شناسه ی ثبت اپلیکیشن (registration ID) را ارائه نماید.

در تمرین حاضر قصد داریم تا سرور را به واسطه ی یک برنامه ی نوشته شده با جاوا شبیه سازی کنیم. از آنجایی که امکان برقراری ارتباط اپلیکیشن از طریق پروتکل HTTP برای ما وجود ندارد، شناسه ی ثبت (registration ID) دستگاه به صورت hardcode در اپلیکیشن قرار دارد. به خاطر داشته باشید این اپلیکیشن صرفا یک مثال بوده و جهت تست آسان C2DM طراحی شده است.

جهت ذخیره ی شناسه ی کاربری و اعتبارنامه (credentials)، کلاس زیر را پیاده سازی نمایید.

```
package de.vogella.java.c2dm.server.secret;
public class SecureStorage {
	public static final String USER = "your_registeredUser";
	public static final String PASSWORD = "your_password";
}
```

یک پروژه ی جدید جاوایی به نام "de.vogella.java.c2dm.server" ایجاد نمایید. سپس کلاس زیر را پیاده سازی کنید. این کلاس در واقع یک کلاس کمکی (utility class) است که به منظور بازیابی شناسه ی امنیتی/احرازهویت (authentication token) از سرور Google مورد استفاده قرار می گیرد.

```
package de.vogella.java.c2dm.server.util;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
public class AuthenticationUtil {
	private AuthenticationUtil() {
		// Util class cannot get instanziated
	}
	public static String getToken(String email, String password)
			throws IOException {
		// create the post data
```

```
        // Requires a field with the email and the password
        StringBuilder builder = new StringBuilder();
        builder.append("Email=").append(email);
        builder.append("&Passwd=").append(password);
        builder.append("&accountType=GOOGLE");
        builder.append("&source=MyLittleExample");
        builder.append("&service=ac2dm");
        // Setup the Http Post
        byte[] data = builder.toString().getBytes();
        URL url = new URL("https://www.google.com/accounts/ClientLogin");
        HttpURLConnection con = (HttpURLConnection) url.openConnection();
        con.setUseCaches(false);
        con.setDoOutput(true);
        con.setRequestMethod("POST");
        con.setRequestProperty("Content-Type",
                "application/x-www-form-urlencoded");
        con.setRequestProperty("Content-Length", Integer.toString(data.length));
        // Issue the HTTP POST request
        OutputStream output = con.getOutputStream();
        output.write(data);
        output.close();
        // read the response
        BufferedReader reader = new BufferedReader(new InputStreamReader(
                con.getInputStream()));
        String line = null;
        String auth_key = null;
        while ((line = reader.readLine()) != null) {
            if (line.startsWith("Auth=")) {
                auth_key = line.substring(5);
            }
        }
        // Finally get the authentication token
        // To something useful with it
        return auth_key;
    }
}
```

کلاسی به نام GetAuthenticationToken را با بدنه ی زیر پیاده سازی نمایید. کلاس نام برده قادر است شناسه ی امنیتی (auth token) را واکشی کند.

```
package de.vogella.java.c2dm.server;
import java.io.IOException;
import de.vogella.java.c2dm.server.secret.SecureStorage;
import de.vogella.java.c2dm.server.util.AuthenticationUtil;
public class GetAuthenticationToken {
    public static void main(String[] args) throws IOException {
        String token = AuthenticationUtil.getToken(SecureStorage.USER,
                SecureStorage.PASSWORD);
        System.out.println(token);
    }
}
```

کلاس GetAuthenticationToken را اجرا نموده و شناسه ی امنیتی (authentication token) را از خط فرمان کپی نمایید. حال کلاس زیر را تعریف کرده، شناسه ی امنیتی (authentication token) و شناسه ی ثبت اپلیکیشن (registration id) را نزد خود نگاه دارید.

```
package de.vogella.java.c2dm.server;
public class ServerConfiguration {
	public static final String AUTHENTICATION_TOKEN = "your_token";
	public static final String REGISTRATION_ID = "registration_id_of_your_device";
}
```

سپس یک کلاس کمکی (utility class) جهت ارسال پیغام به دستگاه مربوطه با پیاده سازی زیر ایجاد نمایید.

```
package de.vogella.java.c2dm.server.util;
import java.io.IOException;
import java.io.OutputStream;
import java.net.URL;
import java.net.URLEncoder;
import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLSession;
public class MessageUtil {
	private final static String AUTH = "authentication";
	private static final String UPDATE_CLIENT_AUTH = "Update-Client-Auth";
	public static final String PARAM_REGISTRATION_ID = "registration_id";
	public static final String PARAM_DELAY_WHILE_IDLE = "delay_while_idle";
	public static final String PARAM_COLLAPSE_KEY = "collapse_key";
	private static final String UTF8 = "UTF-8";
	public static int sendMessage(String auth_token, String registrationId,
			String message) throws IOException {
		StringBuilder postDataBuilder = new StringBuilder();
		postDataBuilder.append(PARAM_REGISTRATION_ID).append("=")
				.append(registrationId);
		postDataBuilder.append("&").append(PARAM_COLLAPSE_KEY).append("=")
				.append("0");
		postDataBuilder.append("&").append("data.payload").append("=")
				.append(URLEncoder.encode(message, UTF8));
		byte[] postData = postDataBuilder.toString().getBytes(UTF8);
		// Hit the dm URL.
		URL url = new URL("https://android.clients.google.com/c2dm/send");
		HttpsURLConnection
				.setDefaultHostnameVerifier(new CustomizedHostnameVerifier());
		HttpsURLConnection conn = (HttpsURLConnection) url.openConnection();
		conn.setDoOutput(true);
		conn.setUseCaches(false);
		conn.setRequestMethod("POST");
		conn.setRequestProperty("Content-Type",
				"application/x-www-form-urlencoded;charset=UTF-8");
```

```
            conn.setRequestProperty("Content-Length",
                    Integer.toString(postData.length));
            conn.setRequestProperty("Authorization", "GoogleLogin auth="
                    + auth_token);
            OutputStream out = conn.getOutputStream();
            out.write(postData);
            out.close();
            int responseCode = conn.getResponseCode();
            return responseCode;
        }
        private static class CustomizedHostnameVerifier implements HostnameVerifier {
            public boolean verify(String hostname, SSLSession session) {
                return true;
            }
        }
}
```

در پایان کلاسی به نام "SendMessageToDevice" که وظیفه ی آن منحصرا ارسال پیغام به دستگاه شما می باشد.

```
package de.vogella.java.c2dm.server;
import java.io.IOException;
import de.vogella.java.c2dm.server.util.MessageUtil;
public class SendMessageToDevice {
    public static void main(String[] args) throws IOException {
        // "Message to your device." is the message we will send to the Android app
        int responseCode = MessageUtil.sendMessage(
                ServerConfiguration.AUTHENTICATION_TOKEN,
                ServerConfiguration.REGISTRATION_ID, "Message to your device.");
        System.out.println(responseCode);
    }
}
```

اپلیکیشن را اجرا نمایید. اپلیکیشن حاضر می بایست پیغامی به دستگاه شما ارسال نموده و کد بازگشتی "200" را در خروجی برگرداند. خواهید دید که در نمایشگر دستگاه یک اطلاعیه (notification) قابل مشاهده است که با کلیک بر روی آن پیغام مورد نظر نشان داده می شود.

بخش پنجم

توابع کتابخانه ای مربوط به تقویم / آموزش Calendar API

آموزش حاضر به شرح نحوه ی فراخوانی و پیاده سازی توابع کتابخانه ای مربوط به تقویم یا Calendar API در اندروید می پردازد. پروژه و مثال های مبحث حاضر در محیط برنامه نویسی

Eclipse 3.7 و با زبان Java 1.6 نوشته شده و مبتنی بر ویرایش 4.0 سیستم عامل اندروید می باشد.

Calendar API از ویرایش 4.0 سیستم عامل اندروید در اختیار توسعه دهندگان اپلیکیشن های اندرویدی قرار گرفت.

جهت اعلان رخدادهای جدید (event) کافی است از طریق آبجکت های Intent اقدام نمایید. برای تعریف رخداد نیازی به دریافت مجوز نیست.

برای مقداردهی property ها و ویژگی های event های مورد نظر لازم است از متغیر extra مربوط به آبجکت های Intent که برای تبادل داده های اضافی بین activity ها مورد استفاده قرار می گیرند، استفاده نمایید. اپلیکیشن شما از کاربر می پرسد آیا رخداد (event) ایجاد شود یا خیر.

به عنوان مثال، کد زیر از کاربر می پرسد آیا رخدادی با جزئیات معین ایجاد شود یا خیر.

```
// ACTION_INSERT does not work on all phones
// use  Intent.ACTION_EDIT in this case
Intent intent = new Intent(Intent.ACTION_INSERT);
intent.setData(CalendarContract.Events.CONTENT_URI);
startActivity(intent);
```

در آینده مجددا رخ دهد، می توانید تاریخ و زمان را نیز اضافه eventدر صورتی که قرار است نمایید.

```
Intent intent = new Intent(Intent.ACTION_INSERT);
intent.setType("vnd.android.cursor.item/event");
intent.putExtra(Events.TITLE, "Learn Android");
intent.putExtra(Events.EVENT_LOCATION, "Home suit home");
intent.putExtra(Events.DESCRIPTION, "Download Examples");
// Setting dates
GregorianCalendar calDate = new GregorianCalendar(2012, 10, 02);
intent.putExtra(CalendarContract.EXTRA_EVENT_BEGIN_TIME,
      calDate.getTimeInMillis());
intent.putExtra(CalendarContract.EXTRA_EVENT_END_TIME,
      calDate.getTimeInMillis());
// make it a full day event
intent.putExtra(CalendarContract.EXTRA_EVENT_ALL_DAY, true);
// make it a recurring Event
intent.putExtra(Events.RRULE, "FREQ=WEEKLY;COUNT=11;WKST=SU;BYDAY=TU,TH");
```

```
// Making it private and shown as busy
intent.putExtra(Events.ACCESS_LEVEL, Events.ACCESS_PRIVATE);
intent.putExtra(Events.AVAILABILITY, Events.AVAILABILITY_BUSY);
```


آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 651 طبقه دوم - واحد7

88146323 - 88446780 - 88146330