# Lab 0 - Preliminaries

Walkthrough of the lab computing environment: Introduction to CentOS, Linux Command Line, Eclipse and Submission

| | |
|---|---|
| Site: | Moodle@York |
| Course: | LE/EECS1710 A - Programming for Digital Media (Fall 2018-2019) |
| Book: | Lab 0 - Preliminaries |
| Printed by: | Matthew J Kyan |
| Date: | Saturday, 8 September 2018, 5:41 AM |

# Table of contents

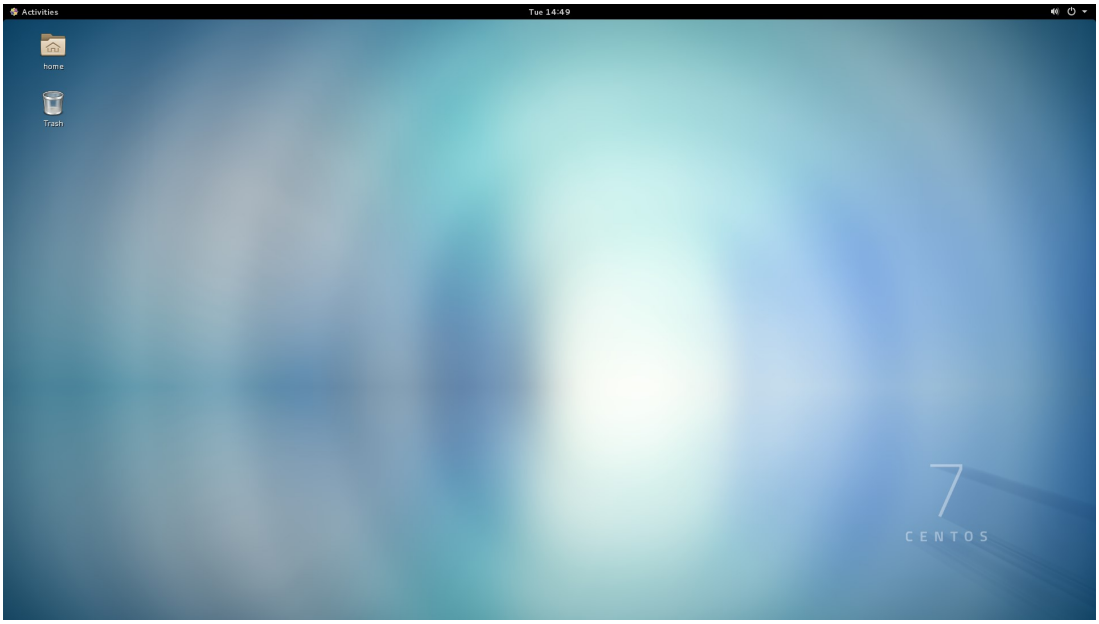
# 1. Preamble

This is a self guided lab that walks you through the EECS PRISM lab environment, including a brief look at configuring look and feel of CentOS, some introduction to basic Linux commands (in the terminal), The Eclipse IDE, and creating a basic "Hello World" java application.

# 2. CentOS

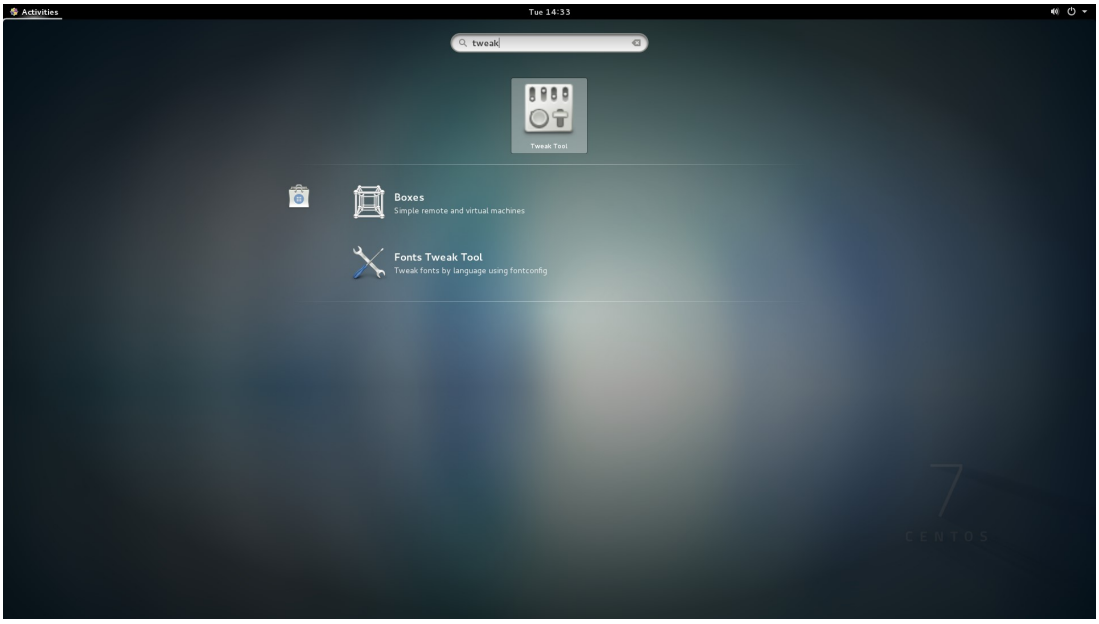
To begin, find a workstation in the Prism lab and log in. Your user name and password are the same as for your Passport York account.

The lab computers you are using run a Linux operating system called CentOS. The default desktop environment is called GNOME. The basic funtionality of the desktop environment is similar to that provided by Microsoft Windows and Mac OS X.

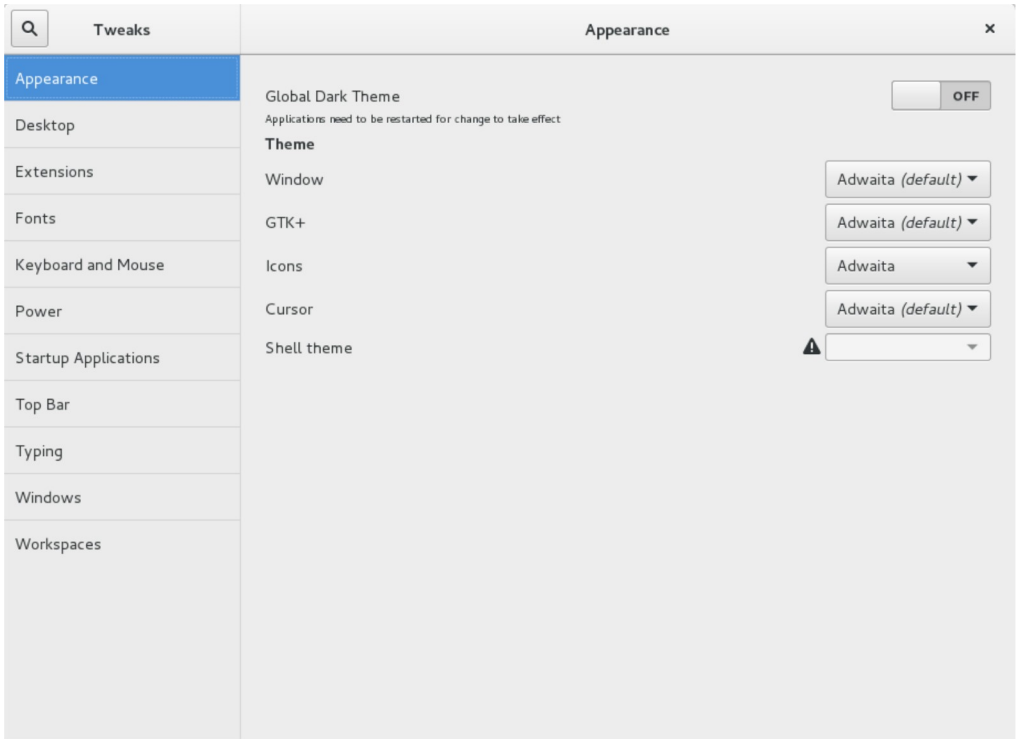



The default user interface provided by GNOME is minimalist which might make it difficult for new users to figure out how to do things. What follows is a list of instructions for customizing the user interface so that it displays more information to the user. You can easily undo these changes as you become more comfortable with the desktop environment.

Click the **Activities** button in the upper left corner of the screen. In the search bar type tweak and the Tweak Tool icon should appear. Click on the **Tweak Tool** icon.




The Tweak Tool application should appear. It allows the user to adjust certain aspects of the desktop environment. In the left column you will see the categories of tweaks that can be adjusted by the user.



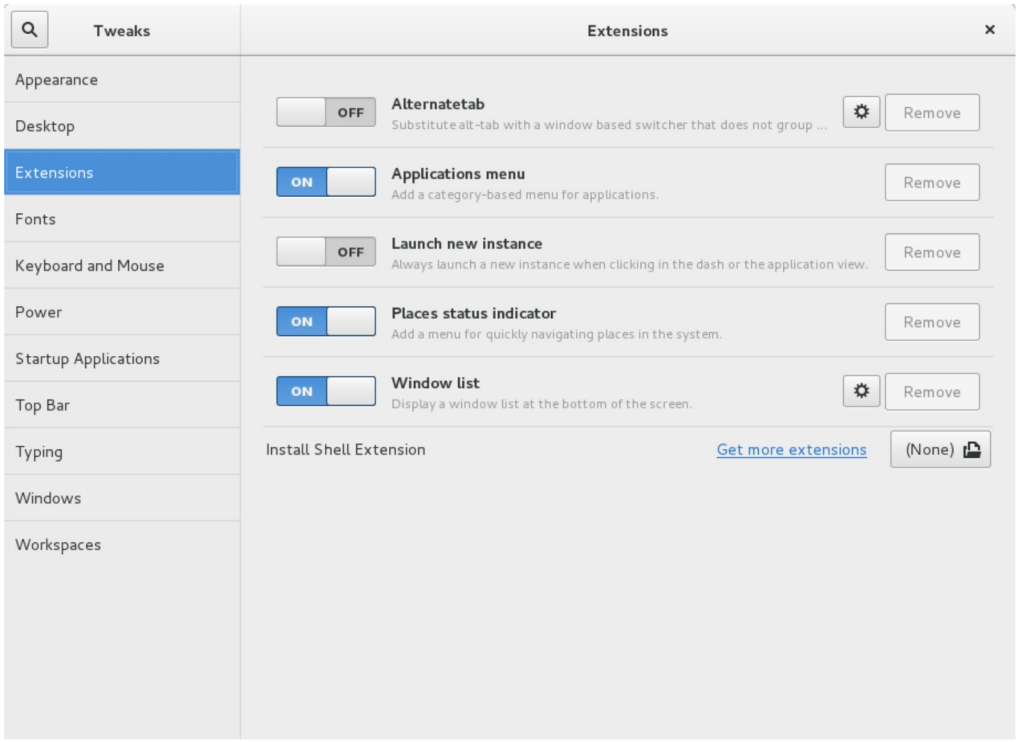

Click the *Extensions* category. Adjust the ON/OFF switches so that they have the settings shown below.

The *Applications menu* switch enables a menu that is similar to the Windows Start menu.

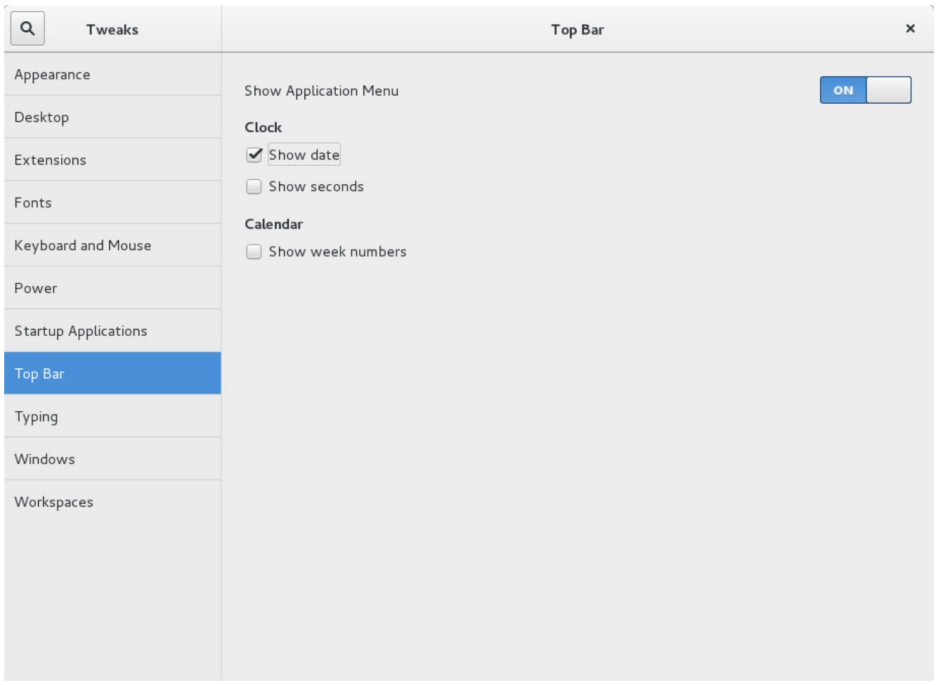
The *Places status indicator* switch enables a menu similar to the Windows *This PC* category in Windows Explorer.

The *Window list* switch enables a list of the currently running desktop applications at the bottom of the screen (similar to the Windows task bar).

*** each of these will re-create a similar feel to your EECS VM (if setup on your own computer - see tutorial section of coursewebsite)*

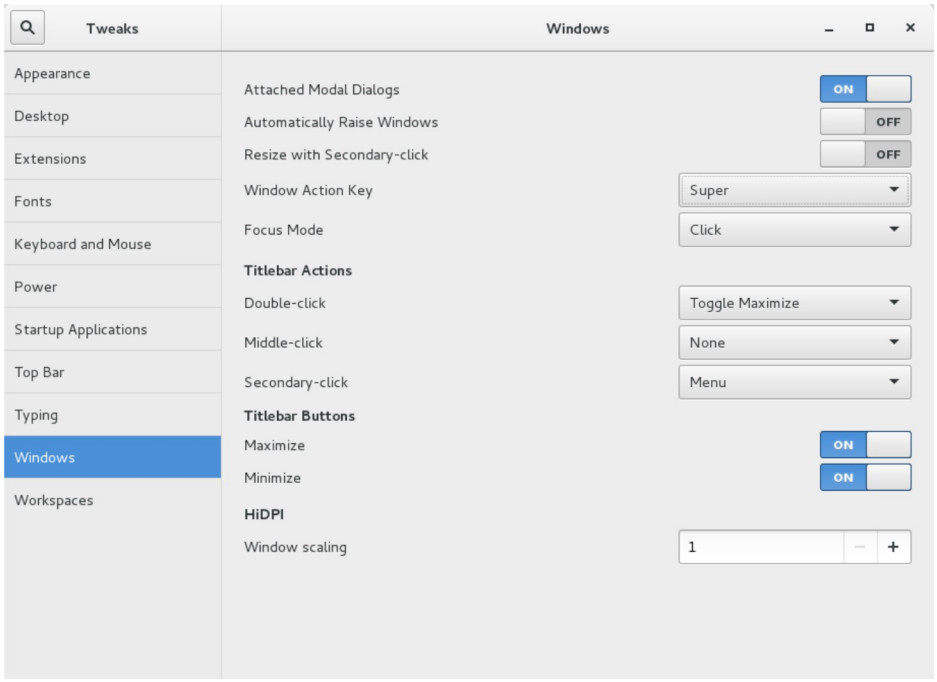



Click the *Top Bar* category. If you want to, check the *Show date* box so that the current date appears beside the clock at the top center of the desktop.




Click the *Windows* category. Adjust the ON/OFF switches for *Titlebar Buttons* so that they have the settings shown below.

These switches enable a minimize and maximize button to appear in the top right corner of an application window. By default, only a close button appears in the top corner of an application window.




Close the **Tweak Tool** application by clicking the close button in the upper right corner of the window, or use the *Tweak Tool* menu found near the top left of the screen.

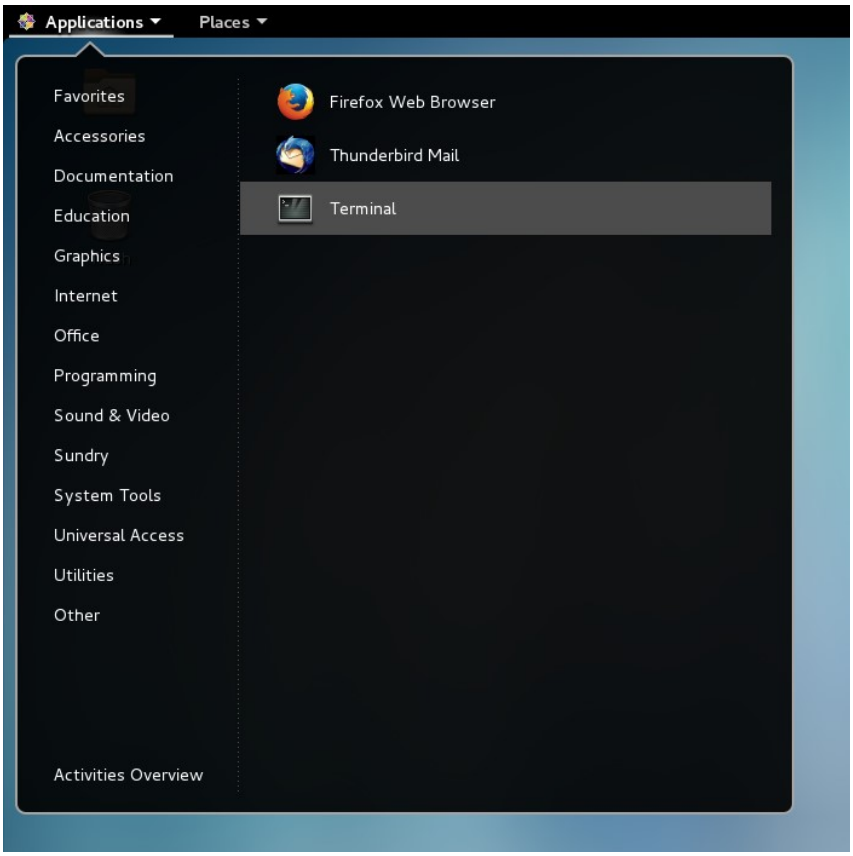
# 3. The Linux Terminal

Quoting the **Introduction to user commands** manual entry:

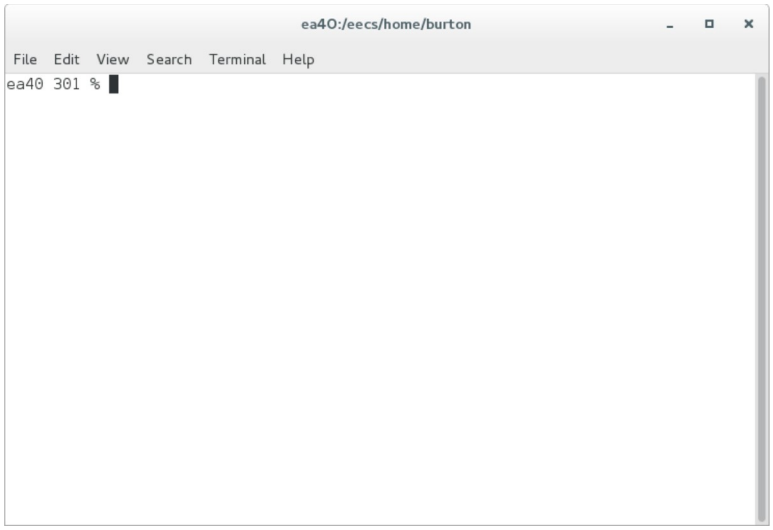
*"Linux is a flavour of Unix, and as a first approximation all user commands under Unix work precisely the same under Linux (and FreeBSD and lots of other Unix-like systems).*

*Under Linux there are GUIs (graphical user interfaces), where you can point and click and drag, and hopefully get work done without first reading lots of documentation. The traditional Unix environment is a CLI (command line interface), where you type commands to tell the computer what to do. That is faster and more powerful, but requires finding out what the commands are."*

The Terminal program provides a command-line interface called a *shell* where you type commands to tell the computer what to do. Start the Terminal program using the *Applications* menu.




A window similar to that below should appear:




The prompt ea40 % indicates that the name of this computer is ea40. You will see a different name corresponding to the name of your computer. You will also see a number beside the name (probably the number 1); the number is the history number of the current command line.

In EECS1710, it is important that you become comfortable working with files from the command line because this is how you will be submitting your program files for grading (including during tests).

You are probably already familiar with the notion of a *computer file* that represents some piece of information on a storage device (e.g., MP3 music files, JPEG picture files, word processor documents).

You are probably also aware that you can use *folders* to organize your computer files (e.g., under Windows you have Desktop, Documents, and Favorites folders). You can even put folders inside of other folders.

You might also know that every file and folder has a unique name called the *absolute path* or *absolute pathname*. The absolute path is simply the full list of folders that you need to traverse to reach a particular file or folder. For example, in Windows the the absolute path **C:\Users\Bob\Music** refers to Bob's **Music** folder, and the backslashes \ are used to separate folder names.

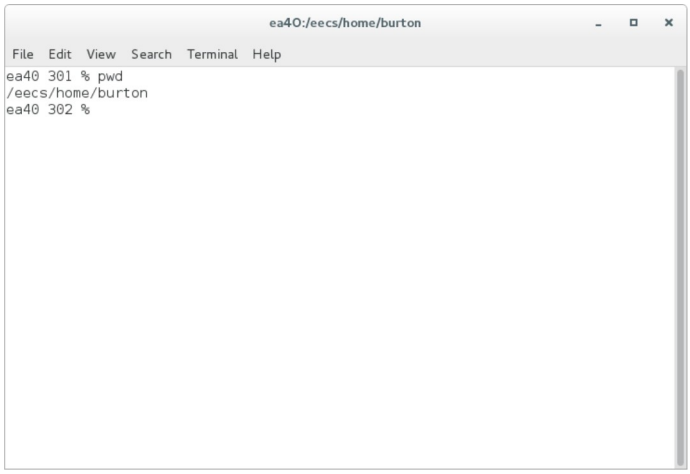
In Linux, it is more common to use the term *directory* instead of folder.

Your *home directory* is where all of your EECS account files are stored. Of course, you can (and should) create directories inside of your home directory to organize your files. Every person with a EECS account has their own home directory.

No matter which lab computer you use, your home directory always has the same absolute pathname.

By default, the shell will start in your home directory.

To see the full pathname of your home directory, invoke the **pwd** command by typing it in the terminal and pressing the ***Enter*** key. You should get a slightly different result than that shown below.



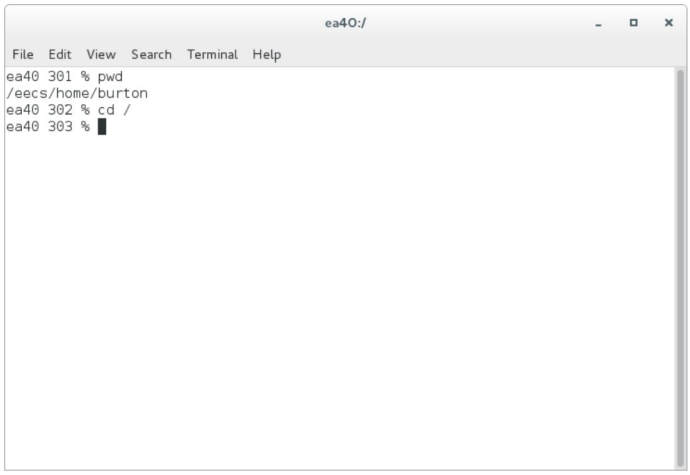

**pwd** prints the full path of the current working directory.

Notice that the terminal you are using has been pre-configured to display the current working directory in the title bar.

Linux uses a single hierarchical directory structure to organize all of its files. This is different than Windows where each drive (C:, D:, etc.) has its own separate directory structure.

The top-most directory in Linux is called the **root** directory, and is identified by the slash character /.

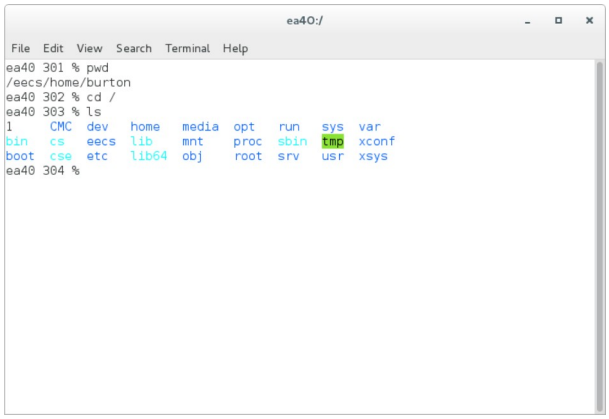
Use the change directory command **cd** to switch to the root directory.




Use the command ls ("ell-ess" not "one-ess") to list the files contained in the current working directory.

*You might not get the exact same output as shown in the figure because the EECS computer systems are regularly updated. Also, if you do this exercise from your home VM setup, you will actually be browsing only files local to your machine (again results will differ).*

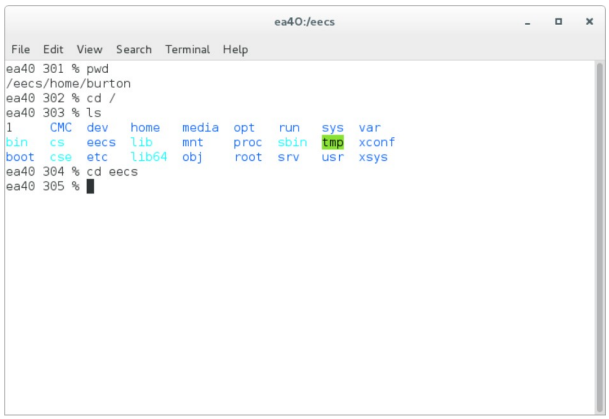
The names listed in blue are all directory names. Many of these directories are common to all standard Linux distributions, and are described by the Filesystem Hierarchy Standard.




Notice that the root directory contains other directories. Those directories in turn contain other directories, and so on. One of the directories in the root directory is named **eecs**.

Use the change directory command to switch to the eecs directory.

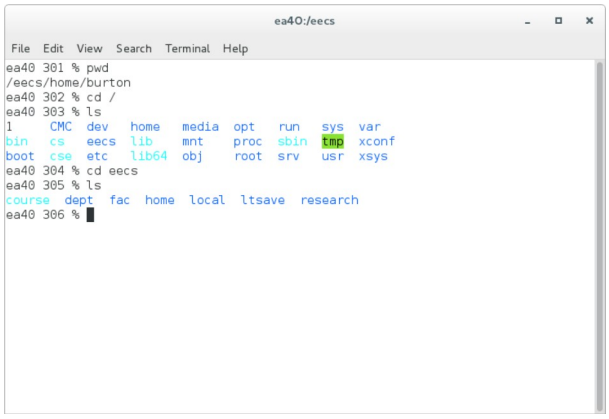
Notice that you did not need to specify the full path **/eecs** because you were already in the root directory **/** (you can always use the full path if you want).




The eecs directory contains all of the files and directories belonging to people in the Department of Electrical Engineering and Computer Science.

Use the command **ls** to list the files contained in the current working directory.

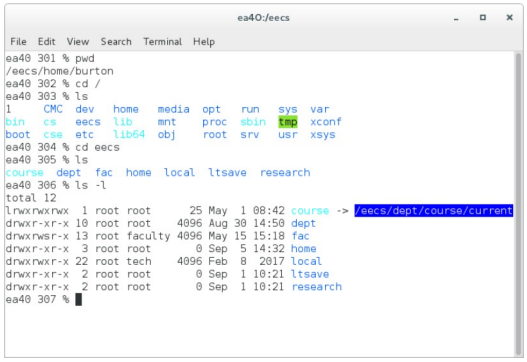
Again, you may see some differences from the figure.




Many of the Linux command-line programs allow the user to specify optional flags that control how the program should behave. The flags are often (but not always) specified using a hyphen **-**.

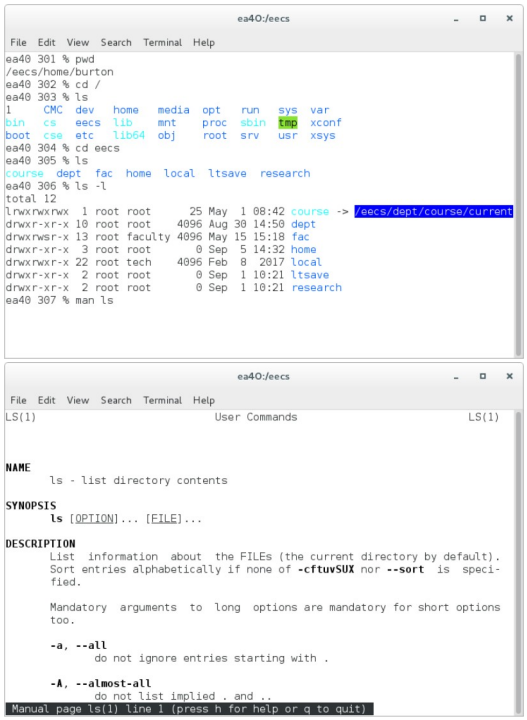
Use the long format flag **-l** ("hypen-ell") with **ls** to display extra information about the files in the current working directory.

The long format listing produces a lot of output, most of which won't make sense to you at this time. We may revisit these details later in the course.




Even expert users have a hard time remembering all of the options for all of the commands. All of the commonly used command-line programs have a manual page that you can access using the man program.

Access the manual for **ls** using the command **man ls**




You can move forward in the manual by pressing the ***Space*** key. Move backwards by pressing the ***b*** key. Quit using the ***q*** key.

ls has a large number of optional flags. Don't worry about trying to understand them all right now; just get used to navigating the man page, but keep the following questions in mind.

What flag would you use to:

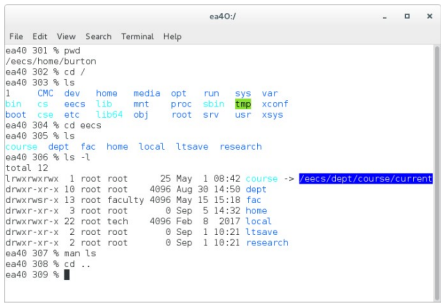
- list files in reverse order?
- list one file per line?

Try out your answers; did they do what you expect?

Recall that we are currently in the directory **/eecs**.

To move up one directory to the parent directory you use the command **cd ..** (which in this case will take us to the root directory).

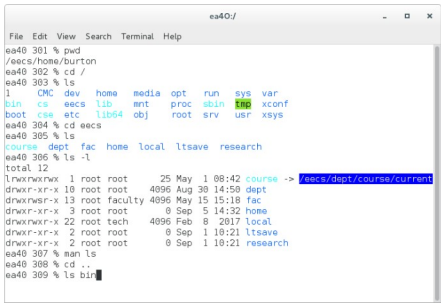
The shell interprets **..** as being the parent directory to the current working directory.



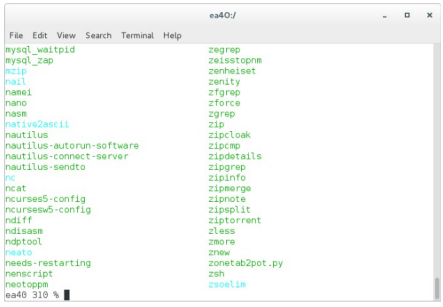

The **ls** command by itself will list the contents of the current working directory. If you want to list the contents of a different directory, you can specify the name of the directory.

**ls bin** will list the contents of the **bin** directory.

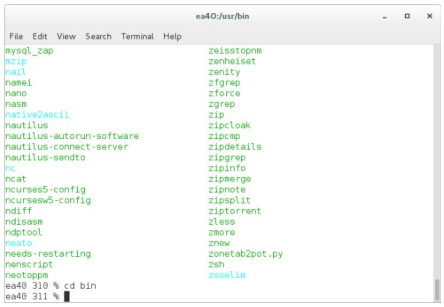
The directory **/bin** contains many of the essential command-line programs.




There are quite a few files in **/bin**. The files listed in green are all programs runnable by the user.




Use the change directory command to switch to the bin directory.

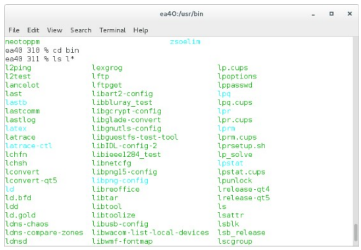


The **ls** command lists information about the files you specify. As you have already seen, if you specify no files, then **ls** lists the files in the current working directory.

Suppose that you want to list all of the files that begin with the letter l. You can do this by using the command **ls l***

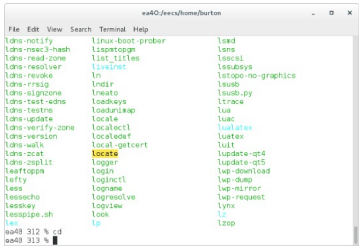
Most Linux shells will interpret the asterisk * to mean "any sequence of characters"; thus, **ls l*** means "list all files with a name starting with l followed by any sequence of characters".

Notice that the **ls** command is found in **/bin**.




You can return to your home directory by using the **cd** on its own. If you don't specify which directory to change to, **cd** assumes you want to go to your home directory.

You could also use **cd ~** (where ~ is the "tilde" character usually located on the top left key of the keyboard).




** we will learn more commands for the terminal as we progress through the course.

## Summary

| Command | Meaning |
|---|---|
| pwd | Print the full (absolute) pathname of the current working directory. |
| cd *dirname* | Change to the named directory. |
| cd .. | Change to the parent directory of the current working directory. |
| cd | Change to the user's home directory. |

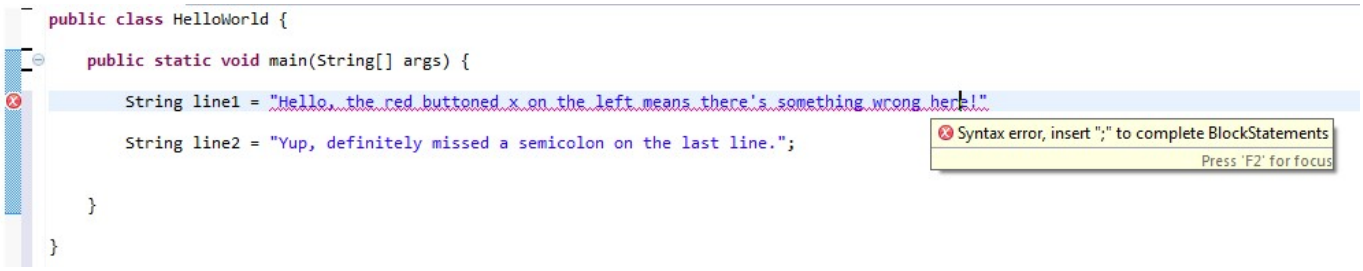
| | |
|---|---|
| ls | List the contents of the current working directory. |
| ls *dirname* | List the contents of the named directory. |
| ls -l | List using long format the contents of the current working directory. |
| ls -l *dirname* | List using long format the contents of the named directory. |
| ls -ld *dirname* | List using long format the name (but not the contents) of the named directory. |

# 4. The Eclipse IDE

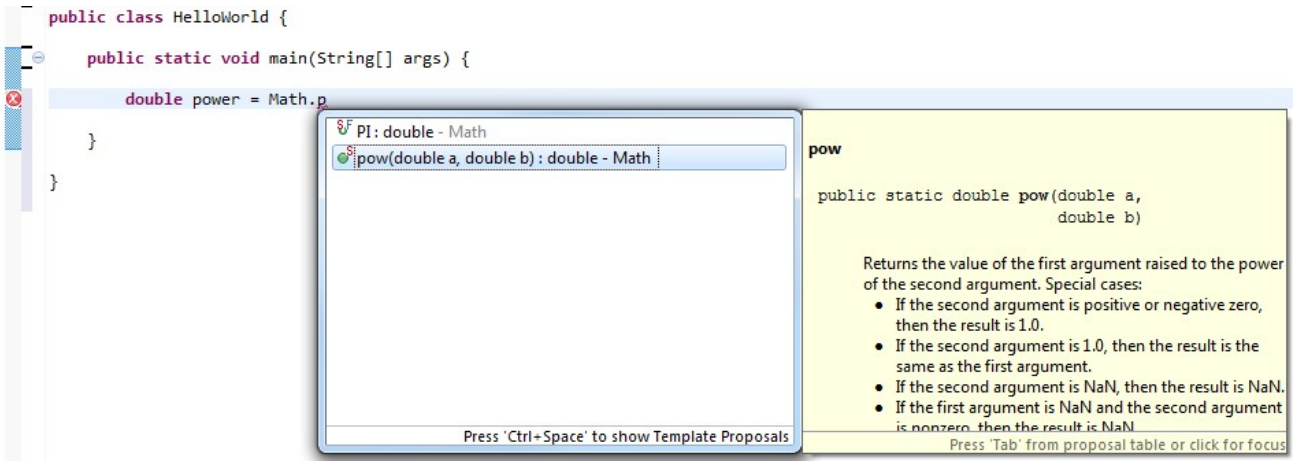
What is Eclipse? Eclipse is an IDE (Integrated Development Environment), meaning that it is a program made to provide you with all the tools you need to code, wrapped up in a nice user interface.

Key features include:

**Automatic errordetection and highlighting:** so that you can identify errors in your code quickly:




**Autocomplete/Content assist:** similar to predictive text, as you type a function you are thinking of using, it will suggest possible functions and give you an idea of their syntax




**Built in compiler:** so that you can run code at the click of a button

ConsoleExample.java

```
import java.util.Scanner;

class ConsoleExample
{
   public static void main(String args[])
   {
      int a;
      String s;

      Scanner in = new Scanner(System.in);

      System.out.println("Enter a string");
      s = in.nextLine();
      System.out.println("You entered string "+s);

      System.out.println("Enter an integer");
      a = in.nextInt();
      System.out.println("You entered integer "+a);
   }
}
```

Problems @ Javadoc Declaration Console

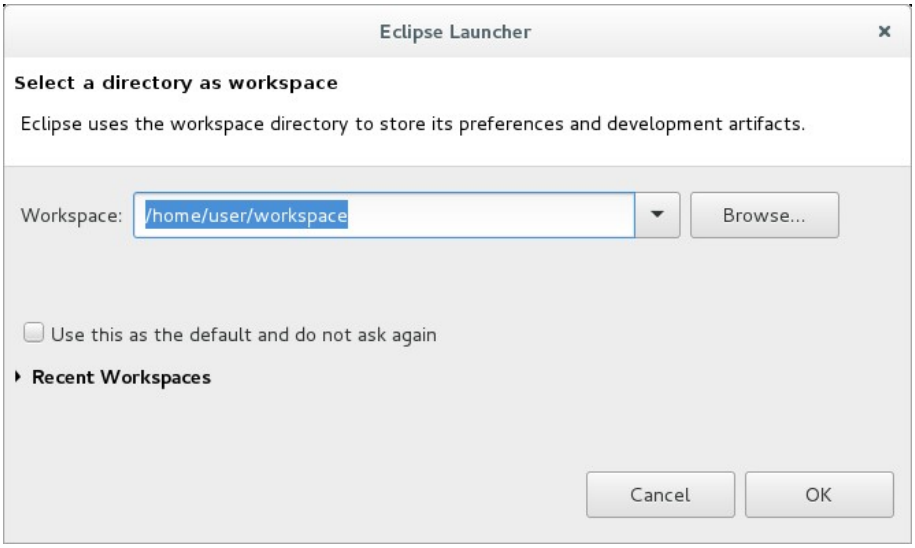
```
<terminated> ConsoleExample [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (Aug 23, 2013 12:36:19 PM)
Enter a string
Hello!!
You entered string Hello!!
Enter an integer
12345
You entered integer 12345
```

**And a debug mode:** to help you find errors and where they occur in your program

** of course there are many other features too!

## 4.1. Basic interface

Start up the application "Eclipse" which is located under the "Applications -> Programming -> Eclipse" drop down menu. Once it is run, you will be greeted with a screen which will look something like this:



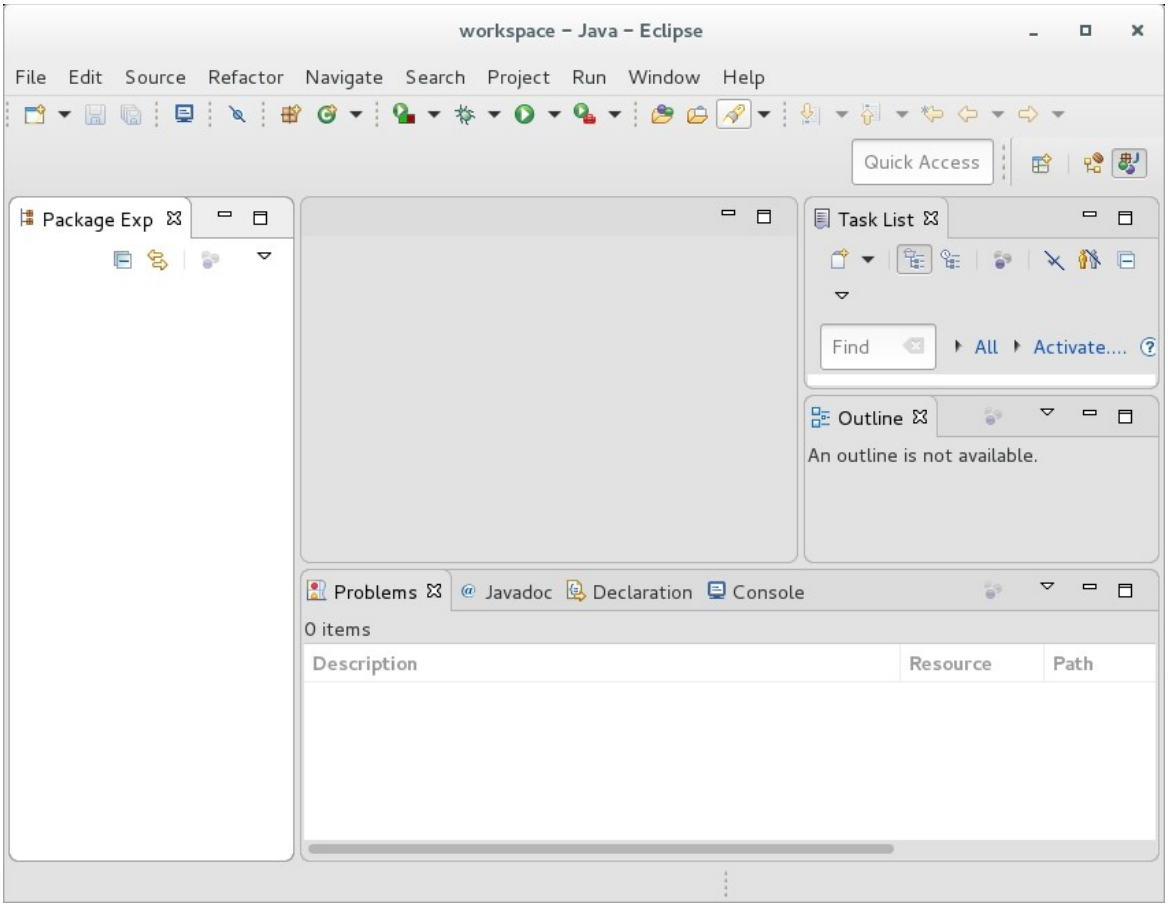

Keep the default settings,

you will be setting up code projects within the 'workspace' folder on your home directory. You can use the terminal later to browse this directory and see the files that are generated when you create new Java programs.

After clicking 'ok', the main interface will start. At this point you may be presented with the "Welcome to Eclipse" tabbed pane. You can dismiss

this for now (click on the 'X'). You can look at it later by looking under **Help -> Welcome**. You might like to try some of the basic tutorials later to become more familiar with the interface itself.

For now, the app will look something like this:




The window above shows a set of four "views" arranged into a "perspective". This particular configuration of views is called the "Java Perspective". We will look into each of the four

"views" (clockwise, starting from top left):

- the "Package Explorer" view
- the editor view (not labelled, the big empty square in the middle)
- the "Outline" view (we can dismiss the "task" view for now to simplify this to 4 views)
- a view consisting of a set of several tabbed panes (labelled Problems, Javadoc, Declaration, Console)

Use **Window -> Perspective -> Open**

**Perspective** in order to select the "Java Browsing" and then the "Debug" perspective. This will demonstrate to you what a few different perspectives look like.

Essentially each "perspective" configures the IDE for a different mode of operation (e.g. debug perspective is tailored to tracking what is happening as your code executes, while the original java perspective is setup for editing files and running

them, etc.)

Switch back to the Java Perspective. Use **Window -> Perspective** to do this. Alternatively, there are also short-cut buttons to switch the view in the upper right-hand corner. Holding the mouse over a button will reveal a tooltip description.

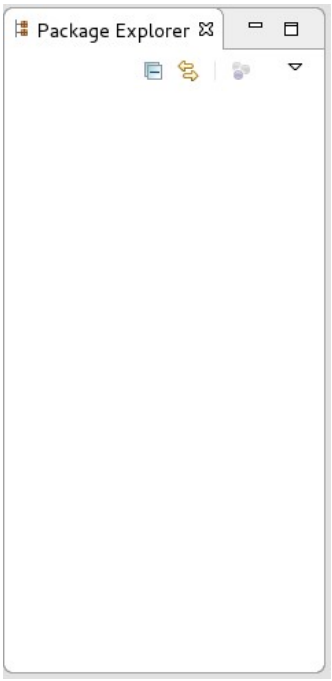
**HINT:** If you ever find yourself in the wrong perspective, you can always get to the Java

Perspective using **Window -> Perspective**.

Now we're back in the Java Perspective. Let's have a look at the editor view. It is the big empty pane in the middle of the window. The title bar is empty, but you can see the title bar *decorators* are there (e.g., to minimize and maximize the view). Once there is a file open within

Eclipse, this view will be populated with content (but at this point you have no project nor any files).

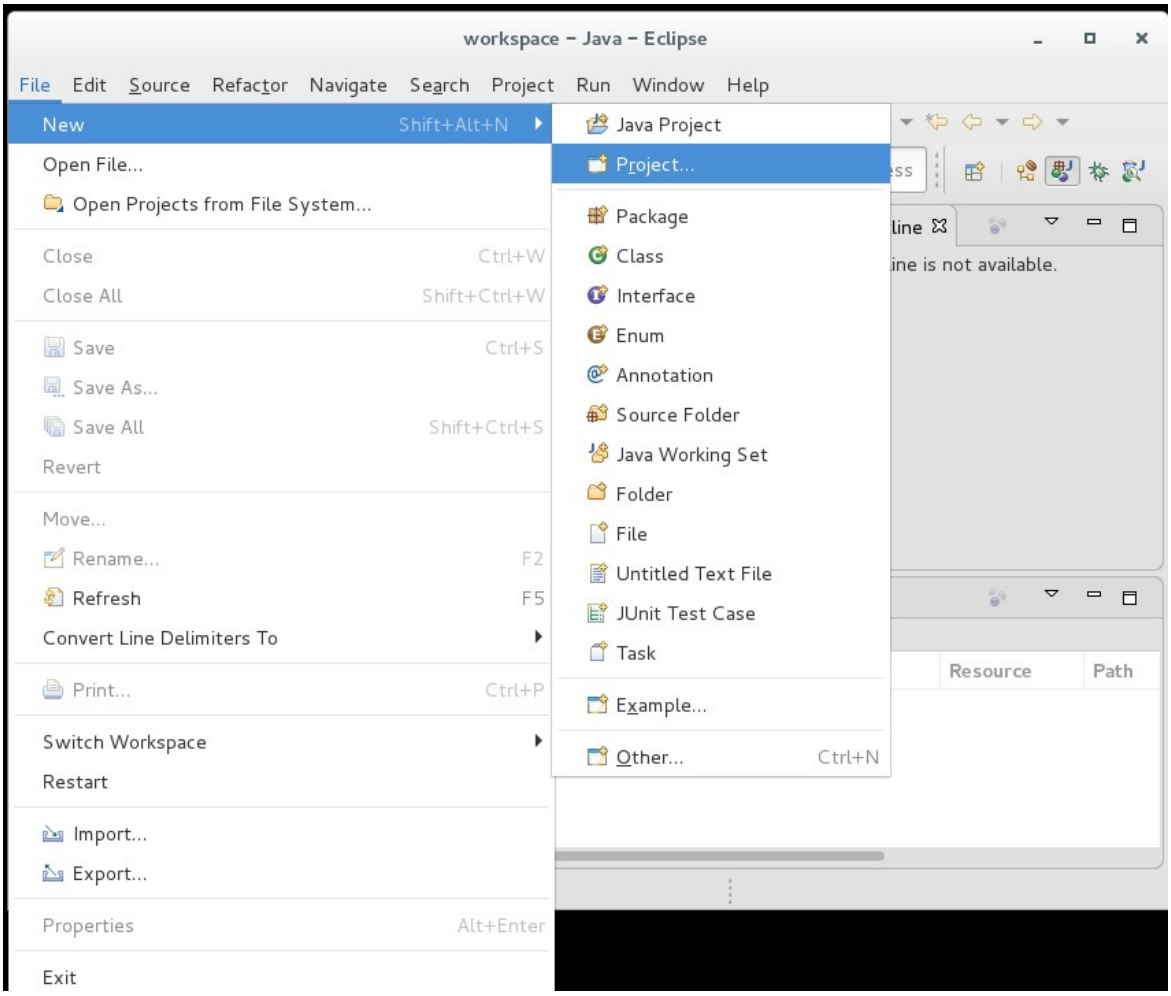
Now let's look on the lefthand pane at the "Package Explorer" view. This is where all of your projects will be listed (as you create/import them). It is shown below


Package Explorer

## 4.2. Creating a new project

Create a new Project called "`1710-lab01`" by selecting **File** > **New** > **Project** and then selecting the **Java Project** wizard

Your version of Eclipse may have the option **File** > **New** > **Java Project**. If so, select that instead and then select the **Java Project** wizard




When you select the wizard, the **New Java Project** dialog box appears as shown below. In this dialog, do the following:

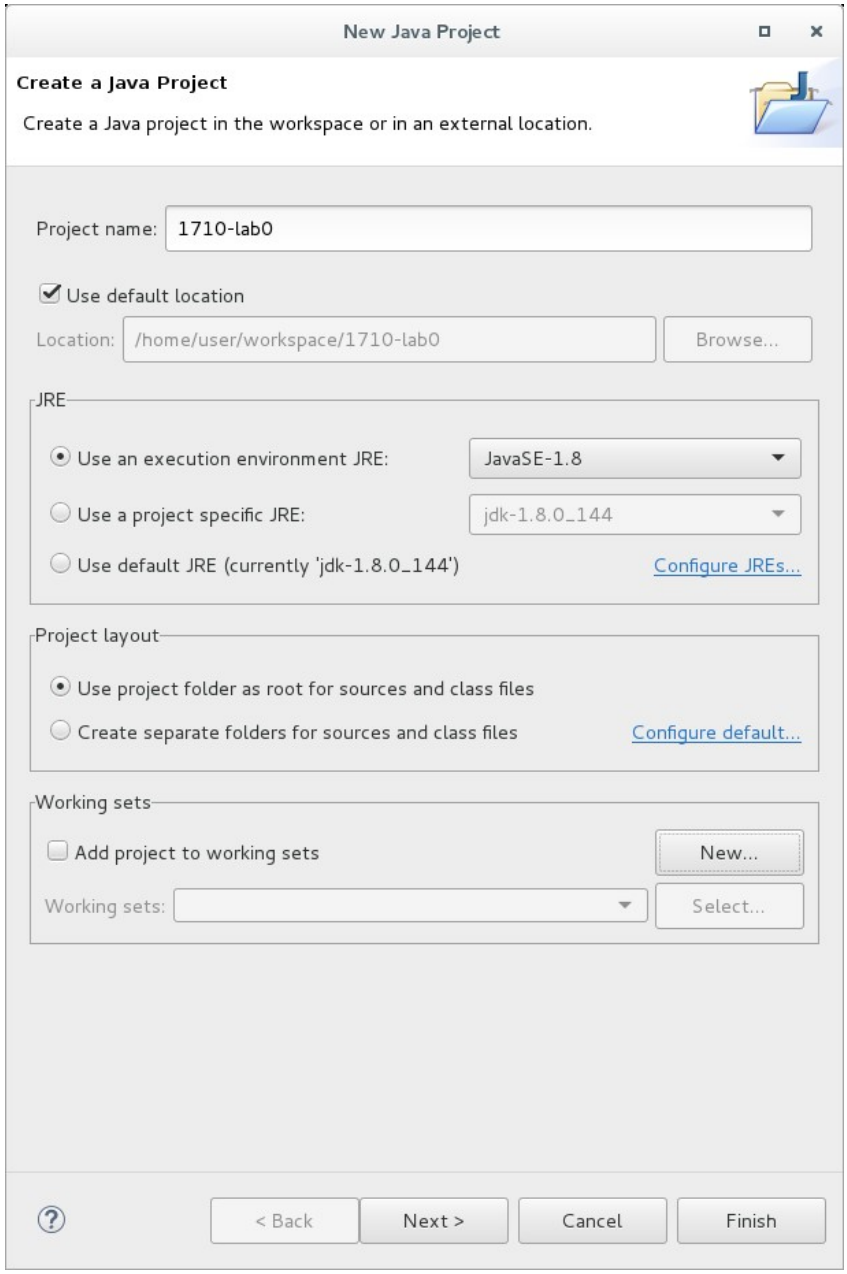
- In the **Project name:** field, enter `1710-`

`lab01` (see below).

- Under **JRE**, leave the option **Use an execution environment JRE JavaSE-1.8** as selected (see below). JRE stands for "Java Runtime Environment" and consists of the Java Virtual Machine (JVM), Java platform core classes, and supporting Java platform libraries. The JRE is the **runtime** portion of Java software.
- Under **Project Layout**, choose the option **Use project folder as root for**

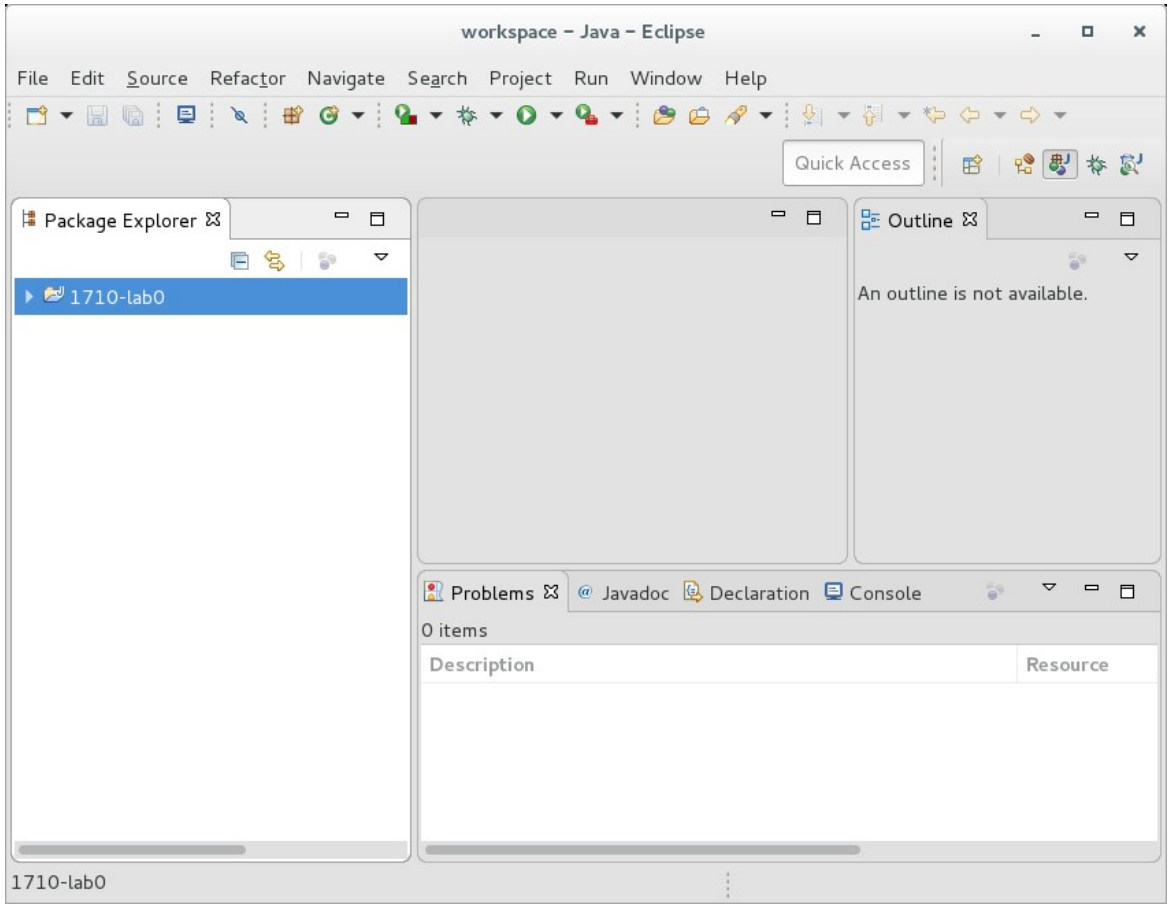
**sources and class files** (see below). NOTE: The other option **Create separate folders for sources and class files** is selected by default, but we do *not* want that option.

- Leave everything else as-is.
- Select the **Finish** button.




Next we are returned to the Java Perspective. In Package Explorer view, you will now see a

folder named "`1710-lab01`". An example is shown below (the project name is slightly mismatched to this example).

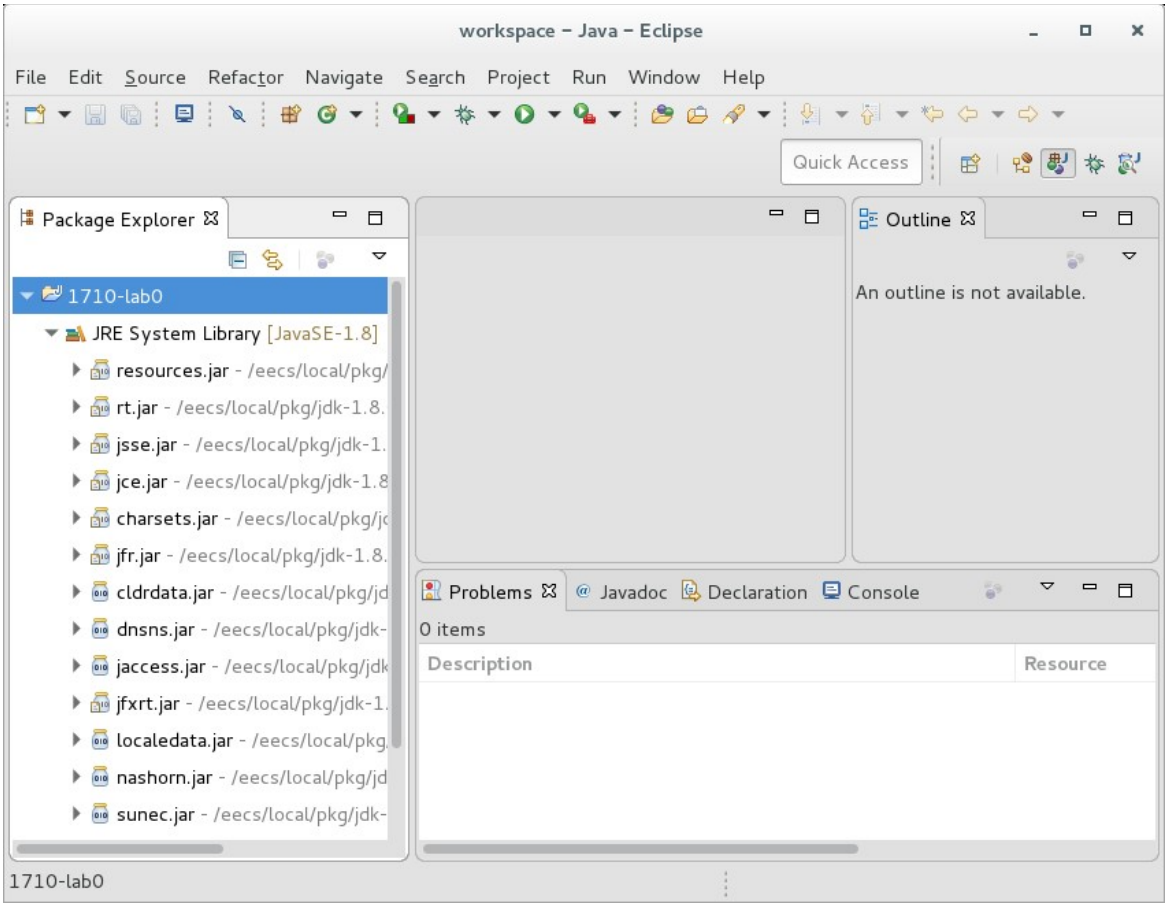



Now we will expand the project fully. Click on the little triangle to

the left of the package name to expand the folder. Do you see a sub-directory named "src"? This means you DID NOT follow the instructions correctly above. Delete the project (right-click on the project and select DELETE from the popup menu). Repeat the steps above.

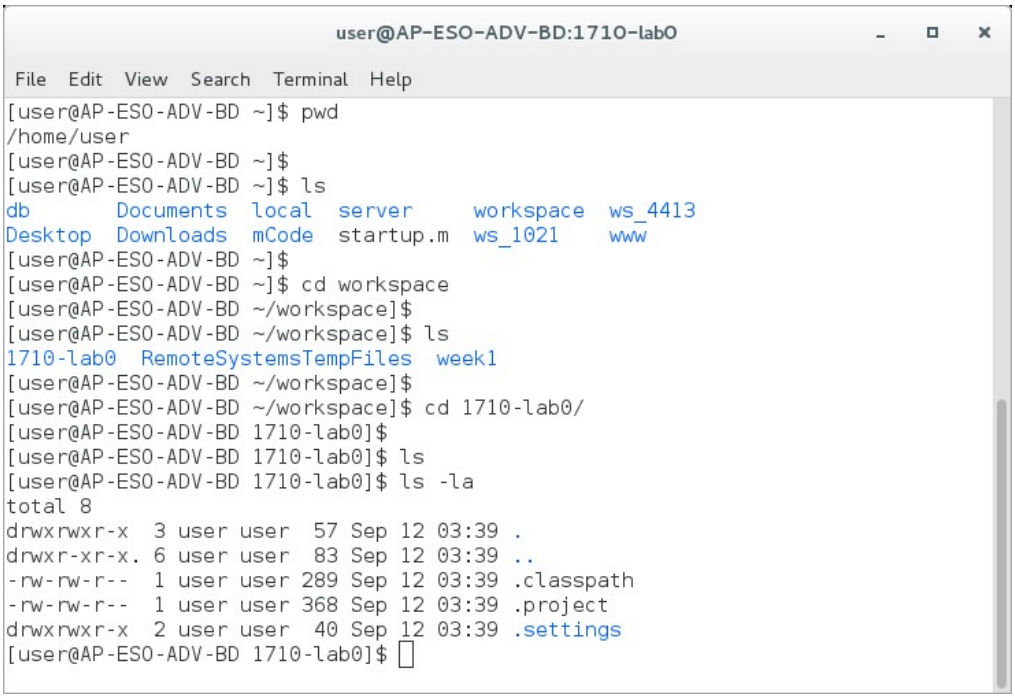
You should see a project entry labelled "JRE System Library" (see below). Click on the little

triangle to the left. You will see many resources that are available for your project. They were placed there automatically by Eclipse. It will look something like the screen shot below (depends on your particular system).




At this point you can go and run the terminal again and check out the files that were created in the workspace! You should see something similar to this (although there

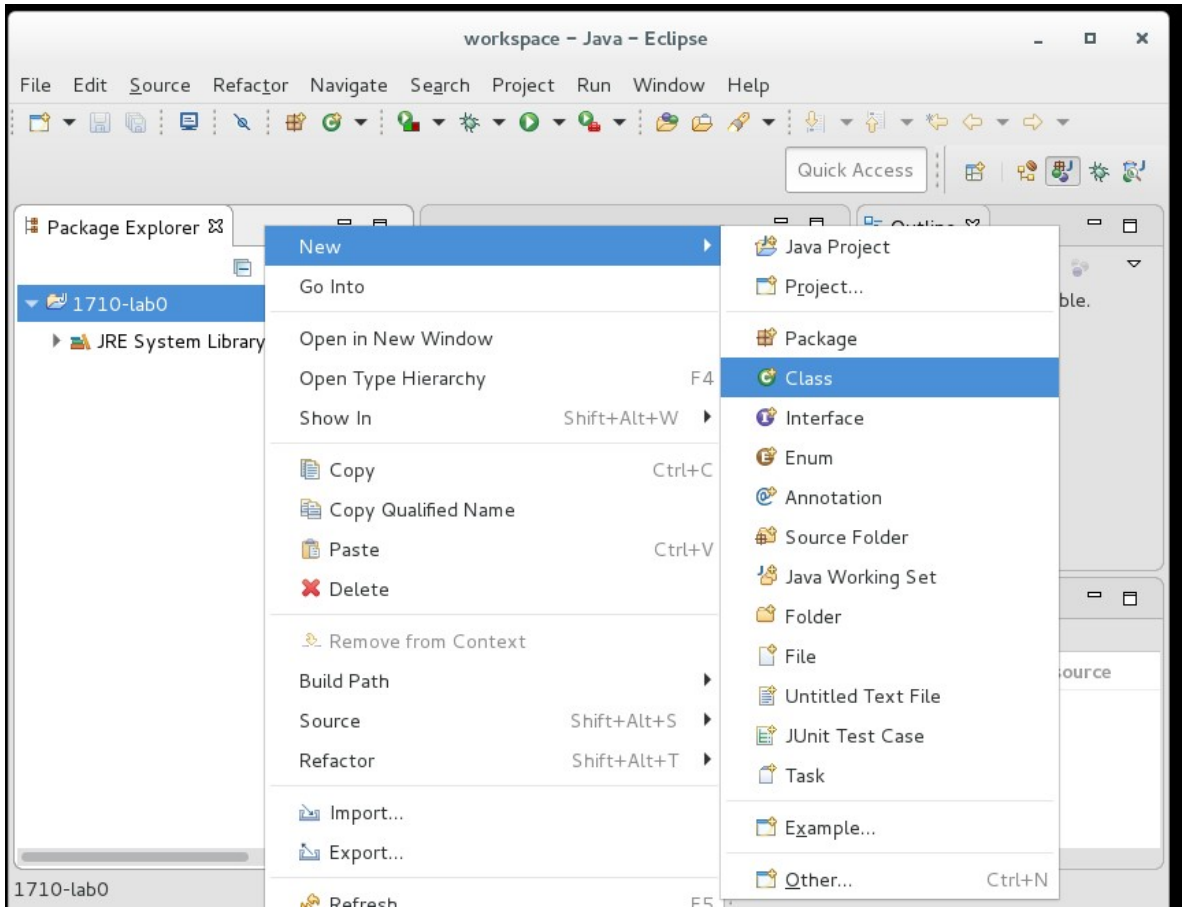
are some extra files in this version). This example runs several commands, and shows the output for each (pwd, cd and ls - within various directories).




HINT: you can clear your terminal window by typing **clear**

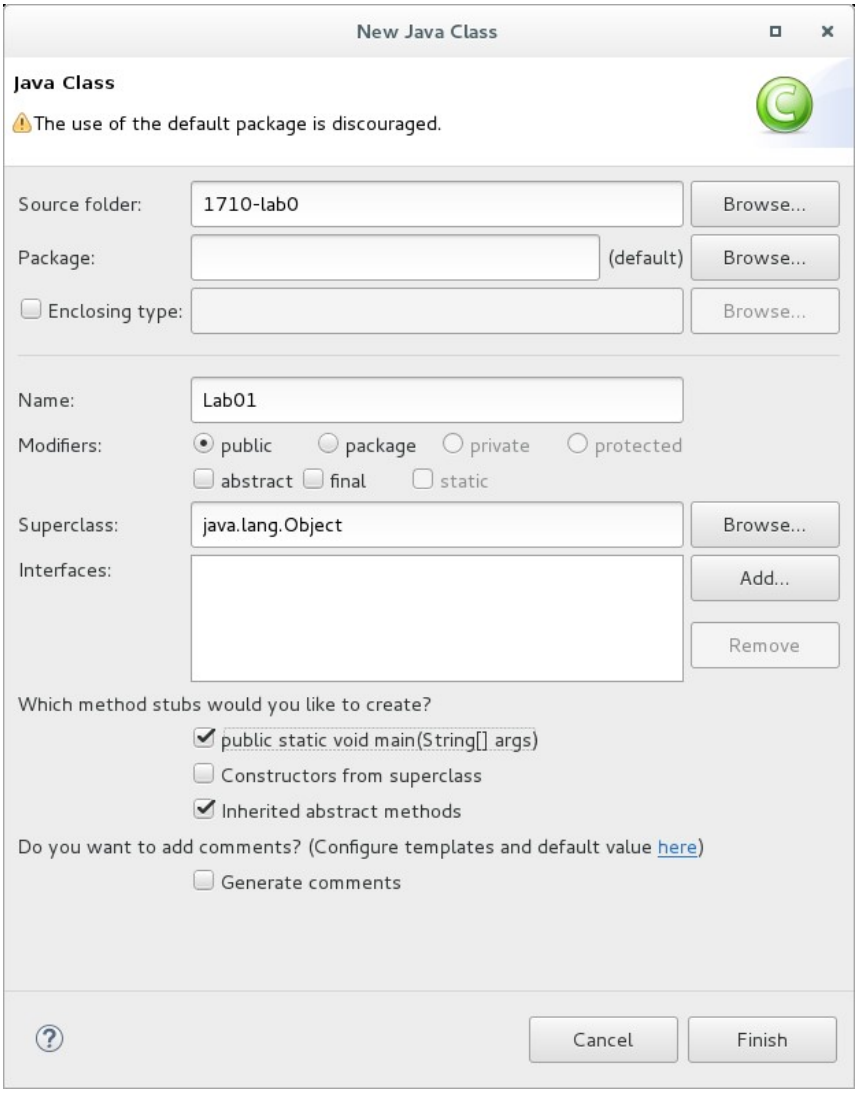
## 4.3. "Hello World"

Within Eclipse, right-click the project name ("`1710-lab01`") in the Package Explorer and select **New** -> **Class**:




In the resulting dialog box, enter `Lab01` in the

**Name**
field
and
select
the
checkbox
where
Eclipse
asks
if
you'd
like
to
include
a
`main`
method:




Make
sure
you
capitalize
the
first

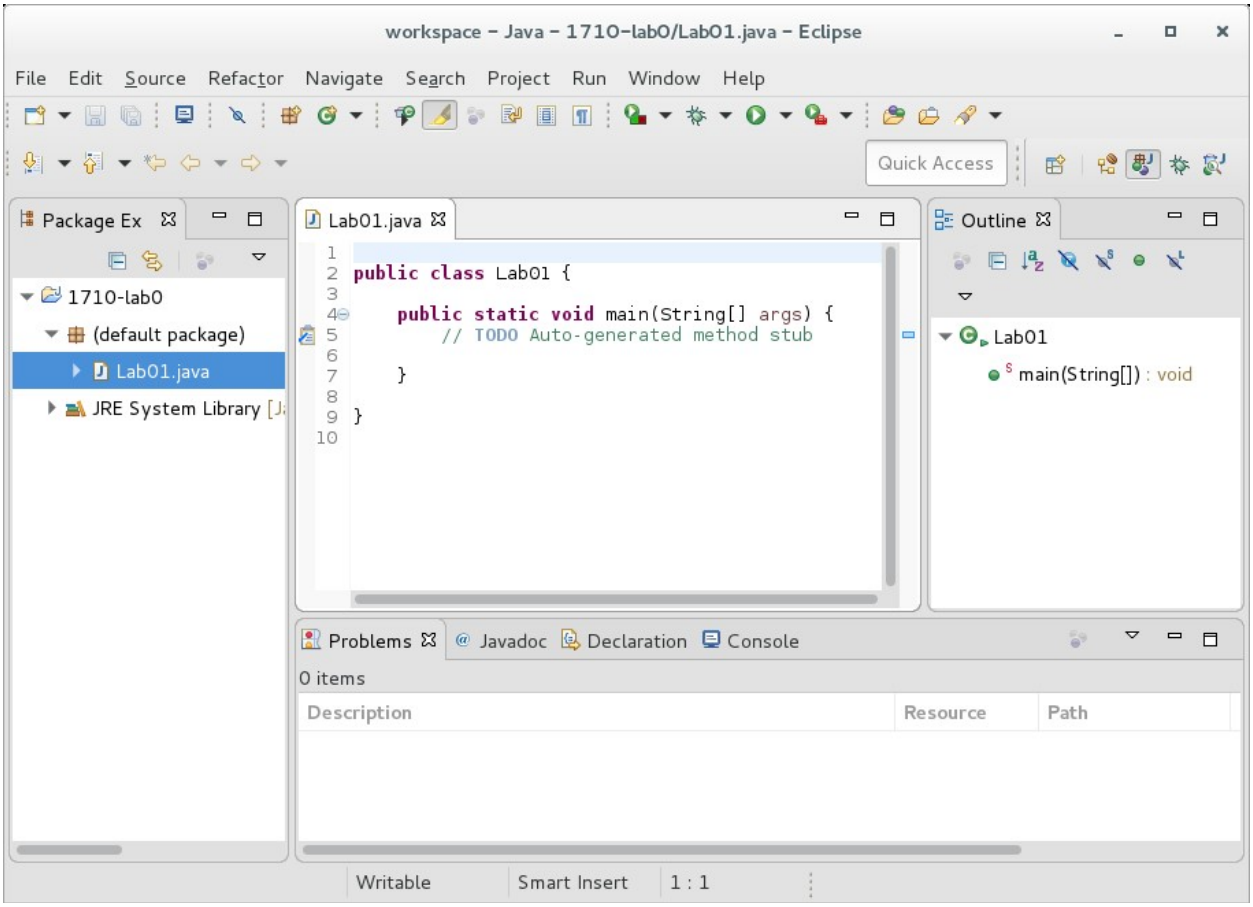
letter in `Lab01`. In Java coding conventions, the names of classes begin with a capital letter. Thus, you should be careful to capitalize the initial letter of any new class you create.

Click "Finish" to create the new class.

Since the `Lab01` class has a `main` method, it is executable by the Java virtual machine (VM). In other words, `Lab01`

is a *Java application*, also called an *app*.

Next you will see this:




Let's have a look in the body of the `main` method. You will see a line like

this:

```
//
TODO
Auto-
generated
method
stub
```

You
can
delete
the
entire
line.
The
body
of
the
method
will
be
empty
and
there
should
only
be
blank
lines
between
the
`{`
and
the
`}`
that
immediate
following
the
`main`
method.

Now
cut
and
paste
the
following
source
code
lines
into
the
body
of
the
main
method.
Use
Ctrl-C
and
Ctrl-V

for copy and paste between applications.

```
System.out.println("******************");
System.out.print("Hello");
System.out.print("
");
System.out.print("World");
System.out.println();
System.out.println("******************");
```

Save the file: **File** > **Save**. Of course, you can use the standard shortcut for "file save" (Ctrl-S).

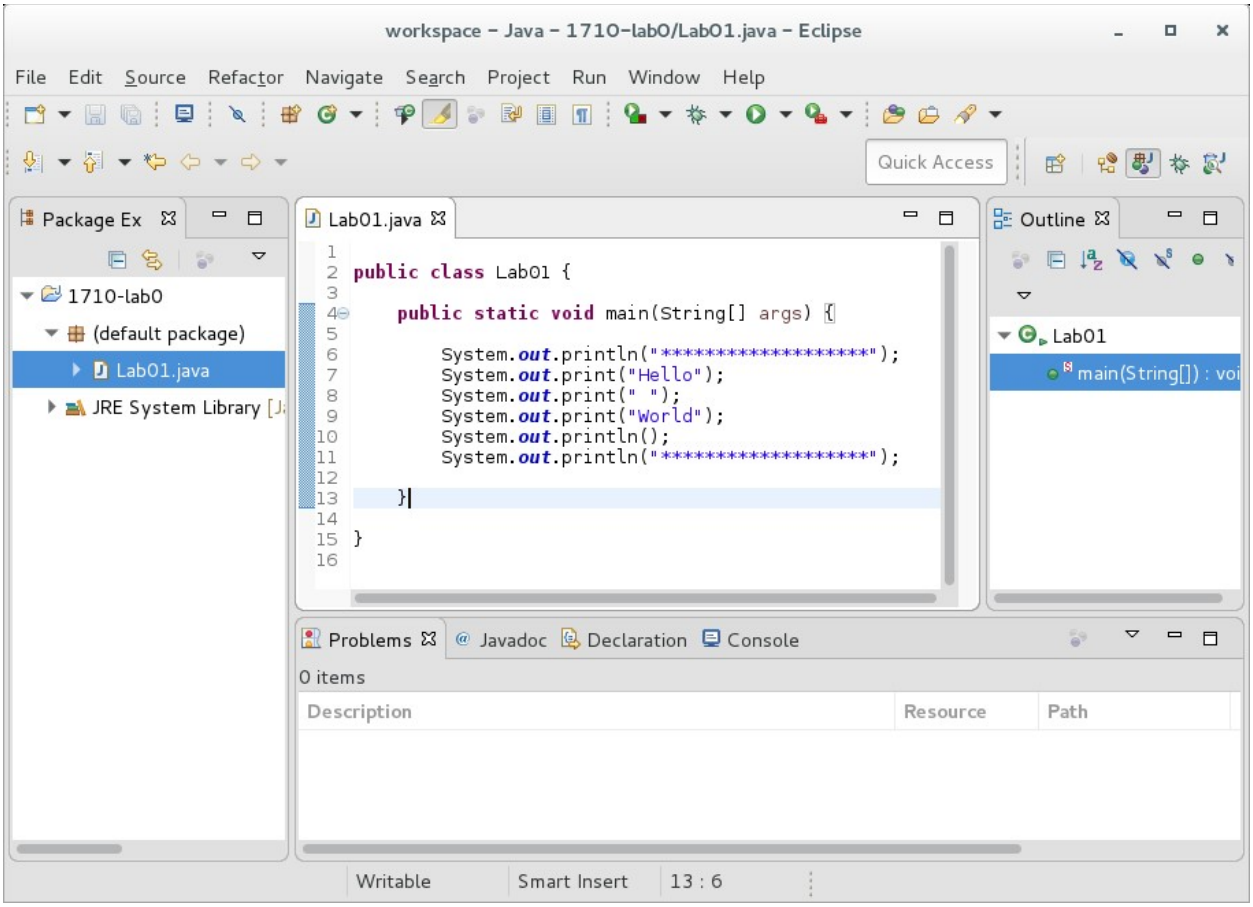
You may not notice it, but Eclipse is automatically compiling the program at repeated intervals in the background. A file called `Lab01.class` has been created (although the Package

Explorer will not show it, in order to reduce clutter). You can examine it using the File Explorer or the Command Prompt.

Almost certainly the code is not formatted according to coding conventions. With Eclipse, you can automatically reformat source code using **Source > Format**. You will use this command frequently, so learn the shortcut, which is Ctrl-Shift-F. This step is

also described in the exercise "Formatting Your Code".

The contents of the editor window should appear as shown below. By now, you should understand that the body of the main method is the portion between the inner "`{`" and "`}`" (curly braces).



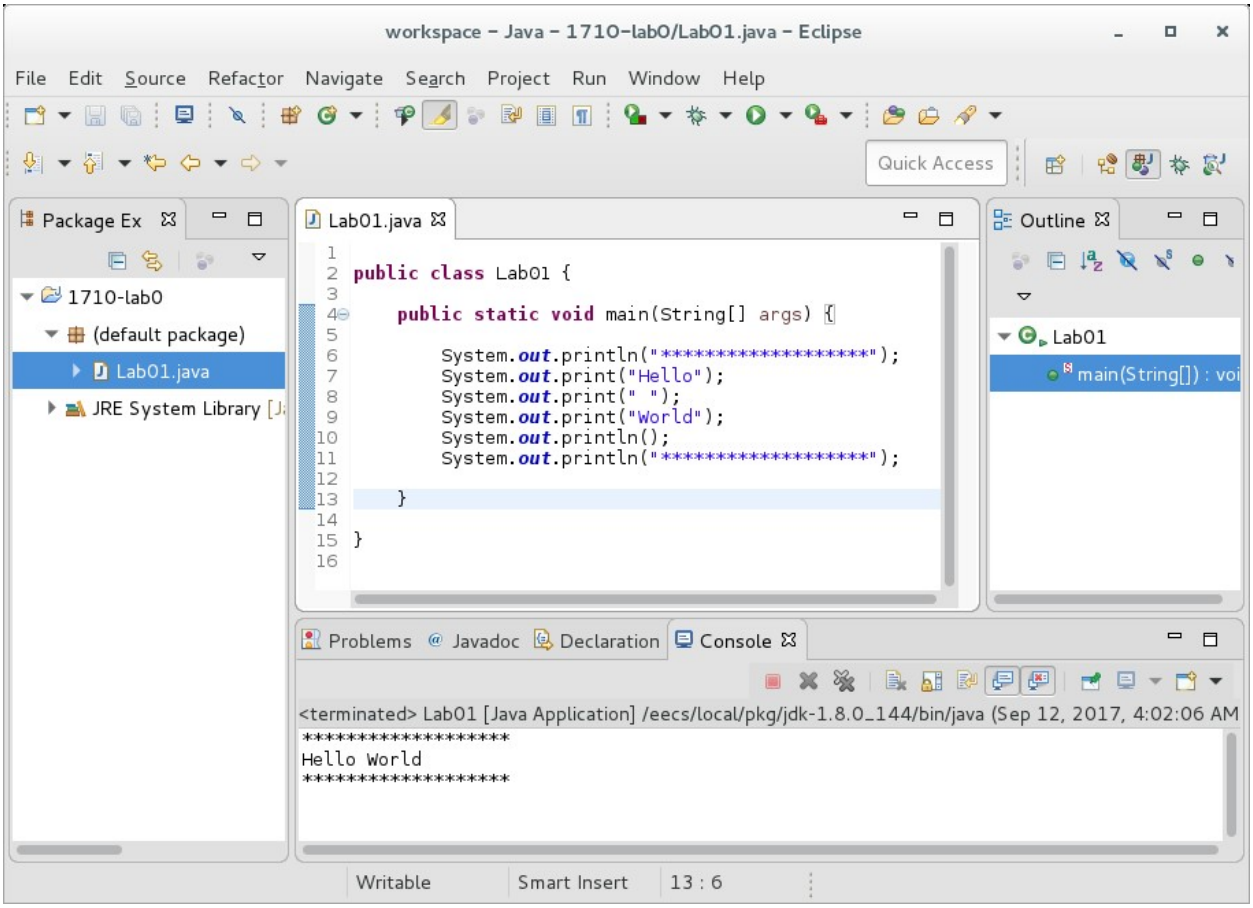

Now, let's run the program. Ensure that `Lab01.java` is selected in the Package Explorer view (click on it). Then select **Run** > **Run**. Alternatively, you can click on the icon showing

a green circle enclosing a right-arrow:

An even faster shortcut is to use the keyboard shortcut, which is Ctrl-F11.

When the program is run, a couple of things will happen. First, in the bottom view, a fourth tabbed pane called "Console" will be highlighted. The output of your app is sent to the console. It

should look like this:




Congratulations!!! You have just written and run your first Java application.

A concluding comment: The best way to learn is **TO BE CURIOUS** and **TO**

**EXPERIMENT**. For instance, you can change any of the statements in the program and examine what happens. Try this: Modify the source code so that "My name is [your name]" appears after the hello-world message.

Also, notice that there are both `println` and `print` statements. Can you identify what each statement does?