Android skeleton và hoạt động

ThS. Trần Hồng Vinh

Nội dung bài học

- Giới thiệu các thành phần của ứng dụng.
- Activity là gì?
- Các trạng thái của Activity
- Vòng đời của Activity
- Thực hành tạo Activity
- Sử dụng các tài nguyên bên trong và bên ngoài

Giới thiệu các thành phần của ứng dụng

- **Activity** là một thành phần cơ bản của ứng dụng Android, có hiển thị giao diện và các xử lý tương tác với người sử dụng.
- **Service** là thành phần chạy ngầm để thực hiện các tác vụ dài hạn mà không có giao diện người dùng. Ví dụ, một Service có thể phát nhạc trong nền hoặc tải dữ liệu từ máy chủ.Service có thể được chia thành hai loại chính: Started Service (bắt đầu bởi một thành phần khác và chạy trong nền) và Bound Service (cho phép các thành phần khác kết nối và tương tác).

Giới thiệu các thành phần của ứng dụng

- **Intent** là một đối tượng cho phép giao tiếp giữa các thành phần của ứng dụng hoặc giữa các ứng dụng khác nhau. Intent có thể được sử dụng để khởi động một Activity, Service, hay gửi broadcast.
- Intent có hai loại chính: Explicit Intent và Implicit Intent

Giới thiệu các thành phần của ứng dụng

- **Content Providers** cho phép các ứng dụng chia sẻ dữ liệu với nhau. Chúng cung cấp giao diện chuẩn để truy cập và sửa đổi dữ liệu lưu trữ trong cơ sở dữ liệu, tệp hoặc trên web.
- Content Providers sử dụng URI (Uniform Resource Identifier) để truy cập dữ liệu. Ví dụ, ứng dụng danh bạ có thể sử dụng Content Providers để chia sẻ thông tin danh bạ với các ứng dụng khác.

Các loại ứng dụng

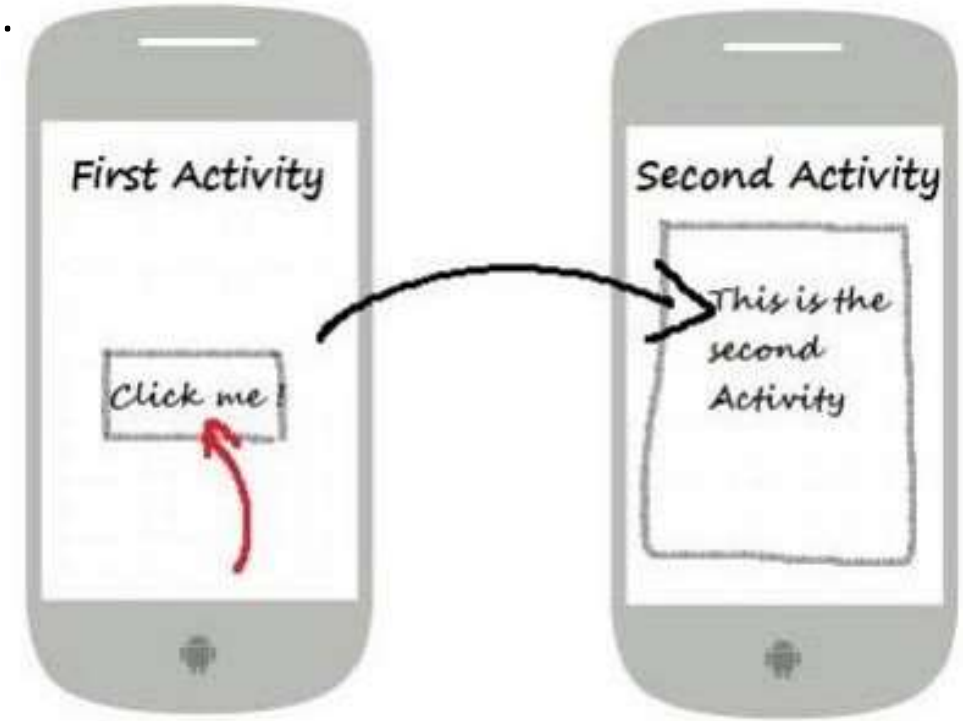
- **Standalone Application** là ứng dụng độc lập, không phụ thuộc vào các ứng dụng khác để hoạt động. Chúng có thể hoạt động mà không cần sự hỗ trợ từ các thành phần hoặc dịch vụ khác. Ví dụ, một ứng dụng đơn giản như máy tính hoặc lịch.
- **Widget** là các thành phần giao diện nhỏ có thể được đặt trên màn hình chính của thiết bị Android để hiển thị thông tin hoặc cung cấp các chức năng nhanh mà không cần mở ứng dụng. Ví dụ, widget thời tiết hiển thị thông tin thời tiết hiện tại và dự báo.
- **Background Application** là các ứng dụng hoặc dịch vụ chạy ngầm để thực hiện các tác vụ mà không cần sự can thiệp của người dùng trực tiếp. Ví dụ, ứng dụng nghe nhạc có thể phát nhạc trong nền hoặc ứng dụng email có thể kiểm tra thư mới định kỳ.

Activity là gì?

- **Activity** là một thành phần cơ bản của ứng dụng Android, có hiển thị giao diện và các xử lý tương tác với người sử dụng.
- Một ứng dụng Android có thể có một hoặc nhiều Activity, Activity được chạy đầu tiên khi khởi động ứng dụng gọi là Activity chính (*main-activity)*.
- Một lớp được gọi là Activity khi nó extend (kế thừa) từ những lớp cha như **Activity**, **AppCompatActivity** hay **FragmentActivity**.

Activity là gì?

- Mỗi Activity sẽ hoạt động độc lập với nhau nhưng có thể tương tác và truyền dữ liệu qua nhau.



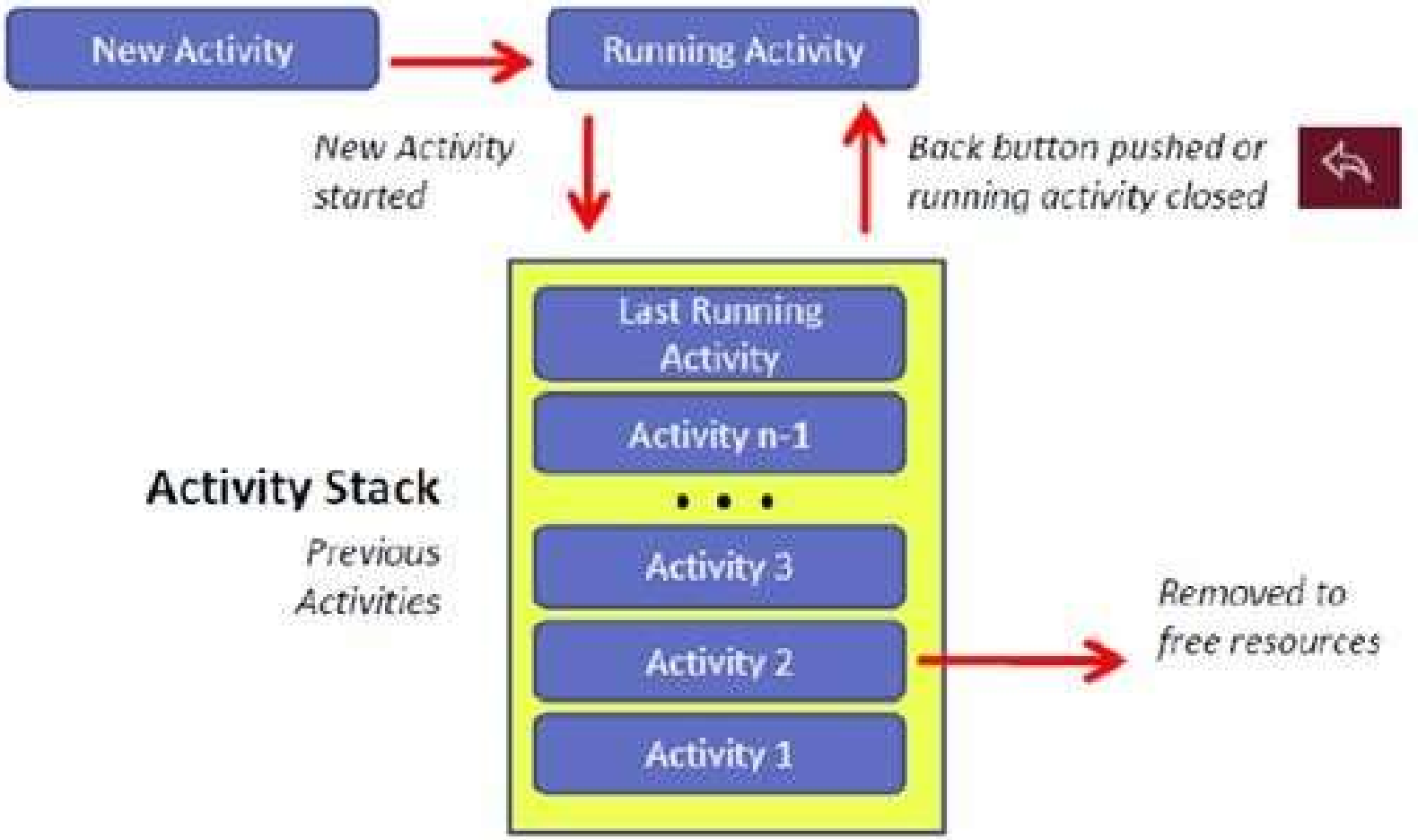
Slide 8

vt1 vinh tran, 05-Sep-21

Hoạt động của Activity

- Trong quá trình hoạt động, mỗi activity có thể khởi động một Activity khác để thực hiện các tác vụ khác.
- Khi một Activity mới được kích hoạt, Activity hiện hành sẽ bị tạm dừng và sẽ được đặt vào ngăn xếp lùi (Back-Stack).
  - Back-stack hoạt động theo cơ chế LIFO
  - Activity sẽ trải qua một số các trạng thái nhất định trong vòng đời của nó.

Hoạt động của Activity – Back-Stack



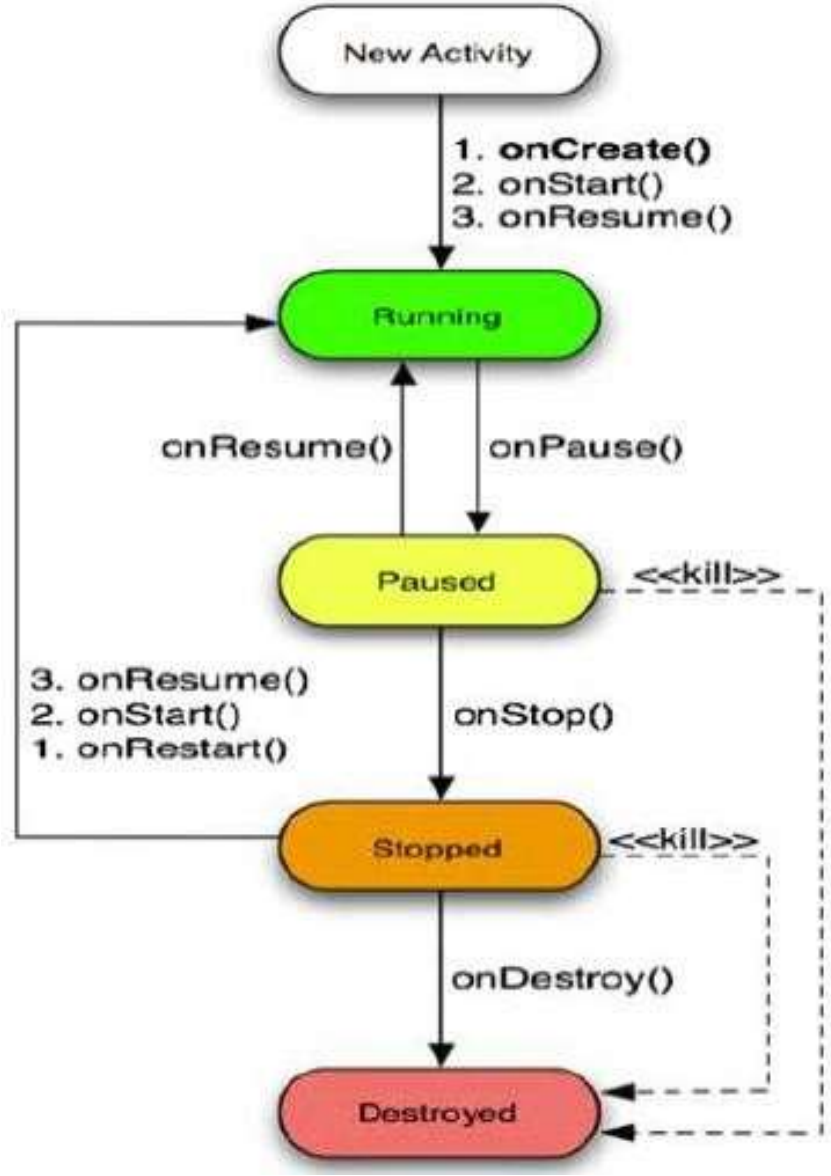
Trạng thái của Activity

- **Running (hoạt động)** : Khi Activity được kích hoạt và được hệ thống để vào ngăn xếp back-stack.
  - Người dùng nhìn thấy và tương tác với ứng dụng
- **Paused (tạm dừng):** Khi bị một thành phần khác che một phần của Activity hiện tại, như bị dialog đè lên.
  - Người dùng nhìn thấy nhưng không tương tác được với ứng dụng
  - Activity bị tạm dừng, nó vẫn tồn tại trong bộ nhớ.

Trạng thái của Activity

- **Stopped (bị dừng):** Khi bị che khuất hoàn toàn bởi một thành phần giao diện nào đó.
  - Người dùng không nhìn thấy và không tương tác được.
  - Ví dụ, người dùng nhấn nút Home
- **Destroyed (kết thúc):** Activity kết thúc khi hoàn tất tác vụ của nó, hoặc người dùng chủ động hủy, hoặc bị hệ thống hủy khi cần tài nguyên.
  - Ví dụ, khi người dùng nhấn nút Back
  - Activity sẽ kết thúc vòng đời của nó

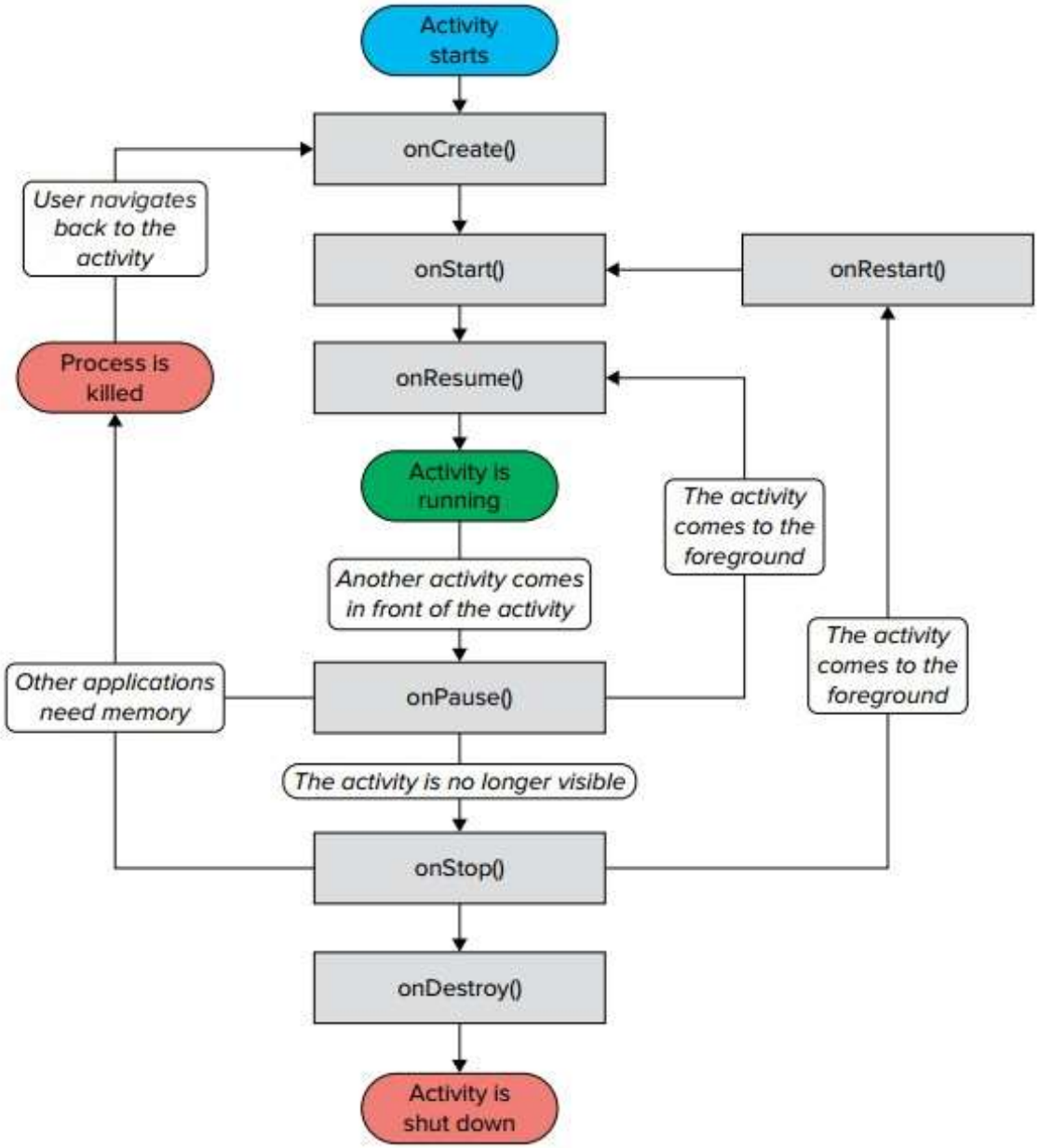
Trạng thái của Activity



Vòng đời của Activity

- Khi một activity chuyển đổi hoặc thoát khỏi các trạng thái khác nhau, activity sẽ được thông báo qua nhiều phương thức callback khác nhau (onCreate, onPause, onResume,…)
- Vòng đời của Activity mô tả quá trình hoạt động và tương tác của một Activity kể từ khi nó bắt đầu chạy, cho tới khi kết thúc.

Vòng đời của Activity



3 chu kỳ trong vòng đời của Activity

- **Entire lifetime**: Từ khi gọi onCreate() cho tới onDestroy() – tức là từ lúc Activity được gọi ra cho đến lúc nó bị huỷ.
- **Visible liftetime**: Từ khi gọi onStart() cho tới lúc gọi onStop(), trong trường hợp này ta vẫn có thể thấy màn hình Activity.
- **Foreground lifetime**: Từ khi gọi onResume() cho tới lúc gọi onPause(), quá trình này Activity luôn nằm ở foreground và ta có thể tương tác được với nó.

Các phương thức callback

- **onCreate()** – được gọi khi activity được kích hoạt, trước khi hiển thị giao diện.
  - Callback này chỉ được gọi một lần duy nhất
  - Dùng để khởi tạo cho activity như load giao diện, các lời gọi API, load database,…
- **onStart()** – được gọi khi Activity bắt đầu được hiện ra, trước khi nhận tương tác với người dùng.
  - Callback này ít dùng trong lập trình
- **onResume()** – được gọi khi Activity đã nhìn thấy và nhận tương tác với người dùng.
  - Callback dùng để khôi phục hoạt động các tác vụ

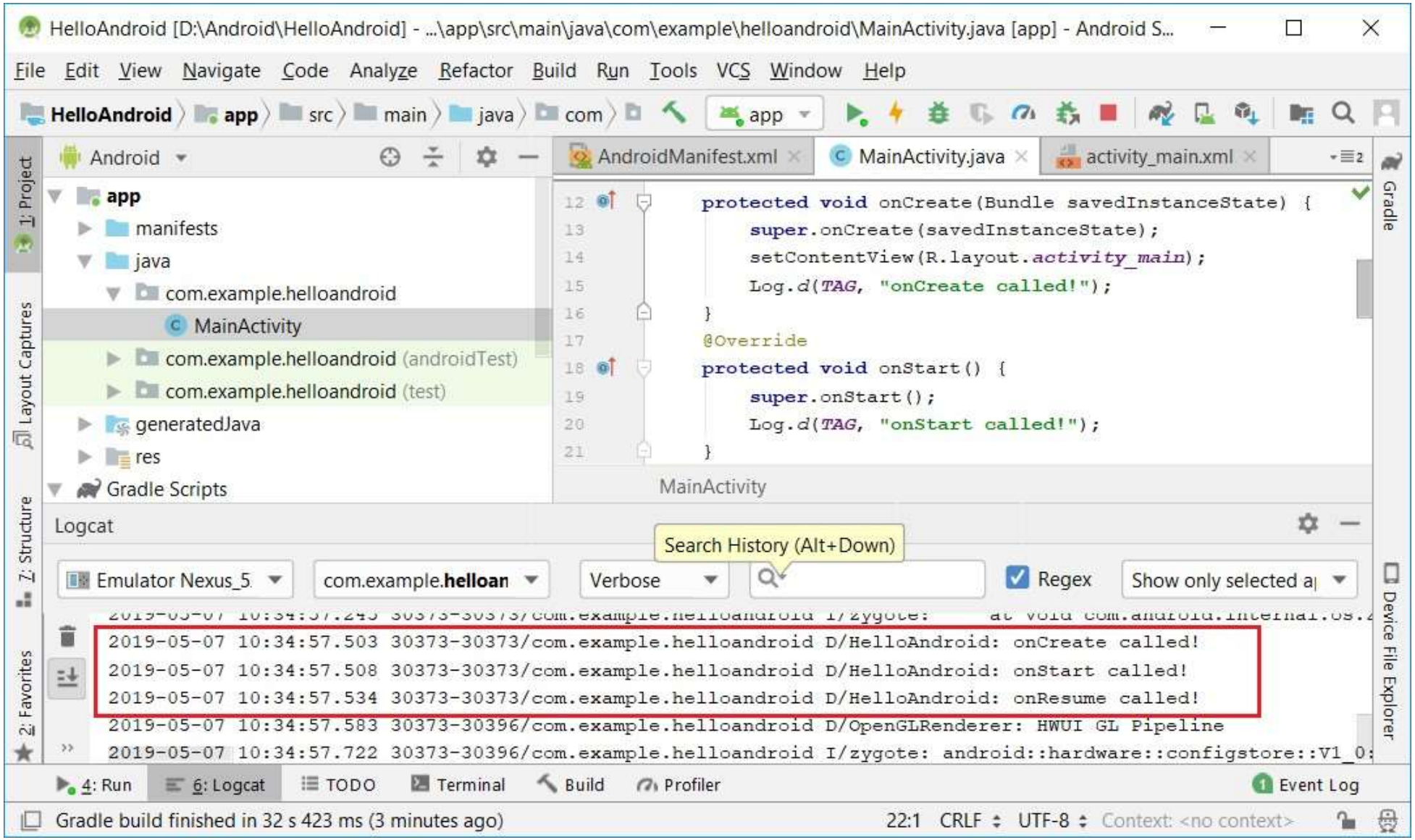
Các phương thức callback (tt)

- **onPause()** – được gọi khi activity khi có thành phần nào đó che một phần Activity hiện tại.
  - Thường dùng để thực hiện tạm dừng các tác vụ đang chạy, như tạm dừng sound, game pause…
- **onStop()** – được gọi khi Activity bị che khuất hoàn toàn hoặc nhấn nút Home.
  - Callback này ít dùng trong lập trình
- **onDestroy()** – được gọi khi Activity kết thúc, như bị hệ thống kill hoặc người tắt ứng dụng.
  - Callback được dùng để giải phóng tài nguyên

Thực hành với Activity

- Mở file MainActivity của project HelloAndroid, ghi đè các phương thức callback và thêm log hiển thị
  - Dùng Toast.makeText().show()
  - Hoặc: Log.d(), Log.e(), Log.w(), Log.i()
- Thực hiện các thác làm ảnh hưởng để hoạt động của Activity và cho nhận xét hiển thị ở cửa sổ logcat.
  - Chạy ứng dụng khác, nhấn nút Back, nút Home,…

Kết quả hiển thị



Sử dụng các tài nguyên bên trong và bên ngoài

❑ Tài nguyên bên trong (Internal Resources):

Tài nguyên bên trong là những tài nguyên được lưu trữ trong ứng dụng và không phụ thuộc vào mạng hoặc bộ nhớ ngoài.

❖ String Resources (res/values/strings.xml):

- Lưu trữ các chuỗi văn bản để hỗ trợ đa ngôn ngữ và dễ dàng bảo trì.
- Ví dụ: res/values/strings.xml

```
<resources>
    <string name="app_name">MyApp</string>
    <string name="welcome_message">Welcome to
MyApp!</string>
</resources>
```

Sử dụng các tài nguyên bên trong và bên ngoài

- **Drawable Resources (res/drawable/):**
  - Lưu trữ các tài nguyên đồ họa như hình ảnh, vector, và các định dạng khác.
  - Ví dụ: res/drawable/logo.png
- **Layout Resources (res/layout/):**
  - Lưu trữ các tệp XML định nghĩa giao diện người dùng của ứng dụng.
  - Ví dụ: res/layout/activity_main.xml

```
<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/welcome_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/welcome_message"/>
</LinearLayout>
```

Sử dụng các tài nguyên bên trong và bên ngoài

- Color Resources (res/values/colors.xml):
  - Định nghĩa các màu sắc được sử dụng trong ứng dụng.
  - Ví dụ: res/values/colors.xml

```
<resources>
    <color name="primary_color">#6200EA</color>
    <color name="primary_dark_color">#3700B3</color>
    <color name="accent_color">#03DAC5</color>
</resources>
```

Sử dụng các tài nguyên bên trong và bên ngoài

❑ Tài nguyên bên ngoài (External Resources):

Tài nguyên bên ngoài là những tài nguyên được truy cập từ bên ngoài ứng dụng, chẳng hạn như từ mạng internet, bộ nhớ ngoài, hoặc các dịch vụ web.

- ❖ Tệp từ bộ nhớ ngoài (External Storage):
  - Đọc và ghi tệp từ bộ nhớ ngoài của thiết bị, chẳng hạn như thẻ nhớ SD.
- ❖ Dữ liệu từ mạng (Network Data):
  - Truy cập và tải dữ liệu từ internet, chẳng hạn như API RESTful hoặc tải hình ảnh.
- ❖ Tích hợp dịch vụ bên ngoài (External Services Integration):
  - Sử dụng các dịch vụ bên ngoài như Firebase, Google Maps API, hoặc các dịch vụ bên thứ ba khác để thêm tính năng vào ứng dụng.