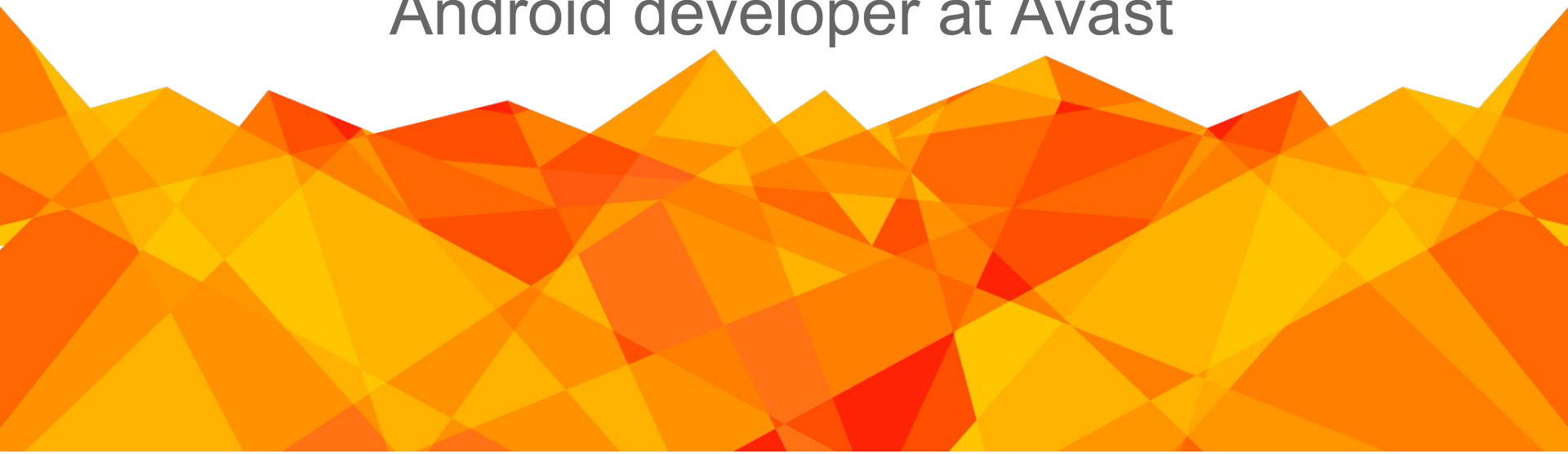
Android History & Basics

Lukas Prokop
Android developer at Avast



Android history

- October 2003 - Android Inc.
  - Intended as software for digital cameras
- July 2005 - Android Inc. acquired by Google
- November 5, 2007 - Open Handset Alliance
  - Consortium of technology companies (Google), manufacturers (HTC, Sony,...), carriers (Sprint, T-mobile,..) and chipset makers (Qualcomm, …)
- October 22, 2008 - T-mobile G1 (HTC)

T-Mobile G1 (HTC Dream)

Android

- Linux based OS
  - No bash, no access to root (by default)
  - Missing glibc (standard C library)
- Open source
  - https://source.android.com/
  - Just the OS, not the Play store or Google play services

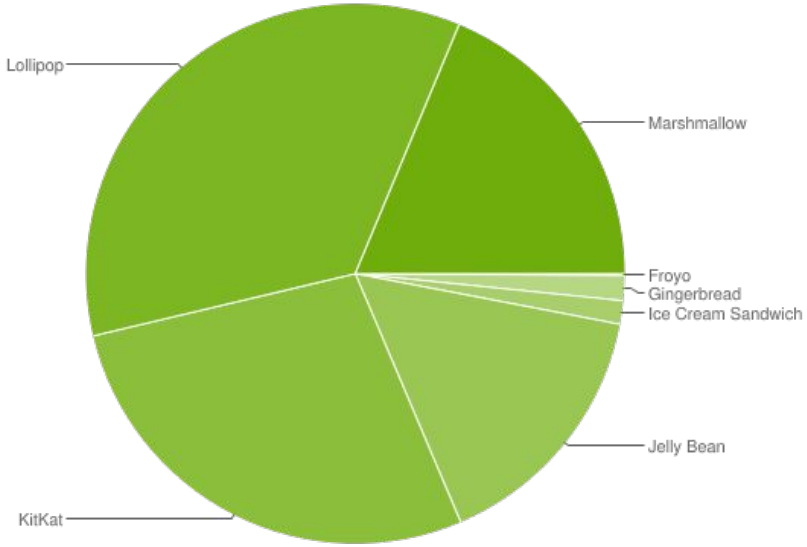
Android versions

- 1.0 - 23 September 2008 (API 1)
- 1.1 - 9 February 2009 (API 2)
- 1.5 - Cupcake 27 April 2009 (API 3)
- 1.6 - Donut 26 October 2009 (API 4)
- 2.0 -2.1 - Eclair (API 5 - 7)
- 2.2 - Froyo (API 8)
- 2.3.x - Gingerbread (API 9 - 10)

Android versions

- 3.x - Honeycomb (API 11 - 13) - Tablet only
- 4.0.x - Ice cream sandwich (API 14-15)
- 4.1 - 4.3 - Jelly Bean (API 16 - 18)
- 4.4 - Kitkat (API 19)
- 4.4W - Kitkat for wearables (API 20)
- 5.0 - 5.1 - Lollipop (API 21 - 22)
- 6.0 - Marshmallow (API 23) October 2015
- 7.0 - Nougat (API 24) - August 2016

Android today - versions



Version	Codename	API	Share
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	1.5%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.4%
4.1.x	Jelly Bean	16	5.6%
4.2.x		17	7.7%
4.3		18	2.3%
4.4	KitKat	19	27.7%
5.0	Lollipop	21	13.1%
5.1		22	21.9%
6.0	Marshmallow	23	18.7%

Data from September 5, 2016
https://developer.android.com/about/dashboards/index.html version < 0.1% are not shown

Android today

- Android - phones and tablets
- Android wear - smartwatch
- Android TV - televisions, consoles
- Android auto - cars
- Google glass - currently suspended project

Android ecosystem

- Apps are available on Google Play store
- Paid and free apps
- Subscription
- In-app purchases
- Android users not like to pay for apps => a lot of advertisements
- Available in most countries

Mobile devices specifics

- Not enough computation power
- Changing state - screen orientation
- Non stable network connection
- Small battery

Android “issues”

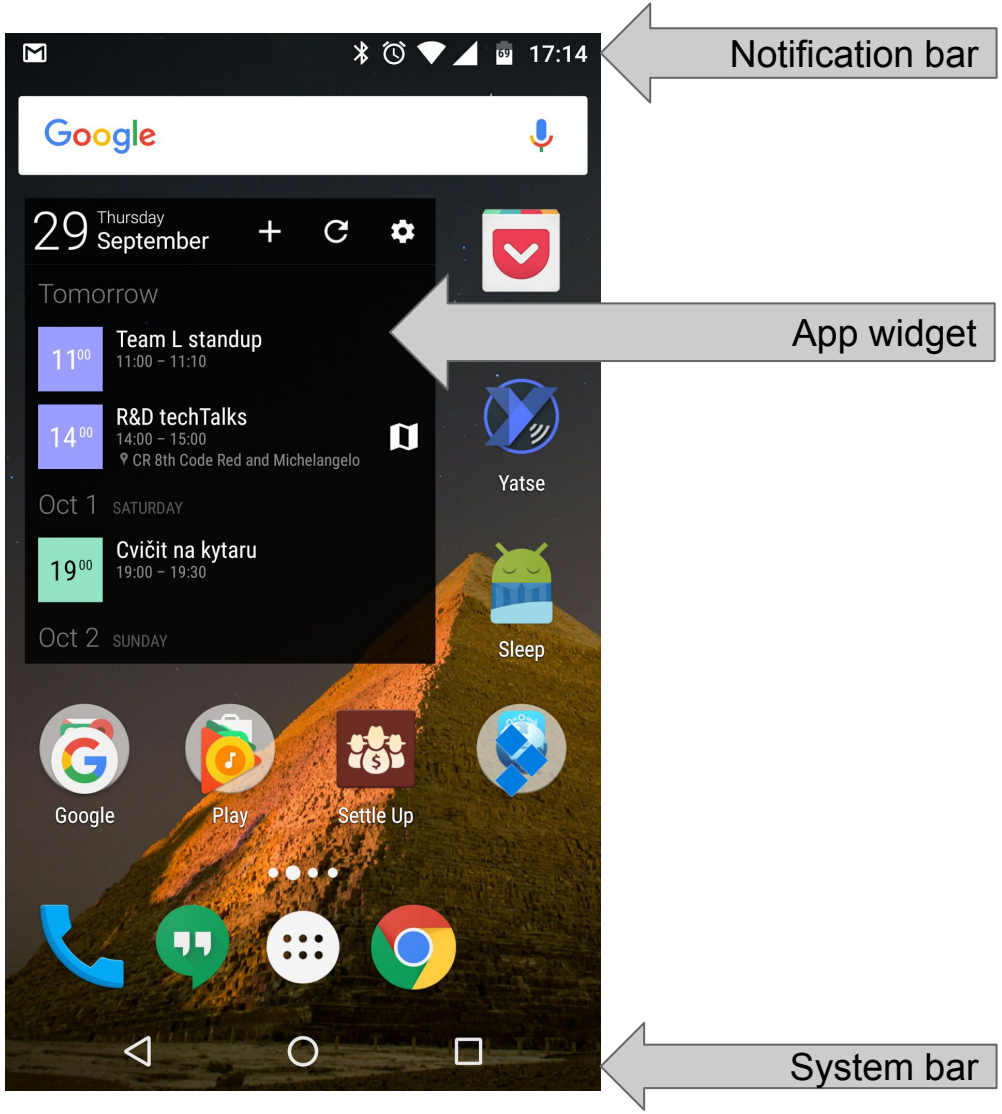
- Fragmentation, slow upgrades, manufacturer changes (TouchWiz, Emotion UI)
  - http://opensignal.com/reports/2015/08/android-fragmentation/
- Low-end devices
- Low-quality apps in Google play, malicious apps
  - Solar charger app

Android - positives

- Open source
- Lot of users
- Lot of apps
- Freedom (keyboard, launcher, …)
- Developers one time fee 25$
- Dev tools available for Linux, Windows and Mac
- A lot of OSS libraries
  - Retrofit, Dagger, Styled dialogs, Butterknife, Stetho
- Nexus/Pixel devices

Description

What is what - Launcher



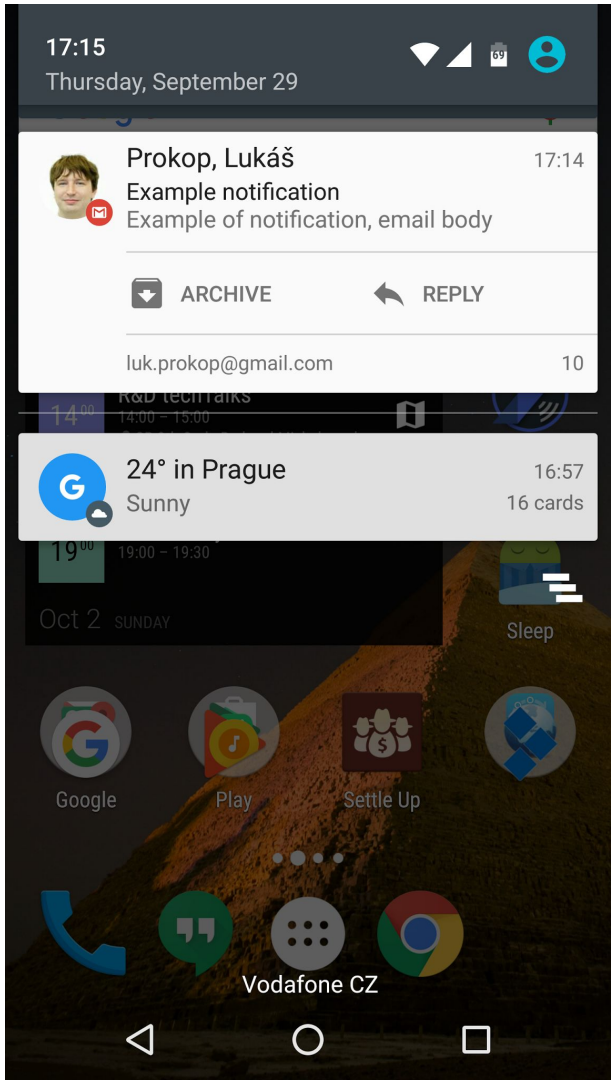
Notification bar

App widget

System bar

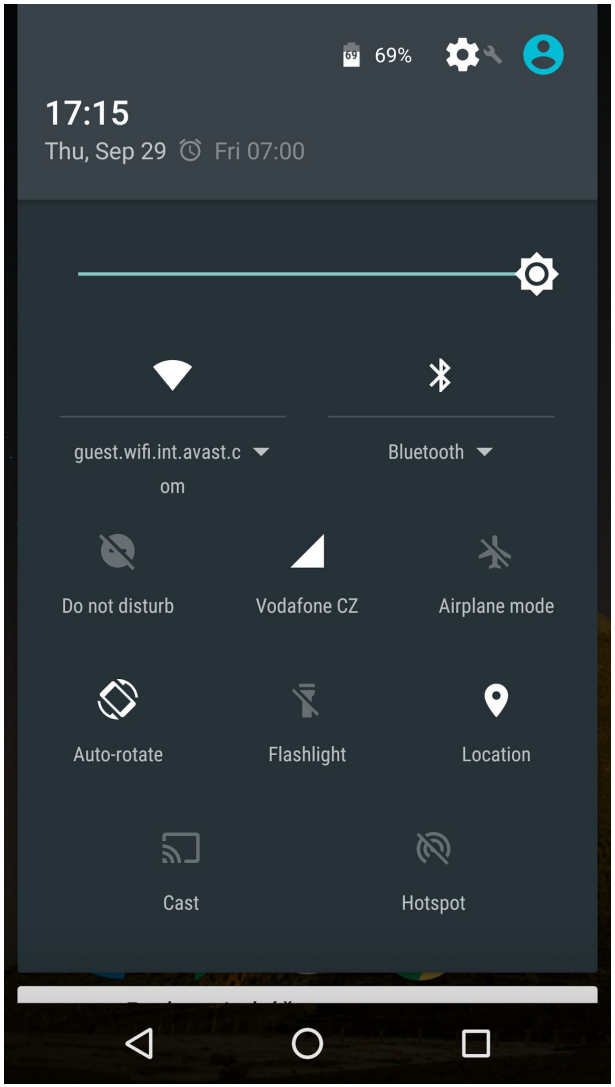
Warning !
Widgets vs. App widgets
AppWidget - on launcher
Widget - GUI item (Button, Checkbox, …)

What is what - Notifications



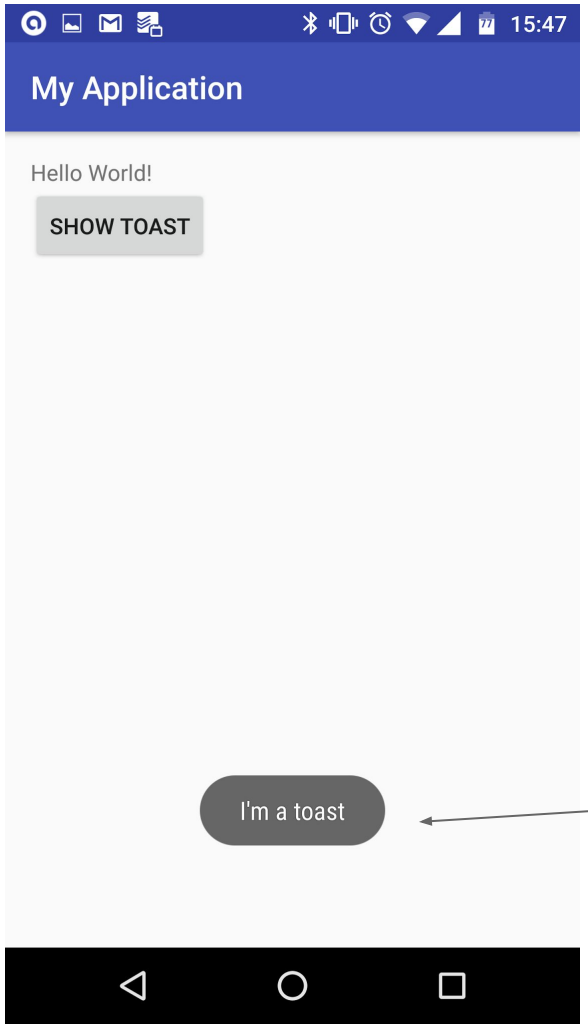
- Inform user (email, progress, …)
- Actions in notifications since API-16
- Notifications are visible on lockscreen
- Optional sound, vibration, LED light

What is what - Quick settings



- Most of vendors added this themselves before it became part of AOSP (API-8)
- Available since API-16 as part of AOSP
- Since API-24 (latest) possibility to add custom action

What is what - other



Toolbar (actionbar)

Toast - short message to user

Material design

- Design language from Google
- Metaphor for real world materials
- Says how the materials can interact, use elevation of material
- Not only for android

https://material.google.com/

Android - security

- Root user is not available by default
  - Rooting devices to get more features, often void warranty
- Permissions
  - Until API-23 before installation “all or nothing”
  - Since API-23 permissions are requested at runtime
  - Empty data or crash, depend on TargetSDK
- Bouncer
  - Service that scan google play store for malicious apps

Development options

- App-like mobile web
- Other language frameworks (Xamarin, Kotlin)
- C-based frameworks (Unity)
- WebView based frameworks (PhoneGap)
- Native
  - Java
  - C/C++ (mainly games and libs)

Development tools

- Android studio
- SDK
  - Adb
  - Lint, Uiautomatorviewer, Emulator
  - Aapt, aidl, dx, jack & jill
- NDK (C/C++ development)
- Gradle
  - Groovy based build system
  - Flexible
  - Extensible

Components

AndroidManifest

- Essential information about app for system
- Contains package name (unique id of app)
- Describes components
- Declares permissions
- Declares min required API level
- Declares supported screens, features
- Filtering in play store

Activity

- Screen with user interface
- The only visual component
- Typically one activity per screen in application
- Activity stack
- Lifecycle
- Contains fragments and views

Service

- No UI
- Optional notification (mandatory for foreground services)
- Long-running tasks
  - Download service
  - Music playback

Content provider

- Manages and shares app data
- Don’t depend od data storage (db, files, web)
- Data can be queried or modified
- Optional permissions
- System dbs
  - SMS
  - Contacts
  - Call log

Broadcast receivers

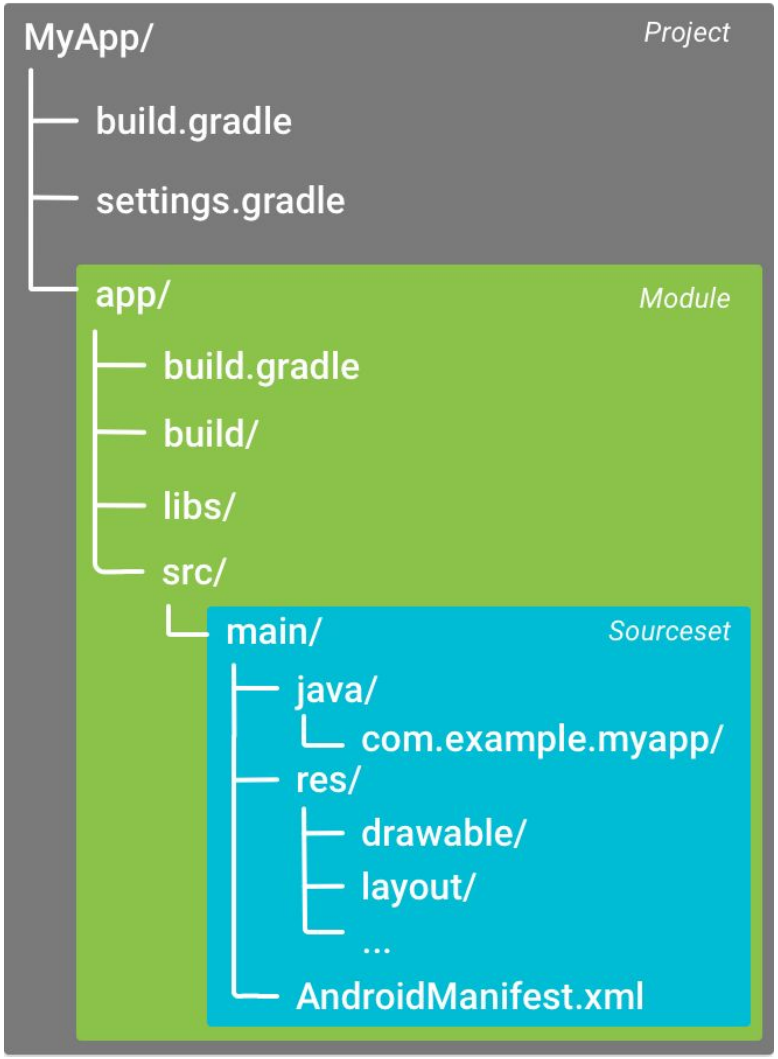
- Listens for broadcast from system or other application
- Static or dynamic registration
- Broadcasts are system wide
- Examples
  - Incoming SMS
  - Low battery, battery percentage changed
  - Connectivity change
  - Headphones connected/disconnected

Intent

- Asynchronous message between components
- Starts activities
- Stars or bind Services
- Sends broadcasts

Hello World

Project structure



Project

- Common configuration for modules
  - Common dependencies versions

Module

- Application or library
- Different modules for phone and watch
- Multiple source sets (optional)
  - Different version of same app (free vs. paid)

Sourceset

- Source code and resources
- Source code form main source set is available in other source sets
- Resources can be overridden in different source sets

Project files

build.gradle

- Configuration that applies to all modules
- Defines android build plugin version
- List of repositories where to download dependencies and gradle build plugins

settings.gradle

- List of modules to build

Project files - gradle.properties

- Project wide gradle fields
- Customization of how it will run
  - Heap size
  - Daemon or not
  - Java_home and java arguments
  - Parallel run
  - Proxy
  - And much more
- Common fields for all modules
  - Version of shared dependencies across modules

Project files - local.properties

- Contains paths to sdk and ndk
- Can’t be shared between developers
- Generated by android studio, do not modify it manually

Module files - build.gradle

- Configure build setting for specific module
- Defines build variants and their source sets
- applicationID
- Min and target SDK version
- compileSdkVersion and buildToolsVersion
- Dependencies

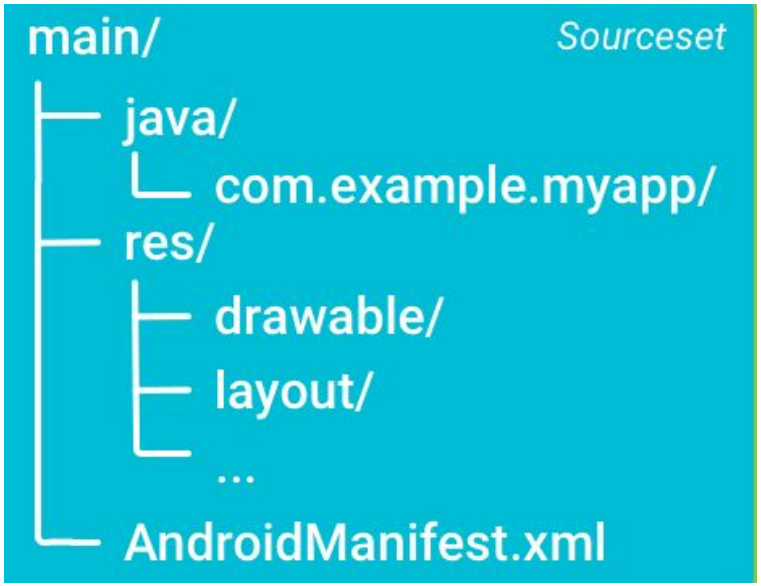
Module files - libs/

- *.jar libraries
- If it is possible use library as gradle dependency

Module files - src/

- Source code
- Resources
- Assets
- Main - default sourceset for all build variants
- Recommended to split code into packages

Source set



- java/
  - Source codes
- res/
  - Resources
    - Drawables
    - Layouts
    - Values
    - …
- assets/
- AndroidManifest.xml

Resources

- Layout
- Strings
- Menu
- Animations
- Icons
- Dimensions
- Drawables
- Mipmap

Resource qualifiers

- Resources in different variants
- Drawable, drawable-mdpi…
- Values, values-cs, values-de
- Layout, layout-sw600dp

Resources - drawables

- Bitmaps
- 9-patch png
- State lists
- Vector drawables
  - Since API-21
  - Backward compatibility with support library

Resources - layout

- Definition how the UI will looks like

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout android:id="@+id/activity_main"
   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="16dp"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="16dp"
   tools:context="com.avast.android.helloworld.MainActivity">

   <TextView
       android:id="@+id/txt_headline"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:text="Hello World!" />
</LinearLayout>
```

Resources units

- Dp - density independent pixel
  - On 160dpi screen 1dp = 1px
- Sp - scale independent pixel (fonts)
  - Similar to dp, but scaled by the user's font size preference
- Never use px

Binding between resources and java

- XML elements has id generated in R.java
- R.id.txt_headline
- R.layout.activity_main

Layouts

- Extends from ViewGroup
- Defined in XML or programmatically
- In folder res/layout

Types:

- `FrameLayout` – simplest, used for placeholders
- `LinearLayout`
- `RelativeLayout`
- `TableLayout`
- `GridLayout`
- `ConstraintLayout` - introduced on GoogleIO 2016, beta

Layouts - LinearLayout

- Places childs vertically or horizontally
- Possible to use weight to size item in some ratio

Layout - RelativeLayout

- Place views relatively to each other
  - Item A is left of Item B
  - Item B is aligned to its parent
- Flatten hierarchy
- Till API 17 needs to be defined separately for RTL languages, API-17 is buggy
- For complex layouts faster than LinearLayout

Layout - FrameLayout

- Places all items in top left corner
- Usage as placeholder for other view/fragment
- Fast

Widgets

- Extends View
- width and height needs to be set
  - Can be replaced by weight
  - Can be defined as
    - `match_parent`
      - Fills the whole width/height of parent
    - `wrap_content`
      - Wraps around the content

Widgets

- Button
- TextView
- EditText
- ImageView
- CheckBox
- RadioButton
- WebView
- AdapterView
  - ListView
  - Spinner

Activity

- Presentation layer of application
- Only UI component
- Contains Views or Fragment
- Every activity defined in manifest
- Runs on UI thread
- All components run in one process by default
- Lifecycle
- Activity back stack

Starting activity

- Intent describes which activity to start
- Can contain some data for activity (extras)
- Flags - manipulation with activity stack

```
Intent i = new Intent(MainActivity.this, SecondActivity.class);
i.putExtra("key", "value");
i.putExtra("keyInt", 1);
startActivity(i);
```

Explicit vs. implicit intent

- Explicit intent
  - Specify component by fully qualified class name
  - Typically component in our application
- Implicit intent
  - Just declare general action to perform
  - Enables multiple apps to handle that action
  - Examples
    - Send email - `ACTION_SEND`
    - Open browser - `ACTION_VIEW`
  - If multiple apps are capable to handle intent, system shows picker
  - Intent filters defined in manifest

Starting activity for result

```
private void pickContact() {
    // Create an intent to "pick" a contact, as defined by the content provider URI
    Intent intent = new Intent(Intent.ACTION_PICK, Contacts.CONTENT_URI);
    startActivityForResult(intent, PICK_CONTACT_REQUEST);
}
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    // If the request went well (OK) and the request was PICK_CONTACT_REQUEST
    if (resultCode == Activity.RESULT_OK && requestCode == PICK_CONTACT_REQUEST) {
        // Perform a query to the contact's content provider for the contact's name
        Cursor cursor = getContentResolver().query(data.getData(),
        new String[] {Contacts.DISPLAY_NAME}, null, null, null);
        if (cursor.moveToFirst()) { // True if the cursor is not empty
            int columnIndex = cursor.getColumnIndex(Contacts.DISPLAY_NAME);
            String name = cursor.getString(columnIndex);
            // Do something with the selected contact's name...
        }
    }
}
```

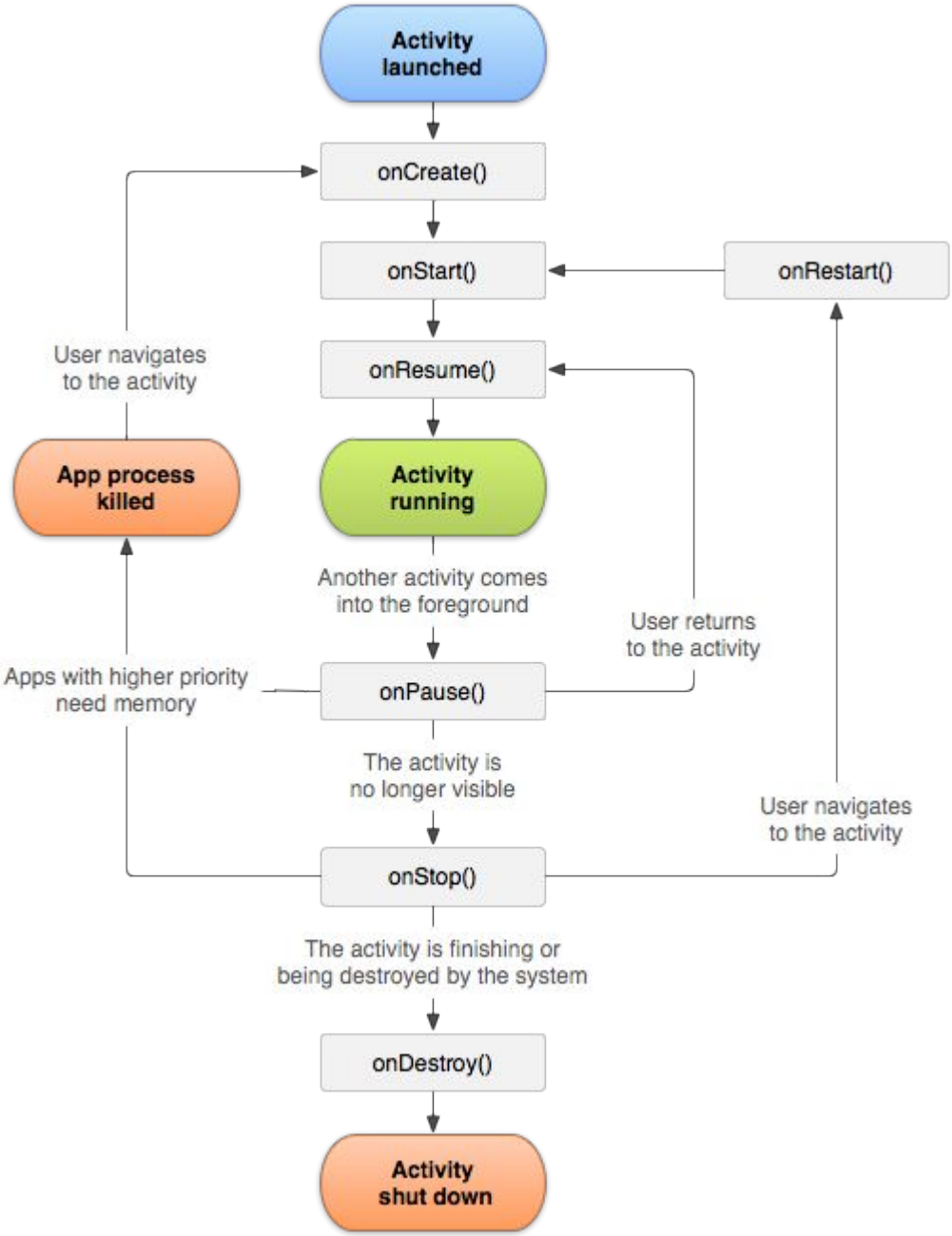
Starting activity for result

- Request `Activity#startActivityForResult(intent, requestCode)`
- Response delivered in `Activity#onActivityResult(int requestCode, int resultCode, Intent data)`
- Set result for calling activity by `Activity#setResult`

Activity - states

- Resumed
  - Running, is visible
- Paused
  - Partially visible, remains in memory
- Stopped
  - Different activity is on top
  - Moved to background
  - Still alive, remains in memory, but hosting process can be killed
- Destroyed

Activity - lifecycle



Activity#onCreate(Bundle)

- Activity is being created
- Create views
- Bind data to list
- Passed Bundle object contains activity previous state
- Read data from starting intent
- Always followed by `#onStart()`

Activity#onStart()

- Called before the activity become visible to the user
- Followed by
  - `onResume()` if come to the foreground
  - `onStop()` if becomes hidden
- Activity is partially visible, register listeners for changing UI

Activity#onResume()

- Called just before activity start interacts with user
- Activity is on top of activity stack
- Run stuff for user
- Always followed by `onPause()`

Activity#onPause()

- System is about to resume another activity.
- Commit unsaved changes, persist data, stop animations and CPU intensive stuff
- Should be very fast, because another activity `onResume()` wait until this finish
- Followed by
  - `onResume()` if the activity returns back to the front
  - `onStop()` if became invisible to the user
- Activity can be killed by system

Activity#onStop

- Called when is no longer visible to the user
- Being destroyed or another activity has been resumed and covering it.
- Finish stuff started in `#onStart()`
- Followed by
  - `onRestart()` - coming back to interact with user
  - `onDestroy()` - activity is going away

Activity#onDestroy

- Called before activity is destroyed
- Activity is finished by `#finish()` method
- System needs more resources (RAM)

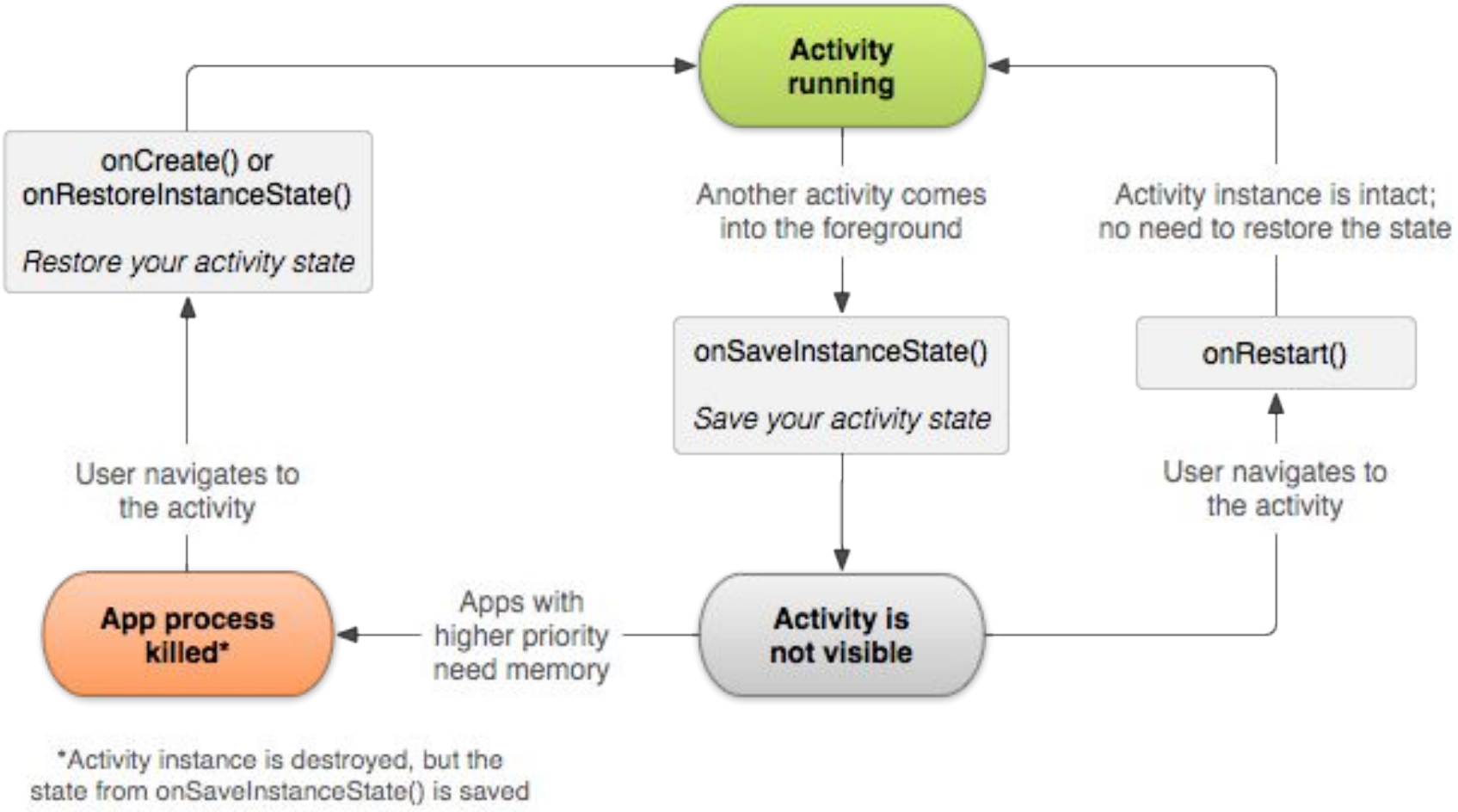
Activity#onRestart()

- Called after activity has been stopped, and before is started again

Configuration changes

- When configuration changes, activity is destroyed and recreated
  - Screen rotation
  - Language change
  - HW keyboard opens
  - Projector is connected
- Activity is destroyed and recreated
- Needs to be handled properly
  - `Activity#onSaveInstanceState`

Save activity state



Saving activity state

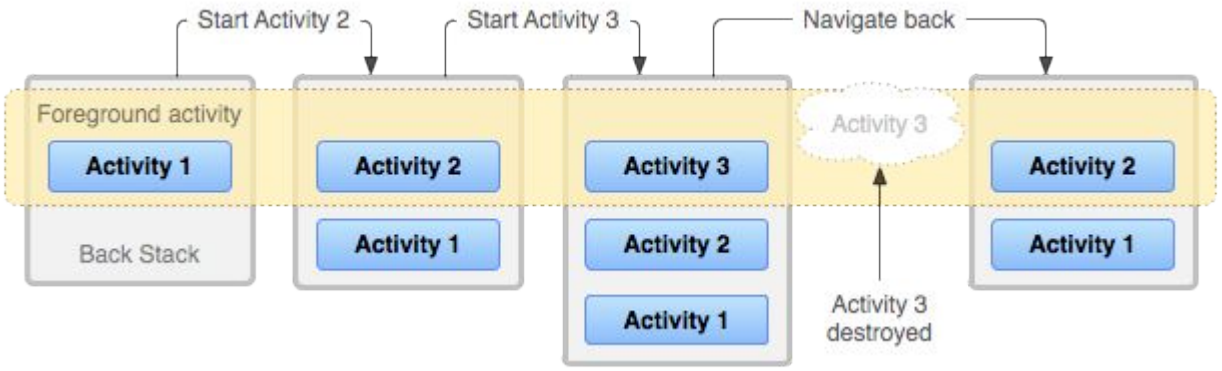
- System can kill background activity to free up resources => state of the activity is lost
- Implement `#onSaveInstanceState`
  - Called before activity is vulnerable to destruction
  - Passed Bundle is for remembering it's state
  - Bundle with the stored state is passed into `#onCreate` and `#onRestoreInstanceState` (called before `#onStart()`)
- Default implementation takes care of widget with unique id (user input), but doesn't store state (enabled/disabled)

Bundle

- Mapping parcelable and serializable objects
- String keys
- `#putString, #putInt`
- `#getString, #getInt`

Tasks and back stack

- Task is collection of activities, to perform certain job
  - Activity in task can be from different application (send email)
- Activities arranged in a stack, in order in which there were opened
- Task has it’s own back stack



Tasks and back stack

- Sometimes is necessary to change behaviour of back stack
- Manifest attributes
  - `taskAffinity`
  - `launchMode`
  - `allowTaskReparenting`
  - `clearTaskOnLaunch`
  - `alwaysRetainTaskState`
  - `finishOnTaskLaunch`
- Intent flags
  - `FLAG_ACTIVITY_NEW_TASK`
    - Start activity in new task, or bring task with that activity
  - `FLAG_ACTIVITY_CLEAR_TOP`
    - If the activity is in stack, pick them and destroy all other activities on top
  - `FLAG_ACTIVITY_SINGLE_TOP`
    - Do not start new instance of activity, if is already on top of stack

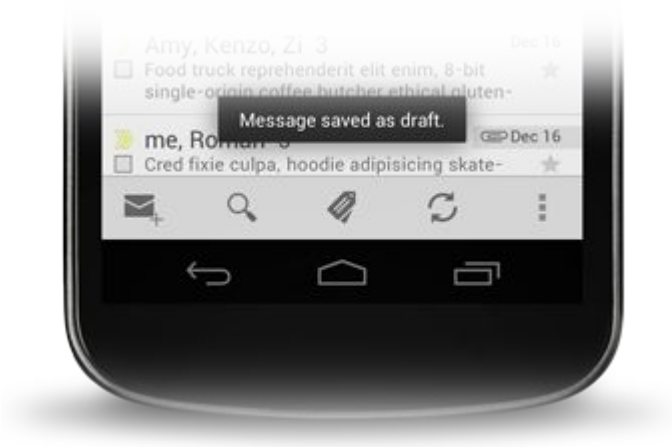
Task affinity

- If you need that flag `FLAG_ACTIVITY_NEW_TASK` open activity in new task you need to set different affinity for that activity
- It needs to be set for independent apps in one APK, we use it for debug tools (separate app which allows us to (re)set some values in main apps)

Toast

- Simple non modal information
- Displayed for short period of time
- Doesn’t have user focus

```
Toast.makeText(this, "This is a toast",
Toast.LENGTH_LONG).show();
```



Log messages

- Static method in Log class
- `Log.{v,d,i,w,e,wtf}(String tag, String message, Throwable e)`

Exercises

1. When ‘choose user’ is clicked, open `UserListActivity`
2. When user selects item, go back and return result.
3. Show selected item in EditText
4. Show EditText content in a Toast
5. Download data about user with method `GitHubApi.downloadUser()` and `parseUser()` when ‘Show user detail’ is clicked

Exercises

6. Do the same, but use `GitHubApi.downloadUserMock()` instead of `GithubApi.downloadUser()`
7. Show name, repo URL and number of repos in TextViews on `MainActivity`
8. Add button ‘Open web’ which opens website of the repo in the browser
9. Make sure the rotation works - selected user shouldn’t disappear after rotation.

Exercises

10. Add Czech localization (or just translation to other language)
11. Add landscape layout
12. Experiment!!!

Thank You

Q&A?

Feedback is appreciated

prokop@avast.com

Please use [mff-android] in subject