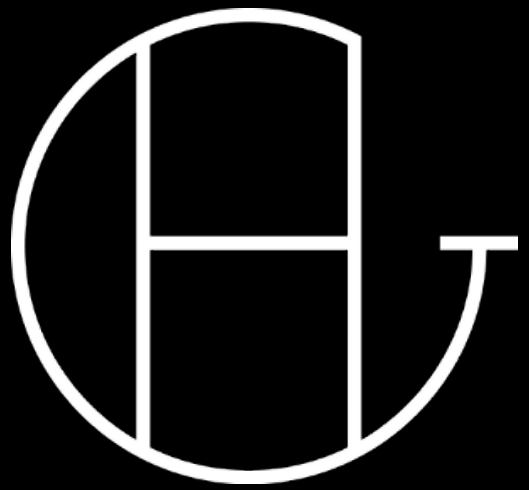


HoGent

BEDRIJF
EN
ORGANISATIE

Quick Workshop Android

Dr. Jens Buysse

Welcome to Mobile: Android

Hello Android

I've come up with a set of rules that describe our reactions to technologies:

1. Anything that is in the world when you're born is normal and ordinary and is just a natural part of the way the world works.
2. Anything that's invented between when you're fifteen and thirty-five is new and exciting and revolutionary and you can probably get a career in it.
3. Anything invented after you're thirty-five is against the natural order of things.

– Douglas Adams, author of *Hithhiker's Guide to the Galaxy*

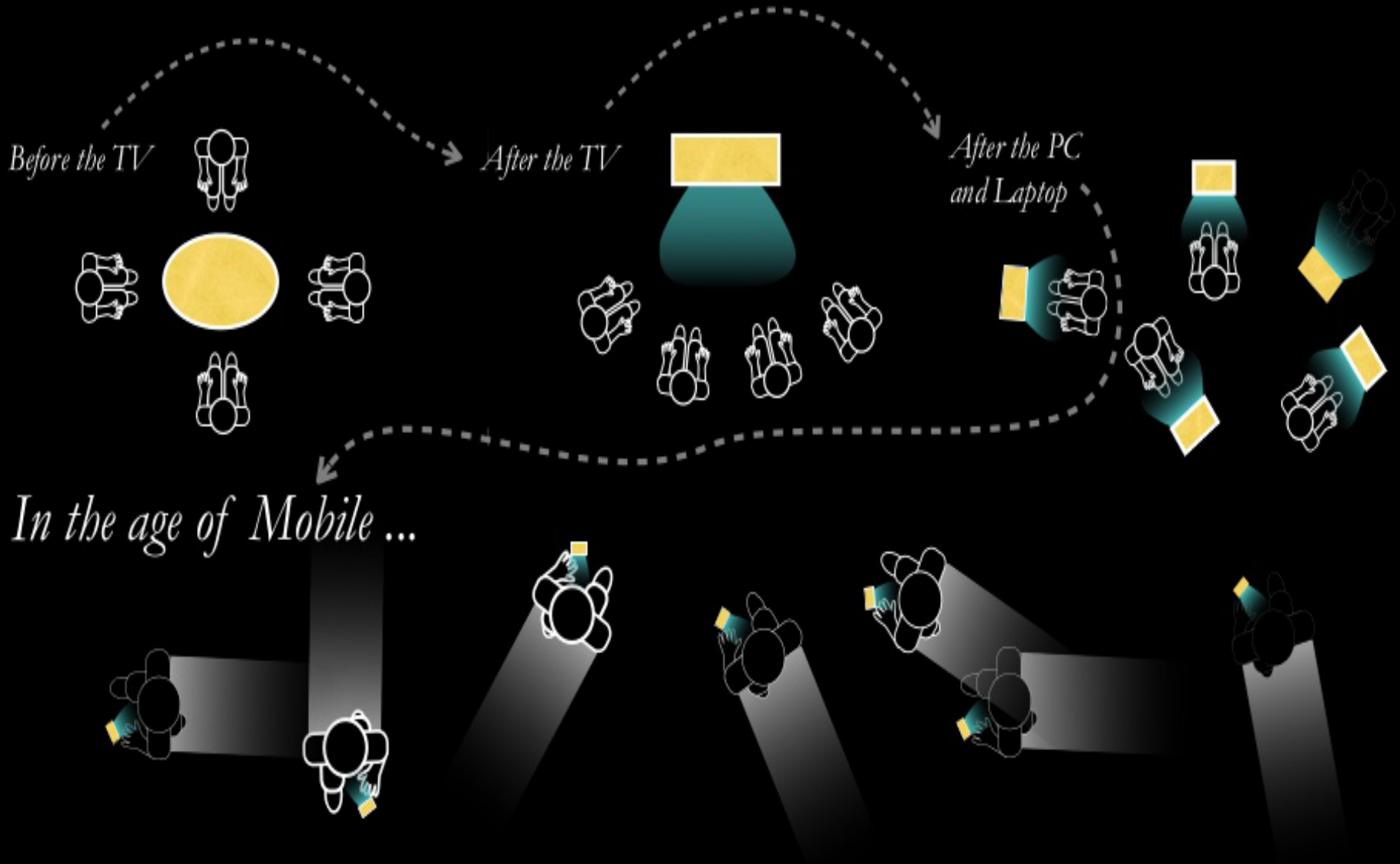
Mobile Development

Mobile Development

Mobile Development

Mobile Development

Tech is no longer for Tech-ies, because Mobile is for Everybody (Right) Now



The smartphone revolution brought design's value into the foreground. We want to do in our palm, while walking, what we used to do on a big screen while sitting down at a desk. The interaction design challenges presented by that shift are huge.

Source: @kpcb @johnmaeda @heif #DesignInTech

http://kpcb.com/design

Design in Tech KPCB

What is Android

Android is a ecosystem consisting of the following components:

1. A free, open source, operating system for embedded devices

What is Android

Android is a ecosystem consisting of the following components:

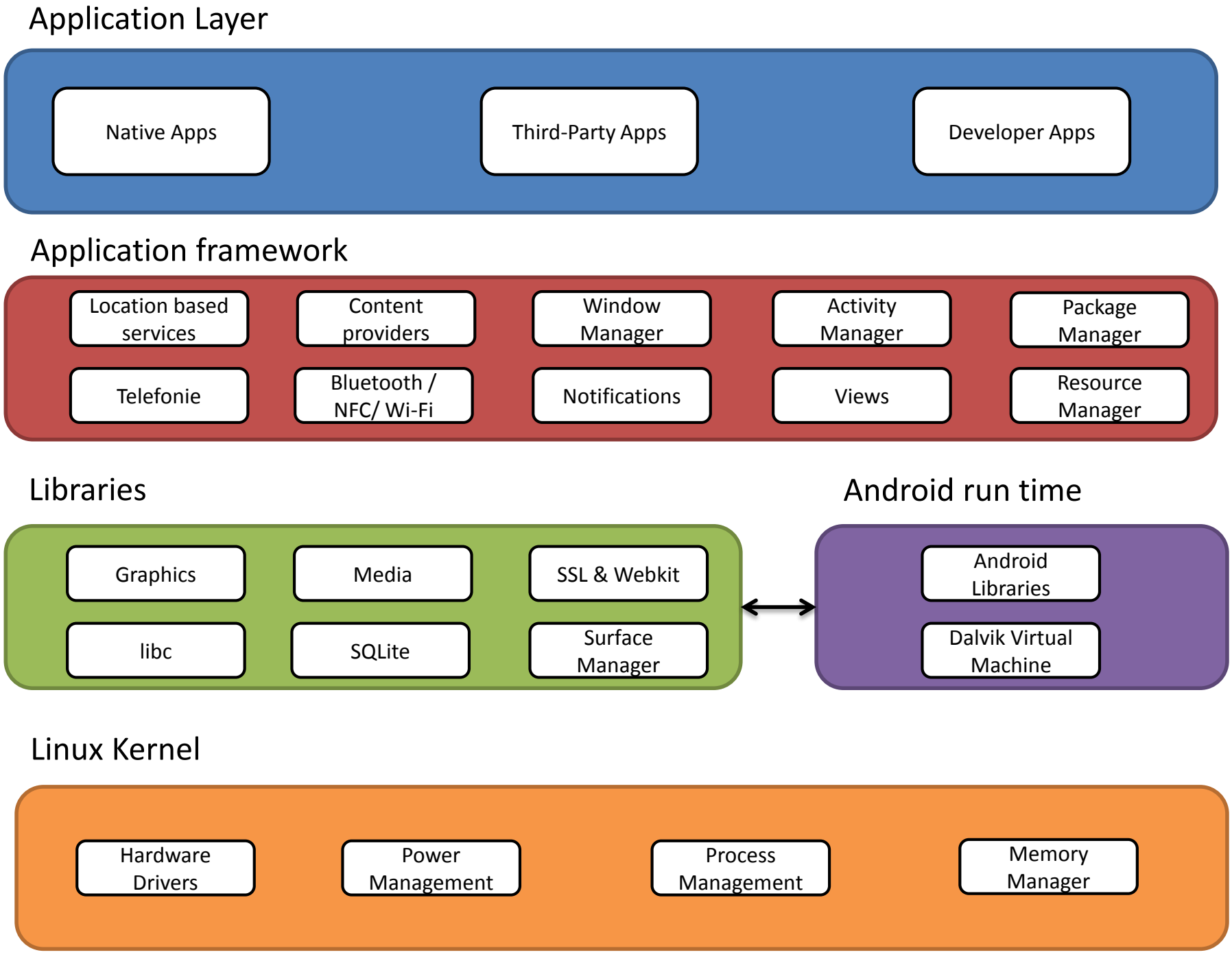
1. A free, open source, operating system for embedded devices
2. An open source development platform to create mobile applications

What is Android

Android is a ecosystem consisting of the following components:

1. A free, open source, operating system for embedded devices
2. An open source development platform to create mobile applications
3. Devices, specifically mobile devices, which run the mobile Android OS and the applications made for it

See `http://developer.android.com/training`

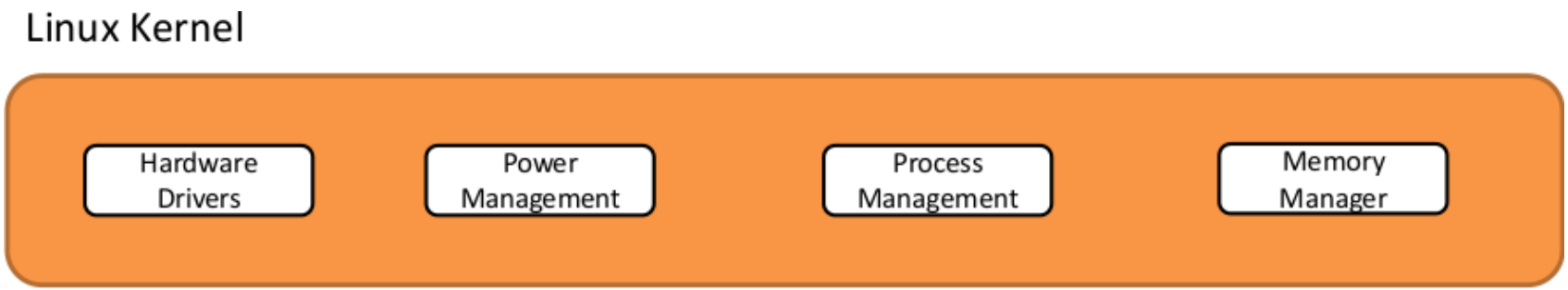

Application Layer
Native Apps
Third-Party Apps
Developer Apps
Application framework
Location based services
Content providers
Window Manager
Activity Manager
Package Manager
Telefonie
Bluetooth / NFC/ Wi-Fi
Notifications
Views
Resource Manager
Libraries
Android run time
Graphics
Media
SSL & Webkit
libc
SQLite
Surface Manager
Android Libraries
Dalvik Virtual Machine
Linux Kernel
Hardware Drivers
Power Management
Process Management
Memory Manager

Linux Kernel

Core services:

- Hardware drivers
- Process and Memory management
- Security and Power Management

De Kernel is a layer of abstraction between hardware and rest of Software Stack



Libraries

- Media Library
- Graphical libraries
- SQLLite for database support
- ...

Run time

- Core Libraries: proper implementations for the Dalvik Virtual Machine $\neq$ Java VM
- Dalvik VM: virtual machine for Android, optimized for mobile devices (battery, processor & memory management)

Dalvik Virtual Machine

- Typical Java VMs $\Rightarrow$ stack machines
- Dalvik VM $\Rightarrow$ a register-based architecture: cooler and thus uses less battery energy
- Each app uses its own process: sandboxing $\Rightarrow$ cornerstond of security in Android

Dalvik Virtual Machine

- Typical Java VMs $\Rightarrow$ stack machines
- Dalvik VM $\Rightarrow$ a register-based architecture: cooler and thus uses less battery energy
- Each app uses its own process: sandboxing $\Rightarrow$ cornerstond of security in Android

But ...

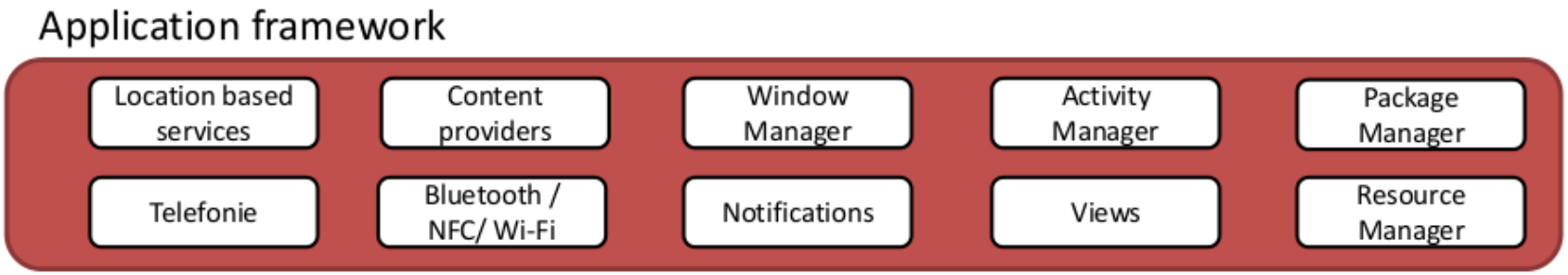
- Android 5.0 *Lollipop*: Dalvik was entirely replaced by *ART* (Android RunTime)

A typical work flow

1. App written in java
2. Compiled to Java bytecode files
3. dx converts java bytecode files to a single dex bytecodefile (classes.dex)
4. Dalvik executes dex bytecode file

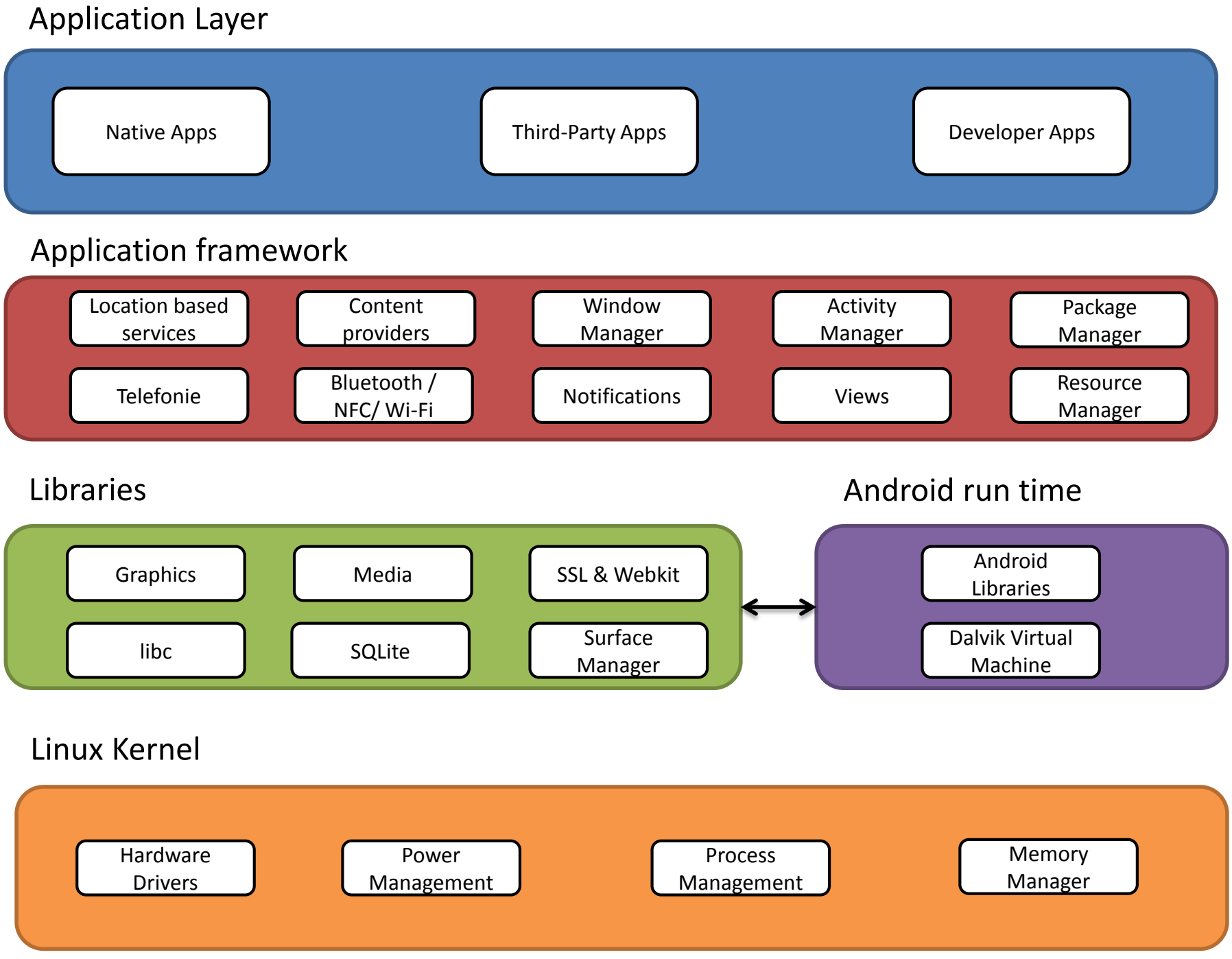
Application Framework

The application framework provides the classes which can be used to create Android applications: an interface for access to the hardware



Application Layer

All applications are build in the application layer, using the different API libraries, and executed in the Dalvik VM.


Application Layer
Native Apps
Third-Party Apps
Developer Apps
Application framework
Location based services
Content providers
Window Manager
Activity Manager
Package Manager
Telefonie
Bluetooth / NFC/ Wi-Fi
Notifications
Views
Resource Manager
Libraries
Android run time
Graphics
Media
SSL & Webkit
libc
SQLite
Surface Manager
Android Libraries
Dalvik Virtual Machine
Linux Kernel
Hardware Drivers
Power Management
Process Management
Memory Manager

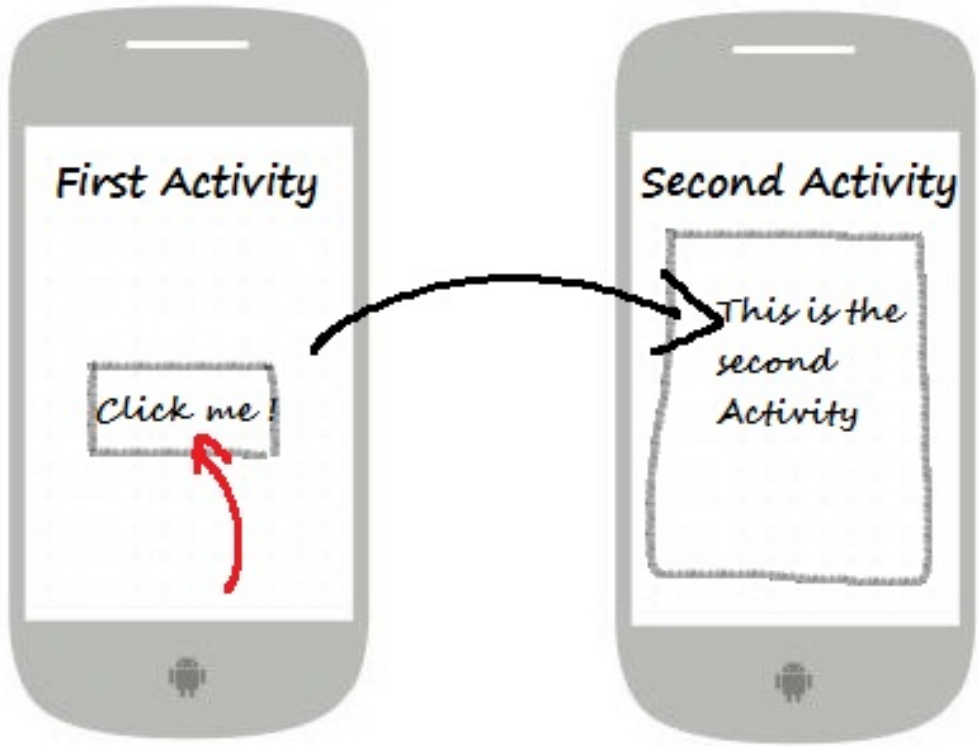
Basic Programming in Android

Google is your friend!

- `http://developer.android.com/index.html`
- `http://android-developers.blogspot.be/`
- `...`

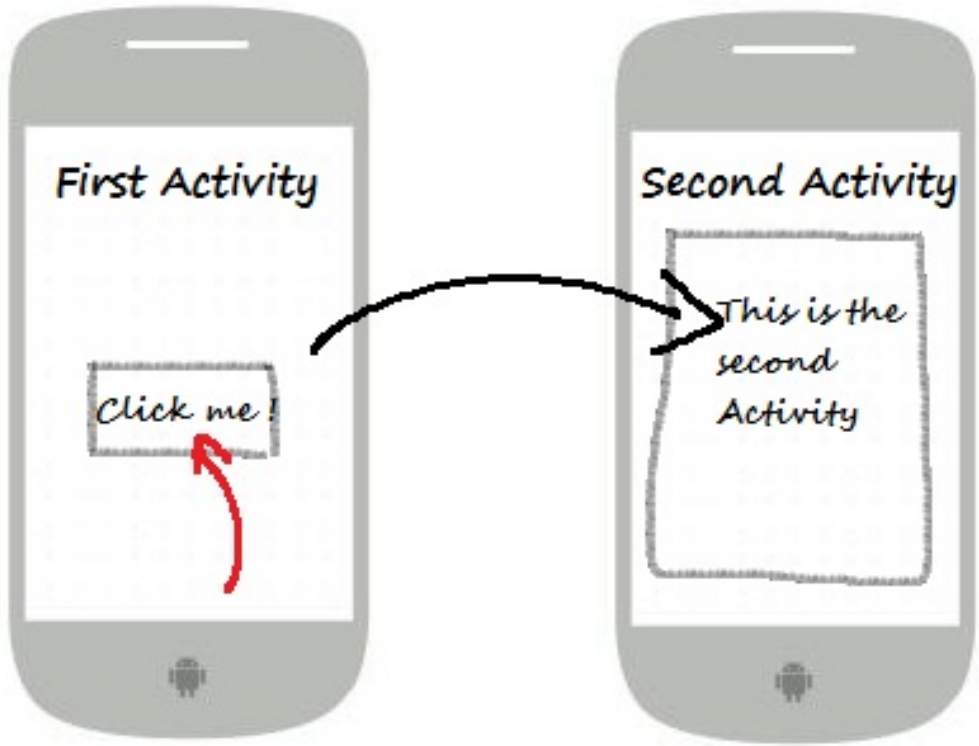
Activities & Activity Lifecycle

An Activity is a single screen the user sees on the device at one time



Activities & Activity Lifecycle

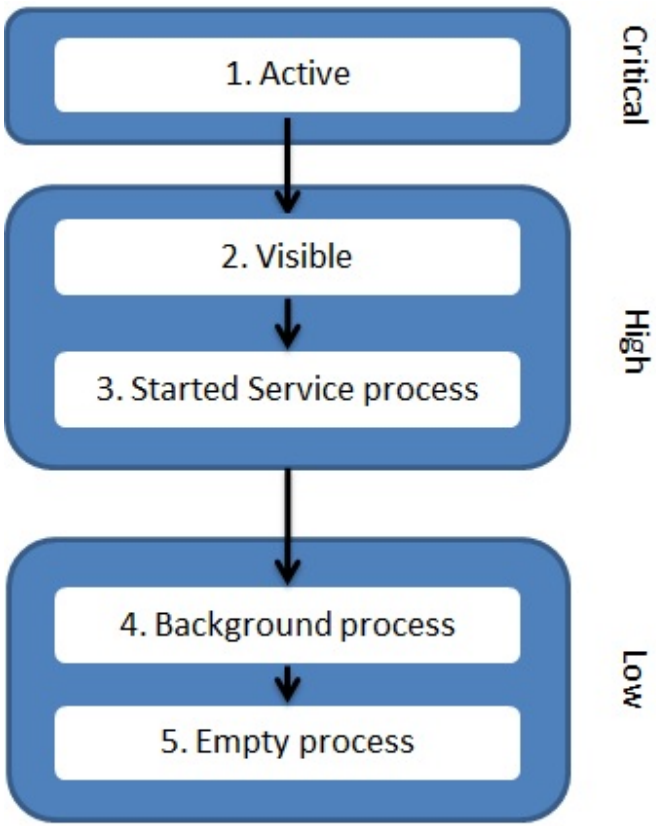
An Activity is a single screen the user sees on the device at one time



Comparable: www

Activity Life Cycle

The activity manager is responsible for creating, destroying and managing activities.



- **Active process**: have components which interact with the user
- **Visible process**: visisble but inactive process (e.g. non-full screen or transparant)
- **Started service process**: Process hosting and services
- **Background process**: process with non-visible activities and without started services
- **Empty process**: to improve memory management, apps are stored in memory

De Activity Lifecylce

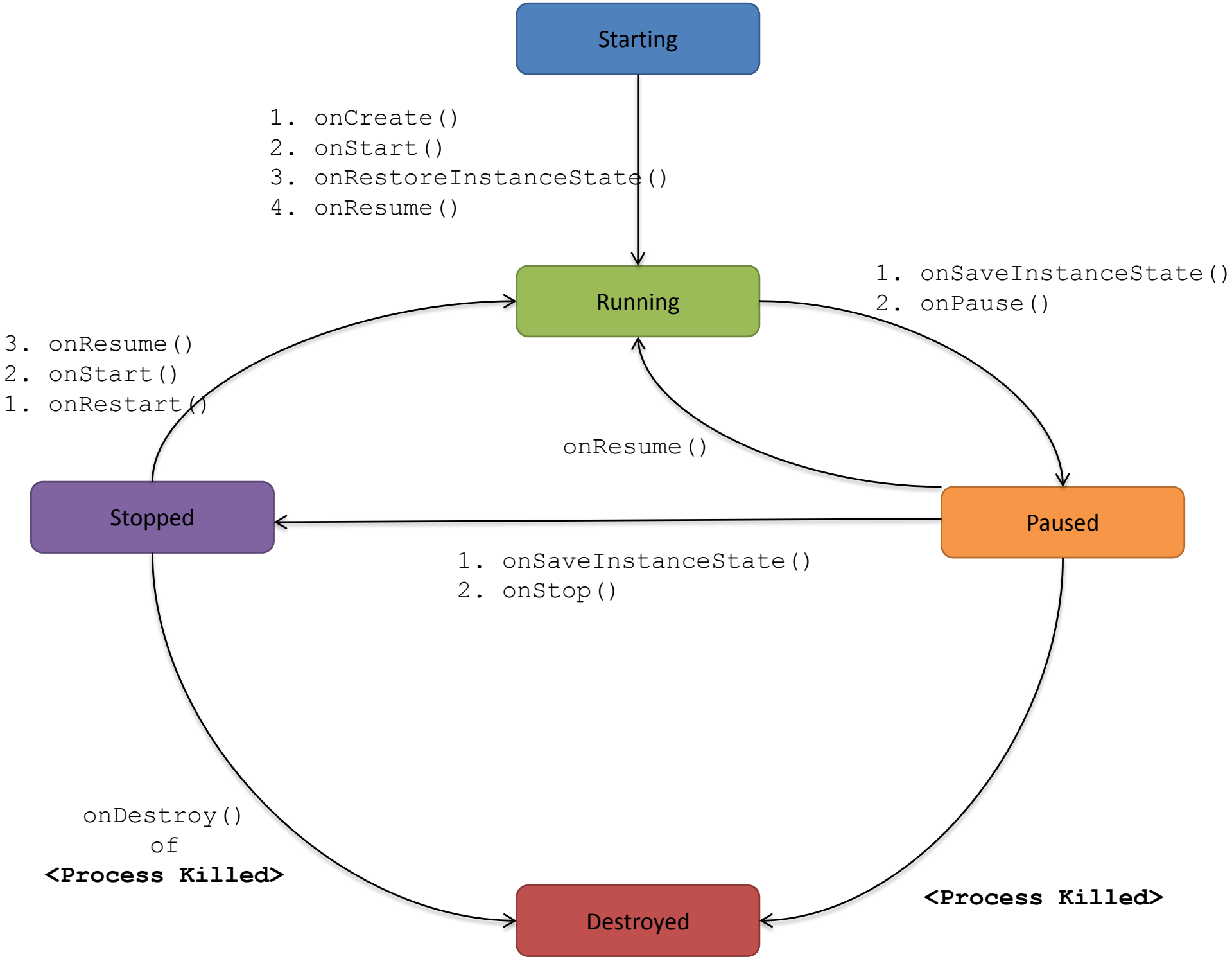
An acitivity has 4 states:

Running is on top of the activity stack, is able to be manipulated and Android will do everything to keep this alive

Paused activity still visible, but interaction is not possible (e.g. when transparant)

Stopped activity is invisible, but still in memory

Destroyed activity is not on the stack any more and needs to be restarted


Starting
1. onCreate()
2. onStart()
3. onRestoreInstanceState()
4. onResume()
Running
1. onSaveInstanceState()
2. onPause()
3. onResume()
2. onStart()
1. onRestart()
onResume()
Stopped
Paused
1. onSaveInstanceState()
2. onStop()
onDestroy()
of
<Process Killed>
<Process Killed>
Destroyed

Anatomy Android Project

Manifest File Glues the project together (building blocks, permissions ...)

String resources `res/values/strings.xml` $\Rightarrow$ all the strings for the app

Layout XML colde `res/layout/` declares the layouts of the activities

Drawable Resources all images used in the app

R file The connections between java and the external resources: automatically generated

Source Code the java source code for the app

Emulator

- The SDK comes with an emulator (lacks in performance though . . .)
- Genymotion is an emulator which is better

Demonstration

See `https://github.com/eothein/Presentation-Android-Introduction`

Goal of application

We will build an application which downloads a random Chuck Norris Joke and add this to a list

Step 1: create the activity

- Start a new project
- Choose to create a new Activity, lets call this MainActivity
- Create the layout (use LinearLayout and add a ListView and a Button)
- Do not forget to create the string constant for the button text
- You Should be able to start the application!

Step 1: create the activity

- Start a new project
- Choose to create a new Activity, lets call this MainActivity
- Create the layout (use LinearLayout and add a ListView and a Button)
- Do not forget to create the string constant for the button text
- You Should be able to start the application!

Don't forget ...

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

Asynctask

An Asynctask allows to perform background operations and publish results on the UI thread without having to manipulate threads and/or handlers.

```
public void onClick(View v) {
    new
        DownloadImageTask().execute("http://example.com/image.png");
}

private class DownloadImageTask extends AsyncTask<String, Void,
    Bitmap> {
    /** Excutes in worker thread */
    protected Bitmap doInBackground(String... urls) {
        return loadImageFromNetwork(urls[0]);
    }

    /** Returns results */
    protected void onPostExecute(Bitmap result) {
        mImageView.setImageBitmap(result);
    }
}
```

Asynctask

onPreExecute() used to setup the task

doInBackground(Params...) invoked on the background thread immediately after onPreExecute() finishes executing. This step is used to perform background computation that can take a long time.

onProgressUpdate(Progress ...) , invoked on the UI thread after a call to publishProgress(Progress...) and used to display any form of progress in the user interface

onPostExecute(Result) invoked on the UI thread after the background computation finishes

Step 2: create an AsyncTask

Create the AsyncTask The three types used by an asynchronous task are the following:

Params	the type of the parameters sent to the task upon execution.
Progress	the type of the progress units published during the background computation.
Result	the type of the result of the background computation.

Step 3: use GSON

In File $\rightarrow$ Project Structure $\rightarrow$ app $\rightarrow$ Dependencies add GSON

Gson is a Java library that can be used to convert Java Objects into their JSON representation and vice versa

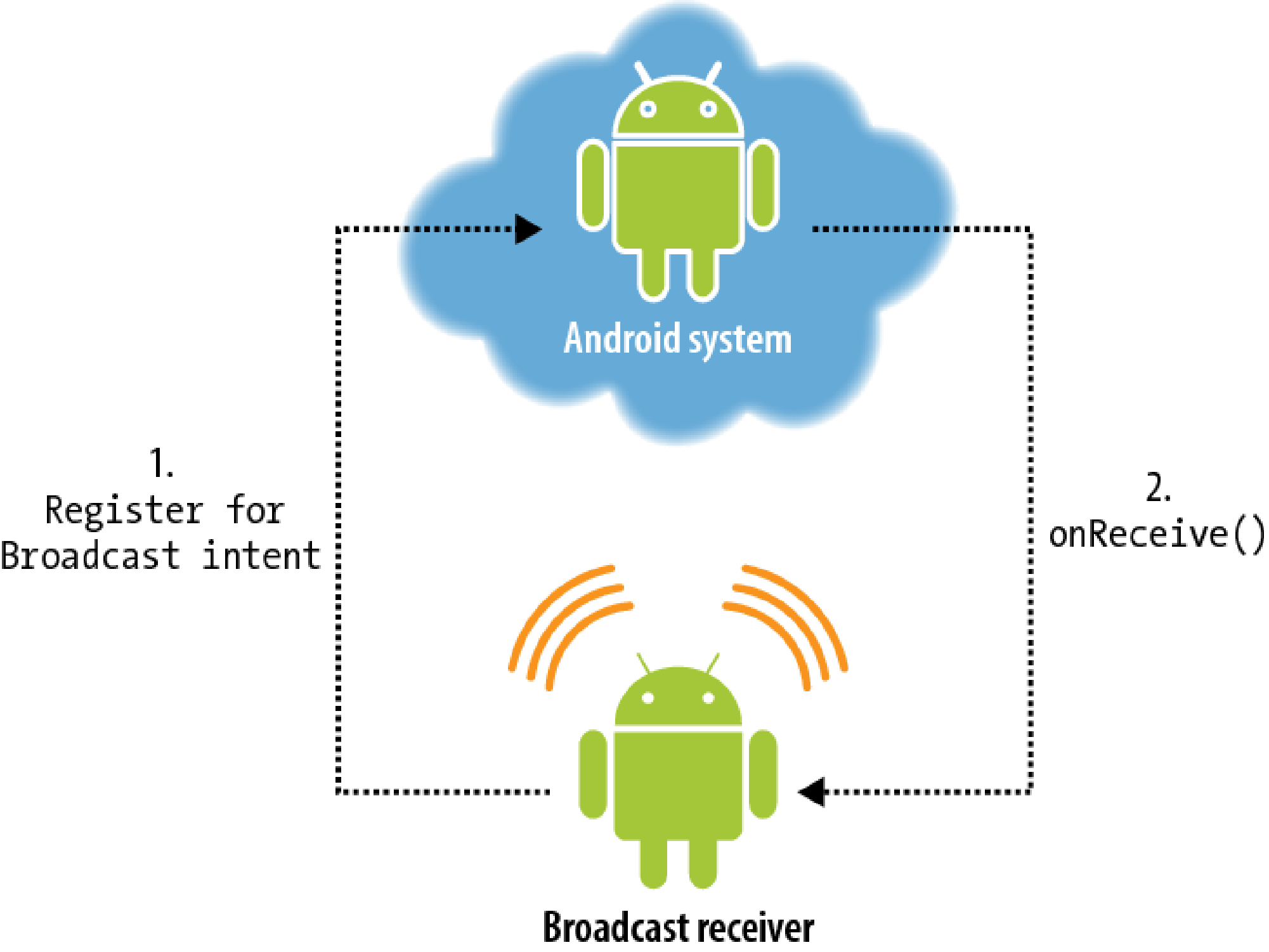
```
class BagOfPrimitives {
  private int value1 = 1;
  private String value2 = "abc";
  private transient int value3 = 3;
  BagOfPrimitives() {
    // no-args constructor
  }
}
BagOfPrimitives obj = gson.fromJson(json, BagOfPrimitives.class);
```

Step 3: use GSON

- Make a connection to `http://api.icndb.com/jokes/random?`
- Download the JSON response
- Use GSON to parse into a joke object
- In Mainactivty add a listener to the button and start the asynctask
- Test by logging to the logcat

BroadcastReceivers and Intents

A broadcast receiver is an Android component which allows you to register for system or application events. All registered receivers for an event are notified by the Android runtime once this event happens.


Android system
1.
Register for
Broadcast intent
2.
onReceive()
Broadcast receiver

Create a broadcastreceiver

- Create the JokeReceiver
- Implements the onReceive methode
- Make an interface which the activity must implements (to add the joke)
- Create the intent and send it to the broadcastreceiver

ListView and Adapter