Mobile Application Lab 3

Instructors: Legand Burge, Asad Ullah Naweed

Date: 08/23/2018

Topics: Activities and Lifecycles

Be sure to click on the links in this doc to view more information about certain topics. These may or may not help you in your lab, but can always provide some extra context and information.

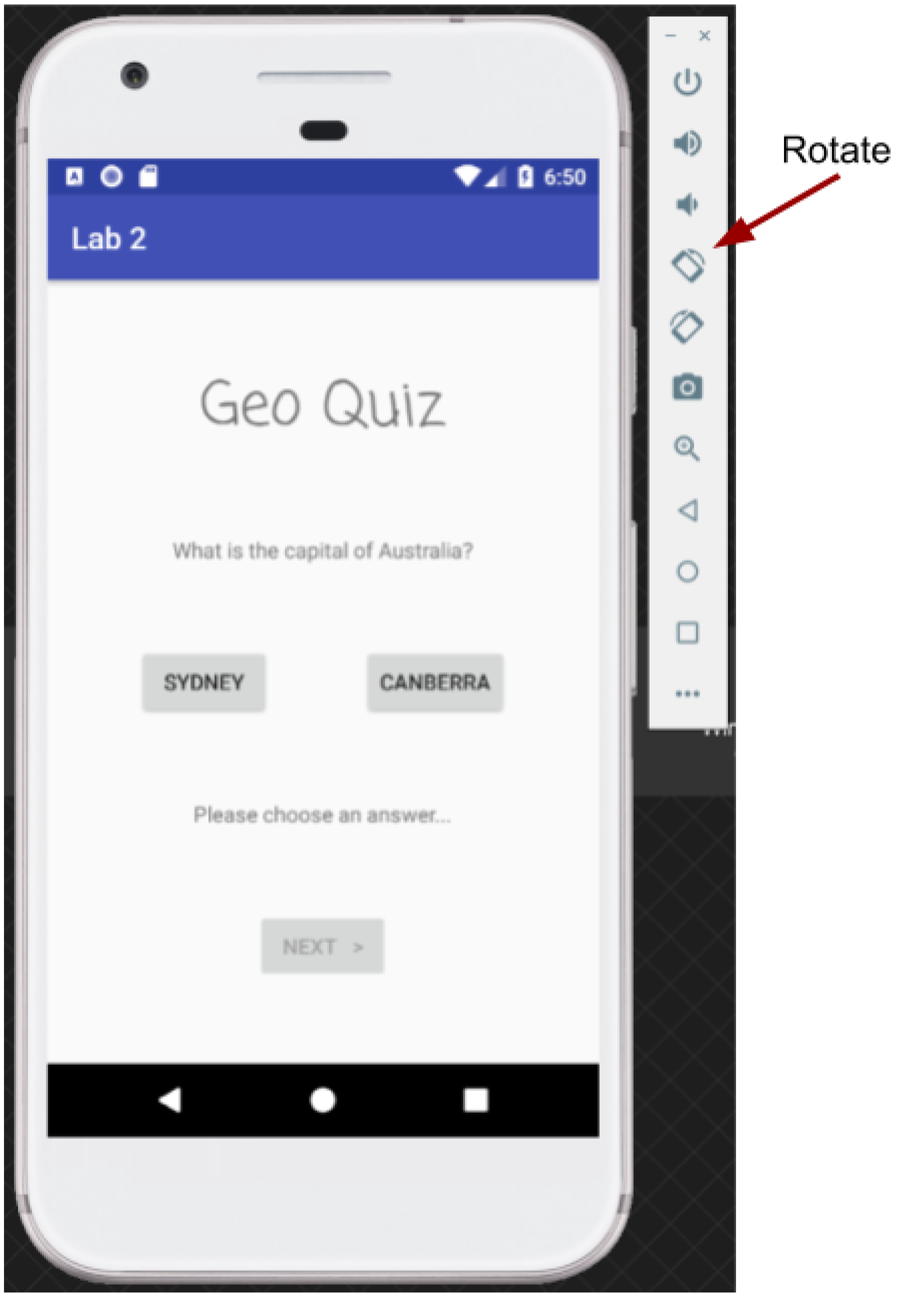
Initial Setup

1. Create a copy of your project from Lab 2 (Geo Quiz) and open it up in Android Studio.
2. Open the AVD Manager, and launch a new Android Virtual Device running Android Oreo.
3. Build and run your newly created project on the Virtual Device.

For this lab, you will use your project from Lab 2 as a starting point and iterating on top of that. There are a few issues with that app which we will identify and fix. We're also going to add a final screen when the user has gone through all the questions.

Rotated Geo Quiz App

Build and run the Geo Quiz app in the Android Emulator. Answer a couple of questions, and then rotate the emulator device by clicking on this button:



Notice what happens: The quiz resets to the initial question when the device rotates. We're going to investigate what happens.

First, in your `MainActivity` class, override the `onStart`, `onDestroy`, `onStop` and `onPause` methods. You can either write code for them manually, or get the IDE to generate these overrides for you. Recall the instructions from Lab 2 about how to get Android Studio to generate code.

In the overridden methods, make sure to always call the superclass's implementation of that method before adding more stuff to it, or your application will misbehave. If you got the IDE to generate these methods for you, they should already contain those initial lines.

Now, add a single logging statement to each overridden method. An example is given below:

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

After you've added these lines, build and run your app again, and also bring up Logcat inside Android Studio. This should be a button on the toolbar at the bottom of your Android Studio window.

Pay close attention to the output shown in the Logcat console, and what happens when the app starts, and when you rotate the emulator. Try and see if you can identify the problem.

Now, you'll need to override another method in the `MainActivity` class, called `onSaveInstanceState`. Overriding this method will allow you to save some state variables of the Activity before it is destroyed. You can set named properties in the `Bundle` argument to this method, and those properties will be made available in the `Bundle` that is an argument of the `onCreate` method.

Add the following to your `MainActivity` class:

```
@Override
protected void onSaveInstanceState(Bundle outState) {
    super.onSaveInstanceState(outState);
    outState.putInt("CurrentIndex", currentQuestionIndex);
}

@Override
protected void onCreate(Bundle savedInstanceState) {
    ...
    currentQuestionIndex = 0;
    if (savedInstanceState != null) {
        currentQuestionIndex = savedInstanceState.getInt("CurrentIndex");
    }
}
```

Note what's happening. The `onSaveInstanceState` method is called just before the Activity is destroyed, giving the components a chance to save state. The saved state is made available when the Activity is recreated.

Build and run your app again, and see what happens if you rotate your device now.

Scoring the Quiz

Previously, in the app, we were not keeping any kind of score, and users were also able to change their answers before clicking the Next button. In this part, we are going to fix both those issues. Additionally, once the whole set of questions has been exhausted, a new Activity will be launched, showing the score and a button to retake the quiz.

As an exercise, first implement the scorekeeping, and also disabling of the answer buttons once a button has been pressed. Do not proceed without completing this task.

Now, take a look at your app's directory, and right-click on the package that contains your MainActivity. Create a new Empty Activity, and call it ScoreActivity. This will do the following:

1. Create the `ScoreActivity` Java file.
2. Create the `activity_score.xml` file containing the layout for that Activity.
3. Add a new line for the activity in the `AndroidManifest.xml` file.

Make sure that all three of these three things are done before proceeding.

Now, create a layout for your ScoreActivity. Copy-paste the following XML into your `activity_score.xml` file, replacing all of its existing contents.

```
<?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=".ScoreActivity">

    <TextView
        android:id="@+id/score_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Score: "
        app:layout_constraintBottom_toTopOf="@id/again_button"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/again_button"
        android:text="Take Quiz Again"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintTop_toBottomOf="@id/score_text"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"/>

</android.support.constraint.ConstraintLayout>
```

Now, go back to MainActivity, and go to the method that is invoked when the Next button is pressed. Update that method to look like the following:

```
private void onNextButtonPressed() {
    currentQuestionIndex++;
    if (currentQuestionIndex < questionList.size()) {
        updateView();
    } else {
        Intent scoreIntent = new Intent(this, ScoreActivity.class);
        startActivityForResult(scoreIntent, 0);
    }
}
```

Here, you are creating an Intent. An Intent is a representation of a request that an application makes to the underlying OS. The information in the intent is then used to perform the specific operation requested. In this case, the requested operation is the launch of `ScoreActivity`. The `ScoreActivity` will be launched as a child Activity of `MainActivity`.

Take a look at the documentation for Intent to see what the constructor is doing. Also take a look at the documentation for startActivityForResult to see what the two arguments to that method mean.

Build your app and see what happens now when all you manage to answer all the questions. You should see a new Activity pop up when you answer all the questions.

You now need to do the following:

1. Keep score.
2. Send the score from `MainActivity` to `ScoreActivity`.

3. Send back a signal from `ScoreActivity` to `MainActivity`.
4. Restart the quiz in `MainActivity` when the signal from (3) is received.

Keep Score

Add the following new instance variable to your `MainActivity` class:

```
private int currentScore;
```

As an exercise, put in the logic for initializing and updating the score yourself. Identify the places where it needs to be incremented and where it needs to be reset.

Send score to ScoreActivity

First, add the following new variable in `MainActivity`:

```
public class MainActivity {
    ...
    static final String KEY_SCORE = "Score";
}
```

Now, in your callback for the Next button, add the following:

```
private void onNextButtonPressed() {
    currentQuestionIndex++;
    if (currentQuestionIndex < questionList.size()) {
        updateView();
    } else {
        Intent scoreIntent = new Intent(this, ScoreActivity.class);
        scoreIntent.putExtra(KEY_SCORE, currentScore);
        startActivityForResult(scoreIntent, 0);
    }
}
```

What you're doing here is attaching an "extra" to the intent. This extra will be accessible from `ScoreActivity`. Inside the `onCreate` method of `ScoreActivity`, add the following:

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

    TextView scoreText = findViewById(R.id.score_text);
    int score = getIntent().getIntExtra(MainActivity.KEY_SCORE, 0);
    scoreText.setText("Quiz Score: " + score);
}
```

Take a look at the documentation for Intent to see what the second argument of the `getIntExtra` method means.

Build and run your app. You should now see the score displayed in the ScoreActivity.

Send a signal from ScoreActivity to MainActivity

You now need to define a callback function for the button in ScoreActivity, so that it finishes. Add the following to your ScoreActivity class:

```
public class ScoreActivity extends AppCompatActivity {

    static final String KEY_RESTART_QUIZ = "RetakeQuiz";

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

        Button againButton = findViewById(R.id.again_button);
        againButton.setOnClickListener(v -> onAgainButtonPressed());

        TextView scoreText = findViewById(R.id.score_text);
        int score = getIntent().getIntExtra(MainActivity.KEY_SCORE, 0);
        scoreText.setText("Quiz Score: " + score);
    }

    private void onAgainButtonPressed() {
        Intent data = new Intent();
        data.putExtra(KEY_RESTART_QUIZ, true);
        setResult(Activity.RESULT_OK, data);
        finish();
    }
}
```

Note that here, you're creating an empty Intent. This is created for the sole purpose of containing extras that will be passed back to the MainActivity when ScoreActivity finishes. The call to `finish()` will finish the `ScoreActivity`, and resume its parent Activity, which in this case is `MainActivity`.

Restarting the Geo Quiz

How does the `MainActivity` know when a child activity has finished with a result? The OS will inoke its `onActivityResult` method. So, you should now override that method, just like you've overridden other Activity lifecycle methods like `onCreate` and `onResume`.

You should add the following block of code to your MainActivity class:

```
public class MainActivity {
    ...
    @Override
    protected void onActivityResult(
            int requestCode, int resultCode, @Nullable Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (resultCode != Activity.RESULT_OK || requestCode != 0 || data == null) {
            finish();
        }
        // More code here...
    }
}
```

The `data` argument is the same Intent that you created in the `ScoreActivity` class. As an exercise, write the code required to get the boolean value set in ScoreActivity, and restart the quiz if it is true.