GIỚI THIỆU TỔNG QUAN

Tiếp tục thực hiện trên mã nguồn ứng dụng TwoActivities.

Task 1: Thêm lifecycle callbacks vào ứng dụng TwoActivities

Cài đặt tất cả các phương thức lifecycle callback của `Activity` để in ra các message trên Logcat khi các phương thức này được gọi thực thi.

Các log message này giúp chúng ta xem được việc thay đổi trạng thái trong lifecycle của `Activity` và ảnh hưởng như thế nào khi ứng dụng chạy.

1.1 (Tùy chọn) Copy project TwoActivities

Chúng ta có thể sao chép project TwoActivities từ bài tập trước để thực hành bài tập này

1.2 Cài đặt các lifecycle callback trong MainActivity

1. Trong phương thức `onCreate()`: thêm các câu lệnh ghi log như sau:

```
Log.d(LOG_TAG, "-------");
Log.d(LOG_TAG, "onCreate");
```

2. Ghi đè phương thức `onStart()`: ghi log với nội dung “onStart”

```
@Override
public void onStart() {
    super.onStart();
    Log.d(LOG_TAG, "onStart");
}
```

Cách tắt để ghi đè các phương thức: chọn **Code > Override Methods**. Chọn một hoặc nhiều phương thức trong hộp thoại chứa danh sách các phương thức có thể ghi đè trong Class.

3. Tương tự phương thức `onStart()`, cài đặt cho các phương thức: `onPause()`, `onRestart()`, `onResume()`, `onStop()`, và `onDestroy()`.

4. Chạy ứng dụng.
5. Xem các log message trong **Logcat** tab:

```
D/MainActivity: -------
D/MainActivity: onCreate
D/MainActivity: onStart
D/MainActivity: onResume
```

1.3 Cài đặt các lifecycle callback trong SecondActivity

Cài đặt các lifecycle callback cho SecondActivity tương tự như trong MainActivity.

1. Mở **SecondActivity**.
2. Khai báo hằng số LOG_TAG:

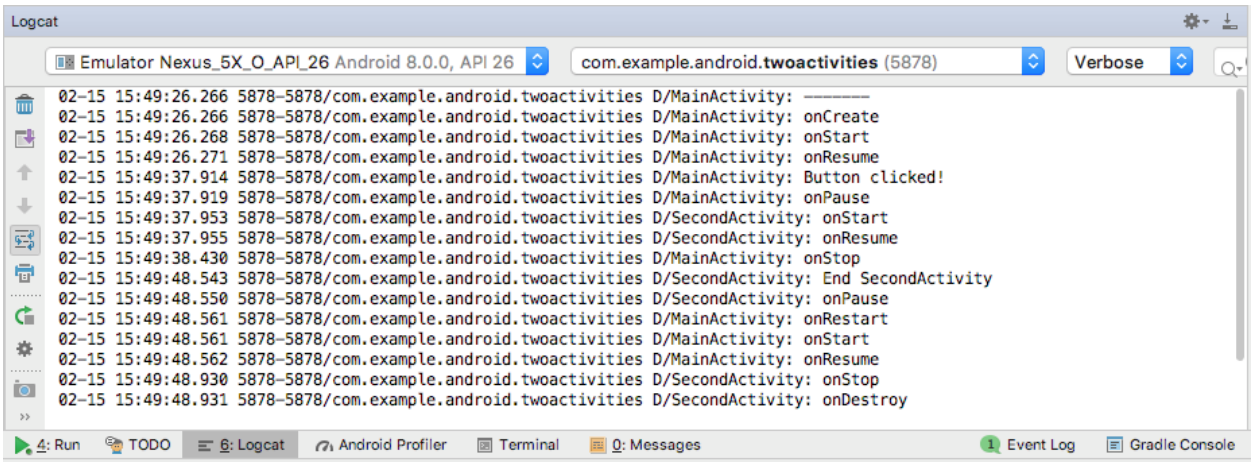
```
private static final String LOG_TAG = SecondActivity.class.getSimpleName();
```

3. Cài đặt các lifecycle callback (có thể copy từ MainActivity)
4. Trong phương thức returnReply(): thêm câu lệnh ghi log trước khi gọi phương thức finish():

```
Log.d(LOG_TAG, "End SecondActivity");
```

1.4 Quan sát Logcat khi ứng dụng chạy

1. Chạy ứng dụng.
2. Click vào tab **Logcat**.
3. Gõ "**Activity"** vào ô tìm kiếm.



Hãy thử các bước sau và quan sát Logcat:

- Sử dụng bình thường (gửi 1 message và reply lại).
- Sử dụng nút Back để quay lại MainActivity từ SecondActivity.
- Sử dụng mũi tên Up trên thanh tiêu đề ứng dụng (app bar) để quay lại MainActivity từ SecondActivity.
- Xoay thiết bị trên cả main và second Activity và quan sát điều gì xảy ra trên Logcat và màn hình.
- Nhấn vào nút hình vuông bên phải nút Home (overview button) và tắt ứng dụng.
- Quay về màn hình chính của điện thoại và khởi động lại ứng dụng.

Task 2: Lưu và khôi phục trạng thái Activity

Trạng thái (state) của mỗi Activity được lưu trữ dưới dạng một tập hợp các cặp key/value trong một đối tượng Bundle gọi lại Activity *instance state*. Hệ thống lưu thông tin trạng thái mặc định vào Bundle trước khi Activity dừng lại và truyền Bundle này vào thể hiện Activity mới để khôi phục.

Để không bị mất dữ liệu trong một Activity khi bị hủy hoặc tạo lại một cách không mong muốn, chúng ta cần cài đặt phương thức onSaveInstanceState(). Hệ thống gọi phương thức này trên Activity (giữa onPause() và onStop()) khi có một khả năng Activity có thể bị hủy hoặc tạo lại.

Dữ liệu được lưu trữ cho một Activity cụ thể trong phiên hiện tại của ứng dụng. Khi chúng ta dừng hoặc khởi động một phiên ứng dụng mới thì Activity instance state sẽ bị mất và Activity trở lại hình dạng mặc định của nó.

Nếu chúng ta muốn lưu trữ dữ liệu người dùng giữa các phiên ứng dụng thì chúng ta có thể sử dụng shared preferences hoặc một CSDL.

2.1. Lưu trạng thái Activity với onSaveInstanceState()

You may have noticed that rotating the device does not affect the state of the second Activity at all. This is because the second Activity layout and state are generated from the layout and the Intent that activated it. Even if the Activity is recreated, the Intent is still there and the data in that Intent is still used each time the onCreate() method in the second Activity is called.

In addition, you may notice that in each Activity, any text you typed into message or reply EditText elements is retained even when the device is rotated. This is because the state information of some of the View elements in your layout are automatically saved across configuration changes, and the current value of an EditText is one of those cases.

So the only Activity state you're interested in are the TextView elements for the reply header and the reply text in the main Activity. Both TextView elements are invisible by default; they only appear once you send a message back to the main Activity from the second Activity.

Chúng ta sẽ viết mã nguồn để lưu trữ trạng thái của 2 TextView bằng việc sử dụng onSaveInstanceState().

1. Mở **MainActivity**.
2. Ghi đè phương thức onSaveInstanceState() vào Activity (**Code > Override Methods**).

```
@Override
public void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
}
```

3. Kiểm tra nếu TextView mReplyHeaderTextView không bị ẩn thì đưa trạng thái visibiity vào Bundle bằng phương thức putBoolean() và key "reply_visible".

```
    if (mReplyHeadTextView.getVisibility() == View.VISIBLE) {
        outState.putBoolean("reply_visible", true);
    }
```

4. Tương tự, chúng ta thêm reply text vào Bundle.

```
outState.putString("reply_text",mReplyTextView.getText().toString());
```

2.2. Khôi phục trạng thái Activity trong onCreate()

Chúng ta có thể khôi phục trạng thái Activity trong onCreate(), hoặc cài đặt onRestoreInstanceState() (được gọi sau onStart() sau khi Activity được tạo).

1. Trong phương thức onCreate(), sau khi khởi tạo các biến tham chiếu đến các View bằng phương thức findViewById(), nếu saveInstanceState không null thì chúng ta lấy trạng thái visibility trong Bundle bằng key "reply_visible".

```
// Initialize all the view variables.
mMessageEditText = findViewById(R.id.editText_main);
mReplyHeadTextView = findViewById(R.id.text_header_reply);
mReplyTextView = findViewById(R.id.text_message_reply);

// Restore the state.
if (savedInstanceState != null) {
    boolean isVisible =
                savedInstanceState.getBoolean("reply_visible");
}
```

2. Nếu giá trị của key "reply_visible" là true thì cho hiển thị mReplyHeaderTextView và mReplyTextView, đồng thời lấy giá trị của key "reply_text" trong Bundle gán vào mReplyTextView.

```
if (isVisible) {
    mReplyHeadTextView.setVisibility(View.VISIBLE);
    mReplyTextView.setText(savedInstanceState.getString("reply_text"));
    mReplyTextView.setVisibility(View.VISIBLE);
}
```

3. Chạy ứng dụng: thử xoay thiết bị để chắc chắn reply message vẫn còn trên màn hình khi Activity được tạo lại.

Coding challenge

Thử thách: Tạo một ứng dụng shopping-list đơn giản với màn hình chính hiển thị danh sách sản phẩm do người dùng chọn, và một màn hình thứ hai cho danh sách các sản phẩm để người dùng chọn.

- Sử dụng 10 `TextView` rỗng để tạo thành danh sách các sản phẩm được người dùng chọn trên màn hình chính (main activity).
- Nhấn vào nút **Thêm SP** trên màn hình chính sẽ mở màn hình thứ hai chứa một danh sách các sản phẩm mua sắm (VD: **Gạo**, **Bơ**, **Táo**,…). Sử dụng các đối tượng `Button` để hiển thị các sản phẩm.
- Chọn một sản phẩm và quay về màn hình chính → cập nhật một `TextView` rỗng để chứa sản phẩm được chọn.

Sử dụng `Intent` để truyền thông tin giữa các `Activity`. Và đảm bảo các trạng thái hiện tại của danh sách được chọn vẫn còn hiển thị trên màn hình khi người dùng xoay thiết bị.

Homework

1. Tạo một ứng dụng với bố cục gồm một counter `TextView`, một `Button` để tăng counter, và một `EditText`.

2. Xử lý sự kiện click vào `Button` để tăng counter.
3. Chạy ứng dụng và nhấn vào Button để tăng counter. Nhập một vài dòng chữ vào `EditText`.
4. Xoay thiết bị và quan sát màn hình *(Counter bị reset, nhưng `EditText` thì không).*
5. Cài đặt phương thức `onSaveInstanceState()` để lưu trạng thái hiện tại của ứng dụng.
6. Cập nhật `onCreate()` để khôi phục trạng thái của ứng dụng.
7. Đảm bảo trạng thái của ứng dụng được lưu lại khi chúng ta xoay thiết bị.