***MAKE SURE testOutputEnabled, etc, in DataAggregator is FALSE when building.***

***MAKE SURE USE\_JOYSTICK is true.***

How to take a project from eclipse and make executable jar using IntelliJ (mac OS)

(specifically with Recon Custom)

1. Make a new folder. This will be the directory for the intellij project.
   1. MAKE SURE THE FOLDER NAMES DOES NOT CONTAIN ANY “/”
      1. For reason messes up jar execution
   2. MAKE SURE IT DOES NOT CONTAIN ANY “-“
      1. For reason messes up jar execution
   3. USE ONLY LETTERS TO BE SAFE.
2. Copy all the files from the eclipse project root into this folder.
   1. The purpose of copying the contents of the folder and not the entire folder is so that the other hidden files do not come with us. (for example, we probably don’t want the .git file)
   2. See appendix A to see what this directory should look like at this point.
3. Launch IntelliJ
4. If there is a project open, go to File-> Close Project
5. You should now be at the welcome screen.
6. Select “import project”
7. In the file selection screen, select the new folder we made, hit open.
8. Select “create project from existing sources”
9. Give the project a name.
   1. This will change the location.
10. After you give it the name, change the location back to the new folder with all the resources.
11. The next screen will present the directories that contain source code. Make sure all the ones you want are checked.
12. The next page shows the libraries in your project, and their contents. You should not have to do anything here, hit next.
13. The next page shows modules. You should not have to do anything.
14. The next page shows the Software Development Kit version, you should not have to do anything here.
15. The next page shows frameworks. Should not have to do anything.

**Now all the files are present, and its time to configure running.**

1. Go to “run-> edit configurations”
2. Hit the plus button, choose application.
3. Give it a name, and enter the main class without the .java extension.
4. For “vm options” put in “-Djava.library.path=libs/” DON’T FORGET THE DASH -

Now, we are going to set up dependencies make that directory and put in native library files.

1. Hit OK.
2. Go to “file -> project structure”
   1. Go the the modules page, then the dependencies tab.
3. Hit the plus symbol in the BOTTOM left, select “jars or directories”
4. Select all the .jar files the project depends on. (acm and jinput), hit open.
5. Hit OK.
6. Right clock your project name at the top of the left sidebar, select “new ->directory”
7. Name it “libs”
8. Now, go back to finder/window explorer to your project root (the new folder)
9. COPY the native library files into the libs folder.
   1. See Appendix A to figure out which files are the native library files.
   2. It is important that we **COPY** them. They are needed in “libs” by intelliJ, and will be needed in the project root for the jar to run later.
10. ***TEST THAT IT RUNS IN INTELLIJ***

At this point, the program should run within intelliJ, now we will make the executable jar

1. Go to “file -> project structure”. Go the artifacts page.
2. Hit the plus, jar, empty.
3. Give it a name (you should see a couple other places change as you type)
4. **I DON’T THINK YOU NEED THE BOLD STEPS BELOW, TEST THIS LATER. BUT IT WORKS WHEN YOU DO THEM.**
5. **Under available elements you should see your project root folder.**
6. **Hit plus button, library files, select your project root folder, and hit OK.**
7. Select “YourJarName.jar” on the left pane.
   1. A few UI elements about manifest should appear below.
8. Hit “’create manifest”
9. Hit open. (you should have nothing selected.
10. Put in the main class.
11. For the “class path” put in “jinput.jar acm.jar”
12. Hit the plus button again. File. Choose all the jar files your project depends on. Hit Open.
13. On the right pane, under your project root folder, you should see a field with your project name, followed by compile output. Double click this. Then it should appear on the left panel. THIS IS THE MOST IMPORTANT STEP.
14. Hit OK.
15. Go to “build -> build artifacts”. Hit Build.
    1. This should have produced the jar file.
    2. It can be found in your file system
       1. Project root-> out -> artifacts -> myArtifactNameFolder
16. Make a copy of the .jar file, and paste it into the project root.

Now you should be able to double click it to run it!!!

Congrats.

Note, when you double click the jar, it must be in the same folder as the native libraries, the other jar files, and other resources your program uses, with correct relative file paths. This should be done already. For example, you wouldn’t move all the images from their folder into the root, because your program thinks they are in that folder, which is resides in the project root.

YOU SHOULDN’T NEED THESE, BUT HERE ARE THE SOURCES.

<https://www.jetbrains.com/help/idea/creating-a-project-by-importing-existing-sources.html>

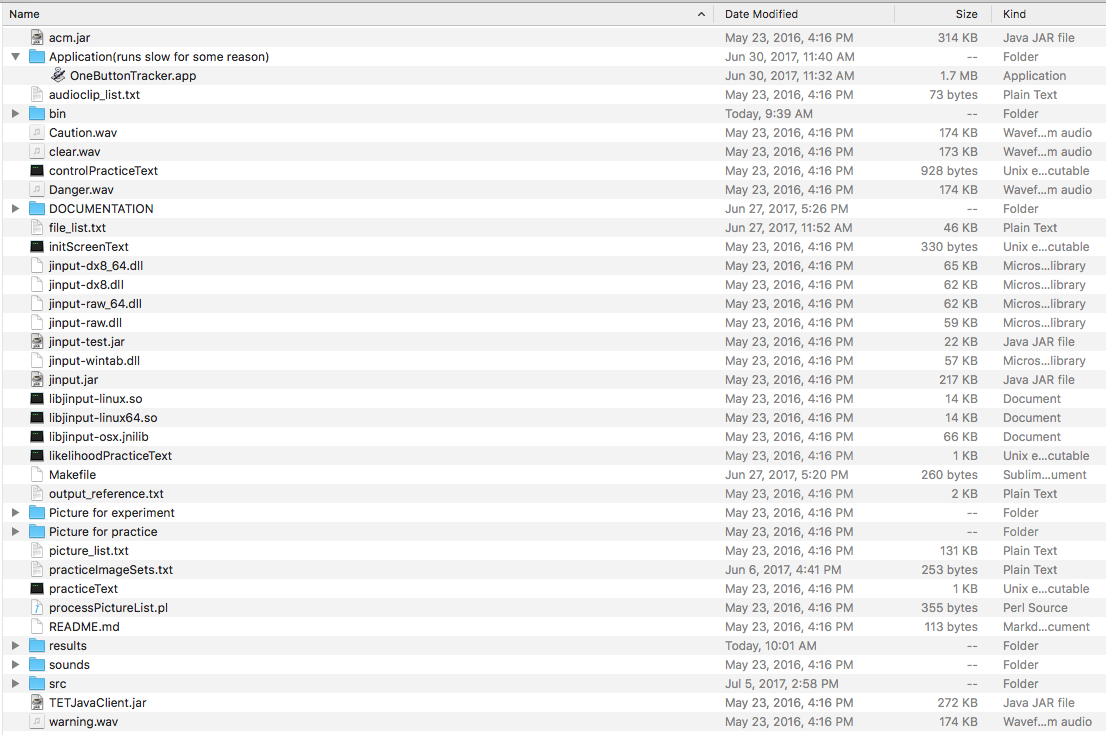
<http://wiki.lwjgl.org/wiki/Setting_Up_LWJGL_with_IntelliJ_IDEA.html>

<https://www.jetbrains.com/help/idea/working-with-artifacts.html>

<https://see.stanford.edu/materials/icspmcs106a/44-packaging-jar-files.pdf>

<https://www.jetbrains.com/help/idea/output-layout-tab.html#reorder_items>

<https://gamedev.stackexchange.com/questions/58324/packing-jar-files-into-library-jar-files>

Appendix A

“bin” contains all the .class files.

“src” contains all the .java files

All the files from “jinput-dx8\_64.dll” to “libjinput-osx.jnilib” are referred to as the “native library files”.