**HANDLE**

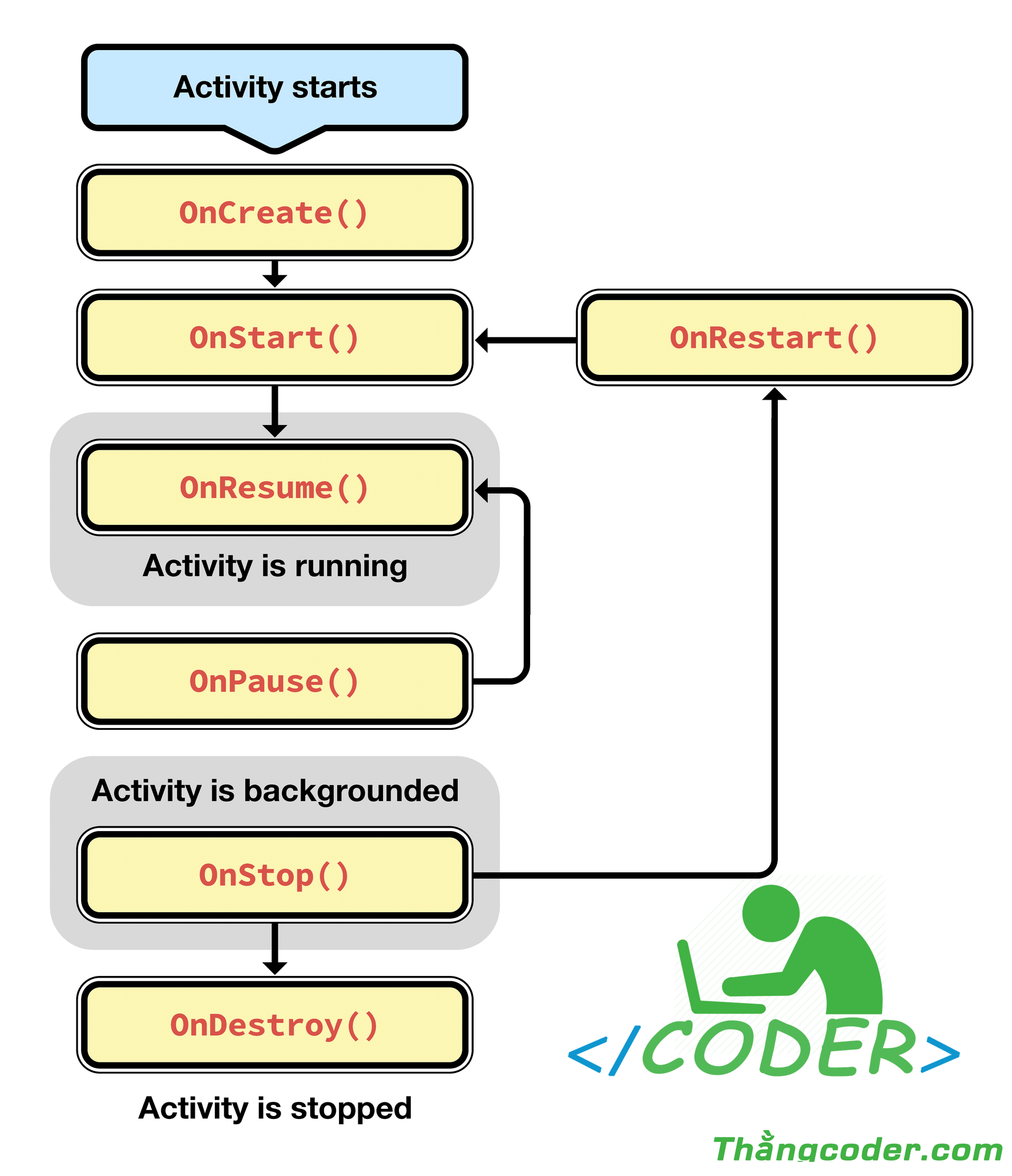
**CONFIGURATION CHANGES**

**Lý thuyết**

Một số cấu hình thiết bị có thể thay đổi trong thời gian chạy (chẳng hạn như hướng màn hình, sự sẵn có của bàn phím, và ngôn ngữ) thay đổi này có thể gọi là Handle Configuration Changes (Xử lý thay đổi cấu hình).

Thông thường, khi sự kiện xoay màn hình diễn ra, Android sẽ khởi tạo lại Activity (onDestroy sẽ được gọi, sau đó là **onCreate**). Hành vi khởi động lại được thiết kế để giúp ứng dụng điều chỉnh phù hợp với cấu hình mới của thiết bị. Tuy nhiên đôi khi việc này sẽ tạo nên trải nghiệm không tốt cho người dùng(ví dụ video đang xem sẽ bị tải lại, kết nối mạng bị hủy và khởi tạo lại tốn thời gian,...).

Để xử lý khởi động lại cho đúng, điều quan trọng là hoạt động của bạn khôi phục lại trạng thái trước đó của nó thông qua vòng đời [**Activity**](https://developer.android.com/guide/components/activities.html#Lifecycle)

,

trong đó Android sẽ gọi [onSaveInstanceState()](https://developer.android.com/reference/android/app/Activity.html#onSaveInstanceState(android.os.Bundle))trước khi nó hủy hoạt động của bạn sao cho bạn có thể lưu dữ liệu về trạng thái của ứng dụng. Khi đó, bạn có thể khôi phục trạng thái trong khi [**onCreate()**](https://developer.android.com/reference/android/app/Activity.html#onCreate(android.os.Bundle)) hoặc [**onRestoreInstanceState()**](https://developer.android.com/reference/android/app/Activity.html#onRestoreInstanceState(android.os.Bundle)).

Vì vậy dưới đây là một số cơ chế của Android để khắc phục các vấn đề này :

## Cấu hình cố định 1 hướng cho Activity

android:screenOrientation

Trong AndroidManifest.xml bạn cấu hình hướng cố định cho Activity của bạn bằng cờ **screenOrientation**. Ở đây mình chọn **portrait** tức màn hình Activity sẽ ở chế độ dọc và nó sẽ không thể bị thay đổi cho dù điện thoại của bạn xoay bất kì hướng nào.

<activity

android:name=".YourActivity"

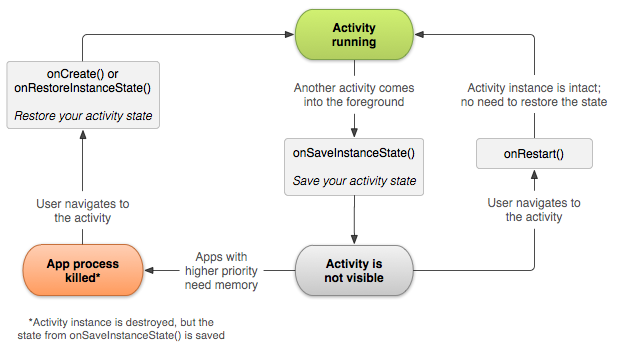
android:screenOrientation="portrait" />

Có lẽ đây cách này khá là nhanh chóng và dễ dàng khi xây dựng một số ứng dụng mà không quá cần thiết cả 2 chiều. Tuy nhiên với một lời khuyên nho nhỏ là khi ứng dụng của bạn có thể xoay cả dọc lẫn ngang thì khả năng sử dụng ứng dụng với người dùng cũng tốt hơn

## Lưu lại trạng thái đối tượng

onSaveInstanceState(Bundle outState)

Như mình đã nói phía trên, khi xoay màn hình Activity sẽ bị hủy và khởi tạo lại => và Android cho phép lưu lại các trạng thái của activity (các đối tượng) trước khi bị hủy và khôi phục lại chúng khi activity được khởi tạo lại.



Ở hình trên là 2 cách mà một Activity ở trạng thái không hiển thị được điều hướng trở lại với người dùng với trạng thái không đổi khi: Acitivity bị hủy (**App process killed**) và sau đó khôi phục lại trạng thái đã lưu trước đó; và khi Activity bị dừng (**onStop()**) rồi tiếp tục hiển thị trở lại với trạng thái giữ nguyên không đổi.

* Trước khi một Activity bị hủy (**onDestroy**) nó sẽ gọi **onSaveInstanceState** (gọi trước onStop và có thể trước cả onPause; không gọi trong tất cả các trường hợp, ví dụ khi người dùng bấm back). Hệ thống sẽ chuyển cho phương thức này một **Bundle**nhờ đó ta có thể lưu các thông tin trạng thái của Activity (các đối tượng).
* Sau đó nếu hệ thống hủy Activity, khi được khởi tạo lại hệ thống sẽ tạo Activity đó và chuyển **Bundle** kia cho **onCreate()** và **onRestoreInstance**. Sử dụng 1 trong 2 phương thức này bạn có thể lấy lại các đối tượng đã lưu từ **Bundle** và khôi phục lại trạng thái của Activity. Nếu không có trạng thái nào để khôi phục **Bundle = null**(Khởi tạo lần đầu).

Dưới đây là một ví dụ đơn giản lưu **STATE\_COUNTER** tại **onSaveInstanceState** và hiển thị ra khi Activity được tạo lại. Và dĩ nhiên, xoay màn hình để thấy điểu bất ngờ nhé

private static final String STATE\_COUNTER = "counter";

private TextView mCounterTextView;

private int mCounter;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

...

// Lấy ra STATE\_COUNTER nếu không có mặc định kết quả trả về là 0

if (savedInstanceState != null) {

mCounter = savedInstanceState.getInt(STATE\_COUNTER, 0);

}

// Hiển thị mCounter

mCounterTextView = findViewById(R.id.counter\_text);

mCounterTextView.setText(Integer.toString(mCounter));

...

}

@Override

protected void onSaveInstanceState(Bundle outState) {

super.onSaveInstanceState(outState);

// Lưu STATE\_COUNTER vào Bundle

outState.putInt(STATE\_COUNTER, mCounter);

}

## Sử dụng fragments

Fragment.setRetainInstance(true)

 Đây là phương pháp tốt nhất bạn nên sử dụng trong đa số các trường hợp.

Như bạn đã biết thì **Bundle không được thiết kế để mang các đối tượng lớn** (chẳng hạn như bitmap) do đó theo cách làm trên với việc khởi tạo lại Activity kéo theo khôi phục lại tập dữ liệu quá lớn có thể gây tiêu tốn bộ nhớ và trải nghiệm người dùng chậm chạp.

Với giải pháp sử dụng fragment sẽ giúp giảm bớt gánh nặng khởi tạo lại cho Activity, các fragment được đánh dấu để giữ lại sẽ không bị hủy. Và với các đối tượng có trạng thái cần giữ, bạn nên đặt chúng trong Fragment:

1. Extend **Fragment** và khai báo các đối tượng có trạng thái cần giữ lại.
2. **setRetainInstance(boolean)**.
3. **Thêm Fragment** vào **Activity**.
4. Sử dụng **FragmentManager** để **lấy lại Fragment** khi Activity được khởi tạo lại.

public class RetainedFragment extends Fragment {

// Đối tượng dữ liệu cần giữ lại

private MyDataObject data;

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

// Chỉ định giữ lại fragment

setRetainInstance(true);

}

public void setData(MyDataObject data) {

this.data = data;

}

public MyDataObject getData() {

return data;

}

}

Chú ý đối tượng cần lưu trữ để tránh **Memory leak** (khi nó được gắn với **Activity**, hoặc đi kèm **context**)

public class MyActivity extends Activity {

private RetainedFragment dataFragment;

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

// Lấy fragment đã được lưu trữ

FragmentManager fm = getFragmentManager();

dataFragment = (DataFragment) fm.findFragmentByTag(“data”);

// Khởi tạo Fragment và dữ liệu lần đầu

if (dataFragment == null) {

// Thêm Fragment vào Activity

dataFragment = new DataFragment();

fm.beginTransaction().add(dataFragment, “data”).commit();

// Load dữ liệu lần đầu

dataFragment.setData(loadMyData());

}

// Dễ dàng lấy lại các dữ liệu qua dataFragment.getData()

...

}

@Override

public void onDestroy() {

super.onDestroy();

// Nếu muốn cập nhật lại dữ liệu muốn giữ lại khi Activity bị hủy

dataFragment.setData(collectMyLoadedData());

}

}

Với cách làm này, fragment sẽ không gọi lại onCreate và onDestroy cũng như hàm khởi tạo vì fragment đã được giữ lại trước đó và chỉ lấy ra để sử dụng.

## Ngăn Activity hủy - tự xử lý các thay đổi cấu hình

android:configChanges

Bằng việc sử dụng cờ **android:configChanges**(khai báo các cấu hình muốn tự xử lý) cho Activity trong **Manifest** sẽ khai báo rằng **Activity** của bạn sẽ **tự động xử lý** thay đổi cấu hình và tránh cho hệ thống khởi động lại Activity của bạn. Thay vào đó Activity sẽ gọi tới **onConfigurationChanged()** và truyền vào đó là đối tượng **Configuration**chứa các cấu hình mới.

<activity android:name=".MyActivity"

android:configChanges="orientation|keyboardHidden"

android:label="@string/app\_name">

Trong ví dụ trên Activity sẽ không bị hủy khi hướng màn hình thay đổi.

Và bạn có thể dễ dàng cấu hình lại giao diện của mình trong **onConfigurationChanged**

@Override

public void onConfigurationChanged(Configuration newConfig) {

super.onConfigurationChanged(newConfig);

// Checks the orientation of the screen

if (newConfig.orientation == Configuration.ORIENTATION\_LANDSCAPE) {

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

} else if (newConfig.orientation == Configuration.ORIENTATION\_PORTRAIT){

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

}

}

Để hiểu thêm những cấu hình nào bạn có thể xử lý, hãy xem tài liệu [**android:configChanges**](https://developer.android.com/guide/topics/manifest/activity-element#config) và [Configuration](https://developer.android.com/reference/android/content/res/Configuration.html)

\*From: <https://viblo.asia/p/android-xu-ly-orientation-changes-63vKjaab52R?fbclid=IwAR3ZEVbm0-mGgz0MeorKNGux1G9LKi-mP7jFzT3oiuUlYQ9MlUPGNJ2qbRA>