Introduction to Android App Development

Samin Yaseer Mahmud

March 2021

Contents

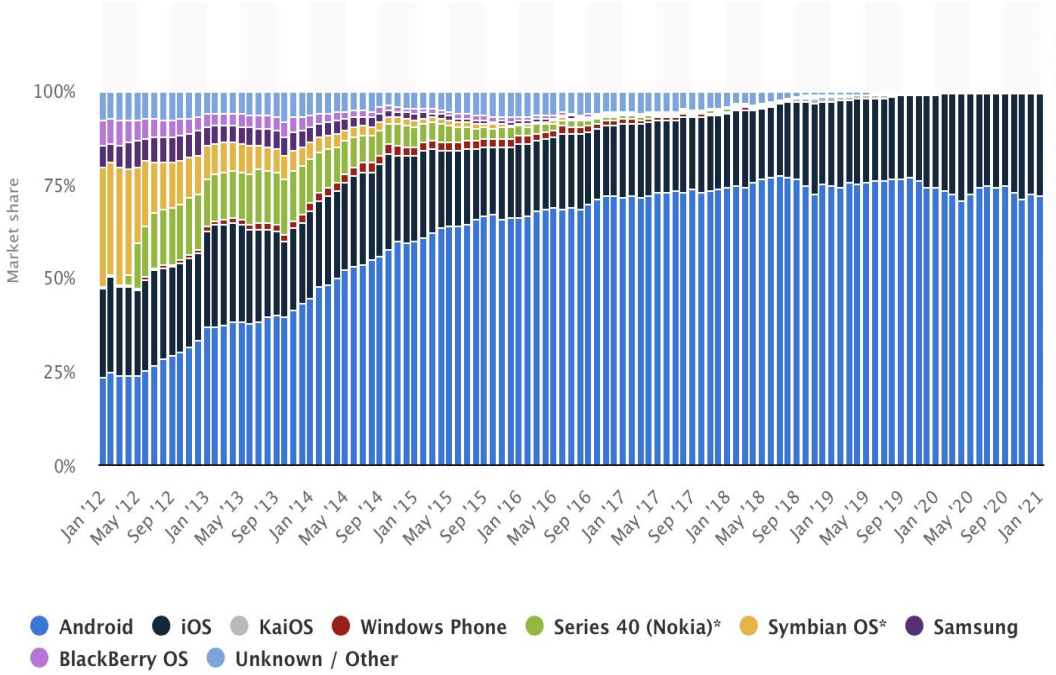
- Android Ecosystem
- Android Architecture
- Developing Environment
- App Fundamentals
- “Hello World” Application

What is Android?

- Mobile Operating System
- Based on **Linux** kernel
- Open Source

- 71.93% of Mobile Market Share
- ~3 million apps in Play Store

- Supports smart watches, TV, Auto systems

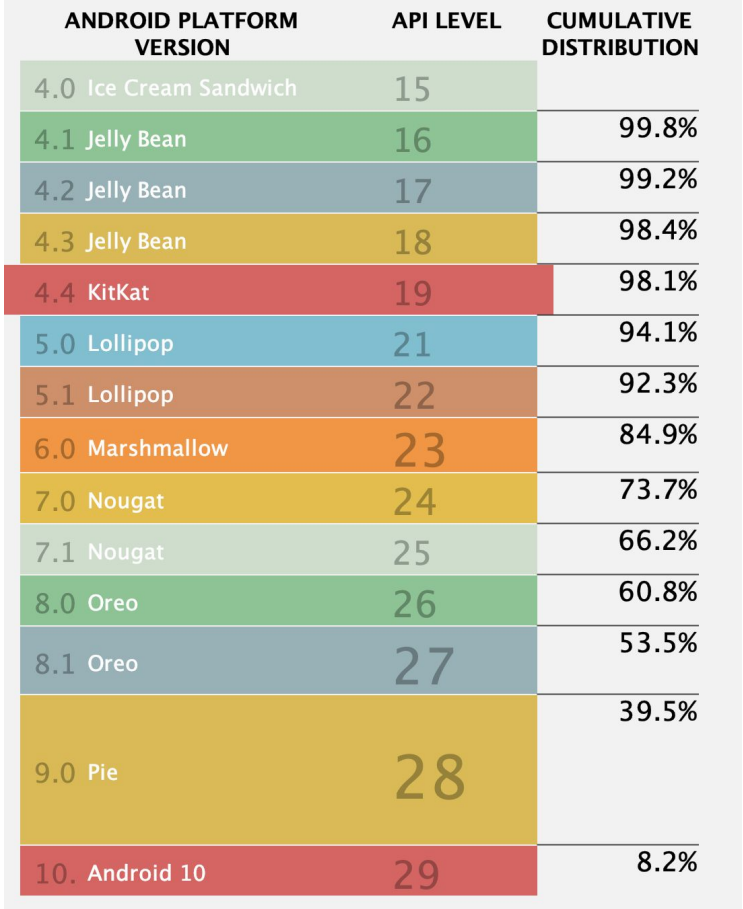


https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/

Android Versions

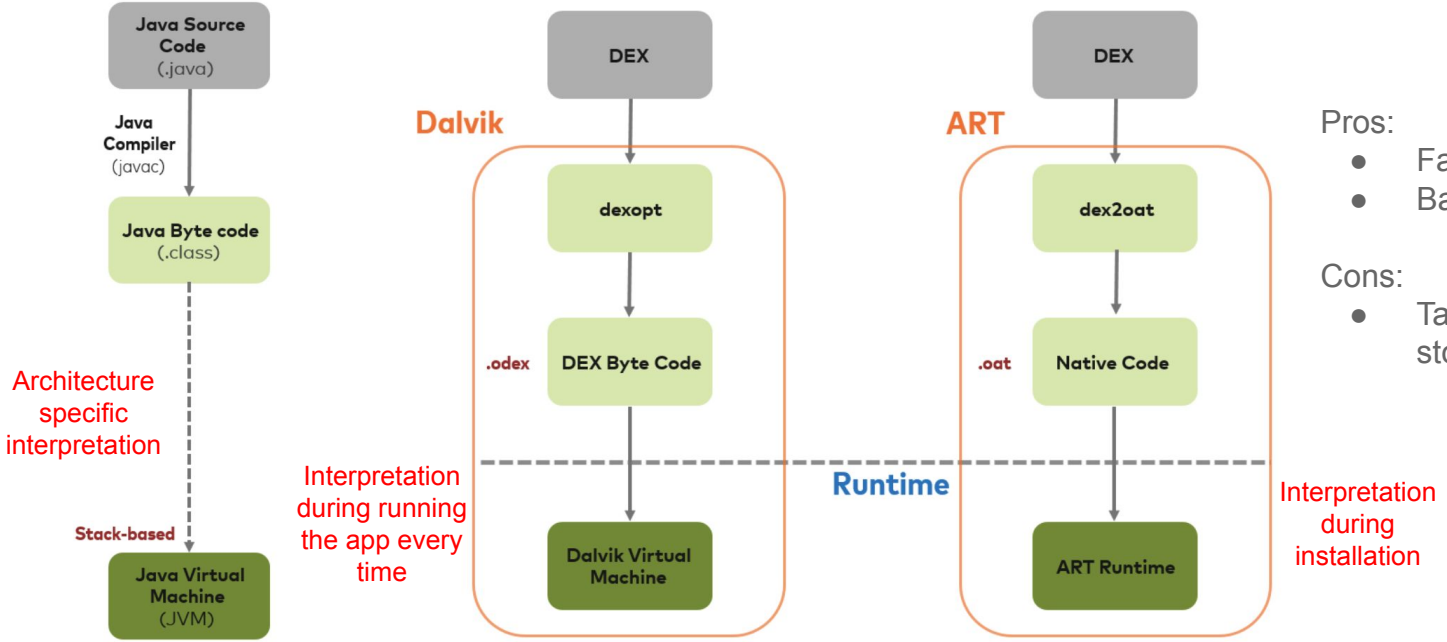
- Initial release September, 2008
- First Codename: **Cupcake** (Android 1.5)
- Dessert themed name, *alphabetically*
- Latest version: **Android 11** (September, 2020)

- Runtime Environment:
  - Dalvik (Android 1 - Kitkat)
  - **ART** (Kitkat - recent)



ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
4.0 Ice Cream Sandwich	15	
4.1 Jelly Bean	16	99.8%
4.2 Jelly Bean	17	99.2%
4.3 Jelly Bean	18	98.4%
4.4 KitKat	19	98.1%
5.0 Lollipop	21	94.1%
5.1 Lollipop	22	92.3%
6.0 Marshmallow	23	84.9%
7.0 Nougat	24	73.7%
7.1 Nougat	25	66.2%
8.0 Oreo	26	60.8%
8.1 Oreo	27	53.5%
9.0 Pie	28	39.5%
10. Android 10	29	8.2%

Android Runtime (ART)



Pros:

- Faster
- Battery Efficiency

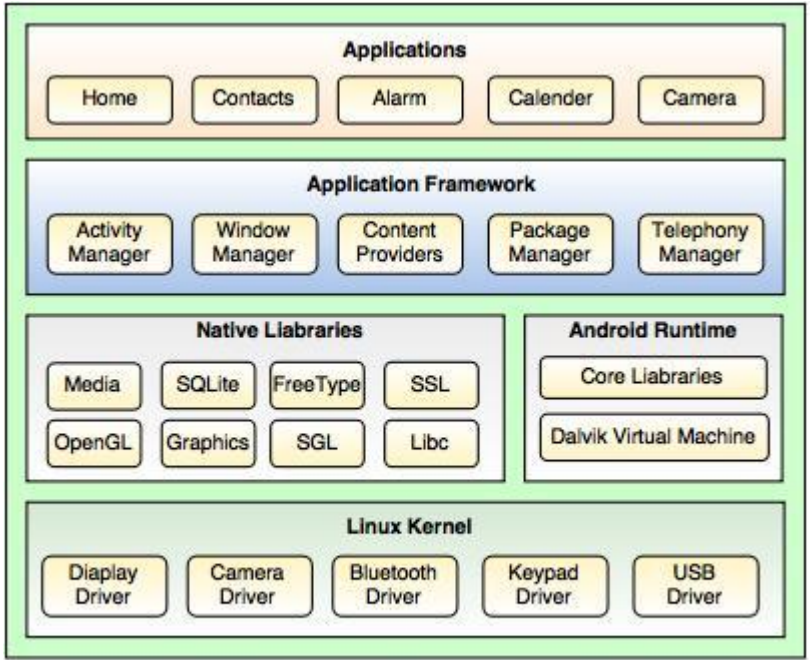
Cons:

- Takes additional storage

More on DVM vs ART

Android Architecture

- Application layer contains system apps and user defined apps
- Application framework is the Android SDK
- Each app runs in its own process with its own instance of the **Android Runtime**
- Core C/C++ libraries give access to system components
- Kernel mandates access to system resources (i.e hardware)



https://www.tutorialride.com/android/architecture-of-android.htm

Challenges of Android app development

- Support multiple screen size and multiple resolution
- Support devices running older OS
- Make app responsive and smooth
- Keep source code and user data secure
- Understanding user needs and demands

What is an Android app?

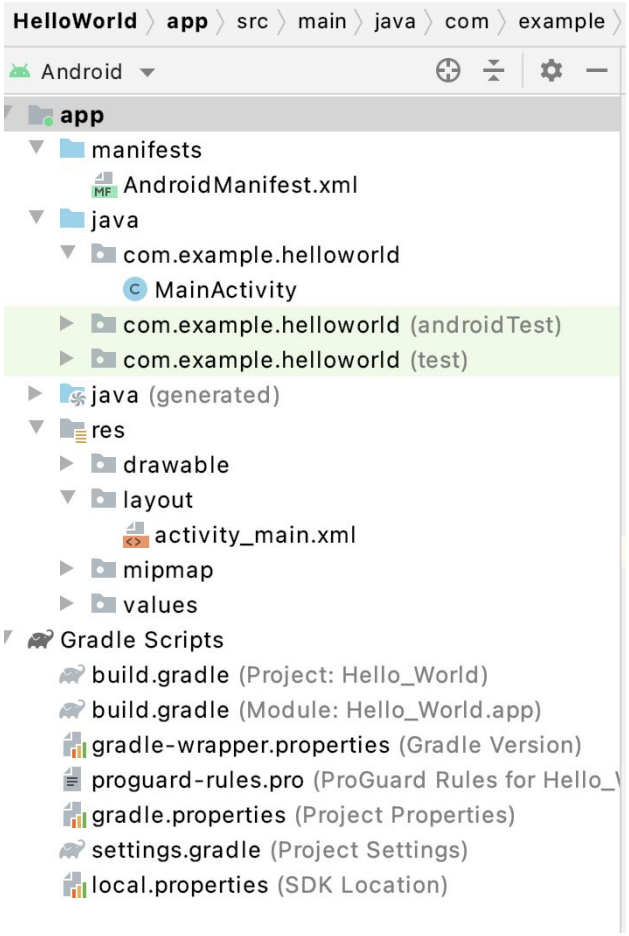
- One of more **interactive** screens
- App **logic** developed in Java or Kotlin
- App **layouts** designed using XML
- Uses Android Software Development Kit (**SDK**)
- Uses **Android libraries** and **Android Application Framework**
- Executed by **Android Runtime** (ART)

Android app development requirements

- Mac, Windows or Linux
- Java Development Kit (JDK) **1.7 or above** from Oracle Java SE
- IDE: **Android Studio** from Android Developers website
- **Android SDK** from Android Studio SDK manager
- Android Virtual Device (AVD) or Real Device for debugging

Android project structure

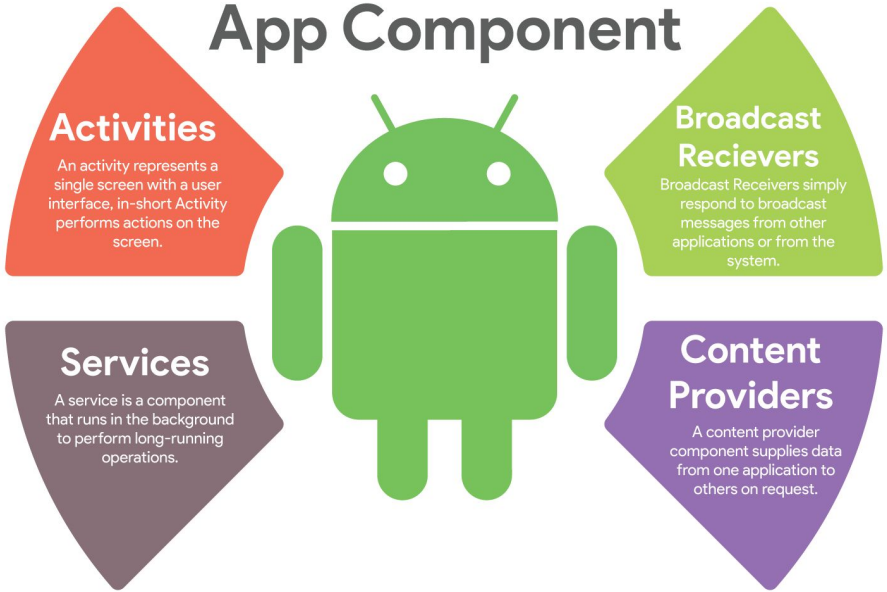
- **Manifest file**: information about app, i.e permissions, components
- **Java files:** activities, services, and other java files.
- **Resource files**: layouts, strings, colors as XML and other media files
- **Configuration files:** gradle scripts, proguard rules



Android Components: The building blocks

Basic Components

1. Activity
2. Services
3. Content Provider
4. Broadcast Receiver



Additional Components: Intent ,Fragment, Views, Layout, Resource, Manifest

Activity

- The user interface consists of a series of **Activity**
- Each activity represents **one window**
- Typically fills the screen, but can be embedded in other activity or appear as a floating window
- An activity typically has **UI layout**
- Layout is usually defined in one or more **XML** files
- Activity **inflates** layout as part of being created
- An activity has a **lifecycle**, several **states** in its lifecycle

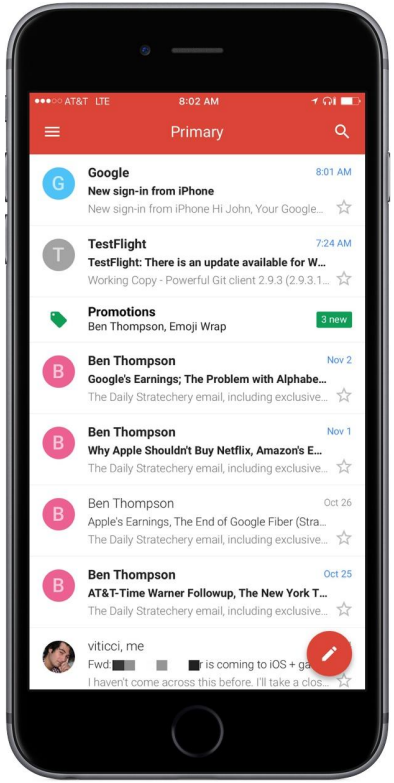
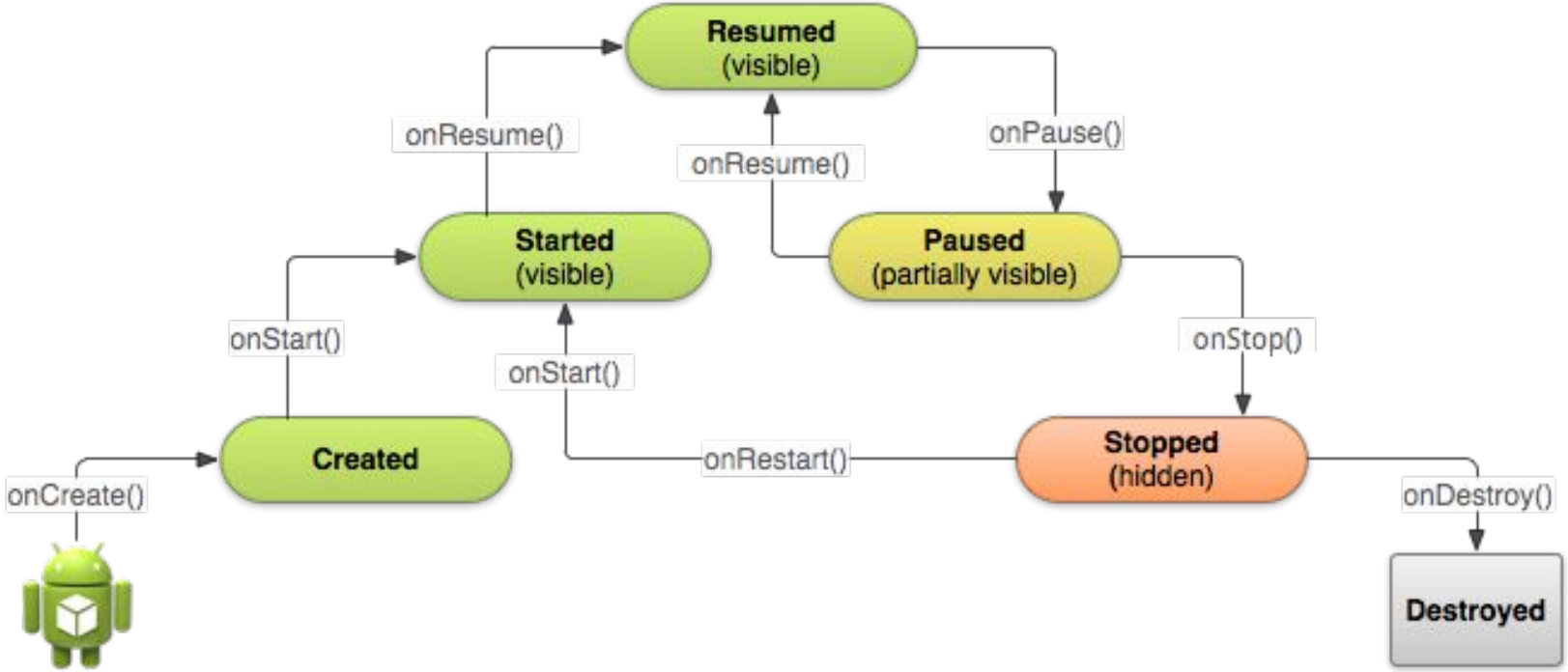


Fig: An activity of Gmail app

Activity LifeCycle



Code Snippet: Activity

```java
package com.example.helloworld;

import ...

public class MainActivity extends Activity {

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

    }


}
```

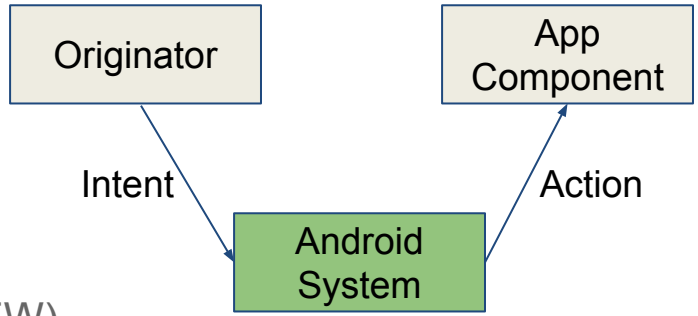
Fig: MainActivity.java

```
<?xml versio
<androidx.co                                    yout
    xmlns:an                                    apk/res/android"
    xmlns:ap                                    res-auto"
    xmlns:to                                    ols"
    android:
    android:
    android:
    tools:co

    <TextVie
        andr
        andr
        andr
        andr
        tool
        tool                                    />

</androidx.c                                    ayout>
```

Intent

An Intent is an object used for inter component signaling via the Android system

- Start an activity of same/different application
- Start background service
- Send message between components
- Deliver broadcasts

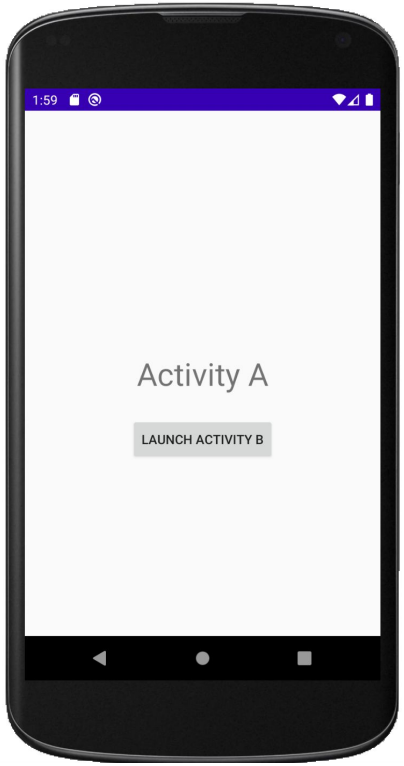


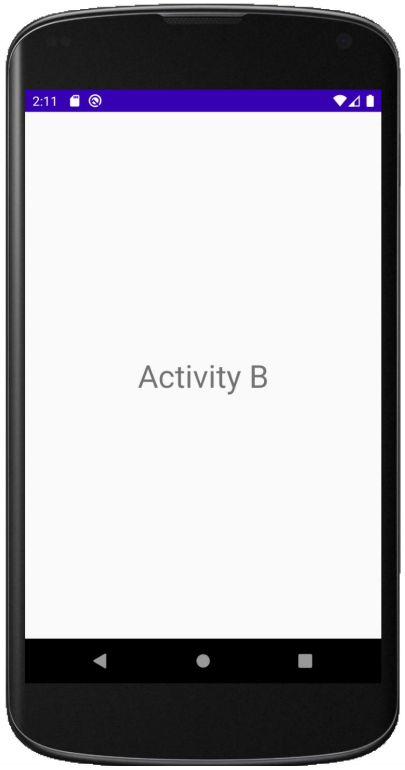
Two types

- Explicit: names the target component class
- Implicit: specifies an “action string” (e.g ACTION_VIEW)

Code Snippet: Explicit Intent

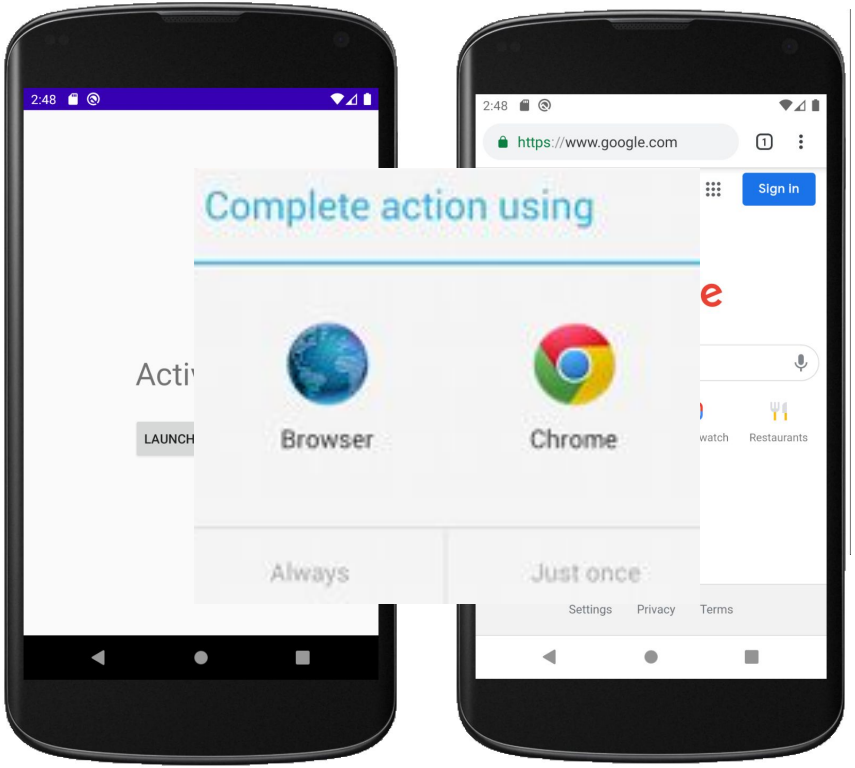
```
public class MainActivity extends Activity {

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

        Button buttonOne = (Button) findViewById(R.id.button3);
        buttonOne.setOnClickListener(new View.OnClickListener(){

            @Override
            public void onClick(View view) {

                launchActivityB();
            }
        });


    }

  private void launchActivityB(){

      Intent i = new Intent( packageContext: this, ActivityB.class);
      i.putExtra( name: "KEY",  value: "VALUE");
      startActivity(i);
  }
```





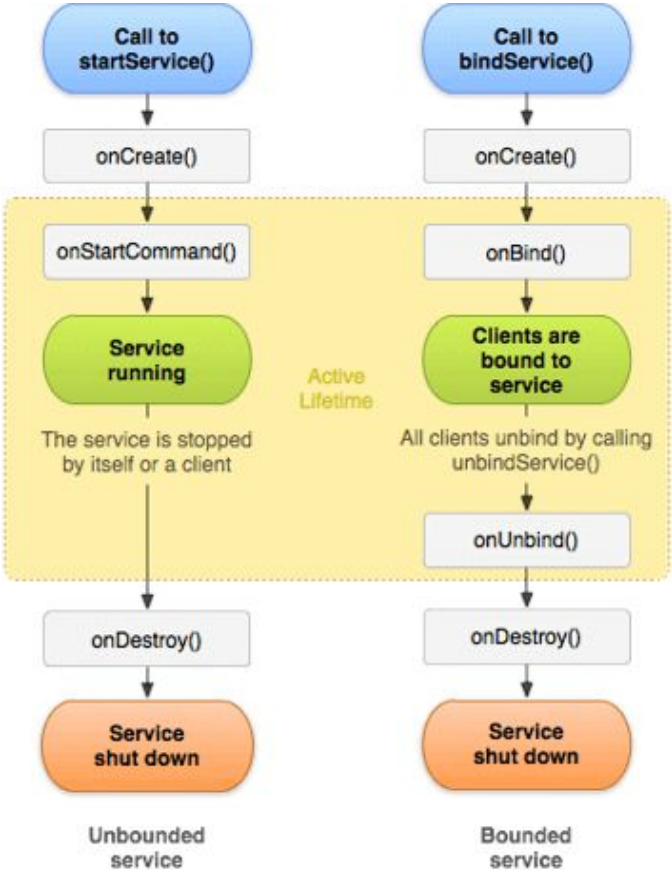
Code Snippet: Implicit Intent

```
public class MainActivity extends Activity {

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

        Button buttonOne = (Button) findViewById(R.id.button3);
        buttonOne.setOnClickListener(new View.OnClickListener(){

            @Override
            public void onClick(View view) {

                launchBrowser();
            }
        });
    }

private void launchBrowser(){

    Intent i = new Intent(Intent.ACTION_VIEW, Uri.parse("https://www.google.com"));
    startActivity(i);
}
```



Services

- Background processing occurs in **Service** components
  - Downloading a file, playing music, tracking location, etc.
- Does not have any user interface
- **Two types:** bounded, unbounded
- Unbounded: not binded with any component
- Services starts with **startService()**, ends with **stopService()**
- Bounded services starts with **bindService()**, **multiple client** can bind to it
- Ends when all clients **unbindService()**
- UI updates through **Broadcast Receiver**



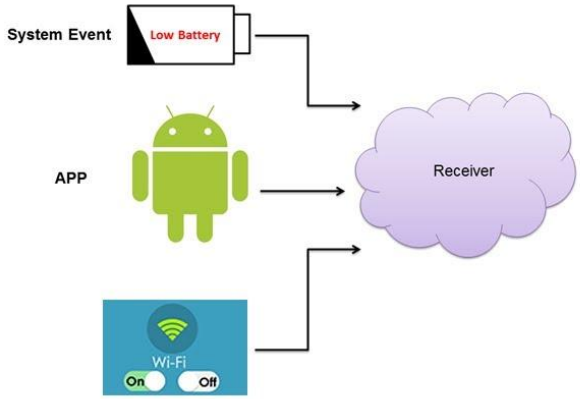
Code Snippet: Unbounded Service

```java
public class MainActivity2 extends AppCompatActivity {

    private Button start, stop;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);

        start = (Button) findViewById( R.id.startButton );
        stop = (Button) findViewById( R.id.stopButton );
        final Intent service = new Intent( packageContext: this,
                NewService.class);


        start.setOnClickListener(new View.OnClickListener(){
            @Override
            public void onClick(View view) {
                startService(service );
            }
        });

        stop.setOnClickListener(new View.OnClickListener(){

            @Override
            public void onClick(View view) {
                stopService(service);
            }
        });
```

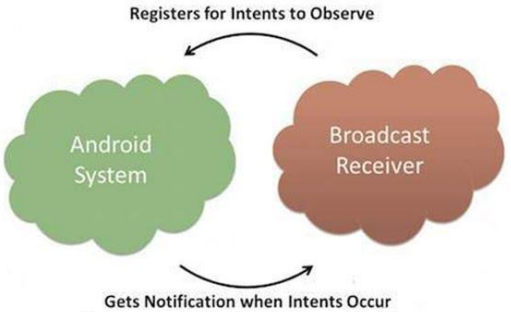
```java
public class NewService extends Service {

    private MediaPlayer player;

    public NewService() {
    }

    public int onStartCommand(Intent intent, int flags, int startId) {

        player = MediaPlayer.create( context: this,
                Settings.System.DEFAULT_RINGTONE_URI );
        player.setLooping( true );
        player.start();
        return START_STICKY;
    }

    public void onDestroy() {
        super.onDestroy();
        player.stop();
    }


    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }
}
```

Broadcast Receiver

- Notify app components when various **system event** occurs (i.e wifi status changed, battery low)
- Also used to send **custom broadcast messages** (i.e specific file downloaded for specific app)
- Application components **registers** for specific **event**
- Android system delivers broadcast message **even app is not running**
- Broadcast receiver does not need to be running all the time
- If a registered event is detected, android system automatically wakes up the Broadcast receiver



- **Security**:
  - Other apps can send broadcast to your receiver
  - Other apps can respond to broadcasts you send
  - Enforce access by assigning permission to sender or receiver

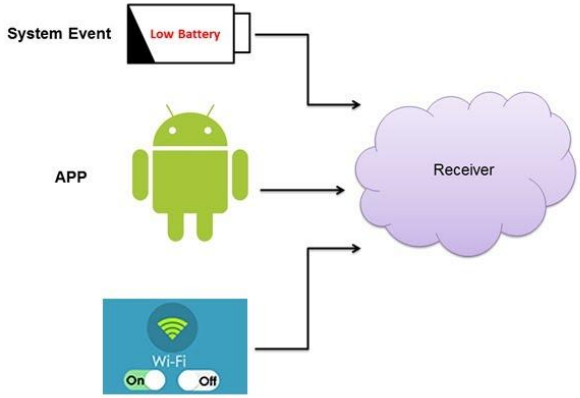


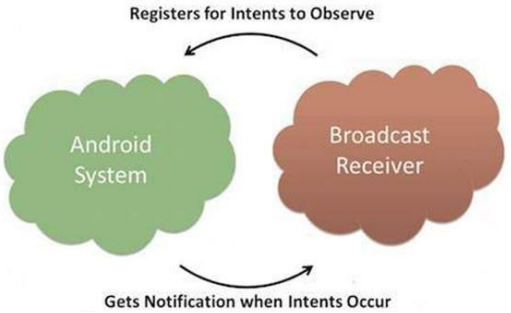
Broadcast Receiver

- Creating Broadcast Receiver:
  - Extend **BroadcastReceiver** class
  - Override **onReceive()** method -> handle what your receiver does inside this method

- Register for events:
  - Statically, using i**ntent-filter** in Manifest file
  - Works even when app is not running
  - Dynamically, **registerReceiver()** method
  - Invocation period is defined

- Broadcasting events:
  - System broadcast events are triggered automatically
  - Custom broadcast events are manually triggered by **sendBroadcast()** method





Code Snippet: Broadcast Receiver

```java
public class MyReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {

        Toast.makeText(context, text: "Broadcast Detected.",
                Toast.LENGTH_LONG).show();
    }
}
```

```xml
<receiver
    android:name=".MyReceiver"
    android:enabled="true"
    android:exported="true">

    <intent-filter>
        <action android:name="android.intent.action.BATTERY_LOW">
        </action>
    </intent-filter>

</receiver>
```

```java
public class BroadcastReceiverActivity extends AppCompatActivity {

    MyReceiver br;
    IntentFilter i;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_broadcast_receiver);

        br = new MyReceiver();
        i = new IntentFilter( action: "android.intent.action.BATTERY_LOW");
        registerReceiver(br,i);
    }
    @Override
    protected void onStart() {
        super.onStart();
        //registerReceiver(br,i);
    }
    @Override
    protected void onStop() {
        super.onStop();
        //unregisterReceiver(br);
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
        unregisterReceiver(br);
    }
}
```

Content Provider

- Provides a standardized interface for sharing data i.e content (between applications)
- Models content in a relational DB
  - Users of content providers can perform queries equivalent to SELECT, UPDATE, INSERT etc.
  - Works well when content is tabular
  - Mechanism for defining data security
- URI addressing scheme
  - content://<authority>/<path>/[<id>]
  - content://com.android.example/people/10

Manifest file

Manifest files are the technique for describing contents of an application package

Typically Contains:

- App's package name
- Required system permissions
- Contained app components
- Icons and Labels
- Intent filters
- Hardware and software features
- App properties: auto backup, debuggable

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.helloworld">

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Hello World"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <activity android:name=".ActivityB">
        </activity>

    </application>

</manifest>
```

Fig: AndroidManifest.xml

Android Studio

Demo on:

- Android Studio
- SDK Manager
- Android Emulators
- Hello World App

Thank you

Samin Yaseer Mahmud
Ph.D Student, Dept. of Computer Science
Research Assistant, WSPR Lab
North Carolina State University
Email: smahmud@ncsu.edu

Content acknowledgement: William Enck, Lalith Singh