Android

Safari Books Online 2017

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)

Publications

O'Reilly video courses: See http://safaribooksonline.com

- Groovy Programming Fundamentals
- Practical Groovy Programming
- Mastering Groovy Programming
- Learning Android
- Practical Android
- Gradle Fundamentals
- Gradle for Android
- Spring Framework Essentials

Home Page

Developer home page

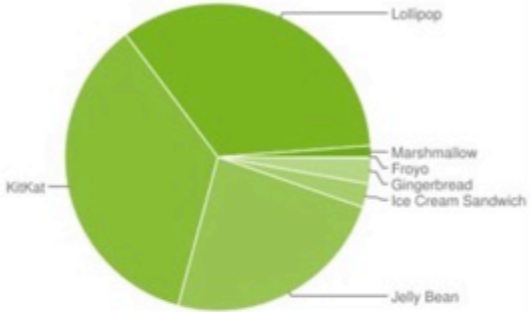
http://developer.android.com

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	0.1%
2.3.3 - 2.3.7	Gingerbread	10	2.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	2.5%
4.1.x	Jelly Bean	16	8.8%
4.2.x		17	11.7%
4.3		18	3.4%
4.4	KitKat	19	35.5%
5.0	Lollipop	21	17.0%
5.1		22	17.1%
6.0	Marshmallow	23	1.2%



Data collected during a 7-day period ending on February 1, 2016.

Any versions with less than 0.1% distribution are not shown.

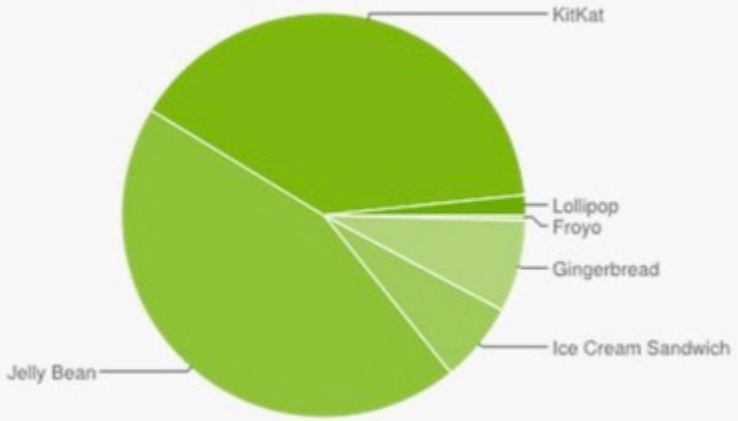
Android dashboards, https://developer.android.com/about/dashboards/

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.

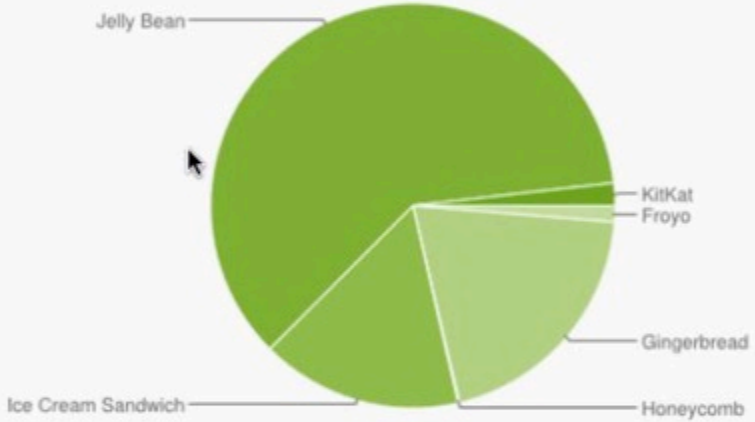
Android dashboards, https://developer.android.com/about/dashboards/

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/

Studio Bundle

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

Android Studio IDE

Android SDK tools

Latest Android Platform

Latest Android Emulator

Android Studio

The only supported IDE

- Based on IntelliJ IDEA
- Uses Gradle for builds

Versions

Platform version:

4.4, 5.0, 5.1, 6.0, 7.0, 7.1

Codename:

KitKat, Lollipop, Marshmallow, Nougat

API numbers:

16, 17, 18, 19, (skip 20), 21, 22, 23, 24, 25

Compatibility library

API changed significantly as of 3.0+

- `ActionBar`
- `Fragment`

Compatibility library available

- Allows for Material design on older devices

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 application ID

`com.mycompany.myapp`

Used in Google Play store

(Not exposed to clients)

Creating an application

Choose min SDK level

Min level willing to support

Target SDK level preselected to current

Manifest

AndroidManifest.xml

```
<uses-permission
android:name="android.permission.INTERNET" />
<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 …

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

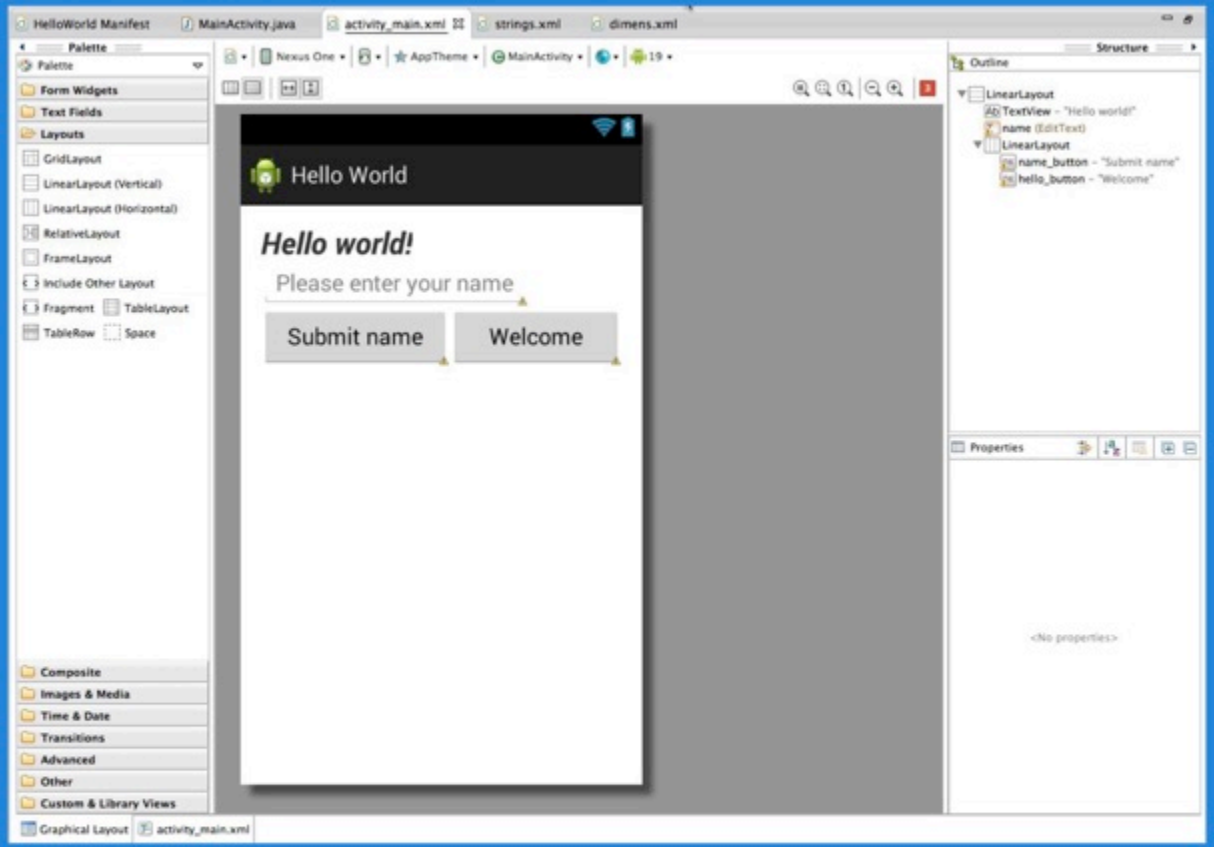
Accessing resources

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)

(ButterKnife fixes this -- more later)

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

Other Alternatives

Retrofit 2

https://square.github.io/retrofit/

Uses OkHttp for networking, then maps JSON

OkHttp

https://square.github.io/okhttp/

Networking client

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

Use a good library, like

- Gson
- Jackson

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

Gradle for Android

Basics

- Android plugin for Gradle
  - Added via `buildscript`
  - Lots of customization

Basics

```
buildscript {
    repositories { jcenter() }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.0.0'
    }
}

apply plugin: 'com.android.application'
```

Properties in build.gradle

```
android {
    versionCode ...
    versionName ...
}
```

or even in gradle.properties

Multi-project Builds

By default, apps in AS are multi-project builds

```
build.gradle
settings.gradle
```

Can add additional libraries, other modules, and more

Build Types

Two default build types:

- `debug`
- `release`

Configuring Build Types

Use buildTypes section of build.gradle

```
buildTypes {
   release { … }
   debug { … }
}
```

Can also add custom build types that way

Configuring Build Types

Each build type defines a source set

- `src/main/ ...`
- `src/debug/ ...`
- `src/release/ ...`
- `src/androidTest/...` (discussed below)

Configuring Build Types

Resources in build type source sets

replace their counterparts in main

Java classes *conflict*, however

Define class in each, or just in main

Generating a Release

Can't assemble a release until

you can sign it

Signing Your App

Use Java's **`keytool`** to generate cert

```
AndroidStudio/ICNDB_AS/app
▶ keytool -genkey -v -keystore ICNDB.keystore -alias ICNDB -keyalg RSA -keysize 2048 -validity 10000
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  Ken Kousen
What is the name of your organizational unit?
  [Unknown]:
What is the name of your organization?
  [Unknown]:  Kousen IT, Inc.
What is the name of your City or Locality?
  [Unknown]:  Marlborough
What is the name of your State or Province?
  [Unknown]:  CT
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=Ken Kousen, OU=Unknown, O="Kousen IT, Inc.", L=Marlborough, ST=CT, C=US correct?
  [no]:  yes

Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 10,000 days
        for: CN=Ken Kousen, OU=Unknown, O="Kousen IT, Inc.", L=Marlborough, ST=CT, C=US
Enter key password for <ICNDB>
        (RETURN if same as keystore password):
Re-enter new password:
[Storing ICNDB.keystore]
```

Signing Your App

```
signingConfigs {
    release {
        storeFile file('ICNDB.keystore')
        keyAlias 'ICNDB'
        storePassword 'gradle_rules'
        keyPassword 'carlos_ray_aka_chuck'
    }
}
```

Signing Your App

Passwords don't have to be in build file

- Can use system properties
- Can prompt user
- Can use gradle.properties

See docs for suggestions

Signing Your App

Add `signingConfig` to build type config

```
buildTypes {
    release {
        // …
        signingConfig signingConfigs.release
    }
}
```

Signing Your App

The `signingReport` task shows details

Signing Your App

Invoke **`assembleRelease`** task

Resulting apk in `build/outputs/apk` folder

Flavors and Variants

buildTypes

debug, release

flavors → different versions of same app

arrogant, friendly, obsequious

Flavors and Variants

Each flavor generates an apk

Variants combine buildTypes and flavors

debug+arrogant, debug+friendly, debug+obsequious

release+arrogant, release+friendly, release+obsequious

Multiple Flavors

```
flavorDimensions 'attitude', 'client'
productFlavors {
    arrogant { dimension 'attitude' }
    …
    stark { dimension 'client' }
    wayne { dimension 'client' }
}
```

Custom Tasks

```
task copyApks(type: Copy, dependsOn: assembleDebug) {
    from("$buildDir/outputs/apk") {
        exclude '**/*unsigned.apk', '**/*unaligned.apk'
    }
    into '../apks'
}
```

Custom Tasks

```
task printVariantNames {
    doLast {
        android.applicationVariants.all { variant ->
            println variant.name
        }
    }
}
```

Testing

- Unit testing
- Functional testing
  - Robotium
  - Espresso

Testing

With Gradle:

Tests run on all connected devices simultaneously

Testing

Use `androidTest` source set

`src/androidTest/java`

Testing

Use `androidTest` source set

`src/androidTest/java`

`androidTestCompile` dependencies

Testing

Use `androidTest` source set

`src/androidTest/java`

Run **`connectedCheck`** task

References

Android new build system

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

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

Developer's Guide section on Gradle

https://developer.android.com/sdk/installing/studio-build.html

Android Developers on G+

https://plus.google.com/+AndroidDevelopers/posts

References

Gradle Recipes for Android

https://www.amazon.com/Gradle-Recipes-Android-Master-System/dp/1491947020

Summary

Activities and XML layouts

Intents and IntentFilters

Widgets

Services

Storage and SQLite

Content providers