CREATING AN ANDROID APP USING GLUON AND ECLIPSE

Jeffrey Stephen, Jeevan Bains

BCIT

Contents

Introduction

Gluon is an Eclipse *plugin* that can convert *JavaFX* code to an *android APK*, native *IOS* code, or a desktop application.

The benefit of using gluon is that you are able to directly develop applications for various platforms, without having to use another IDE such as *Android* Studio and the fact that you can create *IOS* apps, without have to learn a new language such as Swift.

This instruction set is aimed at Casual users with an understanding of Java, as well as the Eclipse IDE.

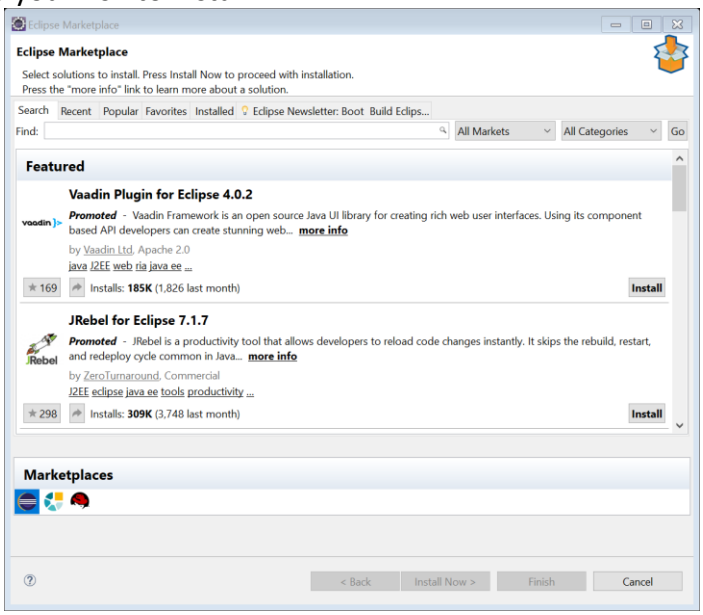
Setup/Creating your first gluon project

This section will focus on setting up the gluon *plugin* using the Eclipse marketplace, as well as creating your first project.

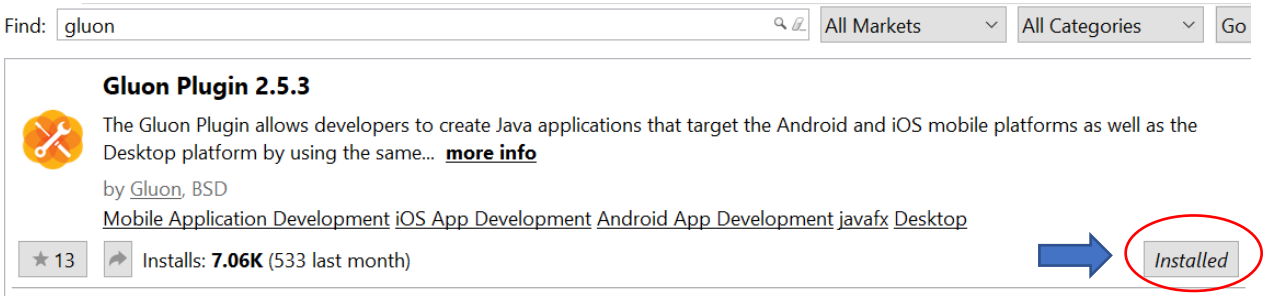
Note: You must have Java and the Eclipse IDE installed before proceeding with these instructions.

Setup

1) Open the Eclipse market place: [Help]>[Eclipse Marketplace]
   At this point, the Eclipse marketplace should be open, and you should now be able to search for *plugins* you wish to install.

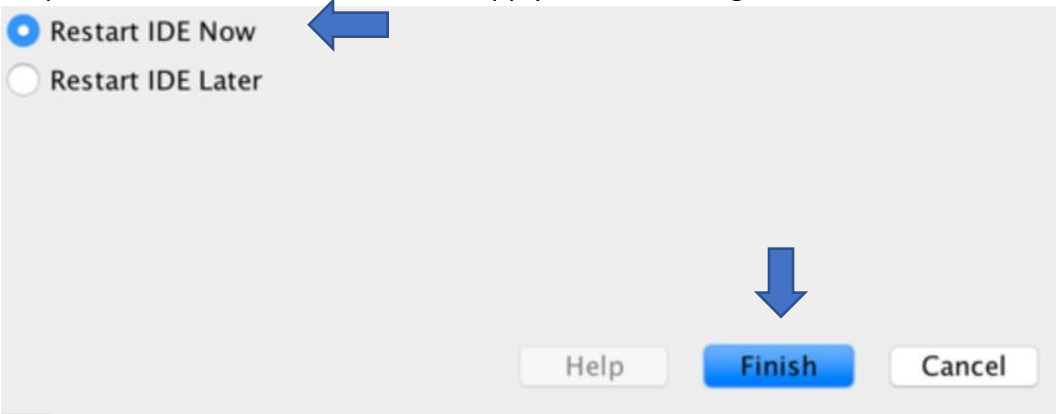


2) In the “find” search bar, type gluon then click install on the first *plugin* that appears.



Note: Where the image says “Installed”, if it is your first time downloading the program it should say “Install” and redirect you to a setup wizard.

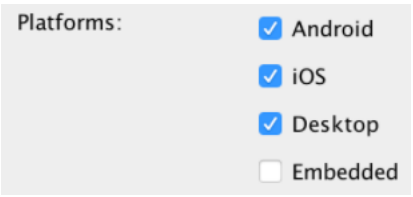
3) Follow the install wizard steps, and once the final step is reached choose to restart the Eclipse IDE now and click “Finish” to apply the new changes.



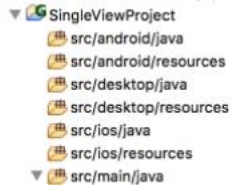
Creating your first Gluon project

1) Create a new project: [File]>[New]>[Project]
When choosing the type of project to create, navigate to the Gluon folder and select “Gluon Mobile – Single View Project”, and click finish

2) Name the project and select the platforms you wish to develop for.



After creating the project, you should now see a new project in the file directory of Eclipse. There will be multiple views that control different aspects of the project, this will be discussed in detail later.



Note: When creating your first project, you will be asked for an email and a license key, a free licence key can be obtained at http://gluonhq.com/products/mobile/buy/

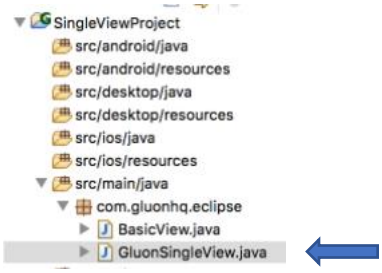
Coding your first “hello world” project

After creating your gluon project, there may be up to four different views depending on what options where selected when creating the project.

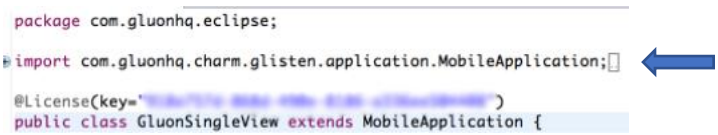
These views can compose of one or more of the following: Main/Desktop/*Android*/*IOS*
Each view is for specific code exclusively for that platform, whereas main contains the code that will be used across all platforms.

This section will focus on the “Main” view of the project

1) Navigate to the “GluonSingleView.java” *class* in the file explorer

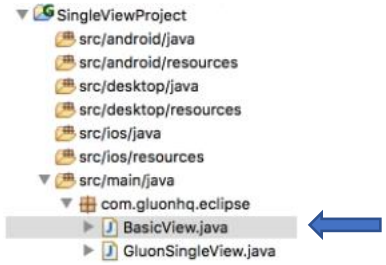


2) Import the Gluon mobile application *package*:
“import com.gluonhq.charm.glisten.application.MobileApplication;”



Note: This must be done at the top of the *class* file before the *class* is defined

3) Navigate to the “BasicView.java” *class* in the file explorer



4) Import the Gluon listener *package*:
“import com.gluonhq.charm.glisten.control.Appbar;”

Note: This must be done at the top of the *class* file before the *class* is defined

5) Now within the "BasicView" *class*, create a label, that says "hello JavaFX world" as shown below.

```
Label label = new Label("Hello JavaFX World!");
```

6) Create a button to change the text, whenever it is clicked. This can be done in a series of steps, including creating the button then setting the button to do an action when it is pressed.


```
Button button = new Button("Change the World!");
button.setGraphic(new Icon(MaterialDesignIcon.LANGUAGE));
button.setOnAction(e -> label.setText("Hello from Eclipse!"));
```

Note: the middle line shown in this code is optional and does not change the functionality of the button. It is just used to add a visual component to the application.

7) Create a VBox and add the text, and button to it as shown below.

```
VBox controls = new VBox(15.0, label, button);
```

Note: A VBox is needed to add elements that you want to be displayed on the final working application.

8) Center the VBox to the middle of the page as shown below.

```
controls.setAlignment(Pos.CENTER);
```

9) Display the VBox on the application, so it is visible when using the app.

```
setCenter(controls);
```

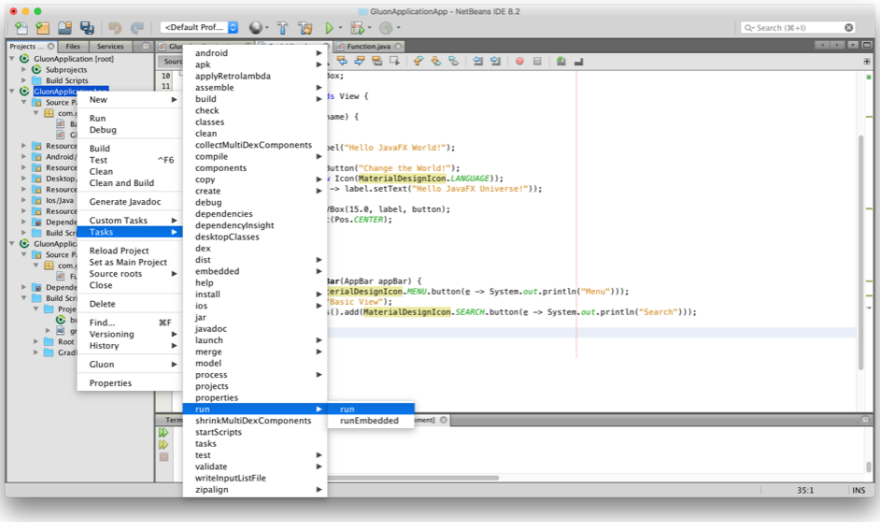
Note: This must be done for the application to have anything displayed.

After this step, the application has been created and you are now able to *compile* and test the application for the various platforms. Compilation of an application will be talked about in more detail in the following section.

Compiling

Before deploying the program to a mobile device, it is always good to check how it runs on the desktop or laptop that you are using

1. Click on window in the taskbar
2. Select tasks in the dropdown
3. Select run -> run (*This step allows you to test the application on the desktop*)



4. Verify that the program works as you planned, if everything works as it should we can move on to deploying this application to an *android* device

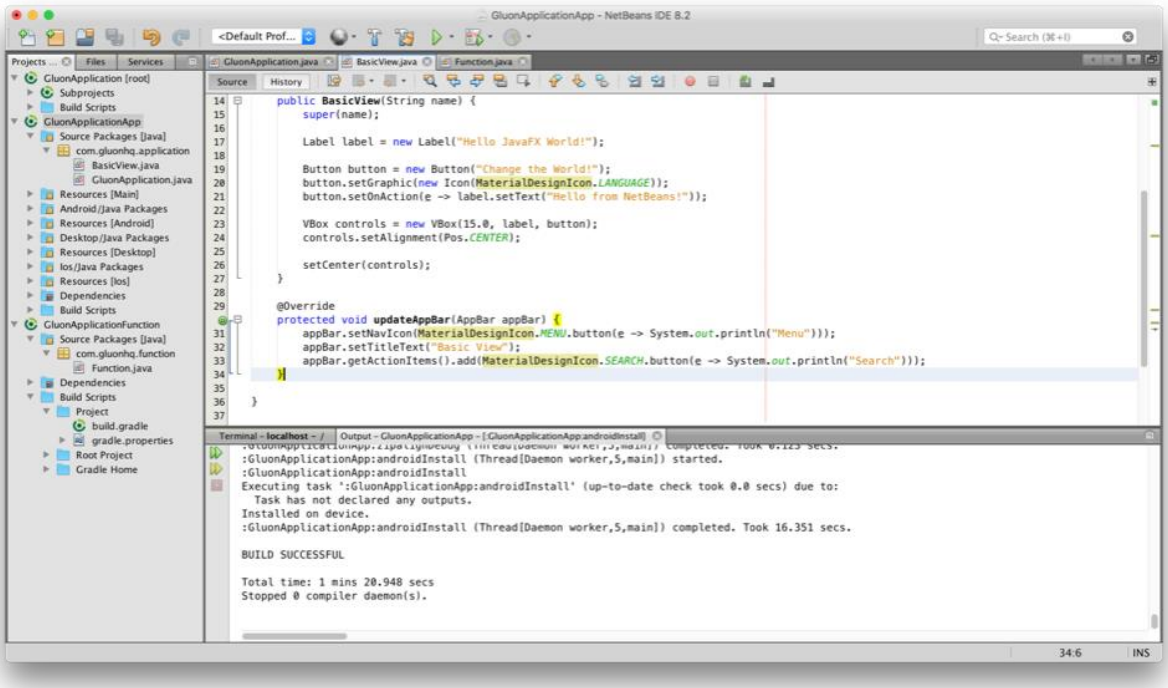
Deploying to an *android* device

To summarize, these are the most important tasks added by the jfx-mobile plugin:

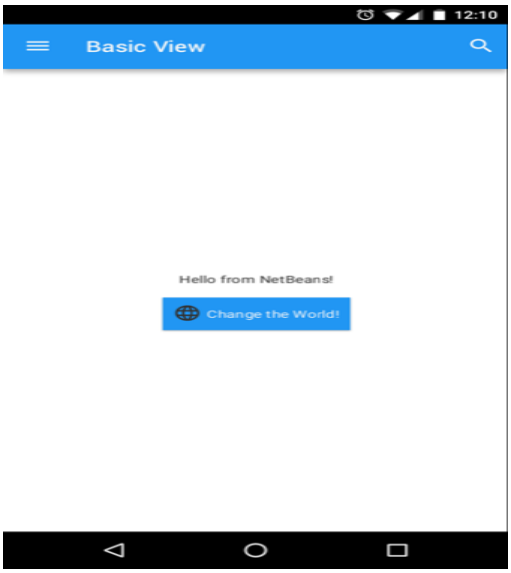
android	Generates a debug Android apk containing the JavaFX application.
androidInstall	Launches the application on a connected android device.
androidRelease	Generates a release Android apk containing the JavaFX application.
launchIOSDevice	Launches the application on a connected ios device.
launchIPadSimulator	Launches the application on an iPad simulator.
launchIPhoneSimulator	Launches the application on an iPhone simulator.
runEmbedded	Launches the application on a remote embedded platform.

1. For *android* click on the task lists, and select *android* -> *android*, this will create the *APK*, you can also do *android* -> androidInstall, if the device is connected to the computer

2. After the build is complete, the application should be installed on the device. Otherwise, check your console for any errors.



3. Once successful find the application on your mobile device and open it up



Glossary

Package – A bunch of classes related to each other or part of a similar project

Class – it is a basic building block of object oriented languages such as java, it is a template that describes the data and behavior associated with instances of that class

Compile – the process of converting code to machine code

JavaFX – A graphical user interface based in Java

Plug in – an add on extension used to help speed up longer tasks through pre coded shortcuts

Android – a type of operating system that is based in java

IOS – the operating system that apple uses for their mobile devices

APK - Android Package is the package file format used by the Android operating system for distribution and installation of mobile apps and middleware.

Troubleshooting

Compiling error	Check that the code is formatted right, make sure every call is legal in code, usually eclipse does this check for you and highlights it if it is illegal or cannot compile.
Installation error	Java should be up to date, current version is 8 or higher, check internet connection and stability.
App not showing on the device	Check if the right command is used for the right type of device
Certain lines not responding	Check that all the required packages are imported properly, without an import some commands would not work properly, in some cases not even compile.
Overridden methods are not functioning	Check that the method is overridden properly and called properly, eclipse would not necessarily find the compiler error, as it could be syntactically right but logically wrong.