Introduzione ad Android

Lezione 3
Elements of Android OS

Ruggero Donida Labati

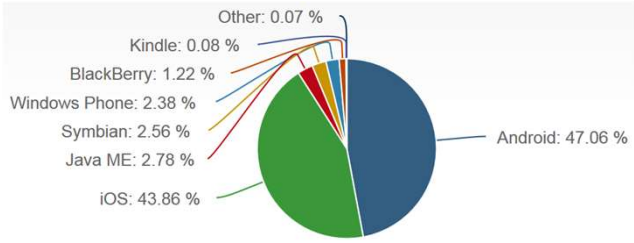
Laboratorio di Sistemi Operativi
Università degli Studi di Milano
Dipartimento di Informatica
A.A. 2024/2025

Sommario

1. The Android platform
2. Kernel
3. Libraries
4. Framework
5. Applications
6. Building an application

The Android platform (1/4)

- Current platforms
  - iOS
  - Android OS
  - Windows phone
  - Blackberry...



The Android platform (2/4)

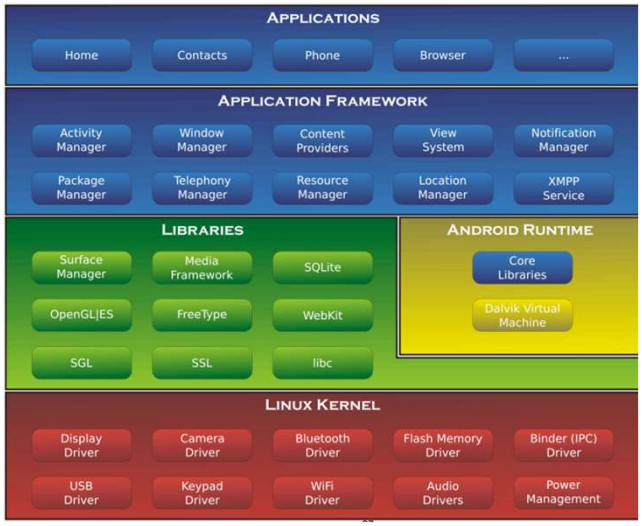
- A software stack for mobile devices
  - OS kernel
  - System libraries
  - Application frameworks
  - Key apps
- Android SDK for creating apps
  - Libraries and development tools
  - Lots of documentation

The Android platform (3/4)

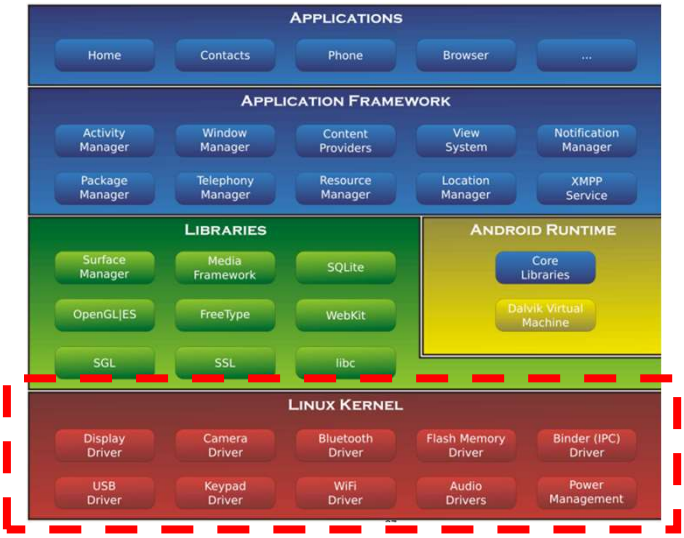
The Android platform (4/4)

- Application framework
- Dalvik virtual machine
- Integrated browser
- Optimized graphics
- SQLite
- Media support
- GSM, Bluetooth, EDGE, 3G, and Wi-Fi
- Camera, GPS, compass, and accelerometer
- Rich development environment

Android kernel (1/3)



Android kernel (1/3)



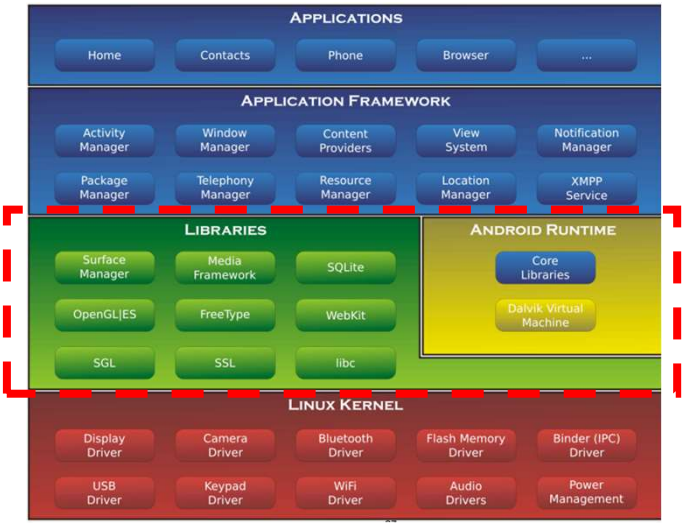
Android kernel (2/3)

- Linux kernel
  - Standard services
    - Security
    - Memory management
    - Process management
    - File and network I/O
    - Device drivers

Android kernel (3/3)

- Linux kernel
  - Android-specific
    - Power management
    - Android shared memory
    - Low memory killer
    - Interprocess communication
    - …

Android libraries (1/5)



Android libraries (2/5)

- Libraries
  - System C
  - Webkit
  - Surface manager
    - Display management
  - OpenGL
    - Graphics engines
  - Media framework
    - Audio/video
  - SQLite
    - Relational database engine

Android libraries (3/5)

- Android runtime
  - Core Java libraries
    - Basic Java classes
    - App lifecycle
    - Internet/web services
    - Unit testing

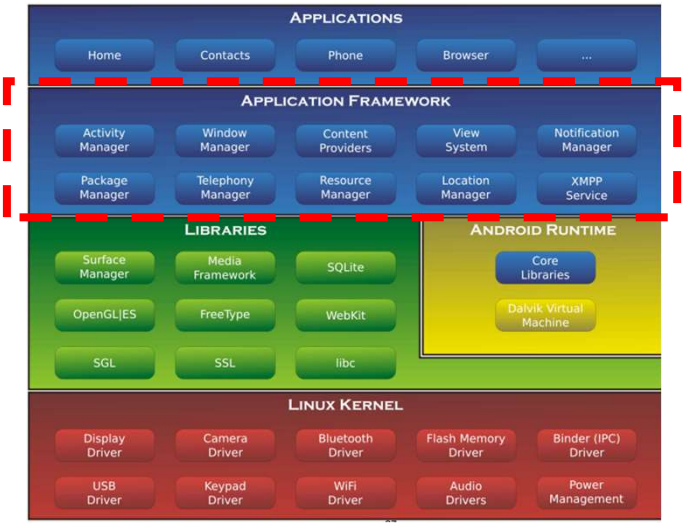
Android libraries (4/5)

- Android runtime
  - Dalvik Virtual Machine
    - Executes app
    - Resource-contrained environments
    - Slower CPU
    - Less RAM
    - Limited battery life

Android libraries (5/5)

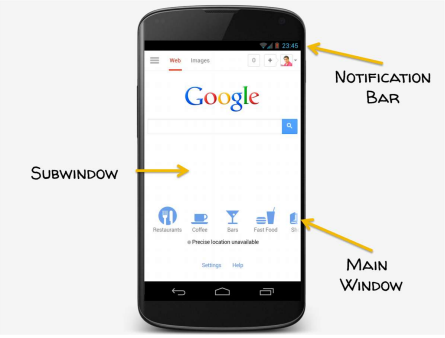
- Workflow
  - App written in Java
  - Compiled to Java bytecode files
  - DX converts Java bytecode to a single DEX bytecode file
  - Dalvik executes the DEX bytecode file

Android application framework (1/8)



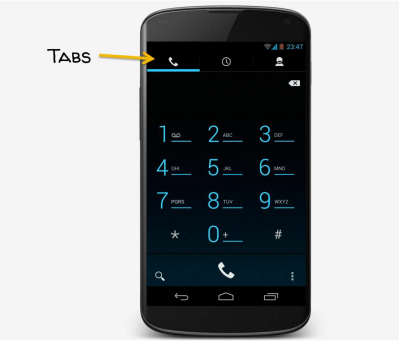
Android application framework (2/8)

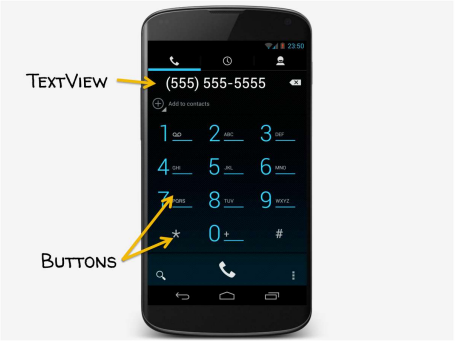
- Package manager
  - Keeps track of app packages on device
- Window manager
  - Manages the windows comprising an app



Android application framework (3/8)

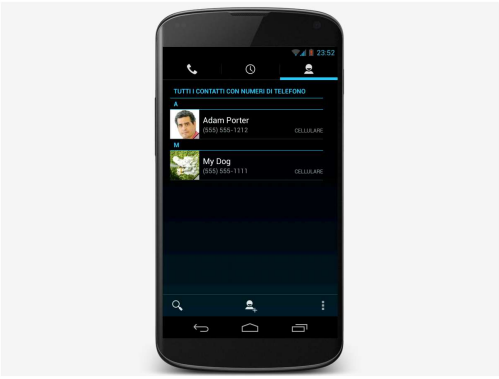
- View system
  - Provides common user interface elements
  - Icons, text boxes, buttons, …





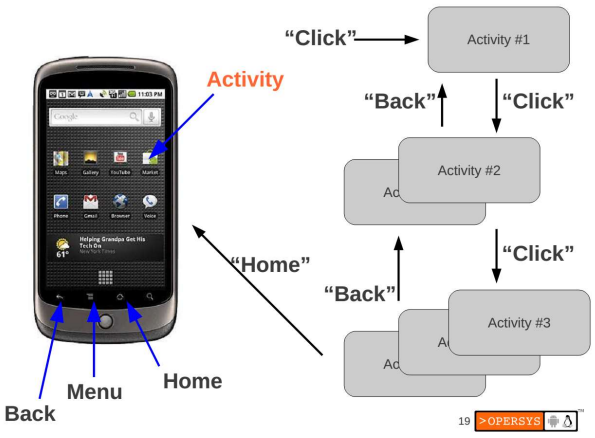
Android application framework (4/8)

- Resource manager
  - Manages non-compiled resources
  - Strings, graphics, layout files



Android application framework (5/8)

- Activity manager
  - Manages app lifecycle and navigation stack

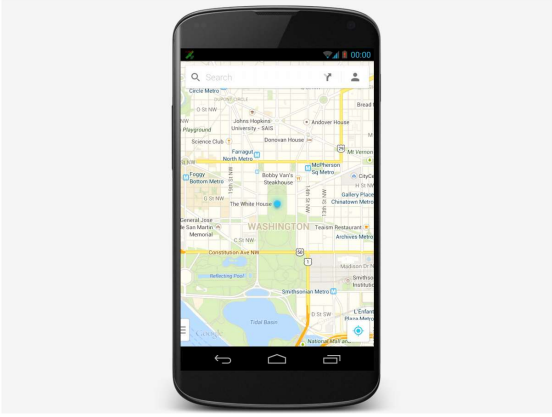


Android application framework (6/8)

- Content provider
  - Inter-application data sharing

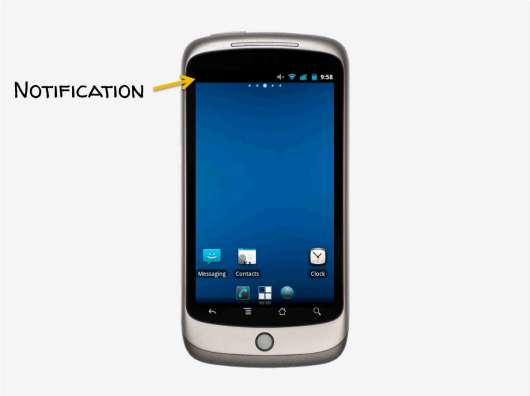
Android application framework (7/8)

- Location manager
  - Provides location and movement information

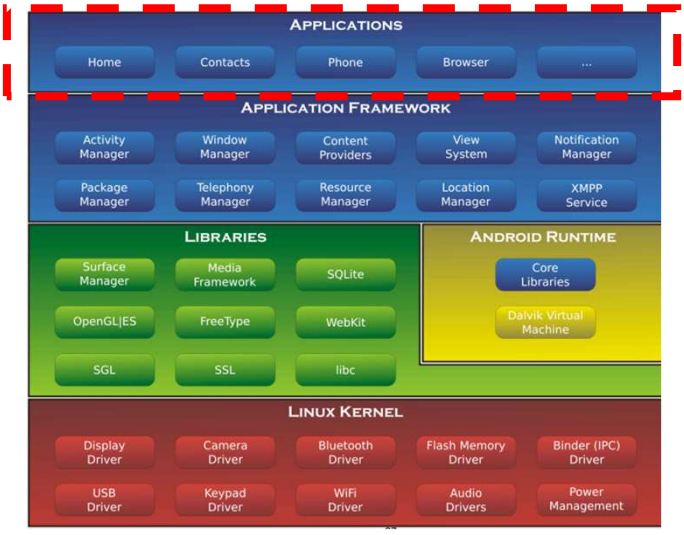


Android application framework (8/8)

- Notification manager
  - Place notification icons in the status bar when important events occur



Android applications (1/12)



Android applications (2/12)

- Standard apps
  - Home – main screen
  - Contacts
  - Phone
  - Browser
  - Email client

Android applications (3/12)

- Application components
  - Activity
  - Service
  - Broadcastreceiver
  - contentprovided

Android applications (4/12)

- Applications
  - Apps are made from components
  - Android instantiates and runs them as needed
  - Each component has its own purpose and APIs

Android applications (5/12)

- Activity
  - Primary class for user interaction
  - Usually implements a single, focused task that the user can do

Android applications (6/12)

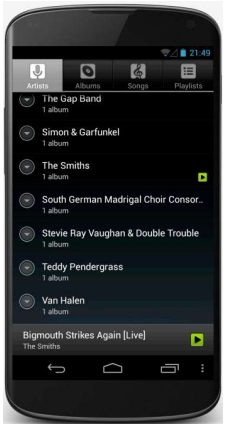
```
package com.android.contacts.activities;

import android.app.ActionBar;

/**
 * The dialer activity that has one tab with the virtual 12key
 * dialer, a tab with recent calls in it, a tab with the contacts and
 * a tab with the favorite. This is the container and the tabs are
 * embedded using intents.
 * The dialer tab's title is 'phone', a more common name (see strings.xml).
 */
public class DialtactsActivity extends TransactionSafeActivity
        implements View.OnClickListener {
    private static final String TAG = "DialtactsActivity";

    public static final boolean DEBUG = false;

    /** Used to open Call Setting */
    private static final String PHONE_PACKAGE = "com.android.phone";
    private static final String CALL_SETTINGS_CLASS_NAME =
            "com.android.phone.CallFeaturesSetting";

    /**
     * Copied from PhoneApp. See comments in Phone app for more detail.
     */
    public static final String EXTRA_CALL_ORIGIN = "com.android.phone.CALL_ORIGIN";
    /** @see #getCallOrigin() */
    private static final String CALL_ORIGIN_DIALTACTS =
            "com.android.contacts.activities.DialtactsActivity";

    /**
     * Just for backward compatibility. Should behave as same as {@link Intent#ACTION_DIAL}.
     */
    private static final String ACTION_TOUCH_DIALER = "com.android.phone.action.TOUCH_DIALER";

    /** Used both by {@link ActionBar} and {@link ViewPagerAdapter} */
    private static final int TAB_INDEX_DIALER = 0;
    private static final int TAB_INDEX_CALL_LOG = 1;
    private static final int TAB_INDEX_FAVORITES = 2;

    private static final int TAB_INDEX_COUNT = 3;

    private SharedPreferences mPrefs;

    /** Last manually selected tab index */
```

Android applications (7/12)

- Service
  - Runs in background
  - Performs long-running operations
  - Supports interaction with remote processes

Android applications (8/12)

```
package com.android.music;

import android.app.Notification;

/**
 * Provides "background" audio playback capabilities, allowing the
 * user to switch between activities without stopping playback.
 */
public class MediaPlaybackService extends Service {
    /** used to specify whether enqueue() should start playing
     * the new list of files right away, next or once all the currently
     * queued files have been played
     */
    public static final int NOW = 1;
    public static final int NEXT = 2;
    public static final int LAST = 3;
    public static final int PLAYBACKSERVICE_STATUS = 1;

    public static final int SHUFFLE_NONE = 0;
    public static final int SHUFFLE_NORMAL = 1;
    public static final int SHUFFLE_AUTO = 2;

    public static final int REPEAT_NONE = 0;
    public static final int REPEAT_CURRENT = 1;
    public static final int REPEAT_ALL = 2;

    public static final String PLAYSTATE_CHANGED = "com.android.music.playstatechanged";
    public static final String META_CHANGED = "com.android.music.metachanged";
    public static final String QUEUE_CHANGED = "com.android.music.queuechanged";

    public static final String SERVICECMD = "com.android.music.musicservicecommand";
    public static final String CMDNAME = "command";
    public static final String CMDTOGGLEPAUSE = "togglepause";
    public static final String CMDSTOP = "stop";
    public static final String CMDPAUSE = "pause";
    public static final String CMDPLAY = "play";
    public static final String CMDPREVIOUS = "previous";
    public static final String CMDNEXT = "next";
```



Android applications (9/12)

- BroadcastReceiver
  - Component that listens for and responds to events
  - The subscriber in publish/subscribe pattern
  - Events represents by the intent class and then broadcast
  - BroadcastReceiver receives and responds to broadcast event

Android applications (10/12)

```
package com.android.mms.transaction;

import android.app.Service;

/**
 * Handle incoming SMSes.  Just dispatches the work off to a Service.
 */
public class SmsReceiver extends BroadcastReceiver {
    static final Object mStartingServiceSync = new Object();
    static PowerManager.WakeLock mStartingService;
    private static SmsReceiver sInstance;

    public static SmsReceiver getInstance() {
        if (sInstance == null) {
            sInstance = new SmsReceiver();
        }
        return sInstance;
    }

    @Override
    public void onReceive(Context context, Intent intent) {
        onReceiveWithPrivilege(context, intent, false);
    }

    protected void onReceiveWithPrivilege(Context context, Intent intent, boolean privileged) {
        // If 'privileged' is false, it means that the intent was delivered to the base
        // no-permissions receiver class.  If we get an SMS_RECEIVED message that way, it
        // means someone has tried to spoof the message by delivering it outside the normal
        // permission-checked route, so we just ignore it.
        if (!privileged && intent.getAction().equals(Intents.SMS_RECEIVED_ACTION)) {
            return;
        }

        intent.setClass(context, SmsReceiverService.class);
        intent.putExtra("result", getResultCode());
        beginStartingService(context, intent);
    }
```

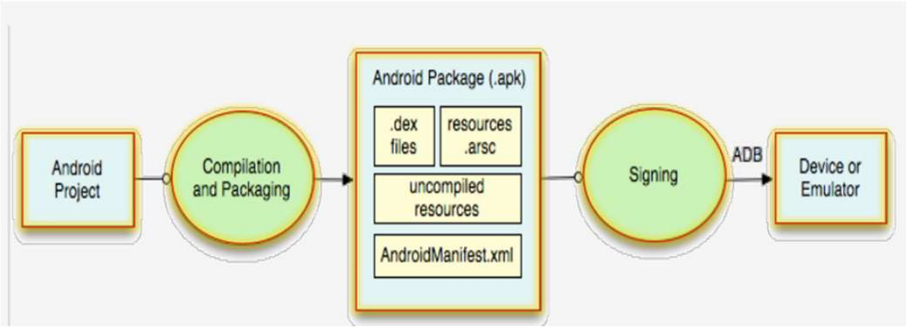
Android applications (11/12)

- Content Providers
  - Store and share data across applications
  - Uses database-style interface
  - Handles interprocess communication

Android applications (12/12)

```
package com.android.browser.provider;

import java.io.File;


public class BrowserProvider extends ContentProvider {

    private SQLiteOpenHelper mOpenHelper;
    private BackupManager mBackupManager;
    static final String sDatabaseName = "browser.db";
    private static final String TAG = "BrowserProvider";
    private static final String ORDER_BY = "visits DESC, date DESC";

    private static final String PICASA_URL = "http://picasaweb.google.com/m/" +
            "viewer?source=androidclient";

    static final String[] TABLE_NAMES = new String[] {
        "bookmarks", "searches"
    };
    private static final String[] SUGGEST_PROJECTION = new String[] {
            "_id", "url", "title", "bookmark", "user_entered"
    };
    private static final String SUGGEST_SELECTION =
            "(url LIKE ? OR url LIKE ? OR url LIKE ? OR url LIKE ?"
                + " OR title LIKE ?) AND (bookmark = 1 OR user_entered = 1)";
    private String[] SUGGEST_ARGS = new String[5];
```

Building an application (1/8)



Building an application (2/8)

- Creating an Android app
  - Define resources
  - Implement application classes
  - Package application
  - Install and run application

Building an application (3/8)

- The activity class
  - Provides a visual interface for user interaction
  - Each activity typically supports one focused thing a user can do
    - Viewing an email message
    - Showing a login screen
  - Applications often comprise several activities

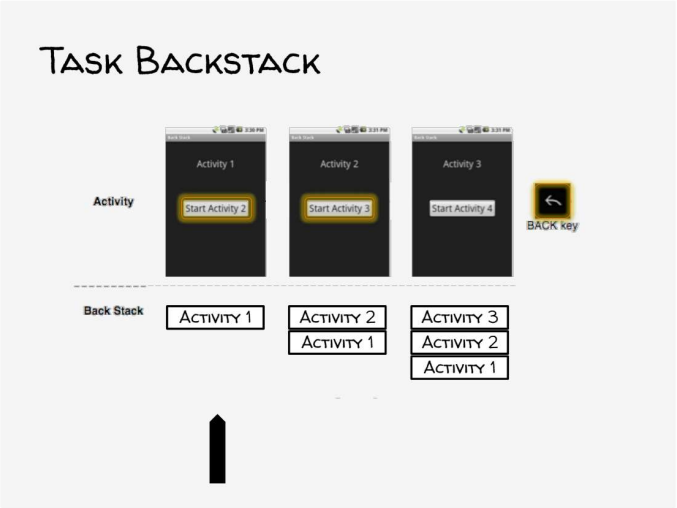
Building an application (4/8)

- Navigation through activities
  - Android supports navigation in several ways
    - Tasks
    - Task backstack
    - Suspending and resuming activities

Building an application (5/8)

- Tasks
  - A task is a set of related activities
  - These related activities don't have to be part of the same application
  - Most tasks start at the home screen
- Task backstack
  - When an activity is launched, it goes on top of the backstack
  - When the activity is destroyed, it is popped off the backstack

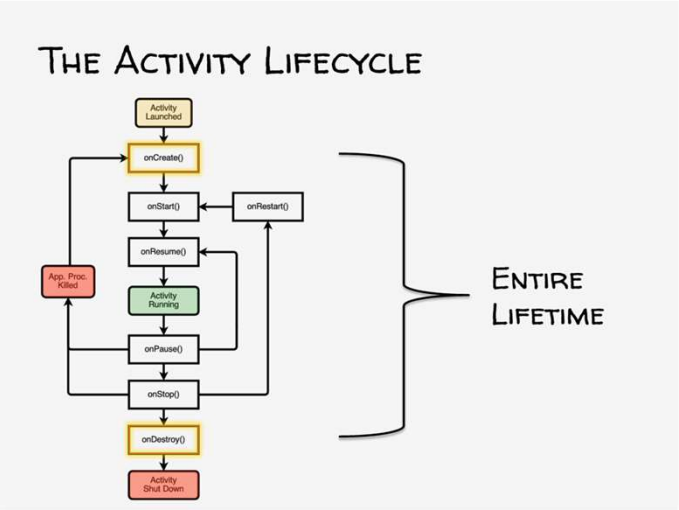
Building an application (6/8)



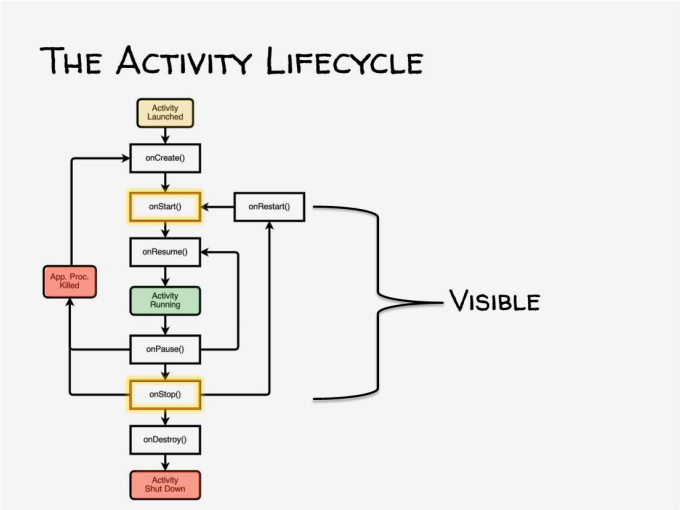
Building an application (7/8)

- The activity lifecycle
  - As necessary when an application executes
    - Created
    - Suspended
    - Resumed
    - Destroyed
  - Some of these actions depend on the user behavior
  - Some depend on android
    - E.g., Android can kill activities when it needs resources

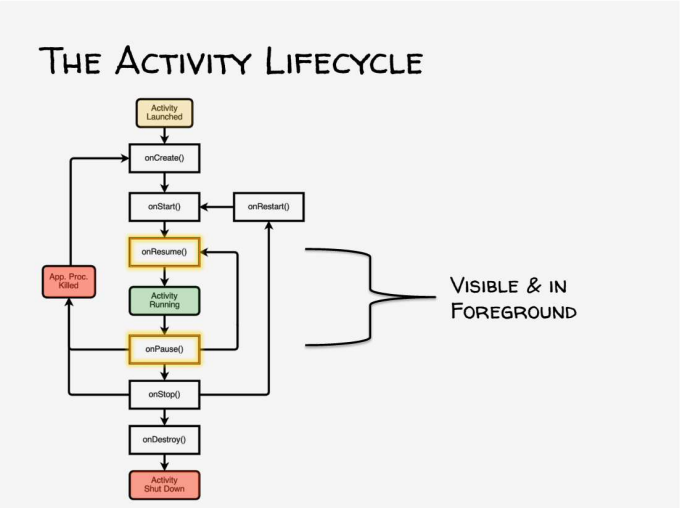
Building an application (8/8)



Building an application (8/8)



Building an application (8/8)



Credits

- Dr. Adam Porter, University of Maryland (USA). On-line course “Programming Mobile Applications for Android Handheld Systems” accessible at https://www.coursera.org/course/android
- Source code of the course available at https://github.com/aporter/coursera-android/tree/master/Examples
- Official website for android developement support http://developer.android.com/training/index.html