1.Vòng đời của Activity



* **onCreate(Bundle savedInstanceState)**:

***Callback*** này được gọi khi Activity được kích hoạt, thậm chí người dùng chưa thấy gì của Activity thì ***callback*** này đã được gọi rồi. ***Callback*** này chỉ được gọi một lần duy nhất khi ***Activity*** được khởi tạo. Sau ***callback*** này bao giờ cũng gọi ngay lập tức ***onStart().***

Ở phương thức này chúng ta thường sử dùng để ***setContentView()***, khởi tạo Adapter cho ***ListView***, ***RecyclerView***…

* **onStart()**:

Sau khi gọi đến ***onCreate()***, hệ thống sẽ gọi đến ***onStart()***. Hoặc hệ thống cũng sẽ gọi lại ***onStart()*** sau khi gọi ***onRestart()*** nếu trước đó nó bị che khuất bởi Activity nào khác (một màn hình khác hoặc một ứng dụng khác) ***che hoàn toàn*** và rơi vào ***onStop()***. Khi hệ thống gọi đến ***callback*** này thì chúng ta chưa nhìn thấy được ***Activity*** và tương tác được với ***Activity***

* **onResume()**:

***Callback*** này được gọi ngay khi Activity bắt đầu có thể tương tác với người dùng, tức là sau khi ***onStart()*** được gọi, và Activity nằm trên cùng trong danh sách các Activity của hệ thống. Sau phương thức này là các chức năng của Activity hoạt động dựa trên tương tác của người dùng ..., cho đến khi có một nguyên nhân nào đó mà phương thức ***onPause()*** được gọi.

* **onPause()**:

Được gọi khi hệ thống sắp kích hoạt một Activity khác, thường để lưu lại trạng thái của các biến trước khi hệ thống kích hoạt Activity khác. Ngay sau phương thức này nó sẽ gọi ***onResume()*** nếu Activity được kích hoạt lại ngay, hoặc gọi ***onStop()*** nếu Activity bị ẩn đi.

* **onStop()**:

Được gọi khi Activity bị ẩn đi. Sau phương thức này, Activity có thể gọi ***onRestart()*** nếu nó được người dùng kích hoạt lại hoặc gọi ***onDestroy()*** để hết thúc.

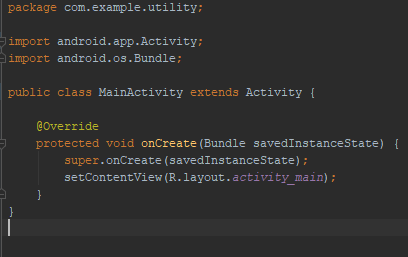
* **onDestroy()**:

Gọi khi Activity bị hủy hoàn toàn (ví dụ gọi finish(), hoặc người dùng kill Activity)

2. Làm thế nào để ánh xạ một View từ layout vào trong môi trường Activity

Khi khởi tạo đối tượng Activity, sẽ gọi đến phương thức ***onCreate(Bundle)*** đầu tiên. => Trong phương thức này, xem xét xem view cần ánh xạ đang thuộc layout xml nào, sau đó gọi phương thức:

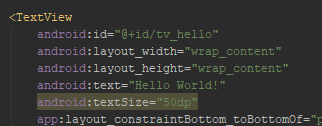
***setContentView(R.layout.ten-file-xml)***



1. Phương thức findViewById(int id) dùng để làm gì?

Khi muốn set cho 1 view nào đó một đặc điểm hoặc 1 action để tương tác lại với người dùng, thì findViewById(int id) là phương thức để gọi tới view cần thao tác.

1. Ví dụ: trong layout activity\_main.xml đang có 1 view với id là tv\_hello như sau



Khi đó, nếu muốn ánh xạ view này vào trong Activity, ta làm 2 bước:

* setContentView(): để khai báo layout chứa nó.
* findViewById(): để khai báo chính view đó thông qua id của nó, việc gọi đến view này nhằm mục đích set cho nó 1 thuộc tính ví dụ click được:

3. Làm thế nào để bắt sự kiện click cho một View

**- Bắt sự kiện bằng Onclick trong XML:**

Đặc điểm: Phổ biến nhưng không tối ưu

+ Khai báo thuộc tính <android: onClick=""> trong file XML

+ Vào MainActivity để viết hàm bắt sự kiện

**- Bắt sự kiện bằng Activity is listener**

Đặc điểm: + Tiện cho việc quản lý code

+ implements interface của sự kiện => Dễ dàng chọn sự kiện phù hợp

+ Hàm OnClickListener (Override hàm onClick): Hàm này được gọi khi view được click

Ví dụ: Click button Login trên view

=> Sử dụng phương thức để lắng nghe sự kiện (Khi button được click vào)

**- Toast**

+ Toast là gì? => Sử dụng Toast để kiểm tra một điều gì đó, hay đơn giản là xuất thông báo.

Đều hiển thị thông báo lên khi các tiến trình khác vẫn đang làm việc

+ Toast có thể tạo và hiển thị trong Activity hoặc Service

+ Không cho phép người dùng tương tác

+ Hiển thị lên sau 1 thời gian sẽ tự đóng lại

+ Hai loại giá trị mặc định cho thời gian hiển thị của Toast:

- Toast.LENGTH\_SHORT hiển thị trong 2 giây

- Toast.LENGTH\_LONG hiển thị trong 3.5 giây

+ Cách tạo Toast:

Toast myToast=Toast.makeText(YourActivity.this, “Hello”, Toast.LENGTH\_SHORT);

myToast.show();

Ví dụ: Bắt 1 sự kiện click có sử dụng Toast

Step1: Tạo button trong file activity\_main.XML

<Button

android:id="@+id/my\_button\_toast"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="My Button"

android:onClick="DisplayToast"

/>

Step2: Trong mainActivity enxtends ActionBarActivity

=> Khai báo cho button:

Button myButton;

=> Trong phương thức onCreat, ta ghi đè phương thức như sau

protected void onCreat(Bundle saveInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

myButton=(Button)findViewById(R.id.my\_button\_toast);

myButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Toast.makeText(MainActivity.this, "Hello", Toast.LENGTH\_SHORT,show();

}

}};

}

3. Intent là gì ?

- Intent (ý định) là những tin nhắn không đồng bộ cho phép các thành phần ứng dụng yêu cầu chức năng từ các thành phần Androidkhác.

Intents cho phép bạn tương tác với các thành phần từ các ứng dụng như nhau cũng như với các thành phần đóng gói trong một ứng dụng khác.

Một Intent có thể chứa dữ liệu thông qua một Bundle.

Android hỗ trợ 2 loại Intent là Intent tường minh (explicit) và Intent không tường minh (implicit).

- Intent tường minh (Explicit intents): Là những ý định (intent) chỉ định rõ ràng tên của các thành phần mục tiêu để xử lý; trong đó, trường mục tiêu (tùy chọn) được sét một giá trị cụ thể thông qua các phương thức setComponent() hoặc setClass().

- Intent không tường minh (Implicit Intents): Là những ý định (intent) không chỉ định rõ một mục tiêu thành phần, nhưng bao gồm đầy đủ thông tin cho hệ thống để xác định các thành phần có sẵn là tốt nhất để chạy cho mục đích đó.

**Code:**

// Tạo một Intent với mục tiêu là Activity2.

Intent intent = new Intent(this, Activity2.class);

// Các dữ liệu gửi kèm

intent.putExtra("firstName",firstName);

intent.putExtra("lastName", lastName);

// Yêu cầu start Activity và sẽ có phản hồi từ Activity được gọi.

this.startActivityForResult(intent, MY\_REQUEST\_CODE);