Project 1: Choose Your Evil Overlord

Due Wednesday January 21 at 11:59pm

All projects in this course are submitted via CourSys(http://courses.cs.sfu.ca). Submit a ZIP file of all project deliverables via CourSys by the project deadline. For every 24 hours past the original deadline, 10% of the original score shall be taken away, up to a maximum of two days.

This assignment is to be done individually. Do not share your code or solution. Do not copy code found online. Do not post questions about the assignment online. Please direct all questions to the instructor or TA(cmpt-276-help@sfu.ca). You *may* make use of any code provide by the instructor, or in help guides/documentation provided by the instructor.

Installation

Install and configure an Android development environment on a computer. Android makes use of and thus requires the Java Development Kit, as well.

1) Download and install the latest Java JDK (which should include the JRE): http://www.oracle.com/technetwork/java/javase/downloads/index.html
2) Download and install Android Studio from the Android development site. Note that this is the "All Android Studio Packages" download. http://developer.android.com/sdk/index.html
3) Installing Android Studio on your personal computer, but you should still be able to complete assignments using the CSIL lab at SFU Surrey.

Resources

There are several resources that you may find helpful in completing this assignment:

- Chapters 1-6 of the Android in 24 hours book.
- Dr Fraser's Videos on Android App Development:
  - Button Tutorial
  - UI Layouts
  - Troubleshooting
- Toast Documentation
- Logging Documentation

Choose Your Evil Overlord Application

Evil overlords are holding a vote and you have to choose a side. You also have to develop the Android application that will *let* you choose a side. You will create an Android application adhering to the following specifications.

- Your application must target one of Android verions: 2.3.3, 4.2, 4.3, 4.4. 2.3.3 may run more smoothly on your computer if you have performance problems.
- Name your application "EvilOverlords". Feel free to customize other aspects of your app like its icon.
- Have the main screen look like the screenshot in Figure 1 with:
  - The given title text centered at the top
  - Two `TextViews`, centred vertically, showing "Daleks" on the left and "Cylons" on the right. The left text must be blue, the right text must be red.
  - Two buttons at the bottom left and right corners of the screen, both titled "Support". These buttons allow you to back one of the two vast, evil empires seeking to take over the world. Choose wisely.

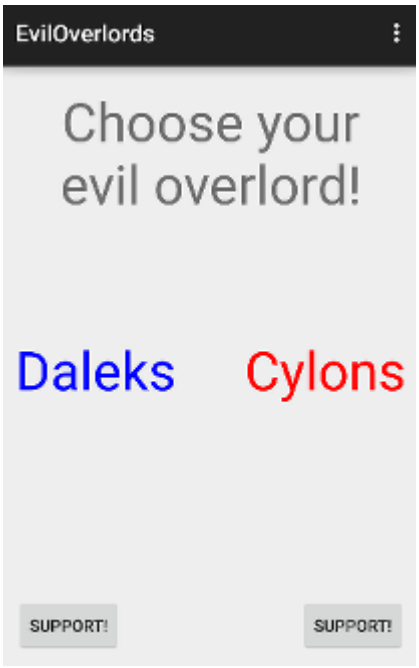

Figure 1: Appearance of the main activity

In order to show your support, the buttons must be wired up to recognize your vote. When the left "Support" button is clicked, the app must:

1) Display a Toast message with the string "Your Dalek vote has been exercised".
2) Log the same string to the LogCat.

When the right "Support" button is pressed, the app must take the same steps but instead using the string "By your command, you now support the Cylons".

In addition to the above required behaviors, you must use good Android app development practices. All strings that appear in the TextViews or as labels on Buttons must not be hard-coded in the `.java` files; they must be stored as resources in `strings.xml`. You do not need to do this for your Toast/Log messages, but you should feel free to do so. The colors for the blue and red text must be placed in colors.xml (see Chapter 3, "Working with Colors").

Showing it in action

To prove to the evil overlords that your app is working, you will need to capture a screen shot that shows the app in action. This is also an excellent opportunity for you to get experience using the *debugger* in Android Studio.

In the emulator, click your application's left and right "Support" buttons to trigger the messages in the LogCat. Make sure the LogCat view is visible in Android Studio and showing the log messages from your application. Stretch its view wide enough so that the whole Log message your application prints is visible.

Now set a breakpoint in your `onCreate()` method. You can do this by clicking to the left of a line of code in `onCreate()`. Debug the application and step through the `onCreate()` method while it's running on the emulator. You can do this by clicking one of the stepping controls in the debugging display:

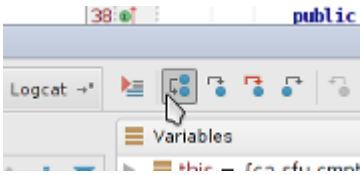

Figure 2: Stepping controls for debugging

Resume the app using the green arrow to the left. Then close and reopen your application within the emulator (use emulator's back button to close it, relaunch by finding it in the all programs list) and notice whether it triggers your breakpoint again. Can you explain why? Take a single screenshot showing

Android Studio with the debugger running, with your application stopped at the breakpoint in `onCreate()`, with the LogCat view visible, and with your two Log messages shown. Create a new folder in your project named `docs/`. Save the screenshot into this folder.

Discussion Questions

Locate your project directory with the Android Studio Projects directory and create a subdirectory called `docs`. Create a plaintext file called `DISCUSSION.TXT` in the `docs/` folder. In it, describe each of the following in your own words using a sentence or two for each.

1) Android SDK
2) Android Studio
3) Android Manifest
4) Android Emulator
5) Purpose of `MainActivity.java`
6) Purpose of `activity_main.xml`

Deliverables

Submit a ZIP file of your Android Studio project through CourSys.

Locate your project directory in the filesystem. In Android Studio, you can find the path to your project by right clicking on your project name in the path bar:

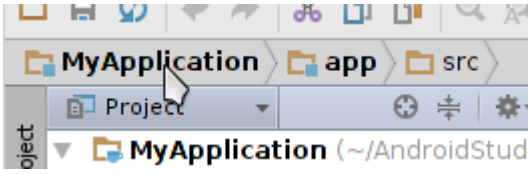

Figure 3: Project name in the path bar

and selecting "Copy Path".

Create a ZIP file containing this directory. This ZIP file should contain not only your code and `.xml` files, but also the `docs/` folder including the screenshot and `DISCUSSION.TXT` file. In Linux, this can be done with the command:

```
zip -r project1 ProjectName/
```

This will create `project1.zip` containing the directory `ProjectName` and all of its subdirectories.

Please remember that all submissions will automatically be compared for unexplainable similar submissions. Everyone's submissions will be quite similar, given the nature of this assignment, but please make sure you do your own original work; we will still be checking.