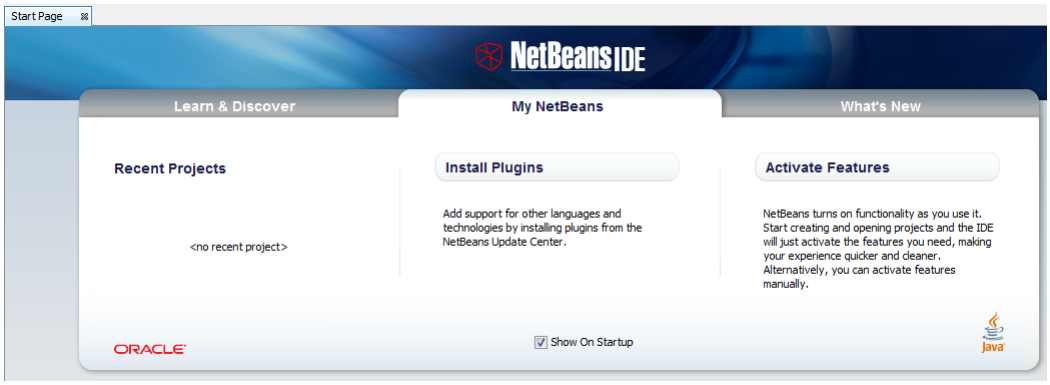
Understanding Java Code Window

Here you will learn how to create your first Java project and how to interpret the NetBeans code editor window.

If you have not yet opened the Java NetBeans Integrated Development Environment (IDE) then **double click** the NetBeans icon on your desktop and open the Java Netbeans IDE.

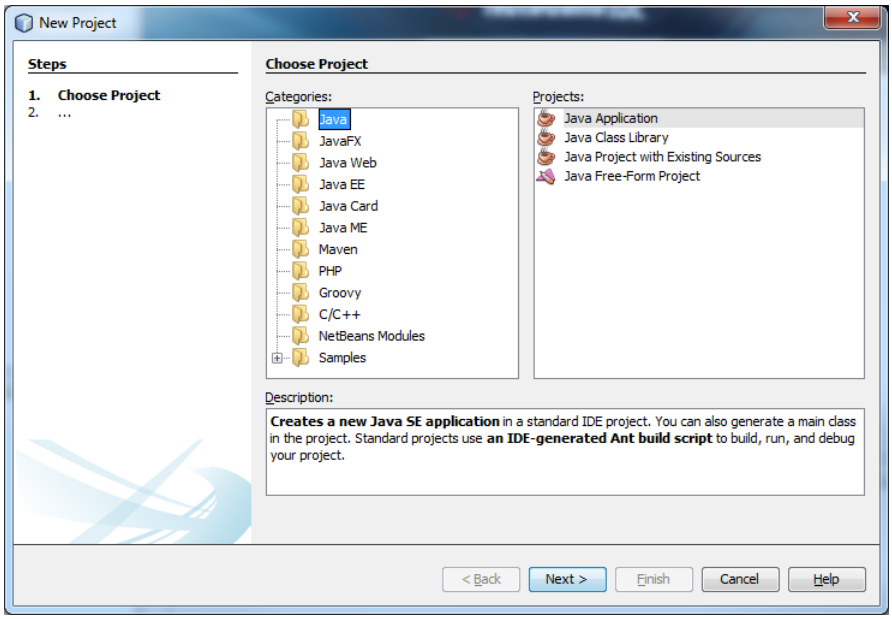
You are ready now to start creating your first project. The first time running, and unless you close the tab, you will see the following:



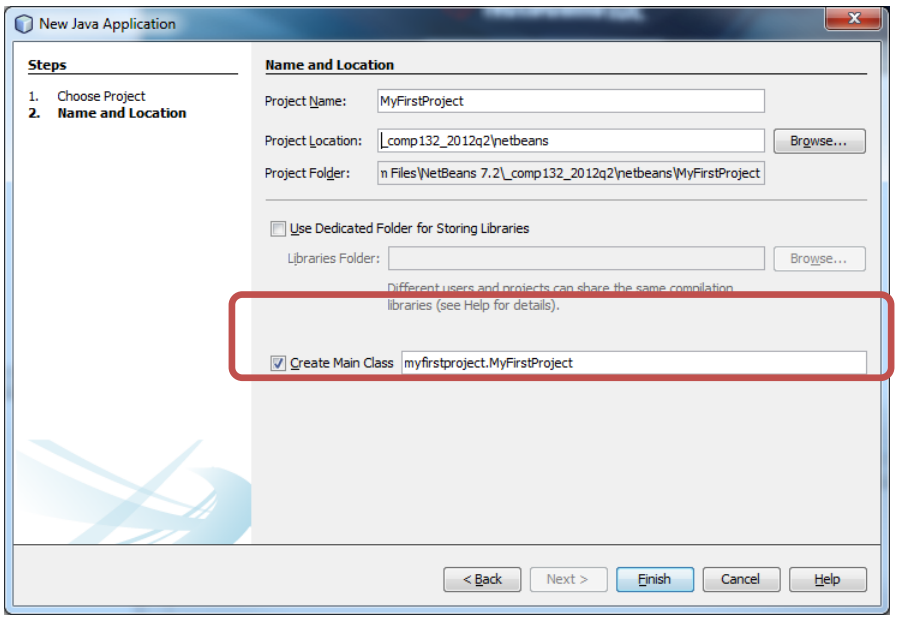
As a beginner, spare some time to familiarize with the interface and go through the provided tutorials, sample projects and demos that provide valuable information.

You can access these resources by clicking **Help** >> **Start Page** on the menu bar at the top of the screen.

To create a new project, click **File** >> **New Project** from the NetBeans menu bar and you'll get the following dialogue box. The list of categories will vary based on which version of Java/Netbeans was installed.



Click **Java** under **Categories** and **Java Application** under **Projects**, then click the **Next** button to proceed to the next step.



In the **Project Name** part at the top, type a name for the Project. You will notice that the text at the bottom changes to match the project name in the **Create Main Class** text box. Now concentrate on the red bordered part above.

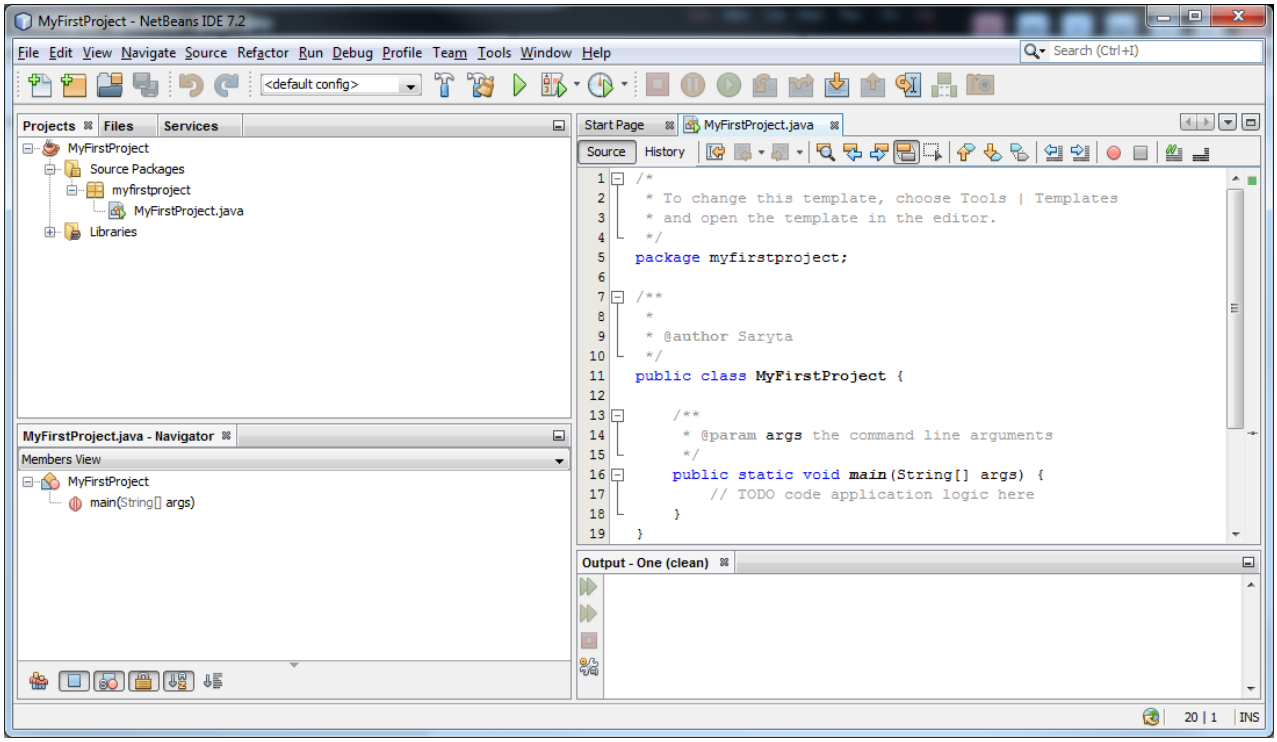
What we are doing here is that we are creating a Class called **MyFirstProject** with capital "**M**", capital "**F**", capital "**P**" which is contained in a package called **myfirstproject**, but with a lowercase "**m**", lowercase "**f**" and lower "**p**".

The **Project Location** shows where your projects will be saved and in which **Project Folder**. You can change these to suit your own preferences.

Click the **Finish** button and NetBeans will create all the necessary files for your project. Clicking finish will take you back to NetBeans IDE.

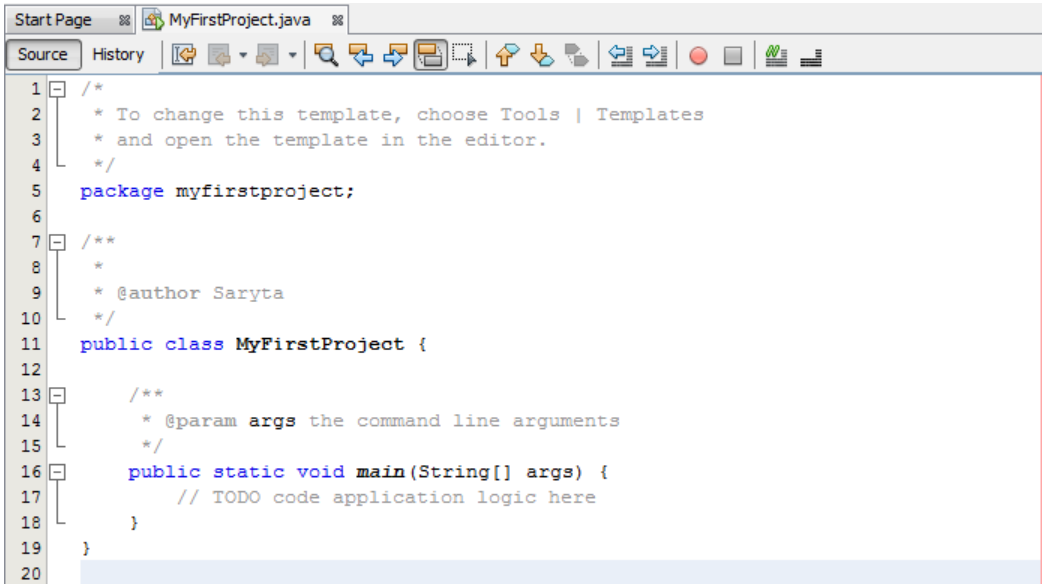
Look at the **Projects** area at the top left of the screen (if you can't see it, click **Window** >> **Projects** on the menu bar).

Projects area



If you can't see **MyFirstProject.java** click the plus symbol on the **Source Packages** folder to expand your project. On the right hand side of the screen is the code editor window and it should display the same **MyFirstProject.java** project. If you can't see the code editor window, double click **MyFirstProject.java** in your **Projects** window above. The code will appear like shown below, with your name as the program author.

Code editor window



Notice in the code editor diagram above the class is called **MyFirstProject.java**

public class MyFirstProject.java {

This is the same name as the java source file in the project window **MyFirstProject.java**. When you run the program the compiler demands that the source file and the class name should match.

So if the .java file is called **MyFirstProject** but the class is called **myfirstproject** then an error will occur in your program, however the package name can be different and it doesn't have to be the same as the source file or the class name.

Java Comments

If you look on the code editor window, you'll notice that some text is grayed out, with forward slashes (/) and asterisks (*). The grayed out areas are called comments.

Comments are ignored by the compiler and so you can use the comments to document your program.

To use a single line comment use double forward slashes (//). To use multi-lines comments you can either use many single line comments or you can use slashes (/) and asterisks (*)

```
//Here is a single line comment.

/*
   Here is a multi-lines comment
   that makes use of slashes (/) and asterisks (*)
*/
```

Notice the above multi-line comment starts with /* and ends with */.

Sometimes we also use comments when we want to trace errors in the program. You can select a segment of a program code and then use the comment and uncomment toolbar shown below:

Have a look at the code editor window diagram again. The comments that starts with a single forward slash and two asterisks (/**) and ends with an asterisk and one forward slash (*/) are called **Javadoc** comments. Each text line of Javadoc comment starts with one asterisk.

```
/**
*  This is a Javadoc comment
*  Each line starts with one asterisk
*/
```

The purpose of Javadoc comments is to document your code. The documentation can then be converted into a HTML (Hyper Text Markup Language) document that can be helpful to your code users. To convert your code to HTML, click **Run** from the menu bar and select **Generate Javadoc**.

You might not see a useful HTML document since you have not added much code to your program yet. For now you can write your code with no or minimal comments so we are going to delete all the comments inserted by NetBeans but we shall add comments as we proceed.

Java is **case sensitive**, that means if a keyword is supposed to be typed in uppercase and then you type it in lowercase you'll get an error when compiling the code.

Also, note that Java code instructions ends with a semicolon (;). Missing on the proper case and omitting the semicolon are some of the common errors in Java programming by beginners so, watch out!

The structure of a Java class

Let us now examine the following segment of our code:

```
public class MyFirstProject {

}
```

The above part of the code is called a class segment and it denotes the starting and ending of a Java class segment code. The start of a code segment is done with a left curly brace (**{**) and ends with a right curly brace (**}**). Any code inside the left and right curly braces belongs to that code segment.

Notice inside the curly braces for the class segment is another code segment shown below:

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

}
```

The above part of the code is called a method, **public** denotes that this segment of the code can be accessed outside the class without creating any new objects as denoted by the keyword **static**, **void** means this part of the code does not return any value.

The "**main**" part denotes the starting point of the Java program. When you run the program, Java compiler locate for this part of the code and execute any code inside "**main**". You will always need to include a "**main**" in your code.