# Android Development from scratch

David Vávra, Android GDE

## Agenda

- Android platform: history, today’s ecosystem
- Problems & opportunities & success stories
- Development options
- Hello world
- Android building blocks, UI, resources, …
- Threads, logging, toasts, preferences
- Exercise

## Android platform

- Linux-based operating system
- open-source (http://source.android.com/)

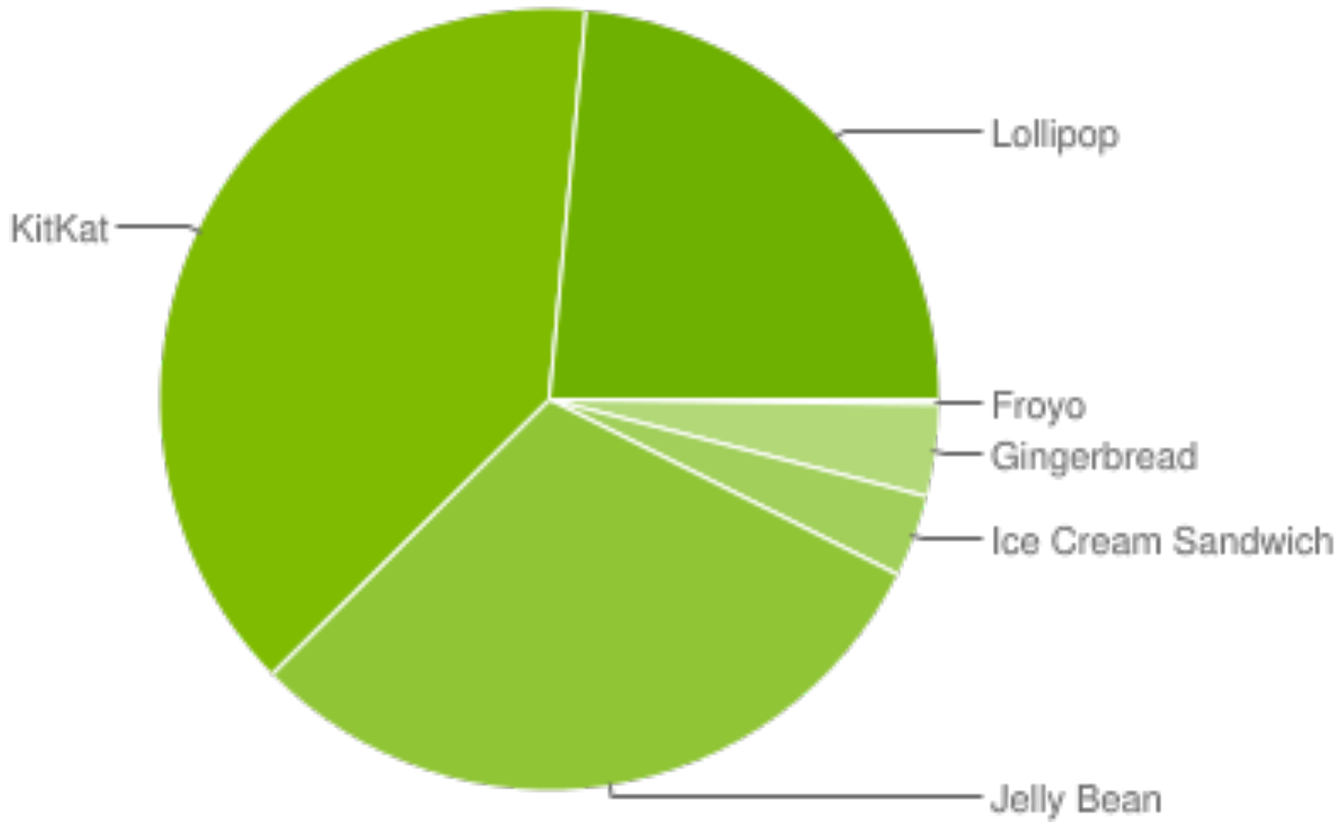
# History

- 2003, Android inc., digital cameras
- 2005, acquired by Google, 2007 iPhone
- Sep 2008, the first Android phone
  - T-Mobile G1
- May 2010, Froyo (Android 2.2)
- Feb 2011, Honeycomb (Android 3.0)

# History

- Oct 2011, Ice Cream Sandwich (4.0)
- July 2012, Jelly Bean (Android 4.1)
- July 2013, Jelly Bean (Android 4.3)
- Oct 2013, KitKat (Android 4.4)
- June 2014, Lollipop (Android 5.0)
- September 2014, Marshmallow (6.0)

## Android today



## Android today

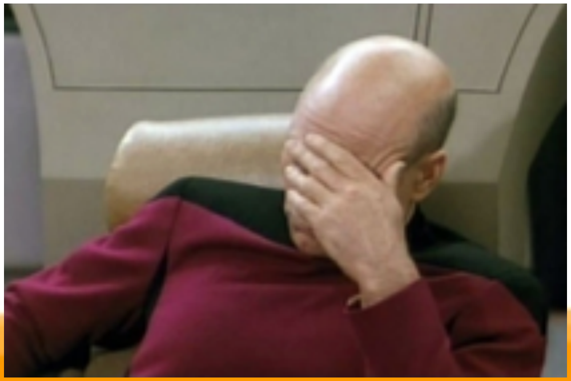
- global marketshare 78.4%
- 1.5 million devices daily activated
- tablet marketshare 36.5%
- >1.7 million apps in Play Store
- $1.8 billion from app sales in 2014

## Android today

- Phones
- Tablets
- Android Wear
- Android TV
- Android Auto
- Project Tango
- Brillo
- (Google Glass)

## Android “problems”

- fragmentation, slow upgrades, manufacturer changes
- Android users less likely to pay
- low-end devices
- lower quality apps in Google Play, malware
- no Play Store in China
- API is getting restricted



# But!

- Tons of users!
- Almost instant publishing
- No yearly fees, no need for Mac :)
- Open-source, built to handle various factors
- Developer freedom
- Support library & Google Play services
- Support from Google
- Nexus & Motorola devices

## Success Stories

- Urbandroid
- Tomáš Hubálek
- TappyTaps
- Inmite
- Ackee
- STRV
- and many more

## Development options

- App-like mobile web
- Other language frameworks (Xamarin, Scala, Kotlin…)
- C-based frameworks (Unity)
- WebView-based frameworks (PhoneGap)
- Native

## Native Development

- programming in “Java”
  - Java SE 7 (KitKat)
- native apps possible (C++)
- Android Studio (IntelliJ Idea)
  - Windows, Linux, Mac OS X

- https://www.youtube.com/watch?v=Z98hXV9GmzY

Hello world

# Android building blocks

- Gradle
- AndroidManifest.xml
- resources
- Activity
- Service
- Content provider
- Broadcast receiver
- Intents

## Gradle

- Build system based on Groovy
- Ties everything together
- Produces APKs
- Allows multiple app flavors
- Manages dependencies (Maven repos)

## AndroidManifest.xml

- defines what parts the app have
- defines which endpoints are exposed
- minimum/maximum API level
- permissions
- declare hardware and software features
- require configuration

## Activity

- screen with user interface
- the only visual component
- contains Fragments which contains Views
- examples: home screen, list of emails, create new email, …

## Service

- has no UI, but can fire notification
- long-running tasks
- examples
  - music playback service
  - download service
  - sync service

## Content Provider

- managers and shares application data
- data storage doesn’t matter (db, web, filesystem)
- apps can query and modify data through content provider
- r/w permissions can be defined
- examples - all system dbs (SMS, contacts, ...)

## Broadcast Receiver

- responds to broadcasts
- broadcasts are system wide
- can be registered statically or dynamically
- system or custom messages
- examples - incoming SMS, incoming call, screen turned off, low baterry, removed SD card, BT device available, ...

# Intent

- asynchronous message
- binds components together
- starting activities
- starting services and binding to services
- sending broadcasts

## Activity

- a subclass of android.app.Activity
- app usually has many activities
- activities managed in activity stack
  - newly started activity is placed on the top of the stack

## Activity Lifecycle

- activity can be in different states during its lifecycle
  - foreground, paused, stopped, killed
- when activity state changes a system callback is called

## Activity callbacks

- onCreate() - activity created
- onStart() - activity visible for the user
- onResume() - activity gains user focus

## Activity callbacks

- onPause() - system resuming another activity
- onStop() - activity becoming invisible to the user
- onDestroy() - before activity is destroyed

## Configuration changes

- when configuration changes, activities are destroyed and recreated
  - default behaviour, can be changed
- properly handle config changes
  - `onSaveInstanceState(Bundle)`

## Intent & Activity

- starting activity explicitly
  - `new Intent(context, MyActivity.class)`
- starting activity implicitly
  - `new Intent(Intent.ACTION_VIEW, Uri.parse(“http://developer.android.com”))`
- starting activity for result
  - `startActivityForResult(intent, REQUEST_CODE)`
  - `onActivityResult(int requestCode, int resultCode, Intent data)`

## User Interface

- defined by a hierarchy of views
- layouts = containers
  - LinearLayout, RelativeLayout, FrameLayout, ...

## User Interface

- widgets
  - UI objects
  - Button, TextView, EditText, RadioButton, ...
  - WebView

# User Interface

- list widgets
  - display a list of items
  - use adapter to bind list to data
  - RecyclerView, ListView, GridView, Spinner, ...

## Resources

- drawables
  - bitmaps
  - 9-patch png
  - state lists
  - …

## Resources

- layout
- strings
- menus
- …

## Resources

- resources can be created in several versions
  - the best version is selected according to current device configuration in runtime

## Resource qualifiers

- suffixes for resource folders
  - drawable, drawable-mdpi, ...
  - values, values-cs
  - layout, layout-sw640dp
  - drawable-hdpi-v11

## Resources

- resource units
  - dp - density-independent pixel
  - sp - scale-independent pixel (for fonts)
  - never use px

## Binding between xml and Java

- accessed from code via generated R.java file and resource ids
  - `view.findViewById(R.id.txt_name)`
  - `txtName.setText(R.string.txt_name_label)`

## Threads

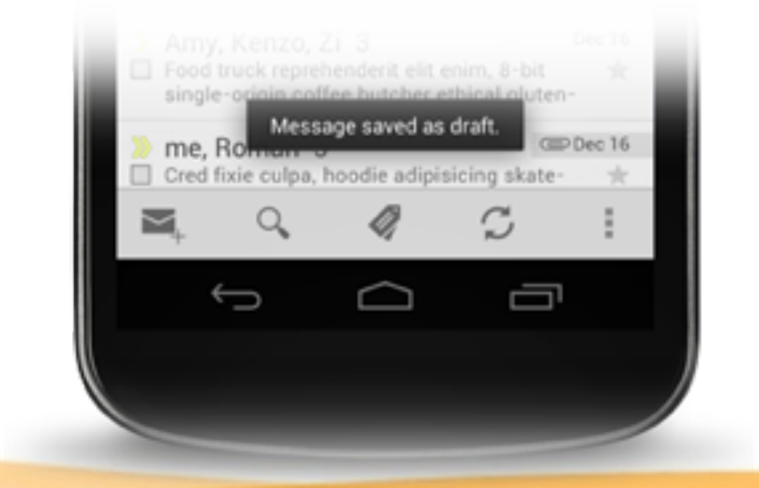
- main thread = UI thread
- do not ever block the UI thread!!!
- use worker threads for time consuming operations (network, disk, CPU)
- Tools: Services, AsyncTask, Loaders, RxJava

## Logging

- `android.util.Log`
- `Log.d(TAG, “Debug log”);`
- `Log.e(TAG, “Error log”);`

# Toast

- simple non-modal information
- displayed for a short period of time
- doesn't have user focus


Amy, Kenzo, Zi 3
Dec 16
Food truck reprehenderit elit enim, 8-bit
single-origin coffee butcher ethical gluten-
Message saved as draft.
me, Roman 3
Dec 16
Cred fixie culpa, hoodie adipisicing skate-

# Preferences

```
SharedPreferences prefs = PreferenceManager
    .getDefaultSharedPreferences(context);
SharedPreferences prefs =
    config.getSharedPreferences(PREFS_FILE_NAME,
    Activity.MODE_PRIVATE);

int storedValue = prefs.getInt(SOME_KEY, defaultValue);

SharedPreferences.Editor editor = prefs.edit();
editor.putInt(SOME_KEY, storedValue);
editor.apply();
```

## Sources

- developer.android.com
- android-developers.blogspot.com
- source.android.com
- stackoverflow.com
- youtube.com/androiddevelopers
- G+

## Exercise

1. Download the code: http://github.com/avast/android-lectures
2. Import project to Android studio
3. Run the app on device/emulator
4. When ‘choose user’ is clicked, open UserListActivity
5. When user selects item, go back and return result.
6. Show selected item in EditText

# Exercise

7. Show EditText content in a Toast
8. Download data about user with method GitHubApi.downloadUser() and parseUser() when ‘Show user detail’ is clicked
9. Do the same using AsyncTask when ‘User detail - better’ is clicked
10. Show name, repo URL and number of repos in TextViews on MainActivity

# Exercise

11. Add button ‘Open web’ which opens website of the repo in the browser
12. Make sure rotation works - selected user shouldn’t disappear after rotation.
13. Make sure the selected user stays after you close the app - save to SharedPreferences
14. Add Czech localization
15. Experiment!

THE END

+David Vávra
@destil