Android Workshop

DevNexus
Atlanta, GA
March 2015

Contact Info

Ken Kousen

Kousen IT, Inc.

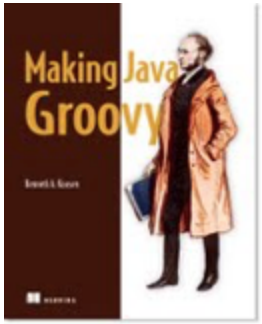
ken.kousen@kousenit.com

http://www.kousenit.com

http://kousenit.wordpress.com (blog)

@kenkousen

https://github.com/kousen (repo)



Home Page

Developer home page

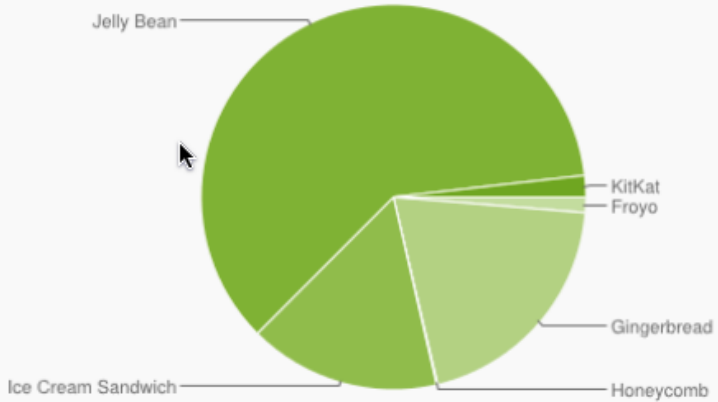
http://developer.android.com

Platform Versions

This section provides data about the relative number of devices running a given version of the Android platform.

For information about how to target your application to devices based on platform version, read Supporting Different Platform Versions.

Version	Codename	API	Distribution
2.2	Froyo	8	1.3%
2.3.3 - 2.3.7	Gingerbread	10	20.0%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	16.1%
4.1.x	Jelly Bean	16	35.5%
4.2.x		17	16.3%
4.3		18	8.9%
4.4	KitKat	19	1.8%



Data collected during a 7-day period ending on February 4, 2014.
Any versions with less than 0.1% distribution are not shown.

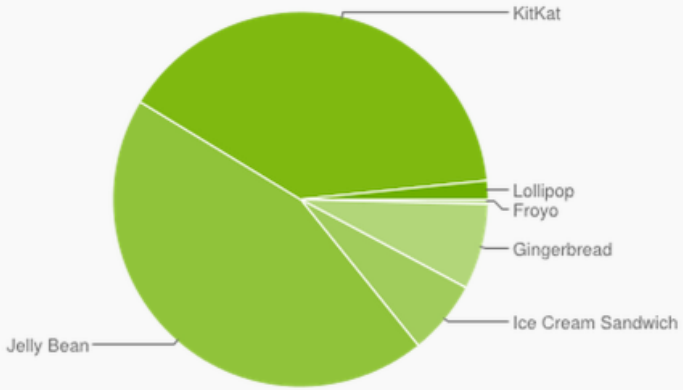
Android dashboards, https://developer.android.com/about/dashboards/

Platform Versions

This section provides data about the relative number of devices running a given version of the Android platform.

For information about how to target your application to devices based on platform version, read Supporting Different Platform Versions.

Version	Codename	API	Distribution
2.2	Froyo	8	0.4%
2.3.3 - 2.3.7	Gingerbread	10	7.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	6.4%
4.1.x	Jelly Bean	16	18.4%
4.2.x		17	19.8%
4.3		18	6.3%
4.4	KitKat	19	39.7%
5.0	Lollipop	21	1.6%



Data collected during a 7-day period ending on February 2, 2015.
Any versions with less than 0.1% distribution are not shown.

Android dashboards, https://developer.android.com/about/dashboards/

SDK Bundle

https://developer.android.com/sdk/index.html

Android Studio IDE

Android SDK tools

Android 5.0 (Lollipop) Platform

Android 5.0 Emulator

Android Studio

The only supported IDE

- Based on IntelliJ IDEA
- Uses Gradle for builds

Versions

Platform version:

2.3.3, 4.4, 5.0

Codename:

Gingerbread, ICS, JellyBean, KitKat, Lollipop

API numbers:

13, 14, 15, …, 19, (skip 20), 21

Compatibility library

API changed significantly as of 3.0+

- `ActionBar`
- `Fragment`

Compatibility library available

Training

https://developer.android.com/training/index.html

Brief tutorials

Getting Started

Thin, but useful

Reference

https://developer.android.com/reference/packages.html

- Javadocs
  - Good search capabilities
  - Use magnifying glass

Creating an application

Must select unique package name

```
com.mycompany.myapp
```

Used in Google Play store

(Not exposed to clients)

Creating an application

Choose min SDK level

- Min level willing to support

Choose target SDK level

- As late as possible

Manifest

AndroidManifest.xml

```
<uses-sdk … />  // Moved to Gradle build
<application>
   <activity>... </activity>
   …
</application>
```

Manifest

All activities

Permissions

Intents and Intent filters

Services

Content providers

…

Activities

Each screen is an *activity*

Extends `android.app.Activity`

Full of callback methods

Activities

Each activity has an XML *layout*

`activity_main.xml`

`activity_welcome.xml`

XML tags with many attributes

Activities

Activity callback methods:

```
onCreate, onDestroy
onStart,  onStop
onPause,  onResume
```

… many others …

HAXM

Intel Hardware Acceleration Execution Manager

https://software.intel.com/en-us/android/articles/intel-hardware-accelerated-execution-manager

Installer + SDK Manager entry

Activities

Activity diagram (no pun intended):

https://developer.android.com/guide/components/activities.html

Moves from state to state

invoking callback methods

res

Resources folder contains subfolders

- drawable
- layout
- menu
- values
- ...

Providing resources

https://developer.android.com/guide/topics/resources/providing-resources.html

- Specially named subdirectories
  - values
- Configuration *qualifiers*
  - values-v11
  - values-sw720dp-land

dp and sp

dp: density-independent pixels

Used for images

sp: scale-independent pixels

Used for strings

values

Keys and values → layer of indirection

strings.xml:

```
<string name="hello_world">Hello world!</string>
```

Declaring ids

If you need to access a resource from Java need to provide an id

```
android:id="@+id/name"
```

+ defines, otherwise references

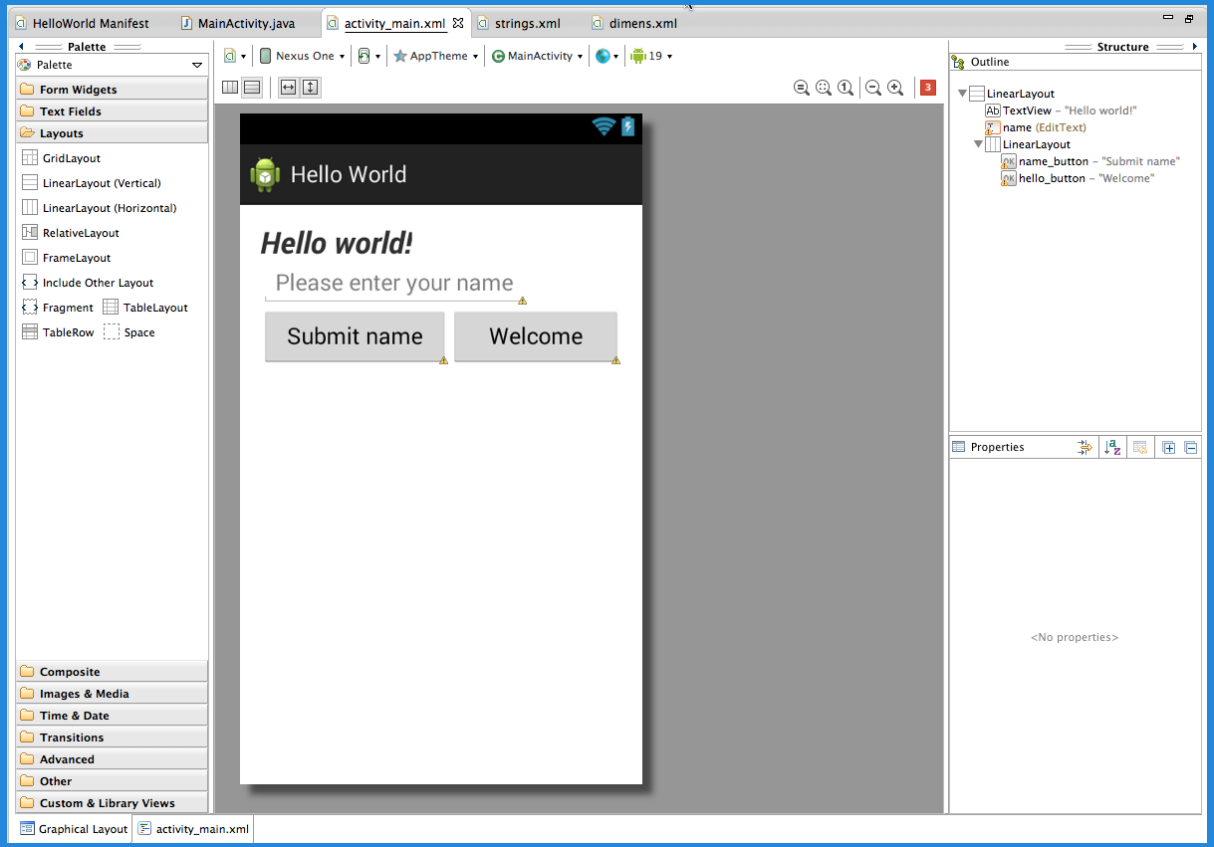
Accessing resources

XML → compiled into

R.java: full of public inner classes

generated file → do not modify

```
(Button) findViewById(R.id.hello_button)
```



Graphical editor

Layouts

`<LinearLayout>`

`<RelativeLayout>`

… others, less common …

Layouts

Add components to layouts

Must specify:

- `layout_width`
- `layout_height`

Layout

Add layout to activity

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
}
```

Widgets

Widgets generate events

Buttons → `View.OnClickListener(...)`

(Yes, anonymous inner classes)

Buttons

Adding a button listener

```
helloButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        sayHello(v);
    }
});
```

Widgets

`<TextView>` (Label in HTML)

`<EditText>` (TextField in HTML)

`<Button>`

`<CheckBox>`

`<ToggleButton>`

`<DatePicker>` See `android.widget` pkg

Widgets

`<EditText>` with text types

`text, textEmailAddress, textUri`

`number, phone`

https://developer.android.com/guide/topics/ui/controls/text.html

Toast

Brief message over UI

```
Toast.makeText(context, text, Toast.LENGTH_LONG).show();
```

Logging

`android.util.Log`

- static methods
  - `Log.d()`, `Log.v()`, `Log.i()`, `Log.w()`, `Log.e()`

Two args: TAG and message

Logging

TAG → String constant used as filter

Add filter to LogCat

Log messages in classes

Intent

Messaging object

Three use cases

- Start an activity
- Start a service
- Deliver a broadcast

Intent

Start an activity

- Pass an intent to `startActivity()`
- Or `startActivityForResult()`

Intent

Start a service

Services run in background

Pass intent to `startService()`

Intent

Deliver a broadcast

- Sends messages to receivers

- Pass intent to `sendBroadcast()`

Intent

Explicit

Specify component to start

Implicit

Declare action to perform (in manifest)

Intent

Extras → data carried to destination

(like a map of keys and values)

```
Intent intent = new Intent(this, WelcomeActivity.class);
intent.putExtra("name", name);
startActivity(intent);
```

Views and adapters

`ListView` with Adapters

`ArrayAdapter`

creates view for each item

`setAdapter` on `ListView`

ActionBar

Apps with version > 3.0

Inside <menu>:

```
<item
    android:id="@+id/action_joke"
    android:showAsAction="ifRoom|withText"
    android:icon="@drawable/ic_launcher"
    android:title="@string/get_joke"/>
```

ActionBar

```
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
    case R.id.action_joke:
        // do whatever click should do
        return true;
    default:
        return super.onOptionsItemSelected(item);
    }
}
```

AsyncTask

Perform asynchronous work off UI thread

Publish results to UI thread

AsyncTask

```
AsyncTask<Params, Progress, Results>
```

Short operations (few seconds)

AsyncTask

```
onPreExecute()
doInBackground()
onProgressUpdate()
onPostExecute()
```

https://developer.android.com/reference/android/os/AsyncTask.html

Longer running tasks

Use `java.concurrent` package

- `Executor`
- `ThreadPoolExecutor`
- `FutureTask`

Or, better yet, use services…

Services

Long-running, background operations

- network operations
- play music
- file I/O

Services

Started

- Runs to completion

Bound

- Interacts with calling client
- Only exists when bound

Services

"Runs in background"

- Service runs in application thread
- Keeps running if user switches apps

You can (and should) start new thread

- Use AsyncTask, for example

REST

- org.json package
  - JSONArray
  - JSONObject
- android.util package
  - JsonReader
  - JsonWriter

REST

`org.apache.http.client` packages

- `HttpClient`
- `HttpGet`
- `HttpPost`
- …

REST

Alternative:

Spring for Android

http://projects.spring.io/spring-android/

`RestTemplate` class

Map classes to JSON structure

Storage options

Shared preferences

Internal storage on device

External storage

SQLite databases

https://developer.android.com/guide/topics/data/data-storage.html

Storage options

Shared preferences

- key/value pairs of primitives

`getSharedPreferences()`

- multiple files by name

`getPreferences()`

Storage options

Internal storage

`openFileOutput()` → `FileOutputStream`

```
fos.write(...)
fos.close()
```

Same with input

Storage options

External storage

- SD card or internal

Can share files with other apps

Storage options

- SQLite database
  - accessible within app only

SQLite

Extend `SQLiteOpenHelper`

- Supply constructor
- Override `onCreate()`
- Create tables with `execSQL()`

SQLite

Read and write using

- `getReadableDatabase()`
- `getWriteableDatabase()`

Assorted `query()` methods

SQLite

- Can access from adb shell
  - Use `sqlite3` tool

Content Providers

Provide data to other processes

Existing providers for calendar, contacts

Fragments

Portions of a user interface

Managed by activities

Fragments

Extend `Fragment`

or one of its subclasses

Use `FragmentManager` to manipulate

in a `FragmentTransaction`

Fragments

Fragments are portions of a UI

- Owned by Activities
- Additional callback methods

adb tool

- Android Debug Bridge
  - Part of platform tools

adb tool

`devices` → list attached devices

`pull`, `push` → copy files to device

`shell` → open shell on device

Summary

Activities and XML layouts

Intents and IntentFilters

Widgets

Services

Storage and SQLite

Content providers