Alten

2018-06-14

Intro to Android

https://github.com/arneson/intro-to-android

Tonight's speaker: Simon Arneson from devies

Out with the old...

kicksort → devies

devies

2014

22

Personal Growth

Schedule

Tonight’s topic: ‘Intro to Android’

Workshop

Q&A/Discussion

devies

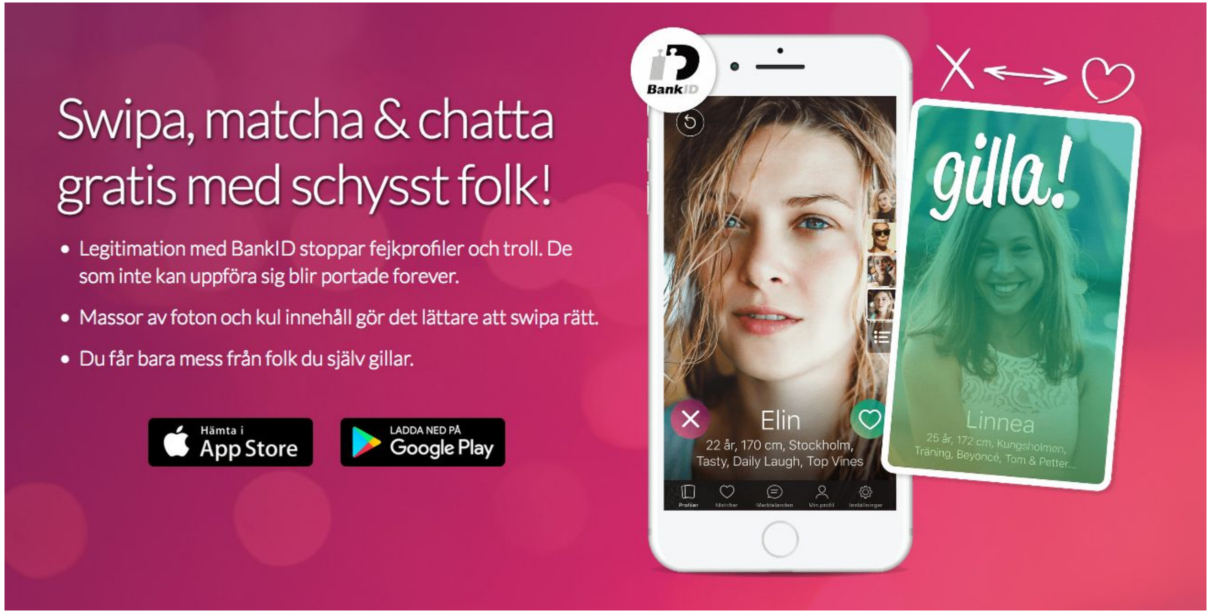
Who am I?

Simon Arneson

- Android, iOS, React (Native), Node.js, …

devies

GoodOnes


Swipa, matcha & chatta gratis med schysst folk!
• Legitimation med BankID stoppar fejkprofiler och troll. De som inte kan uppföra sig blir portade forever.
• Massor av foton och kul innehåll gör det lättare att swipa rätt.
• Du får bara mess från folk du själv gillar.
Hämta i App Store
LADDA NED PÅ Google Play
BankID
Elin
22 år, 170 cm, Stockholm, Tasty, Daily Laugh, Top Vines
gilla!
Linnea
25 år, 172 cm, Kungsholmen, Träning, Beyoncé, Tom & Petter...

devies

Matpriskollen

matpriskollen
veckans extrapriser

Android

Intro

devies

Platform & devices

devies

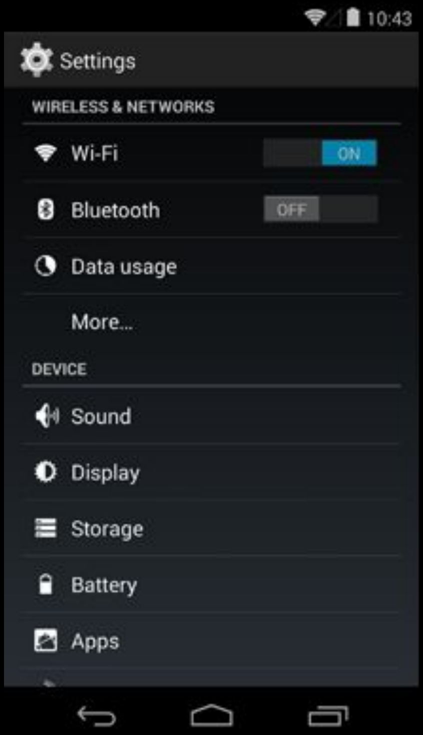
Android

- 24000 different devices (2015)
- More than 400 manufacturers (2015)
- Many different Android versions still supported
- Plus a lot of custom ROMs
- 1 billion active device that won't be updated
- Different screen sizes
- Low-end products

devies

Look and Feel

devies



10:43

Settings

WIRELESS & NETWORKS

Wi-Fi ON

Bluetooth OFF

Data usage

More...

DEVICE

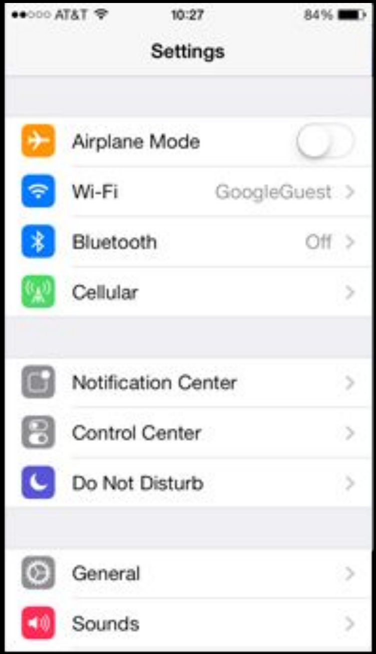
Sound

Display

Storage

Battery

Apps



●●○○○ AT&T 10:27 84%

Settings

Airplane Mode

Wi-Fi GoogleGuest

Bluetooth Off

Cellular

Notification Center

Control Center

Do Not Disturb

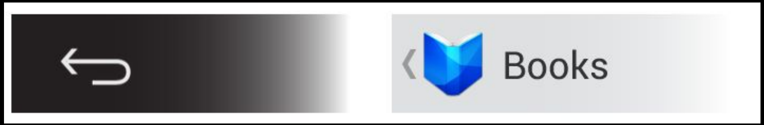
General

Sounds

devies

Android Look and Feel

- Don't mimic UI elements from other platforms
- Don't carry over platform-specific icons
- Don't hardcode links to other apps
- Don't use right-pointing carets on line items
- Don't use labeled back buttons on action bars



Handle back stack

- “Physical” back button (always visible)
- Activities handle back stack automatically
- Otherwise you have to manage it yourself

devies

Google Play Services

- Google Maps
- Google Drive
- Google Location
- Google Wallet
- Android Wear
- And many more...

https://developers.google.com/android/guides/setup#add_google_play_services_to_your_project

devies

Grundläggande byggstenar

- Context
- Activity
- Service
- Application

- AndroidManifest
- Fragment
- View
- Resources

Components

devies

Activity

- Corresponds to ViewController on iOS
- Usually represents a single screen

devies

Fragment

- Lifecycle-aware component that is tied to an Activity

Custom view

- Corresponds to UIView on iOS

- No right or wrong!
- Multiple Activities
- Single Activity -> multiple Fragments
- Custom views
- Architecture components

devies

Lifecycle

devies

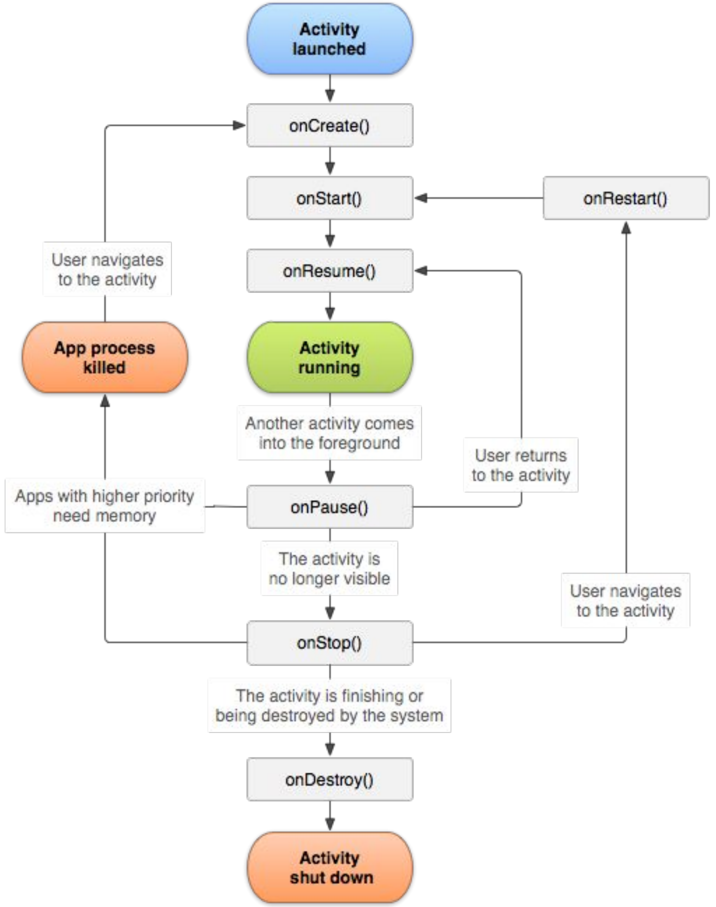
iOS Lifecycle

- ViewDidLoad
- ViewWillAppear
- ViewDidAppear
- ViewWillDisappear
- ViewDidDisappear

React Lifecycle

- contructor
- render
- willReceiveProps
- shouldComponentUpdate
- etc.

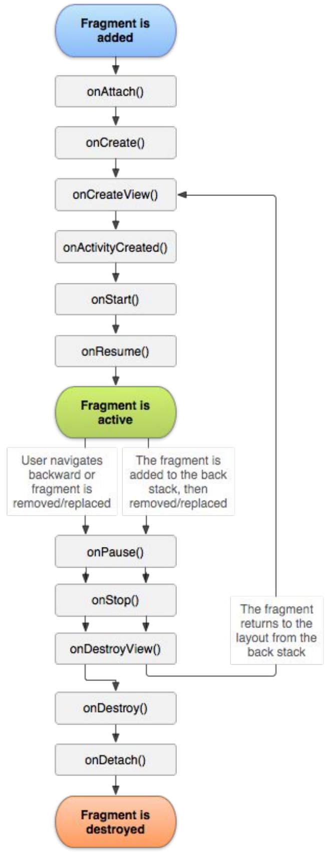
Activity Lifecycle



Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

devies

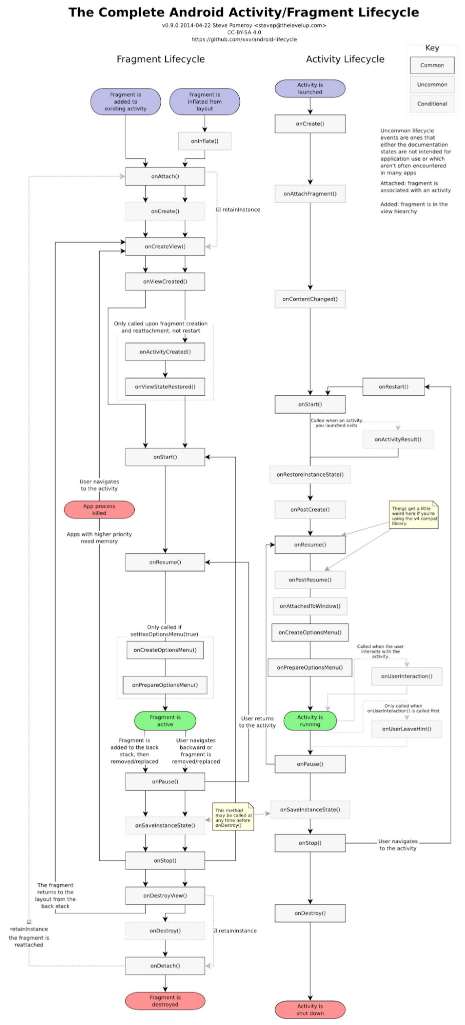
Fragment Lifecycle



devies

Complete Android Lifecycle

https://i.stack.imgur.com/1lIRw.png

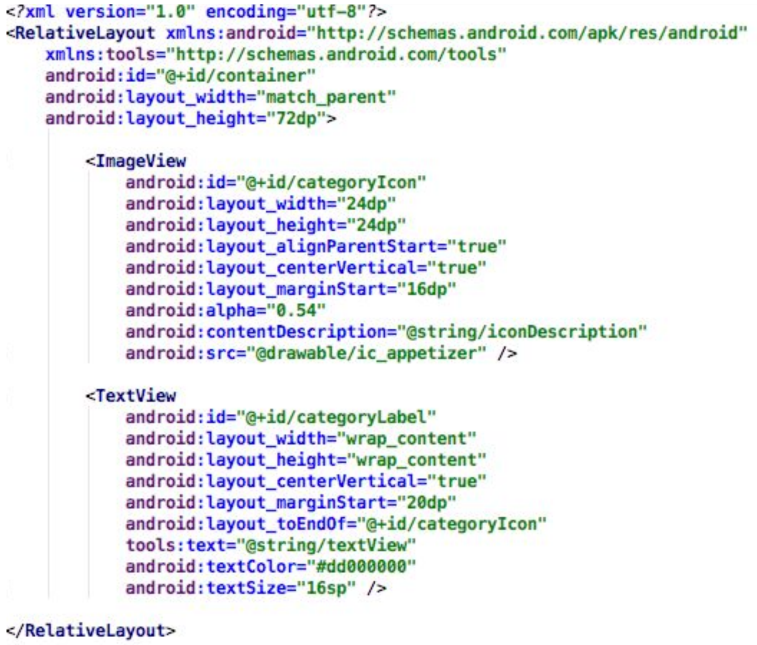

The Complete Android Activity/Fragment Lifecycle
v0.9.0 2014-04-22 Steve Pomeroy <stevep@thelevelup.com>
CC-BY-SA 4.0
https://github.com/xxv/android-lifecycle
Fragment Lifecycle
Activity Lifecycle
Key
Common
Uncommon
Conditional
Uncommon lifecycle events are ones that either the documentation states are not intended for application use or which aren't often encountered in many apps
Attached: fragment is associated with an activity
Added: fragment is in the view hiearchy
Fragment is added to existing activity
Fragment is inflated from layout
onInflate()
onAttach()
onCreate()
retainInstance
onCreateView()
onViewCreated()
Only called upon fragment creation and reattachment, not restart
onActivityCreated()
onViewStateRestored()
onStart()
User navigates to the activity
App process killed
Apps with higher priority need memory
onResume()
Only called if setHasOptionsMenu(true)
onCreateOptionsMenu()
onPrepareOptionsMenu()
Fragment is active
Fragment is added to the back stack, then removed/replaced
User navigates backward or fragment is removed/replaced
User returns to the activity
onPause()
This method may be called at any time before onDestroy()
onSaveInstanceState()
onStop()
The fragment returns to the layout from the back stack
onDestroyView()
retainInstance
the fragment is reattached
onDestroy()
retainInstance
onDetach()
Fragment is destroyed
Activity is launched
onCreate()
onAttachFragment()
onContentChanged()
onRestart()
onStart()
Called when an activity you launched exits
onActivityResult()
onRestoreInstanceState()
onPostCreate()
Things get a little weird here if you're using the v4 compat library.
onResume()
onPostResume()
onAttachedToWindow()
onCreateOptionsMenu()
Called when the user interacts with the activity.
onPrepareOptionsMenu()
onUserInteraction()
Only called when onUserInteraction() is called first
Activity is running
onUserLeaveHint()
onPause()
onSaveInstanceState()
onStop()
User navigates to the activity
onDestroy()
Activity is shut down

Android Lifecycle

- Nothing is guaranteed to remain in memory
- Implement functionality to save and restore state
- Rotating the device will also create a new activity!

Layout

XML Editor

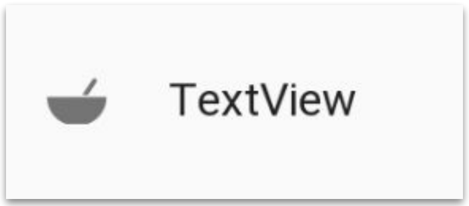


```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="72dp">

        <ImageView
            android:id="@+id/categoryIcon"
            android:layout_width="24dp"
            android:layout_height="24dp"
            android:layout_alignParentStart="true"
            android:layout_centerVertical="true"
            android:layout_marginStart="16dp"
            android:alpha="0.54"
            android:contentDescription="@string/iconDescription"
            android:src="@drawable/ic_appetizer" />

        <TextView
            android:id="@+id/categoryLabel"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerVertical="true"
            android:layout_marginStart="20dp"
            android:layout_toEndOf="@+id/categoryIcon"
            tools:text="@string/textView"
            android:textColor="#dd000000"
            android:textSize="16sp" />

</RelativeLayout>
```



devies

Layouts

- RelativeLayout
- LinearLayout
- ConstraintLayout

devies

Gradle

What is Gradle?

- DSL based on Groovy
- Built-in dependency management through Maven and/or Ivy
- Flexible!
- Uses Plug-in system for custom task API:s
- Gradle wrapper gives same Gradle version for everyone.
- minSdkVersion and targetSdkVersion should be set in gradle file

http://tools.android.com/tech-docs/new-build-system/user-guide

devies

AndroidManifest.xml

- Every app must have one
- Must be named AndroidManifest.xml
- Names the Java package
- Describes components
- Declares permissions

http://developer.android.com/guide/topics/manifest/manifest-intro.html

devies

What is Context?

- Interface to global information about an application environment
- Access resources
- Communicate with other parts of the app
    - Broadcasts
    - Intents
    - Start new activities
- Activity and Application are subclasses

Application

- Extends Context
- Global singelton
- Subclasses must be declared in Manifest
- Context.getApplicationContext()

http://developer.android.com/reference/android/app/Application.html

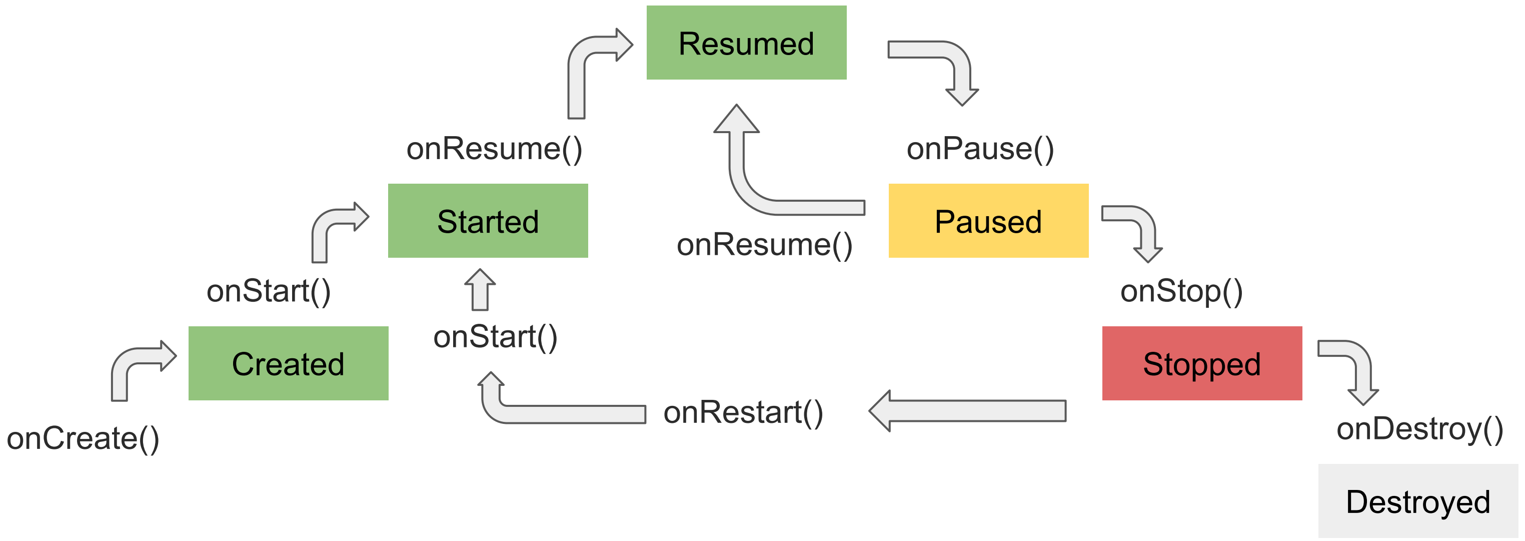
devies

Activity

- Main building block for Android apps
- One Activity == One screen (or one app)
- More controller then view
- Entry point to an application
- Must be declared in the Manifest

http://developer.android.com/reference/android/app/Activity.html

The Activity Lifecycle



http://developer.android.com/shareables/training/ActivityLifecycle.zip

Creating an Activity

AndroidManifest.xml

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

http://developer.android.com/guide/topics/manifest/activity-element.html

devies

Creating an Activity

MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
   Button button = (Button) findViewById(R.id.button);
 }
```

http://developer.android.com/reference/android/app/Activity.html

devies

Creating an Activity

activity_main.xml

```
<RelativeLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent">
        <Button
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:text="Button"
          android:id="@+id/button"
          android:layout_centerInParent="true"/>
</RelativeLayout>
```

http://developer.android.com/guide/topics/ui/declaring-layout.html

Explicit intents

```
// AnActivity.java

Intent intent = new Intent(context, AnotherActivity.class);

intent.putExtra(EXTRA_TEXT, text);
context.startActivity(intent);

// AnotherActivity.java

String theText = "";
Intent intent = getIntent();

if (intent.hasExtra(EXTRA_TEXT)) {
        theText = intent.getStringExtra(EXTRA_TEXT);
}
```

Implicit intents

```
// Setup intent

Uri number = Uri.parse("tel:5551234");
Intent callIntent = new Intent(Intent.ACTION_DIAL,
number);
context.startActivity(intent);

// Check if intent is safe

List activities =
packageManager.queryIntentActivities(intent,
PackageManager.MATCH_DEFAULT_ONLY);

boolean isIntentSafe = activities.size() > 0;
```

http://developer.android.com/reference/android/content/Intent.html

View

- Basic building block for UI:s
- Base class for interactive UI components
- ViewGroup base class for layouts
- Responsible for drawing and event handling
- XML properties

http://developer.android.com/reference/android/view/View.html

devies

Commonly used View subclasses

TextView - Show text to the user
EditText - Let’s the user input text. Opens keyboard when focussed
Button - Clickable view
ImageView - Show image
ViewGroup - A View subclass that can hold child views

ViewGroup and subclasses
LinearLayout - Child views layed out vertically or horizontally
RelativeLayout - Layout child views relative to each other.

http://developer.android.com/reference/android/view/View.html

Layout XML

```
// activity_layout_example.xml
<?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:text="New Text"
      android:id="@+id/textView2"
      android:layout_gravity="center_horizontal"/>
</LinearLayout>
// LayoutExampleActivity.java
setContentView(R.layout.activity_layout_example);
```

devies

Attributes breakdown

Mandatory attributes
```
android:layout_width="match_parent"  || wrap_content || dp
android:layout_height="match_parent" || wrap_content || dp
```

Semi mandatory
```
android:id="@+id/textView2"
```

Class specific attributes for TextView
```
android:text="New Text"
```

Class specific attributes for ImageView
```
android:src="@drawable/fancy_image"
```

Class specific child attributes RelativeLayout)
```
android:layout_alignParentBottom="true"
android:layout_alignTop="@id/someViewId"
android:layout_above="@id/someViewId"
```

Programmatically

```
// LayoutExampleActivity.java
LinearLayout layout = new LinearLayout(context);
LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(ViewGroup.LayoutParams.MATCH_PARENT,
                        ViewGroup.LayoutParams.WRAP_CONTENT );
layout.setLayoutParams(params);
layout.setBackgroundColor(Color.BLUE);

LinearLayout.LayoutParams textParams = new
          LinearLayout.LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT,
          ViewGroup.LayoutParams.WRAP_CONTENT);
TextView textView = new TextView(context);
textView.setLayoutParams(textParams);
textView.setText("Programmatically added view");

layout.addView(textView);
```

Dynamic inflation

```
// simple_layout.xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:background="@android:color/holo_green_dark"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
  <TextView
     android:text="Dynamically inflated"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"/>
</LinearLayout>
```

```
// LayoutExampleActivity.java
LayoutInflater inflater = getLayoutInflater();
inflater.inflate(R.layout.simple_layout, root);
```

devies

Service (sort of depricated)

- Extends Context
- Meant for long running operations
- Can live without Activity
- Runs on Main Thread!
- Must be declared in Manifest

http://developer.android.com/reference/android/app/Service.html

Fragment

- Since api 11
- Controller
- Can be placed in Activity
- Can be nested (since api 17)

http://developer.android.com/reference/android/app/Fragment.html

Resources

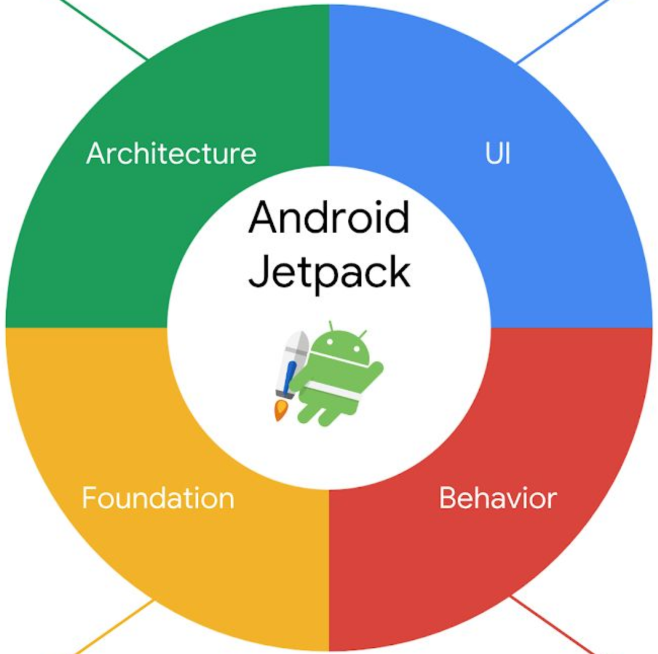
- Layouts
- Strings
- Animations
- Styles
- Colors
- Compiled in gen/R.java

Simulator vs Emulator

- iOS simulator
  - Runs native iOS code on the Macs CPU
  - Better performance
  - The simulator just gives access to iOS API:s
  - Possible because Mac OS and iOS have many things in common
- Android emulator
  - A virtual machine that emulates the exact Android environment
  - More accurate environment

Data Binding
Lifecycles
LiveData
Navigation *new!*
Paging *new!*
Room
ViewModel
WorkManager *new!*

Animation & Transitions
Auto, TV & Wear
Emoji
Fragment
Layout
Palette



AppCompat
Android KTX *new!*
Multidex
Test

Download Manager
Media & Playback
Permissions
Notifications
Sharing
new! Slices

devies

Workshop

devies

Final words

- Java vs Kotlin
- With great power comes great responsibility
- Don’t mimic iOS
- Stick to RelativeLayout and LinearLayout

devies

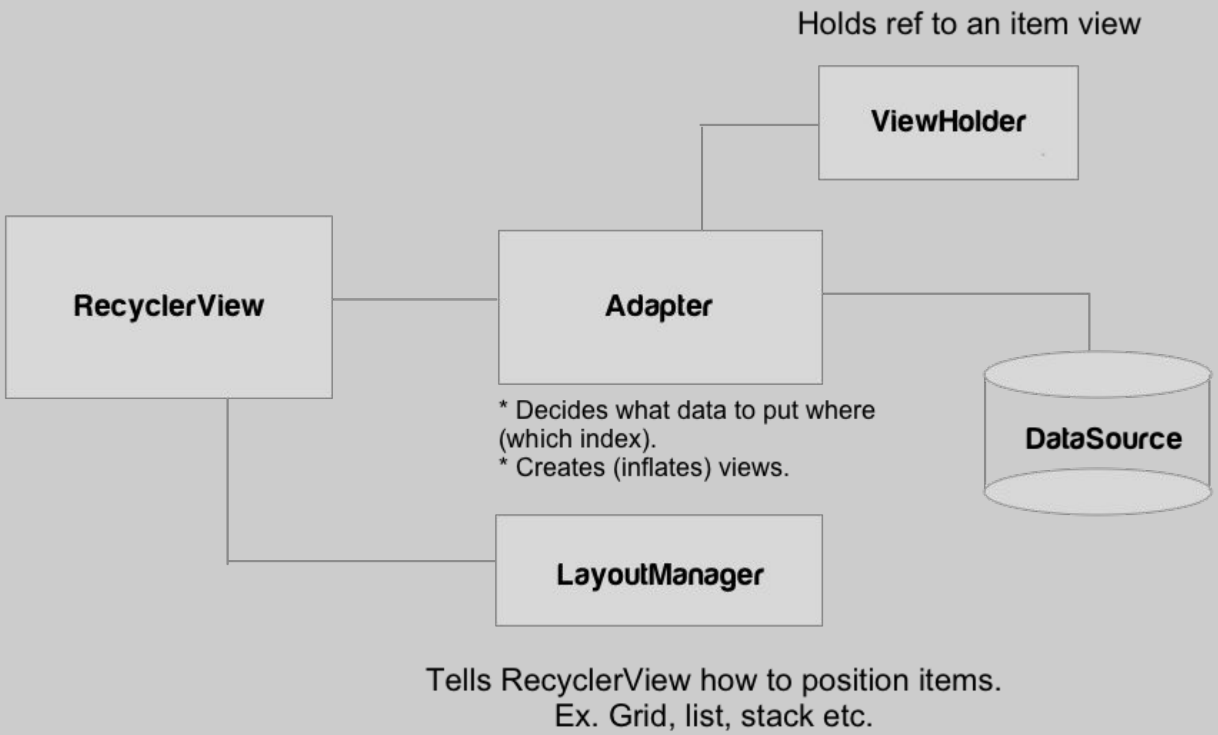
Q&A

Bonus slide! Save and restore state

```
@Override
public void onSaveInstanceState(Bundle savedInstanceState) {
	super.onSaveInstanceState(savedInstanceState);
	savedInstanceState.putBoolean("MyBoolean", true);
}

@Override
public void onRestoreInstanceState(Bundle savedInstanceState) {
	super.onRestoreInstanceState(savedInstanceState);
	boolean myBoolean = savedInstanceState.getBoolean("MyBoolean");
}
```

RecyclerView+Adapter+ViewHolder pattern


Holds ref to an item view
ViewHolder
RecyclerView
Adapter
DataSource
* Decides what data to put where (which index).
* Creates (inflates) views.
LayoutManager
Tells RecyclerView how to position items.
Ex. Grid, list, stack etc.