

# ACTIVITY & FRAGMENT

NGUYỄN CHÍ HIẾU



1

- 1. Giới thiệu Activity**
- 2. Vòng đời của Activity**
- 3. Gửi nhận dữ liệu giữa các Activity**
- 4. Tích hợp Fragment vào Activity**

## NỘI DUNG

2

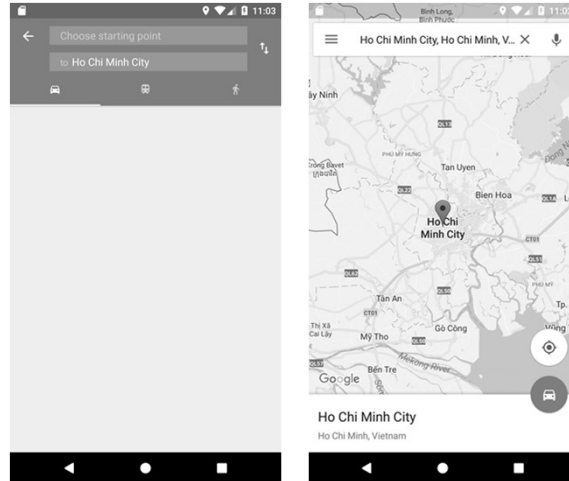
# 1. GIỚI THIỆU ACTIVITY

## ACTIVITY LÀ GÌ?

- Activity là một trong những thành phần cơ bản của ứng dụng Android
- Activity định nghĩa một giao diện (UI - *User Interface*) để hiển thị các điều khiển: Button, EditText, ListView,
- Người dùng tương tác với Activity để thực hiện một tác vụ nào đó như: gọi điện, gửi tin nhắn, gửi mail, ...

## ACTIVITY

- Mỗi ứng dụng là một tập hợp gồm một hay nhiều Activity kết hợp với nhau.
- Mỗi ứng dụng sẽ chỉ định một Activity chính khi khởi động.



## TẠO MỚI ACTIVITY

- Khi tạo dự án mới, Android Studio sẽ tự động tạo một Activity mặc định (MainActivity).
- Tạo mới Activity
  - *NameActivity.java* (viết mã nguồn xử lý cho Activity)
  - *activity\_name.xml* (thiết kế giao diện Activity)

***activity\_main.xml***

- Phương thức onCreate() được override mặc định.

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

***MainActivity.java***

- Thiết kế giao diện của Activity

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.admin.exlifecycle.MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</android.support.constraint.ConstraintLayout>
```

## CẤU HÌNH ACTIVITY TRONG ANDROIDMANIFEST

- Trong AndroidManifest.xml, Activity phải được khai báo giữa cặp thẻ <application> ... </application>

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="ExLifeCycle"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

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

Nguyễn Chí Hiếu

2021 9

9

## CẤU HÌNH ACTIVITY TRONG ANDROIDMANIFEST

- Activity nào khai báo action là android.intent.action.MAIN sẽ được chạy đầu tiên.

```
<application>
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

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

Nguyễn Chí Hiếu

2021 10

10

## KHÁI NIỆM CONTEXT

- **Khái niệm Context**

- Cung cấp thông tin về ngữ cảnh hiện tại của ứng dụng, Activity, Service và cho phép truy xuất tài nguyên ứng dụng.
- Hai loại Context cơ bản
  - Application
  - Activity

## TÀI NGUYÊN ỨNG DỤNG

Nguyễn Chí Hiếu

2021 11

11

## KHÁI NIỆM CONTEXT

- Application: gắn liền với vòng đời của một ứng dụng
  - *getApplicationContext(), getApplication()*

```
Toast.makeText(
    getApplicationContext(),
    "Thông báo",
    Toast.LENGTH_LONG
).show();
```

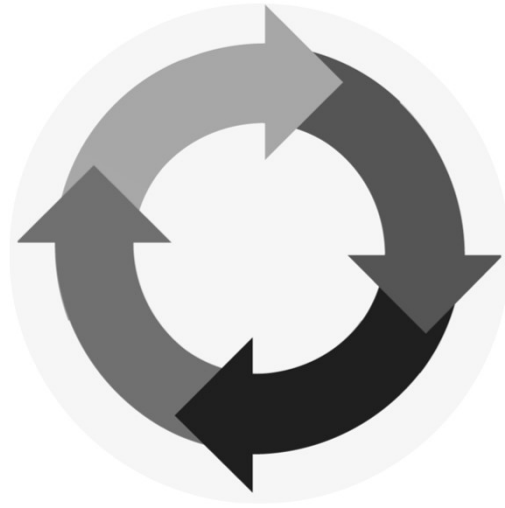
- Activity: gắn liền với vòng đời của một Activity
  - *getBaseContext(), <Tên lớp Activity>.this*
  - *getActivity()* lấy Activity liên kết với Fragment

Nguyễn Chí Hiếu

2021 12

12

## 2. VÒNG ĐỜI ACTIVITY

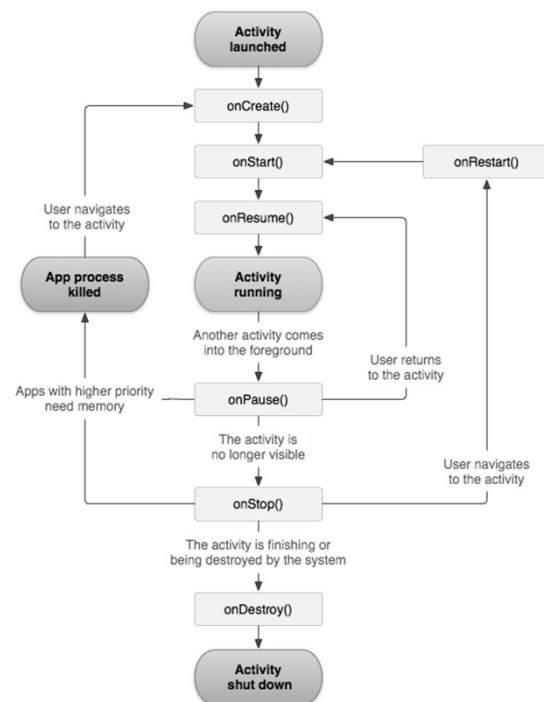


Nguyễn Chí Hiếu

2021 13

13

## VÒNG ĐỜI ACTIVITY



Nguyễn Chí Hiếu

2021 14

14





## VÒNG ĐỜI CỦA ACTIVITY

- `onCreate(Bundle bundle)`
  - Phương thức đầu tiên được gọi khi một Activity khởi tạo và được override mặc định trong mã nguồn.
  - Dùng để khởi tạo các biến, điều khiển, hay dữ liệu tĩnh ...
  - Tham số là một kiểu Bundle dùng để chuyển đổi thông tin giữa các Activity với nhau.
- `onStart()`
  - Được gọi mặc định bởi hệ thống sau khi `onCreate()` thực hiện xong.

## VÒNG ĐỜI CỦA ACTIVITY

- `onResume()`
  - Được gọi khi ứng dụng bắt đầu tương tác người sử dụng.
  - Thường được override trong trường hợp: lắng nghe các sensor (cảm biến), GPS, các cảnh báo, ...
- `onPause()`
  - Gọi khi hệ thống kích hoạt một activity khác

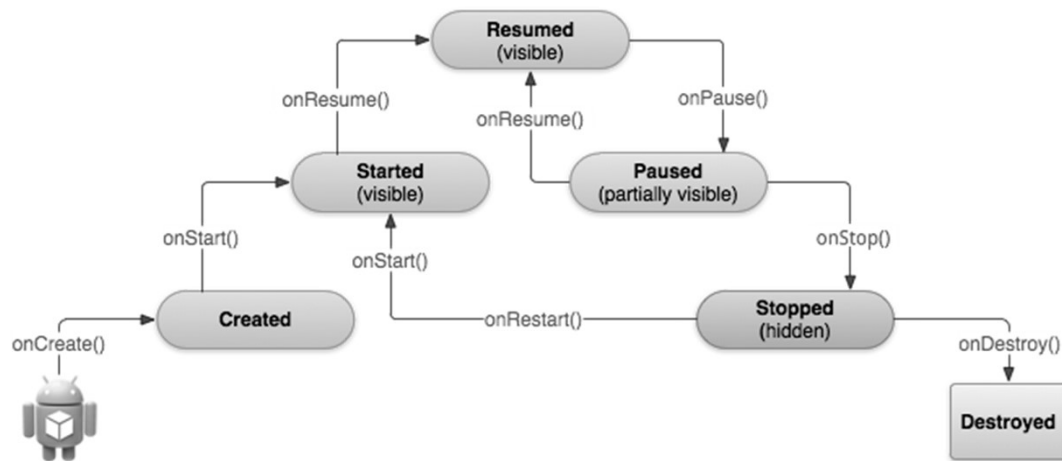
## VÒNG ĐỜI CỦA ACTIVITY

- **onStop()**
  - Được gọi khi người dùng không tương tác Activity trong thời gian dài.
  - Một Activity mới được chạy hay Activity hiện tại bị hủy.
- **onRestart()**
  - Khởi động lại một Activity bị dừng sau khi gọi onPause() hay onStop()
- **onDestroy()**
  - Được gọi để hủy một Activity khỏi vùng nhớ.

## CHUYỂN TRẠNG THÁI TRONG ACTIVITY

- Một Activity có 4 trạng thái
  - **Active** (Created, Started, Resumed): activity đang chạy trên màn hình (*foreground*)
  - **Paused**: khi một activity mất focus nhưng vẫn đang chạy trên màn hình (*background – not focus*)
  - **Stopped**: khi một activity bị che khuất hoàn toàn bởi một activity khác (*background – not visible*)
  - **Destroyed**: khi một activity đang Paused hay Stopped, hệ thống có thể xóa activity ấy nếu cần (*empty*)

## CHUYỂN TRẠNG THÁI TRONG ACTIVITY



Nguyễn Chí Hiếu

2 0 2 1 2 1

21

## CHUYỂN TRẠNG THÁI TRONG ACTIVITY

- Các Activity được quản lý bởi một ngăn xếp (back stack)
  - Khi ứng dụng mở lên, Activity chính được chạy và thêm vào đỉnh stack.
  - Tất cả các Activity khác chuyển về trạng thái dừng hoạt động.
  - Khi đóng một Activity thì nó sẽ bị hủy khỏi đỉnh stack và activity kế tiếp trong stack sẽ chuyển trạng thái tạm dừng sang hoạt động.

Nguyễn Chí Hiếu

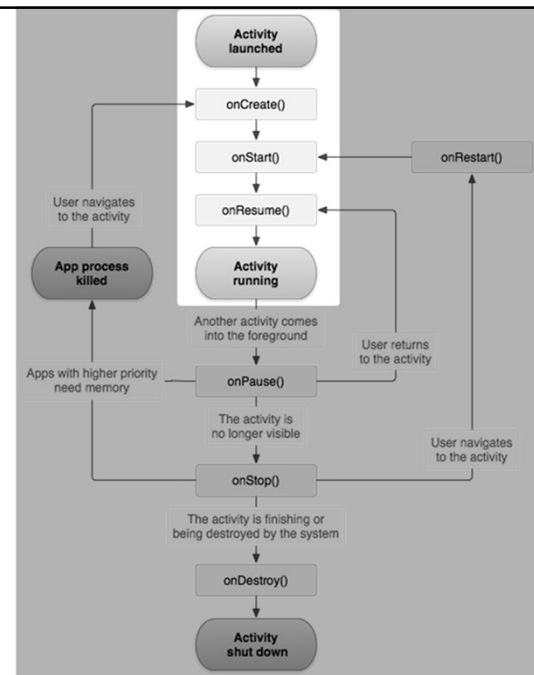
2 0 2 1 2 2

22

# CHUYỂN TRẠNG THÁI TRONG ACTIVITY

• Ví dụ: Khi một Activity được gọi:

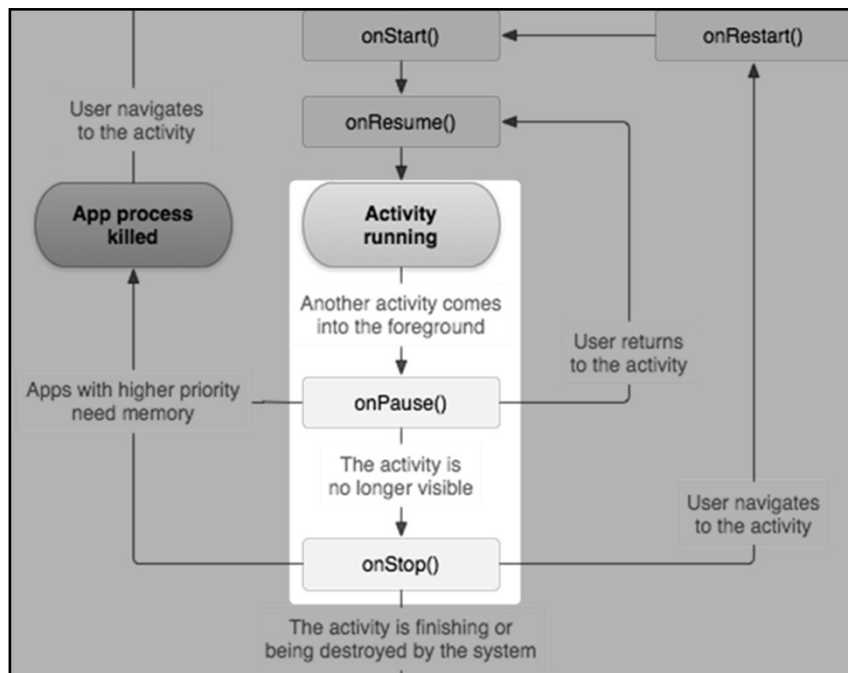
1. onCreate()
2. onStart()
3. onResume()



Nguyễn Chí Hiếu

2 0 2 1 2 3

23



Nguyễn Chí Hiếu

2 0 2 1 2 4

24

# CHUYỂN TRẠNG THÁI TRONG ACTIVITY

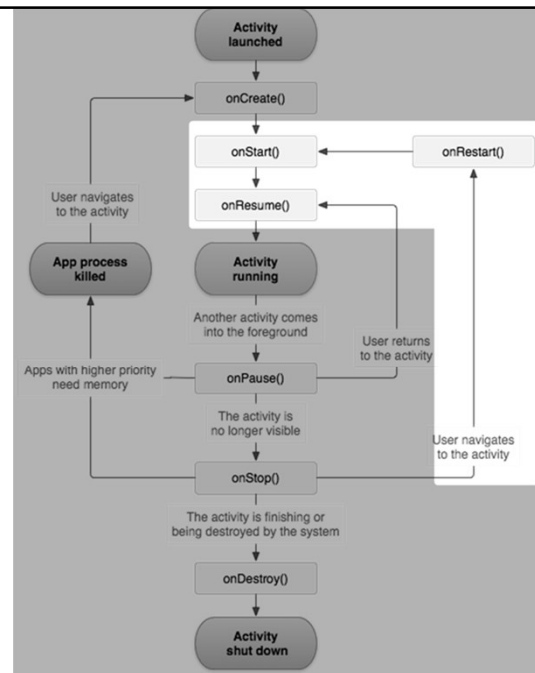
• Ví dụ: Một Activity khác được gọi, Activity hiện tại sẽ thực hiện:

1. onPause()
2. onStop()

# CHUYỂN TRẠNG THÁI CỦA ACTIVITY

• **Ví dụ:** Một Activity khác được gọi và Activity hiện tại không hiển thị trong thời gian dài. Gọi lại Activity trước đó:

1. `onRestart()`
2. `onStart()`
3. `onResume()`



Nguyễn Chí Hiếu

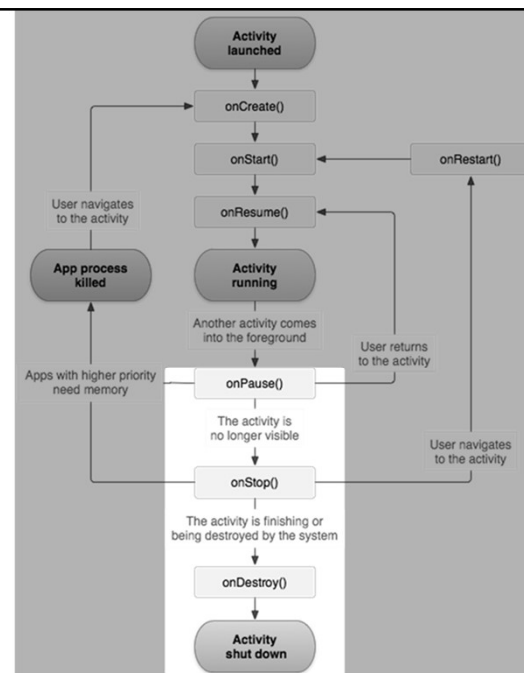
2 0 2 1 2 5

25

# CHUYỂN TRẠNG THÁI TRONG ACTIVITY

• **Ví dụ:** Nhấn phím Back để thoát ứng dụng hay thiết bị không đủ vùng nhớ:

1. `onPause()`
2. `onStop()`
3. `onDestroy()`



Nguyễn Chí Hiếu

2 0 2 1 2 6

26

## CHUYỂN TRẠNG THÁI TRONG ACTIVITY

- **Ví dụ:** Tạo một dự án mới và đặt tên

ExLifecycle.

- Trong *MainActivity.java*, override tất cả các phương thức chuyển trạng thái của Activity.

```
@Override
protected void onStart() {
    super.onStart();
    Log.d("[Activity Lifecycle]", "onStart()");
}
```

Nguyễn Chí Hiếu

2021 27

27

## CHUYỂN TRẠNG THÁI TRONG ACTIVITY

- **Ví dụ:** Tạo một dự án mới và đặt tên ExLifecycle.

- Trong *MainActivity.java*, override tất cả các phương thức chuyển trạng thái của Activity.
- Thêm dòng lệnh hiển thị Log.

```
@Override
protected void onStart() {
    super.onStart();
    Log.i("[Activity Lifecycle]", "onStart()");
}
```

Nguyễn Chí Hiếu

2021 28

28

## QUẢN LÝ CÁC TRẠNG THÁI

- Trong một số trường hợp, ứng dụng tự động chuyển trạng thái của Activity hiện tại:
  - Xoay màn hình thiết bị.
  - Điện thoại có cuộc gọi đến.
- Nhiều trường hợp cần lưu trạng thái trước khi thoát ứng dụng.

### onSaveInstanceState()

- Gọi trước khi *onDestroy()* thực hiện.

### onRestoreInstanceState()

- Gọi là sau khi *onCreate()* kết thúc và có thể khôi phục lại trạng thái của Activity sau khi quá trình khởi tạo hoàn tất.



## QUẢN LÝ CÁC TRẠNG THÁI

## QUẢN LÝ CÁC TRẠNG THÁI

```
@Override
protected void onSaveInstanceState(Bundle outState) {
    super.onSaveInstanceState(outState);
    Log.i("[Activity]", "onSaveInstanceState()");
    mCounter++;
    outState.putInt("counter", mCounter);
}
@Override
protected void onRestoreInstanceState(Bundle savedInstanceState) {
    super.onRestoreInstanceState(savedInstanceState);
    Log.i("[Activity]", "onRestoreInstanceState()");
    mCounter = savedInstanceState.getInt("counter");
    Log.i("[Activity restored]", String.valueOf(mCounter));
}
```

Nguyễn Chí Hiếu

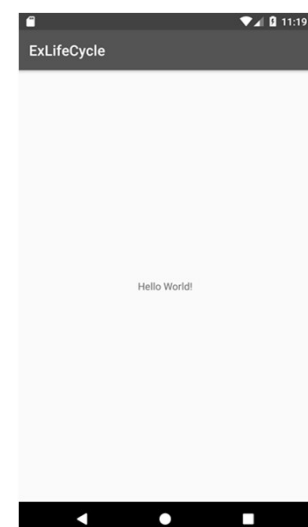
2021 31

31

## QUẢN LÝ CÁC TRẠNG THÁI

- Xoay màn hình máy ảo.

```
[Activity Lifecycle]: onCreate()
[Activity Lifecycle]: onStart()
[Activity Lifecycle]: onResume()
[Activity Lifecycle]: onPause()
[Activity]: onSaveInstanceState()
[Activity Lifecycle]: onStop()
[Activity Lifecycle]: onDestroy()
[Activity Lifecycle]: onCreate()
[Activity Lifecycle]: onStart()
[Activity]: onRestoreInstanceState()
[Activity restored]: 1
[Activity Lifecycle]: onResume()
```



Nguyễn Chí Hiếu

2021 32

32



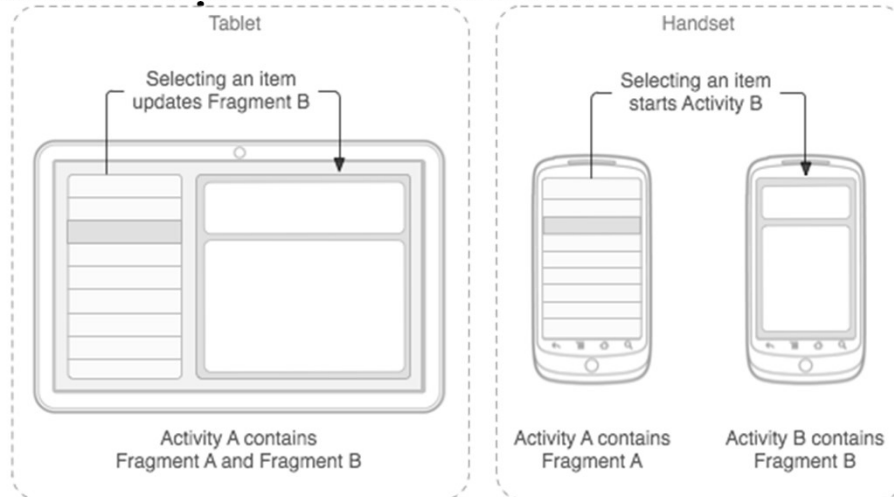
## 4. TÍCH HỢP FRAGMENT VÀO ACTIVITY

- Giới thiệu Fragment
- Vòng đời của Fragment
- Quản lý Fragment trong Activity

## GIỚI THIỆU FRAGMENT

- Fragment là một thành phần của ứng dụng có vòng đời và giao diện riêng tương tự như một Activity.
- Phải được *tích hợp vào một Activity nào đó* và *mỗi Activity có thể chứa nhiều Fragment*.
- Fragment có thể tái sử dụng giúp tạo giao diện Activity linh động hơn.

## KHÁI NIỆM FRAGMENT



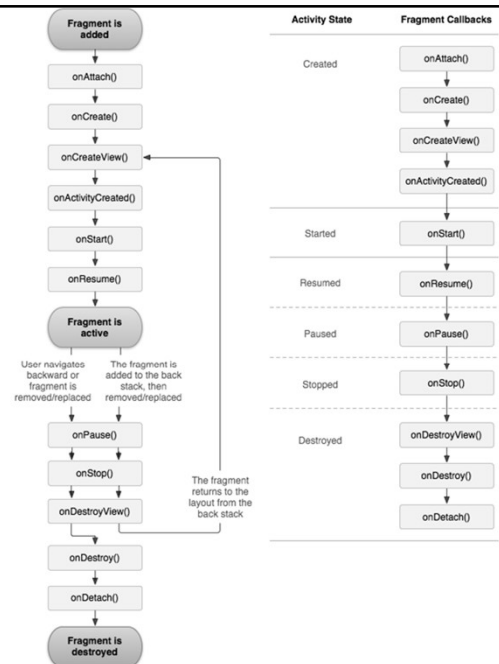
Nguyễn Chí Hiếu

2021 35

35

## VÒNG ĐỜI CỦA FRAGMENT

- Fragment có một số phương thức chuyển trạng thái tương tự như Activity.
- Khi trạng thái Activity thay đổi sẽ gọi một số phương thức chuyển trạng thái của Fragment.



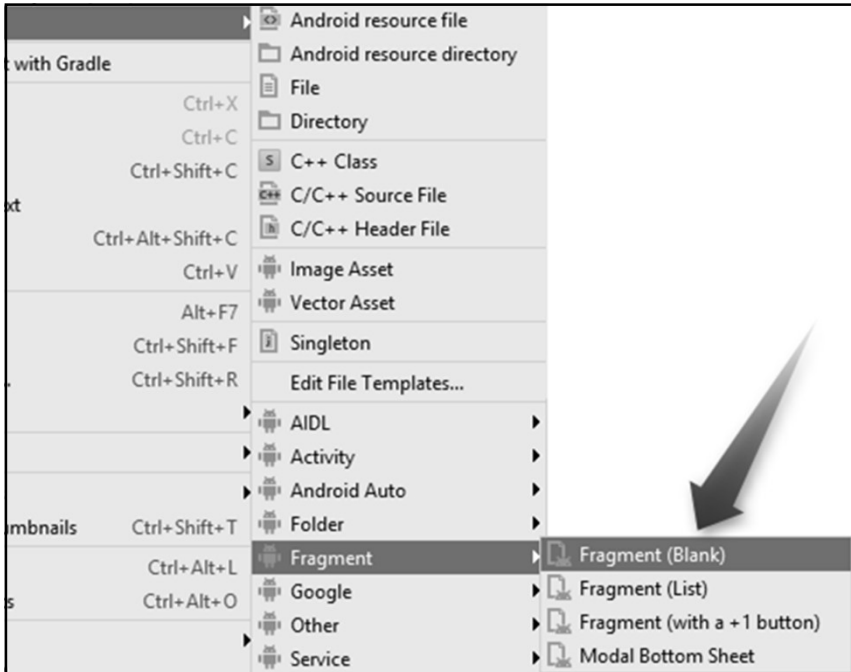
Nguyễn Chí Hiếu

2021 36

36

## QUẢN LÝ FRAGMENT TRONG ACTIVITY

- Ví dụ 7. Thêm Fragment
  - Tạo một dự án mới và đặt tên ExFragment.
  - Chọn res\New\Fragment\Fragment (Blank)

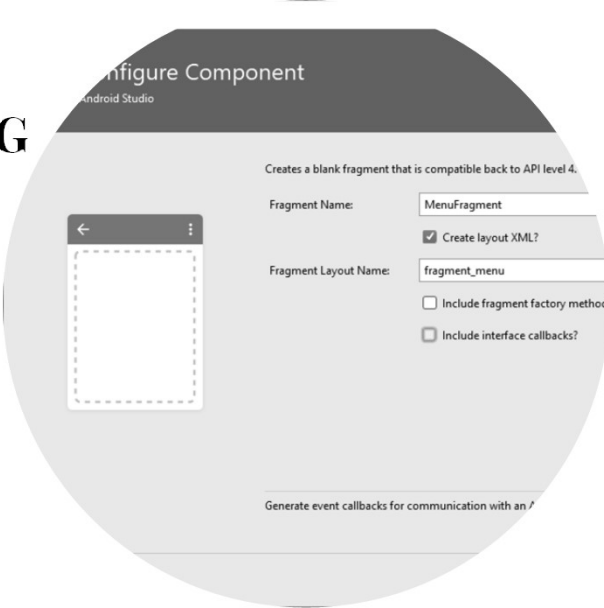


Nguyễn Chí Hiếu 2021 37

37

## QUẢN LÝ FRAGMENT TRONG ACTIVITY

- Đặt tên Fragment là MenuFragment
- Android sinh ra 2 tập tin:
  - MenuFragment.java
  - fragment\_menu.xml (trong package layout).



Nguyễn Chí Hiếu 2021 38

38

## QUẢN LÝ FRAGMENT TRONG ACTIVITY

- Trong *fragment\_menu.xml*, thêm vào đoạn mã sau:

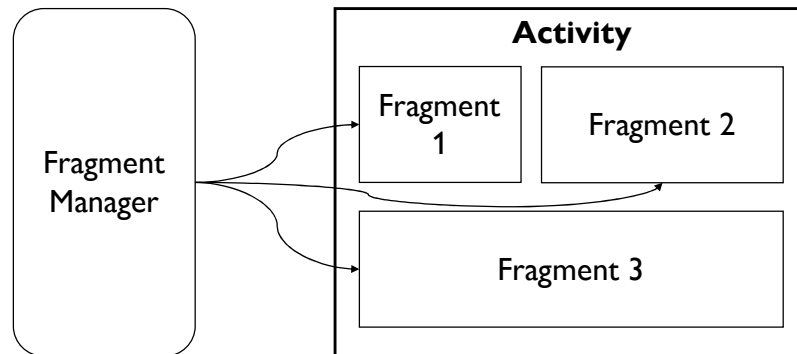
```
<FrameLayout ...
    android:id="@+id/fragmentMenu">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textAlignment="center"
        android:text="MenuFragment" />
</FrameLayout>
```

## TÍCH HỢP FRAGMENT VÀO ACTIVITY

- **Cách 1** [Khai báo tĩnh]: trong tập tin *\*.xml* của Activity sẽ tích hợp Fragment.

```
<LinearLayout ...
    android:orientation="vertical">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="200dp"
        android:textAlignment="center"
        android:text="MainActivity"/>
    <fragment
        android:id="@+id/fragmentMenu"
        android:name="com.example.admin.exfragment.MenuFragment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

- **Cách 2** [Khai báo động]: trong tập tin *\*.java* của Activity (có thể dùng để cập nhật lại Fragment).
- Mỗi Activity có một **FragmentManager** để quản lý các Fragment.

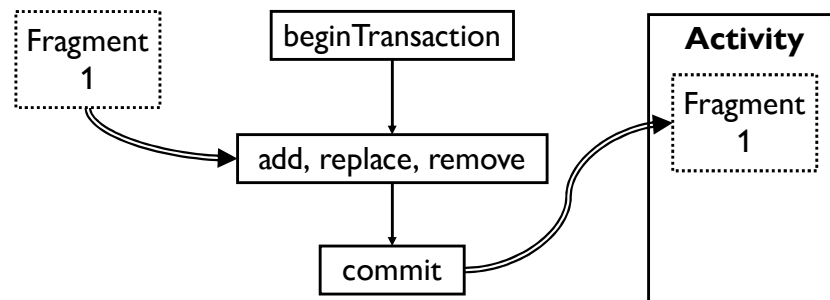


## QUẢN LÝ FRAGMENT TRONG ACTIVITY

- FragmentManager
  - `getSupportFragmentManager()`
  - `getFragmentManager()`
  - ...
- FragmentTransaction
  - `beginTransaction(), commit()`
  - `add(), replace(), remove(),`
  - ...

## QUẢN LÝ FRAGMENT TRONG ACTIVITY

- Các bước thực hiện



- Trong *activity\_main.xml*, thêm id cho Activity là *activityMain*.
- Trong *MainActivity.java*, thêm đoạn mã tích hợp Fragment vào.

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    //
    FragmentManager manager = getSupportFragmentManager();
    FragmentTransaction transaction = manager.beginTransaction();
    transaction.add(R.id.activityMain, new MenuFragment());
    transaction.commit();
}
  
```

## BÀI TẬP

- Cài đặt ứng dụng Quiz với câu hỏi trắc nghiệm:
  - Câu hỏi có thể là: đúng sai, chọn một đáp án, chọn một hay nhiều đáp án khác nhau.
  - Danh sách các câu hỏi và trả lời cho sẵn. Định nghĩa lớp Question gồm 2 thuộc tính: id, type (loại câu hỏi), question (câu hỏi), answer (đáp án)

## TÀI LIỆU THAM KHẢO

1. J. F. DiMarzio. *“Beginning Android Programming with Android Studio”*. Wrox 2016.
2. <https://developer.android.com/>
3. <https://developer.android.com/studio>