10403 Lab and Eclipse Introduction

(Last updated by asanchez,bmei I/15/20)

I. Access and check your student account.

1. Log into the TCU network with your TCU student account Username and Password. (If your password does not work, try your original password - DDMMMYY of your birthday. For example, 05MAR79.)

   Username:
   Password:
   Domain: TCU

   If at anytime during the semester you have problems with your account or password you should go to Information Commons Help Desk on the first floor of the TCU Library. The phone number to the Help Desk is X5855. If you have Internet access, you can reset the password on your account by accessing https://mypw.tcu.edu/pwreset/.

2. Take a moment to familiarize yourself with the Windows desktop. Find each of the applications noted below. *You should realize that the desktop might look a little different on different computers.*

   Firefox – The web browser you should always use to check the information of the course

   ECLIPSE IDE - The integrated development environment (IDE) used to create your Java programs. This program is available from the Start menu and may also be available from a desktop shortcut labeled "Computer Science Applications".

3. Double-click on **My Computer** to access the various resources of this machine. Besides the usual C: and D: drives you should see a shared **W: "**//studentweb/**"** with your username. This is **your TCU student drive space** and the location you will store your work for grading. You will be able to access it from any computer connected to the TCU network or that can connect to the Internet.

   IF YOU CANNOT LOGIN OR DO NOT SEE YOUR W: DRIVE, STOP HERE AND GET ASSISTANCE IMMEDIATELY! (Your account may not be set up correctly yet.)

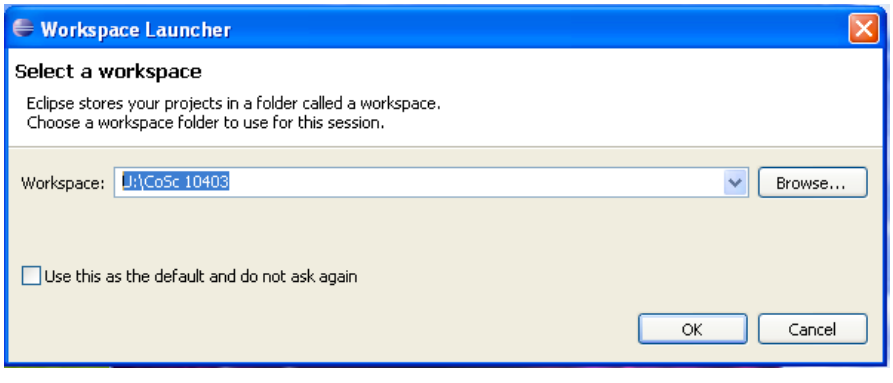
4. It is wise to occasionally check the available storage space of your student account on your **W:** drive. To check the space you can open the **W:** drive (by double-clicking), from the Edit menu select "Select All", right-click on the highlighted icons and select "Properties" from the drop-down menu. You will be shown the amount of used space on your account. Make sure to always keep plenty of available space for saving lab projects and your email. Most student accounts are given 50 Mbytes (approx. 50,000,000 bytes) of storage. If your space quota is exceeded you will not be able to run and save your labs correctly!

II. Use Eclipse (latest version for java) to create and run a Java application.

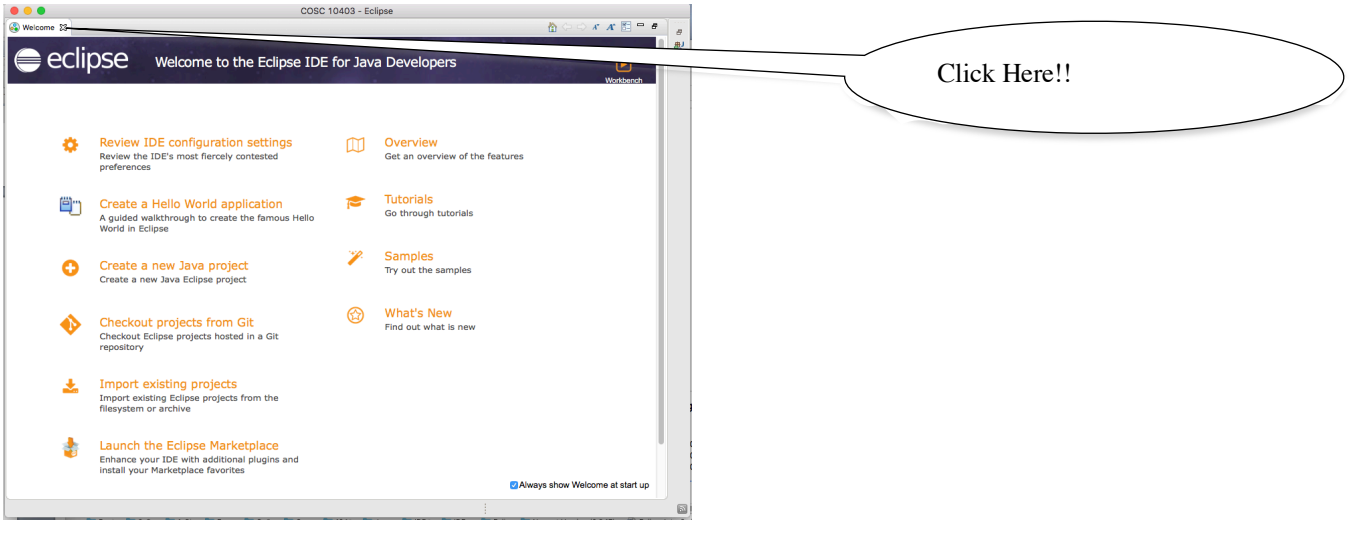
Eclipse is an open source IDE (integrated development environment) available to programmers who want to develop Java projects. Since it is an IDE, the tools needed for editing, compiling, building, debugging, executing, and obtaining online help are all integrated in one graphical user interface. Using these tools effectively can greatly increase your productivity. *However there are many options available and you may get easily confused; follow these instructions to gain an initial feel for the IDE, later you can spend time using the various options*. The current version of Eclipse are using is **Eclipse 2019-12.**

1. Before beginning, you should create a **COSC10403** folder to hold all of your work related to this class. **Note**: *this folder should reside on **your** **W: drive** (but outside of your **wwwpub** folder)!*

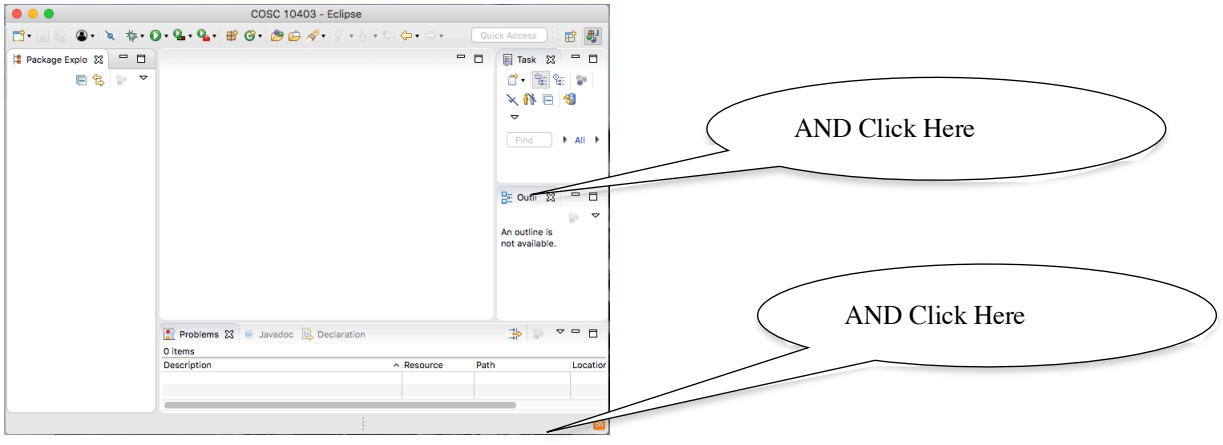
2. Start **Eclipse** by selecting **Eclipse** from the **Start** menu or by double-clicking the shortcut found inside the **Computer Science Applications** desktop folder on TTC lab computers. The first thing that you have to do is to select the workspace area where you will store your Eclipse Java Programs. *This workspace will most commonly be the* **COSC10403** *folder that you created in Step #1 above (this is to say in the **W**: drive rather than the **C**: drive) - so you may simply select OK after* **Browsing** *to get to that location.*



3. If the Eclipse *Welcome Window* appears, close it by clicking on the "**X**" to the far right of its title bar. (Note that the welcome page in Eclipse Neon may be a bit different)



After this, you should now be observing only the **File, Edit, Source** … bar and the various **Package, Tasks** and empty windows (*as seen in the figure below*).

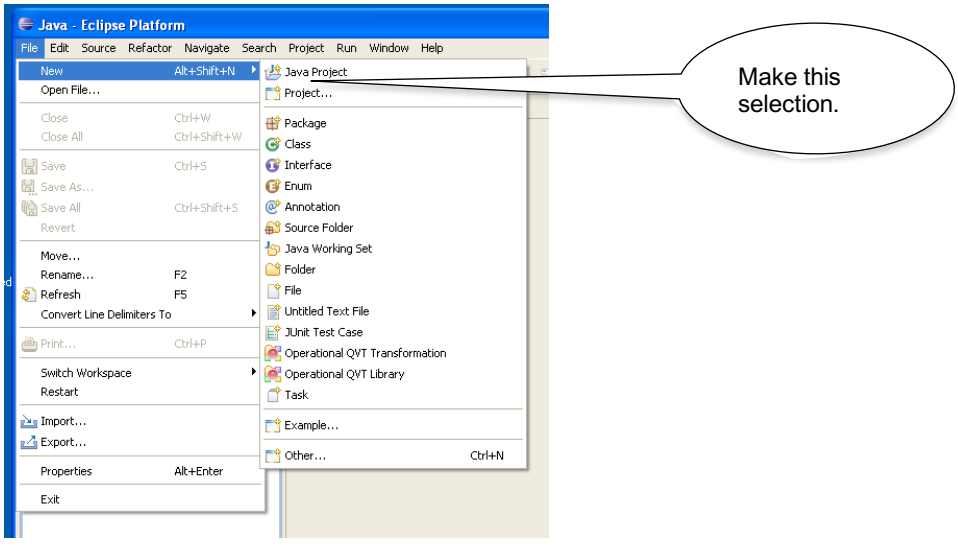


4. If this is your the first time to launch Eclipse, your **Package Explorer** window should be empty indicating that you have not yet created any projects; if you have used **Eclipse** before other Java projects may be listed *(for example Lab0Project, Lab1Project, Lab2Project, and so on). To continue working on a project that was begun earlier, simply click on the desired project in the Projects window and continue with the development.*

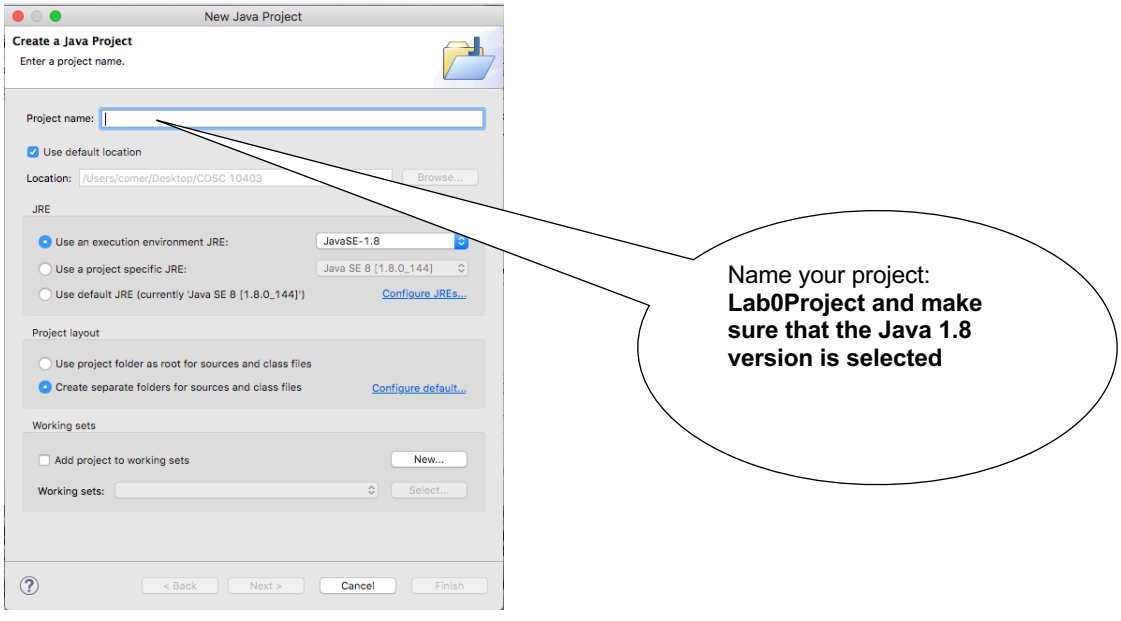
   ***Note**: To switch between **Projects**, **Files**, or **Services** windows, simply click on the desired window's title bar.*

5. To create and run a new Java program you must first create a project. *Follow the steps below*

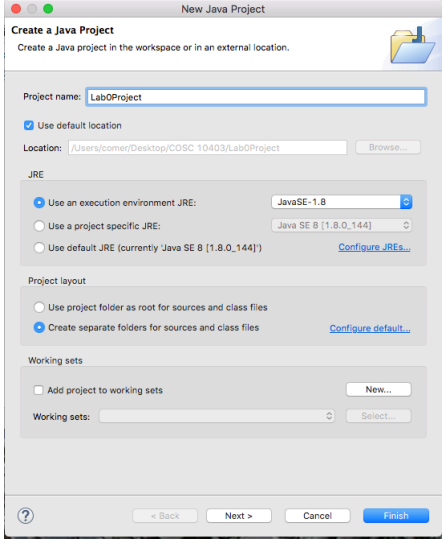
   a. Choose **New/Java Project…** from the **File** menu. *The screen below should appear.*



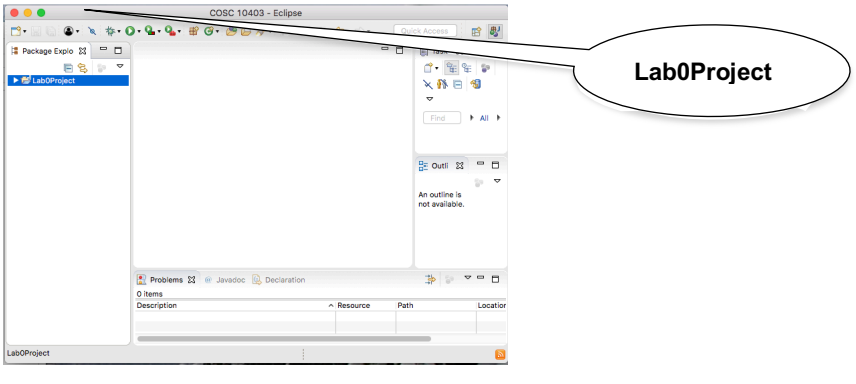
   The following window appears - to create a new project, let us name it as **Lab0Project** and select ***Next***. (**Note**: *you should always name your projects as:* **LabiProject**, *where* **i** *will correspond to the current lab number).*



A new window appears, simply select ***Finish***

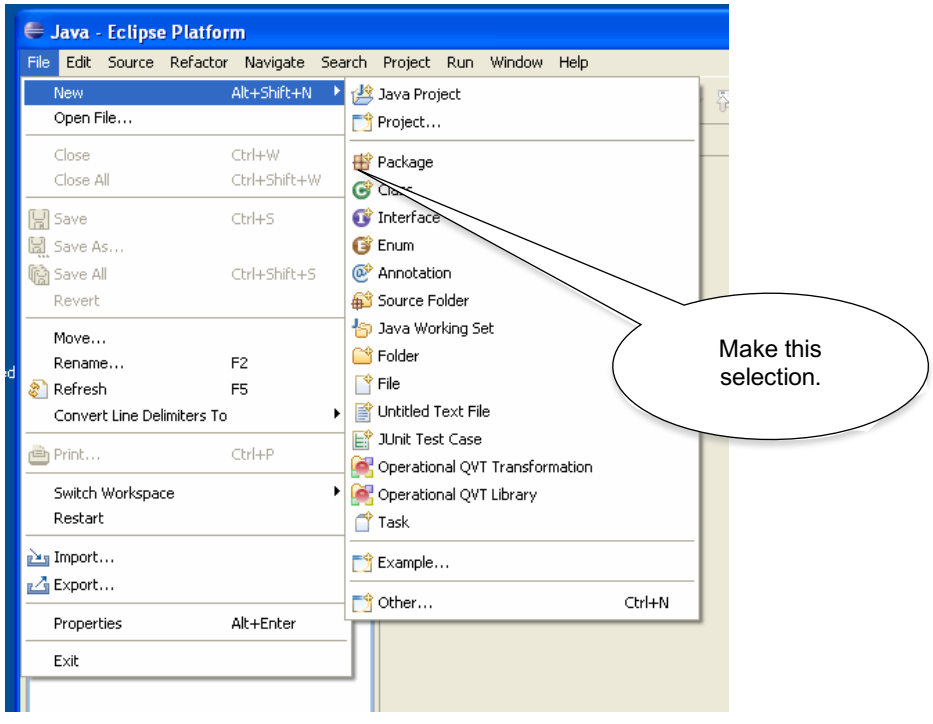


The **Project Explorer** window will show a new folder named **Lab0Project**

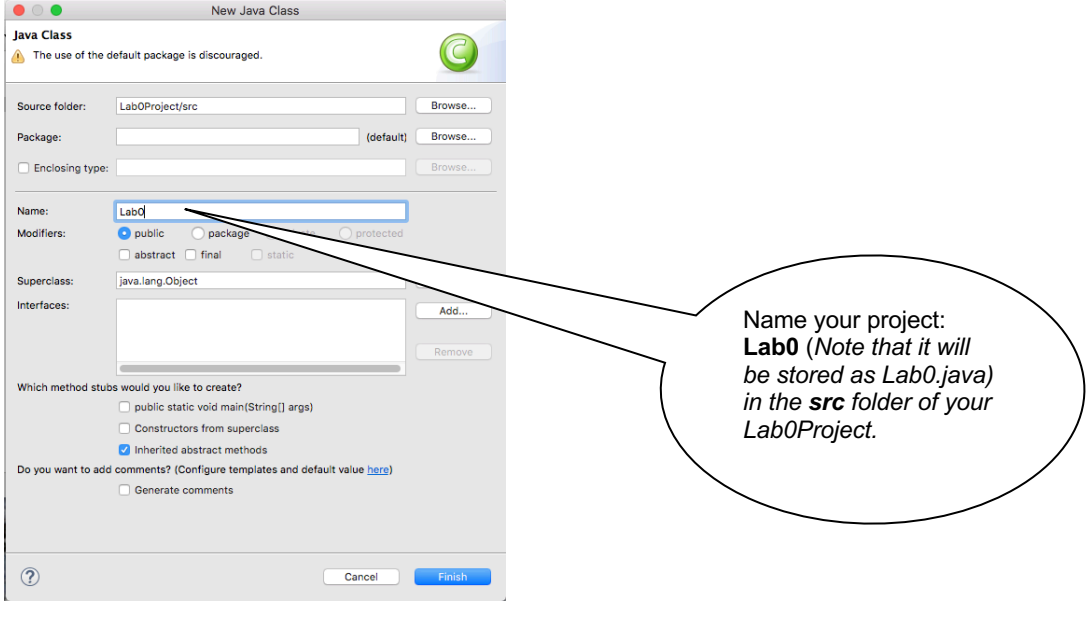


6. Create an initial class for Lab0Project

Choose **New Class…** from the **File** menu. *The screen below should appear.*

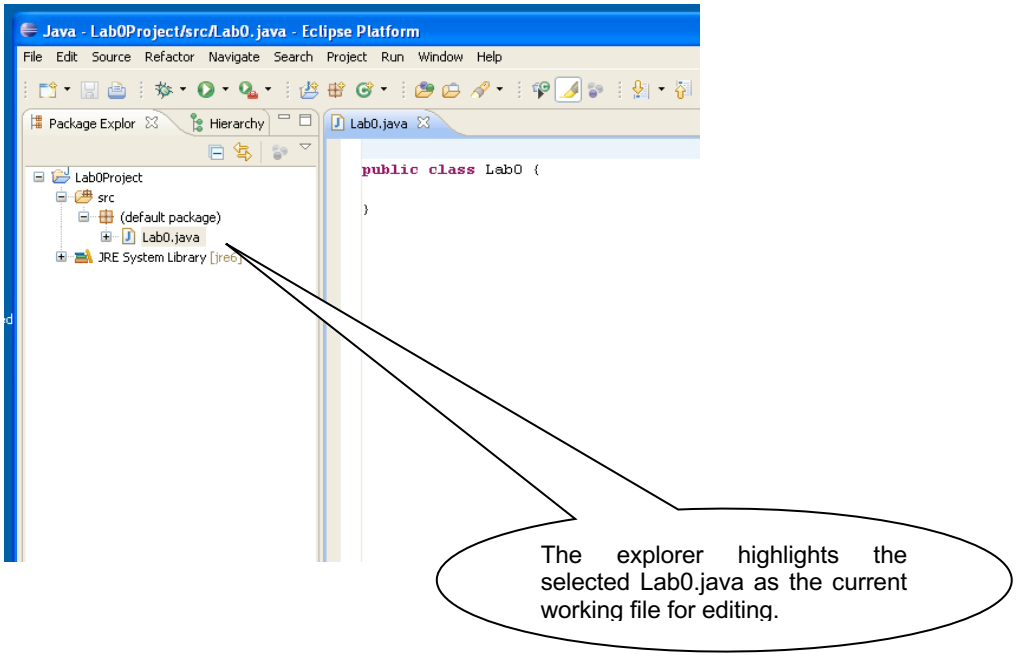


Name your class as **Lab0**, *as follows:*



Press the "Finish" button.

After you select ***Finish*** the right window of the IDE will display the source code for this new class (*as shown below):*



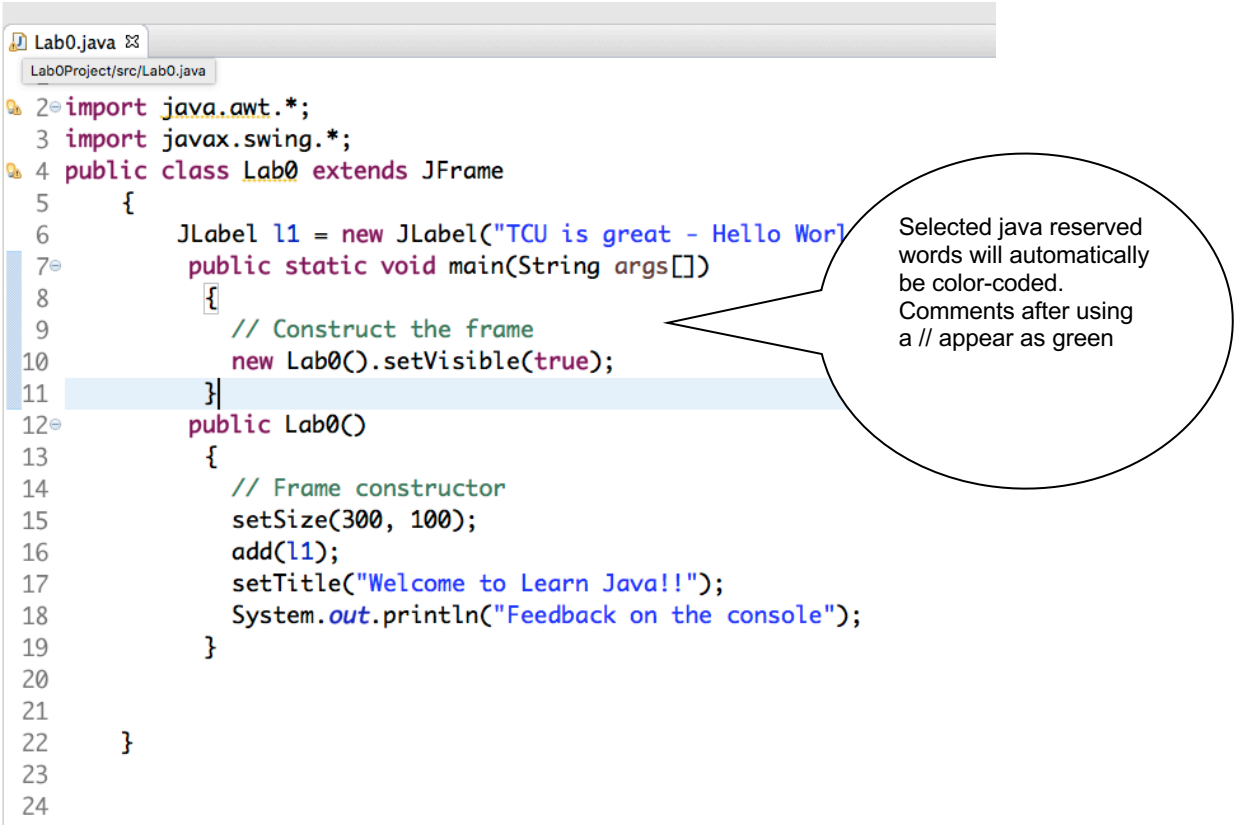
7. Editing a java file.

For this initial practice, replace (overwrite) the java source code in the right-side window with the following new java code **EXACTLY** as it is given here. *(Otherwise you will have Java errors and the program will not run):*

```
import java.awt.*;
import javax.swing.*;
public class Lab0 extends JFrame
     {   JLabel l1 = new JLabel("TCU is great - Hello World");
         public static void main(String args[])
            { new Lab0().setVisible(true);
            }
         public Lab0()
            { setSize(300, 100);
              add(l1);
              setTitle("Welcome to Learn Java!!");
              System.out.println("Feedback on the console");
            }

      }
```

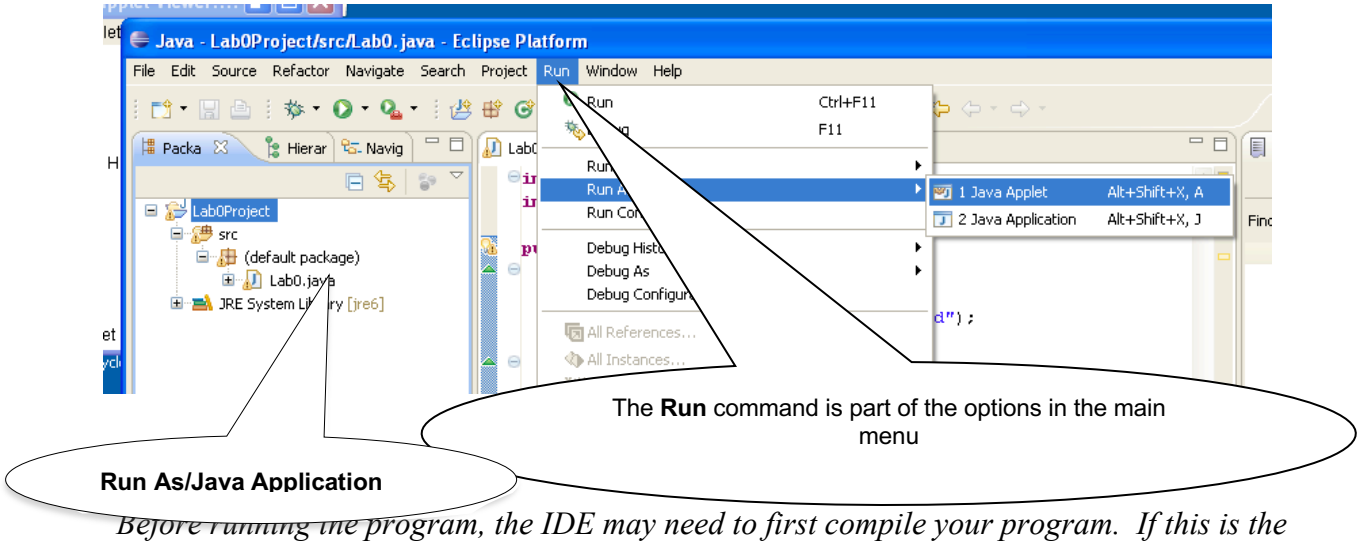
Your source code window *should now look like the following:*



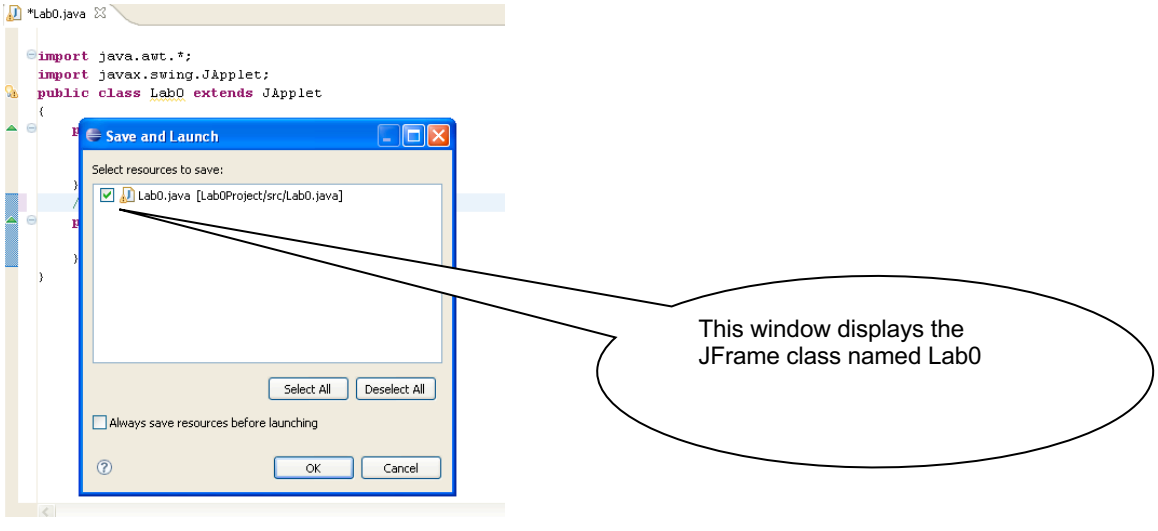
We will be discussing the significance of each of these Java statements in class shortly but here is a brief explanation. The two ***import*** *statements are needed to allow your program to access two predefined packages of Java code (useful for developing GUI's). The public class is your Lab0 program and it consists of the* **init** *method. Each method is a set of instructions separated by semicolon* ***(;)*** *and enclosed in braces* ***{ }****. The complete class is also enclosed by a pair of* ***{ }.***

8. Running the program

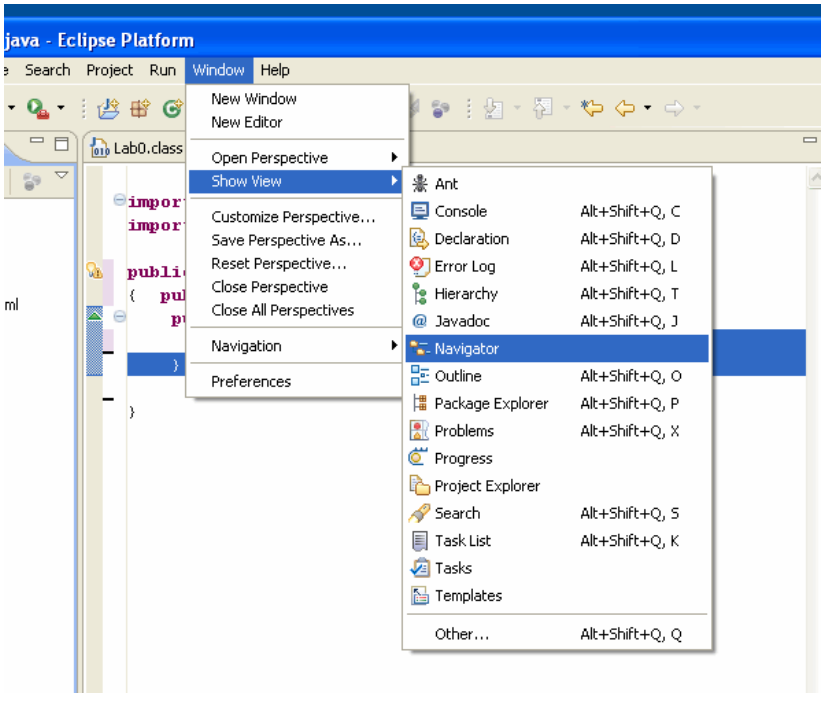
To run your program you must insure that the class **Lab0.java** file is *highlighted* in the right-side Package Explorer window. From the top menu bar, choose the **Run/Run As/Java Application** option as follows:



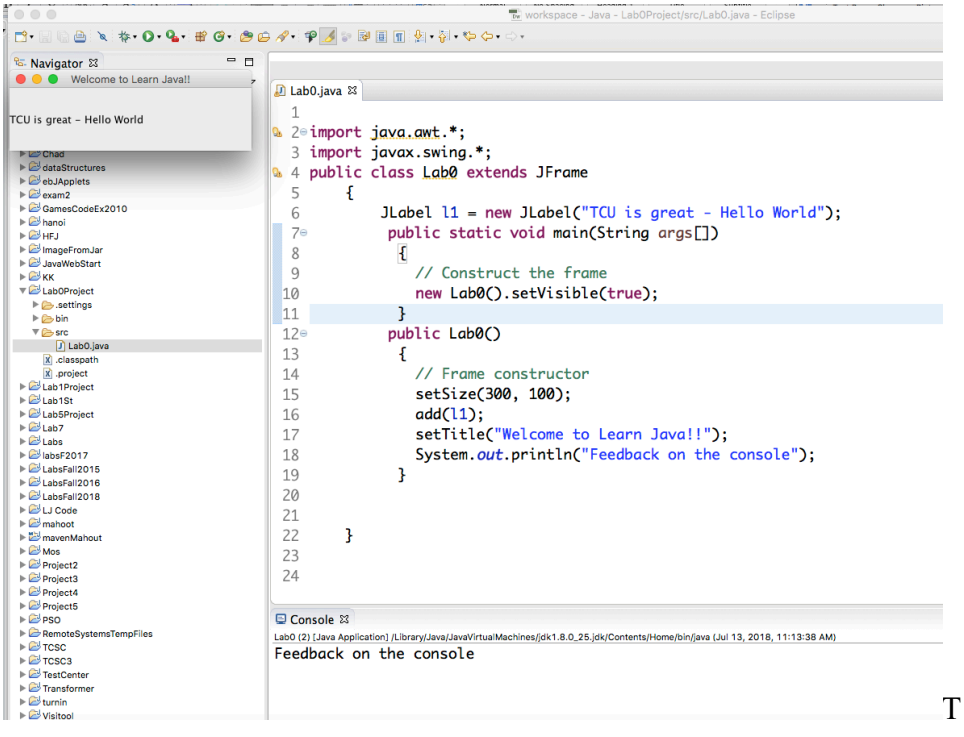
Before running the program, the IDE may need to first compile your program. If this is the case simply select OK.



You can use the Window option in Eclipse and select Show View and select Navigator, this a better view in the left sidebar the explorer

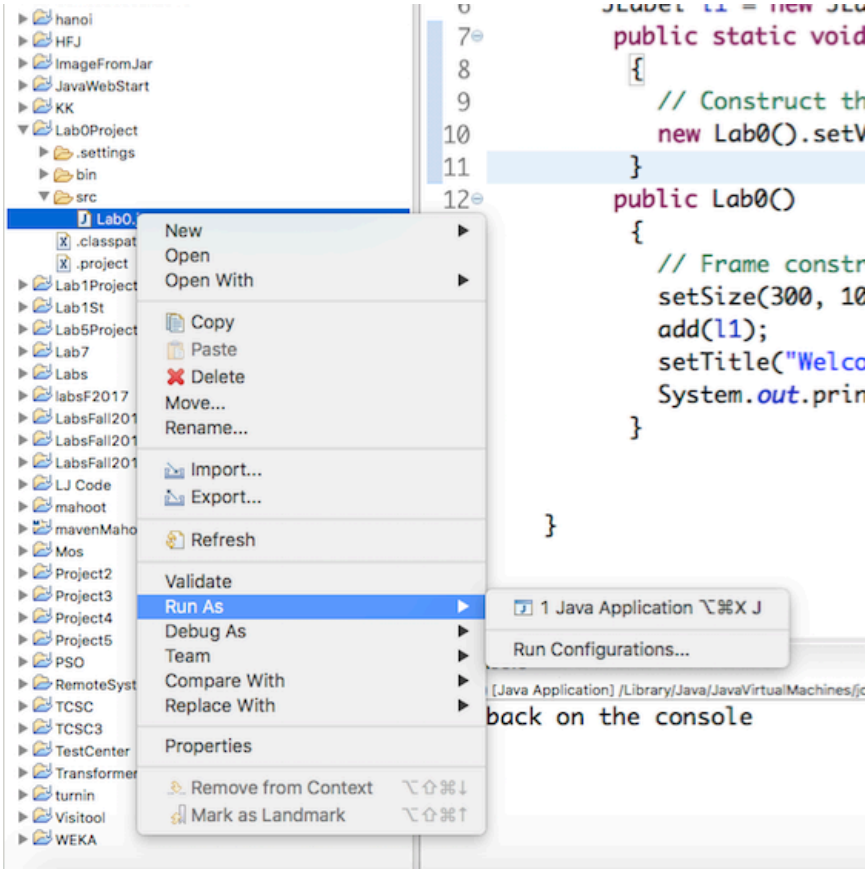


Now to show your program open the src folder and choose Lab0.java , it should open a window in the upper left corner (*as shown below)*:



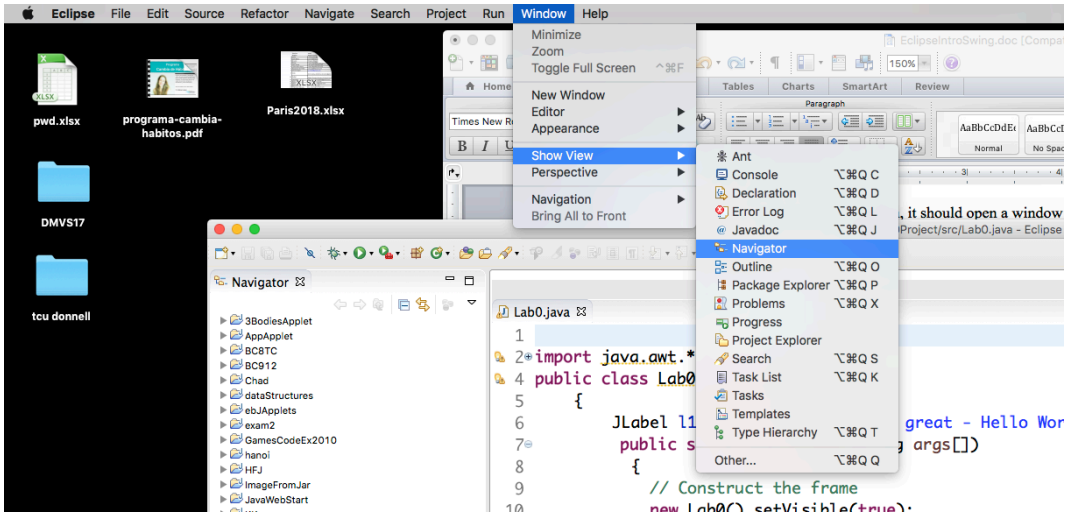
T

Note that you can also run your program using the green and black arrows in the tool bar *(as shown below)*



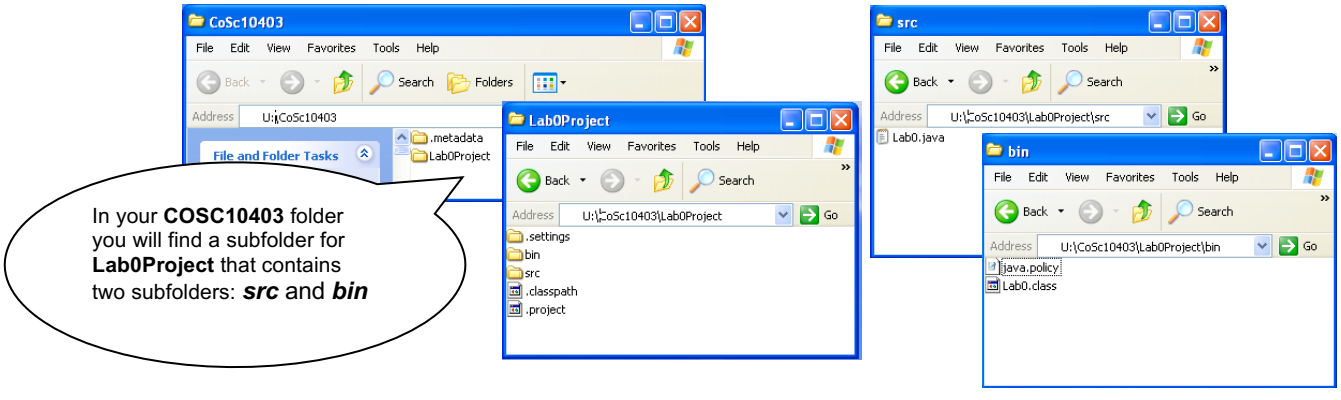
9. About source and binary Java Files (using the Navigator)

Go to the Eclipse Menu and select Window an from there select Show View and select Navigator. The navigator display will appear in the left side of Eclipse, this is a better view than the Explorer *(See below)*



After your program has successfully executed – an inspection will reveal that you now have two folders stored in your original **Lab0Project** folder. *A* **src** *folder (containing the java code that you entered earlier) and a* **bin** *folder (containing a* ***Lab0.class*** *which is the actual code that will ultimately be executed by the computer).*

To verify this, open the **COSC10403** folder. Opening a window in XP (i.e. outside **Eclipse)** you should have two subfolders in the **Lab0Project**, a **bin** and a **src.** Opening the **bin** and **src** folders will show the screen shots below ***(note this is done outside of Eclipse, using the Windows explorer)***



In the **bin** folder you have the object code of your program **Lab0.class** *(this is actually the code that the machine executes)*. In the **src** folder you will find the source code of your program **Lab0.java** *(what you typed earlier).*

a. Saving and copying your files.

Your files are automatically saved before any run command, but you can also select to save the file using the **File …Save** option once you have selected the given file.

As seen in the previous windows, if you are using computers located in one of the TCU labs (e.g., TTC 353, Library, etc), your files are most likely being saved on your **W:** drive in the **COSC10403** folder that you created at the beginning of the IDE session.

If you are using your own computer, at home on in your dorm room, your files most likely reside on your local **C:** drive.

10. Testing your Programs

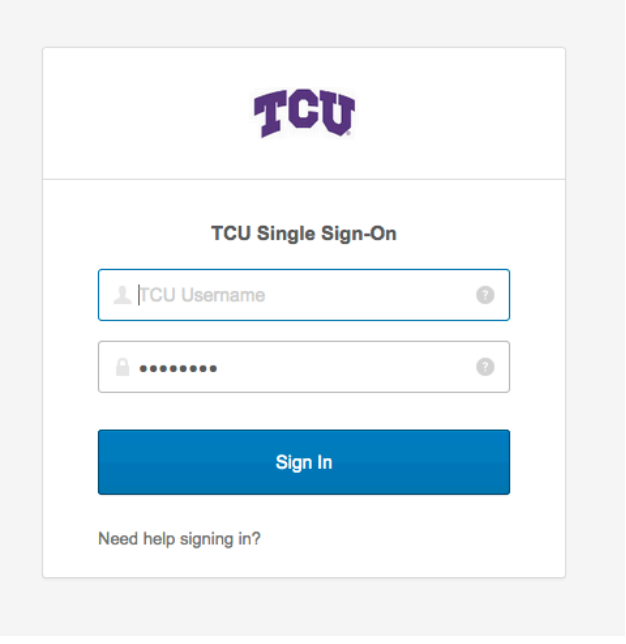
When you work on a new programming assignment you will enter Java source code for the new project (such as the **Lab0.java** file as done above) and test and retest it until everything looks perfect. Each time that you **Run** the program – Eclipse generates two files: an updated **Lab1.java** file and a new **Lab1.class** file containing the new executable code.

Once you are satisfied with your program's results, and it is perfect – you must submit the assignment for grading.

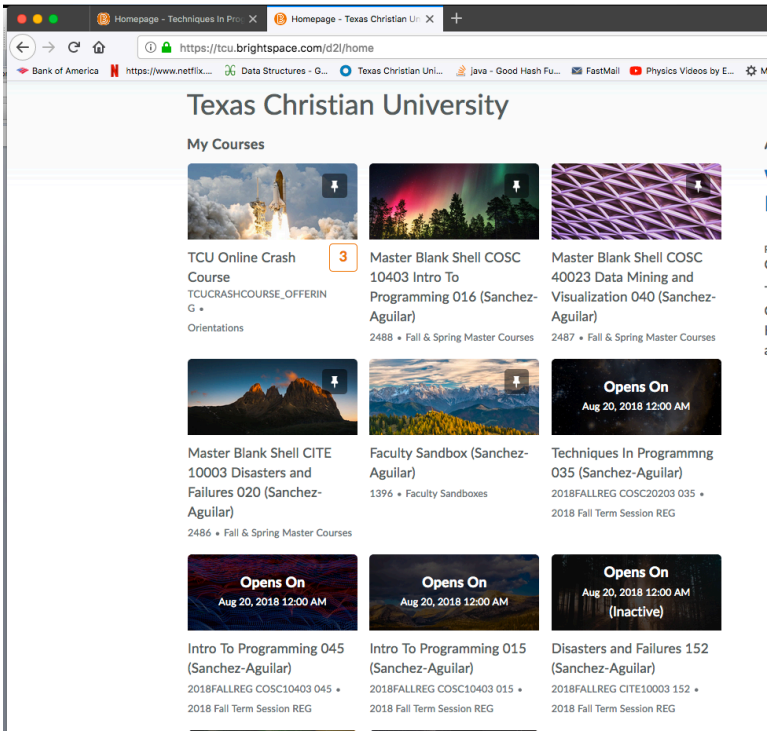
ii. YOU MUST submit your source code using the D2L webpage for this course**.** (See Section IV)

IV. Submit your Java source code (.java file) for grading using – D2L

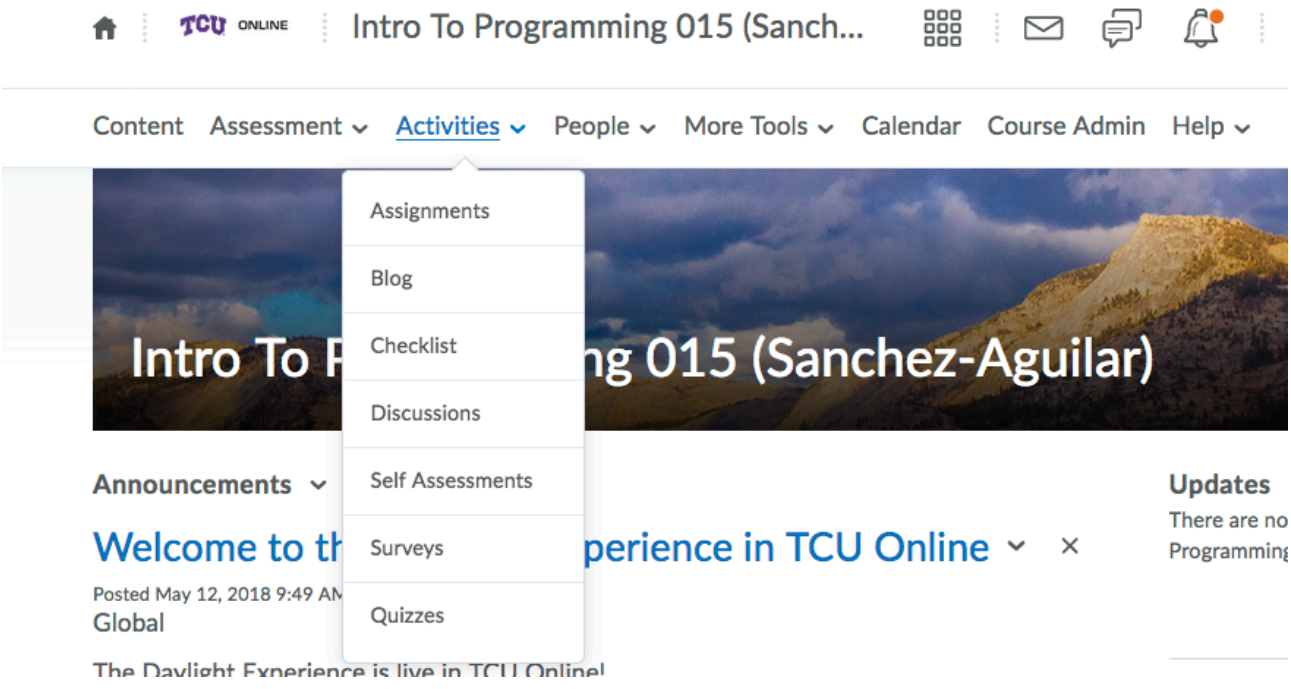
1. Go to the TCU D2L website http://d2l.tcu.edu and enter your username and password



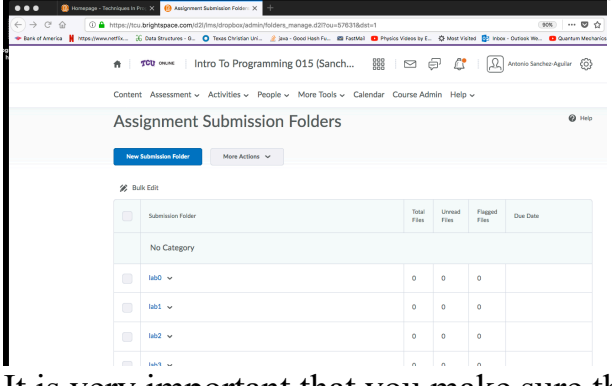
2. Select the course and section of the intro to programming course ***3.***



3. Select the assessment and select the assignments



4. Select the corresponding lab to submit your homework. In this first example use lab0 and submit the Lab0.java code you just tested and run.

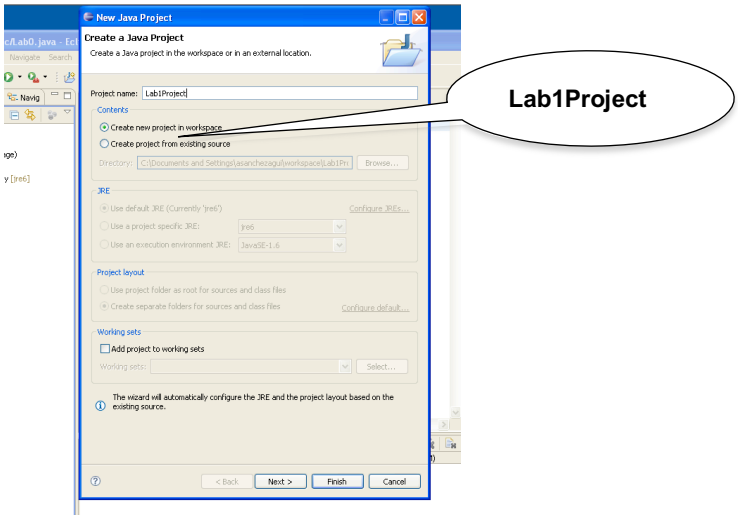


5. It is very important that you make sure that the program is running before you submit it.

11. Creating a New Project

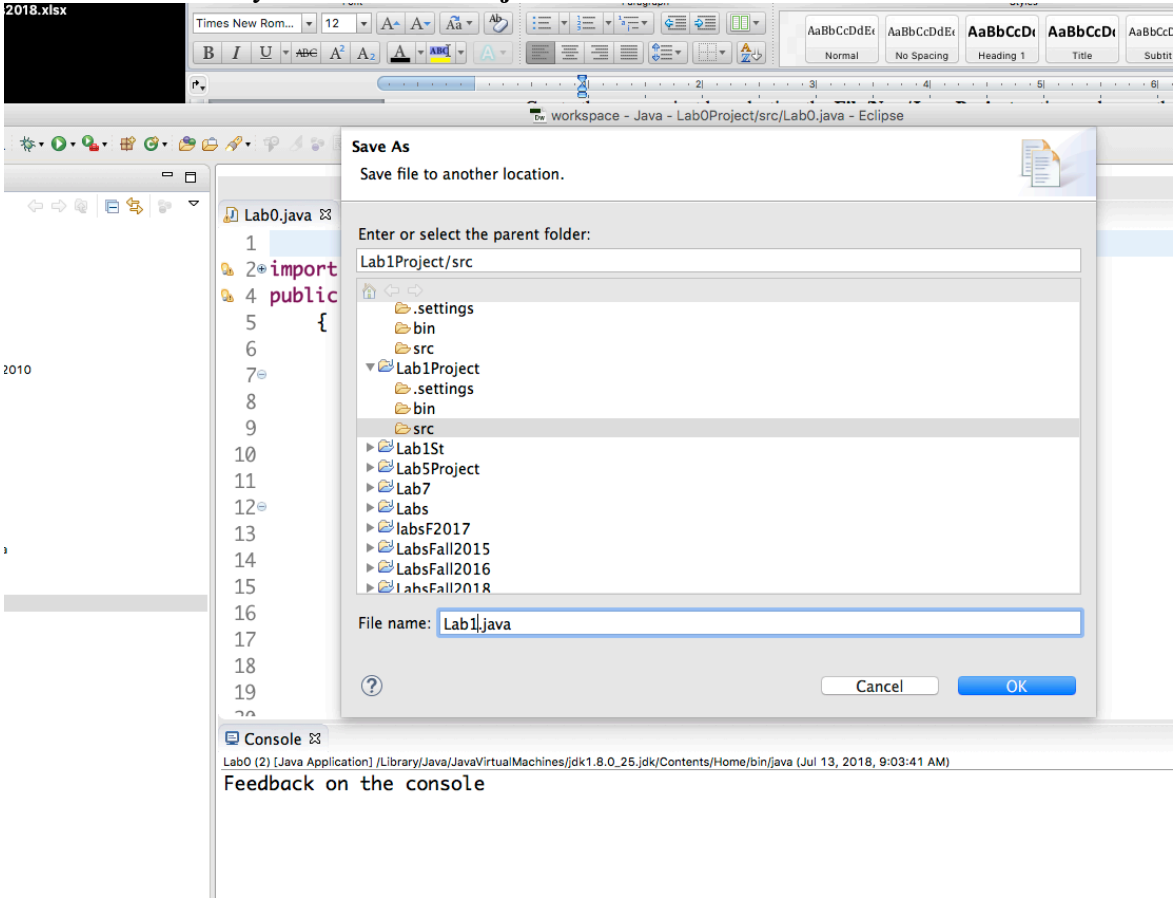
Back again in the Eclipse IDE, you can now use this initial template (saved now as **Lab0.java**) to create other projects.

Create the new project by selecting the **File/New/Java Project** option and name the project **Lab1Project** (as shown in the window below):

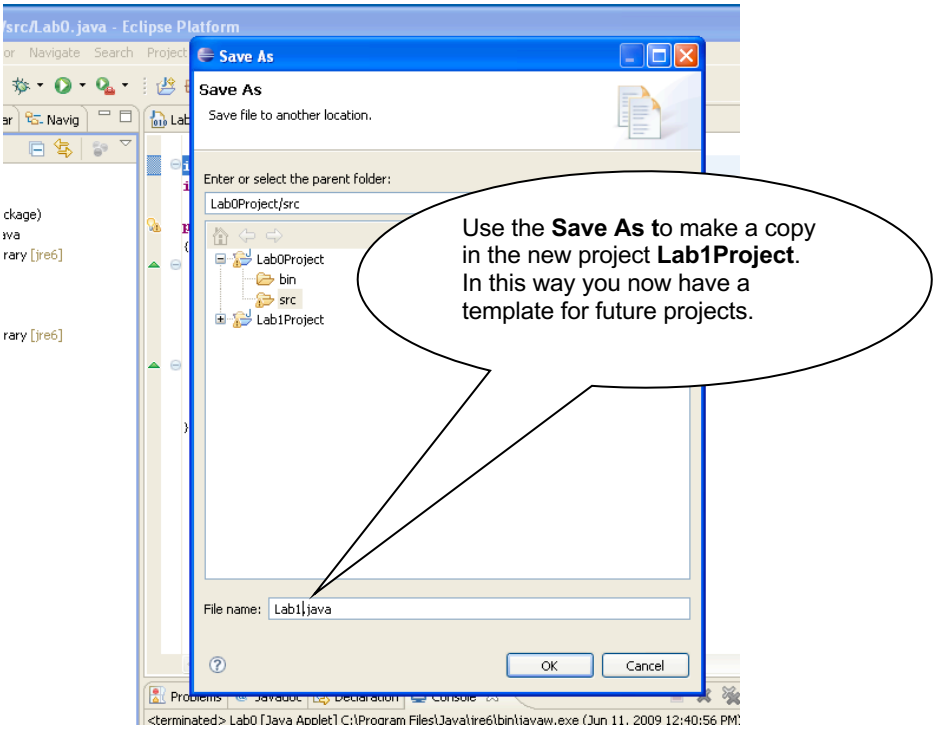


In order to create a new class, this time let us use, *as a starting point*, the code we created for **Lab0.java**.

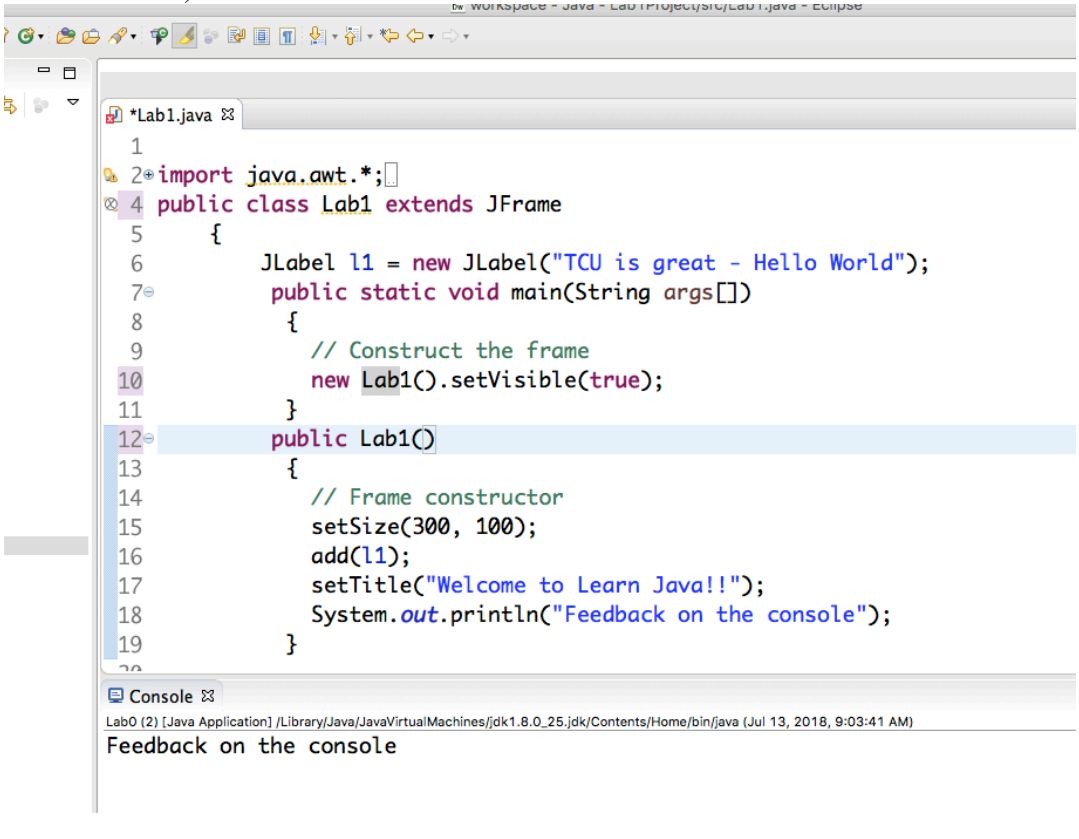
Select the **Lab0.java** code and using the **Save As** option from the **File** menu, save the class in the **src** folder of your new **Lab1Project**.



See the following screen. **Note:** *that **Lab1.java** has been entered into the File name field.*



After selecting the OK, you now have a copy of your **Lab1.java** file in the **src** subfolder of your new **Lab1Project**. You can use Eclipse to open it to edit this new java (*as shown in the window below):*

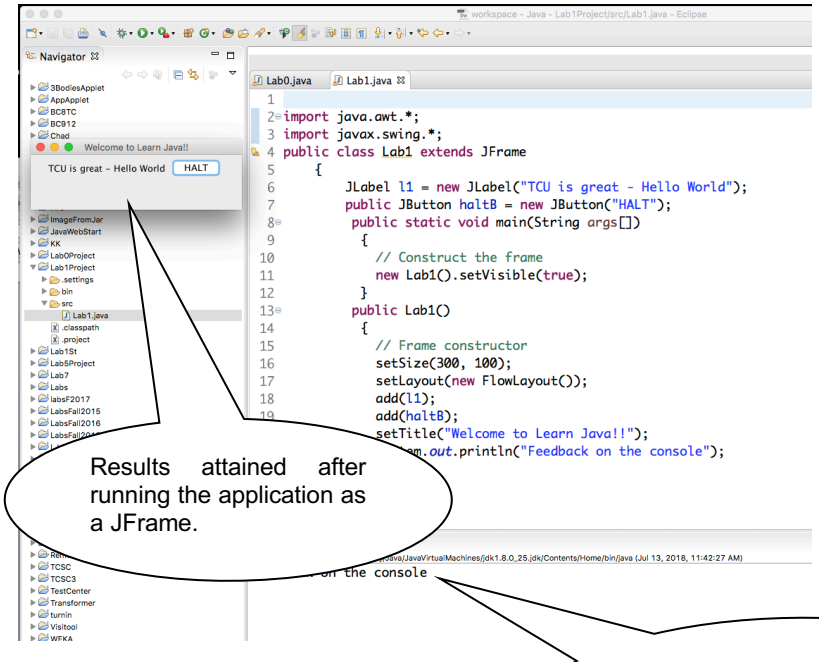


*Obviously the IDE expects you to correct the errors in the new program before it will run; for example: the class should be named **Lab1** instead of **Lab0** since it is located in the **Lab1.java** file. Make that correction and run the Applet again, as you did with **Lab0**.*

*For the sake of practice, modify your **Lab1** class by EXACTLY replacing its current source code with the following code:*

```
import java.awt.*;
import javax.swing.*;
public class Lab1 extends JFrame
      {   JLabel l1 = new JLabel("TCU is great - Hello World");
          public JButton haltB = new JButton("HALT");
          public static void main(String args[])
              {  new Lab1().setVisible(true); }
          public Lab1()
              { setSize(300, 100);
                setLayout(new FlowLayout());
                add(l1);
                add(haltB);
                setTitle("Welcome to Learn Java!!");
                System.out.println("Feedback on the console");
              }
      }
```

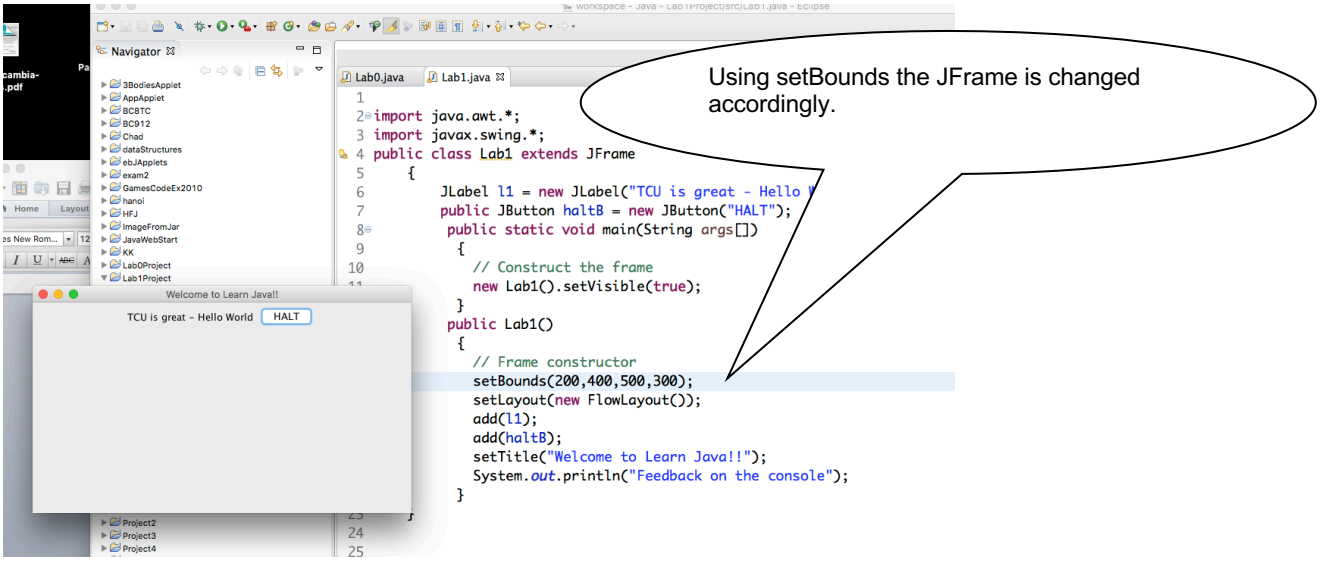
Run this applet from **Run … Run As Java Application** option using **Lab1.java** this time and you should get the following results



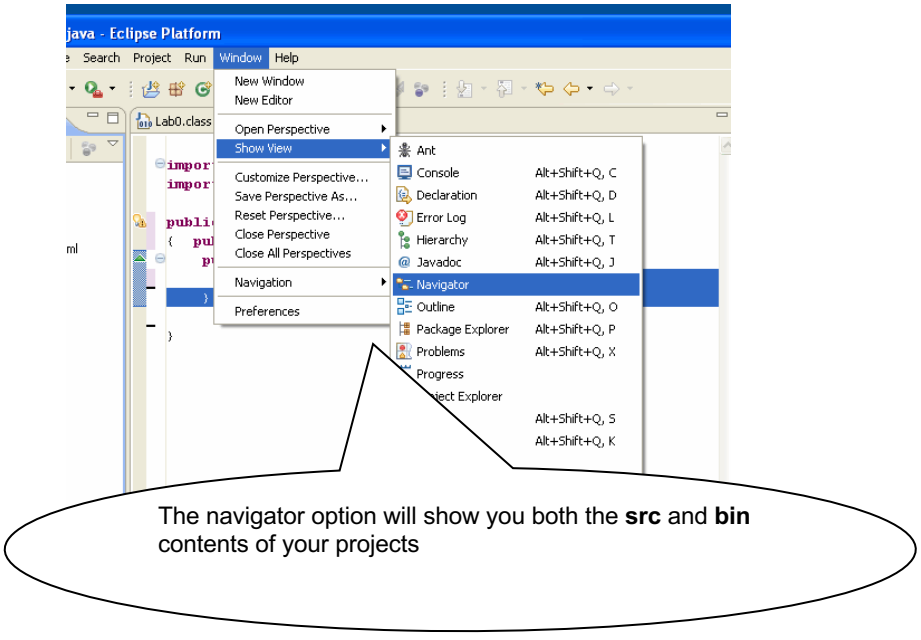
Important note: This sample Lab1.java is not the complete Lab1.java required. You will need to go the Assignments in the course webpage for the detailed description for requirements for Lab 1

12. Other options in Eclipse

a. Occasionally, in order for your JFrame to be properly displayed as as a pop up window, you will have to make changes to the ***width*** and ***height*** parameters of the program of it. You do this modification in your code by modifying the siz of the JFrame using the following code: setSize(width,height) and to define the location use setBounds(loc x,loc y, width,height) as shown below. We prefer to use setBounds().



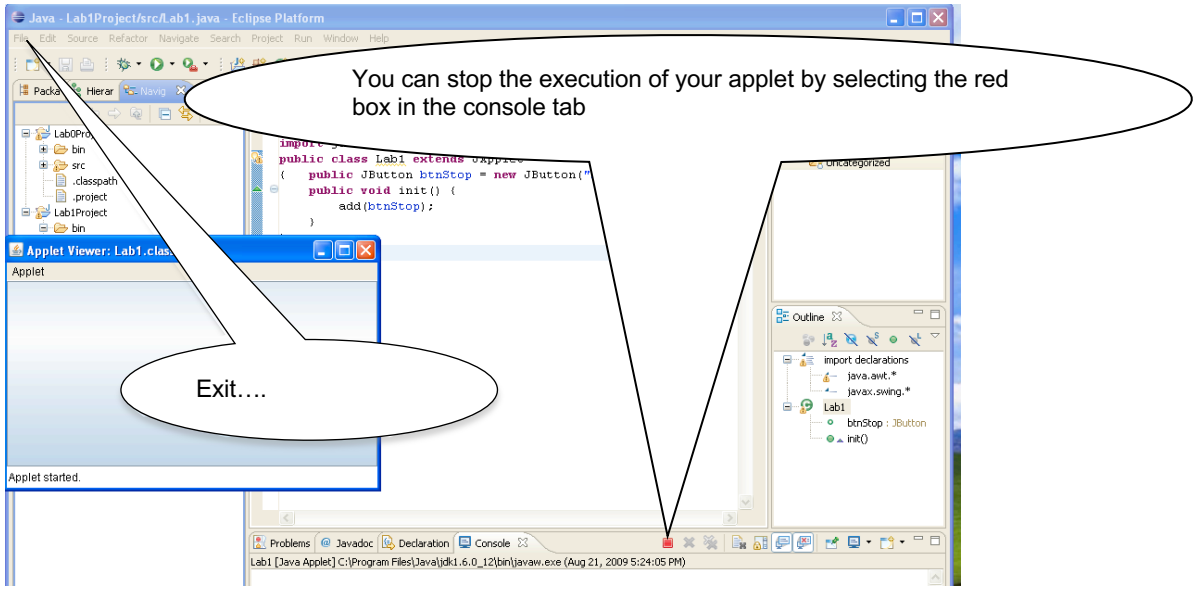
b. As you can see the options available in the Eclipse IDE are multiple and one can get lost rather easily. You will need to use Eclipse for a while in order to become comfortable with it. For Example, as discussed before, you can add different views in your left most window by selecting on the menu bar **Window… Show View** option *as follows:*



By using the Navigator tab you can now see both the src and bin folders for both projects that we've created (*as shown below*).

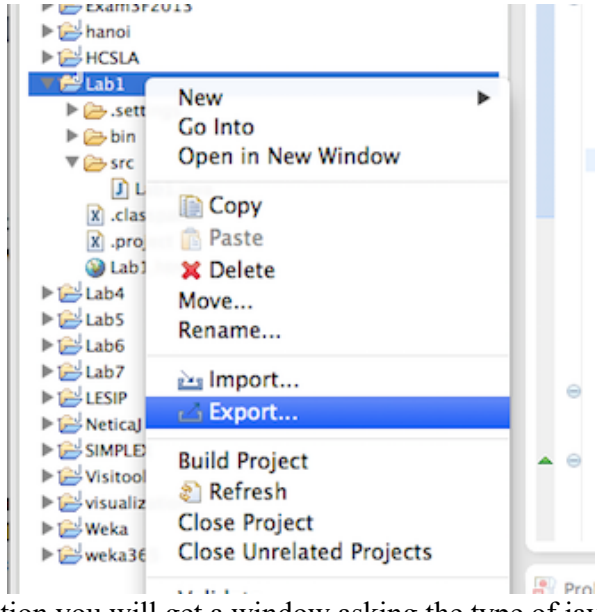
c. Closing Eclipse

To close Eclipse you only need to get out of the application by selecting the **exit** option (*in the right most corner*). However, before you do that you may want to close the Applet you have running, you can do that by selecting the red square in the bottom window (*as shown below)*

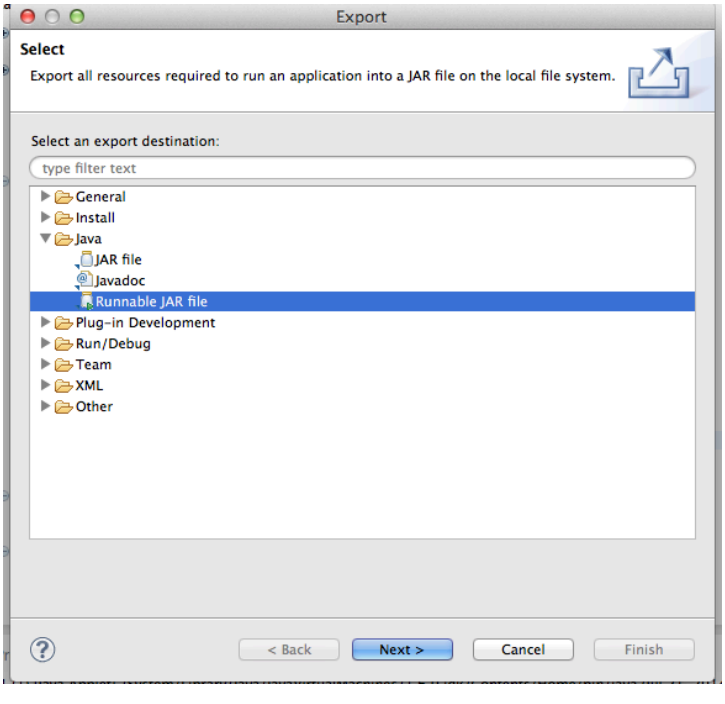


V. Displaying images and sounds in Java Application (VERY IMPORTANT)

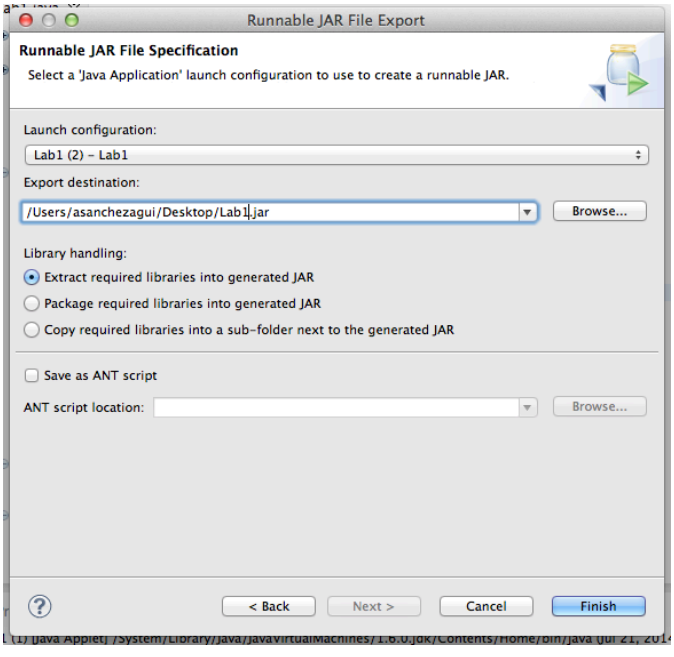
1. **Creating a clickable Jar file:** Now we want to create this program as single clickable application, Eclipse will help doing so, by selecting the option export if you click on the directory where you have the applet class, as follows:



when you select the export function you will get a window asking the type of java jar you want, here you select runnable jar as follows:



by selecting the next button a new will appear asking to select the launch configuration where you have to select the directory and the java class you are working on, for this example in this case Lab1. Also you will need to determine the export destination and the name of the jar file, for this example we have place it on the Desktop with the name Lab1.jar, as shown below



After selecting Finish we will have the file Lab1.jar on the desktop. If just double click on it our application will run on the frame as shown before.

The importance of this approach is that you can run this jar file in any machine that has java and it will run perfectly.

2. **Handling images in Applications:** There is a very important consideration to make here when dealing with images. While applets can read images rather simply from the bin folder, as you will learn in class using the img = getImage(getCodeBase( ), "imgFile.jpg"); command. The case of Applications is different in this case you will have to change the previous line of code by the following one:

   img = Toolkit.getDefaultToolkit().getImage("imgFile.jpg");

   Notice also that the location of the imgFile.jpg will have to be placed outside the bin folder and not inside otherwise the image will not be found

3. Handling Sounds in Applications: Later in the course you will also learn to add sound to your programs using the s = getAudioClip(getCodeBase(),"soundFile.wav"); Applet command. Again for the case of Application programs this will have to be changed to a more obscure code as follows:

```
try{
        File myFile = new File("soundFile.wav");
        s = JApplet.newAudioClip(myFile.toURI().toURL());
        //Or s = JApplet.newAudioClip(myFile.toURL());
  }
catch (Exception e) {}
```

Again note also that the location of the soundFile.wav will have to be placed outside the bin folder and not inside otherwise the image will not be found. **BUT** for the time being do not worry about this last piece of code yet.

However, it is important to note that you may be required to turn both the jar file and the source java later in the course using D2L . This is done by compressing both file in a single .zip file.

VI. Working remotely on your own computer.

You can download Java and Eclipse on your own laptop and run from there, for this you must follow the following approach, here we provide the details for both Mac and Windows:

Java Development Kit (JDK) and Eclipse Installation Guide (macOS Version)

A. Install Java Development Kit (JDK)
   1. Go to https://adoptopenjdk.net/?variant=openjdk12&jvmVariant=openj9
   2. **Select the version for OpenJDK 13 and OpenJ9**
   3. Click on the blue "Latest release" button. The installer will be downloaded to your computer.
   4. Run the installer, and follow the instructions in the installer.

B. Install Eclipse
   1. Go to https://www.eclipse.org/downloads/packages/release/2019-06/r/eclipse-ide-java-developers
   2. Under "Download Links" section, click the "Mac OS X (Cocoa) 64-bit" link.
   3. On the next page, click on the orange "Download" button. The installer will be downloaded to your computer.
   4. Run the installer. Drag the "Eclipse" icon to the "Application" folder.

C. Run Eclipse
   1. Find the "Eclipse" icon in Launchpad, and click on the icon. A window named as "Eclipse IDE Launcher" will be popped up.
   2. **Write down the workspace path.** Check the "Use this as the default and do not ask again" checkbox. Then, click on the "Launch" button.
   3. Uncheck the "Always show Welcome at start up" checkbox at the bottom-right corner. Then, click on the orange "Workbench" play button at the top-right corner.

D. Shortcut the workspace folder
   1. In Finder, find the "eclipse-workspace" folder by following the path that you just wrote down.
   2. Right-click on the folder, and select the "Make Alias" option. A folder shortcut named as "eclipse-workspace alias" will be created.
   3. Drag the "eclipse-workspace alias" folder shortcut to the desktop. Rename the shortcut to just "eclipse-workspace".

Java Development Kit (JDK) and Eclipse Installation Guide (Windows Version)

A. Install Java Development Kit (JDK)
   1. Go to https://adoptopenjdk.net/releases.html?variant=openjdk12&jvmVariant=openj9#x64_win
   2. **Select the version for OpenJDK 13 and OpenJ9**
   3. Click on the blue "Install **JDK**" button. The installer will be downloaded to your computer.
   4. Run the installer, and follow the instructions in the installer.

B. Install Eclipse
   1. Go to https://www.eclipse.org/downloads/packages/release/2019-06/r/eclipse-ide-java-developers
   2. Under "Download Links" section, click the "Windows 64-bit" link.
   3. On the next page, click on the orange "Download" button. A zip file will be downloaded to your computer.
   4. Find the zip file named as "eclipse-java-2019-06-R-win32-x86_64.zip" (typically under the "Downloads" folder) in File Explorer. Double-click on the zip file. You will see a folder named as "eclipse". Keep the window open.
   5. Open a new File Explorer window, and go to C:\Programs Files. Drag the "eclipse" folder from the previous window into the "Programs Files" folder.
   6. Double-click on the "eclipse" folder. Right-click on the "**eclipse.exe**" file, and select the "Send to → Desktop (create shortcut)" option. On the desktop, rename the shortcut from "eclipse.exe – Shortcut" to "Eclipse".

   Additional information
   a. There is an error in the current Windows Eclipse zip file, which causes the downloading issues at Step 3.
   b. It is safe to close the web browser although the browser may show a misleading message like "Your files are still downloading" or "Download is in progress".
   c. After completing Step 7, it is also safe to delete the "eclipse-java-2019-06-R-win32-x86_64.zip" file and its corresponding unzipped folder on the desktop. However, because of the error, they can only be deleted after you reboot the computer.

C. Run Eclipse
   1. On the desktop, double-click on the "Eclipse" shortcut. A window named as "Eclipse IDE Launcher" will be popped up.
   2. **Write down the workspace path.** Check the "Use this as the default and do not ask again" checkbox. Then, click on the "Launch" button.
   3. Uncheck the "Always show Welcome at start up" checkbox at the bottom-right corner. Then, click on the orange "Workbench" play button at the top-right corner.

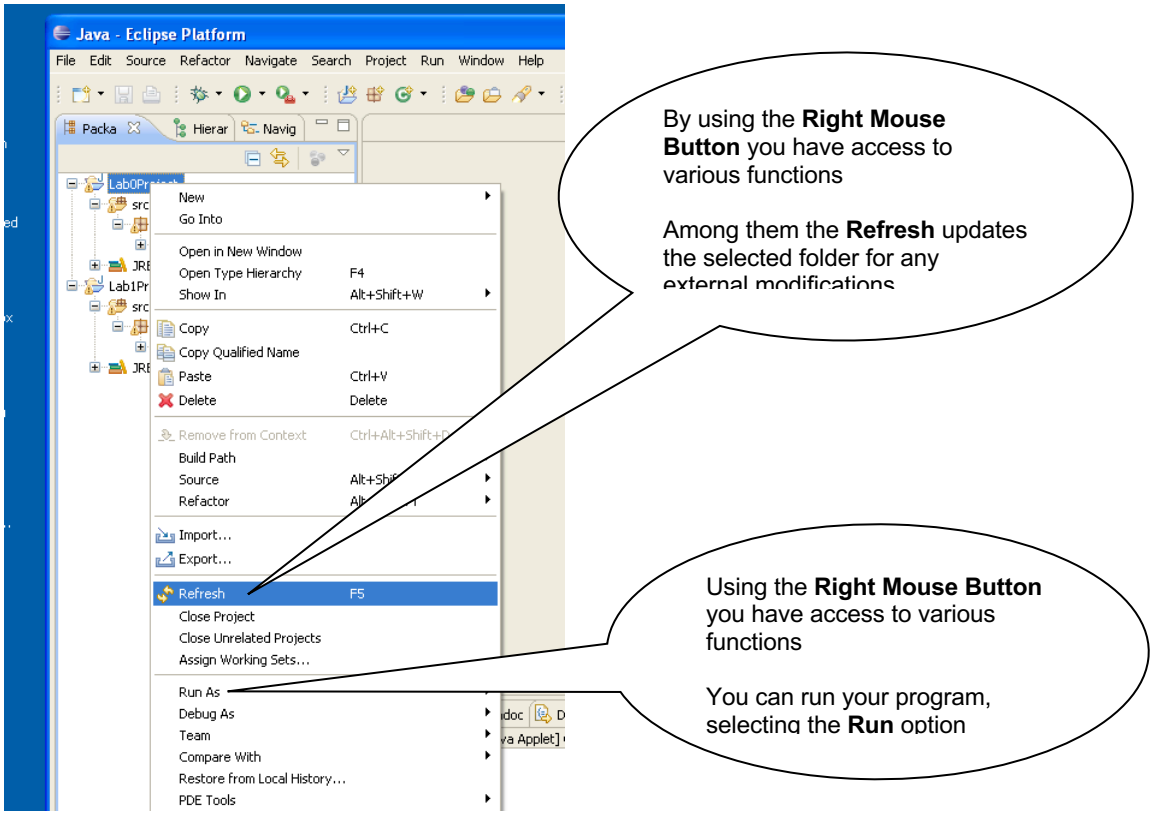
D. Shortcut the workspace folder
    1. In File Explorer, find the “eclipse-workspace” folder by following the path that you just wrote down.
    2. Right-click on the folder, and select the “Send to → Desktop (create shortcut)” option. On the desktop, rename the folder shortcut from “eclipse-workspace – Shortcut” to just “eclipse-workspace”.

VII. Using Eclipse

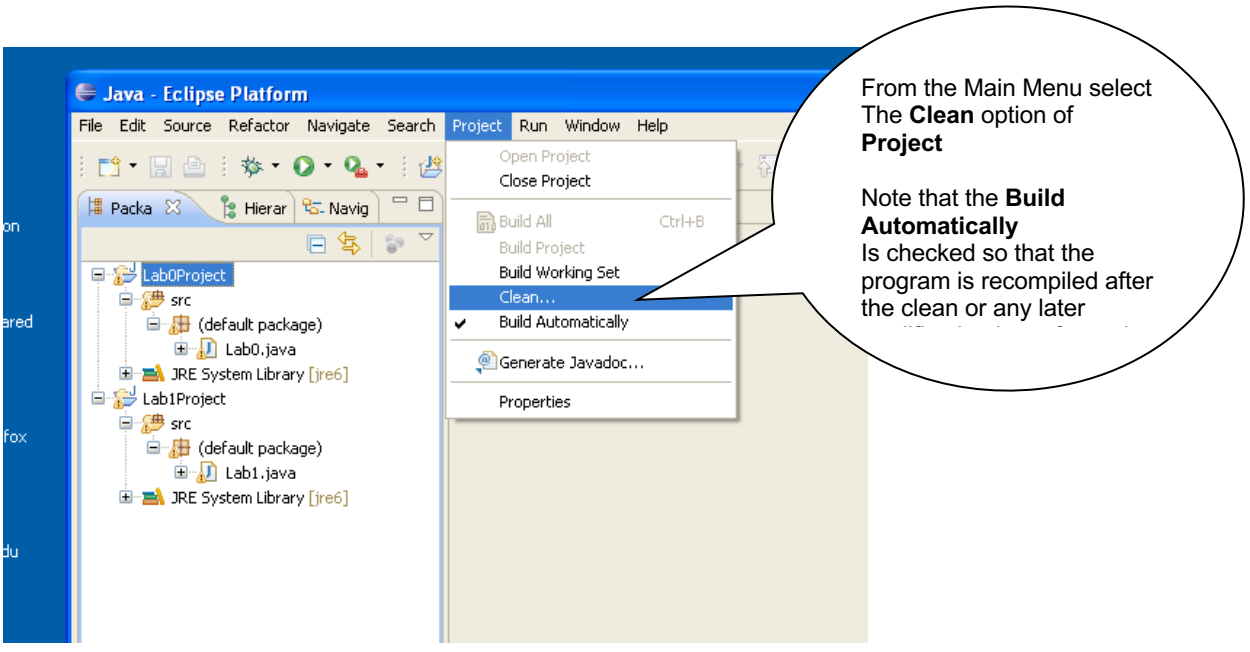
So far we have only used a small portion of the facilities of the Eclipse IDE. Gain more familiarity with the IDE by using the various facilities available. Here we explore some of them. Remember always to **BACK UP YOUR SOURCE FILES** outside of Eclipse to make sure you always have a way to get back if for some unexpected reason your lose some files while testing the IDE.

1. Refreshing and cleaning a project.

   Although most of the time you will be working within the IDE it may be possible that you modify a file of your workspace using a different editor. If this is the case when you go back to Eclipse be sure to select the project folder and using the right mouse button select the option Refresh, in this way the IDE will read the latest version in the workspace. *See the following example:*

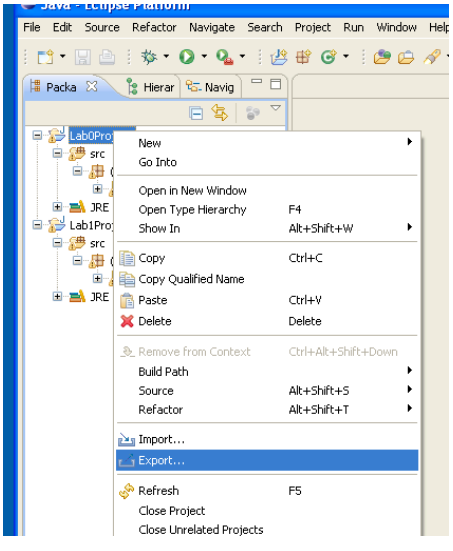


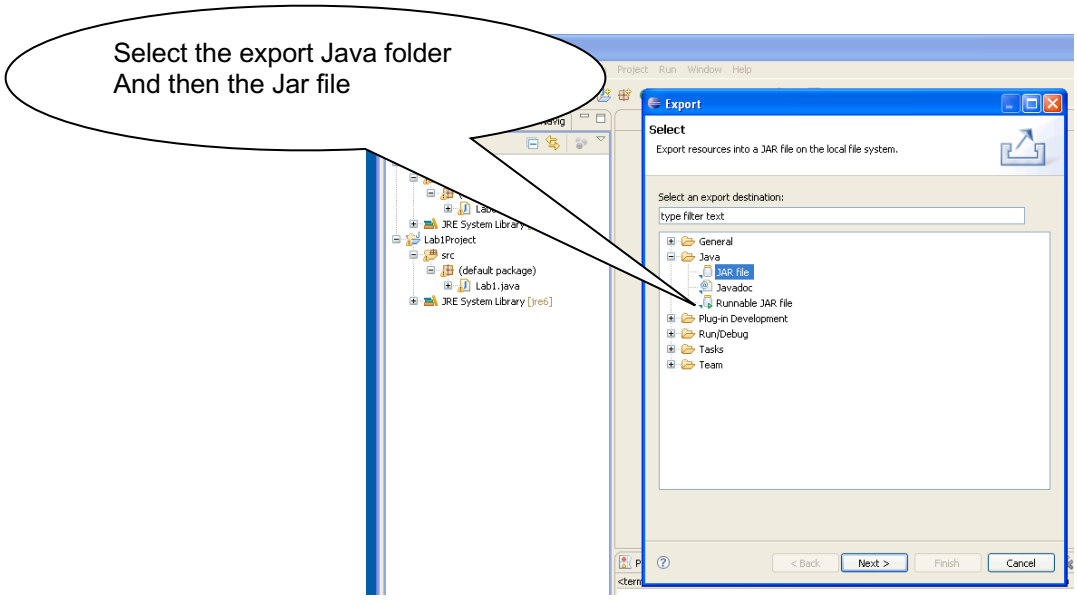
   When a project has been modify externally in its source or object code it is a good practice to clean the object code and rebuild it. This is done by selecting the option **Project… Clean** of the menu bar as shown below

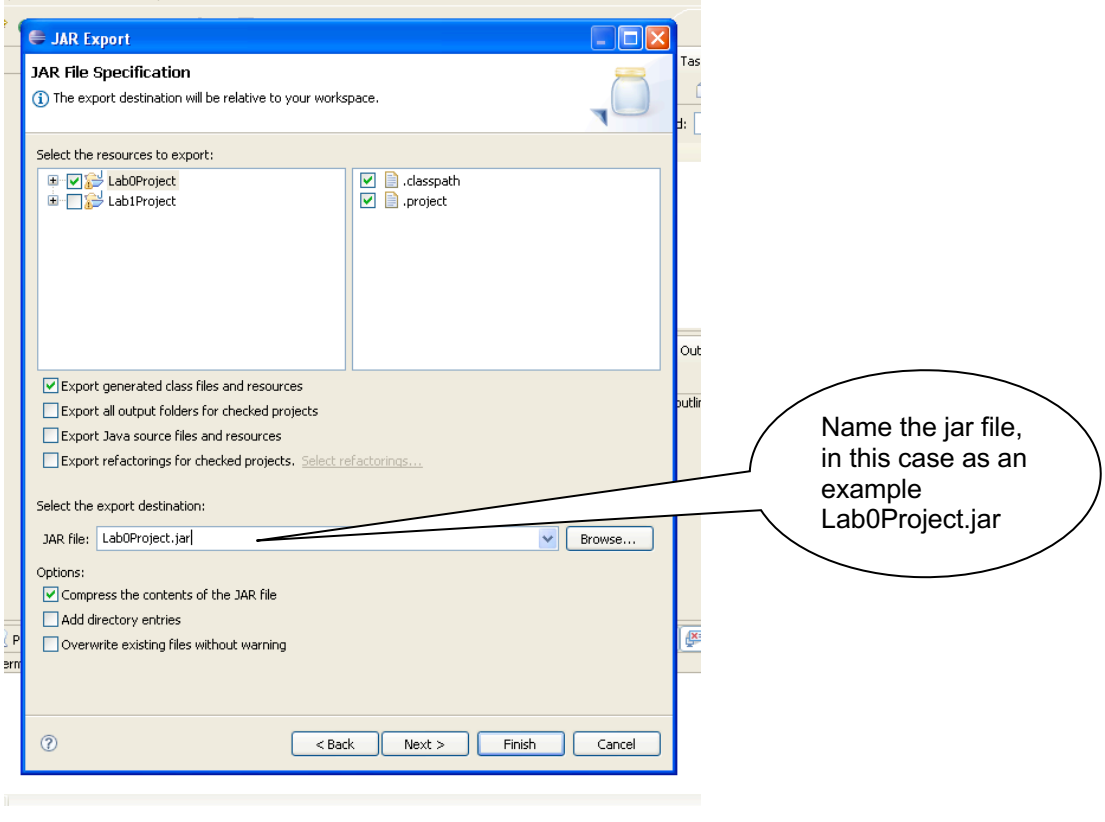


2. Exporting a project using a **Jar**.

In Java it is customary to wrap all the object class into one single compressed file, similar to a **zip** file. In Java these files are called **JAR** files (**J**ava **A**rchived **R**esources). Any project can be turned into a **Jar** using the export facilities available in Eclipse. Use the Right Mouse Button to select the option **Export** and follow the sequence of windows, as presented below:







Continue selecting **Next** until you get to the **Finish** and Eclipse will generate a **JAR** file that will be located in the project's workspace. The **JAR** file contains all the necessary file to run your program, for the case of Applets, you will change the **html** call to specify the location on the desired applet, For example in this case the new html code will have the following applet clause:

<applet archive='Lab0Project.jar' code="Lab0.class" width=200 height=200></applet>

Note that in this case the file is located in the archive named **Lab0Project.jar**

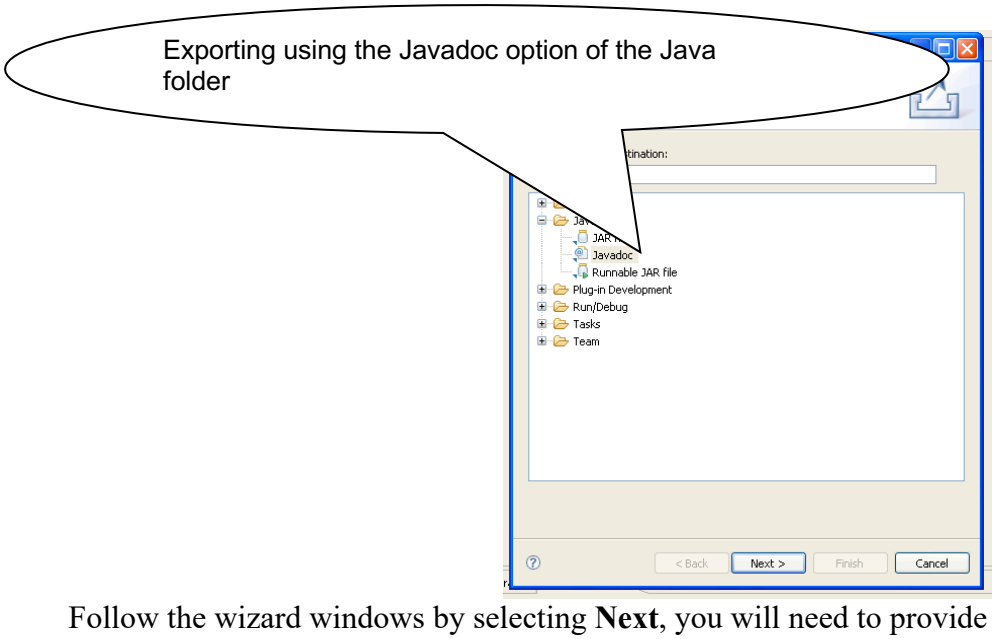
3. Documenting your code using JavaDoc

In any programming language it is important to document your work. For the case of Java we use the double slash (e.g. //) for inline comments and the /** */ for paragraph documentation. Assume we have a program that determines the Julian Number for a given date, the method to compute the number requires as parameters the day, the month and the year and returns a unique sequential number that identifies that date. A possible documentation for the method would look as follows:

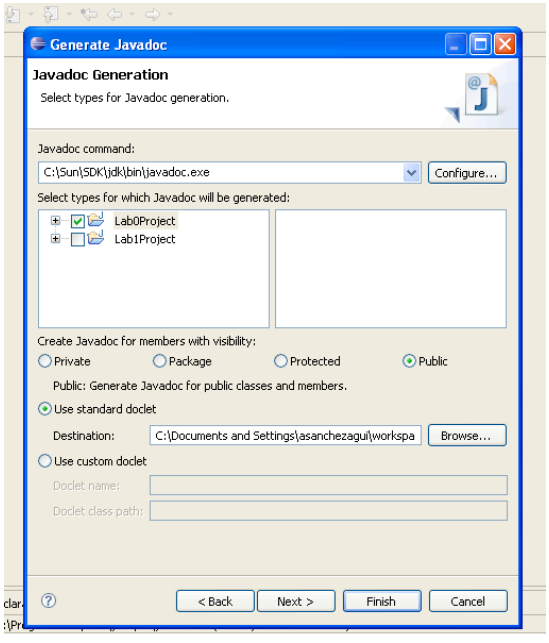
```
/**
   * This method computes the Julian Number
   * @param   day    (e.g.  1 to  31 )
   *  @param   month  ( e.g.  1 to 12 )
   *  @param   year    (  e.g.  an positive integer )
   *  @return  Julian Number
   */
public int calcJDN( int day, int month, int year) {
   //   here the code for the Julian Number method
   //    to do     instructions …….
}
```

But Java goes further in the documentation because it allows us to generate an standardized webpage documentation named the (Application User Interface) that we can use to learn how to use the specific Java class.

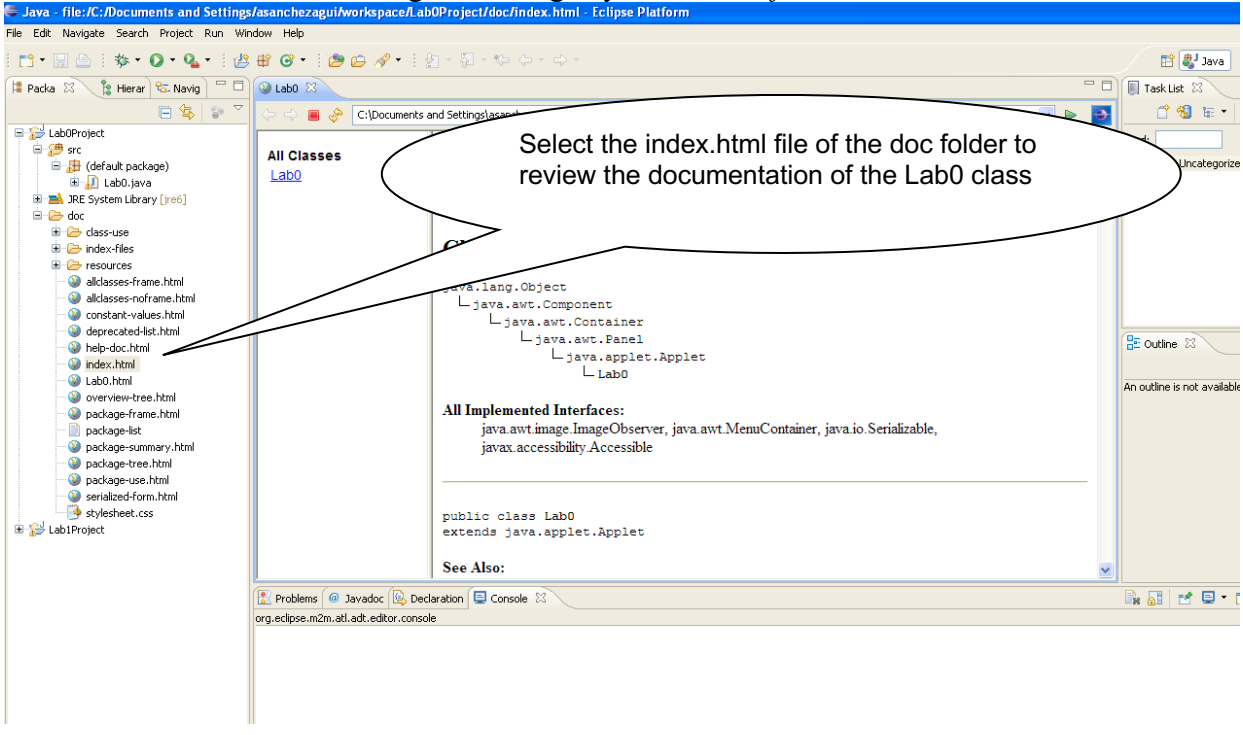
For the sake of practice using Eclipse using the **Right Button** select the export option of a given project as we did before; however this time select the option export as **Javadoc** in the corresponding window:



Follow the wizard windows by selecting **Next**, you will need to provide a name for the folder where you want the documentation to be placed or simple accept the default **doc** folder. Continue until you get to the **Finish** option.



Once you have finished you will find that you have a new folder with the documentation of the program. This folder contains the **API** documentation of your class and you can select the **index.html** file to start the navigation using any browser *as follows:*



In order to get meaningful and useful information in the API it is your responsibility to document your source code using the /** */ standards described before. As your get a better understanding of the Java language you will notice that only the public methods are documented in the API. Look for the Fahrenheit Applet in the webpage of the course for a complete example of proper documentation of Java Programs.