

Ser423/Cse494

Mobile Computing

Lab 2 iOS and Android App and View Lifecycles

Due by midnight Wednesday January 27, 2016. No late submissions will be accepted.

Changes since distribution:

- No changes.

Description

You are again to construct both an Android and iOS app. These apps should demonstrate lifecycles on these platforms. Here is a description of the two apps.

Android

In the Android app, create a two activity app. One activity is the **MainActivity** and the other is a "manually" created alert Activity (**AlertActivity**, as demonstrated in class. Both activities should override the following activity lifecycle methods: **onRestart**, **onStart**, **onResume**, **onPause**, **onStop**, and **onDestroy**. The only actions in each of these methods are to call the parent's corresponding (overridden) method, and to log the name of the class and the name of the method being called. That is, call

android.util.Log.w

passing the string name of the class (**this.getClass().getSimpleName()**) and the string name of the method. The **MainActivity** and its corresponding view (**layout/activity_main.xml**) should include a **button** and a simple message. When the button is clicked, create a new **Intent** for the **AlertActivity** and start the activity associated with the intent. The **AlertActivity** should only take up a portion of the screen (about a third) and include a message, such as "Hello Android Developer", and a **Button** labelled **OK**. Clicking the **OK** button should cause the **AlertActivity** to **finish**.

iOS

In the iOS app, implement as close to the same functionality as you provide in the Android app. You are free to use a built-in alert for the **iOS** app if you must, but you are not permitted to utilize a built-in dialog or alert functionality for the **Android** app. The **iOS** app should produce log messages in as much as possible similar to the **Android** app.

Your UI's should primarily utilize the layouts and simple widgets introduced in the class, and you may form them based on the examples given in class. Your solutions should follow the documentation and packaging constraints outlined in the prior assignment, which are repeated below.

1. Place all of your application source code into the package named: **edu.asu.bsse.asuriteid.appname** where **bsse** designates your academic program: **bscse**, **msse**, **bsse**, **bscs**, which stand for **BS Computer Systems Engineering**, **MS Software Engineering**, **BS Software Engineering**, or **BS Computer Science** respectively. **asuriteid** is your asurite id.
2. As part of all class header comments, that you create and turn-in this semester, include a copyright notice, such as: **Copyright 2015 Your Name**.
3. As part of the class header comments, include a **right to use** statement. The examples presented in class use the Apache License Version 2, but you should put whatever rights you prefer. At the very least, you must provide the instructor and the University with the right to build and evaluate the software package for the purpose of

determining your grade and program assessment.

4. As part of the class header comments, include a reference to the software's author, such as:
@author FirstName LastName mailto:FirstName.LastName@asu.edu.
5. As part of the class header comments, include an indication of the software version, such as:
@version March 2, 2015

These comments are required on all code that you generate this semester. If you hand in an assignment that does not include them, it will not be graded. Here are the **StudentJson** classes that were constructed during the first class session as part of the example **Android** and **iOS** apps. See: [studentJsonClasses.jar](#). Use java's **jar** command to extract this file. It will create a folder containing a **playground** that can be opened in **Xcode** and a **Java** class in the file **StudentJson.java**.

What To Hand-In

The posted quiz does not need to be submitted this week, and should be used as a study guide.

Structure your project as two sub-directories of the folder named **Assign2MyASURITEID**. One sub-directory for the **iOS** app and the other for the **Android** app. You will submit this project, by first cleaning it (to remove all generated files. Then create a **jar** or **zip** archive of the project (Assign2*) directory. You can create a **Java Archive (jar)** by executing the following command from a terminal in the directory which is parent to the project directory:

```
jar -cvf Assign2MyASURITEID.jar Assign2MyASURITEID/
```

That archive will then be submitted via **Blackboard**. See the Content section.

Grading Criteria

- **5 points.** Your solutions build using Xcode and Android Studio. And, your UI's appear properly in the simulator with an **iPhone 6** and Android emulator with a **Nexus5-API23**.
- **5 points.** Your app should define and properly use an **alert** dialog.
- **5 points.** Your solution includes comments described above as headers for each class used in the app.
- **5 points.** Your solution generates log messages that demonstrate the app/view lifecycle activities outlined above and in class.

Email: Tim.Lindquist@asu.edu | [Ser423 Home](#)