CSc 210: Software Development
Section 8: More Eclipse and Testing
October 19th, 2017

Today we will be discussing and delving deeper into the Eclipse IDE. We will specifically be looking into how to create projects, how projects are composed in their directory structure, how to integrate git with Eclipse, and how to create more involved JUnit tests.

Exercises

1. You should have Eclipse installed on your computer. For the purposes of this lab, you can use a lab computer and it should already be installed.
    a. If you are using a personal computer and do not have it installed, follow this link and download and install the "Eclipse for Java Developers" package.
    b. Go ahead and run Eclipse. When you open Eclipse, it will ask you for a path to your workspace. The default path it has should be fine, but take note of where this workspace is located. This is where Eclipse java projects will be stored in your computer by default.

2. First, we will look at an Eclipse project, creating class files, a project's structure, and discuss its purpose.
    a. We must first create a new Eclipse project to work within. Click on File on the top of your page and select New->Java Project.
    b. It will open up a new window asking you for some information about the project you would like to create.
    c. The only information you really need to be concerned about is the project name, so choose something and click Finish. At this point you will have a new Eclipse project.
    d. Now, lets create a new class within our project. Make sure your new project is selected from the list on the left, and click File->New->Class. Again, this will popup a window asking you for some information. Look over the different options you have for creating a class file. For our purposes, the only thing you need to fill in is the name field. Once done, click Finish and Eclipse will generate a new class file for you within your project.
    e. Write a simple program that prints out "Hello world" in your new file. To run this, click the green play button at the top of the interface. What this

button does is it automatically compiles and runs your code for you. Notice how a console will pop up showing you your program's output. If a program you write asks for input on stdin, you will enter it in that same console.

f. Now, intentionally create a bug in your program and try hitting the green play button. Eclipse should report all the errors it finds to you.
g. Examine the list of projects you have on the left side of your screen. Click on and expand the project you are working in (if it is not already). Notice the file structure of the project.
h. Recall the workspace path we defined earlier when we opened Eclipse. Eclipse will only create files within this directory. You are allowed to open and edit individual Java source files found outside of the workspace, but creating new class files from within eclipse must be in a source code folder found within your workspace.

3. Eclipse provides an awesome utility for integrating git called EGit, so let's look at using it to create repositories, commit, push, and pull.
    a. First, you must configure your user settings so it knows how to log in as you. To find these settings, go to Window->Preferences. In the drop down menu, open up Team->Git->Configuration. Click the "*Add Entry…"* Button. Enter *user.name* as *Key* and your github name as *Value* and confirm. Repeat this procedure with *user.email* and your email address associated with your github account and click Apply then *OK* in the Preferences window.
    b. Now we will create a repository out of our project. Select the project in the context menu on the left by right clicking it and navigate to *Team->Share Project…* . At this point, create a git repository. In the following window select your project, hit the *Create Repository*-button and click *Finish*.
    c. We now need to add the files. Left click the project again and go to Team->Add to Index. You should now see a green plus sign on all of your source files meaning they are now being tracked.
    d. At this point we must commit our files. To do this, right click the project and go to Team->Commit... . In the window that pops up, you will need to enter a commit message and select the files you want to commit. Once done, click Commit. Notice the image on your source files should change to a golden square.
    e. Now we want to push our changes to a remote repository. First, we need a repo to push to, so go to GitHub and create one. Get the URL for the repository. Right click your project and go to Team->Push…. In the

window that pop up, place the URL of the repo you created into the URL box. You also need to place your git username and password in so it may push. Once this is filled out, click Next twice and then click Finish. It will work on pushing your code and give you a pop up window letting you know it finished!

f. Your project is now in your GitHub repo.

4. Sometimes it is necessary to access your raw java source files instead of having them inside of an Eclipse project scheme (For instance, turning in your assignments). We will look at how to do this.
    a. Recall the directory leading to your workspace you entered when you opened Eclipse. We need to access the directory. Navigate there either through a folder environment or the command line.
    b. Once in that directory, examine the folders found here. These are all of the projects found in Eclipse. If you wish to access your java source files from a project, change your directory to *project_name*/src where *project_name* is the name of the project you wish to access.
    c. Once there, simply move or copy your files to wherever you wish to place them. Remember, we require your java source code to be in the root directory of your repo when you turn in your assignments. So, if you use Eclipse to develop future assignments, you will need to follow these steps to move your source files to the top level of your directory.

That's all for today, you should all be Eclipse pros now!