Android lecture 2

Basics, Activity

Agenda

- Android UI - overview
- Project structure
  - Gradle
  - Source code
  - Resources
- Android components
  - AndroidManifest
  - Activity
  - Service
  - Content provider
  - Broadcast receiver
  - Intent

avast

Android - UI overview

Launcher

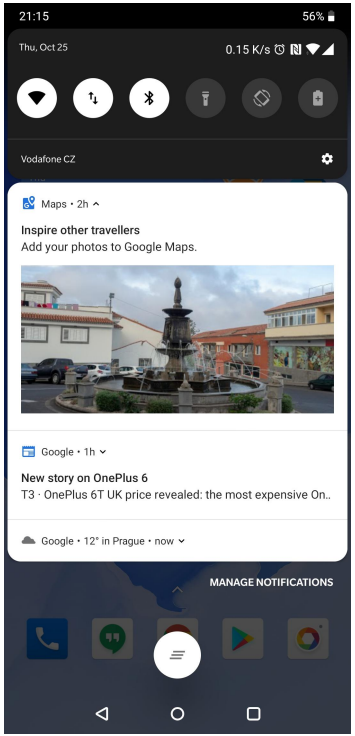
Status bar

App widget

App widget vs. widget
Widget is UI item
(button, checkbox, etc)

System bar

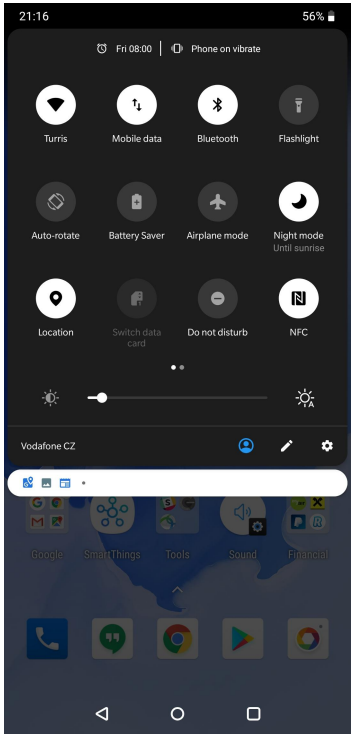
Notifications



- Inform user (email, progress, etc)
- Actions in notifications since API-16
- Can be visible on lock screen
- Optional sound, vibration, LED light

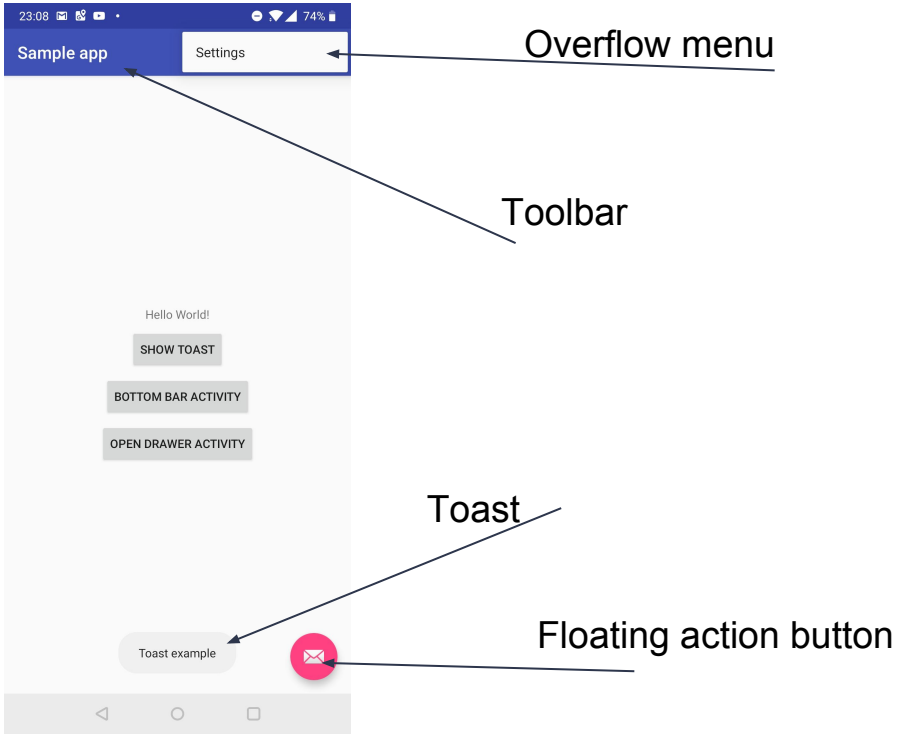
avast

Quick settings

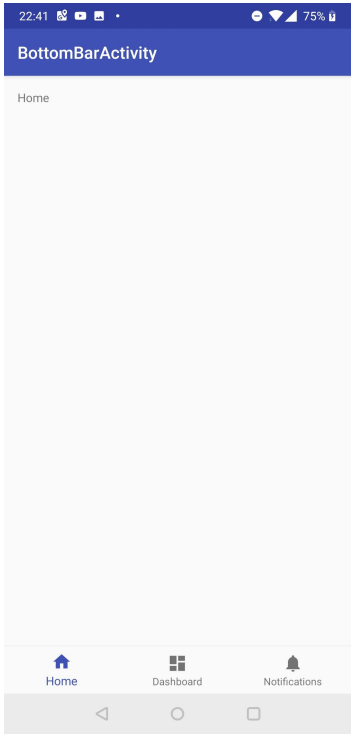


- Since API-16 part of AOSP
- Custom actions since API-24

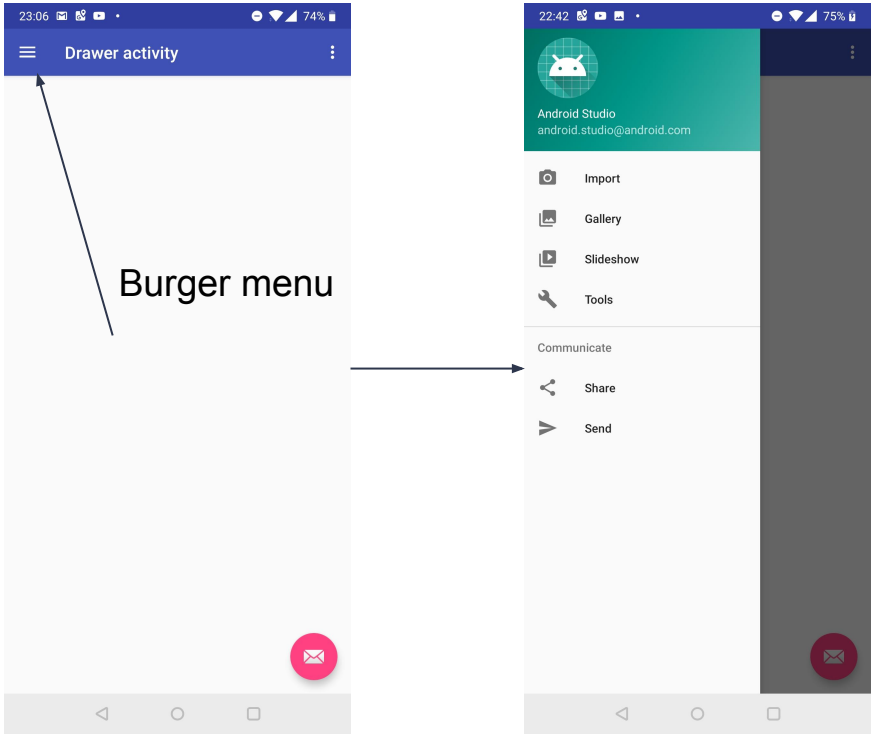
Navigation in app



Navigation in app - bottom bar



Navigation in app - drawer



Material design

- Design language from Google
- Material is the metaphor
  - Inspired by physical world (reflecting light, cast shadow)
- Cross-platform (Android, iOS, Flutter, web)
- Material components available as library
- https://www.material.io/

Components

AndroidManifest.xml

- Essential information about app for OS
- Package name (application unique id)
- Describe components
- Permissions
- Min required API level
- Supported/required screens, features
- Target SDK
  - Compatibility modes
- Used for filtering in app store

Activity

- Screen with UI
- Usually one activity per application screen
- Activity stack
- Lifecycle
- Contains fragments and/or views

Service

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

Content provider

- Manage and share application data
- Doesn’t specify storage implementation (db, file, web)
- Query or modify data
- Optional permissions
  - Custom permissions who can access data
- Used by system for
  - SMS
  - Contacts
  - Call log

Broadcast receiver

- Listends for actions invoked by system or other application
- Static or dynamic registration
- System-wide
- Limited since API-26
- Examples
  - Incoming SMS
  - Low battery, battery percentage changed
  - Connectivity change
  - Headphones connected/disconnected

Intent

- Asynchronous message between component
- Start activities
- Start or bind services
- Send broadcast

Project structure

Project structure

Project

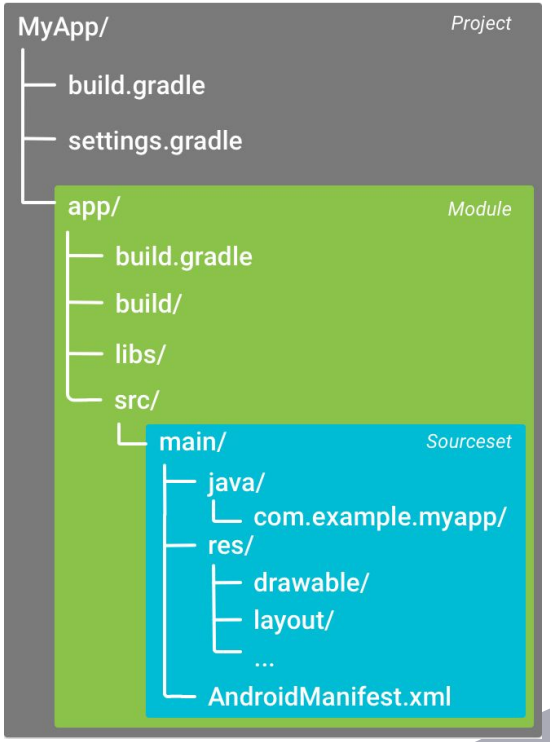
- Common configuration for modules
  - Common dependencies versions

Module

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

Sourceset

- Source code and resources
- Source code from main source set available everywhere
- Resources can be overridden in different source set



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 plugin

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

Project files - local.properties

- Contains paths to sdk and ndk
- Can’t be shared between developers
- Generated during build, 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/

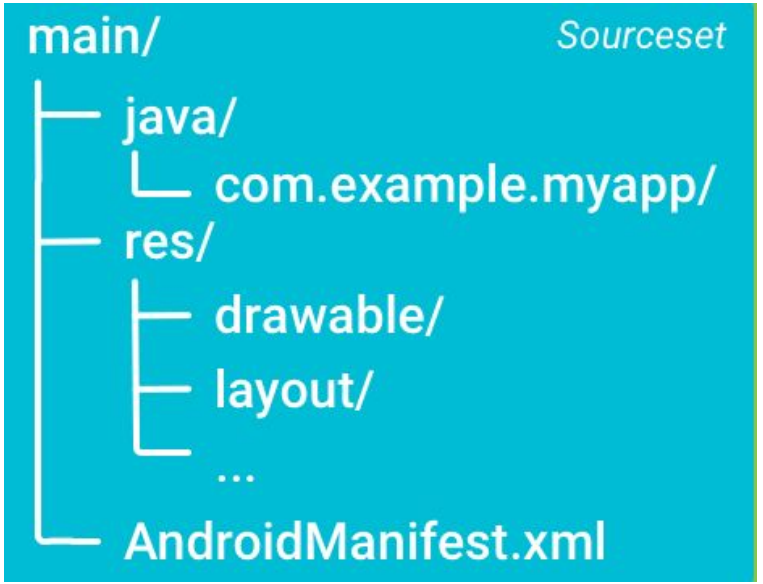
- *.jar libraries
- If it is possible use library as gradle dependency

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

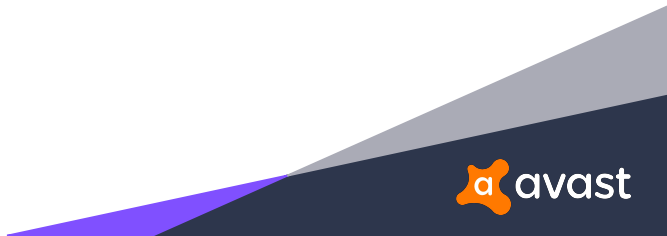
- Resources in different variants
- Drawable, drawable-mdpi…
- Values, values-cs, values-de
- Layout, layout-sw600dp

Resources - drawables

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

Resources - layout

- Definition of UI



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 VeiwGroup
- Defined in XML or programmatically
- Folder res/layout
- Options
  - FrameLayout
  - LinearLayout
  - RelativeLayout
  - TableLayout
  - GridLayout
  - ConstraintLayout
    - Google IO 2016
    - Available in support library

Layout - FrameLayout

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

Layout - 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 - Constraint layout

- Available as dependency
- “Extended relative layout”
- Constraint is connection or alignment to another view/parent/guideline

Widgets

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

Widgets

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


avast
Hello World

Activity & Back stack

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 data for new activity
- Flags - manipulation with activity stack

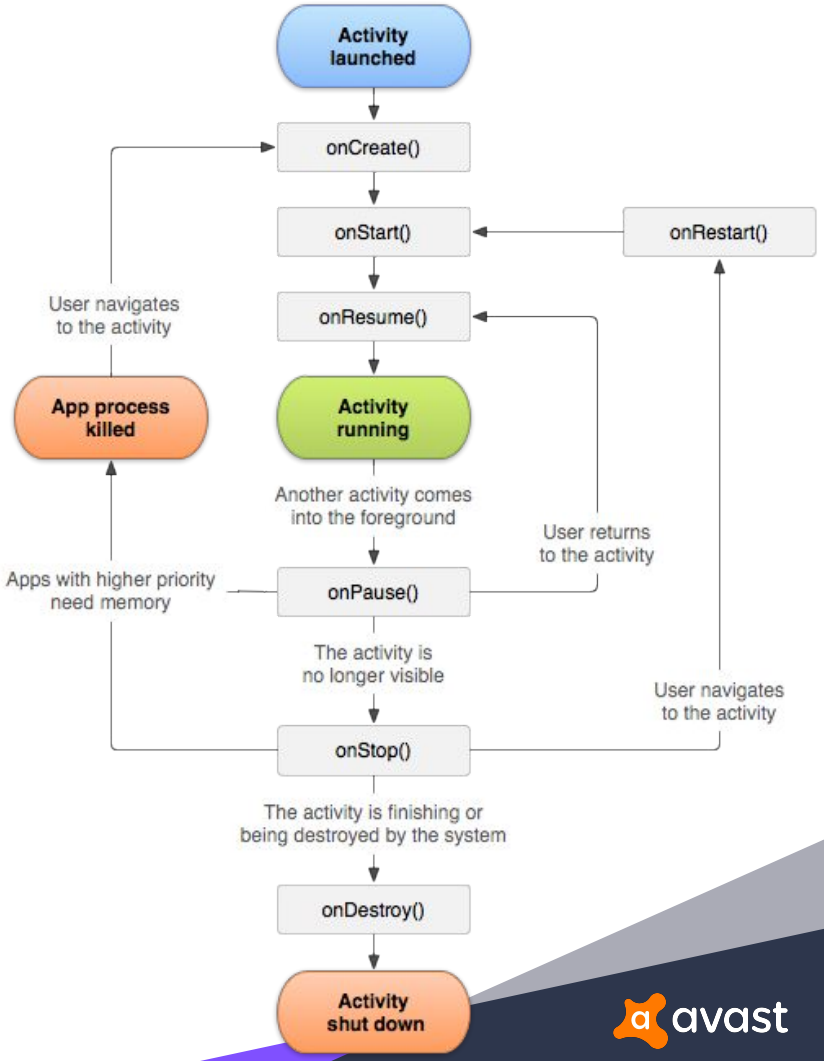
```
val intent = Intent(activity, BottomBarActivity::class.java)
intent.putExtra("key", "value")
intent.putExtra("keyInt", 5)
startActivity(intent)
```

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

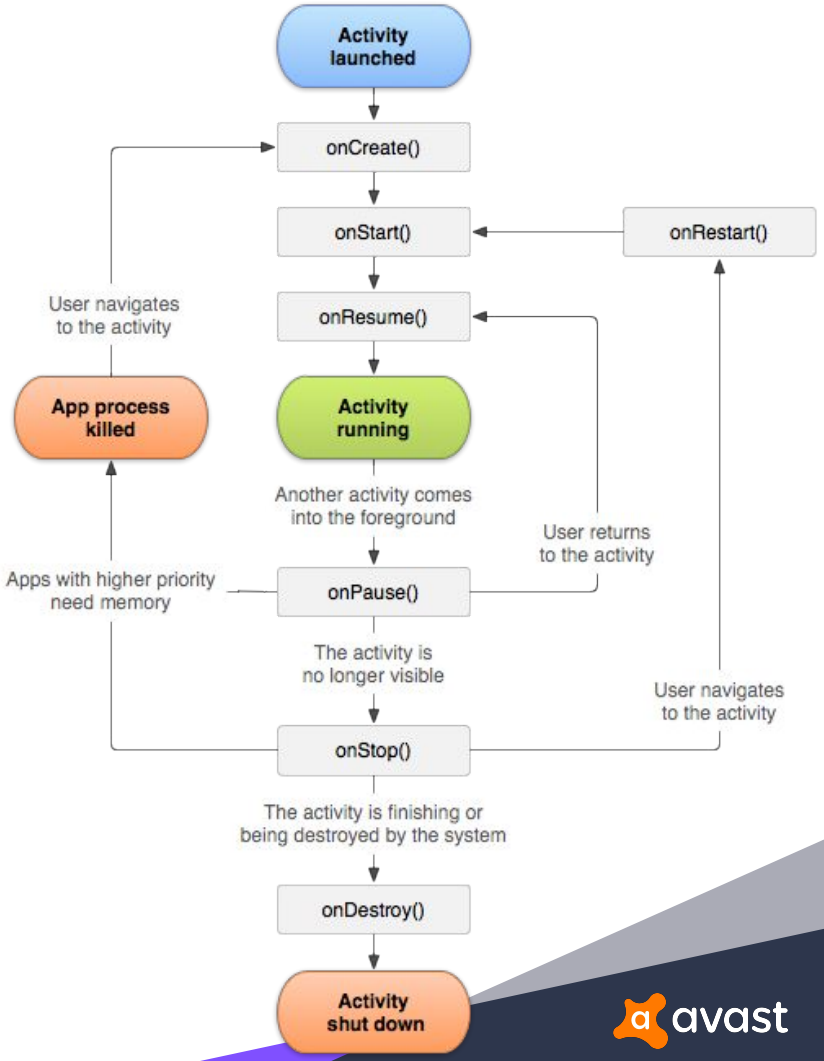
Activity - states

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



Activity#onCreate(Bundle)

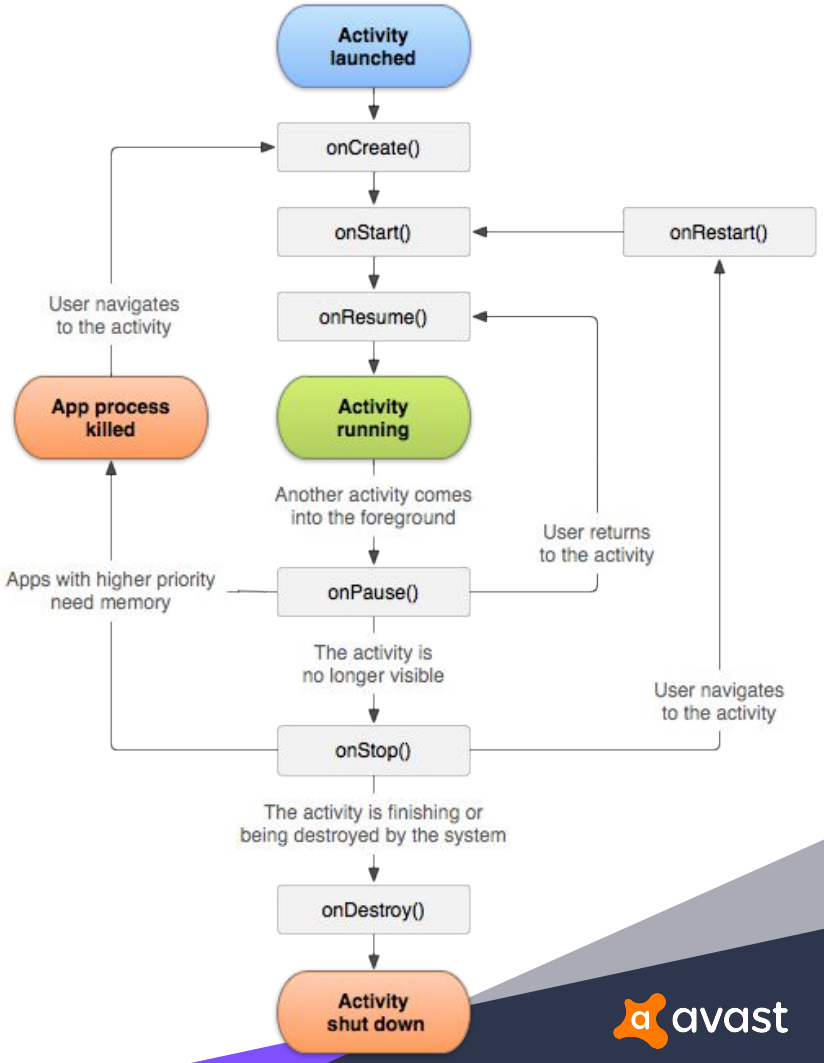
- Activity is being created
- Create views
- Bind data to list
- Passed Bundle object contains activity previous state
- Read data from starting intent
- Always followed by #onStart()



avast

Activity#onStart()

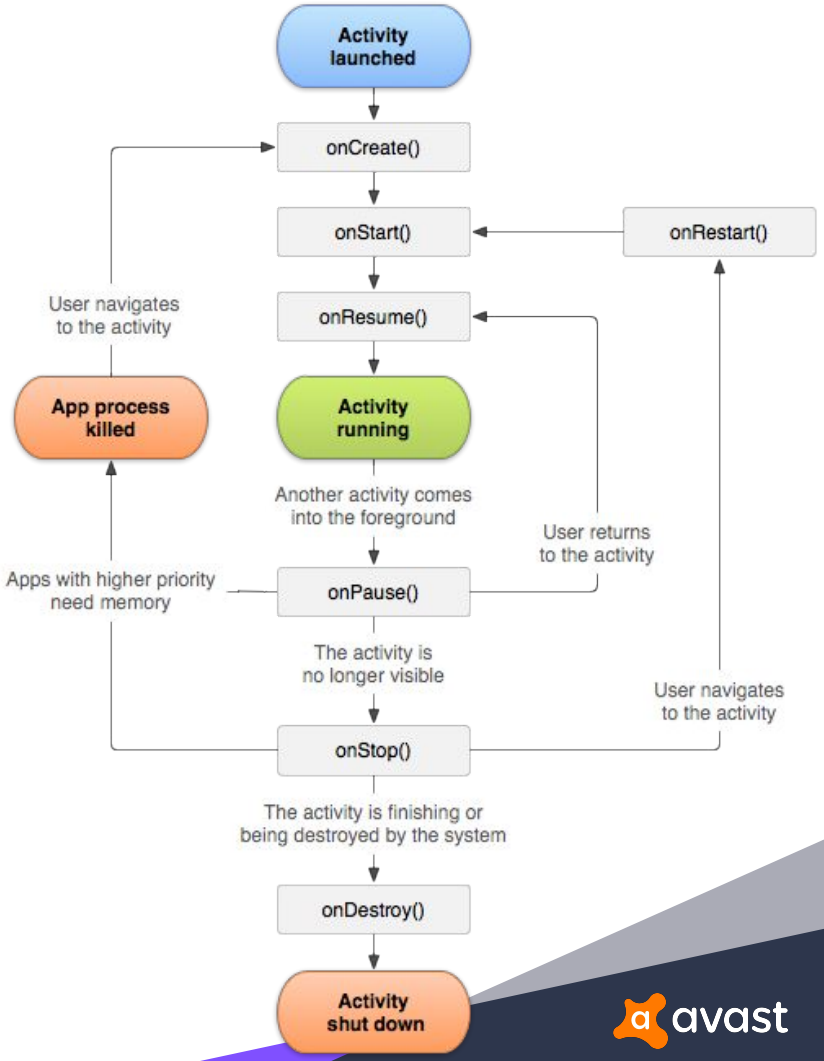
- Called before the activity become visible to the user
- Followed by
  - onResume() if come to the foreground
  - onStop() if becomes hidden
- Activity is partially visible, register listeners for changing UI



avast

Activity#onResume()

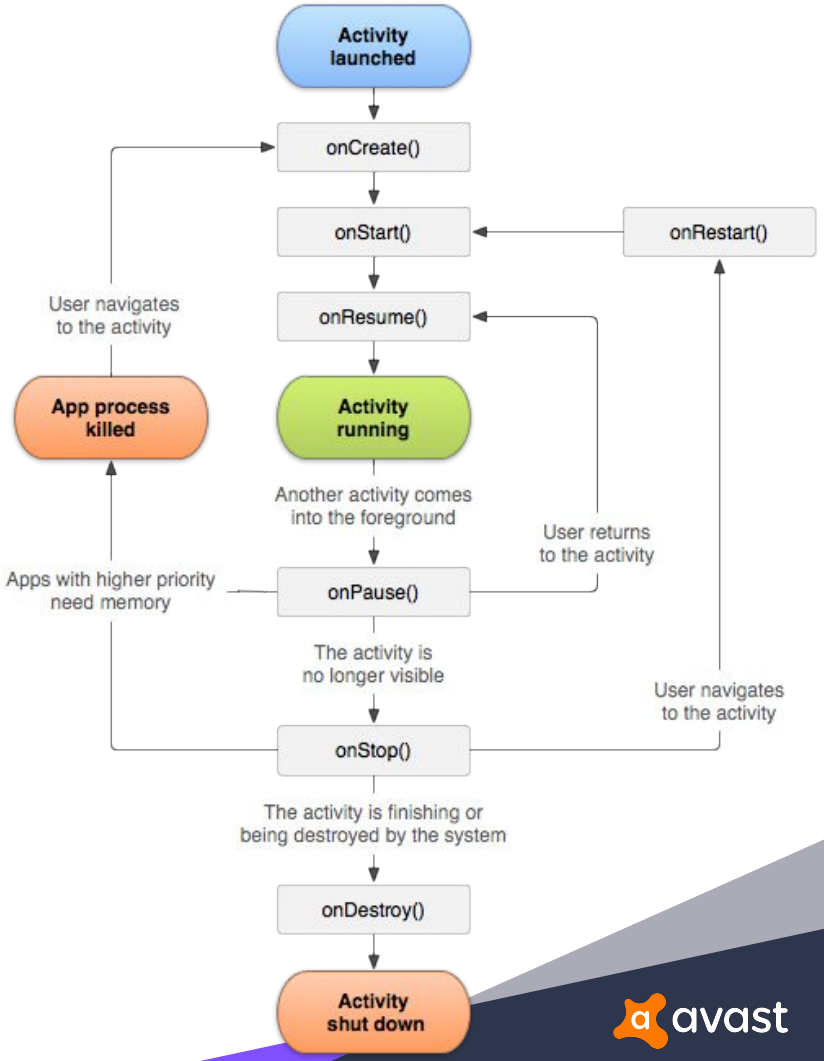
- Called just before activity start interacts with user
- Activity is on top of activity stack
- Run stuff for user
- Always followed by onPause()



avast

Activity#onPause()

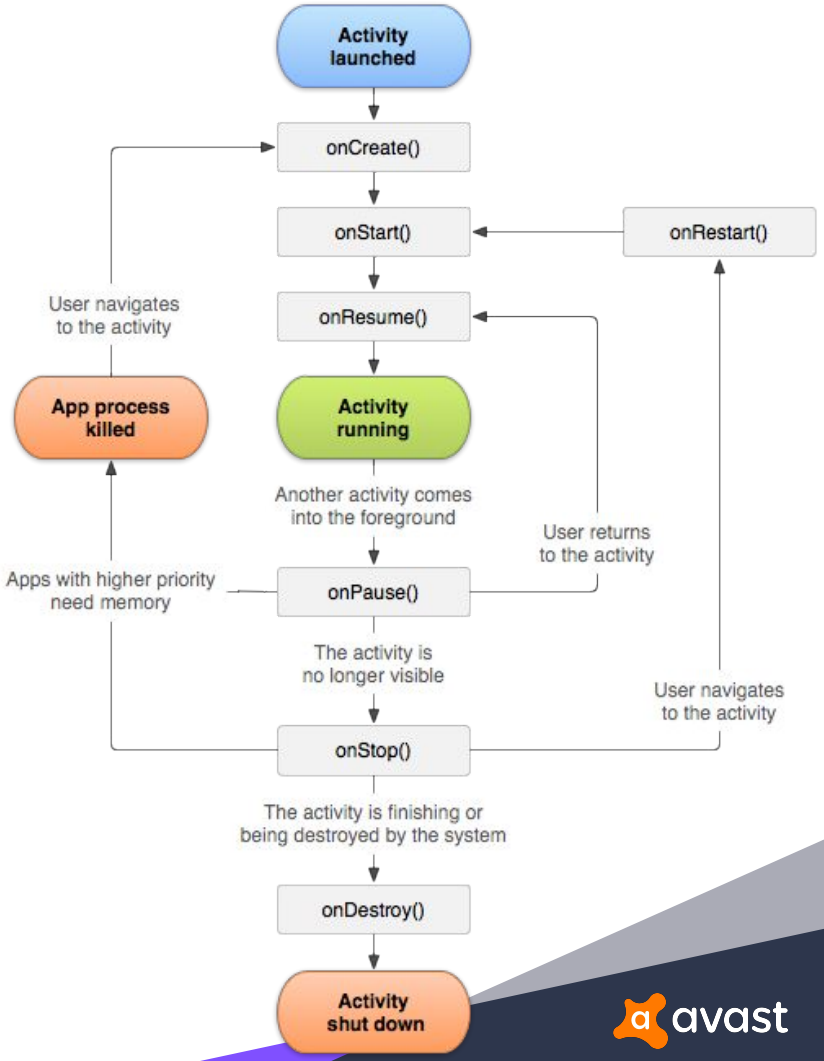
- System is about to resume another activity
- Commit unsaved changes, persist data, stop animations and CPU intensive stuff
- Should be very fast, because another activity onResume() wait until this finish
- Followed by
  - onResume() if the activity returns back to the front
  - onStop() if became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



avast

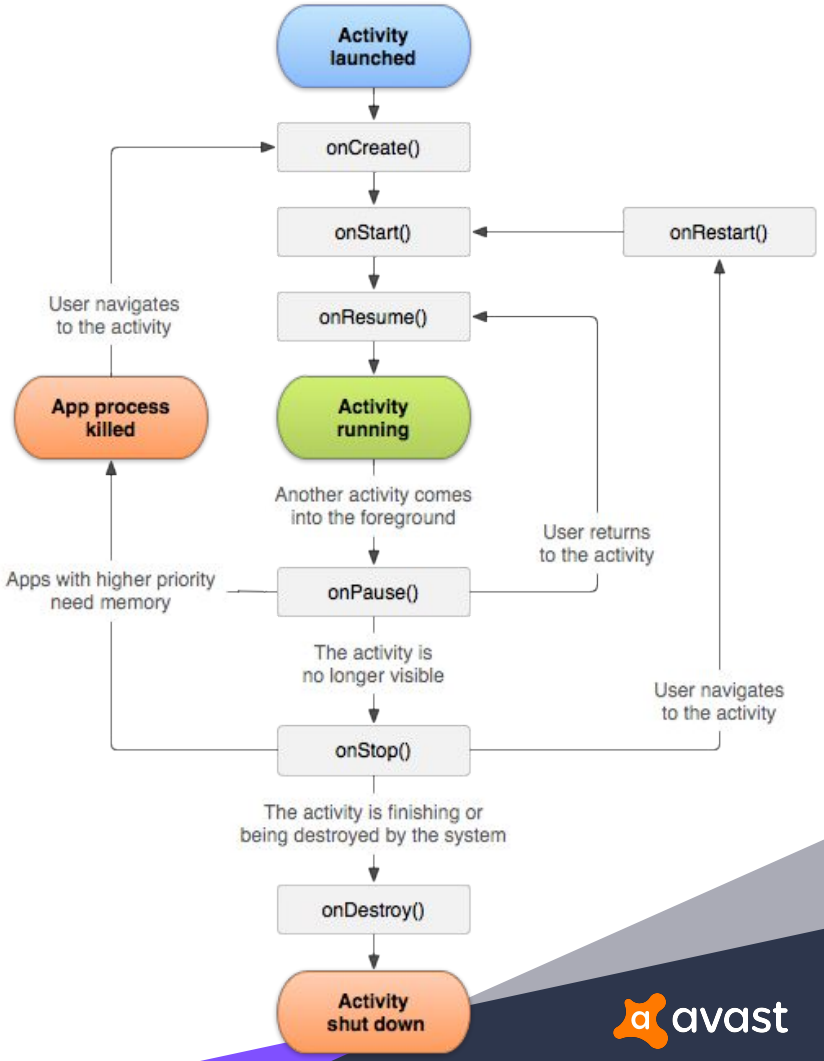
Activity#onStop()

- 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



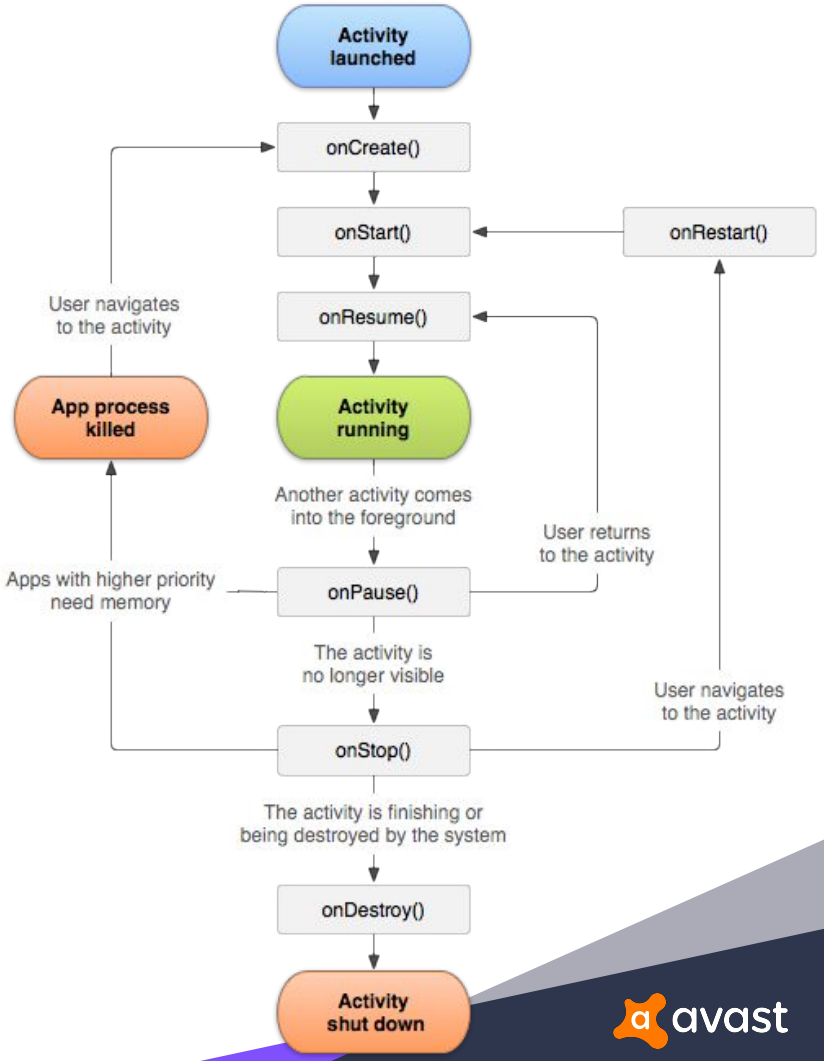
Activity#onDestroy()

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



Activity#onRestart()

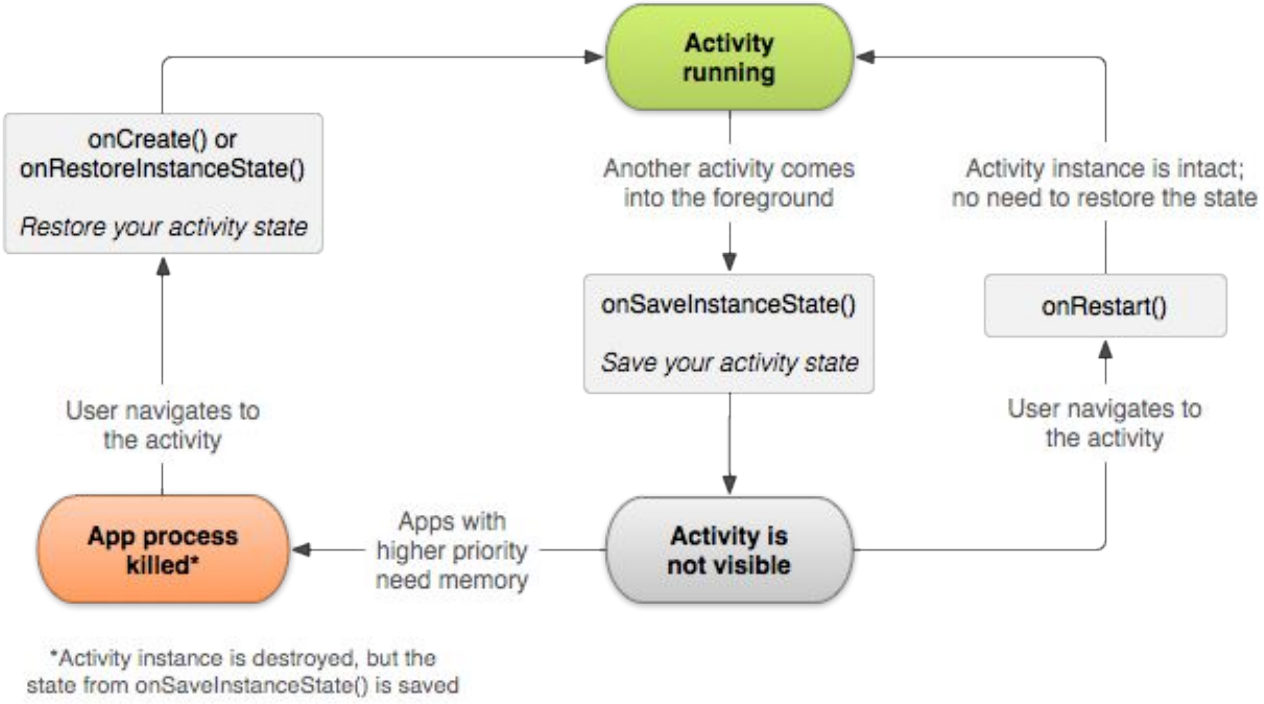
- Called after activity has been stopped, and before is started again



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

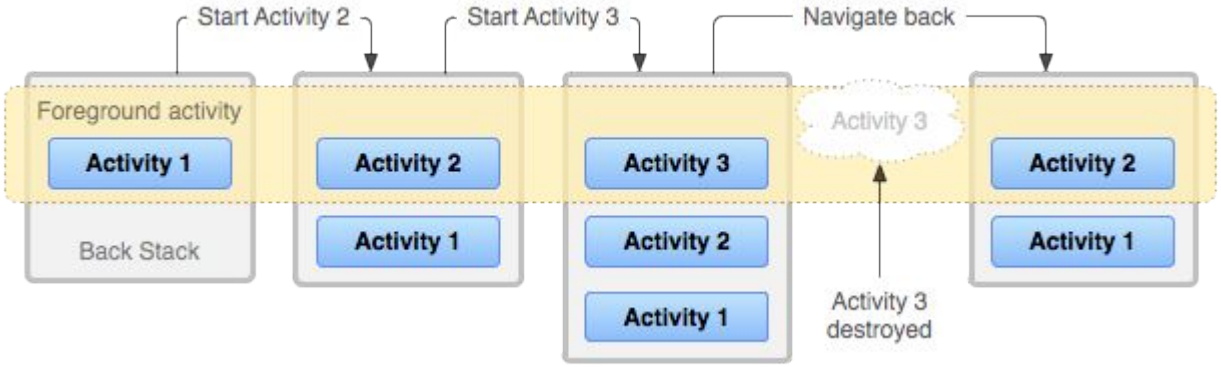
- System can kill background activity to free up resources => state of the activity is lost
- Implement #onSaveInstanceState
  - Called before activity is vulnerable to destruction
  - Passed Bundle is for remembering it’s state
  - Bundle with the stored state is passed into #onCreate and #onRestoreInstanceState (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
- Other java primitives

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 its 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 app)

Toast

- Simple non modal information
- Displayed for short period of time
- Doesn’t have user focus
- `android.widget.Toast`

```
Toast.makeText(context, "Toast example", Toast.LENGTH_LONG).show()
```

Log messages

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

- `Verbose`
- `Debug`
- `Info`
- `Warning`
- `Error`
- `What a terrible failure`

Context

Context

- Abstract class implemented by components
- `android.content.Context`

- Resources access
- Register/unregister BroadcastReceivers
- Run Activity, Services
- Binds Services

Context - types

- Application
  - Single instance
  - Extends Context
- Activity/Service
  - Multiple instances
  - Extends Context
  - Can be easily leaked
- BroadcastReceiver
  - Receive instance of Context in `BroadcastReceiver#onReceive()`
  - `registerReceiver()` and `bindService()` doesn't work
- ContentProvider
  - Not instance of Context
  - `getContext()` returns Context of application which called the receiver

Thank you
Q&A

Feedback is appreciated

prokop@avast.com

Please use [mff-android] in subject