Lab 1 – Introduction to Java Programming in Netbeans

Aim

The aim of this lab is to learn the very basics of Java and NetBeans. At the end of the lab, you should be able to open NetBeans, create a project, and modify simple Java programs to display information.

Resources

You will find your lecture notes useful, as well as "showsquares.java", which can be downloaded from ICE.

Tips:

1. If you are not sure why you are doing something, ask a TA. This is what they are here for.

2. These labs are expected take more than the 2 allocated hours. You should complete them in your own time before the next classes. Practice makes perfect!

3. Talk to each other. Talk to your classmates, it is encouraged!

4. Do not save on the C: D: E: folders. You should always save on your network folder. This will ensure your work is not deleted.

Folder Management

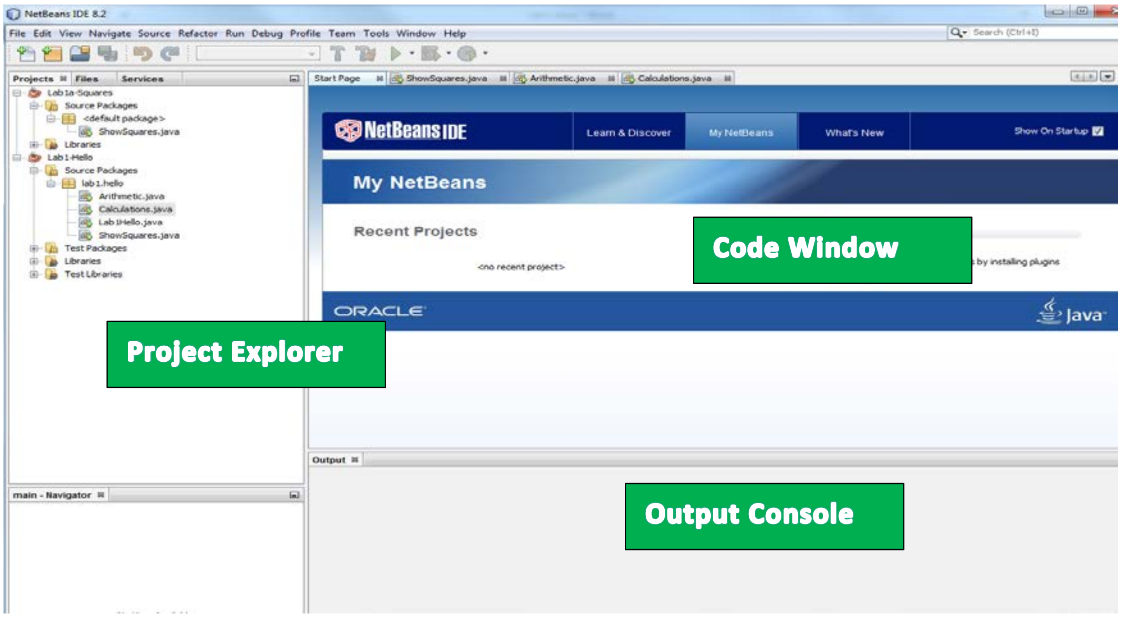
- Navigate to your network folder. If you are not sure where this is, ask.

- It is important to manage your folders correctly. You should create a CSE105 folder, and then a folder for this week's work. All your work in this lab should be saved within this folder.

Programming with NetBeans

- Open NetBeans, which should be in your start menu. Look for this icon: NetBeans IDE 8.2

- When open you should see several panels:



- The top left panel is the **project explorer**. This is where the java files and projects you create will be displayed. They can be opened by clicking on them.

- The **code window** is where you will type your code. All the java files that you are working on can be edited in this window.

- At the bottom is the **output console**, which will display all your program output. If you give your program instructions, any written output will be displayed here.

A First Java Project

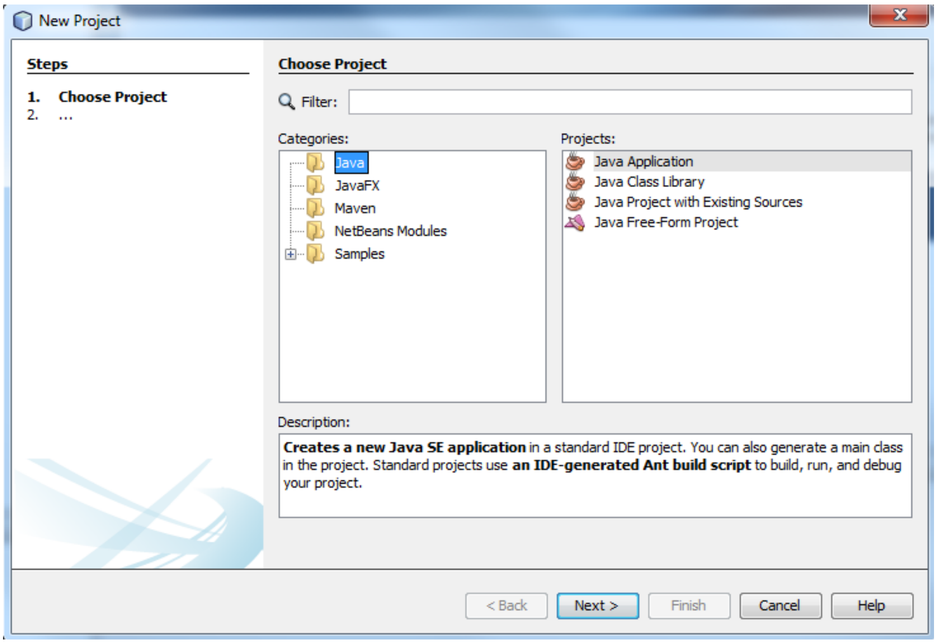
- Create a new NetBeans project by navigating to “File” in the menu at the top of the screen, and then “New Project”.

- Choose to create a new “Java Application”

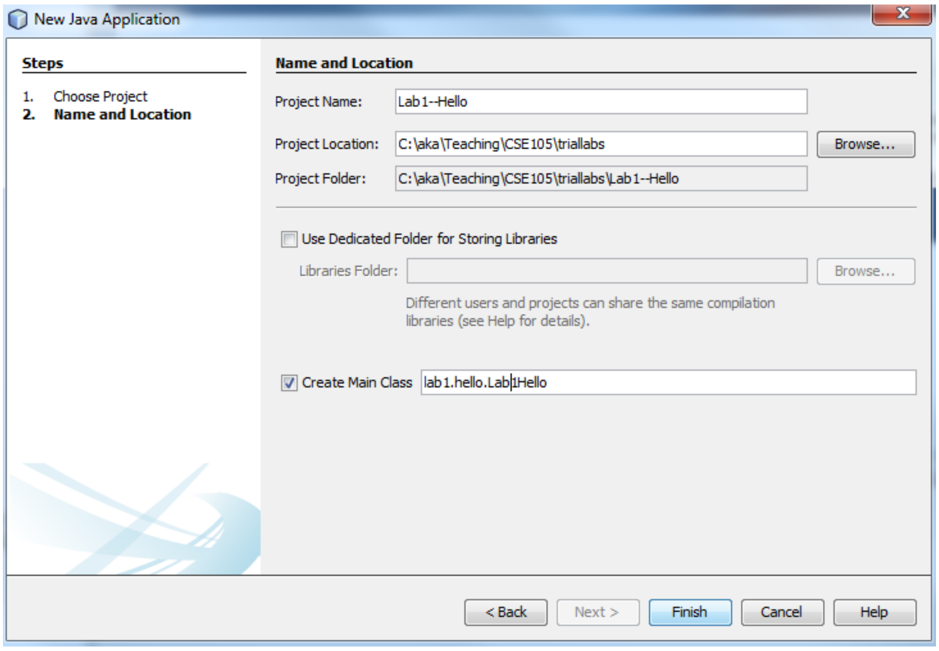
- Call your project “lab1-Hello”. Make sure to set the folder location to the folder you created previously.

- Make sure the “Create New Class” box is ticked. See the screenshots below.

- Press Finish to create your new project.



Creates a new Java SE application in a standard IDE project. You can also generate a main class in the project. Standard projects use **an IDE-generated Ant build script** to build, run, and debug your project.



New Java Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name: Lab1--Hello

Project Location: C:\aka\Teaching\CSE105\triallabs

Project Folder: C:\aka\Teaching\CSE105\triallabs\Lab1--Hello

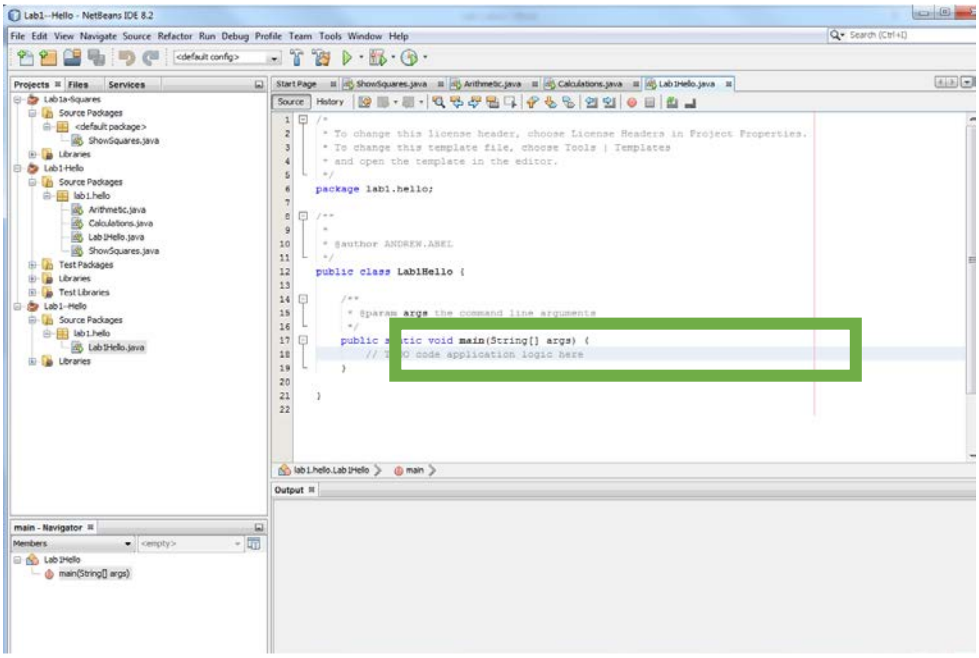
□ Use Dedicated Folder for Storing Libraries

Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

☑ Create Main Class lab1.hello.Lab1Hello

- As you chose to create a new class, the project will automatically create a Java file, which has a main method (This will be discussed in future weeks). Everything you need to do this week is within the main method.



```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package lab1.hello;

/**
 *
 * @author ANDREW.ABEL
 */
public class Lab1Hello {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
    }

}
```

- In the main method, add a System.out, which is an instruction to output text to the **console window**. Copy the line below to your main method.

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

- Run your program by clicking on the green triangle at the top of the screen, or pressing “F6”.

- Look at the console window at the bottom of the screen, check that you can see the message. Ask if you can’t see it!

- Change the System.out commands from System.out.print, to System.out.println, i.e.:
  `System.out.println(“Hello World”);`

- What is the difference? Do you understand it? If you do not, ask!

- Make some modifications to this System.out. Make it say hello to you (add your name). Add another line, very similar, and make the program say hello to the person sitting next to you!

- Add several more System.out lines to say something interesting about yourself. Add a line to give an interesting fact about yourself.

- Add a line to give your school, name, and hometown.

- Add a line to give your name, age, and student number.

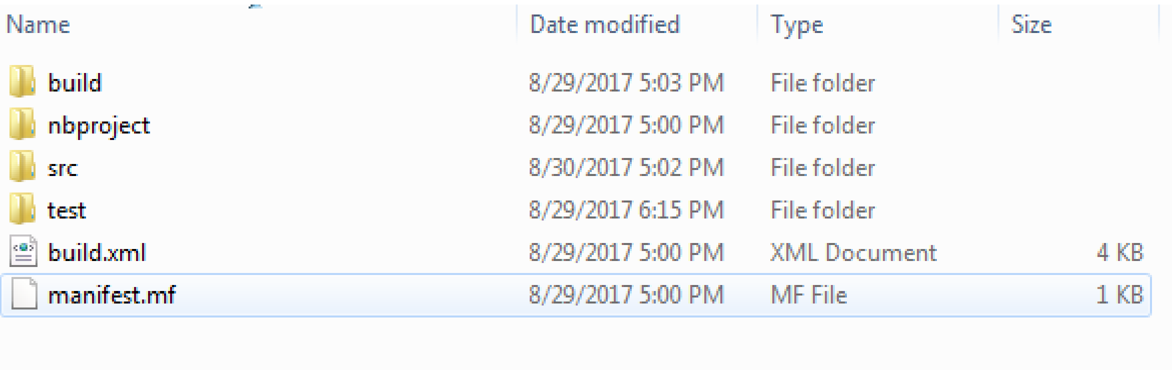
- Add a line to give your student number and major.

- The console output should look something like:

```
Hello Andrew

Hello Joe

My name is Andrew, and I'm a lecturer in CSSE

My name is Andrew, my school was Montrose, and my hometown is
Arbroath.

Name: Andrew, Age: 32, Student number: 12345678
```

Class and Java Files

The files you have worked on have been Java files, i.e. hello.java. As discussed in the lecture, these are not the files that the computer runs. Navigate to the folder that was created for this project. You should see something like:



Name	Date modified	Type	Size
build	8/29/2017 5:03 PM	File folder	
nbproject	8/29/2017 5:00 PM	File folder	
src	8/30/2017 5:02 PM	File folder	
test	8/29/2017 6:15 PM	File folder	
build.xml	8/29/2017 5:00 PM	XML Document	4 KB
manifest.mf	8/29/2017 5:00 PM	MF File	1 KB

- Click on the "src" folder, and then click through the different packages until you reach the java files themselves. These are the files that you edit and make changes to.

- When you compile and run the project though, they are used to create class files, which run on the Java virtual machine. To see these, navigate to the project folder again, and click on "build".

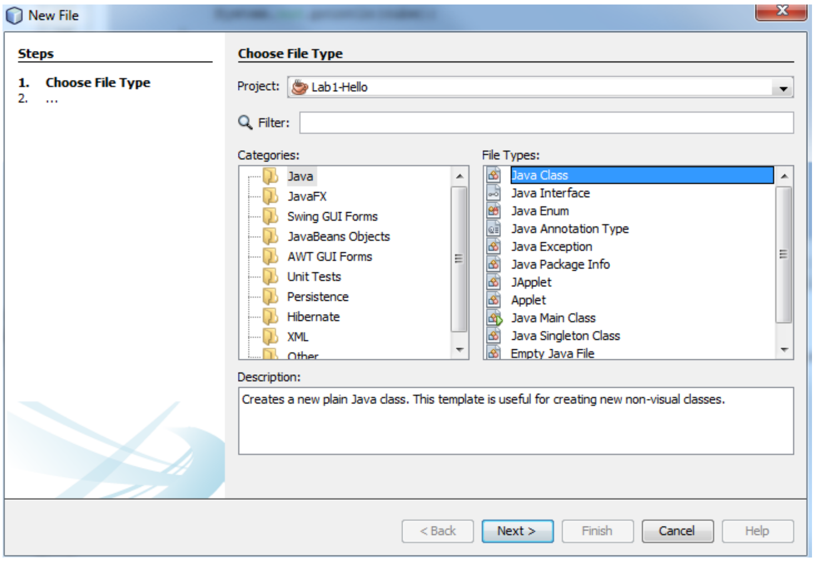
- Click on "Classes" and then click on the different package folders until you see a number of class files, with names such as hello.class.

- These are the class files that are created from your Java files, and are run on the virtual machine. It is important to understand the difference between these and the Java source files. If you are unsure, ask!

From Strings to Arithmetic

As well as displaying information, calculations can also be performed with Java. Here, we will create a new class and modify it to make use of integers, variables, calculations, and then display outputs.

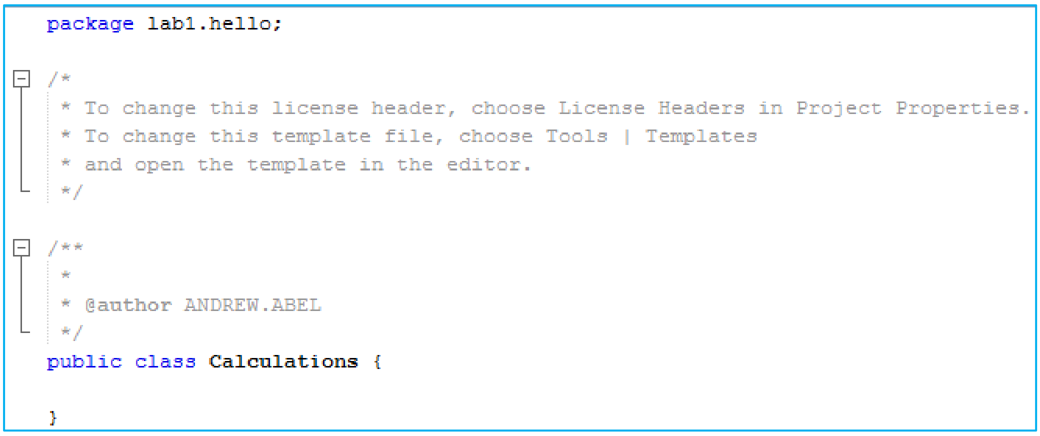
- Create a new class (File > New File), and select "Java Class"



Project: Lab1-Hello

Description: Creates a new plain Java class. This template is useful for creating new non-visual classes.

- Name it "Calculations", and press "finish". This will give you an empty class with only this text in it:



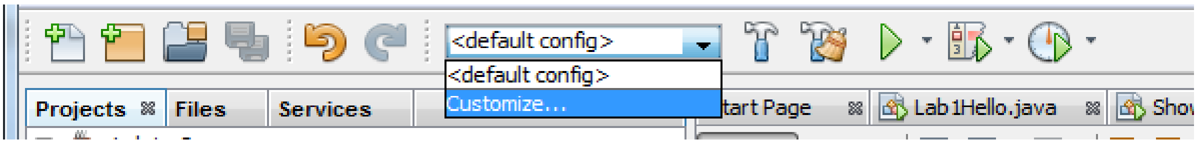
```
package lab1.hello;

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
 *
 * @author ANDREW.ABEL
 */
public class Calculations {

}
```

- You need to create a main method (discussed in future weeks). To do this, you can copy the following text into your code:

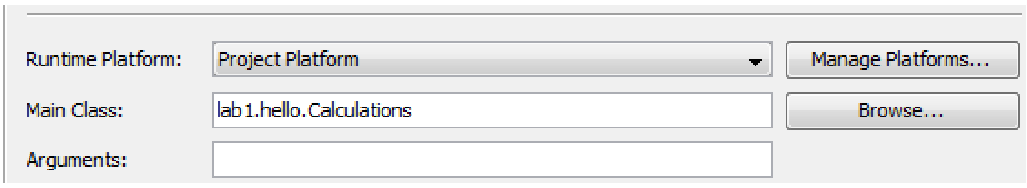
```
public static void main(String[] args) {

}
```

- To test it, add a System.out with your name in it, as you did in the previous program, and run it to make sure it works.

- You may find that when you run it, it still runs the other program. This is because your project now has two **main** classes, and it is running the other one. To solve this:
    1. Select the “customise” option in the toolbar



    2. In the window that appears, next to the “Main Class”, click the “Browse” button



Runtime Platform: Project Platform
Main Class: lab1.hello.Calculations
Arguments:

    3. Select the class that you have created. This will set it as main.
    4. Run the project to check that your message displays on the **console**

- Once the program is running, you need to change the main method. Add an extra System.out to display a calculation. Make it as simple as complicated as you like. An example is:

```
System.out.println("5 * 7 + 4 - 2 / 2 = 38");
```

- You should create your own! Make sure it displays correctly in the **console**.

- You can use integer variables to store numbers. Similar to your previous program, replace the numbers in your System.out with variables (i.e. int num1 = 5;). This means you must **declare** and **initialise** the variables, and then use them in the System.out.

- This will make it easy to change the numbers. Experiment with changing your number variables to make sure they display on screen. Does it look the same with the variables?

- You can concatenate your variables and text together with a +, i.e. `num1 +"  *  "+ num2`

Hint, you may need to use + “ “ + spaces to make it look nice!

- One thing you should have noticed is that when you change the numbers, the answer is not changed. This is because you are not doing the calculation. You need to create an additional variable that will do the calculation. For the sum above, it should look like:

```
int calculation = num1 * num2 + num3 - num4  /  num5;
```

- Remember, you should use your own calculation! Change your System.out to display the result of the calculation. Did it calculate correctly?

- Experiment by trying some different numbers. Congratulations, you have made your first calculation in Java!

Next Steps?

Finished this? Complete the homework sheet!