

LẬP TRÌNH ANDROID CƠ BẢN

NỘI DUNG

CHƯƠNG I: KHÁI QUÁT VỀ ANDROID	4
1.1 Giới thiệu về Hệ Điều Hành Android	4
1.2 Thiết lập môi trường lập trình Android	11
1.3 Các quy tắc lập trình trên Android (Android Coding Convention)	23
CHƯƠNG II: KIẾN TRÚC CƠ BẢN CỦA ANDROID	26
2.1 Kiến trúc Android	26
2.2 Những thành phần chính trong ứng dụng Android.....	27
2.3 Cấu trúc thư mục chính trong một dự án Android (Android project).....	29
CHƯƠNG III: XÂY DỰNG GIAO DIỆN NGƯỜI SỬ DỤNG.....	33
3.1 Activity.....	33
3.2 Xây dựng giao diện người dùng (User Interface Design).....	34
3.3 Menu trong ứng dụng Android	54
3.4 Xử lý sự kiện trong Android (Event Handling)	56
CHƯƠNG IV: INTENT, INTENT FILTER, SERVICE AND BROADCAST RECEIVER.....	58
4.1 Intent	58
4.2 Intent Filter.....	62
4.3 Service.....	65
4.4 Broadcast Receiver	71
CHƯƠNG V: LƯU TRỮ DỮ LIỆU	73
5.1 Đọc ghi dữ liệu trên File	73

5.2	Cơ sở dữ liệu trong Android	74
5.3	Content Provider	76
CHƯƠNG VI: MULTIMEDIA.....		79
6.1	Audio.....	79
6.2	Video	82
6.3	Camera	84
CHƯƠNG VII: WEB SERVICE VÀ WEBVIEW		86
7.1	Web Service	86
7.2	WebView	93

CHƯƠNG I: KHÁI QUÁT VỀ ANDROID

1.1 Giới thiệu về Hệ Điều Hành Android

1.1.1. Giới thiệu

Android là một Hệ Điều Hành được Google phát triển dành cho các thiết bị di động, dựa trên nền tảng Linux (Linux Kernel). Với giao diện người dùng (User Interface) dựa trên cơ chế thao tác trực tiếp (direct manipulation), Android được thiết kế chủ yếu là dành cho các thiết bị di động có màn hình cảm ứng (touchscreen mobile devices) như: điện thoại thông minh (smartphones) hay máy tính bảng (tablet computers), tivi màn hình cảm ứng (Android TV), và cả đồng hồ đeo tay (Android Wear).

Những thông tin chi tiết về Hệ Điều Hành Android:

- Nhà phát triển : Google, Open Handset Alliance
- Phát triển bằng ngôn ngữ : C (core), C++
- Mô hình mã nguồn : Mã nguồn mở (Open Source)
- Ngày ra mắt đầu tiên : 23/09/2008
- Ngày ra mắt mới nhất : 19/06/2014 (KitKat 4.4.4)
- Phiên bản thử nghiệm mới nhất : 17/10/2014 (5.0 Lollipop)
- Thị trường mục tiêu : Điện thoại thông minh, máy tính bảng
- Hỗ trợ ngôn ngữ : Đa ngôn ngữ (46 ngôn ngữ)
- Nền tảng hỗ trợ (Platforms) : 32-bit, 64-bit ARM, x86, x86-64
- Kiểu nhân (Kernel type) : Monolithic (modified Linux kernel)
- Bản quyền (License) : Apache License 2.0

1.1.2. Các phiên bản Android

Kể từ khi ra đời, Google đã tiến hành phát triển không ngừng và phát hành chính thức trên 20 phiên bản Android khác nhau. Trong số đó, các phiên bản đầu tiên hiện nay hầu như không được sử dụng, một số phiên bản được sử dụng rất hạn chế,

các nhà sản xuất thiết bị di động chủ yếu sử dụng những phiên bản mới nhất với những tính năng đầy đủ và tối ưu hơn.

Phiên bản	Ngày ra mắt	Tính năng
Android 1.0 (API 1)	23/09/2008	<ul style="list-style-type: none"> - Trình duyệt Web hỗ trợ hiển thị, phóng to, thu nhỏ, di chuyển các trang HTML và XHTML. - Camera, nhưng không có chức năng hiệu chỉnh - Truy cập mail server với chuẩn POP3, IMAP4, SMTP - Gmail, Google Contacts, Google Calendar, Google Maps, Google Search. - Media Player, YouTube video player - Wi-Fi, Bluetooth.
Android 1.1 (API 2)	09/02/2009	<ul style="list-style-type: none"> - Hiển thị thông tin chi tiết cho phép người dùng xem trước khi Search thông tin kinh doanh trên Maps. - Khả năng lưu trữ thông tin các Files đính kèm trong tin nhắn. - Thêm các hiệu ứng trong các Layout hệ thống.
Android 1.5 Cupcake (API 3)	27/04/2009	<ul style="list-style-type: none"> - Bàn phím ảo với chức năng đoán từ (Third-party virtual keyboards with text prediction) - Widgets - Video recording, playback (MPEG-4, 3GP) - Xoay màn hình (Auto-rotation) - Các hiệu ứng màn hình (animated screen transtions) - Upload video (YouTube) - Upload photos (Picasa)
Android 1.6 Donut (API 4)	15/09/2009	<ul style="list-style-type: none"> - Tìm kiếm bằng chữ hoặc giọng nói (voice and text entry search) - Đọc một chuỗi các chữ bằng chương trình phát âm đa

		ngôn ngữ (Multi-lingual speech synthesis engine) - Chọn và xóa nhiều hình ảnh đồng thời - Tăng tốc độ trong việc tìm kiếm và xử lý camera. - Text-to-speech engine
Android 2.0 Eclair (API 5)	26/10/2009	- Microsoft Exchange Email - Bluetooth 2.1 - Tăng tốc độ xử lý bàn phím ảo - HTML5 - Tối ưu hóa tốc độ phần cứng - Hỗ trợ nhiều màn hình với kích thước và độ phân giải khác nhau. - Nâng cấp Google Maps 3.1.2 - MotionEvent (Muti-touch events)
Android 2.0.1 Eclair (API 6)	03/12/2009	- Thay đổi một số API - Sửa lỗi phiên bản trước
Android 2.1 Eclair (API 7)	12/01/2010	- Thay đổi một số API - Sửa lỗi phiên bản trước
Android 2.2 – 2.2.3 Froyo (API 8)	20/05/2010	- Tối ưu tốc độ xử lý, vùng nhớ - Tăng tốc độ chạy ứng dụng thông qua cơ chế JIT (Just In Time compilation) - Tích hợp chương trình xử lý JavaScript V8 của Chrome vào ứng dụng Browser (V8 JavaScript engine) - Android Cloud

		<ul style="list-style-type: none"> - USB, Wi-Fi hotspot - Adobe Flash - Cài đặt ứng dụng vào bộ nhớ mở rộng
Android 2.3 – 2.3.2 Gingerbread (API 9)	06/12/2010	<ul style="list-style-type: none"> - Thêm một số thiết kế giao diện người dùng mới đơn giản và nhanh hơn. - Hỗ trợ màn hình với kích thước và độ phân giải cao (WXGA and higher) - SIP VoIP - Đọc dữ liệu từ thẻ NFC (Near Field Communication – read only mode) - Download Manager - Power Management - GC (Garbage Collection)
Android 2.3.3 – 2.3.7 Gingerbread (API 10)	09/02/2011	<ul style="list-style-type: none"> - Sử dụng Google Talk hỗ trợ Voice/Video chat - Sửa lỗi phiên bản trước: Bluetooth, voice search...
Android 3.0 Honeycomb (API 11)	22/02/2011	<ul style="list-style-type: none"> - Hỗ trợ giao diện người dùng trên máy tính bảng - System bar - Action bar - Đa nhiệm (Simplified multitasking) - Mở nhiều tab trên Brower - Sử dụng camera nhanh hơn, nhiều chức năng hơn - Hiển thị Album ở chế độ toàn màn hình (full-screen mode) - Đa xử lý (Multi-core processors)

		- Cảm biến gia tốc (Hardware acceleration)
Android 3.0 Honeycomb (API 12)	10/05/2011	<ul style="list-style-type: none"> - Thiết kế giao diện đẹp hơn - Kết nối với các phụ kiện USB - Mở rộng danh sách các ứng dụng đã hoạt động (Recent Application) - Thay đổi kích thước màn hình Home - FLAC audio playback - Kết nối Wi-Fi tốc độ cao
Android 3.2 Honeycombe (API 13)	15/07/2011	<ul style="list-style-type: none"> - Nâng cấp phần cứng, tối ưu cho nhiều dòng máy tính bảng - Có thể truy cập File trong Sdcard - Hỗ trợ chức năng hiển thị, tương thích giữa các thiết bị Android khác nhau. - Sửa các lỗi phiên bản trước.
Android 4.0 – 4.0.2 Ice Cream Sandwich (API 14)	18/10/2011	<ul style="list-style-type: none"> - Hỗ trợ giao diện đẹp hơn (Holo interface with new Roboto font family) - Các phím ảo được chính thức sử dụng cho điện thoại - Chức năng chụp màn hình (screenshot capture) - Có thể truy cập ứng dụng từ màn hình khóa (lock-screen) - Android Beam (NFC) - Wi-Fi Direct
Android 4.0.3 – 4.0.4 Ice Cream Sandwich (API 15)	16/12/2011	<ul style="list-style-type: none"> - Sửa lỗi và tối ưu hóa hệ thống - Nâng cấp đồ họa, cơ sở dữ liệu, Bluetooth - API mới - Nâng cao tính ổn định

Android 4.1 Jelly Bean (API 16)	09/07/2012	<ul style="list-style-type: none"> - Giao diện người dùng mượt hơn, đẹp hơn - Truyền dữ liệu qua Bluetooth theo cơ chế Android Beam. - Tự động sắp xếp, thay đổi kích thước các Shortcuts, Widget - Nâng cấp ứng dụng Camera
Android 4.2 Jelly Bean (API 17)	13/11/2012	<ul style="list-style-type: none"> - Nâng cấp chức năng khóa màn hình, có thể sử dụng Camera trực tiếp khi khóa màn hình - Tạo nhiều tài khoản người sử dụng trên máy tính bảng (Multiple user accounts) - Thay đổi Bluetooth Stack (BlueZ -> BlueDroid) - Thiết kế giao diện Layout cho tất cả các thiết bị
Android 4.3 Jelly Bean (API 18)	24/07/2013	<ul style="list-style-type: none"> - BLE (Bluetooth Low Energy) - AVRCP 1.3 (Bluetooth Audio/Video Remote Control Profile) - OpenGL ES 3.0
Android 4.4 KitKat (API 19)	31/10/2013	<ul style="list-style-type: none"> - Làm mới giao diện với các thành phần màu trắng thay cho màu xanh da trời - Chức năng kết nối máy in không dây - NFC host card emulation - Nâng cấp WebView (Chromium engine) - ART (Android Runtime) - MAP (Bluetooth Message Access Profile)
Android 4.4w KitKat (API 20)	Chưa chính thức	<ul style="list-style-type: none"> - Hỗ trợ cho thiết bị đeo tay (Android Wear)

Android 5.0 Lollipop (API 21)	Chưa chính thức	<ul style="list-style-type: none"> - ART (Android Runtime) với cơ chế AOT (ahead-of-time compilation) được chính thức sử dụng thay thế cho Dalvik với cơ chế JIT (just-in-time compilation) - Hỗ trợ 64-bit CPUs - OpenGL ES 3.1 - Hỗ trợ chức năng xem trước khi in (print previews)

Thống kê cho việc sử dụng các phiên bản Android trên thị trường (tính đến ngày 09/09/2014)

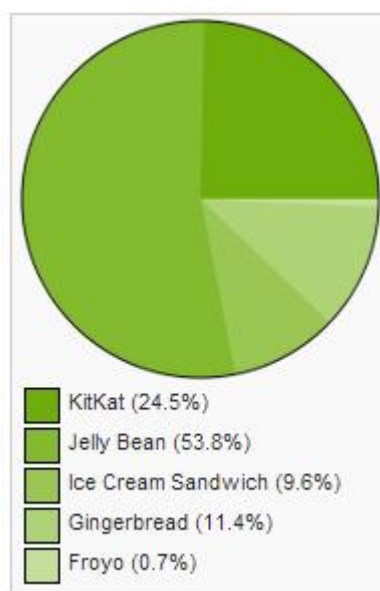


Figure 1: Thống kê sử dụng các phiên bản Android

1.2 Thiết lập môi trường lập trình Android

1.2.1. Eclipse

1.2.1.1. Giới thiệu

Eclipse là một ứng dụng được thiết kế để xây dựng các môi trường phát triển tích hợp (IDE – Integrated Development Environment). Trong lập trình Android, Eclipse được sử dụng như là một công cụ phổ biến nhất để phát triển các ứng dụng Android.

1.2.1.2. Cài đặt

- Download Eclipse ADT mới nhất từ website:

<https://developer.android.com/sdk/index.html?hl=i>

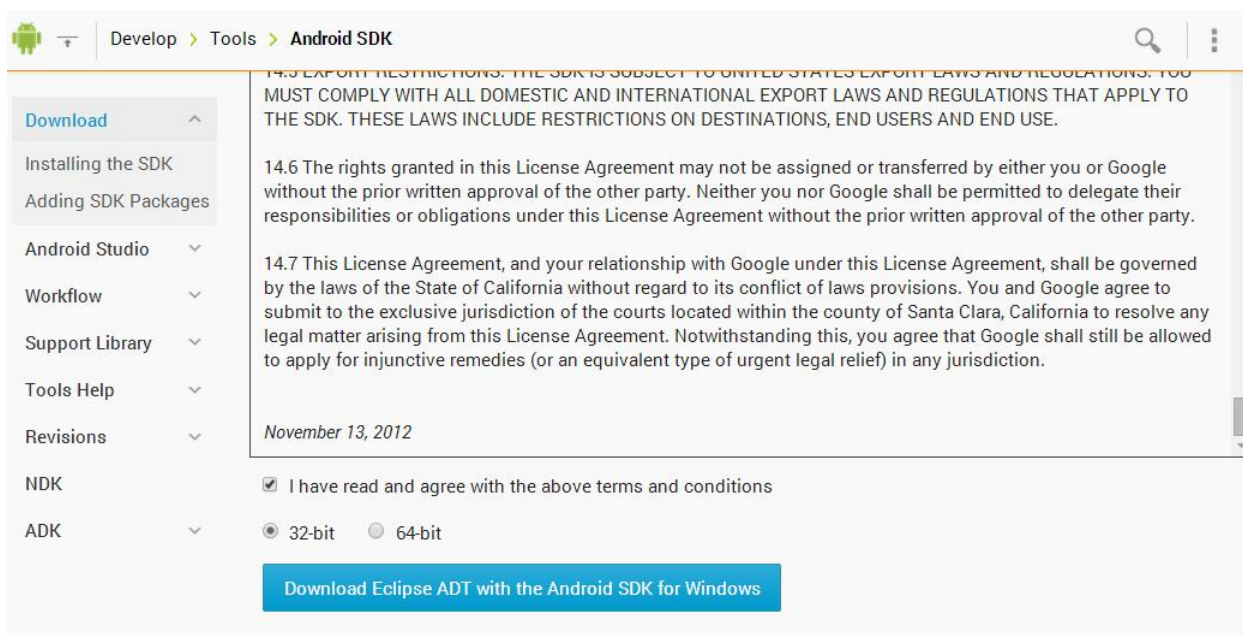


Figure 2: Download Eclipse ADT

- o Giải nén ZIP file vừa download được (adt-bundle-windows-x86-20140702.zip)
- o Vào thư mục [\\adt-bundle-windows-x86-20140702\\eclipse\\eclipse.exe](#) để khởi động chương trình Eclipse
- o Chọn Workspace để lưu các dự án trong Eclipse:

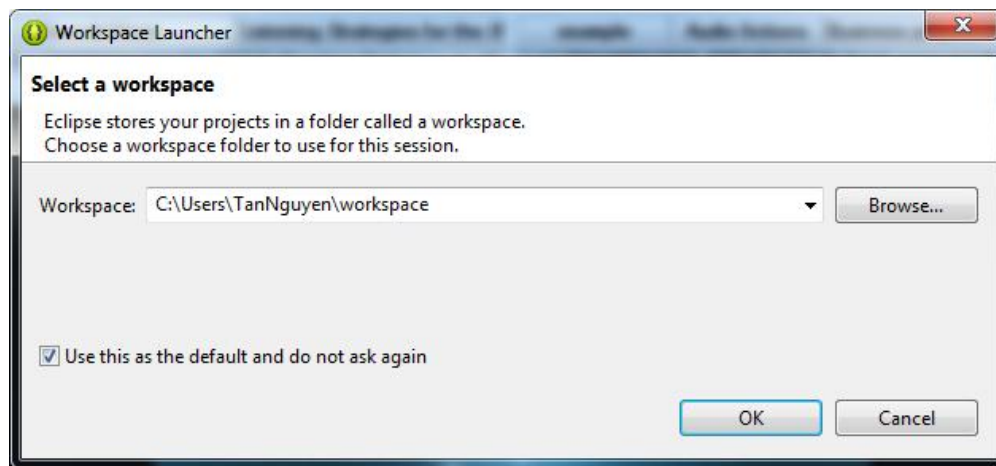


Figure 3: Chọn Workspace

- Màn hình chính Eclipse:

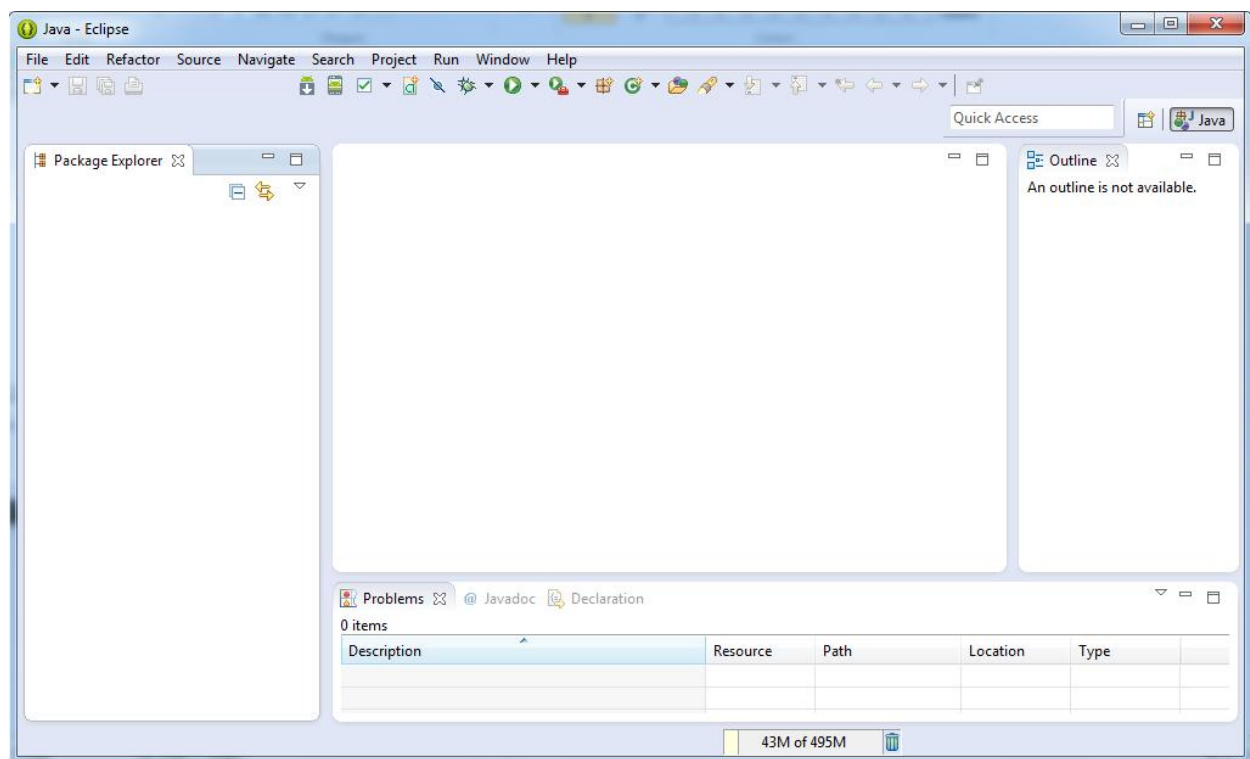


Figure 4: Màn hình chính Eclipse

- Tạo dự án trên Eclipse: vào File/New/Project

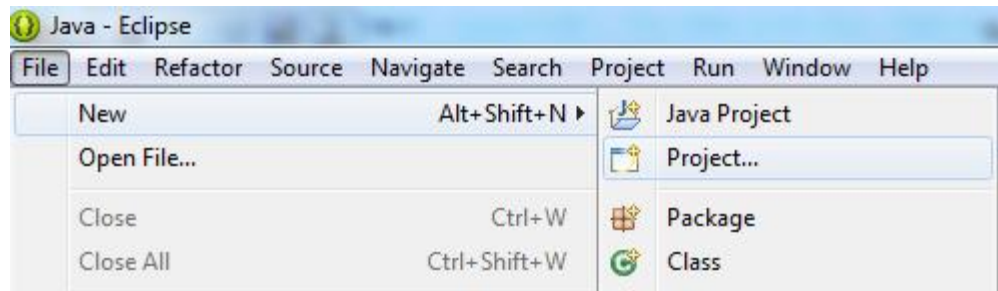


Figure 5: Tạo dự án mới trong Eclipse

- Chọn dự án Android/Android Application Project

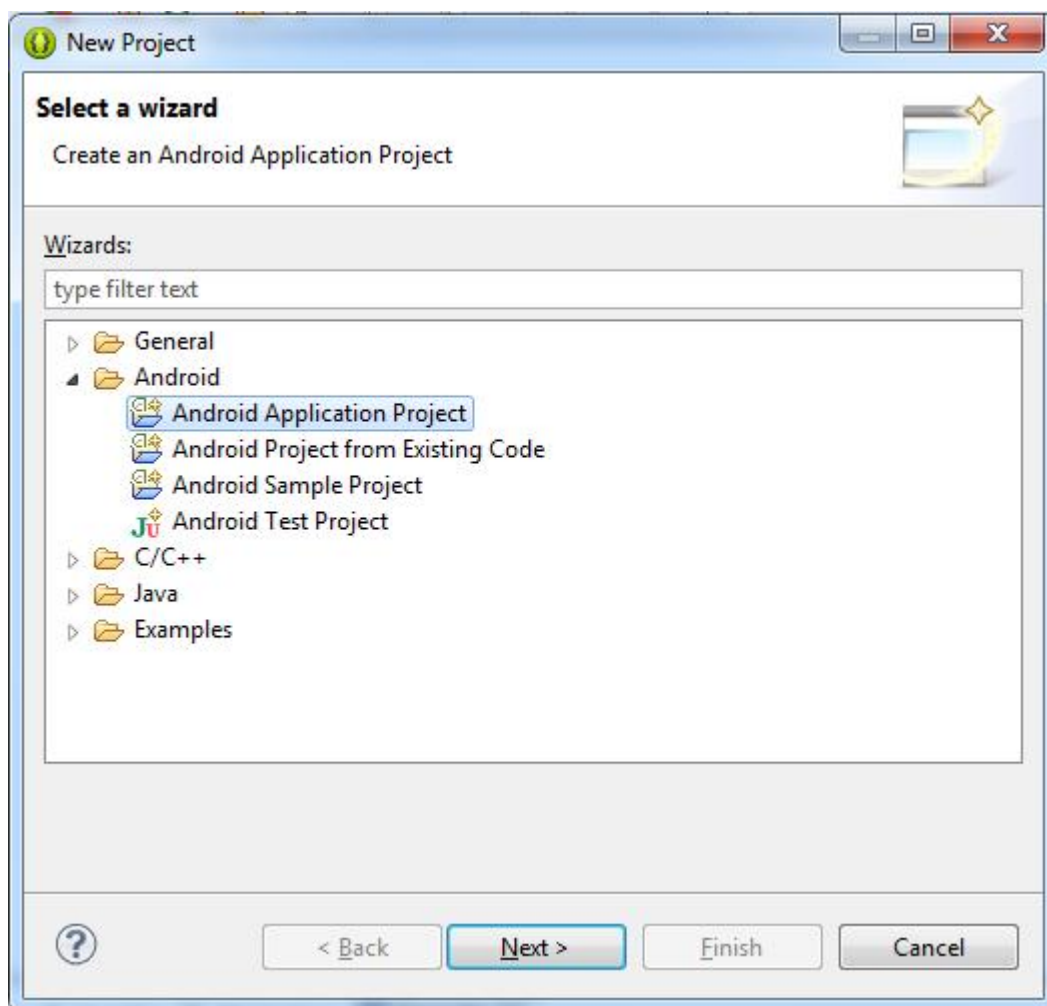


Figure 6: Chọn dự án Android (Android Application Project)

- Đặt tên và các thông số cơ bản cho dự án mới

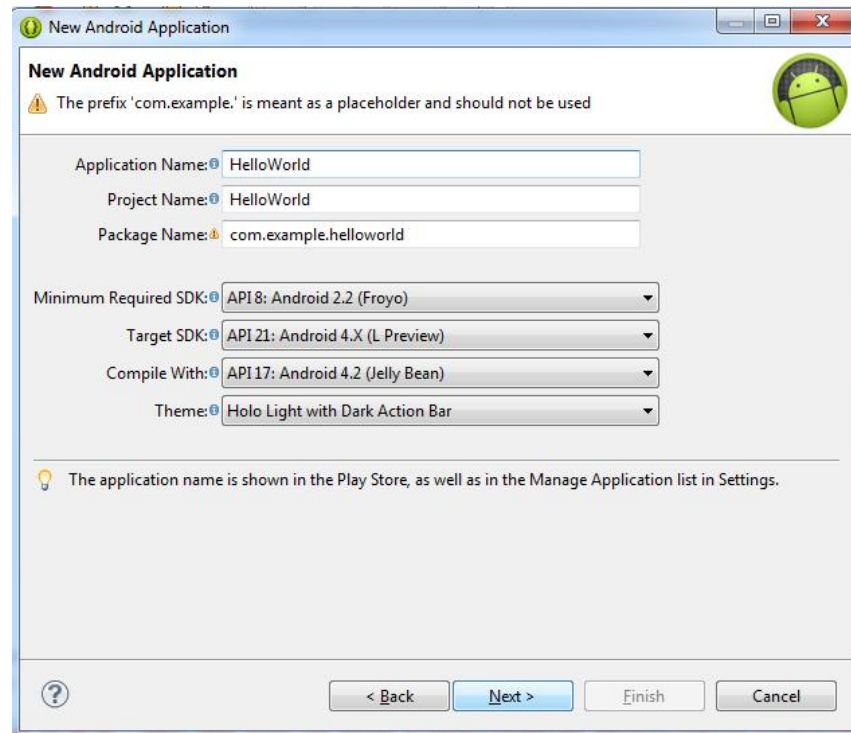


Figure 7: Đặt tên và các thông số cơ bản

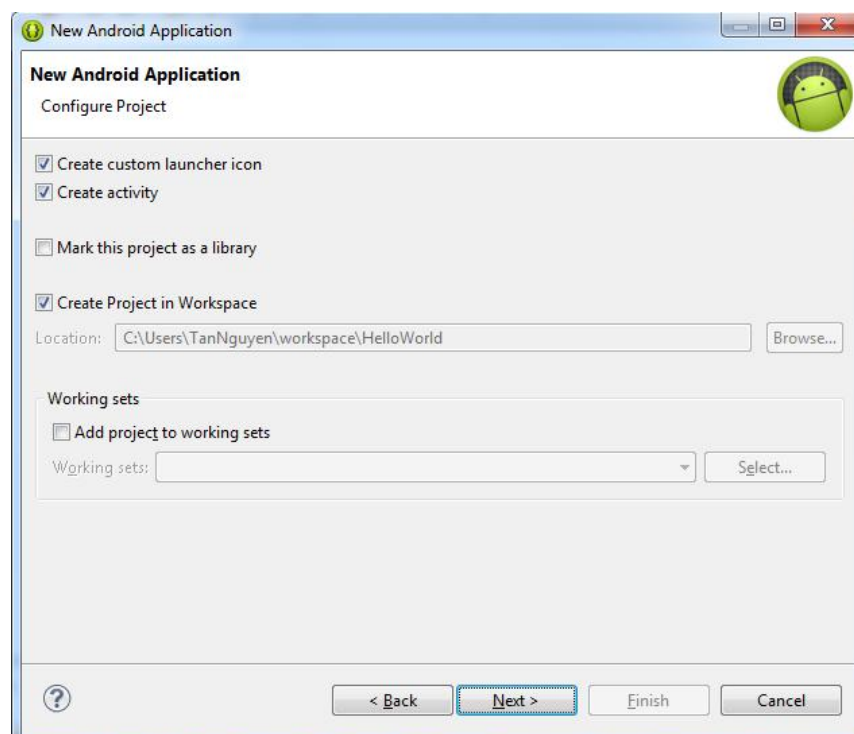


Figure 8: Thiết lập đường dẫn cho dự án

- Thiết lập các thuộc tính icon cho dự án

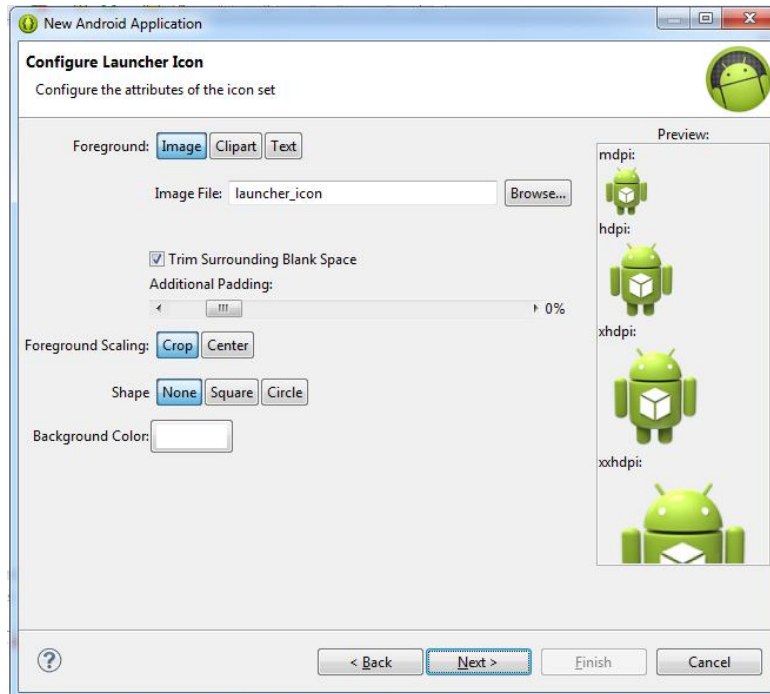


Figure 9: Thiết lập các thuộc tính icon cho dự án

- Chọn kiểu Activity cho dự án:

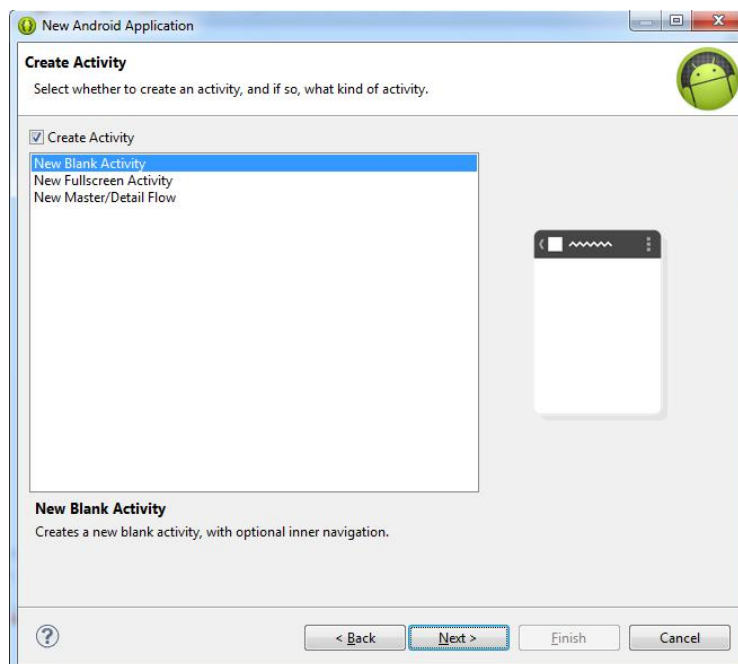


Figure 10: Chọn kiểu Activity

- Đặt tên cho Activity vừa tạo:

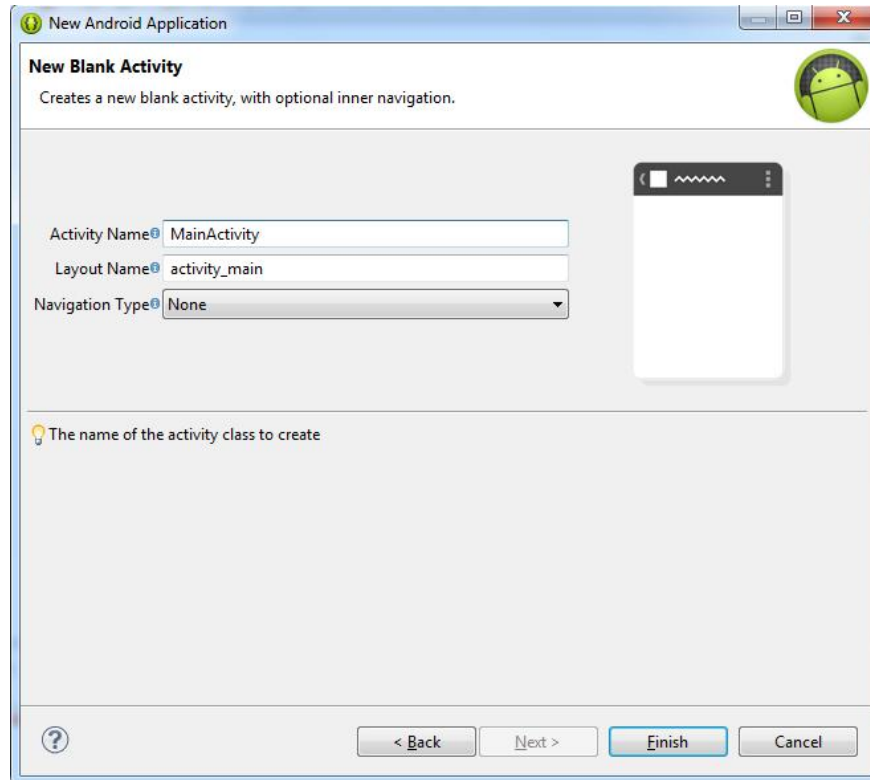


Figure 11: Đặt tên cho Activity

- Kết thúc quá trình tạo dự án

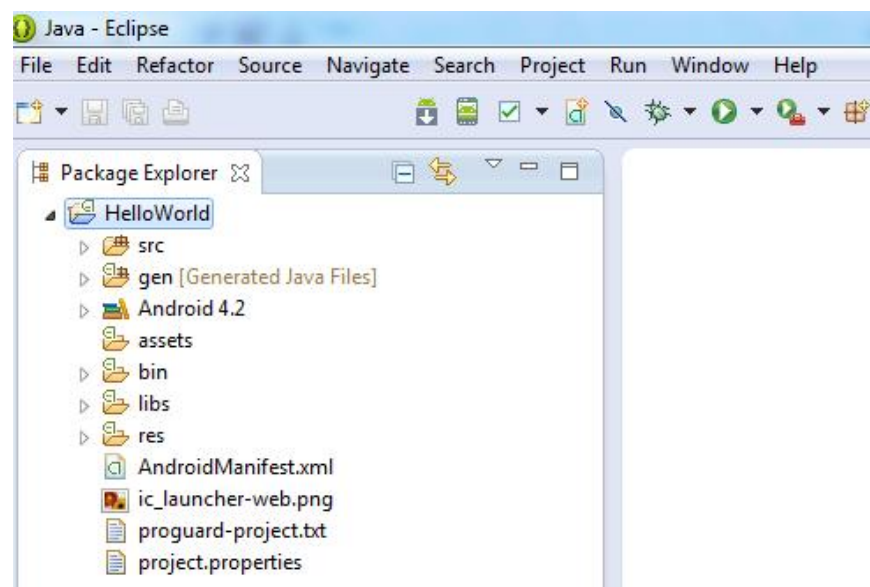


Figure 12: Dự án mới được khởi tạo

1.2.2. Android SDK (Software Development Kit)

1.2.2.1. Giới thiệu

Android SDK cung cấp các thư viện và công cụ dùng để xây dựng, phát triển các ứng dụng trên nền tảng Hệ Điều Hành Android.

1.2.2.2. Cài đặt

- Đối với Eclipse ADT, Android SDK đã được tích hợp sẵn
- Đối với những phiên bản Eclipse cũ hơn, download Android SDK từ Website:

<http://developer.android.com/sdk/index.html/>

- Tích hợp Android SDK vào Eclipse:
 - o Vào Window -> Preferences:

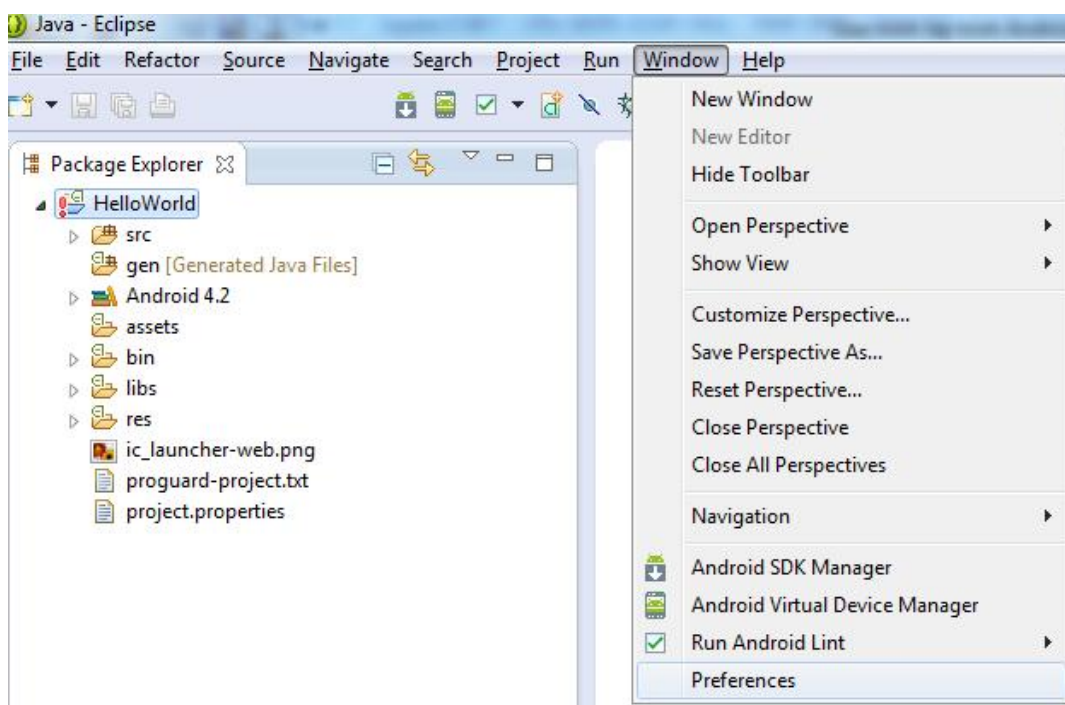


Figure 13: Eclipse Preferences

- o Chọn tab Android
- o Click vào Browser để chọn đường dẫn đến thư mục chứa SDK đã được Download trước đó

- Click Apply để xác lập Android SDK vừa chọn

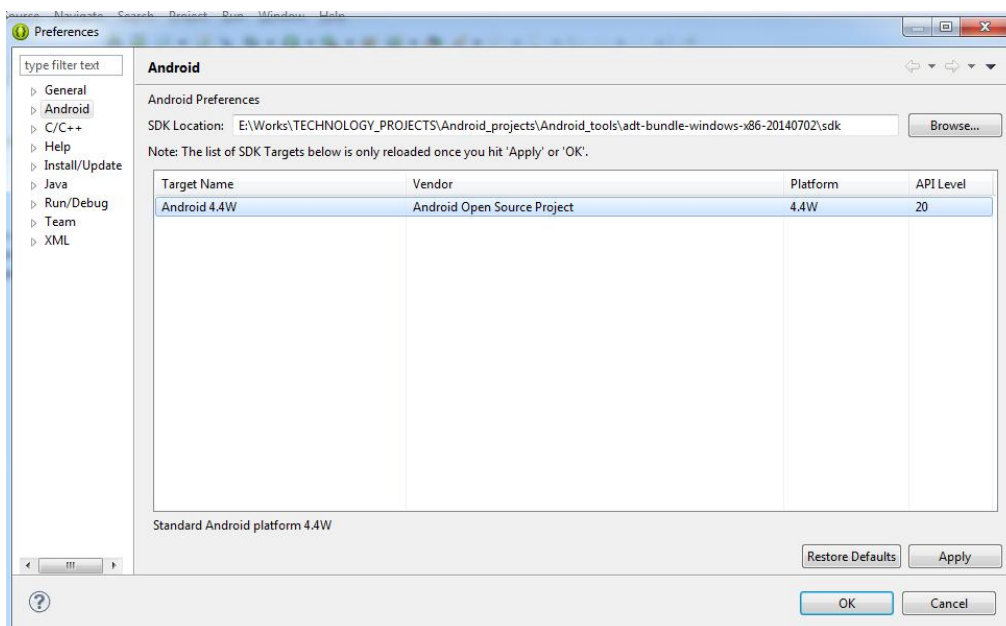


Figure 14: Chọn đường dẫn đến Android SDK

- Để cài đặt thêm SDK cho từng phiên bản Android, vào Eclipse, chọn Window -> Android SDK Manager :

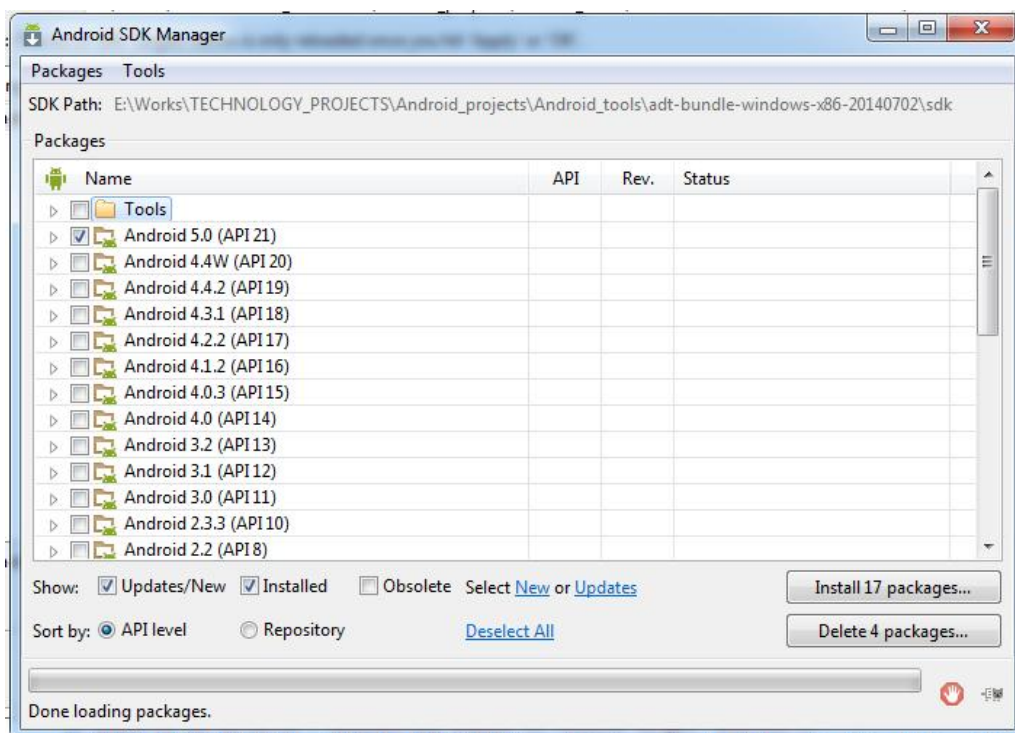


Figure 15: Cài đặt Android SDK

1.2.3. Android Emulator

1.2.3.1. Giới thiệu

Android Emulator là một ứng dụng mô phỏng các thiết bị Android (hay được gọi là máy ảo) được sử dụng để chạy các ứng dụng Android ngay trên máy tính Windows (Linux hay MAC).

1.2.3.2. Cài đặt

- Vào Eclipse, chọn Window -> Android Virtual Device Manager, chọn Create để tạo máy ảo mới hoặc Edit (nếu muốn thay đổi các tính năng của máy ảo đã được tạo ra trước đó)

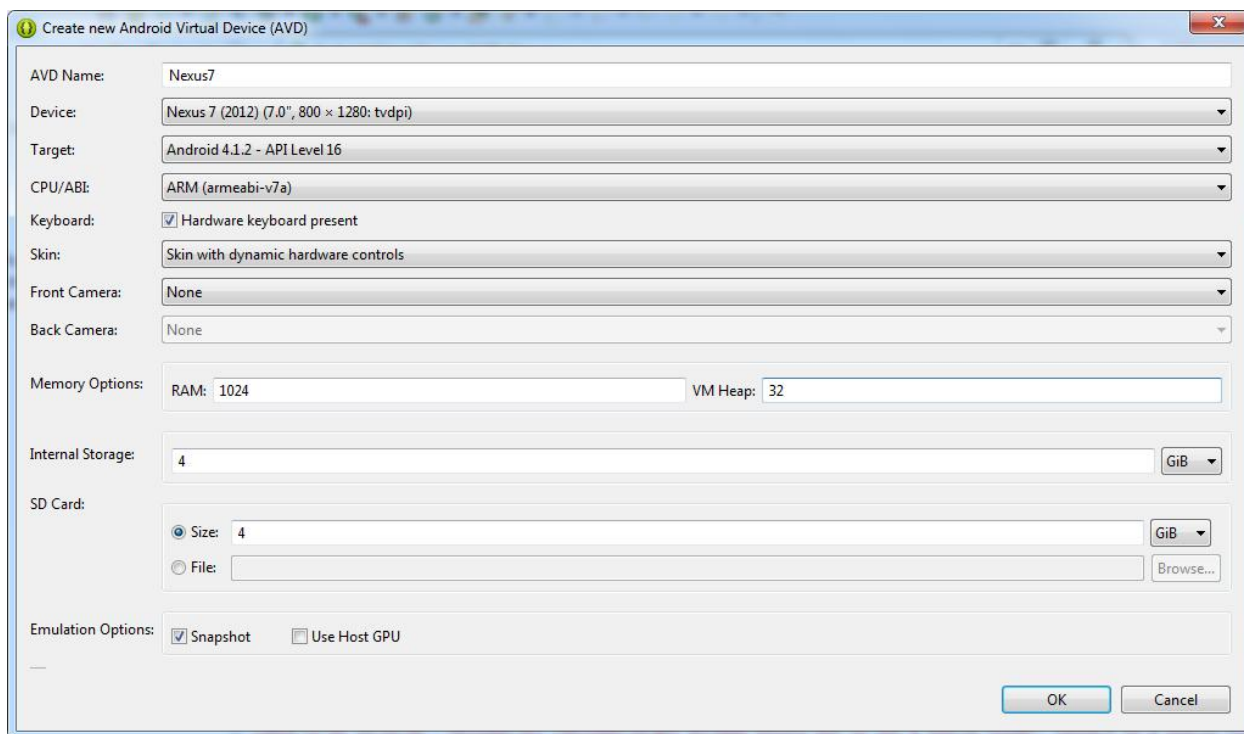


Figure 16: Tạo máy ảo Nexus 7

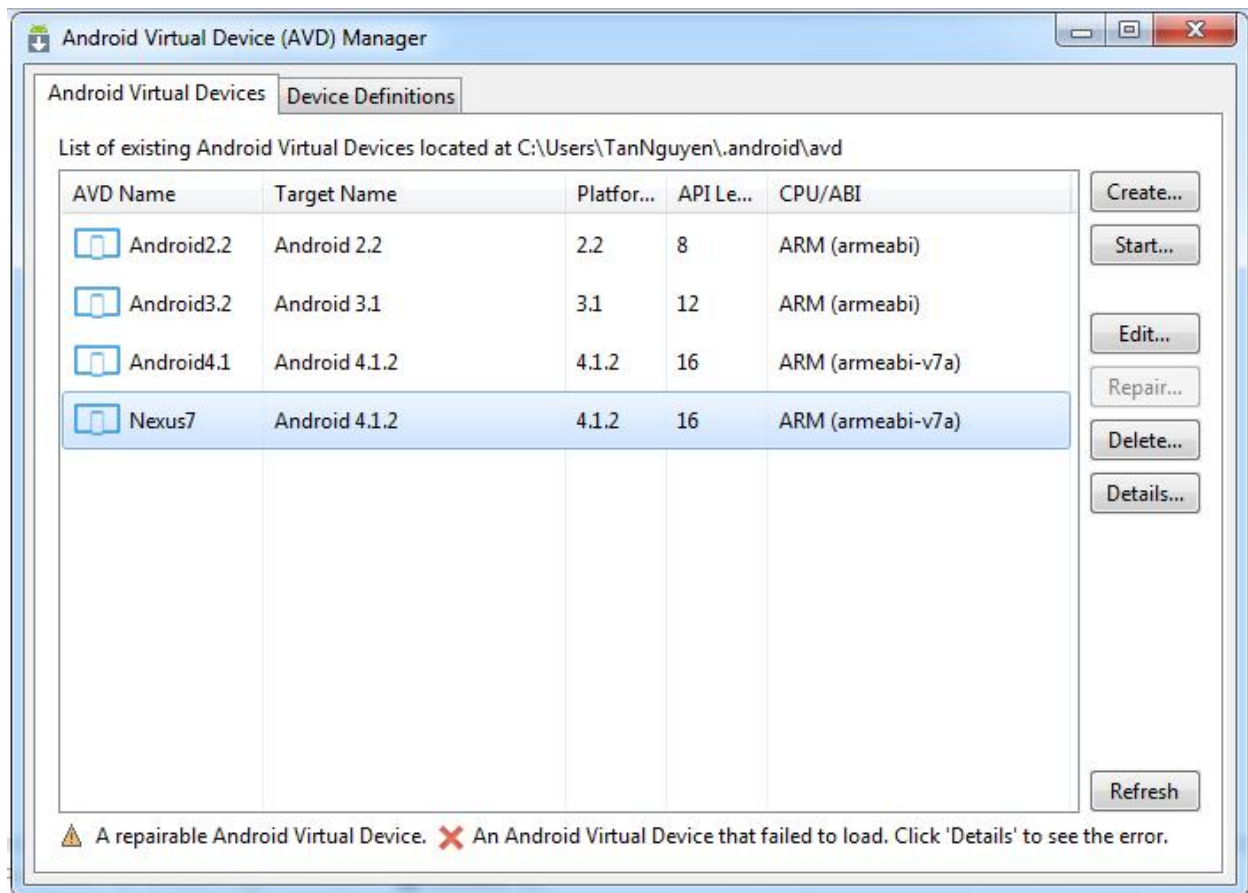


Figure 17: Danh sách các máy ảo được tạo

1.2.4. Davik Debug Monitor Service (DDMS)

- DDMS là công cụ dùng để Debug trên Android, bao gồm các chức năng: chụp màn hình, hiển thị các thông tin liên quan đến Thread, Heap, logcat, cửa sổ File Explorer...
- DDMS được tích hợp trong Eclipse, để hiển thị DDMS trên Eclipse, vào Window -> Open Perspective -> Other... chọn DDMS:

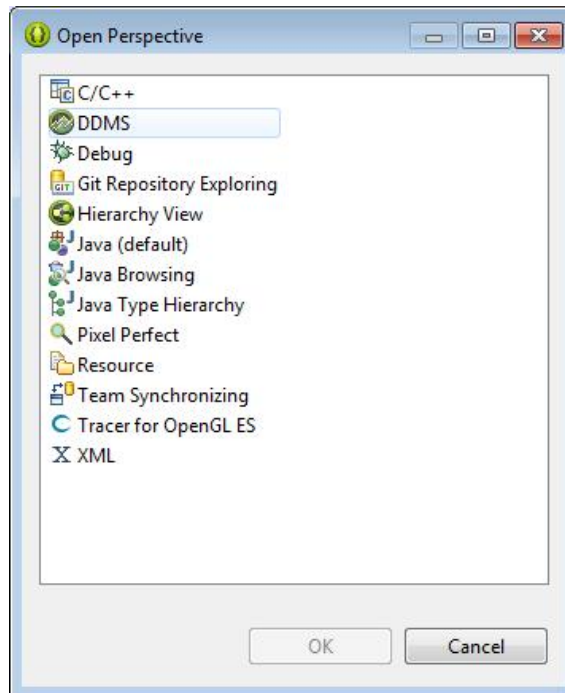


Figure 18: Màn hình Open Perspective

- Cửa sổ DDMS

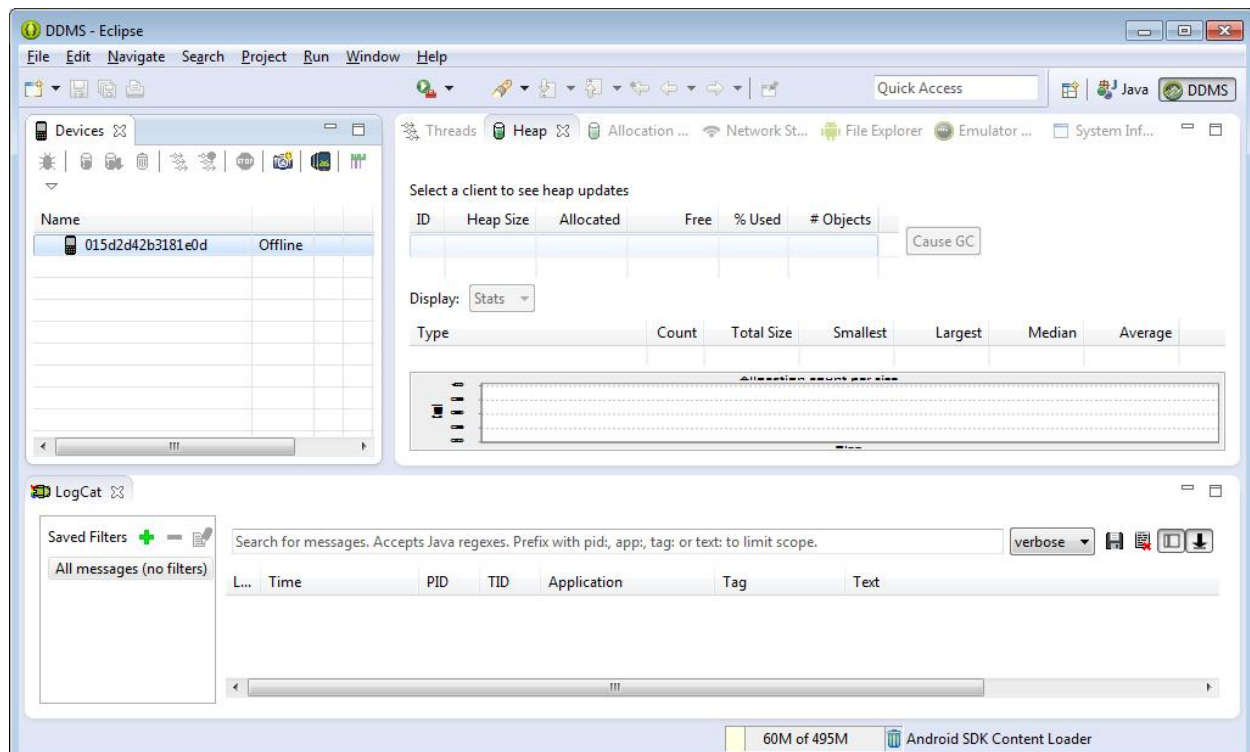


Figure 19: Màn hình DDMS

1.2.5. Android Debug Bridget (ADB)

- ADB là công cụ được sử dụng để giao tiếp với máy ảo (AVD) hoặc là thiết bị Android thông qua các câu lệnh (command line)

```

C:\Windows\system32\cmd.exe

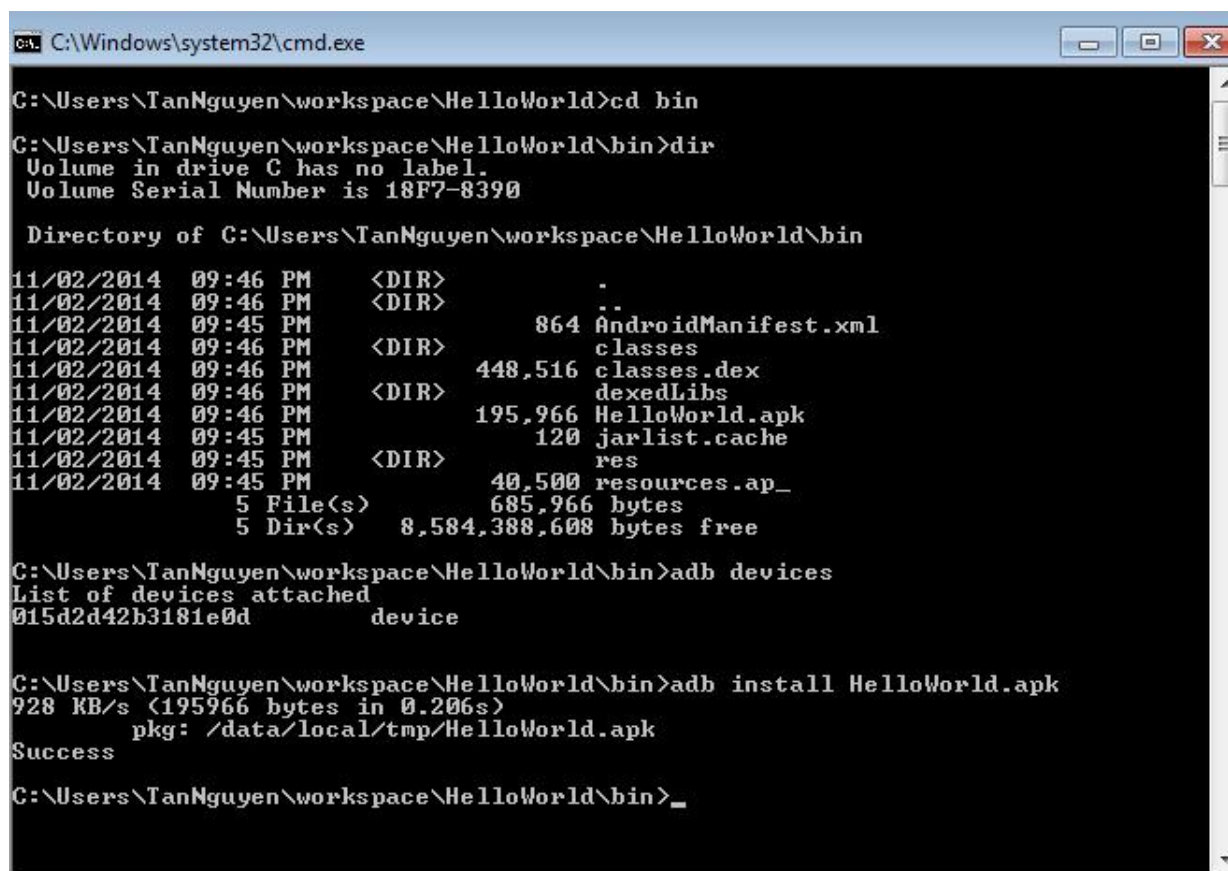
C:\Users\TanNguyen>adb
Android Debug Bridge version 1.0.31

  -d                - directs command to the only connected USB device
                    returns an error if more than one USB device is
                    present.
  -e                - directs command to the only running emulator.
                    returns an error if more than one emulator is r
                    unning.
  -s <specific device> - directs command to the device or emulator with
                    the given
                    SERIAL
                    environment variable.
  -p <product name or path> - simple product name like 'sooner', or
                    a relative/absolute path to a product
                    out directory like 'out/target/product/sooner'.
                    If -p is not specified, the ANDROID_PRODUCT_OUT
                    environment variable is used, which must
                    be an absolute path.
  devices [-l]      - list all connected devices
                    <'l' will also list device qualifiers>
  connect <host>[:<port>] - connect to a device via TCP/IP
                    Port 5555 is used by default if no port number
                    is specified.
  disconnect [<host>[:<port>]] - disconnect from a TCP/IP device.
                    Port 5555 is used by default if no port number
                    is specified.
                    Using this command with no additional arguments
                    will disconnect from all connected TCP/IP device
                    es.

device commands:
  adb push <local> <remote> - copy file/dir to device
  adb pull <remote> [<local>] - copy file/dir from device
  adb sync [ <directory> ] - copy host->device only if changed
                           <-l means list but don't copy>
                           <see 'adb help all'>
  adb shell                - run remote shell interactively
  adb shell <command>      - run remote shell command
  adb emu <command>        - run emulator console command
  adb logcat [ <filter-spec> ] - View device log
  adb forward <local> <remote> - forward socket connections
                               forward specs are one of:
                               tcp:<port>
                               localabstract:<unix domain socket name>
                               localreserved:<unix domain socket name>
                               localfilesystem:<unix domain socket name>
                               dev:<character device name>
                               jdwp:<process pid> <remote only>
  adb jdwp                 - list PIDs of processes hosting a JDWP transport
  
```

Figure 20: Các cấu trúc lệnh của ADB

- Có thể sử dụng ADB để kiểm tra sự kết nối, cài đặt ứng dụng, xem logcat...trên máy ảo hoặc thiết bị thực.



```
C:\Windows\system32\cmd.exe

C:\Users\TanNguyen\workspace\HelloWorld>cd bin
C:\Users\TanNguyen\workspace\HelloWorld\bin>dir
Volume in drive C has no label.
Volume Serial Number is 18F7-8390

Directory of C:\Users\TanNguyen\workspace\HelloWorld\bin

11/02/2014  09:46 PM    <DIR>          .
11/02/2014  09:46 PM    <DIR>          ..
11/02/2014  09:45 PM             864  AndroidManifest.xml
11/02/2014  09:46 PM    <DIR>          classes
11/02/2014  09:46 PM    448,516  classes.dex
11/02/2014  09:46 PM    <DIR>          dexedLibs
11/02/2014  09:46 PM    195,966  HelloWorld.apk
11/02/2014  09:45 PM    120     jarlist.cache
11/02/2014  09:45 PM    <DIR>          res
11/02/2014  09:45 PM    40,500  resources.ap_
               5 File(s)          685,966 bytes
               5 Dir(s)      8,584,388,608 bytes free

C:\Users\TanNguyen\workspace\HelloWorld\bin>adb devices
List of devices attached
015d2d42b3181e0d    device

C:\Users\TanNguyen\workspace\HelloWorld\bin>adb install HelloWorld.apk
928 KB/s (195966 bytes in 0.206s)
  pkg: /data/local/tmp/HelloWorld.apk
Success

C:\Users\TanNguyen\workspace\HelloWorld\bin>_
```

Figure 21: Kiểm tra kết nối, cài đặt ứng dụng trên thiết bị Android

1.3 Các quy tắc lập trình trên Android (Android Coding Convention)

1.3.1. Giới thiệu

- Các quy tắc được đặt ra nhằm mục đích chuẩn hóa các thao tác khi lập trình, để có thể tạo ra những sản phẩm phần mềm chất lượng, dễ quản lý, bảo trì, nâng cấp...
- Đối với mỗi ngôn ngữ lập trình đều có những quy tắc riêng tùy theo đặc điểm từng ngôn ngữ. Tuy nhiên, tất cả các quy tắc đó đều tương đồng với nhau nhằm hướng tới sự nhất quán, rõ ràng và nâng cao sự hiệu quả trong việc lập trình.

1.3.2. Quy tắc lập trình trên Java/Android

- **package và import:** package thường được đặt đầu tiên trong file Java, theo sau đó là import. Trong một file Java có thể có nhiều phát biểu

import tùy theo các thư viện được sử dụng, thứ tự các phát biểu import phải được sắp xếp như sau:

- Android import
- Các import khác (third-party: org, com, junit, net)
- Java và javax
- Các phát biểu package và import phải được viết bằng chữ in thường

```
package com.example.helloworld;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import java.util.*;
```

- **Số lượng ký tự trên 1 dòng code (Number per line):** thường quy định trong khoảng 80 ~ 100 ký tự. Không nên để code quá dài, dễ gây rối và rất khó bảo trì.
- **Khai báo phương thức:**
 - Không để khoảng trống (space) dựa tên phương thức (method) và dấu ngoặc mở “{”.
 - Tên phương thức nên là một động từ bắt đầu bằng một từ in thường.
 - Dấu ngoặc nhọn mở “{” phải đặt cùng dòng với tên phương thức, và cách 1 khoảng trống (trừ trường hợp phương thức null)
 - Dấu ngoặc nhọn đóng “}” phải đặt cuối phương thức, trên một dòng mới

```
class Sample extends Object {
    int mInt;

    Sample(int i) {
        mInt = i;
    }

    int emptyMethod() {}
    ...
}
```


- Đặt tên biến:

- Không phải biến public, static thì tên biến bắt đầu bằng ký tự “m”
- Biến static bắt đầu bằng ký tự “s”
- Các biến khác bắt đầu bằng một ký tự thường
- Các hằng số phải được khai báo bằng các ký tự in hoa

```
public class MyClass {  
    public static final int SOME_CONSTANT = 42;  
    public int publicField;  
    private static MyClass singleton;  
    int mPackagePrivate;  
    private int mPrivate;  
    protected int mProtected;  
}
```

- Đặt tên class:

- Nên đặt tên class là một danh từ
- Chữ cái đầu tiên phải viết hoa

```
public class MyClass {}  
private class YourClass {}
```

- Đặt tên Interface:

- Tương tự như class, nhưng chữ cái đầu tiên luôn luôn là ký tự “I”

```
public interface IListener {}  
private interface IDelegate {}
```

CHƯƠNG II: KIẾN TRÚC CƠ BẢN CỦA ANDROID

2.1 Kiến trúc Android

Hệ Điều Hành Android gồm có 5 phần chính, được thiết kế theo dạng 4 lớp (4 layers)

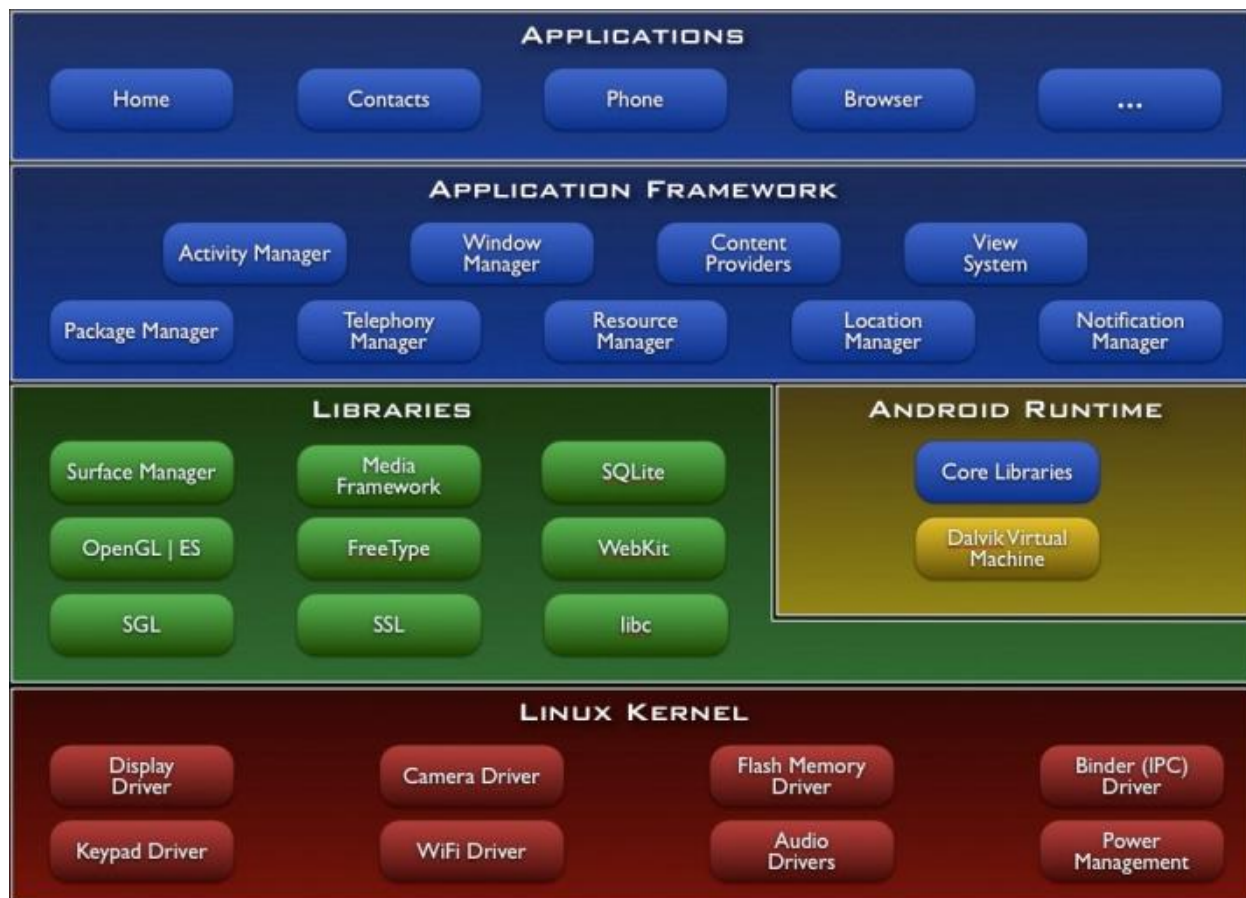


Figure 22: Kiến trúc Android

- **Linux Kernel:** là lớp thấp nhất trong mô hình kiến trúc Android, cung cấp những chức năng hệ thống cơ bản như: quản lý tiến trình (process management), quản lý bộ nhớ (memory management), quản lý các thiết bị tích hợp (camera, keypad, display)...
- **Libraries:** nằm phía trên Linux Kernel, là lớp tập hợp tất cả các thư viện liên quan đến xử lý web (Webkit), đồ họa (OpenGL ES), media và cơ sở dữ liệu (SQLite)...
- **Android Runtime:** là bộ phận hết sức quan trọng trong kiến trúc Android, có chứa máy ảo có tên gọi là Davik (Davik Virtual Machine) được phát triển, tối ưu hóa dựa trên máy ảo Java (Java Virtual Machine)

để sử dụng cho Hệ Điều Hành Android. Cùng với Davik Virtual Machine và những thư viện chính (Core Libraries), Android Runtime cho phép chạy tất cả các ứng dụng viết dưới ngôn ngữ Java.

- **Application Framework:** cung cấp các dịch vụ cho ứng dụng bên ngoài dưới dạng các lớp Java
 - **View:** được sử dụng để tạo ra các giao diện cho người sử dụng
 - **NotificationManager:** quản lý các thông báo từ các ứng dụng đến người sử dụng thông qua các tin nhắn trên thanh trạng thái (status bar)
 - **ActivityManager:** quản lý vòng đời của các Activity
- **Application:** là lớp cao nhất quản lý tất cả các ứng dụng Android (phone, web browser, SMS, Map, Calendar...)

2.2 Những thành phần chính trong ứng dụng Android

2.2.1 Activity

- Một Activity đại diện cho một màn hình đơn trong Android với một giao diện người sử dụng (user interface).
- Ví dụ: ứng dụng gửi tin nhắn sẽ có một Activity hiển thị các danh sách tin nhắn đến, một Activity khác hiển thị danh sách tin nhắn đi và một Activity dùng để đọc tin nhắn.
- Muốn hiện thực một Activity trong Android chỉ cần khai báo một lớp kế thừa từ lớp Activity:

```
public class MainActivity extends Activity {  
  
}
```

2.2.2 Service

- Một Service là một thành phần quan trọng trong Android được sử dụng trong những tác vụ cần chạy ngầm (background) mà không cần phải có giao diện như Activity, cũng như không cần tương tác với người sử dụng.

- Ví dụ: một Service có thể chơi nhạc trong khi người sử dụng đang thao tác trên một ứng dụng khác. Hoặc, một Service cũng có thể được sử dụng để Download dữ liệu từ Internet mà không ảnh hưởng đến Activity khác đang tương tác với người sử dụng.
- Muốn hiện thực một Service trong Android chỉ cần khai báo một lớp kế thừa từ lớp Service:

```
public class MyService extends Service {  
  
}
```

2.2.3 Content Provider

- Là một trong những thành phần chính trong ứng dụng Android, Content Provider cung cấp cơ chế truy xuất dữ liệu giữa các ứng dụng Android.
- Có nhiều cách để lưu trữ dữ liệu trong Android như: ghi file, cơ sở dữ liệu, hay lưu dữ liệu trên Internet... Và Content Provider giúp cho những ứng dụng khác nhau có thể truy xuất vào chung một nguồn dữ liệu đó.
- Muốn hiện thực một Content Provider trong Android chỉ cần khai báo một lớp kế thừa từ lớp ContentProvider:

```
public class MyContentProvider extends ContentProvider {  
  
}
```

2.2.4 Broadcast Receiver

- Cho phép ứng dụng đăng ký một sự kiện nào đó với hệ thống hoặc với các ứng dụng khác.
- Được sử dụng để nhận các sự kiện hoặc yêu cầu do các ứng dụng khác hoặc hệ thống phát đi, sau đó thực thi theo yêu cầu đó.
- Có 2 hình thức phát-nhận:
 - o Không có thứ tự (Normal broadcast -> sendBroadcast): đây là hình thức hoạt động bất đồng bộ (Asynchronous), tất cả các

receiver đã được đăng ký đều có thể nhận được những sự kiện hay yêu cầu tương ứng, và có thể xảy ra đồng thời.

- Có thứ tự (Ordered broadcast -> sendOrderedBroadcast) tùy thuộc vào độ ưu tiên khi đăng ký mà các receiver nhận được các sự kiện tương ứng. Sau khi một receiver nhận được sự kiện thì các receiver tiếp theo mới được tiếp nhận sự kiện.
- Muốn hiện thực Broadcast Receiver trong Android chỉ cần khai báo một lớp kế thừa từ lớp BroadcastReceiver:

```
public class MyReceiver extends BroadcastReceiver {  
  
}
```

2.3 Cấu trúc thư mục chính trong một dự án Android (Android project)

2.3.1 src

- chứa tất cả các file .java của dự án (.java source files)
- khi tạo một dự án Android trên Eclipse, thông thường file MainActivity.java sẽ mặc định nằm trong thư mục này và trong file sẽ hiện thực một Activity có thể hiển thị thông tin lên màn hình thiết bị, khi người dùng chọn vào icon của ứng dụng.

2.3.2 gen

- chứa file R.java được Android tự động tạo ra, người lập trình không được chỉnh sửa trong file này.
- R.java sẽ tham chiếu tới tất cả các tài nguyên, các biến được sử dụng trong toàn bộ dự án, thông qua file này, Android sẽ truy xuất được các đối tượng trong quá trình thực thi tác vụ.

2.3.3 assets

- có thể chứa tất cả các file dữ liệu hoặc hình ảnh cần sử dụng cho những mục đích quan trọng trong ứng dụng như: nhạc chuông, nhạc nền, html...

- những file dữ liệu thường có kích thước nhỏ để không làm tăng dung lượng của ứng dụng khi cài đặt trên thiết bị.

2.3.4 bin

- chứa các file được tạo ra sau quá trình biên dịch (.dex, .apk, .xml,...) và được sử dụng để thực thi chương trình. Đặc biệt file .apk có thể được cài đặt trực tiếp trên thiết bị để người sử dụng có thể chạy ứng dụng.

2.3.5 libs

- chứa các thư viện cần thiết cho ứng dụng, thường là các file .jar

2.3.6 res

- chứa tất cả các tài nguyên của ứng dụng, bao gồm các thư mục chính:
 - drawable: chứa các hình ảnh, icon được sử dụng trong ứng dụng (hình nền, icon,...) và các file .xml dùng trong một số mục đích đặc biệt khác. Tùy theo độ phân giải hay kích thước của màn hình thiết bị mà Android hỗ trợ các thư mục tương ứng: drawable-hdpi, drawable-ldpi, ...
 - layout: chứa các file thiết kế giao diện dưới dạng xml
 - menu: chứa các file thiết kế menu trong ứng dụng dưới dạng xml
 - raw: chứa các file phục vụ cho việc cấu hình ứng dụng
 - values: chứa các file lưu giá trị như cờ chữ, mã màu,... (bao gồm các thư mục values-hdpi, values-ldpi,... được sử dụng tương tự với thư mục drawable)

2.3.7 AndroidManifest

- Đây là file được dùng để mô tả những thông tin cơ bản của ứng dụng dưới dạng xml: tên ứng dụng, đăng ký Activity, Service, Receiver, SDK, đăng ký quyền tương tác với thiết bị như: Read/Write file, Network, Bluetooth,...
- Bất cứ dự án Android nào cũng được tạo ra với một file AndroidManifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.hello07thc"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="18"
        android:targetSdkVersion="19" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".Hello07THC"
            android:label="@string/app_name" >
            <intent-filter>
                <action
                    android:name="android.intent.action.MAIN" />
                <category
                    android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
```

- **xmlns:android:** đường dẫn mặc định xác định nguồn tài nguyên được sử dụng trong ứng dụng (mặc định là android)
- **package:** tên package của ứng dụng hiện hành
- **versionCode:** số phiên bản code của ứng dụng
- **versionName:** tên phiên bản của ứng dụng
- **minSdkVersion:** phiên bản của SDK thấp nhất mà ứng dụng hỗ trợ
- **targetSdkVersion:** phiên bản SDK mục tiêu để chạy ứng dụng
- **application-allowBackup:** cho phép backup dữ liệu của ứng dụng
- **application-icon:** hình đại diện cho ứng dụng

- **application-label:** tên đại diện của ứng dụng khi cài đặt trên thiết bị
- **application-theme:** hình nền của ứng dụng
- **activity-name:** khai báo tên của lớp Activity trong ứng dụng
- **activity-label:** tên của Activity được hiển thị trên màn hình
- **intent-filter-action:** khai báo các thuộc tính cho Activity
 - **android.intent.action.MAIN:** màn hình chính của ứng dụng, được hiển thị đầu tiên khi bắt đầu chạy ứng dụng
 - **android.intent.category.LAUNCHER:** cho phép người sử dụng có thể chạy ứng dụng từ icon trên thiết bị.

CHƯƠNG III: XÂY DỰNG GIAO DIỆN NGƯỜI SỬ DỤNG

3.1 Activity

3.1.1. Định nghĩa

- Là một trong 4 thành phần chính trong một ứng dụng Android
- Được dùng để hiển thị một màn hình trong ứng dụng và có thể tương tác trực tiếp với người sử dụng.

3.1.2. Vòng đời Activity

- Để sử dụng Activity một cách hiệu quả nhất, việc đầu tiên là phải hiểu rõ cơ chế hoạt động và vòng đời của nó.
- Vòng đời Activity mô tả quá trình khởi tạo cho đến khi kết thúc của một Activity, bao gồm 6 trạng thái và 7 sự kiện tương ứng.

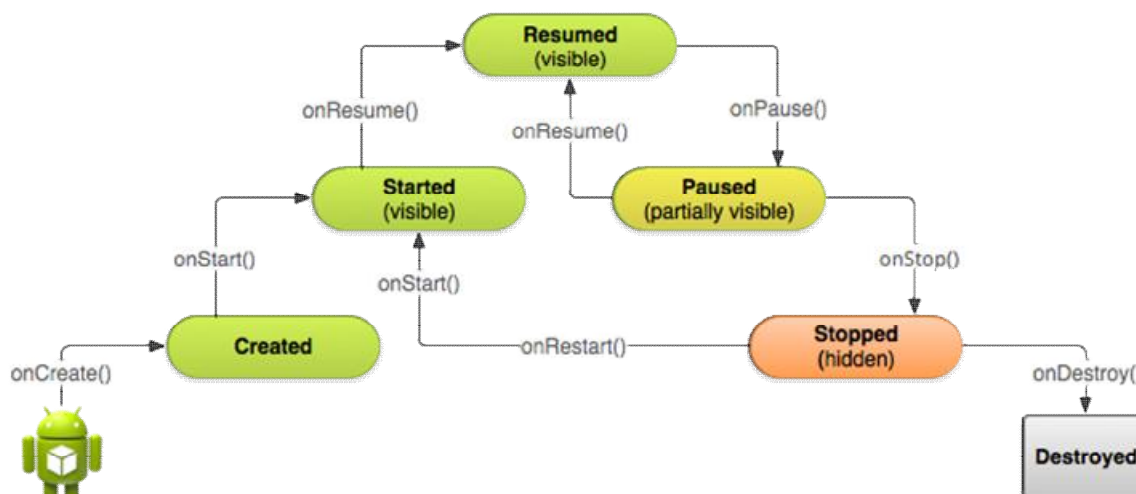


Figure 23: Vòng đời Activity

- **Created:** khi ứng dụng bắt đầu chạy, sự kiện **onCreate()** được hệ thống thực thi (khởi tạo và load các đối tượng giao diện), ứng dụng sẽ ở trạng thái Created.
- **Started (visible):** sau khi các đối tượng giao diện đã được khởi tạo và load vào vùng nhớ (Heap) của hệ thống, sự kiện **onStart()** được hệ thống thực thi để hiển thị các đối tượng giao diện lên

màn hình. Tại thời điểm này, người sử dụng có thể nhìn thấy các thành phần của ứng dụng trên màn hình, tuy nhiên chưa thể tương tác được với chúng.

- **Resumed (visible):** sau khi các đối tượng giao diện được hiển thị lên màn hình, sự kiện **onResume()** được hệ thống thực thi, lúc này người sử dụng có thể tương tác được với ứng dụng. Trạng thái này được xem như là trạng thái **“Running”** của ứng dụng.
- **Paused (partially visible):** khi một phần Activity bị che bởi một màn hình của Activity khác hay là dialog/popup, sự kiện **onPause()** sẽ được hệ thống thực thi đưa Activity về trạng thái dừng hoạt động. Lúc này, ứng dụng sẽ không thực thi bất cứ tác vụ nào và người sử dụng cũng không thể tương tác được với ứng dụng.
- **Stopped (hidden):** khi Activity khác hiển thị lên màn hình và che phủ toàn bộ Activity hiện hành, sự kiện **onStop()** sẽ được hệ thống thực thi đưa Activity hiện hành ẩn xuống background. Lúc này, tất cả các trạng thái, đối tượng của Activity sẽ được lưu lại, nhưng sẽ không có bất cứ tác vụ nào được thực thi.
- **Destroyed:** khi người sử dụng muốn kết thúc ứng dụng, sự kiện **onDestroyed()** sẽ được hệ thống thực thi và kết thúc quá trình hoạt động của ứng dụng.
- **onRestart():** sự kiện này sẽ được hệ thống thực thi khi người sử dụng hiển thị lại Activity sau khi Activity ở trạng thái **Stopped**. Lúc này, ứng dụng sẽ quay về trạng thái **Started**.

3.2 Xây dựng giao diện người dùng (User Interface Design)

3.2.1. Layouts

3.2.1.1. Định nghĩa

- Một Layout định nghĩa một cấu trúc gồm các thành phần giao diện trong Android. Ví dụ: trong một Activity có chứa một Layout, bao gồm nhiều thành phần UI như: TextView, EditText, CheckBox,...hoặc nhiều Layout con như: LinearLayout, RelativeLayout,...
- Có 2 cách mô tả Layout trong Android:

- Mô tả các thành phần UI trong XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a TextView" />
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a Button" />
</LinearLayout>
```

- Tạo Layout trong quá trình chạy chương trình (Runtime)

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    LinearLayout mainLayout = new LinearLayout(this);
    LayoutParams lparams =
        new LayoutParams(LayoutParams.WRAP_CONTENT,
            LayoutParams.WRAP_CONTENT);
    TextView tv = new TextView(this);
    tv.setLayoutParams(lparams);
    tv.setText("HelloWorld");
    mainLayout.addView(tv);
    setContentView(mainLayout);
}
```

3.2.1.2. Phân loại Layout

- Android hỗ trợ rất nhiều dạng Layout khác nhau, giúp người lập trình có thể sử dụng linh hoạt trong việc thiết kế màn hình.
- Bên cạnh những Layout chuẩn của Android, người lập trình cũng có thể kế thừa và điều chỉnh thành những Layout có đặc tính riêng, thích hợp sử dụng trong những ứng dụng đặc thù.
- Sau đây là những Layout được sử dụng phổ biến nhất trong Android:

STT	Tên Layout	Đặc điểm
1	LinearLayout	Các thành phần bên trong được sắp xếp liên kế nhau theo một hướng nhất định: ngang hoặc dọc.
2	RelativeLayout	Các thành phần bên trong được sắp xếp theo những vị trí có quan hệ với nhau: phải, trái, trên, dưới,...
3	TableLayout	Các thành phần bên trong được sắp xếp theo dạng: hàng, cột.
4	AbsoluteLayout	Các thành phần bên trong được sắp xếp theo những vị trí xác định.
5	FrameLayout	Chỉ chứa duy nhất một thành phần giao diện bên trong.
6	ListView	Chứa các thành phần bên trong theo dạng danh sách và người sử dụng có thể kéo lên, xuống để xem nội dung trong danh sách đó.
7	GridView	Chứa các thành phần bên trong theo dạng lưới, người sử dụng có thể kéo lên, xuống, trái, phải để xem nội dung trong đó.

3.2.1.3. Các thuộc tính cơ bản của Layouts

Thuộc tính	Mô tả
android:id	Đây là thuộc tính dùng để xác định đối tượng UI, trong một ứng dụng, giá trị này phải là duy nhất.
android:layout_width	Thuộc tính quy định độ rộng của đối tượng UI.
android:layout_height	Thuộc tính quy định độ cao của đối tượng UI.
android:layout_marginTop	Thuộc tính quy định khoảng trống phía trên của đối tượng UI so với một đối tượng UI khác.
android:layout_marginBottom	Thuộc tính quy định khoảng trống phía dưới của

	đối tượng UI so với một đối tượng UI khác.
android:layout_marginLeft	Thuộc tính quy định khoảng trống phía bên trái của đối tượng UI so với một đối tượng UI khác.
android:layout_marginRight	Thuộc tính quy định khoảng trống phía bên phải của đối tượng UI so với một đối tượng UI khác.
android:layout_gravity	Thuộc tính quy định vị trí của các thành phần bên trong của một đối tượng UI.
android:layout_weight	Thuộc tính quy định khoảng trống theo tỷ lệ dành cho các thành phần bên trong của một đối tượng UI.
android:paddingTop	Thuộc tính quy định khoảng trống của các thành phần bên trong so với đường biên trên của một đối tượng UI.
android:paddingBottom	Thuộc tính quy định khoảng trống của các thành phần bên trong so với đường biên dưới của một đối tượng UI.
android:paddingLeft	Thuộc tính quy định khoảng trống của các thành phần bên trong so với đường biên bên trái của một đối tượng UI.
android:paddingRight	Thuộc tính quy định khoảng trống của các thành phần bên trong so với đường biên bên phải của một đối tượng UI.

3.2.1.4. Ứng dụng

- Các Layout được sử dụng để thiết kế UI trong hầu hết các ứng dụng Android.
- Tùy theo yêu cầu của ứng dụng, người lập trình có thể sắp xếp các Layout theo những cấu trúc khác nhau để tạo ra UI thích hợp.
- Ví dụ cấu trúc Layout trong 1 ứng dụng có thể được sắp xếp như sau:

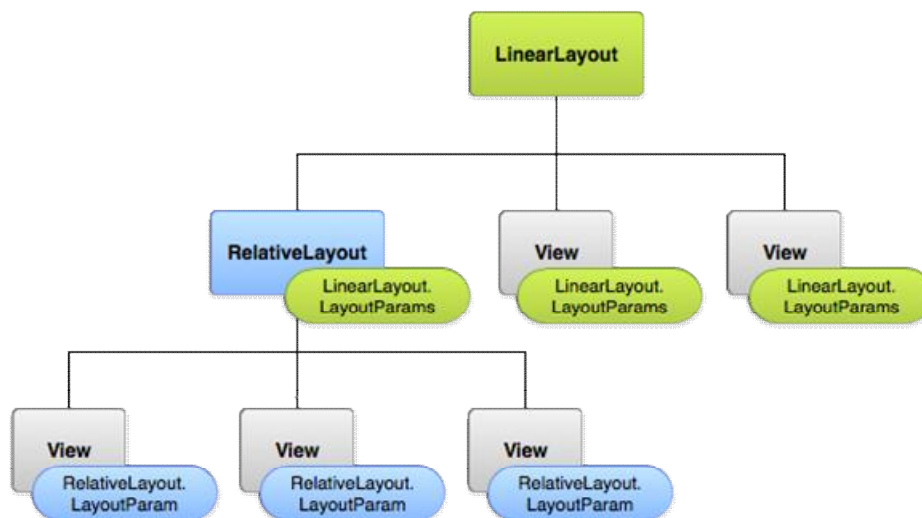


Figure 24: Cấu trúc Layouts

3.2.2. Các thành phần giao diện khác

3.2.2.1. TextView

- Là thành phần được sử dụng để hiển thị nội dung lên màn hình cho người sử dụng.
- Người sử dụng sẽ không thể trực tiếp chỉnh sửa được nội dung của TextView.
- Ví dụ:
 - Trong file *res/main_activity.xml*, khai báo một đối tượng TextView với id là text:

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <TextView
        android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a TextView" />
</LinearLayout>

```

- Trong file *src/MainActivity.java*, có thể gán nội dung cho đối tượng TextView như sau:

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

    TextView tv = (TextView) findViewById(R.id.text);
    tv.setText("HelloWorld");
}
```

3.2.2.2. EditText

- Là thành phần được kế thừa từ TextView nên có thể hiển thị nội dung lên màn hình thiết bị như TextView.
- Ngoài ra, người sử dụng có thể trực tiếp chỉnh sửa được nội dung của EditText.
- Ví dụ:
 - Trong file *res/main_activity.xml*, khai báo một đối tượng EditText với id là editText:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <EditText
        android:id="@+id/edittext"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="Hello, I am an EditText. Please input your text!"
        android:inputType="text" />
</LinearLayout>
```

- Trong file *src/MainActivity.java*, có thể lấy được nội dung mà người sử dụng nhập vào EditText như sau:

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

    EditText editText = (EditText) findViewById(R.id.edittext);
    String userInput = editText.getText().toString();
    Log.d("onCreate", "userInput = " + userInput);
}
```

3.2.2.3. Button

- Là thành phần được sử dụng để thực hiện một tác vụ nào đó khi người sử dụng nhấn chọn.
- Ví dụ:
 - o Trong file *res/main_activity.xml*, khai báo một đối tượng Button với id là *sendBtn*:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <Button
        android:id="@+id/sendBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/button_icon"
        android:text="Hello, I am a Button. Click on me!"
        android:onClick="sendMessage"/>
</LinearLayout>
```

- o Trong file *src/MainActivity.java*, có thể hiện thực một tác vụ được thực hiện khi người sử dụng nhấn vào Button:

```
public void sendMessage(View view) {
    // Implement function to send the message here
}
```


3.2.2.4. Toggle Button

- Là thành phần được kế thừa từ Button, Toggle Button được sử dụng trong trường hợp cần chuyển đổi giữa 2 trạng thái ON/OFF.
- Ví dụ:
 - o Trong file *res/main_activity.xml*, khai báo một đối tượng ToggleButton với id là toggleBtn:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <ToggleButton
        android:id="@+id/toggleBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a ToggleButton. Click on me!"
        android:textOn="Vibrate on"
        android:textOff="Vibrate off"
        android:onClick="onToggleClicked"/>
</LinearLayout>
```

- o Trong file *src/MainActivity.java*, có thể hiện thực một tác vụ được thực hiện khi người sử dụng nhấn vào ToggleButton:

```
public void onToggleClicked(View view) {
    // Is the Toggle on?
    boolean isOn = ((ToggleButton) view).isChecked();
    if (isOn) {
        // Enable vibrate
    } else {
        // Disable vibrate
    }
}
```

3.2.2.5. CheckBox

- Giúp cho người sử dụng có thể chọn lựa 1 hay nhiều đối tượng trong một danh sách.
- Mỗi CheckBox chỉ có 2 trạng thái: checked/unchecked
- Ví dụ:

- Trong file *res/main_activity.xml*, khai báo 2 đối tượng CheckBox với id là:
 - id 1: checkbox_meat
 - id 2: checkbox_cheese

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <CheckBox
        android:id="@+id/checkbox_meat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/meat"
        android:onClick="onCheckBoxClicked"/>
    <CheckBox
        android:id="@+id/checkbox_cheese"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/cheese"
        android:onClick="onCheckBoxClicked"/>
</LinearLayout>
```

- Trong file *src/MainActivity.java*, có thể hiện thực một tác vụ được thực hiện khi người sử dụng nhấn vào CheckBoxes:

```
public void onCheckBoxClicked(View view) {
    // Is the View checked?
    boolean isChecked = ((CheckBox) view).isChecked();
    // Check which CheckBox was clicked?
    switch (view.getId()) {
        case R.id.checkbox_meat:
            if (isChecked()) {
                // Put some meat on the sandwich
            } else {
                // Remove meat
            }
            break;
        case R.id.checkbox_cheese:
            if (isChecked()) {
                // Put some cheese on the sandwich
            } else {
                // Remove cheese
            }
            break;
    }
}
```

3.2.2.6. ImageView

- Là thành phần được sử dụng để hiển thị một hình ảnh trên màn hình ứng dụng.
- Ví dụ:
 - o Trong file *res/main_activity.xml*, khai báo một đối tượng ImageView với id là *vietnamImg*:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <ImageView
        android:id="@+id/vietnamImg"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@Drawable/vietnam_map"/>
</LinearLayout>
```

- o Trong file *src/MainActivity.java*, có thể thay thế hình ảnh trong quá trình chạy ứng dụng bằng các dòng code như sau:

```
private ImageView mVietnamImg;

public class MainActivity extends Activity {

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

        mVietnamImg = (ImageView) findViewById(R.id.vietnamImg);
        mVietnamImg.setImageResource(R.drawable.vietnam_ocean);
    }

    private void changeImage(Bitmap bitmap) {
        mVietnamImg.setImageBitmap(bitmap);
    }
}
```

3.2.2.7. Spinner

- Giúp cho người sử dụng có thể chọn lựa 1 đối tượng từ trong một danh sách.
- Ví dụ:
 - o Trong file *res/main_activity.xml*, khai báo 1 đối tượng Spinner với id là *planets_spinner*:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <Spinner
        android:id="@+id/planets_spinner"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

- o Trong file *res/values/strings.xml*, tạo ra một danh sách các đối tượng sẽ được hiển thị trong danh sách của Spinner:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="planets_array">
        <item>Mercury</item>
        <item>Venus</item>
        <item>Earth</item>
        <item>Mars</item>
        <item>Jupiter</item>
        <item>Saturn</item>
        <item>Uranus</item>
        <item>Neptune</item>
    </string-array>
</resources>
```

- Trong file *src/MainActivity.java*, có thể hiện thực một Adapter để chứa danh sách các đối tượng của Spinner:

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

    Spinner spinner = (Spinner) findViewById(R.id.spinner);
    // Create an ArrayAdapter using the string array and a default spinner layout
    ArrayAdapter<CharSequence> adapter =
        ArrayAdapter.createFromResource(this,
            R.array.planets_array, android.R.layout.simple_spinner_item);
    // Specify the layout to use when the list of choices appears
    adapter.setDropDownViewResource(
        android.R.layout.simple_spinner_dropdown_item);
    // Apply the adapter to the spinner
    spinner.setAdapter(adapter);
    spinner.setOnItemSelectedListener(new OnItemSelectedListener() {
        @Override
        public void onItemSelected(AdapterView<?> arg0,
            View arg1, int arg2, long arg3) {
            // TODO Auto-generated method stub
        }

        @Override
        public void onNothingSelected(AdapterView<?> arg0) {
            // TODO Auto-generated method stub
        }
    });
}
```

3.2.2.8. ProgressBar

- Là thành phần được sử dụng để hiển thị tiến trình đã được thực thi của một tác vụ nào đó.
- Thường được biểu diễn tiến trình từ khi bắt đầu, đến khi kết thúc: 0 -> 100%
- Ví dụ:

- Trong file *res/main_activity.xml*, khai báo một đối tượng ProgressBar với id là readingBookProgressBar:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <ProgressBar
        android:id="@+id/progress_loading"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        style="?android:attr/progressBarStyle"/>
</LinearLayout>
```

- Dựa vào thuộc tính style trong xml, có thể chọn được khoảng 11 kiểu ProgressBar khác nhau do Android hỗ trợ.
- Trong file *src/MainActivity.java*, có thể hiển thị thanh ProgressBar trong quá trình xử lý dữ liệu hay download/upload dữ liệu:

```
public class MainActivity extends Activity {

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

        ProgressBar loadingProgressBar =
            (ProgressBar) mDeviceListDialog.
                findViewById(R.id.progress_loading);
        if (loadingProgressBar != null) {
            loadingProgressBar.setVisibility(View.VISIBLE);
        }
    }
}
```

3.2.2.9. Dialog

- Là một thành phần UI dạng cửa sổ nhỏ, được sử dụng để hiển thị một thông báo, hay là một hành động cần sự xác nhận của người sử dụng trước khi được thực thi.
- Có 3 loại Dialog chính trong Android:

- **AlertDialog**: dùng để hiển thị một cửa sổ thông báo có thể bao gồm: tiêu đề, tối đa 3 Button, một danh sách các đối tượng người sử dụng có thể chọn, hay một Custom Layout.
- **DatePickerDialog hoặc TimePickerDialog**: dùng để hiển thị các thông tin về ngày giờ và cho phép người sử dụng chọn ngày tháng.

- Ví dụ:

- Thêm tiêu đề cho Dialog:

```
public class MainActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.main_activity);  
        // 1. Instantiate an AlertDialog.Builder with its constructor  
        AlertDialog.Builder builder = new AlertDialog.Builder(this);  
        // 2. Chain together various setter methods to set the dialog characteristics  
        builder.setMessage(R.string.dialog_message).  
            setTitle(R.string.dialog_title);  
        // 3. Get the AlertDialog from create()  
        AlertDialog dialog = builder.create();  
    }  
}
```

- Thêm Button cho Dialog:

```
// 1. Instantiate an AlertDialog.Builder with its constructor  
AlertDialog.Builder builder = new AlertDialog.Builder(this);  
// 2. Add the buttons  
builder.setPositiveButton(R.string.ok, new  
    DialogInterface.OnClickListener() {  
        public void onClick(DialogInterface dialog, int id) {  
            // User clicked OK button  
        }  
    });  
builder.setNegativeButton(R.string.cancel, new  
    DialogInterface.OnClickListener() {  
        public void onClick(DialogInterface dialog, int id) {  
            // User cancelled the dialog  
        }  
    });  
// Set other dialog properties  
...  
// 3. Get the AlertDialog from create()  
AlertDialog dialog = builder.create();
```

3.2.2.10. Popup

- Tương tự như Dialog, Popup cũng là một cửa sổ nhỏ hiển thị thông tin trong Android, tuy nhiên, Popup có thể được hiển thị bất cứ vị trí nào phía trên Activity hiện hành.
- Có 2 loại Popup cơ bản:
 - **PopupWindow**: dùng để hiển thị một cửa sổ nhỏ chứa một View tùy ý, có thể được di chuyển ở mọi vị trí trên Activity hiện hành.
 - **PopupMenu**: dùng để hiển thị một Menu, được định vị phía trên hoặc dưới của một View trong Activity hiện hành.
- Ví dụ:
 - PopupWindow:
 - Tạo file Layout res/popup_window.xml để hiển thị nội dung trên PopupWindow:

```
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_margin="20dp">
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="It's a PopupWindow" />
    <ImageView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:src="@drawable/ic_launcher" />
    <Button
        android:id="@+id/dismiss"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Dismiss" />
</LinearLayout>
```

- Trong file src/MainActivity.java, hiển thị PopupWindow bằng cách load Layout vừa tạo lên một đối tượng PopupWindow:


```
private PopupWindow pw;
private void initiatePopupWindow() {

    //We need to get the instance of the LayoutInflater, use the context of this activity

    LayoutInflater inflater = (LayoutInflater) this
        .getSystemService(Context.LAYOUT_INFLATER_SERVICE);

    //Inflate the view from a predefined XML layout

    View layout = inflater.inflate(R.layout.popup_window, null);

    // create a 300px width and 470px height PopupWindow

    pw = new PopupWindow(layout, 300, 470, true);
    // display the popup in the center

    pw.showAtLocation(layout, Gravity.CENTER, 0, 0);
    Button dismissBtn = (Button) layout.findViewById(R.id.dismiss);
    dismissBtn.setOnClickListener(new OnClickListener() {

        // dismiss popup

        pw.dismiss();

    });

}
```

- PopupMenu:

- Tạo file Layout res/popup_menu.xml để hiển thị nội dung trên PopupMenu:

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
    <item
        android:id="@+id/one"
        android:title="One"/>

    <item
        android:id="@+id/two"
        android:title="Two"/>

    <item
        android:id="@+id/three"
        android:title="Three"/>

</menu>
```

- Trong file `src/MainActivity.java`, hiển thị `PopupMenu` bằng cách load `Layout` vừa tạo lên một đối tượng `PopupMenu`:

```
private void initiatePopupMenu(View view) {  
  
    PopupMenu popupMenu = new PopupMenu(this, view);  
  
    popupMenu.setOnMenuItemClickListener(this);  
  
    popupMenu.inflate(R.menu.popup_menu);  
  
    popupMenu.show();  
  
}
```

3.2.2.11. ListView

- Là một thành phần được sử dụng để hiển thị danh sách các đối tượng UI một chiều.
- Người sử dụng có thể kéo danh sách lên/xuống (danh sách dọc) hoặc trái/phải (danh sách ngang), hoặc chọn mỗi đối tượng trong danh sách để thực thi một tác vụ riêng biệt.
- Danh sách các đối tượng UI được tự động thêm vào `ListView` nhờ một đối tượng gọi là `Adapter`.
- Ví dụ:
 - Trong file `res/main_activity.xml`, khai báo một đối tượng `ListView` với id là `listview`:

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
    <ListView  
        android:id="@+id/listview"  
        android:layout_height="wrap_content"  
        android:layout_width="match_parent">  
    </ListView>  
</LinearLayout>
```

- Trong file *src/MainActivity.java*, có thể hiển thị một danh sách các đối tượng cần hiển thị

```
public class MainActivity extends Activity {

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

        // Get ListView object from xml
        ListView listView = (ListView) findViewById(R.id.listview);
        // Defined Array values to show in ListView
        String[] values = new String[] { "Android List View",
            "Adapter implementation",
            "Simple List View In Android",
            "Create List View Android",
            "Android Example",
            "List View Source Code",
            "List View Array Adapter",
            "Android Example List View"
        };

        // Define a new Adapter
        // First parameter - Context
        // Second parameter - Layout for the row
        // Third parameter - ID of the TextView to which the data is written
        // Forth - the Array of data
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
            android.R.layout.simple_list_item_1, android.R.id.text1, values);
        // Assign adapter to ListView
        listView.setAdapter(adapter);
        // ListView Item Click Listener
        listView.setOnItemClickListener(new OnItemClickListener() {

            @Override
            public void onItemClick(AdapterView<?> parent, View view,
                int position, long id) {

                // ListView Clicked item index
                int itemPosition = position;
                // ListView Clicked item value
                String itemValue =
                    (String) listView.getItemAtPosition(position);
                // Show Alert
                Toast.makeText(getApplicationContext(),
                    "Position : " + itemPosition+ " ListItem : "
                    + itemValue , Toast.LENGTH_LONG).show();

            }

        });
    }
}
```

3.2.2.12. GridView

- Tương tự như ListView, đây là một thành phần được sử dụng để hiển thị danh sách các đối tượng UI theo dạng lưới 2 chiều.
- Người sử dụng có thể chọn mỗi đối tượng trong danh sách để thực thi một tác vụ riêng biệt.
- Danh sách các đối tượng UI được tự động thêm vào GridView nhờ một đối tượng gọi là Adapter.
- Ví dụ:
 - o Trong file *res/main_activity.xml*, khai báo một đối tượng GridView với id là *gridview*:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <GridView
        android:id="@+id/gridview"
        android:layout_height="wrap_content"
        android:layout_width="match_parent">
    </GridView>
</LinearLayout>
```

- o Trong file *src/MainActivity.java*, có thể hiển thị một danh sách các đối tượng cần hiển thị:

```
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main_activity);
        GridView gridview = (GridView) findViewById(R.id.gridview);
        gridview.setAdapter(new ImageAdapter(this));
        gridview.setOnItemClickListener(new OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent,
                View v, int position, long id) {
                Toast.makeText(MainActivity.this, "" + position,
                    Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

- Tạo một lớp ImageAdapter kế thừa từ lớp BaseAdapter:

```
public class ImageAdapter extends BaseAdapter {

    private Context mContext;

    public ImageAdapter(Context c) {
        mContext = c;
    }

    public int getCount() {
        return mThumbIds.length;
    }

    public Object getItem(int position) {
        return null;
    }

    public long getItemId(int position) {
        return 0;
    }

    // create a new ImageView for each item referenced by the Adapter
    public View getView(int position, View convertView, ViewGroup parent) {
        ImageView imageView;
        if (convertView == null) { // if it's not recycled, initialize some attributes
            imageView = new ImageView(mContext);
            imageView.setLayoutParams(
                new GridView.LayoutParams(85, 85));
            imageView.setScaleType(
                ImageView.ScaleType.CENTER_CROP);
            imageView.setPadding(8, 8, 8, 8);
        } else {
            imageView = (ImageView) convertView;
        }
        imageView.setImageResource(mThumbIds[position]);
        return imageView;
    }

    // references to our images
    private Integer[] mThumbIds = {
        R.drawable.img_0, R.drawable.img_1,
        R.drawable.img_2, R.drawable.img_3,
        R.drawable.img_4, R.drawable.img_5,
        R.drawable.img_6, R.drawable.img_7,
        R.drawable.img_8, R.drawable.img_9,
        R.drawable.img_10, R.drawable.img_11,
        R.drawable.img_12, R.drawable.img_13,
        R.drawable.img_14, R.drawable.img_15,
        R.drawable.img_16, R.drawable.img_17,
        R.drawable.img_18, R.drawable.img_19,
        R.drawable.img_20
    };
}
```

3.3 Menu trong ứng dụng Android

3.3.1 Định nghĩa

- Là một thành phần UI được sử dụng rất rộng rãi, dùng để chứa các mục liên quan đến chức năng của ứng dụng như: thiết lập, cài đặt, thông tin,...
- Menu được tích hợp trên thanh công cụ Action Bar của thiết bị Android từ Android phiên bản 3.0 (API 11).

3.3.2 Phân loại

- Options menu: dùng để hiển thị các chức năng cần thực hiện của một Activity. Vị trí của Options menu thường nằm trên thanh công cụ Action Bar.
- Context menu: dùng để hiển thị các chức năng cần thực hiện của một View bất kỳ trong Activity, vị trí của Context menu thường nằm giữa màn hình. Muốn hiển thị Context menu cho View nào thì phải đăng ký Context menu cho View đó. Như vậy, lúc người sử dụng nhấn giữ một lúc vào View đã đăng ký, thì sẽ hiển thị Context menu.
- Popup menu: dùng để hiển thị các chức năng của một đối tượng UI bất kỳ trong Activity, vị trí của Popup menu phụ thuộc vào đối tượng UI mà nó được thiết lập quan hệ vị trí (trên, dưới).

3.3.3 Ứng dụng

- Tạo một Options menu:
 - o Trong file menu/activity_main_menu.xml, tạo ra một danh sách các đối tượng chức năng hiển thị trong menu:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/menu_settings"
        android:title="@string/menu_settings"/>
    <item
        android:id="@+id/finish"
        android:title="@string/finish" />
</menu>
```

- Trong file MainActivity.java, hiện thực lại hai phương thức kế thừa từ lớp Activity:

```
public class MainActivity extends Activity {  
    @Override  
    public boolean onCreateOptionsMenu(Menu menu) {  
        MenuInflater inflater = getMenuInflater();  
        inflater.inflate(R.menu.activity_main_menu, menu);  
        return true;  
    }  
  
    @Override  
    public boolean onOptionsItemSelected(MenuItem item) {  
        // Handle item selection  
        switch (item.getItemId()) {  
            case R.id.menu_settings:  
                settings();  
                return true;  
            case R.id.finish:  
                finish();  
                return true;  
            default:  
                return super.onOptionsItemSelected(item);  
        }  
    }  
}
```

- Tạo một Context menu:

- Trong file xml/main_activity.xml, tạo một đối tượng hình ảnh ImageView với id là imageview:

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
    <ImageView  
        android:id="@+id/imageview"  
        android:layout_height="wrap_content"  
        android:layout_width="match_parent">  
    </ImageView>  
</LinearLayout>
```

- Trong file MainActivity.java, hiện thực một Context menu chứa các hành động nhằm xử lý hình ảnh vừa tạo trong xml:

```
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main_activity);
        ImageView img = (ImageView) findViewById(R.id.imageview);
        registerForContextMenu(img);
    }
    @Override
    public void onCreateContextMenu(ContextMenu menu, View v,
        ContextMenuInfo menuInfo) {
        super.onCreateContextMenu(menu, v, menuInfo);
        menu.setHeaderTitle("Select The Action");
        menu.add(0, v.getId(), 0, "Edit");
        menu.add(0, v.getId(), 0, "Zoom in");
        menu.add(0, v.getId(), 0, "Zoom out");
    }

    @Override
    public boolean onContextItemSelected(MenuItem item) {
        if (item.getTitle() == "Edit") {
            Toast.makeText(this, "Edit image",
                Toast.LENGTH_SHORT).show();
        } else if (item.getTitle() == "Zoom in") {
            Toast.makeText(this, "Zoom in image",
                Toast.LENGTH_SHORT).show();
        } else if (item.getTitle() == "Zoom out") {
            Toast.makeText(this, "Zoom out image",
                Toast.LENGTH_SHORT).show();
        }
    }
}
```

3.4 Xử lý sự kiện trong Android (Event Handling)

- Mỗi sự kiện trong Android đại diện cho một phản hồi của hệ thống khi người sử dụng tương tác với các đối tượng điều khiển trong ứng dụng như: Button, CheckBox, ListView, GridView, Spinner,...
- Android framework sẽ đẩy tất cả các sự kiện xảy ra vào trong một vùng nhớ được gọi là Queue, được thiết kế dựa trên cơ chế FIFO (First in, first out).
- Các sự kiện sẽ được hệ thống lắng nghe và xử lý tuần tự theo từng sự kiện một. Mỗi sự kiện xảy ra sẽ có một cơ chế để lắng nghe và xử lý sự kiện đó tương ứng.
- Một số sự kiện phổ biến trong Android:

- `onClick()` – `onClickListener()`: xảy ra khi người sử dụng kích chọn vào đối tượng UI.
- `onLongClick()` – `onLongClickListener()`: xảy ra khi người sử dụng kích chọn và giữ đối tượng UI trong một thời gian ngắn.
- `onTouch()` – `onTouchListener()`: người sử dụng chạm nhẹ vào màn hình.

- Ví dụ:

```
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main_activity);

        //--- find both the buttons---
        Button sButton = (Button) findViewById(R.id.button_s);
        Button lButton = (Button) findViewById(R.id.button_l);

        // -- register click event with first button --
        sButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                // --- find the text view --
                TextView txtView =
                    (TextView) findViewById(R.id.text_id);
                // -- change text size --
                txtView.setTextSize(14);
            }
        });

        // -- register click event with second button --
        lButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                // --- find the text view --
                TextView txtView =
                    (TextView) findViewById(R.id.text_id);
                // -- change text size --
                txtView.setTextSize(24);
            }
        });
    }
}
```

CHƯƠNG IV: INTENT, INTENT FILTER, SERVICE AND BROADCAST RECEIVER

4.1 Intent

4.1.1 Định nghĩa

- Là thành phần được sử dụng để truyền thông điệp để thực hiện một hành động nào đó giữa các thành phần khác nhau trong một ứng dụng, hoặc giữa các ứng dụng với nhau.
- Intent có thể xem là một cấu trúc dữ liệu mô tả các hành động và cách thức cần được thực hiện, và hoạt động như là cầu nối trong Android.

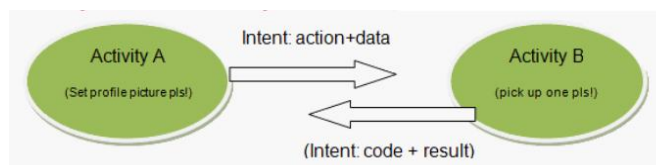


Figure 25: Intent làm cầu nối giữa 2 Activity

- Các thuộc tính của Intent:
 - **action**: là hành động cần được thực hiện, như: ACTION_VIEW, ACTION_MAIN,...
 - **data**: là dữ liệu được truyền từ thành phần này sang thành phần khác, thường được diễn tả dưới dạng Uri.
 - Ví dụ:
 - ACTION_VIEW content://contacts/people/1 -> Hiển thị thông tin về người với id là 1
 - ACTION_DIAL content://contacts/people/1 -> Hiển thị màn hình gọi đến người với id là 1
 - ACTION_DIAL tel: 0123456789 -> Hiển thị màn hình gọi với số gọi là 0123456789

```
Intent intent = new Intent (Intent.ACTION_DIAL, Uri.parse("tel:0123456789"));
startActivity(intent);
```

- Ngoài ra còn có một số thuộc tính khác như:
 - **category**: bổ sung thêm thông tin cho action của Intent. Ví dụ:
 - CATEGORY_LAUNCHER: thông báo với Launcher là ứng dụng top-level.
 - **type**: kiểu của data
 - **component**: thành phần sẽ nhận và xử lý Intent, và khi thuộc tính này được xác định thì các thuộc tính khác sẽ là thuộc tính phụ, có thể có hoặc không cần khai báo.
 - **extras**: mang theo đối tượng Bundle, chứa các giá trị bổ sung.
- Ví dụ:
 - Dùng Google để tìm kiếm thông tin:

```
Intent intent = new Intent(Intent.ACTION_WEB_SEARCH);
intent.putExtra(SearchManager.QUERY, "Cao đẳng Viễn Đông");
startActivity(intent);
```

- Gởi tin nhắn:

```
Intent intent = new Intent(Intent.ACTION_SENDTO, Uri.parse("sms:0123456789"));
intent.putExtra("sms_body", "We are 07THC!!!");
startActivity(intent);
```

4.1.2 Phân loại

Intent được phân biệt bằng 2 loại:

- Explicit Intent: là loại Intent có thuộc tính component đã được xác định rõ, và không cần phải bổ sung thêm các thuộc tính khác như: data, action. Được sử dụng để chạy các Activity hoặc Service khác trong cùng một ứng dụng. Ví dụ:

```
Intent intent = new Intent(MainActivity.this, SubActivity.class);
startActivity(intent);
```

- Implicit Intent: là loại Intent chưa xác định rõ component, mà cần phải sử dụng những thuộc tính khác để xác định thành phần sẽ nhận và xử lý Intent như: data, action.

```
// play song "hello.mp3" saved in the SD
Intent intent = new Intent(android.content.Intent.ACTION_VIEW);
Uri data = Uri.parse("file:///sdcard/hello.mp3");
String type = "audio/mp3";
intent.setDataAndType(data, type);
startActivity(intent);
```

4.1.3 Ứng dụng

Intent đóng vai trò rất quan trọng trong Android, và được sử dụng rộng rãi trong hầu hết các ứng dụng. Trong đó, có một số tính năng được sử dụng phổ biến nhất như sau:

4.1.3.1 Mở Activity khác

```
Intent intent = new Intent(MainActivity.this, SubActivity.class);
startActivity(intent);
```

4.1.3.2 Truyền dữ liệu cho một Activity khác

- Truyền dữ liệu thông qua một đối tượng Serializable:

- Activity bên gửi:

```
Intent intent = new Intent(MainActivity.this, SubActivity.class);
Serializable object = new SerializableObject();
intent.putExtra("Key", object);
startActivity(intent);
```

- Activity bên nhận:

```
Intent intent = getIntent();
SerializableObject object = (SerializableObject) intent.getSerializableExtra("Key");
```

* **Chú ý:** khi mở một Activity khác, nếu muốn nhận được kết quả sau khi Activity đó kết thúc thì phải dùng phương thức “startActivityForResult()” thay cho phương thức “startActivity()”.

+ Trong file MainActivity.java, mở một Activity mới là SubActivity.java:

```
Intent intent = new Intent(MainActivity.this, SubActivity.class);  
// khởi tạo một đối tượng SerializableObject  
SerializableObject object = new SerializableObject ();  
intent.putExtra("Key", object);  
startActivityForResult(intent, 1); //1 – là request code đăng ký Activity nhận kết quả
```

+ Trong file SubActivity.java, trước khi kết thúc sẽ trả về kết quả cho MainActivity.java:

```
Intent intent = new Intent();  
SerializableObject result = new SerializableObject();  
intent.putExtra("Key", result);  
setResult(Activity.RESULT_OK, intent);
```

+ Lúc này, MainActivity.java sẽ nhận được kết quả trả về thông qua phương thức “onActivityResult()” kế thừa từ lớp Activity:

```
@Override  
protected void onActivityResult(int requestCode, int resultCode, Intent intent) {  
    super.onActivityResult(requestCode, resultCode, intent);  
  
    switch (requestCode) {  
        case 1: //1 – là request code đã đăng ký nhận kết quả trước đó  
  
            if (Activity.RESULT_OK == resultCode) {  
                SerializableObject result =  
                    (SerializableObject) intent  
                        .getSerializableExtra("Key");  
                if (mUserSetting != null) {  
                    mUserSetting.update(userSetting);  
                }  
            }  
            return;  
        default:  
    }  
}
```

- Truyền dữ liệu thông qua đối tượng Bundle:

- Activity bên gửi:

```
Intent intent = new Intent(MainActivity.this, SubActivity.class);
Bundle bundle = new Bundle();
bundle.putInt("Key", 0123456789);
intent.putExtra(bundle);
startActivity(intent);
```

- Activity bên nhận:

```
Intent intent = getIntent();
Bundle bundle = intent.getExtras();
int value = bundle.getInt("Key");
```

4.1.3.3 Mở một Service:

```
Intent intent = new Intent(MainActivity.this, MyService.class);
startService(intent);
```

4.2 Intent Filter

4.2.1 Định nghĩa

- Là bộ lọc Intent, được sử dụng trong trường hợp là loại Implicit Intent, giúp cho hệ thống phân biệt được loại Intent mà nó có thể nhận và xử lý dữ liệu.
- Mô tả khả năng của component định nghĩa nó, khi hệ thống bắt được một Implicit Intent (chỉ chứa những thông tin chung chung về action, data, category...), nó sẽ so sánh với những thông tin trong Intent Filter của các component được khai báo trong ứng dụng. Nếu thông tin trùng hợp, hệ thống sẽ cho phép chạy ứng dụng thích hợp nhất để xử lý Intent bắt được.

4.2.2 Ứng dụng

4.2.2.1 Intent Filter trong Activity

- Cho phép Activity có khả năng bắt được và thực thi những Intent của hệ thống.
- Ví dụ: khai báo Intent Filter cho MainActivity trong Android Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.helloworld"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="11"
        android:targetSdkVersion="16" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:theme="@style/AppTheme"
        android:label="12345">
        <activity
            android:name="com.example.helloworld.MainActivity"
            android:label="Hehe" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
            <intent-filter>
                <action
                    android:name="android.intent.action.CALL_BUTTON" />
                <category
                    android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

- action “*android.intent.action.MAIN*” cho phép Activity hoạt động như là Activity chính của ứng dụng, được hiển thị làm màn hình đầu tiên.
- category “*android.intent.category.LAUNCHER*” cho phép ứng dụng được chạy khi người sử dụng click chọn vào icon trên màn hình thiết bị.
- action “*android.intent.action.CALL_BUTTON*” cho phép ứng dụng bắt được sự kiện người sử dụng bấm nút “CALL” trên thiết bị.
- category “*android.intent.category.DEFAULT*” khai báo cho hệ thống biết là ứng dụng sẽ được chạy chế độ mặc định khi có sự kiện “CALL” xảy ra.

4.2.2.2 Intent Filter trong Receiver

- Cho phép Broadcast Receiver đăng ký nhận những Intent để thực thi những tác vụ cần thiết trong ứng dụng.
- Ví dụ:
 - o khai báo Intent Filter cho chức năng nhận tin nhắn SMS trong Android Manifest:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.helloworld"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-permission android:name="android.permission.RECEIVE_SMS" />
    <uses-sdk
        android:minSdkVersion="11"
        android:targetSdkVersion="16" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:theme="@style/AppTheme"
        android:label="12345">
        <receiver android:name=".os.SmsMessageReceiver" android:enabled="false">
            <intent-filter>
                <action android:name="android.provider.Telephony.SMS_RECEIVED" />
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

- o Hiện thực một lớp os/SmsMessageReceiver.java kế thừa từ BroadcastReceiver:

```
public class SmsMessageReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        // Thực thi tác vụ cần thiết cho ứng dụng
        Log.d("SmsMessageReceiver", "Vừa mới nhận được một tin nhắn!");
    }
}
```


4.3 Service

4.3.1 Định nghĩa

Một Service là một thành phần quan trọng trong Android được sử dụng trong những tác vụ cần chạy ngầm (background) mà không cần phải có giao diện như Activity, cũng như không cần tương tác với người sử dụng: chơi nhạc, download...

4.3.2 Vòng đời Service

Một Service có thể hoạt động theo 2 dạng:

4.3.2.1 Started Service:

- Service này sẽ được chạy khi một thành phần nào đó của ứng dụng (Activity) gọi phương thức “startService()”. Một khi đã chạy, loại Service này có thể hoạt động mãi mãi ngầm bên dưới, và chỉ kết thúc khi một thành phần của ứng dụng gọi phương thức “stopService()” hoặc tự bản thân nó kết thúc bằng cách thực thi “stopSelf()”.
- Thường thực thi một tác vụ đơn và không phải trả kết quả về cho thành phần gọi nó. Khi tác vụ hoàn tất, tự Service sẽ kết thúc quá trình hoạt động.
- Tương tự như Activity, Service cũng có vòng đời hoạt động của riêng nó:

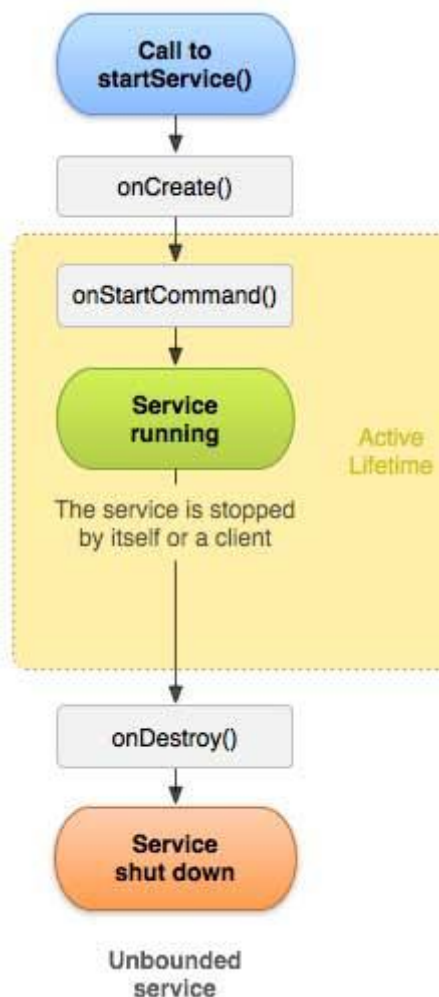


Figure 26: Vòng đời Started Service

4.3.2.2 Bound Service

- Service này sẽ được chạy khi một thành phần nào đó của ứng dụng (Activity) tạo một kết nối tới Service bằng cách gọi phương thức "bindService()". Kết nối cũng có thể được đóng lại khi phương thức "unbindService()" được thực thi.
- Cũng có thể tạo liên kết đến một Started Service bằng cách sử dụng phương thức "bindService()", lúc này Started Service sẽ không thực sự được kết thúc sau khi phương thức "stopService()" được thực thi, mà phải chờ đến khi liên kết được đóng lại.
- Hoạt động dựa trên cơ chế Client-Server interface, cho phép thành phần khởi chạy nó có thể tương tác trực tiếp với Service: gửi yêu cầu, nhận

kết quả trả về. Có thể tương tác giữa các ứng dụng khác nhau thông qua giao thức IPC (Interprocess Communication).

- Chỉ tồn tại và hoạt động khi có một thành phần nào đó của ứng dụng tạo liên kết tới nó (bound). Khi liên kết cuối cùng được đóng lại, Service cũng kết thúc quá trình hoạt động.
- Vòng đời hoạt động:

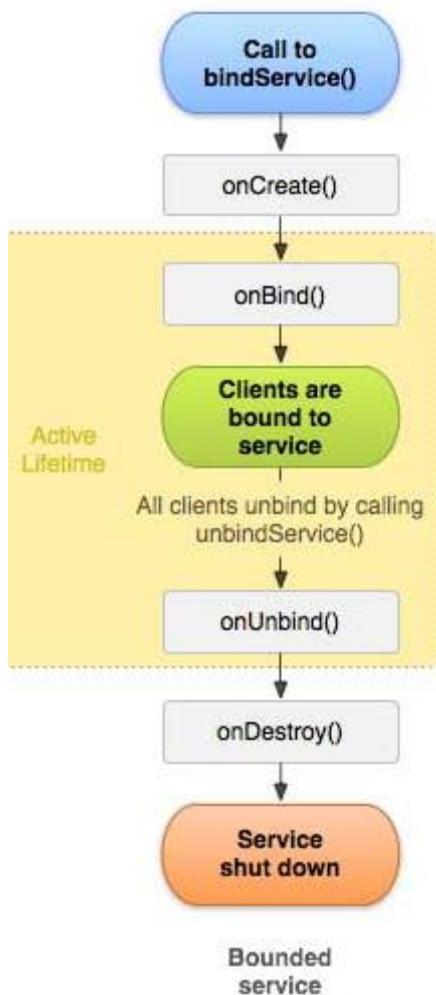


Figure 27: Vòng đời Bound Services

4.3.3 Ứng dụng

- Service được sử dụng trong những ứng dụng không cần một màn hình để giao tiếp với người sử dụng như Activity. Ví dụ: chương trình nghe nhạc, hoặc download dữ liệu từ internet...

- Ví dụ:
 - Tạo một Service dùng để download hình ảnh:

```
public class DownloadImage extends Service {  
  
    @Override  
    public IBinder onBind(Intent intent) {  
        return null;  
    }  
  
    @Override  
    public void onCreate() {  
        super.onCreate();  
    }  
  
    @Override  
    public int onStartCommand(Intent intent, int flags, int startId) {  
        int result = super.onStartCommand(intent, flags, startId);  
        downloadImage(intent.getStringExtra(EXTRA_URL));  
        return result;  
    }  
  
    private void downloadImage(String url) {  
        // Download image  
        Log.d("DownloadImage", "Start downloading image...");  
    }  
}
```

- Trong file Android Manifest, đăng ký Service vừa tạo ra:

```
<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
    package="com.example.helloworld"  
    android:versionCode="1"  
    android:versionName="1.0" >  
    <application  
        android:allowBackup="true"  
        android:icon="@drawable/ic_launcher"  
        android:theme="@style/AppTheme"  
        android:label="12345">  
  
        <service  
            android:name="DownloadImage"  
            android:label="@string/download_img">  
  
        </application>  
  
</manifest>
```

- Trong file MainActivity.java, gọi phương thức “startService” để bắt đầu quá trình download:

```
Intent intent = new Intent(MainActivity.this, DownloadImage.class);
startService(intent);
```

4.3.4 Binding Service to Activity

- Tạo một Bound Service dùng để liên kết với MainActivity:

```
public class LocalService extends Service {
    // Binder given to clients
    private final IBinder mBinder = new LocalBinder();
    // Random number generator
    private final Random mGenerator = new Random();

    /**
     * Class used for the client Binder. Because we know this service always
     * runs in the same process as its clients, we don't need to deal with IPC.
     */
    public class LocalBinder extends Binder {
        LocalService getService() {
            // Return this instance of LocalService so clients can call public methods
            return LocalService.this;
        }
    }

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

    /** method for clients */
    public int getRandomNumber() {
        return mGenerator.nextInt(100);
    }
}
```

- Trong file Android Manifest, đăng ký Service vừa tạo ra:

```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:theme="@style/AppTheme"
    android:label="12345">

    <service
        android:name="LocalService"
        android:label="@string/local_service">

</application>
```

- Trong file MainActivity.java, gọi phương thức “bindService” để tạo liên kết đến LocalService:

```
public class MainActivity extends Activity {
    LocalService mService;
    boolean mBound = false;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
    @Override
    protected void onStart() {
        super.onStart();
        // Bind to LocalService
        Intent intent = new Intent(this, LocalService.class);
        bindService(intent, mConnection, Context.BIND_AUTO_CREATE);
    }
    @Override
    protected void onStop() {
        super.onStop();
        // Unbind from the service
        if (mBound) {
            unbindService(mConnection);
            mBound = false;
        }
    }
    /** Called when a button is clicked (the button in the layout file attaches to
    * this method with the android:onClick attribute) */
    public void onClick(View v) {
        if (mBound) {
            // Call a method from the LocalService.
            // However, if this call were something that might hang, then this request should
            // occur in a separate thread to avoid slowing down the activity performance.
            int num = mService.getRandomNumber();
            Toast.makeText(this, "number: " + num,
                Toast.LENGTH_SHORT).show();
        }
    }
    /** Defines callbacks for service binding, passed to bindService() */
    private ServiceConnection mConnection = new ServiceConnection() {
        @Override
        public void onServiceConnected(ComponentName className,
            IBinder service) {
            // We've bound to LocalService, cast the IBinder and
            // get LocalService instance
            LocalBinder binder = (LocalBinder) service;
            mService = binder.getService();
            mBound = true;
        }
        @Override
        public void onServiceDisconnected(ComponentName arg0) {
            mBound = false;
        }
    };
}
```

4.4 Broadcast Receiver

4.4.1 Định nghĩa

- Cho phép ứng dụng đăng ký một sự kiện nào đó với hệ thống hoặc với các ứng dụng khác.
- Được sử dụng để nhận các sự kiện hoặc yêu cầu do các ứng dụng khác hoặc hệ thống phát đi, sau đó thực thi theo yêu cầu đó.
- Có 2 hình thức phát-nhận:
 - o Không có thứ tự (Normal broadcast -> sendBroadcast): đây là hình thức hoạt động bất đồng bộ (Asynchronous), tất cả các receiver đã được đăng ký đều có thể nhận được những sự kiện hay yêu cầu tương ứng, và có thể xảy ra đồng thời.
 - o Có thứ tự (Ordered broadcast -> sendOrderedBroadcast) tùy thuộc vào độ ưu tiên khi đăng ký mà các receiver nhận được các sự kiện tương ứng. Sau khi một receiver nhận được sự kiện thì các receiver tiếp theo mới được tiếp nhận sự kiện.

4.4.2 Ứng dụng

- Được sử dụng trong trường hợp ứng dụng muốn lắng nghe tin nhắn hay yêu cầu được phát đi từ hệ thống hoặc ứng dụng khác. Ví dụ: tin nhắn đến, cuộc gọi đến, cập nhật hệ thống, thông báo kết quả download,...
- Hiện thực một Broadcast Receiver:
 - o Tạo một Broadcast Receiver bằng cách kế thừa từ lớp BroadcastReceiver của Android:

```
public class MyReceiver extends BroadcastReceiver {  
  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        Toast.makeText(context, "Intent Detected.",  
            Toast.LENGTH_LONG).show();  
    }  
}
```

- o Đăng ký Broadcast Receiver trong file AndroidManifest.xml:

```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:theme="@style/AppTheme"
    android:label="12345">

    <receiver android:name="MyReceiver">
        <intent-filter>
            <action
                android:name="android.intent.action.BOOT_COMPLETED">
            </action>
        </intent-filter>
    </receiver>
</application>
```


CHƯƠNG V: LƯU TRỮ DỮ LIỆU

5.1 Đọc ghi dữ liệu trên File

5.1.1 Tạo File

- Android hỗ trợ rất nhiều cách lưu dữ liệu, một trong số đó là ghi dữ liệu vào file và lưu vào bộ nhớ trong (Internal storage) hoặc bộ nhớ ngoài (External storage – Sdcard).
- Cách tạo file trong Android:

```
File file = new File("/sdcard/myfile.txt");
if (!file.exists()) {
    try {
        file.createNewFile();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

5.1.2 Truy cập File để đọc và ghi dữ liệu

- Ghi dữ liệu và lưu file:

```
try {
    File myFile = new File("/sdcard/myfile.txt");
    myFile.createNewFile();
    FileOutputStream fOut = new FileOutputStream(myFile);
    OutputStreamWriter myOutWriter =
        new OutputStreamWriter(fOut);
    myOutWriter.append(txtData.getText());
    myOutWriter.close();
    fOut.close();
    Toast.makeText(getBaseContext(), "Done writing SD 'myfile.txt'",
        Toast.LENGTH_SHORT).show();
} catch (Exception e) {
    Toast.makeText(getBaseContext(), e.getMessage(),
        Toast.LENGTH_SHORT).show();
}
```

- Đọc dữ liệu từ file:

```
try {
    File myFile = new File("/sdcard/myfile.txt");
    FileInputStream fis = new FileInputStream(myFile);
    BufferedReader myReader =
        new BufferedReader(new InputStreamReader(fis));
    String aDataRow = "";
    String aBuffer = "";
    while ((aDataRow = myReader.readLine()) != null) {
        aBuffer += aDataRow + "\n";
    }
    txtData.setText(aBuffer);
    myReader.close();
    Toast.makeText(getApplicationContext(),
        "Done reading SD 'mysdfile.txt'",
        Toast.LENGTH_SHORT).show();
} catch (Exception e) {
    Toast.makeText(getApplicationContext(), e.getMessage(),
        Toast.LENGTH_SHORT).show();
}
```

5.2 Cơ sở dữ liệu trong Android

5.2.1 Giới thiệu SQLite

- Là cơ sở dữ liệu SQL mã nguồn mở được phát triển để sử dụng trên các thiết bị di động, trong đó có Android.
- Hỗ trợ tất cả các tính năng vốn có của một công cụ quản lý dữ liệu: thêm, bớt, xóa, chỉnh sửa,...

5.2.2 Xây dựng, kết nối CSDL với SQLite trong Android

- Tạo CSDL trong Android:

```
public class DBHelper extends SQLiteOpenHelper {
    public static final String DATABASE_NAME = "MyDBName.db";
    public static final String CONTACTS_TABLE_NAME = "contacts";
    public static final String CONTACTS_COLUMN_ID = "id";
    public static final String CONTACTS_COLUMN_NAME = "name";
    public static final String CONTACTS_COLUMN_EMAIL = "email";
    public static final String CONTACTS_COLUMN_STREET = "street";
    public static final String CONTACTS_COLUMN_CITY = "place";
    public static final String CONTACTS_COLUMN_PHONE = "phone";
    public DBHelper(Context context) {
        super(context, DATABASE_NAME, null, 1);
    }
    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("create table contacts " +
            "(id integer primary key, name text, phone text, email text, street text, place text)");
    }
}
```

- Hiện thực các thao tác với CSDL:

```
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE IF EXISTS contacts");
    onCreate(db);
}

public boolean insertContact (String name, String phone,
    String email, String street,String place) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("name", name);
    contentValues.put("phone", phone);
    contentValues.put("email", email);
    contentValues.put("street", street);
    contentValues.put("place", place);
    db.insert("contacts", null, contentValues);
    return true;
}

public Cursor getData(int id) {
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor res = db.rawQuery( "select * from contacts where id="+id+"", null );
    return res;
}

public int numberOfRows() {
    SQLiteDatabase db = this.getReadableDatabase();
    int numRows =
        (int) DatabaseUtils.queryNumEntries(db, CONTACTS_TABLE_NAME);
    return numRows;
}

public boolean updateContact (Integer id, String name,
    String phone, String email, String street,String place) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("name", name);
    contentValues.put("phone", phone);
    contentValues.put("email", email);
    contentValues.put("street", street);
    contentValues.put("place", place);
    db.update("contacts", contentValues, "id = ? ", new String[] { Integer.toString(id) } );
    return true;
}

public Integer deleteContact (Integer id) {
    SQLiteDatabase db = this.getWritableDatabase();
    return db.delete("contacts", "id = ? ", new String[] { Integer.toString(id) });
}
```

- Kết nối với CSDL để lấy dữ liệu trong MainActivity.java:

```
public class MainActivity extends Activity {  
  
    private DBHelper mydb ;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.main_activity);  
  
        mydb = new DBHelper(this);  
        Cursor rs = mydb.getData(Value);  
        rs.moveToFirst();  
        String nam = rs.getString(rs.getColumnIndex(  
            DBHelper.CONTACTS_COLUMN_NAME));  
        String phon = rs.getString(rs.getColumnIndex(  
            DBHelper.CONTACTS_COLUMN_PHONE));  
        String emai = rs.getString(rs.getColumnIndex(  
            DBHelper.CONTACTS_COLUMN_EMAIL));  
        String stree = rs.getString(rs.getColumnIndex(  
            DBHelper.CONTACTS_COLUMN_STREET));  
        String plac = rs.getString(rs.getColumnIndex(  
            DBHelper.CONTACTS_COLUMN_CITY));  
        if (!rs.isClosed()) {  
            rs.close();  
        }  
    }  
}
```

5.3 Content Provider

5.3.1 Định nghĩa

- Là một trong những thành phần chính trong ứng dụng Android, Content Provider cung cấp cơ chế truy xuất dữ liệu giữa các ứng dụng Android.
- Có nhiều cách để lưu trữ dữ liệu trong Android như: ghi file, cơ sở dữ liệu, hay lưu dữ liệu trên Internet... Và Content Provider giúp cho những ứng dụng khác nhau có thể truy xuất vào chung một nguồn dữ liệu đó.

5.3.2 Ứng dụng

- Tạo một Content Provider bằng cách kế thừa từ lớp ContentProvider trong Android:

```
public class ExampleProvider extends ContentProvider {
    /*
     * Defines a handle to the database helper object.
     * The MainDatabaseHelper class is defined in a following snippet.
     */
    private MainDatabaseHelper mOpenHelper;
    // Defines the database name
    private static final String DBNAME = "mydb";
    // Holds the database object
    private SQLiteDatabase db;

    public boolean onCreate() {
        /*
         * Creates a new helper object. This method always returns quickly.
         * Notice that the database itself isn't created or opened
         * until SQLiteOpenHelper.getWritableDatabase is called
         */
        mOpenHelper = new MainDatabaseHelper(
            getContext(),    // the application context
            DBNAME,          // the name of the database
            null,            // uses the default SQLite cursor
            1                // the version number
        );

        return true;
    }

    // Implements the provider's insert method
    public Cursor insert(Uri uri, ContentValues values) {
        /*
         * Gets a writeable database. This will trigger its creation if it doesn't already exist.
         */
        db = mOpenHelper.getWritableDatabase();
    }
}
```

○ Đăng ký Content Provider trong AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.mycontentprovider"
    android:versionCode="1"
    android:versionName="1.0" >
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <provider android:name="ExampleProvider"
            android:authorities="com.example.provider.ExampleProvider">
        </provider>
    </application>
</manifest>
```

- Sử dụng đối tượng Content Provider bằng cách gọi phương thức `getContentResolver()`:

```
/*
 * This defines a one-element String array to contain the selection argument.
 */
String[] mSelectionArgs = {""};
// Gets a word from the UI
mSearchString = mSearchWord.getText().toString();
// Remember to insert code here to check for invalid or malicious input.
// If the word is the empty string, gets everything
if (TextUtils.isEmpty(mSearchString)) {
    // Setting the selection clause to null will return all words
    mSelectionClause = null;
    mSelectionArgs[0] = "";
} else {
    // Constructs a selection clause that matches the word that the user entered.
    mSelectionClause = UserDictionary.Words.WORD + " = ?";

    // Moves the user's input string to the selection arguments.
    mSelectionArgs[0] = mSearchString;
}

// Does a query against the table and returns a Cursor object
mCursor = getContentResolver().query(
    UserDictionary.Words.CONTENT_URI, // The content URI of the words table
    mProjection,                      // The columns to return for each row
    mSelectionClause                  // Either null, or the word the user entered
    mSelectionArgs,                  // Either empty, or the string the user entered
    mSortOrder);                    // The sort order for the returned rows

// Some providers return null if an error occurs, others throw an exception
if (null == mCursor) {
    /*
     * Insert code here to handle the error. Be sure not to use the cursor! You may want to
     * call android.util.Log.e() to log this error.
     */
    // If the Cursor is empty, the provider found no matches
} else if (mCursor.getCount() < 1) {

    /*
     * Insert code here to notify the user that the search was unsuccessful.
     * This isn't necessarily an error.
     * You may want to offer the user the option to insert a new row, or re-type the
     * search term.
     */
} else {
    // Insert code here to do something with the results
}
```

CHƯƠNG VI: MULTIMEDIA

6.1 Audio

6.1.1 Audio formats

Format/Codec	Supported File Types/Container Formats
AAC LC	<ul style="list-style-type: none"> • 3GPP (.3gp) • MPEG-4 (.mp4, .m4a) • ADTS raw AAC (.aac, decode in Android 3.1+, encode in Android 4.0+, ADIF not supported) • MPEG-TS (.ts, not seekable, Android 3.0+)
HE-AACv1 (AAC+)	
HE-AACv2 (enhanced AAC+)	
AAC ELD (enhanced low delay AAC)	
AMR-NB	3GPP (.3gp)
AMR-WB	3GPP (.3gp)
FLAC	FLAC (.flac) only
MP3	MP3 (.mp3)
MIDI	<ul style="list-style-type: none"> • Type 0 and 1 (.mid, .xmf, .mxmf) • RTTTL/RTX (.rtttl, .rtx) • OTA (.ota) • iMelody (.imy)
Vorbis	<ul style="list-style-type: none"> • Ogg (.ogg) • Matroska (.mkv, Android 4.0+)
PCM/WAVE	WAVE (.wav)

6.1.2 Audio Playback

- Tạo Layout Audio Playback trong file res/main_activity.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <ImageButton
        android:id="@+id/imageButton3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_marginBottom="14dp"
        android:onClick="forward"
        android:src="@android:drawable/ic_media_ff" />

    <ImageButton
        android:id="@+id/imageButton4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignTop="@+id/imageButton2"
        android:layout_marginLeft="22dp"
        android:layout_toRightOf="@+id/imageButton2"
        android:onClick="rewind"
        android:src="@android:drawable/ic_media_rew" />

    <ImageButton
        android:id="@+id/imageButton2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/imageButton1"
        android:layout_marginLeft="14dp"
        android:layout_toRightOf="@+id/imageButton1"
        android:onClick="pause"
        android:src="@android:drawable/ic_media_pause" />

    <ImageButton
        android:id="@+id/imageButton1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/imageButton3"
        android:layout_marginLeft="24dp"
        android:layout_toRightOf="@+id/imageButton3"
        android:onClick="play"
        android:src="@android:drawable/ic_media_play" />

    <SeekBar
        android:id="@+id/seekBar1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_above="@+id/imageButton3"
        android:layout_toLeftOf="@+id/textView2"
        android:layout_toRightOf="@+id/textView1" />

</RelativeLayout>
```


- Hiện thực các tác vụ Audio Playback trong MainActivity.java:

```
public class MainActivity extends Activity {
    private MediaPlayer mediaPlayer;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main_activity);
        mediaPlayer = MediaPlayer.create(this, R.raw.song);
    }
    public void play(View view) {
        Toast.makeText(getApplicationContext(), "Playing sound",
            Toast.LENGTH_SHORT).show();
        mediaPlayer.start();
        if(oneTimeOnly == 0){
            seekbar.setMax((int) finalTime);
            oneTimeOnly = 1;
        }
        seekbar.setProgress((int)startTime);
        myHandler.postDelayed(UpdateSongTime,100);
        pauseButton.setEnabled(true);
        playButton.setEnabled(false);
    }
    public void pause(View view) {
        Toast.makeText(getApplicationContext(), "Pausing sound",
            Toast.LENGTH_SHORT).show();
        mediaPlayer.pause();
        pauseButton.setEnabled(false);
        playButton.setEnabled(true);
    }
    public void forward(View view){
        int temp = (int)startTime;
        if ((temp+forwardTime)<=finalTime) {
            startTime = startTime + forwardTime;
            mediaPlayer.seekTo((int) startTime);
        } else {
            Toast.makeText(getApplicationContext(),
                "Cannot jump forward 5 seconds",
                Toast.LENGTH_SHORT).show();
        }
    }
    public void rewind(View view) {
        int temp = (int) startTime;
        if ((temp-backwardTime)>0) {
            startTime = startTime - backwardTime;
            mediaPlayer.seekTo((int) startTime);
        } else {
            Toast.makeText(getApplicationContext(),
                "Cannot jump backward 5 seconds",
                Toast.LENGTH_SHORT).show();
        }
    }
}
```

6.2 Video

6.2.1 Video formats

Format/Codec	Supported File Types/Container Formats
H.263	<ul style="list-style-type: none"> • 3GPP (.3gp) • MPEG-4 (.mp4)
H.264 AVC	<ul style="list-style-type: none"> • 3GPP (.3gp) • MPEG-4 (.mp4) • MPEG-TS (.ts, AAC audio only, not seekable, Android 3.0+)
MPEG-4 SP	3GPP (.3gp)
VP8	<ul style="list-style-type: none"> • WebM (.webm) • Matroska (.mkv, Android 4.0+)

6.2.2 Video playback

- Tạo Layout cho Video playback trong file main_activity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello"/>

    <Button
        android:id="@+id/playvideoplayer"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="- PLAY Video -"/>

    <VideoView
        android:id="@+id/videoview"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

- Hiện thực các tác vụ Video Playback trong MainActivity.java:

```
//Implement SurfaceHolder interface to Play video
//Implement this interface to receive information about changes to the surface
public class AndroidVideoPlayer extends Activity implements SurfaceHolder.Callback {
    MediaPlayer mediaPlayer;
    SurfaceView surfaceView;
    SurfaceHolder surfaceHolder;
    boolean pausing = false;

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button buttonPlayVideo = (Button)findViewById(R.id.playvideoplayer);
        getWindow().setFormat(PixelFormat.UNKNOWN);
        //Displays a video file.
        VideoView mVideoView = (VideoView)findViewById(R.id.videoview)
        String uriPath =
            "android.resource://com.android.AndroidVideoPlayer/"+R.raw.k;
        Uri uri = Uri.parse(uriPath);
        mVideoView.setVideoURI(uri);
        mVideoView.requestFocus();
        mVideoView.start();
        buttonPlayVideo.setOnClickListener(new Button.OnClickListener() {
            @Override
            public void onClick(View v) {
                // VideoView reference see main.xml
                VideoView mVideoView =
                    (VideoView)findViewById(R.id.videoview);
                String uriPath =
                    "android.resource://com.android.AndroidVideoPlayer/"+R.raw.k;
                Uri uri = Uri.parse(uriPath);
                mVideoView.setVideoURI(uri);
                mVideoView.requestFocus();
                mVideoView.start();
            }
        });
    }
    @Override
    public void surfaceChanged(SurfaceHolder holder, int format, int width,
        int height) {
        // TODO Auto-generated method stub
    }
    @Override
    public void surfaceCreated(SurfaceHolder holder) {
        // TODO Auto-generated method stub
    }
    @Override
    public void surfaceDestroyed(SurfaceHolder holder) {
        // TODO Auto-generated method stub
    }
}
```

6.3 Camera

6.3.1 Giới thiệu

Là thành phần của Android được sử dụng để phát triển những ứng dụng có khả năng chụp hình hoặc quay video trên các thiết bị chạy hệ điều hành Android.

6.3.2 Ứng dụng

- Hiện thực Camera Layout trong file main_activity.xml:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity">

    <ImageView
        android:id="@+id/imageView1"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginLeft="34dp"
        android:layout_marginTop="36dp"
        android:contentDescription="@string/hello_world"
        android:src="@drawable/ic_launcher" />

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignRight="@+id/imageView1"
        android:text="@string/tap"
        android:textAppearance="?android:attr/textAppearanceLarge" />

</RelativeLayout>
```

- Hiện thực cơ chế chụp hình trong MainActivity.java

```
public class MainActivity extends Activity {

    ImageView imgFavorite;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        imgFavorite = (ImageView)findViewById(R.id.imageView1);
        imgFavorite.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                open();
            }
        });
    }

    public void open() {
        Intent intent =
            new Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
        startActivityForResult(intent, 0);
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        // TODO Auto-generated method stub
        super.onActivityResult(requestCode, resultCode, data);
        Bitmap bp = (Bitmap) data.getExtras().get("data");
        imgFavorite.setImageBitmap(bp);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

CHƯƠNG VII: WEB SERVICE VÀ WEBVIEW

7.1 Web Service

7.1.1 Định nghĩa

Web Service là một chuẩn được sử dụng để trao đổi thông tin dữ liệu giữa các ứng dụng, hoặc giữa các hệ thống với nhau thông qua giao thức HTTP của mạng Internet.

7.1.2 Ứng dụng

- Trong Android, Web Service thường được dùng để phát triển các ứng dụng có khả năng kết nối với một Server để trao đổi dữ liệu.
- Muốn hiện thực một ứng dụng Web Service trên Android, điều cần thiết là sử dụng một thư viện hỗ trợ giao thức SOAP (Simple Object Access Protocol) là giao thức được phát triển dựa trên giao thức HTTP: **ksoap2-android**.
- Ví dụ: xây dựng một ứng dụng Web Service để chuyển đổi đơn vị nhiệt độ từ Celsius sang Fahrenheit và ngược lại. Quá trình chuyển đổi sẽ do Server thực hiện rồi trả về kết quả cho ứng dụng:

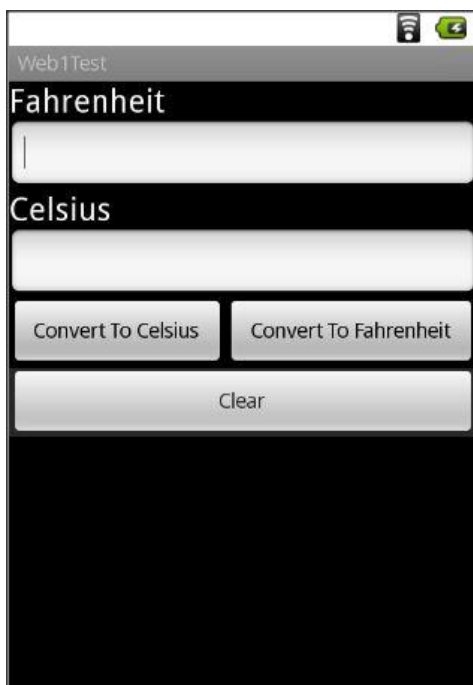


Figure 28: Ví dụ về Web Service

- Tạo layout cho ứng dụng:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Fahrenheit"
        android:textAppearance="?android:attr/textAppearanceLarge" />
    <EditText
        android:id="@+id/txtFar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
        <requestFocus />
    </EditText>
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Celsius"
        android:textAppearance="?android:attr/textAppearanceLarge" />
    <EditText
        android:id="@+id/txtCel"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <LinearLayout
        android:id="@+id/linearLayout1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
        <Button
            android:id="@+id/btnFar"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.5"
            android:text="Convert To Celsius" />
        <Button
            android:id="@+id/btnCel"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.5"
            android:text="Convert To Fahrenheit" />
    </LinearLayout>
    <Button
        android:id="@+id/btnClear"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Clear" />
</LinearLayout>
```

- Sử dụng một Web Service để kết nối với ứng dụng:

<http://www.w3schools.com/webservices/tempconvert.asmx>

<http://www.w3schools.com/webservices/tempconvert.asmx?WSDL>

- Web Service này hỗ trợ 2 chức năng chính:

- Chuyển đổi từ Celsius sang Fahrenheit:

- SOAP_ACTION = "<http://tempuri.org/CelsiusToFahrenheit>"
- NAMESPACE = "<http://tempuri.org/>"
- METHOD_NAME = "CelsiusToFahrenheit"
- URL = "<http://www.w3schools.com/webservices/tempconvert.asmx?WSDL>"

- Chuyển đổi từ Fahrenheit sang Celsius:

- SOAP_ACTION = "<http://tempuri.org/FahrenheitToCelsius>"
- NAMESPACE = "<http://tempuri.org/>"
- METHOD_NAME = "FahrenheitToCelsius"
- URL = "<http://www.w3schools.com/webservices/tempconvert.asmx?WSDL>"

- Hiện thực lớp WebServiceActivity.java:

- Khai báo các thông số của WebService:

```
import org.ksoap2.SoapEnvelope;
import org.ksoap2.serialization.SoapObject;
import org.ksoap2.serialization.SoapSerializationEnvelope;
import org.ksoap2.transport.HttpTransportSE;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
public class WebServiceActivity extends Activity {
    /** Called when the activity is first created. */
    private static String SOAP_ACTION1 = "http://tempuri.org/FahrenheitToCelsius";
    private static String SOAP_ACTION2 = "http://tempuri.org/CelsiusToFahrenheit";
    private static String NAMESPACE = "http://tempuri.org/";
    private static String METHOD_NAME1 = "FahrenheitToCelsius";
    private static String METHOD_NAME2 = "CelsiusToFahrenheit";
    private static String URL =
        "http://www.w3schools.com/webservices/tempconvert.asmx?WSDL";
```


- Khai báo các biến được sử dụng trong Layout:

```
Button btnFar,btnCel,btnClear;  
EditText txtFar,txtCel;  
@Override  
public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.main);  
    btnFar = (Button)findViewById(R.id.btnFar);  
    btnCel = (Button)findViewById(R.id.btnCel);  
    btnClear = (Button)findViewById(R.id.btnClear);  
    txtFar = (EditText)findViewById(R.id.txtFar);  
    txtCel = (EditText)findViewById(R.id.txtCel);  
}
```

- Hiện thực các sự kiện cho các Button trong Layout:

```
btnFar.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        //Initialize soap request + add parameters  
        SoapObject request =  
            new SoapObject(NAMESPACE, METHOD_NAME1);  
        //Use this to add parameters  
        request.addProperty("Fahrenheit",txtFar.getText().toString());  
        //Declare the version of the SOAP request  
        SoapSerializationEnvelope envelope =  
            new SoapSerializationEnvelope(SoapEnvelope.VER11);  
        envelope.setOutputSoapObject(request);  
        envelope.dotNet = true;  
        try {  
            HttpTransportSE androidHttpTransport = new HttpTransportSE(URL);  
            //this is the actual part that will call the webservice  
            androidHttpTransport.call(SOAP_ACTION1, envelope);  
            // Get the SoapResult from the envelope body.  
            SoapObject result = (SoapObject)envelope.bodyIn;  
            if (result != null) {  
                //Get the first property and change the label text  
                txtCel.setText(result.getProperty(0).toString());  
            } else {  
                Toast.makeText(getApplicationContext(),  
                    "No Response",Toast.LENGTH_LONG).show();  
            }  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
});  
btnClear.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        txtCel.setText("");  
        txtFar.setText("");  
    }  
});  
}
```

- Khai báo quyền truy cập Internet trong file AndroidManifest.xml

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

7.1.3 Xử lý dữ liệu dạng JSON và XML

- JSON: là 1 dạng dữ liệu có cấu trúc gọn, nhẹ, dễ đọc, dễ parse được sử dụng trong Web Service.

- Ví dụ dữ liệu dạng JSON:

```
{
  "contacts": [
    {
      "id": "1",
      "name": "07tHC",
      "email": "07htc@gmail.com",
      "address": "xx-xx-xxxx,x - street, x - country",
      "gender": "male",
      "phone": {
        "mobile": "+84 0000000000",
        "home": "00 000000",
        "office": "00 000000"
      }
    }
  ]
}
```

- Hiện thực xử lý JSON:

- Khai báo các thành phần trong cấu trúc JSON:

```
// JSON Node names
private static final String TAG_CONTACTS = "contacts";
private static final String TAG_ID = "id";
private static final String TAG_NAME = "name";
private static final String TAG_EMAIL = "email";
private static final String TAG_ADDRESS = "address";
private static final String TAG_GENDER = "gender";
private static final String TAG_PHONE = "phone";
private static final String TAG_PHONE_MOBILE = "mobile";
private static final String TAG_PHONE_HOME = "home";
private static final String TAG_PHONE_OFFICE = "office";

// contacts JSONArray
JSONArray contacts = null;
```

- Lấy dữ liệu từ JSON:

```
try {
    JSONObject jsonObj = new JSONObject(jsonStr);
    // Getting JSON Array node
    contacts = jsonObj.getJSONArray(TAG_CONTACTS);
    // looping through All Contacts
    for (int i = 0; i < contacts.length(); i++) {
        JSONObject c = contacts.getJSONObject(i);
        String id = c.getString(TAG_ID);
        String name = c.getString(TAG_NAME);
        String email = c.getString(TAG_EMAIL);
        String address = c.getString(TAG_ADDRESS);
        String gender = c.getString(TAG_GENDER);
        // Phone node is JSON Object
        JSONObject phone = c.getJSONObject(TAG_PHONE);
        String mobile = phone.getString(TAG_PHONE_MOBILE);
        String home = phone.getString(TAG_PHONE_HOME);
        String office = phone.getString(TAG_PHONE_OFFICE);
        // tmp hashmap for single contact
        HashMap<String, String> contact = new HashMap<String, String>();
        // adding each child node to HashMap key => value
        contact.put(TAG_ID, id);
        contact.put(TAG_NAME, name);
        contact.put(TAG_EMAIL, email);
        contact.put(TAG_PHONE_MOBILE, mobile);
        // adding contact to contact list
        contactList.add(contact);
    }
} catch (JSONException e) {
    e.printStackTrace();
}
```

- XML: là một định dạng phổ biến được sử dụng trong việc chia sẻ dữ liệu trên Internet.

- Ví dụ dữ liệu dạng XML:

```
<?xml version="1.0"?>
<current>
  <city id="1" name="HCM">
    <coord lon="-0.12345" lat="54.321"/>
    <country>Vietnam</country>
    <sun rise="2014-11-15Sat:07:30" set="2014-11-15Sat:07:30"/>
  </city>
  <temperature value="35" min="31" max="38" unit="celsius"/>
  <humidity value="84" unit="%">
  <pressure value="1010" unit="hPa"/>
</country>
```

○ Xử lý dữ liệu XML:

```
public void parseXMLAndStoreIt(XmlPullParser myParser) {
    int event;
    String text=null;
    try {
        event = myParser.getEventType();
        while (event != XmlPullParser.END_DOCUMENT) {
            String name=myParser.getName();
            switch (event){
                case XmlPullParser.START_TAG:
                    break;
                case XmlPullParser.TEXT:
                    text = myParser.getText();
                    break;
                case XmlPullParser.END_TAG:
                    if (name.equals("country")){
                        country = text;
                    } else if(name.equals("humidity")) {
                        humidity = myParser.getAttributeValue(null,"value");
                    } else if(name.equals("pressure")) {
                        pressure = myParser.getAttributeValue(null,"value");
                    } else if(name.equals("temperature")) {
                        temperature = myParser.getAttributeValue(null,"value");
                    }
                    break;
            }
            event = myParser.next();
        }
        parsingComplete = false;
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public void fetchXML() {
    try {
        URL url = new URL(urlString);
        HttpURLConnection conn = (HttpURLConnection)
            url.openConnection();
        conn.setReadTimeout(10000 /* milliseconds */);
        conn.setConnectTimeout(15000 /* milliseconds */);
        conn.setRequestMethod("GET");
        conn.setDoInput(true);
        conn.connect();
        InputStream stream = conn.getInputStream();
        xmlFactoryObject = XmlPullParserFactory.newInstance();
        XmlPullParser myparser = xmlFactoryObject.newPullParser();
        myparser.setFeature(XmlPullParser.FEATURE_PROCESS_NAMESPACES, false);
        myparser.setInput(stream, null);
        parseXMLAndStoreIt(myparser);
        stream.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

7.2 WebView

7.2.1 Định nghĩa

Là một thành phần giao diện trong Android được sử dụng để hiển thị các trang Web. Người sử dụng có thể thực hiện những thao tác trên WebView tương tự như một trình duyệt Web thông thường.

7.2.2 Ứng dụng

- Phát triển các ứng dụng hiển thị các trang Web, hoặc các trang có định dạng Html, Xml...
- Ví dụ về cách hiện thực một WebView:
 - Xây dựng WebView trong file layout/main_activity.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <WebView
        android:id="@+id/webview"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

- Hiển thị trang Web trong WebView:

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

    WebView webView = (WebView) findViewById(R.id.webview);
    webView.getSettings().setJavaScriptEnabled(true);
    webView.loadUrl("http://www.google.com");
}
```

THAM KHẢO

1. Android Developers Training Online:

<http://developer.android.com/training/index.html>

2. Android Programming Tutorial:

<http://www.tutorialspoint.com/android/index.htm>

3. Android Development Starter Tutorial:

<http://www.vogella.com/tutorials/android.html>