

Bài: Vòng đời của Activity trong Android

Xem bài học trên website để ủng hộ Kteam: [Vòng đời của Activity trong Android](#)

Mọi vấn đề về lỗi website làm ảnh hưởng đến bạn hoặc thắc mắc, mong muốn khóa học mới, nhằm hỗ trợ cải thiện Website. Các bạn vui lòng phản hồi đến Fanpage [How Kteam](#) nhé!

Dẫn nhập

Ở bài [INTENT & MANIFEST](#), chúng ta đã có một cái nhìn tổng quan về các thành phần của một ứng dụng Android như Service, Broadcast Receiver, và không thể thiếu... Activity.

Về cơ bản, các bạn có thể hiểu Activity là một “màn hình” trong ứng dụng Android. Nhưng thật sự có phải chỉ như vậy? Và cái mà người ta hay nói là “**vòng đời**” của **activity** thật sự nó là gì?

Chúng ta sẽ cùng nhau tìm hiểu trong bài viết này nhé!

Nội dung

Để đọc hiểu bài này tốt nhất các bạn nên có kiến thức cơ bản về các phần:

- **Android Manifest** (xem lại bài trước – [INTENT & MANIFEST](#) để biết thêm chi tiết).
- Cách viết một ứng dụng Android cơ bản (xem lại bài [GIỚI THIỆU LẬP TRÌNH ANDROID](#) để biết thêm chi tiết).
- Một suy nghĩ đơn giản. Thật vậy đó, vì bản chất của Activity khá đơn giản, các bạn không nên chỉ trông mấy cái sơ đồ mà thấy sợ.

Trong bài học này, chúng ta sẽ cùng tìm hiểu các vấn đề:

- **Activity** là gì.
- Vòng đời của một Activity.
- Các trạng thái của một Activity.
- Tùy chỉnh trạng thái của Activity.
- Đố vui... không có thưởng (nhưng có lời giải) về Activity.

Biên niên sử về vòng đời ứng dụng

Trong trường hợp lý tưởng nhất có thể, tất cả các ứng dụng Android sau khi chạy đều được lưu lại trong bộ nhớ (cụ thể là RAM), để sau này ứng dụng có thể khởi động lại nhanh hơn.

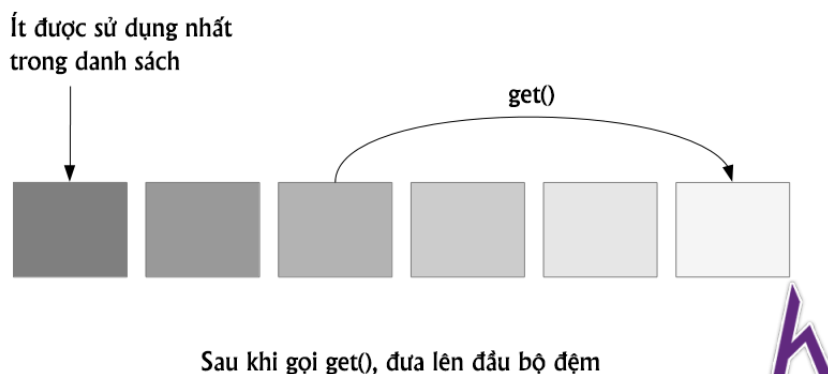
Tuy nhiên trên thực tế, các thiết bị di động có tài nguyên phần cứng khá hạn chế, về cả RAM, tốc độ xử lý,... và vì vậy ứng dụng nào ít được sử dụng sẽ bị hủy tiến trình chạy nhằm thu hồi bộ nhớ.

Quy tắc thu hồi bộ nhớ của Android rất đơn giản: Các tiến trình ứng dụng sẽ được thu hồi theo một thứ tự ưu tiên nhất định. Và thứ tự ưu tiên đó là:

Trạng thái	Mô tả	Mức độ ưu tiên
Foreground (mặt tiền)	Là ứng dụng đang mở, và người dùng đang thao tác với Activity, hoặc có một Service đang gắn kèm với Activity.	1
Visible (hiện hữu)	Người dùng có thể thấy ứng dụng, nhưng không tương tác với nó, hoặc Activity mà người dùng thấy chỉ hiển thị một phần. Một trường hợp khác là có một Service đang đính kèm với một Activity đang không được sử dụng nhưng có thể nhìn thấy được.	2
Service	Ứng dụng đang chạy một Service, nhưng không thỏa mãn thứ tự ưu tiên 1 hoặc 2 ở trên.	3
Background	Ứng dụng đang trong trạng thái "Dừng" (stopped), không có Service hay Broadcast Receiver nào đang chạy. Android sẽ lưu ứng dụng trong danh sách LRU (Least-Recently Used) và triệt tiêu khi được yêu cầu.	4
Rỗng	Ứng dụng không có thành phần nào đang hoạt động (Service hay Receiver cũng không).	5

Tất cả những ứng dụng trong danh sách "Rỗng" sẽ được đưa vào danh sách **LRU** (Least-Recently Used – Ít sử dụng gần đây nhất) và các tiến trình ở đầu danh sách này sẽ bị hủy nếu được yêu cầu từ hệ thống. Nếu ứng dụng được gọi lại bởi người dùng, chúng sẽ lại được đưa xuống cuối danh sách **LRU**.

LRU CACHE



Vòng đời của Activity

Trạng thái của Activity

Vừa rồi chúng ta đã có cái nhìn tổng thể về vòng đời của ứng dụng (xin nhắc lại là cả ứng dụng – **Application** nhé, không chỉ bao gồm **Activity**). Và bây giờ là đến món chính: **Activity**.

Một Activity có thể trong các trạng thái khác nhau tùy vào cách người dùng đang tương tác với nó. Các trạng thái đó được liệt kê trong bảng sau:

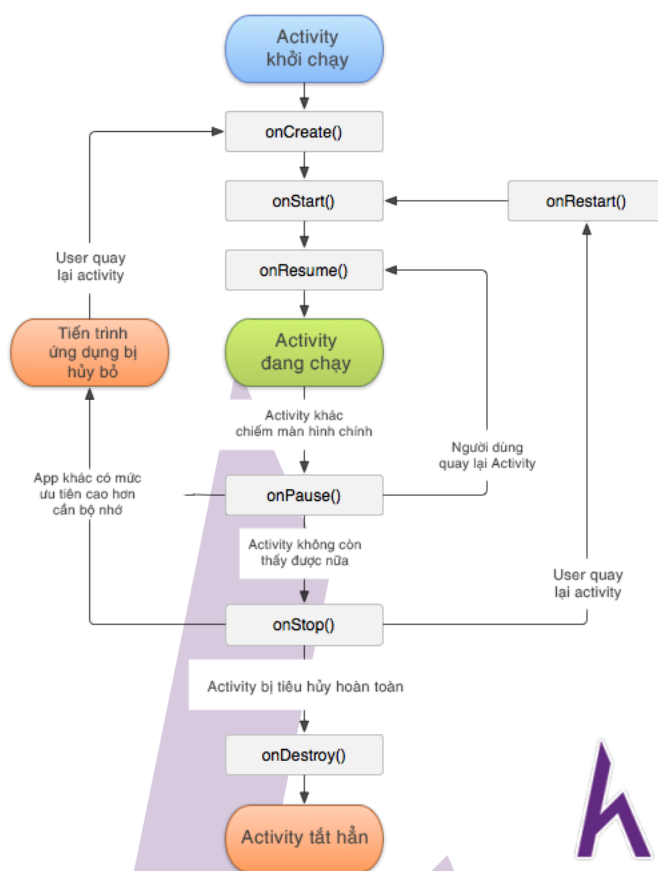
Trạng thái	Mô tả
Running (đang chạy)	Activity đang hiện hữu, và người dùng đang thao tác với nó (đang sờ mó lên màn hình, vuốt vuốt,...).
Paused (tạm dừng)	Activity vẫn hiện, nhưng người dùng đang không thao tác với nó, hoặc bị một Activity khác chen lên chiếm phần nào (ví dụ đang chơi game mà có thông báo cuộc gọi đến chẳng hạn, thì màn hình game sẽ bị pause).
Stopped	Activity không hiện, mặc dù vẫn chạy, và có thể bị triệt tiêu bởi hệ thống.
Killed	Activity bị hủy bằng cách gọi hàm <code>finish()</code>

Các phương thức trạng thái của Activity

Android cung cấp một số các phương thức để điều khiển quy trình làm việc trong vòng đời của Activity. Các phương thức quan trọng có thể kể đến là:

Phương thức	Mục đích
<code>onCreate()</code>	Được gọi ra khi Activity được khởi tạo. Được sử dụng để khởi tạo giao diện, các thành phần hiển thị.
<code>onResume()</code>	Được gọi khi activity lại được trở lại trạng thái hoạt động, tức là người dùng thao tác và nó đang hiện hữu. Sử dụng để khởi tạo các trường, đăng ký các listener, gán với services.
<code>onPause</code>	Được gọi đến khi Activity quay về "hậu cung", luôn luôn được gọi trước khi Activity không còn được nhìn thấy nữa. Thường được sử dụng khi lưu dữ liệu, giải phóng tài nguyên. Ví dụ: Bỏ đăng ký listener, receiver, gỡ bỏ services.
<code>onStop</code>	Được gọi khi Activity không còn hiện hữu. Các tác vụ liên quan đến CPU, database nên được gọi trong khâu này. Từ API 11, phương thức này luôn được đảm bảo gọi ra.

Vòng đời của Activity với một số phương thức quan trọng chính được mô tả bằng biểu đồ sau:



savedInstanceState

savedInstanceState cũng là một trong các thành phần của trạng thái trong vòng đời của một Activity. Đây là...

- Một loại dữ liệu không bền vững.
- Không được lưu trữ cụ thể trong đâu ngoài bộ nhớ RAM.
- Nó được sử dụng để truyền, phục hồi, lưu trạng thái của một Activity.

Vậy nó dùng làm gì?

Các bạn có biết rằng, một Activity, khi đổi chiều xoay màn hình và Activity đó có hỗ trợ chế độ ngang (**landscape**) thì cả Activity đó sẽ bị **destroy**.

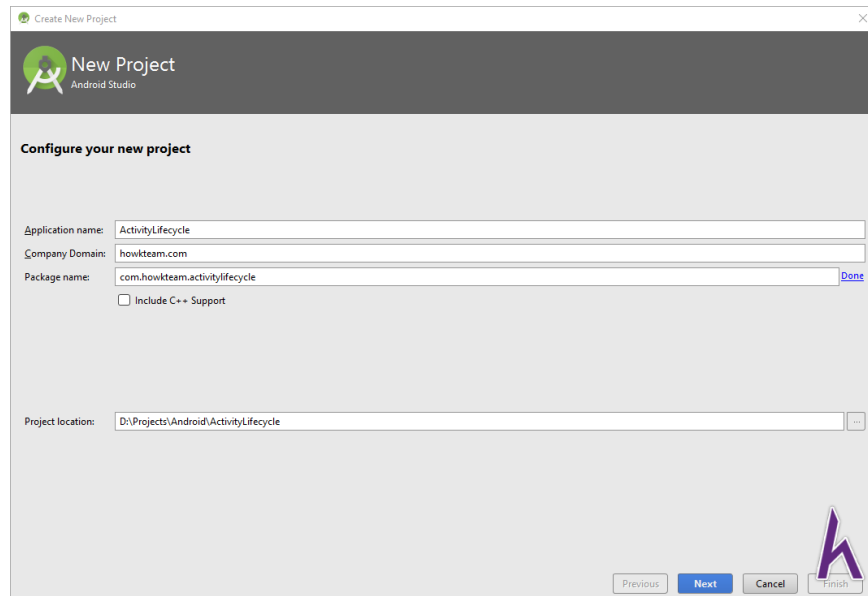
Xin nhắc lại: Cả Activity sẽ bị **destroy** khi bị xoay màn hình.

- Lấy ví dụ đơn giản: Bạn viết một app máy tính bỏ túi đơn giản, chỉ có cộng trừ. Sau khi tính xong phép tính $2+3=5$, bạn xoay màn hình, phép tính biến mất.
 - Dữ liệu trong **savedInstanceState** được lưu dưới dạng **Bundle**.
 - Được phục hồi khi phương thức **onCreate** và **onRestoreSavedInstanceState** được gọi.
 - Được lưu trước **onStop**, với phương thức **onSaveInstanceState**.

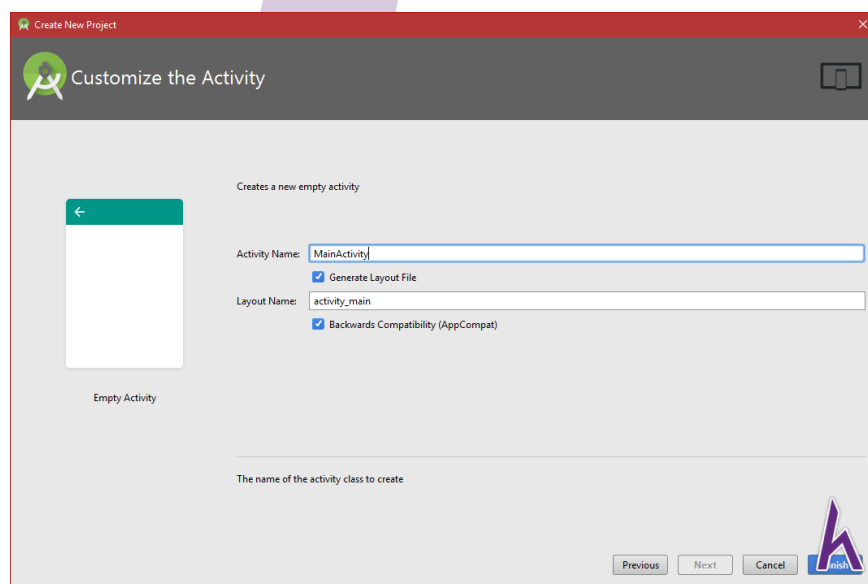
Thực hành: Làm quen với vòng đời của Activity

Để thực hiện phần thực hành này, chúng ta sẽ tạo một project mới, với package name giống như project **HelloWorld** trong bài [GIỚI THIỆU LẬP TRÌNH ANDROID](#), nhưng với tên ứng dụng là **ActivityLifecycle**. Các bạn chưa rõ hay đã quên cách tạo ứng dụng vui lòng xem lại bài [GIỚI THIỆU LẬP TRÌNH ANDROID](#).

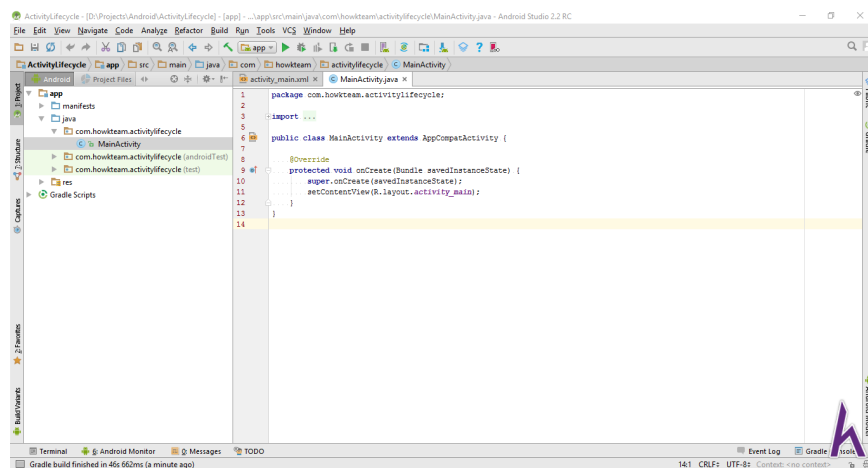
Bước 1: Tạo project với các thông tin như sau:



- **API:** Như project HelloWorld, chúng ta sẽ đặt API ở mức 13 (Android 3.2)
- **Activity:** Chọn **Empty Activity** và nhấn **Finish**.



- Đợi một lúc và chúng ta có được project ban đầu như hình:



Bước 2: Việc chúng ta cần làm là điều tra xem ứng dụng của mình trong quá trình thao tác sẽ rơi vào những trạng thái nào: **Running / Paused / Stop**,.... Chúng ta sẽ sửa code của **MainActivity** thành như sau:

:

```
public class MainActivity extends AppCompatActivity {

    private static final String STATE = "Trang thai";

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

        Log.e(STATE, "onCreate");
    }

    @Override
    protected void onPause() {
        super.onPause();

        Log.e(STATE, "onPause");
    }

    @Override
    protected void onResume() {
        super.onResume();

        Log.e(STATE, "onResume");
    }

    @Override
    protected void onStop() {
        super.onStop();

        Log.e(STATE, "onStop");
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();

        Log.e(STATE, "onDestroy");
    }

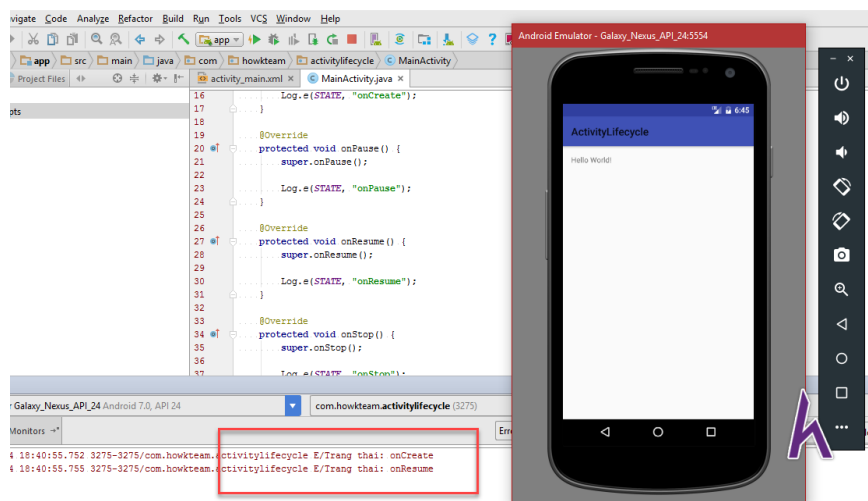
    @Override
    protected void onRestoreInstanceState(Bundle savedInstanceState) {
        super.onRestoreInstanceState(savedInstanceState);

        Log.e(STATE, "onRestoreInstanceState");
    }

    @Override
    protected void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);

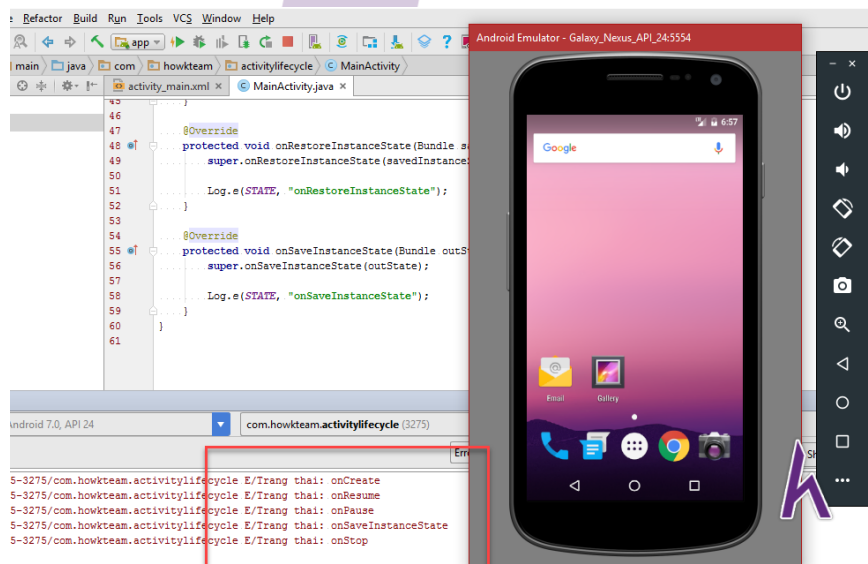
        Log.e(STATE, "onSaveInstanceState");
    }
}
```

Sau đó bấm **Run** để chạy chương trình trên máy. Chúng ta thấy gì?



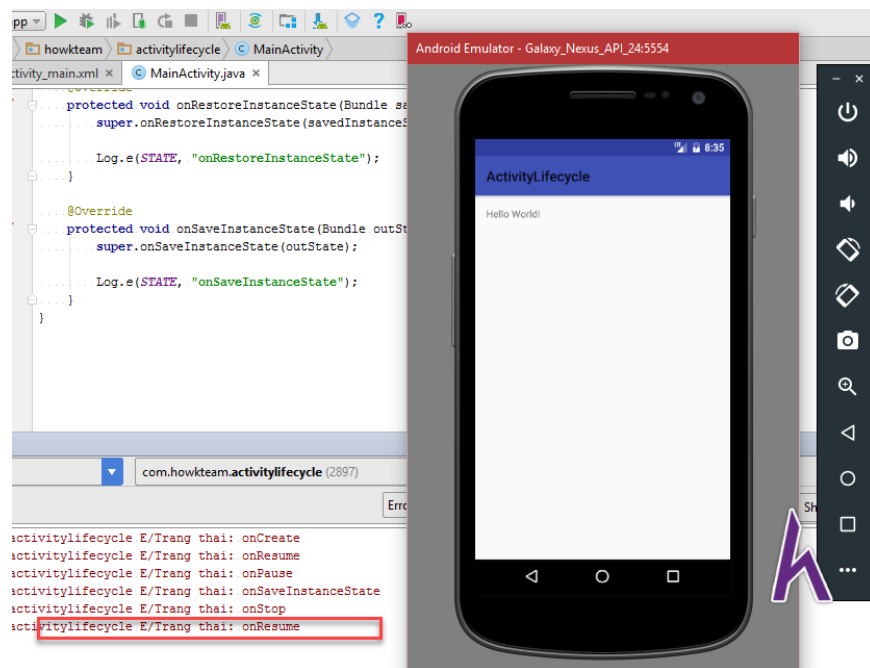
Vậy là theo đúng thứ tự: Ứng dụng khi khởi chạy sẽ bắt đầu từ **onCreate()** đến **onResume()**. Như các bạn đã nhìn thấy trong Log.

Bước 3: Bây giờ các bạn hãy bấm nút **Home** để tạm ra khỏi app và kiểm tra log:




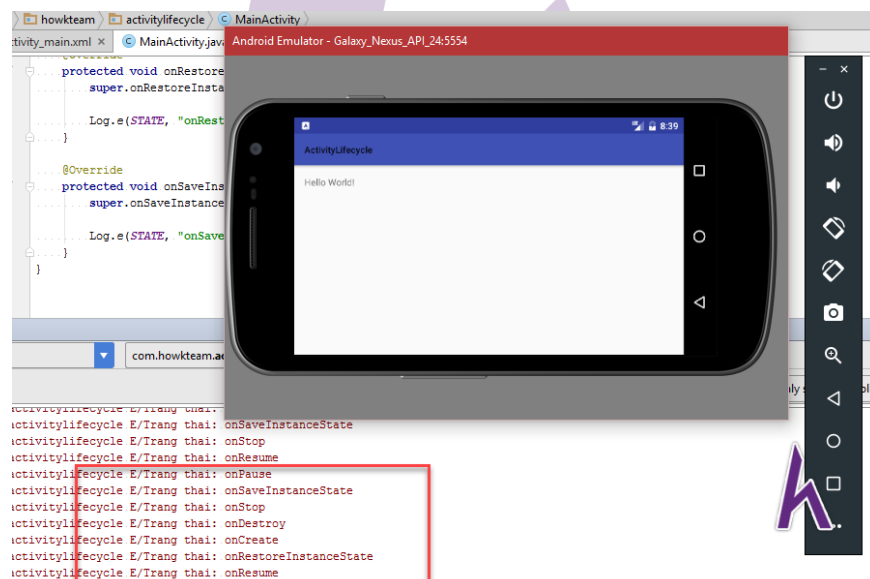
Như các bạn thấy, Activity đi tiếp từ **onPause** đến **onSaveInstanceState** rồi đến **onStop**.

Bước 4: Mở lại ứng dụng và xem Log tiếp nào...



Ồ, ứng dụng đã quay trở lại với việc hàm `onResume` được gọi.

Bước 5: Nào, tiếp theo chúng ta hãy xoay ngang màn hình, như ở ví dụ trên là máy ảo AVD thì icon là  . Và...



Phần khung đỏ là những gì đã xảy ra khi chúng ta xoay màn hình. Cụ thể ở đây là Activity bị tiêu hủy hoàn toàn (`onDestroy`) và được tái tạo lại (`onCreate`).

Câu đố

Câu đố của phần này là: Một Activity luôn có vòng đời từ `onCreate` đến `onDestroy`. Giữa các bước vòng đời này có `onPause` và `onStop`. Có cách nào đi thẳng từ `onCreate` đến `onDestroy` mà không cần qua `onPause` và `onStop` không?

Xem đáp án ở cuối bài viết!

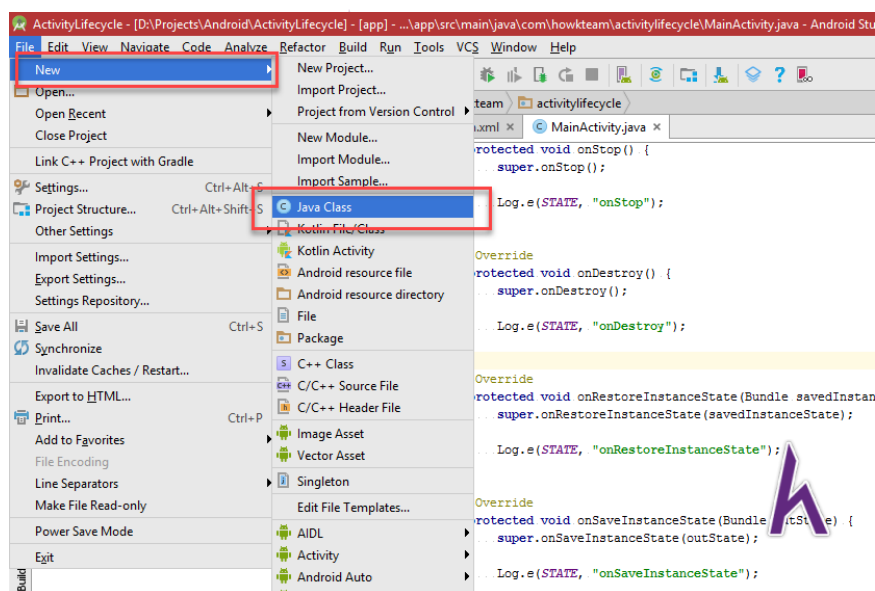
Thực hành: Làm quen với savedInstanceState

Ví dụ sau đây sử dụng tiếp code ở phần Activity ngay trên.

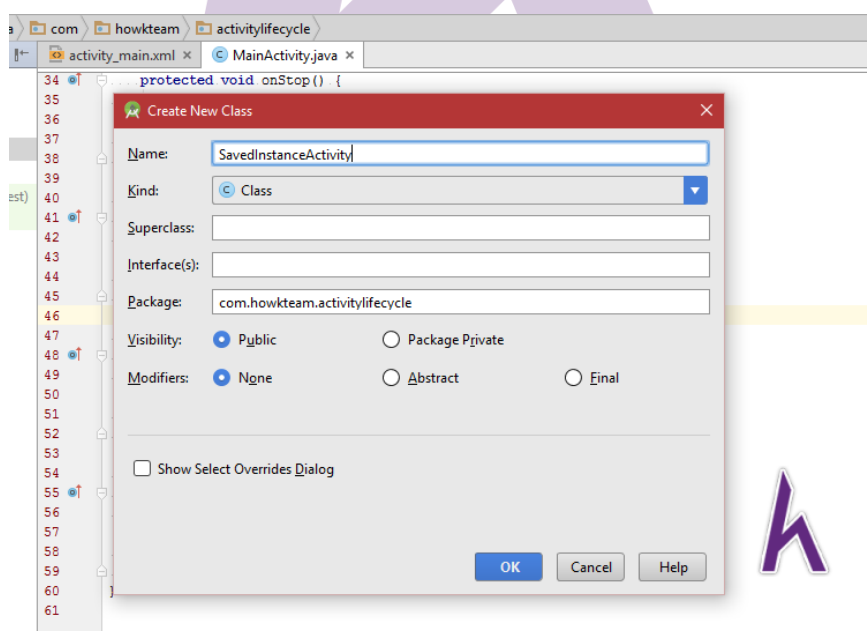
Chúng ta sẽ làm một máy tính cá nhân nhỏ, chỉ có chức năng cộng số.

Mục tiêu là khi xoay ngang xoay dọc màn hình thì số hiển thị vẫn được lưu lại màn hình chứ không bị mất đi.

Bước 1: Tạo một file Java Activity mới. Chỉ cần tạo lại file Java thôi để sau này các bạn xem lại code đỡ nhầm lẫn với phần Activity ở trên. Vào **File > New > Java Class**:



Bước 2: Đặt tên file là **SavedInstanceState**:



Bước 3: Sửa nội dung file **/res/layout/activity_main.xml** thành như sau:

:

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout

    xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:tools="http://schemas.android.com/tools"

    android:id="@+id/activity_main"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:orientation="vertical"

    android:paddingBottom="@dimen/activity_vertical_margin"

    android:paddingLeft="@dimen/activity_horizontal_margin"

    android:paddingRight="@dimen/activity_horizontal_margin"

    android:paddingTop="@dimen/activity_vertical_margin">

    <EditText

        android:id="@+id/et_first_number"

        android:layout_width="match_parent"

        android:layout_height="wrap_content"

        android:inputType="number"/>

    <TextView

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:text="+"/>

    <EditText

        android:id="@+id/et_second_number"

        android:layout_width="match_parent"

        android:layout_height="wrap_content"

        android:inputType="number"/>

    <TextView

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"
```

```

        android:text="" />

<TextView

    android:id="@+id/tv_result"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content" />

<Button

    android:id="@+id/btn_tinh_tong"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content"

    android:text="Tinh tong" />

</LinearLayout>

```

- Và sửa lại file **AndroidManifest.xml** như sau:

:

```

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".SavedInstanceActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>

```

- Class **SavedInstanceActivity** có nội dung như sau:

:

```
package com.howkteam.activitylifecycle;

import android.os.Bundle;

import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

public class SavedInstanceStateActivity extends AppCompatActivity implements View.OnClickListener {

    private EditText etFirstNumber;

    private EditText etSecondNumber;

    private TextView tvResult;

    private Button btnTinhTong;

    private int firstNumber;

    private int secondNumber;

    private int result;

    @Override

    protected void onCreate(@Nullable Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);

        etFirstNumber = (EditText) findViewById(R.id.et_first_number);

        etSecondNumber = (EditText) findViewById(R.id.et_second_number);

        tvResult = (TextView) findViewById(R.id.tv_result);

        btnTinhTong = (Button) findViewById(R.id.btn_tinh_tong);

        btnTinhTong.setOnClickListener(this);

    }

}
```

```
@Override

public void onClick(View view) {

    if (view == btnTinhTong) {

        if (etFirstNumber.getText().toString().isEmpty() || etSecondNumber.getText().toString().isEmpty()) {

            Toast.makeText(this, "Vui long nhap so", Toast.LENGTH_SHORT).show();

        } else {

            firstNumber = Integer.parseInt(etFirstNumber.getText().toString());

            secondNumber = Integer.parseInt(etSecondNumber.getText().toString());

            result = firstNumber + secondNumber;

            tvResult.setText(String.valueOf(result));

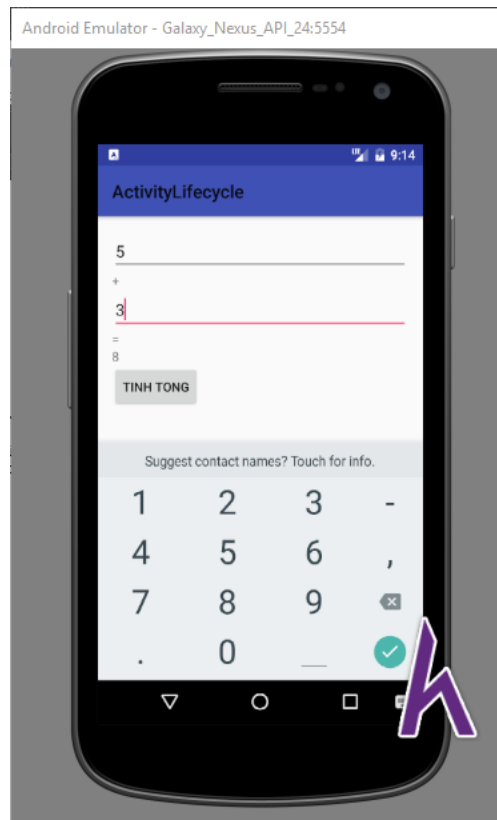
        }

    }

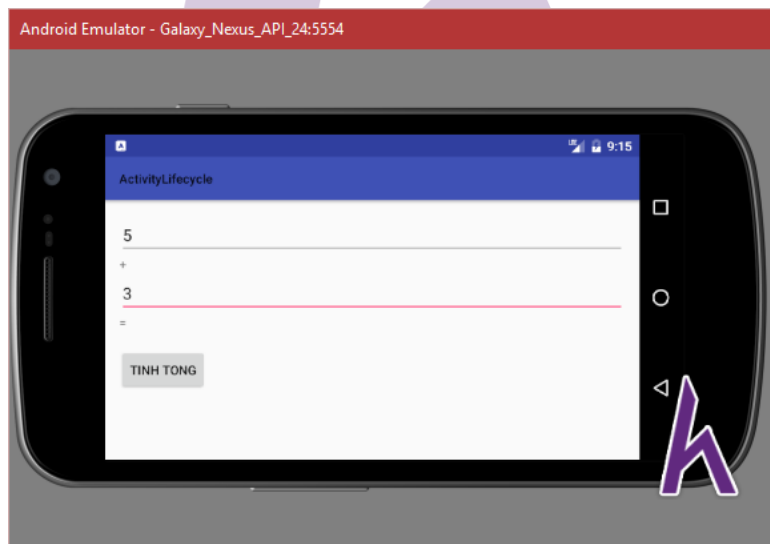
}

}
```

Chạy thử, và tính thử một phép tính đơn giản:



Nhưng khi xoay ngang màn hình thì kết quả đã biến mất do Activity bị Destroy:



Bước 4: Nếu bạn đã thí nghiệm xong về trạng thái của Activity ở **Bước 3** và muốn giữ thông tin ở các ô nhập thì hãy thực hiện tiếp bước này. Sửa code của class **SavedInstanceState** thành như sau:

:

```
package com.howkteam.activitylifecycle;

import android.os.Bundle;

import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

public class SavedInstanceStateActivity extends AppCompatActivity implements View.OnClickListener {

    private EditText etFirstNumber;

    private EditText etSecondNumber;

    private TextView tvResult;

    private Button btnTinhTong;

    private int firstNumber;

    private int secondNumber;

    private int result;

    @Override

    protected void onCreate(@Nullable Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);

        etFirstNumber = (EditText) findViewById(R.id.et_first_number);

        etSecondNumber = (EditText) findViewById(R.id.et_second_number);

        tvResult = (TextView) findViewById(R.id.tv_result);

        if (savedInstanceState != null) {

            etFirstNumber.setText(String.valueOf(savedInstanceState.getInt("SO_THU_NHAT")));

            etSecondNumber.setText(String.valueOf(savedInstanceState.getInt("SO_THU_HAI")));
```



```
        tvResult.setText(String.valueOf(savedInstanceState.getInt("KET_QUA")));
    }

    btnTinhTong = (Button) findViewById(R.id.btn_tinh_tong);

    btnTinhTong.setOnClickListener(this);
}

@Override
protected void onSaveInstanceState(Bundle outState) {
    if (!tvResult.getText().toString().isEmpty()) {
        outState.putInt("SO_THU_NHAT", Integer.parseInt(etFirstNumber.getText().toString()));
        outState.putInt("SO_THU_HAI", Integer.parseInt(etSecondNumber.getText().toString()));
        outState.putInt("KET_QUA", Integer.parseInt(tvResult.getText().toString()));
    }
}

@Override
public void onClick(View view) {
    if (view == btnTinhTong) {
        if (etFirstNumber.getText().toString().isEmpty() || etSecondNumber.getText().toString().isEmpty()) {
            Toast.makeText(this, "Vui long nhap so", Toast.LENGTH_SHORT).show();
        } else {
            firstNumber = Integer.parseInt(etFirstNumber.getText().toString());
            secondNumber = Integer.parseInt(etSecondNumber.getText().toString());

            result = firstNumber + secondNumber;

            tvResult.setText(String.valueOf(result));
        }
    }
}
}
```

Ở đây mình xin giải thích code một chút:

- Trong hàm **onCreate**, mình check xem **savedInstanceState** có **null** không, nếu **null** thì tức là nó không có dữ liệu, còn không **null** thì tức là đã có số được lưu ở trước đó. Và nếu không **null** thì chúng ta lấy các số đó ra đưa lên hiển thị:
 - SO_THU_NHAT** là key để lấy ra giá trị của số thứ nhất trong **savedInstanceState**.
 - SO_THU_HAI** là key để lấy ra giá trị của số thứ hai trong **savedInstanceState**.
 - KET_QUA** là key để lấy ra giá trị của kết quả trong **savedInstanceState**.
- Trong hàm **onSaveInstanceState**, mình lưu các giá trị đã nhập ở các ô cũng theo đúng 3 key ở trên. Trước khi lưu có kiểm tra xem phép tính đã có kết quả hay chưa.

Và bây giờ xoay kiểu gì thì xoay, kết quả vẫn được lưu lại.

Đáp án câu đố

Rất đơn giản, chỉ việc gọi hàm **finish()** ngay trong **onCreate()**. Lúc này **Activity** sẽ bị destroy ngay lúc khởi tạo. Kiểu kiểu như này này:

:

```
package com.howkteam.fragmentexample;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

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

Kết luận

Qua bài này chúng ta đã nắm được về vòng đời của một Activity và hiểu về cơ chế của **savedInstanceState**.

Bài sau chúng ta sẽ tìm hiểu về Fragment, vòng đời của một [FRAGMENT VÀ CƠ CHẾ BACKSTACK](#)

Cảm ơn các bạn đã theo dõi bài viết. Hãy để lại bình luận hoặc góp ý của mình để phát triển bài viết tốt hơn. Đừng quên "**Luyện tập – Thử thách – Không ngại khó**".