Introduction to the NetBeans IDE
Mock

This document is a brief introduction to writing and compiling a program using the NetBeans Integrated Development Environment (IDE). An IDE is a program that automates and makes easier many tasks that programmers would otherwise have to perform themselves. While many IDEs exist for Java, we will focus only on the NetBeans IDE because at the time of this writing it is free and contains many powerful features.

Installation

The CS Lab already has NetBeans installed. If you plan to work exclusively from the lab, then you can skip this section. The machines in the classroom also have NetBeans installed but they don't have as much memory or processing power as the machines in the CS lab, so NetBeans will run slow, especially upon startup.

If you plan to install NetBeans on your own machine, your system should meet these recommended specifications:

Disk space: 1 GB free space
Memory: 1 GB RAM
Processor: 1.4 Ghz Pentium III processor or better
Operating Systems: Mac OS X, Windows, Linux

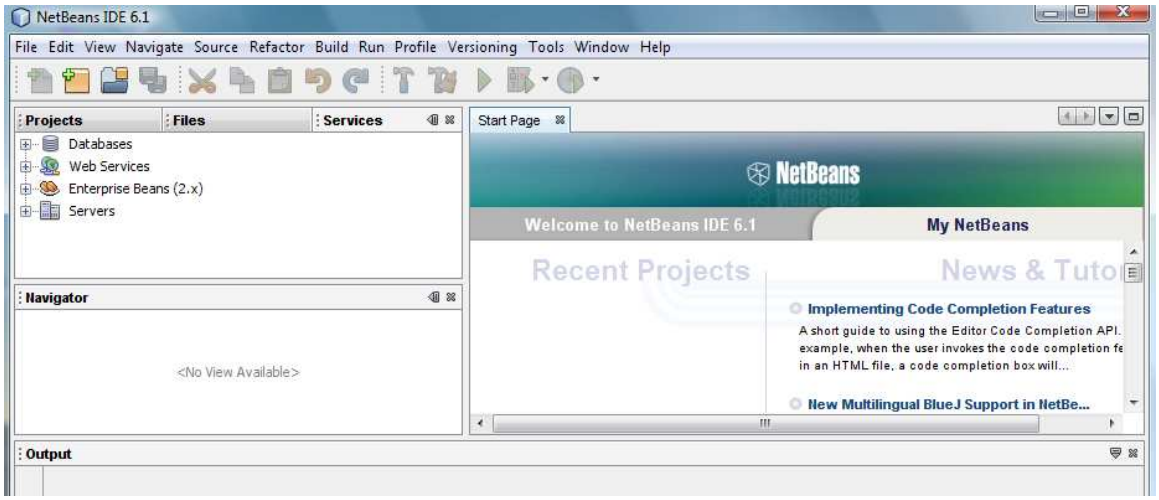
You can download NetBeans from http://www.netbeans.org. There are numerous download options, e.g. including web development, database, etc. The "Java SE" version is sufficient for purposes of this class.

The NetBeans IDE is a big file --- a minimum of around 30 MB. After you have downloaded the file, simply execute the file to install the software.

Starting NetBeans

In the following screens I am assuming that you are running Windows and NetBeans 6.1. Some details are different if using a Mac or Linux machine but the overall process is the same. After you have installed NetBeans, double-click on the icon to get started:

NetBeans will take a moment to start up – if you have previously installed an IDE that NetBeans recognizes then it will give you an option to import old settings. After NetBeans has started it should bring up the main window that looks something like this:



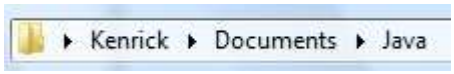
Depending on the version of NetBeans, yours probably won't look exactly the same but should be similar.

Feel free to click and read some of the "News and Tutorial" information located in the right hand pane.

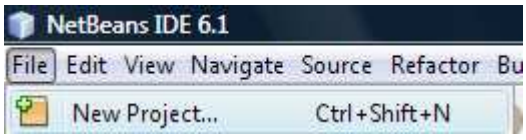
Building a Sample Project – Hello World

Let's start by creating in NetBeans the sample program we covered previously that printed out "Hello, world". The process will be similar when working on your own programs.

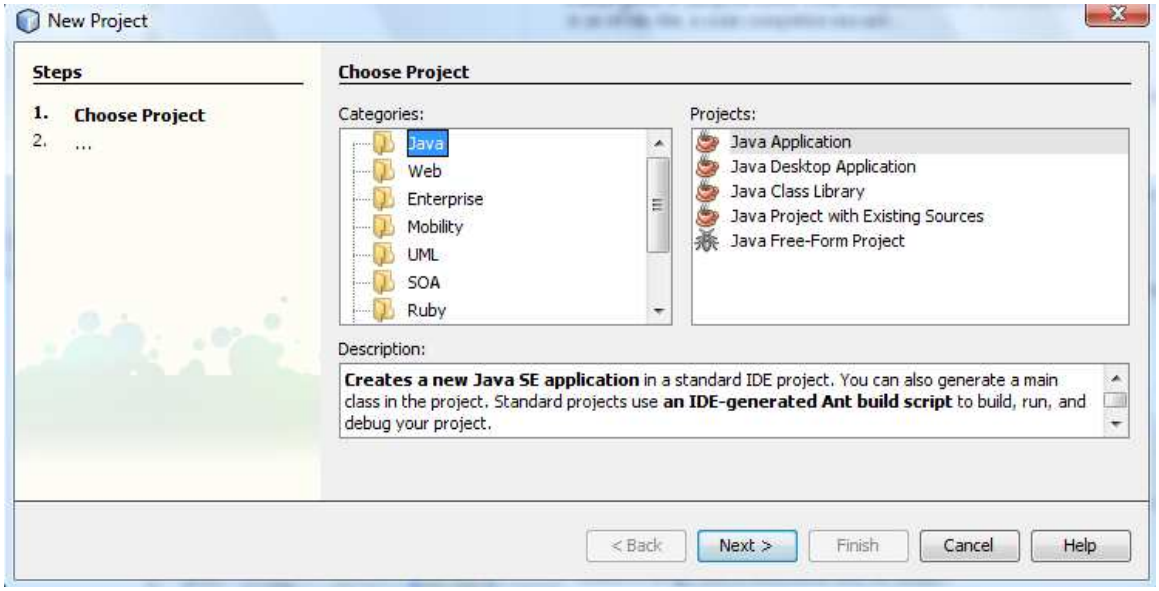
First, let's make a folder to store our Java files. In this case I'll create a folder called "Java" in my "Documents" folder but you could make the folder anywhere you have access. Go back to Windows, open up a File Explorer or My Computer, and right click in your selected folder to create a folder named Java:



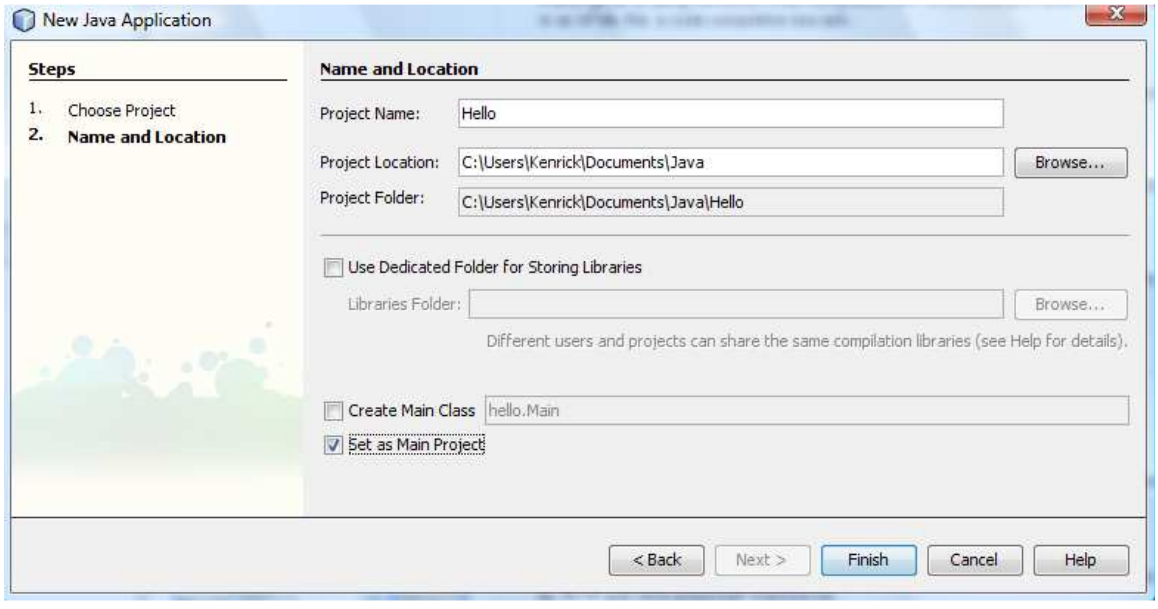
Back in NetBeans, from the main menu select F)ile and New Project:



Choose “General” for the category and “Java Application” for the project and click Next:



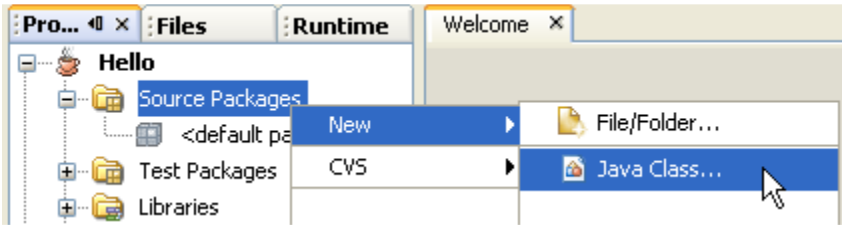
On the next screen NetBeans will ask what you would like to name your project and where you would like to create it. Give it the name of “Hello” and select the Java directory created in My Documents as the location. For now, uncheck the box “Create Main Class” (it’s OK to leave it checked; if so, NetBeans will generate some initial code for you):



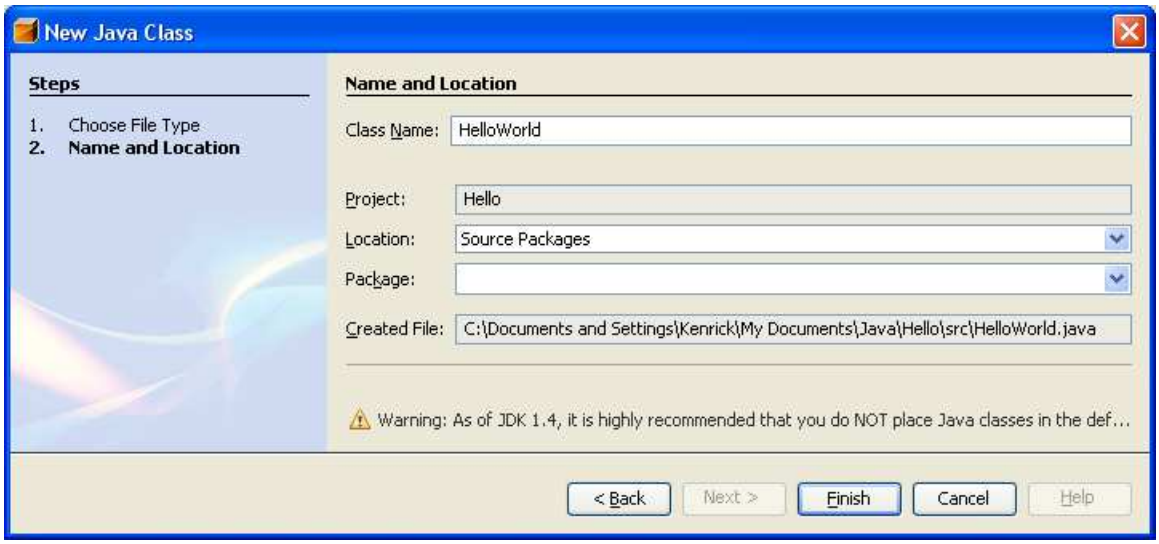
Normally you would be giving this a project name indicative of your homework, e.g. “Homework1” or such.

Click on Finish and NetBeans will create your project.

In the upper left corner of the screen is a Projects tab. The "Hello" project we just created should now be listed as a project. Open it and under Source Packages right-click and create new Java Class:



Give "HelloWorld" for the class name and click on Finish.



This will create a new Java class called "HelloWorld" and put it in the default package. You might notice from the warning that it is not recommended to put code in the default package. A package is just a way to separate different libraries of code. We'll see later how to create our own packages and put our code there instead.

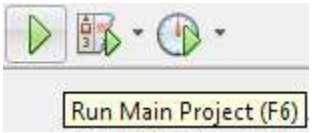
NetBeans will generate a default file that looks like this:

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

/**
 *
 * @author Kenrick
 */
public class HelloWorld {

}
```

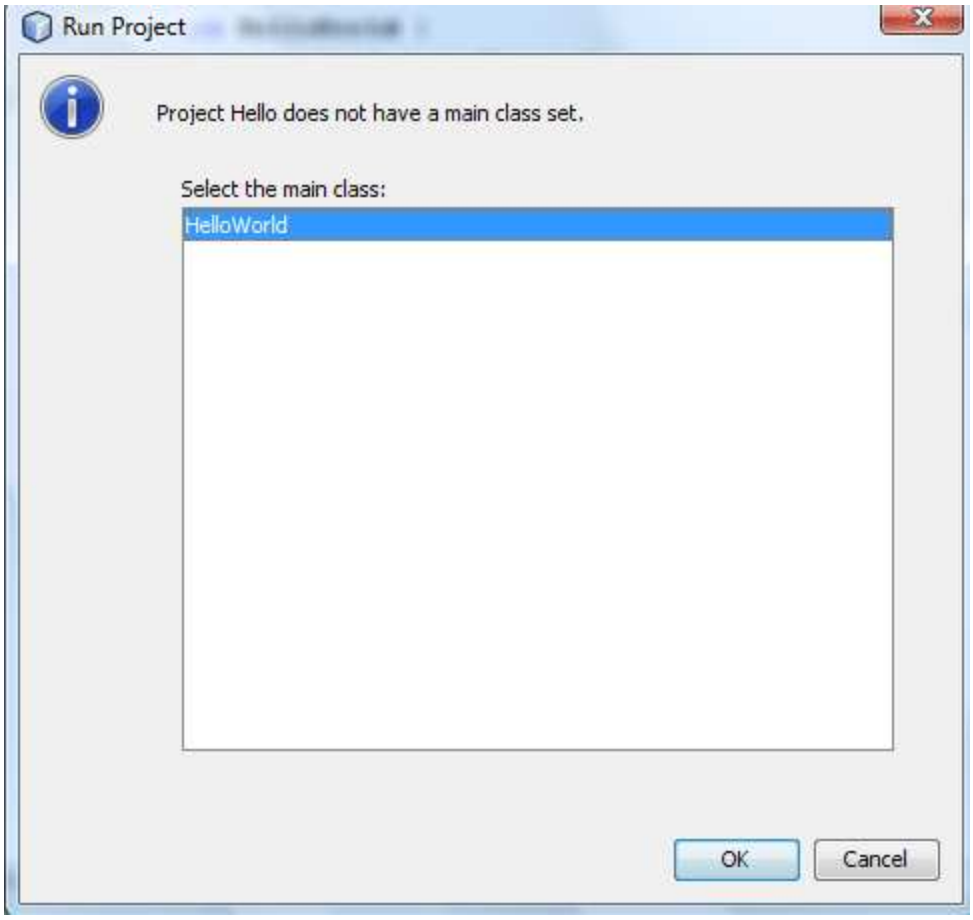
We can now start typing in the code from our Hello World example and inserting it into the document:

```
import java.util.Date;
/*
 * HelloWorld sample program in NetBeans.
 */

/**
 *
 * @author Kenrick
 */
public class HelloWorld {
    public static void main(String[] args)
    {
        System.out.println("Hello, world!");
        Date today = new Date();
        System.out.println("Today is " + today.toString());
    }
}
```

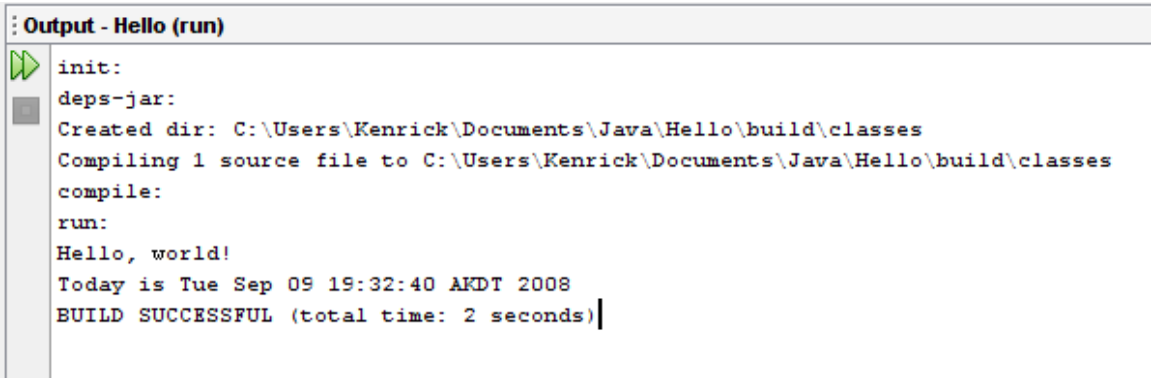
To compile and run your program click on the green triangle "Play" button:



On the next screen NetBeans will ask which is the main class. Select your HelloWorld class since it is the one with "main" defined in it. You only need to do this once per project:



Your program will now run! It will give output in the Output window at the bottom of the screen:



Congratulations, you have just created your first project in NetBeans! You should follow a similar process when working on programs assigned as homework.

If you are going to turn in your project then the easiest way is to locate the project folder (in this example, "Hello" located in "Documents/Java") and then compress the entire folder into a zip file. You can then send the zip file. Note that in the future some instructors may only want your java source files (.java located in the src subdirectory).

Sending the entire project folder includes NetBeans gunk that some may not care for if they don't use NetBeans.

For those that do have NetBeans though, compressing the whole folder is much easier because it includes the NetBeans project and other meta-data used to create the project.

If you want to work on the project at a later date, simply restart NetBeans and all of your files should be visible. If you start to work on many files you may wish to delete some of the .class files or copy out old folders that you no longer use. In particular, the .class files can take up a lot of space.

We have only touched on the basics of using NetBeans here; feel free to explore on your own the many other options that are available. In particular, you may notice that NetBeans will detect many errors as you are typing them. This can be quite helpful in avoiding many common problems.

Later we will look at using NetBeans debugging tools which are quite helpful in tracking down mysterious runtime errors.

NetBeans vs. pico or TextPad?

As you can see NetBeans offers a lot more features than writing your programs in pico or textpad and directly compiling them. However, it also has a lot more baggage if you only want to edit, compile, and run a short program. For small programs I tend to prefer the direct editor/compile approach. However for larger programs with many objects and files an IDE like NetBeans is the superior way to go.