Installing and Using Eclipse

The official Eclipse FAQs are available from www.eclipse.org.

1. Installing Eclipse on my computer

1. Verify JDK installed on your computer *before* you start.
2. Download **"Eclipse IDE for Java Developers (150MB)"** from http://www.eclipse.org/downloads/index.php into the Applications folder. Select **32 Bit** or **64 Bit** depending on your **Operating System type.** The downloaded file will have a name like this: `eclipse-java-juno-SR1-win32-x86_64.zip` (if you are on a PC) `or eclipse-java-juno-SR1-macosx-cocoa-x86_64.tar.gz` (if you are on a Mac). It is a zipped (packed) file so unpack it by clicking on it. As it is unpacked, it will create a folder named **`eclipse`**.
3. In that folder, you will see **`Eclipse.exe or Eclipse.app`** which is what you want. You will see a bunch of other things in that folder.
4. You can copy the folder to C: \ and create a shortcut on your desktop or to the dock if you want easy access every time you use it.

2. Starting Eclipse

1. Start it by clicking on `Eclipse.app` or on the icon on the dock. It will ask for a *workspace*. Create a directory where you want to keep the current Eclipse project that you want to work on, e.g., `ICS4U`.
2. When you first start your new Eclipse, you will see the `"Welcome to the Eclipse IDE for Java Developers"` page. Try `"Tutorials"`. You will see **`"Create a Hello World application"`** there. It will guide you through building a project containing `HelloWorld` class. Try to follow the tutorial step by step. If there is anything that is not clear, the remainder of this document might be helpful. *The remainder of this page will be helpful when you develop a new project later rather than following the tutorial.*
3. If you are not following the tutorial, you can start with the "Workbench".

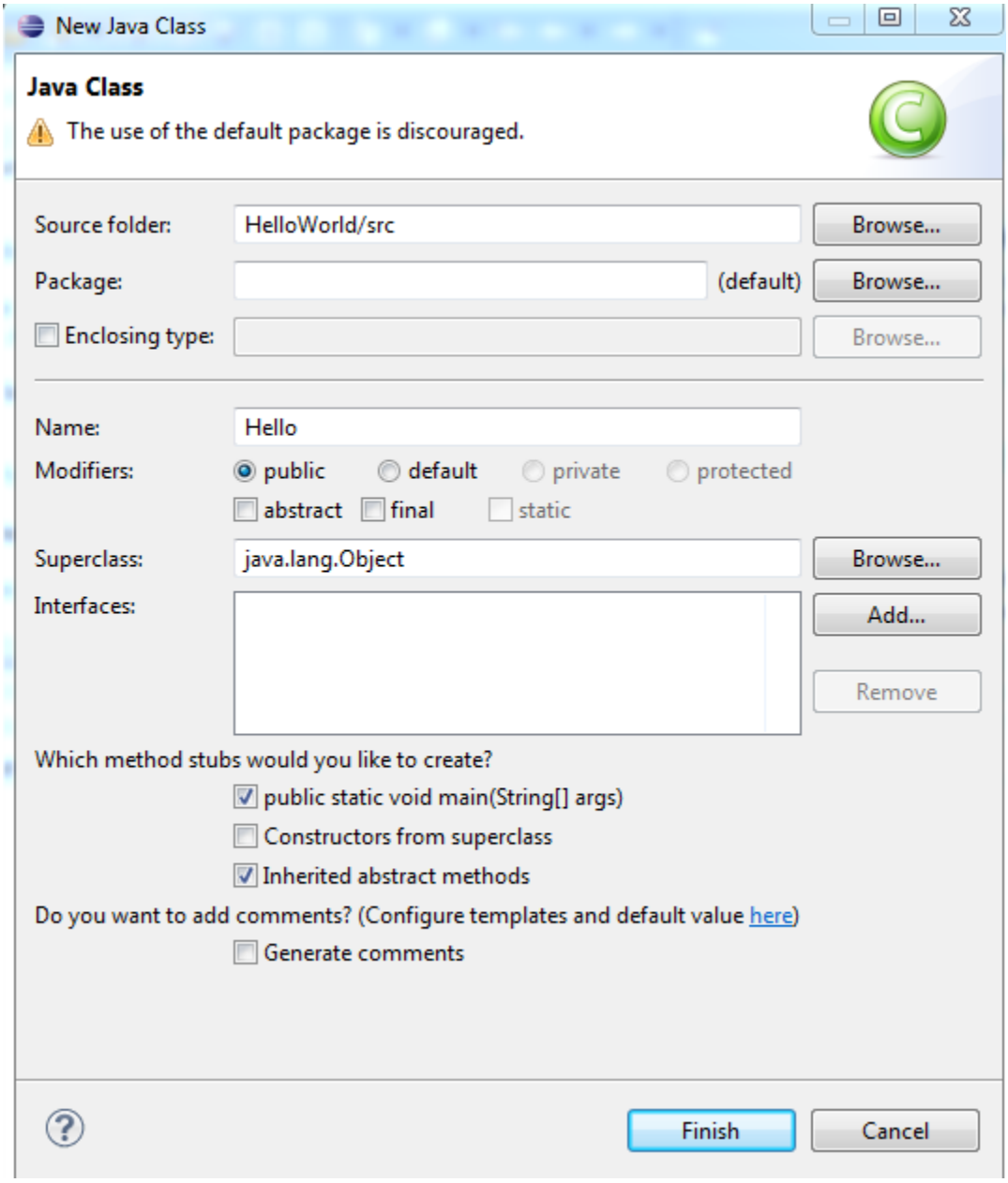
3. Creating a new project in Eclipse

To run a Java program using Eclipse, you must create an empty project and start adding one or more Java files. You can add a Java file either by creating a new Java file using Eclipse's editor or by importing an already existing Java file.

1. Open the `"File"` Menu. Select `"New"` and then `"Java Project"` from the submenu.
2. Enter the Project name, e.g., `HelloWorld`; select a JRE environment, e.g., JavaSE-1.7; and take the default selections, and `"Finish"` it.

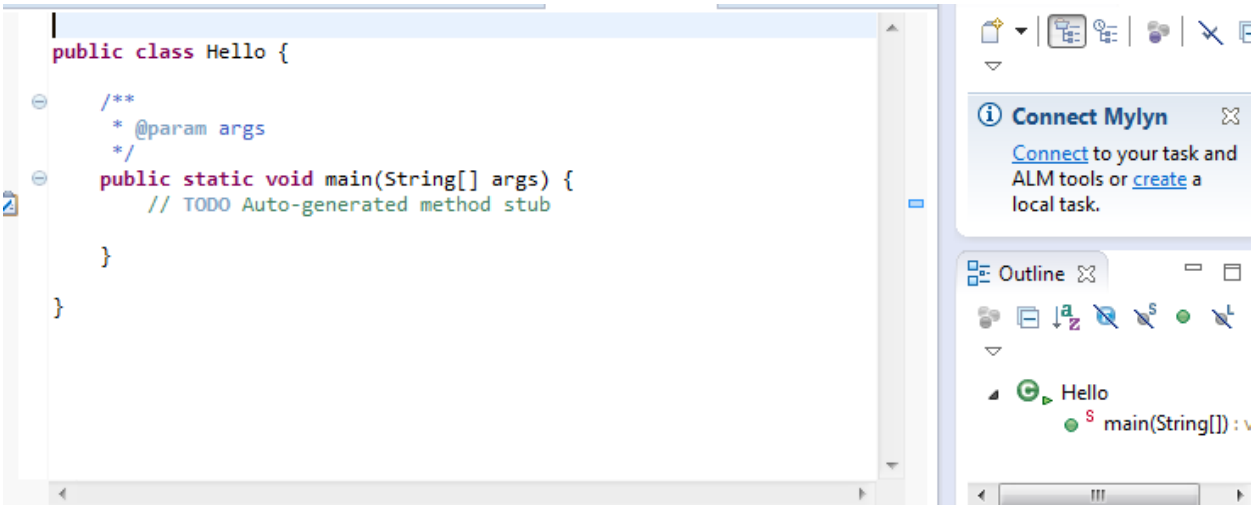
4. Adding a class to a project

First, you should make sure the project you wish to add the class to is selected in the `"Package Explorer"` if there are more than one project created in your session. Then right click on the project, then select `"New"` and then `"Class"` or `Interface"` as opposed to `"Class"` if that is what you want to create, from the submenu.



When a new dialog box appears, make sure that the name of the project is listed as the `"Source folder"`. If not, use the browse button to find the correct project. In the `"Name"` field, type the name of the new class that you want to create, e.g., `HelloWorld`. While you can add information in the `"Superclass"` and `"Interfaces"` fields, it is often simpler just to add them yourself using the editor after the class has been created. You most likely want to include the skeleton for a `main` method so check the box for **public static void main(String[] args)**. Click on `"Finish"` to finish creating the new class as part of your project.

At this point, you will see a template of a class that you can complete.



replace

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

with

```
        System.out.println("Hello world - Here I come");
```

`"Save"` it using the `Save` under the `File` menu when you are done editing your class definition. Now, run it as described in the next section.

5. Running a Java application

Now you can run the saved file from the previous section (assuming you have a `main` method defined in the class) by `"Run As"` `"Java Application"` under the `"Run"` menu or simply by the `Run` menu under `Run`. If there is any error, it will complain. If it runs successfully, it will generate output in the console window at the bottom part of the screen.

6. Adding an image, sound, or Java file to a project

You can import any type of file into an Eclipse project. It can be an image, sound, or Java file. The files will be copied into the project. If you want to use a different editor (e.g., emacs) to write your program and import the file into an Eclipse project, this procedure will be useful. Suppose we want to import `Var.java` from lecture 3 into the current project named `test` which contains one Java file, say, `Hello.java`.

1. Select the project (i.e., `test` in our case) in the `"Package Explorer"` panel. Select the `(default package)` folder which currently has `Hello.java` in it. When we finish importing `Var.java`, you will see `Var.java` in the same folder where `Hello.java` is.
2. Open the `"File"` menu. Select `"Import"`. Select `"General"` followed by `"File system"`. Click `"Next"`.

3. Click the "Browse" button. Find the folder containing the files to import. Select "Open".
4. Click on the folder name (not the check box) in the left column. (If you click on the check box, it will include all the files in the folder. Try it with a folder containing multiple files.)
5. Select the files to import in the right column.
6. Click "Finish".
7. When you have multiple files in a package, double-click on the file that you want to work on.

7. Deleting a file from a project

Select a file in a project with a right-click and try "delete". Note that if you delete a Java file from the project, the file will be deleted from the project directory and it will be gone, gone, gone...!!! If you want to save the file somewhere before you remove it from a project, you will have to make a copy manually yourself before you delete it from the project.

8. Messing up an Eclispe Project

Do not manually delete any file from an Eclipse project. If you want to delete any file from an Eclispe project, use a menu in Eclipse to do it. Do not add any file manually into an Eclipse project either. Use one of its menus to do it. Once you start modifying an Eclipse project manually without going through its interface, i.e., its menus, you can't expect your Eclipse to behave nicely.