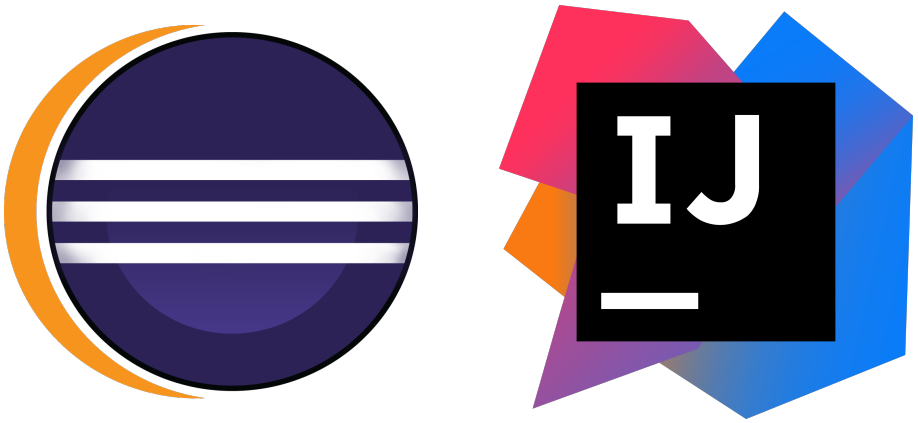
Graphics Ticks: Working with IDEs

Eclipse and IntelliJ



Preliminary

In this document we detail how to set up your Eclipse or IntelliJ environment for the OpenGL graphics ticks. We will show you how to correctly import your source files, and correctly set up your project so that it can reference the Lightweight Java Games Library (LWJGL) and the Java OpenGL Math Library (JOML) required for our OpenGL implementation.

First, please download the relevant `.zip` file from Moodle and extract it to your working directory.

Please refer to ***section 1*** if you are using Eclipse, and ***section 2*** if you are using IntelliJ. Ensure that you have followed this guide correctly and have a working OpenGL implementation before coming to any help sessions.

1 Eclipse

Please download Eclipse from `www.eclipse.org/downloads/packages/eclipse-ide-java-developers/oxygenr`, and install it. When you run it, you may be greeted with a dialogue asking you to choose a workspace – the default is fine.

Creating your project and importing libraries

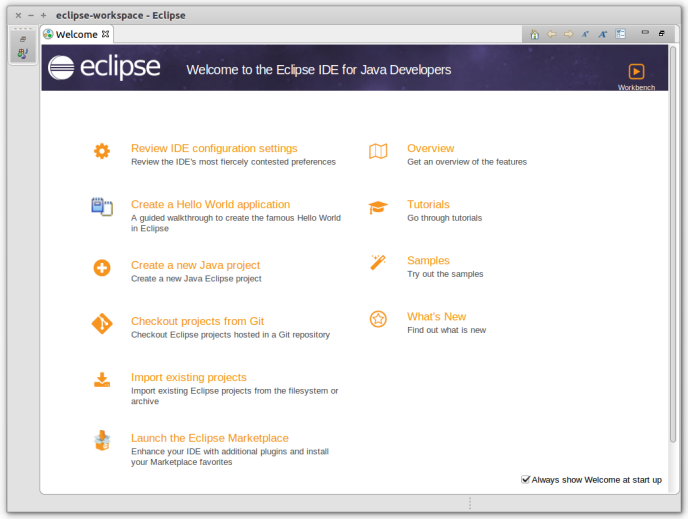


Figure 1: *the Eclipse welcome screen.*

On opening Eclipse, you should be see a welcome screen as shown in ***Figure 1***. From here, please do the following:

1. Click on Create a new Java Project (alternatively go to File → New → Java Project).
2. In the Create a Java Project window (***Figure 2***):
   - Give the project a name.
   - Untick the Use default location option.
   - Browse to the folder you extracted from the tick's `.zip` file – this is the one that contains the `src` folder.
   - Click Next.
3. In the Java Settings window:

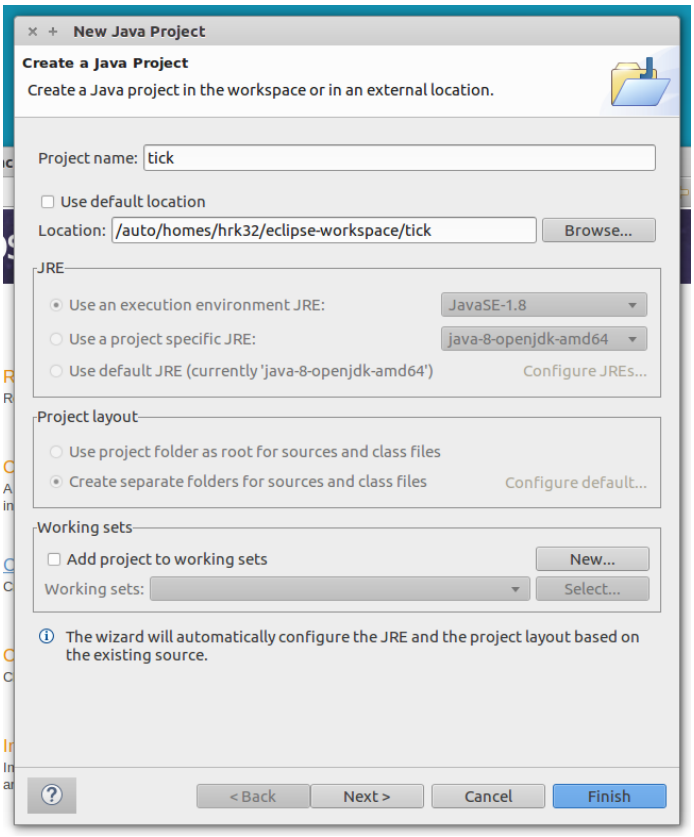


Figure 2: *the Eclipse* Create a Java Project *window.*

- Under the Source tab, ensure the source folders appear as in (***Figure 3***).
- Under the Libraries tab (***Figure 4***), you will see Eclipse has automatically added all the `.jar` files in the `lib` directory. If you are configuring Tick 1, ensure that the `lwjgl.jar`, `lwjgl-util.jar`, and `joml.jar` appear.
- Click Finish.

1.1 Running your project

You should now be at a blank workspace, with the tick shown in the Package Explorer in the left pane. Navigate to the `Tick.java` file which contains our `main()` method and open it. You should then be able to run the project by clicking the green play button in the top toolbar (***Figure 5***). For Tick 2, this will give an error: Exception in thread "main" java.lang.UnsatisfiedLinkError: no

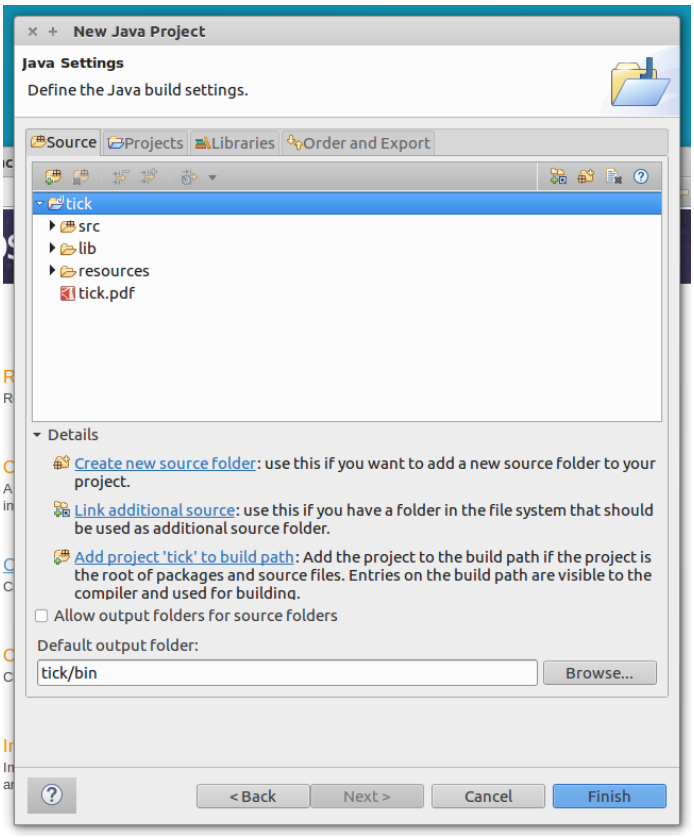


Figure 3: *the Eclipse source file selection.*

lwjgl(64) in java.library.path. This is because the JVM cannot locate the native libraries required to use LWJGL. To correct this, please do the following:

1. Click on the dropdown menu next to the green play button, and click Run Configurations... (***Figure 6a***).
2. In the window that appears, add the following to the VM Options field:

   `-Djava.library.path=lib/lwjgl/native`

   See (***Figure 6b***).

You should now be able to run your program without error. Ensure an OpenGL window opens correctly as described in your tick.

You should now be ready to begin working on the tick.

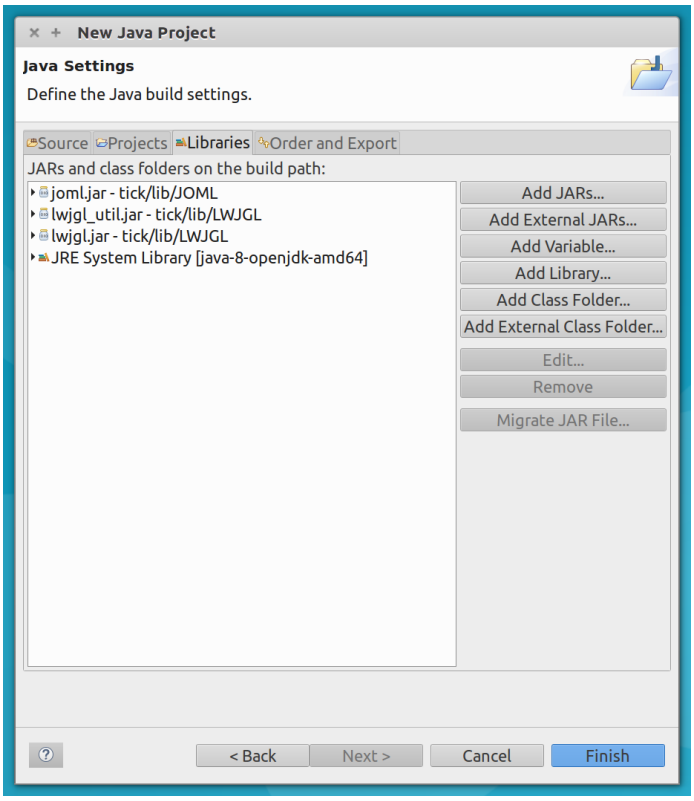


Figure 4: *the Eclipse library selection.*

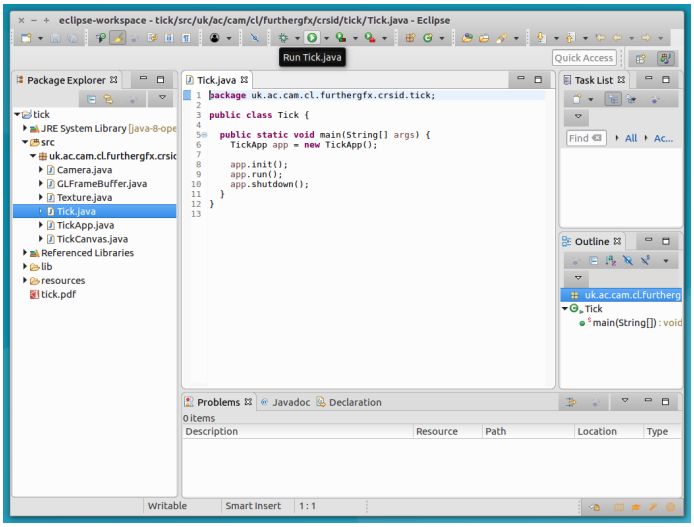
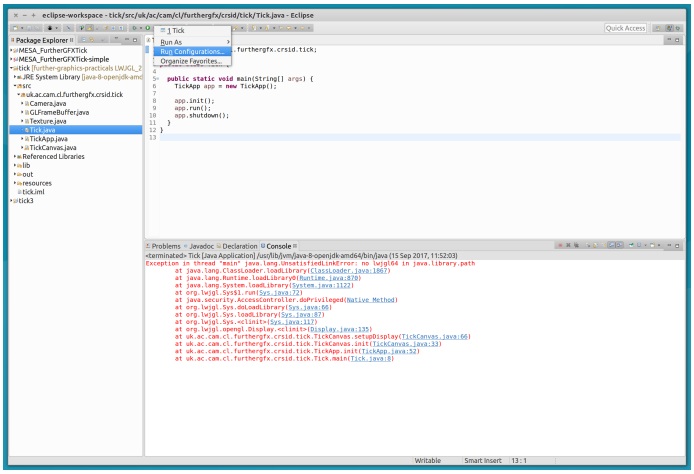
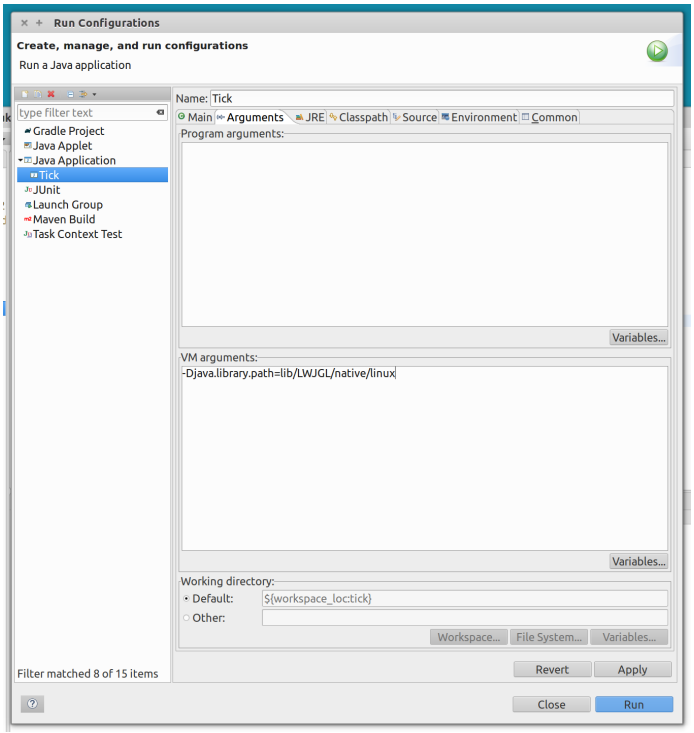


Figure 5: *the Eclipse workspace.*



(a) *editing run configurations in Eclipse.*



(b) *changing VM options in Eclipse.*

Figure 6: *finding the native libraries in Eclipse.*

2 IntelliJ

Please download IntelliJ from `www.jetbrains.com/idea/download`, and install it.

2.1 Creating your project and importing libraries

Figure 7: *the IntelliJ splash screen.*

On opening IntelliJ, you should be see a splash screen as shown in ***Figure 7***. From here, please do the following:

1. Click on Import Project (alternatively go to File → New → Project from Existing Sources...).
2. Select your folder you extracted from the tick's `.zip` file – this is the one that contains the `src` folder.
3. Select Create project from existing sources (***Figure 8***), then click Next.
4. Give the project a name, and ensure the project location is still set to the folder you selected (***Figure 9***). Click Next.
5. Ensure the source files are as shown in ***Figure 10***. Click Next.
6. If you are configuring Tick 2, ensure the LWJGL and JOML libraries are selected as shown in (***Figure 11***). The LWJGL libary should contain two `.jar` files (lwjgl.jar and lwjgl-util.jar), and the joml library should contain a single `.jar` file (joml.jar). Click Next.

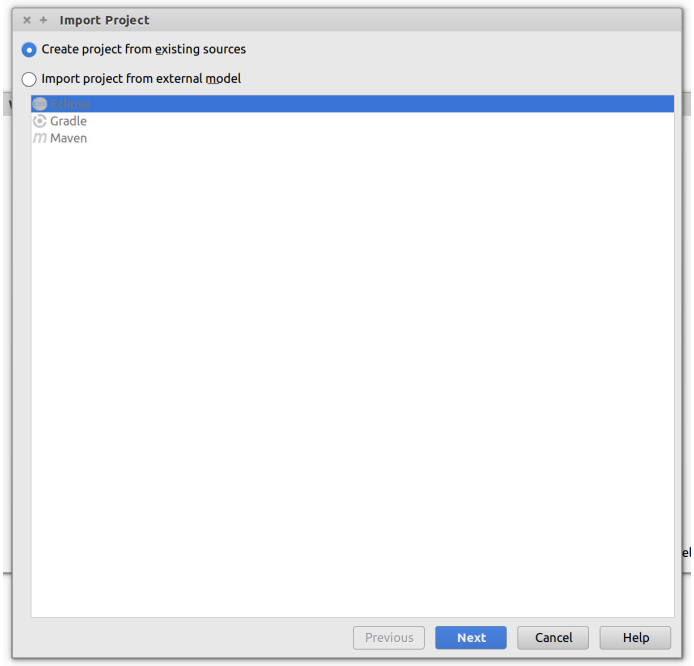


Figure 8: *the IntelliJ import project screen.*

7. You should see a window as in ***Figure 12a*** – check that this is correct, then click Next.

8. In the following window (***Figure 12b***), ensure that the SDK is correctly selected – you should be using version 1.8. Click Next.

9. Click Finish.

2.2 Running your project

You should now be at a blank workspace, with the tick shown in the Project pane on the left. Navigate to the `Tick` file which contains our `main()` method and open it. You should then be able to run the project by clicking Run → Run... (***Figure 13***). If you are running Tick 2, this will give an error: Exception in thread "main" java.lang.UnsatisfiedLinkError: no lwjgl(64) in java.library.path. This is because the JVM cannot locate the native libraries required to use LWJGL. To correct this, please do the following:

1. Click on Run → Edit Configurations... (***Figure 14a***)

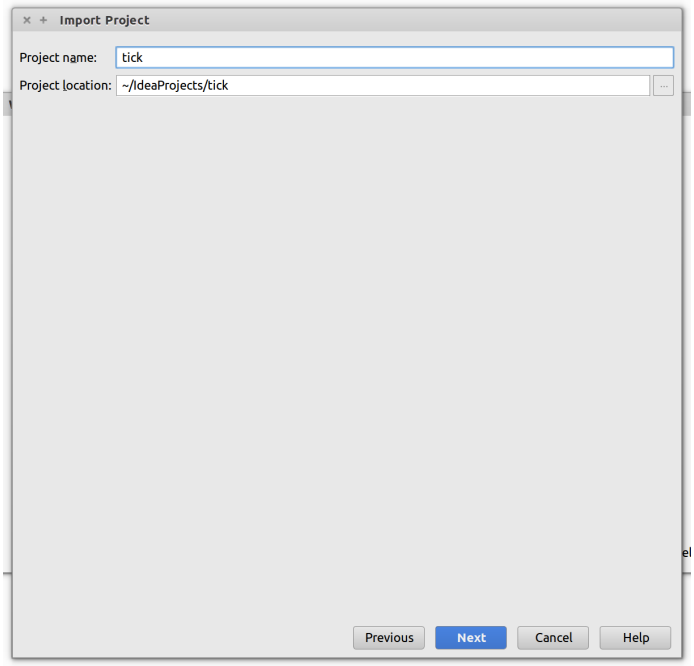


Figure 9: *naming your project in IntelliJ.*

2. In the window that appears, add the following to the VM Options field:
   `-Djava.library.path=lib/lwjgl/native`
   See (***Figure 14b***).

You should now be able to run your program without error. Ensure an OpenGL window opens correctly as described in your tick.

You should now be ready to begin working on the tick.

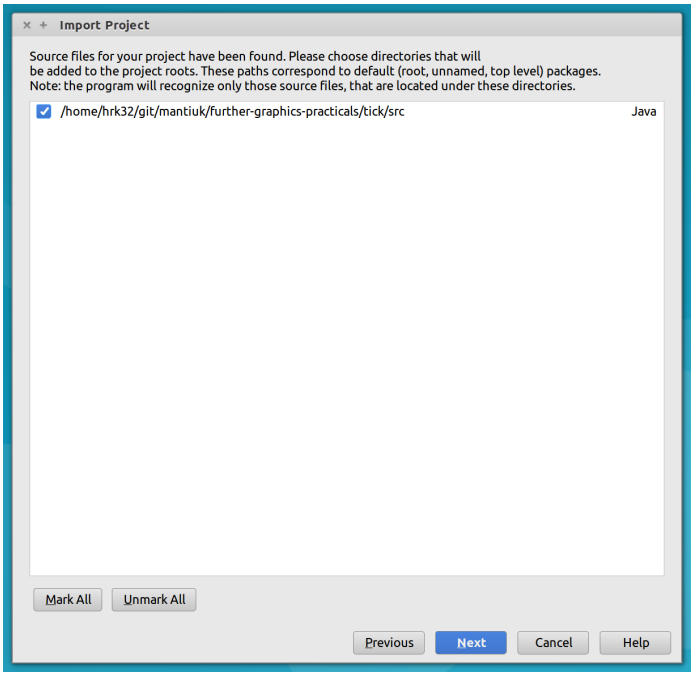


Figure 10: *the IntelliJ sources screen.*

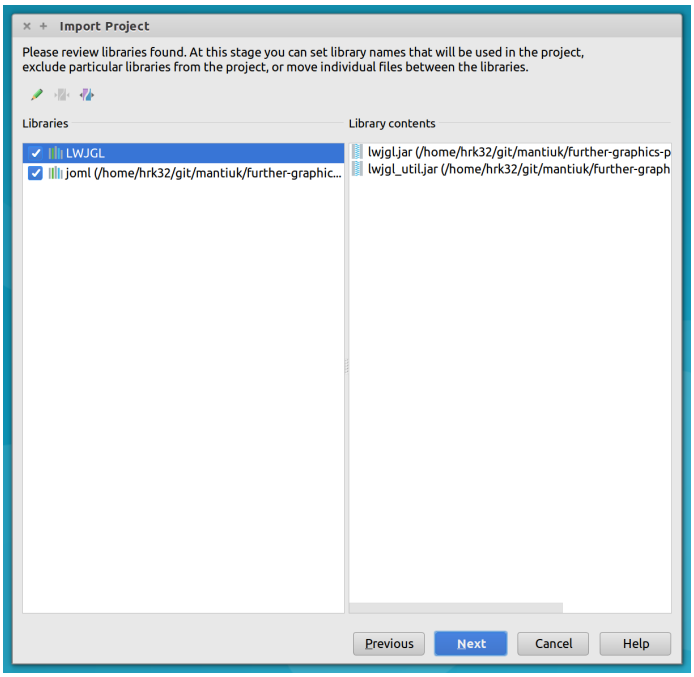
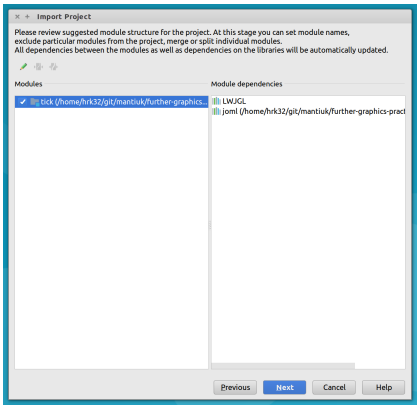
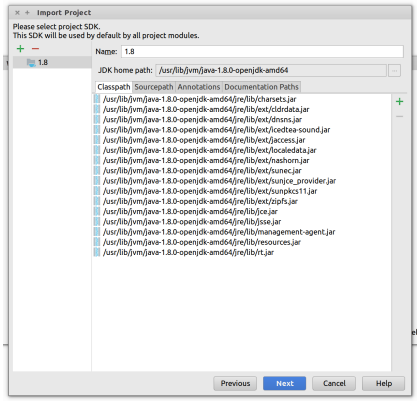


Figure 11: *the library select screen in IntelliJ.*



(a) *the IntelliJ module selection.*



(b) *the IntelliJ Java SDK selection.*

Figure 12: *selecting modules and the Java SDK in IntelliJ.*

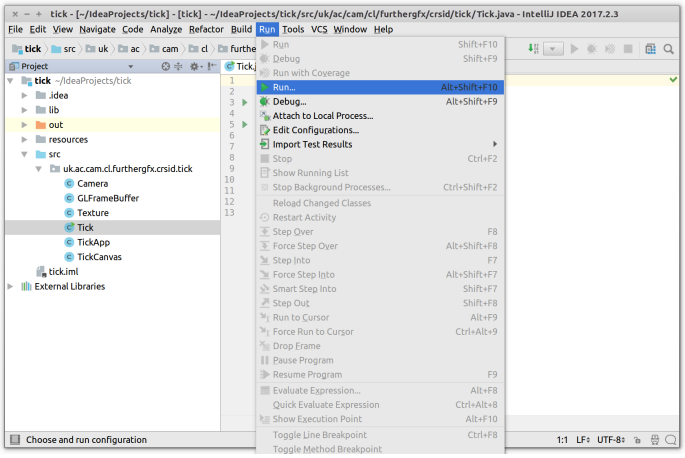
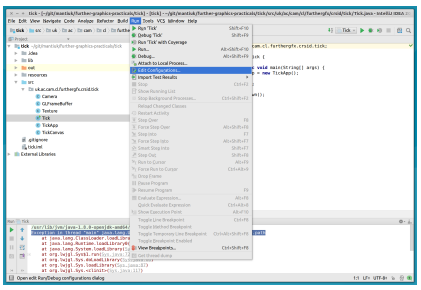
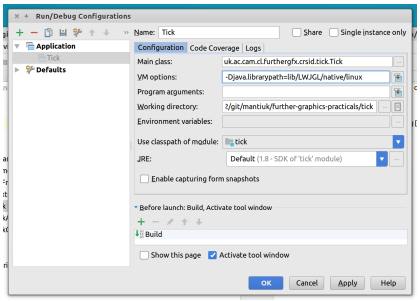


Figure 13: *running the program in IntelliJ.*



(a) *editing run configurations in IntelliJ.*



(b) *changing VM options in IntelliJ.*

Figure 14: *finding the native libraries in IntelliJ.*