JDK Installation

NetBeans 12 requires you to install JDK separately. NetBeans 8.2 contains the JDK built in.

Download JDK:

https://www.oracle.com/java/technologies/javase-downloads.html
Select the installer appropriate for your system.

You must accept the Oracle Technology Network License Agreement for Oracle Java SE to download this software.

- I reviewed and accept the Oracle Technology Network License Agreement for Oracle Java SE

Agree to the license and then click on Download.

Run the installer when it has been downloaded.
Select **Next**.

Take the default settings and select **Next**.
It may appear to hang, give it time to install.

-When done click on **Close**.
Installation guides from Oracle:

https://docs.oracle.com/en/java/javase/14/install/overview-jdk-installation.html

NetBeans does not require you to adjust the PATH or other environmental variables. It worked fine without it.

Now you are ready to install NetBeans
NetBeans

https://netbeans.org/

Select the Download button.
Download the LTS version

Select the installer that matches the version that matches your Operating System (Such as Windows, Linux or Mac).
Select the suggested mirror to download from.

Save and then execute the installer.
Select **Next**.
Accept the License Agreement and then select Next.
Leave the defaults. Select **Next**.
Leave checked to Check for Updates. Then click Install to proceed with the install.
When you have read the message, click on Finish.

Click on the Apache NetBeans IDE icon to begin.
If you used a previous version of NetBeans it will ask to import them. If you wish to import them click on Yes, otherwise to start with the default settings click on No.
Sample NetBeans Project

Making a New Project

From the File menu, select make a New Project.

From the Categories menu select Java with Ant. From the Projects menu select Java Application. Then press Next.
New Java Application

The *Project Name* typed was *HelloWorld*. The NetBeans generated the rest of the values (including the Create Main Class). Click on *Finish* when done. It will take a few seconds as the project is created.
Here is a blank project. We will now fill in some code to make the program do something. Replace line 18 with the following:

```java
System.out.println("Hello World!");
```
Now we have a new program:
Running a NetBeans Project

To run your code.

From the Run menu, select Run Project. You could press the keyboard shortcut F6, if you prefer.
You are also able to press on the green arrow to Run your project.

As shown above, the output from the Output panel in NetBeans.
NetBeans should automatically build when you save the file. If you are not getting the proper output, manually build your project. To make sure the code is built properly, from the Run menu select Clean and Build.

Now run the project to see the output.

This window may pop up.

A popup window about Usage Statistics may appear when you start NetBeans. If you would like to participate in anonymous information being sent to NetBeans to help them improve the program, click on I Agree. Otherwise, you may wish to say No, Thank You.
Debugging Example

Let’s make a more complicated program. We will make a New Java Application. The Project Name will be called Counter. Check on Finish when done.

Now you have a Java Program created with a skeleton code for you to fill in:
We will begin our program at line 19:

```java
int i=0;
i++;
i++;
i++;
System.out.printf("%d\n",i);
```
If we Run the program (Press F6), we see the following output:

![Output Counter (run)](image)

The value of i is displayed on the screen. But how did it become the value of 3. Let us debug the program to see what happens to the variables as the program runs.

Select the location the source code that you would like the program to stop at.

```java
17 | public static void main(String[] args) {
18 |     // TODO code application logic here
19 |     int i=0;
20 |     i++;
21 |     i++;
22 |     i++;
23 |     System.out.printf("%d\n",i);
24 | }
```

In this case we selected in the source code at line 21.
From the **Debug** menu, select **Run to Cursor** (or Press the keyboard shortcut F4).
The debugger stopped the program. We can inspect the variables. Make sure the Variables tab in the lower right in part of the window is selected.

```java
20 | i++;
22 | i++;
23 | i++;
```

Notice at the variable i, has the value of 0.
From the **Debug** menu, select **Step Over** (easier to press F8). The debugger will step over this instruction to the next instruction.

As you write more complex programs, you may need to Step Into (to see inside of a method).

The value of i became 1. Press F8 again and i becomes 2.
If we select line 25 of the program, and select from the debug menu to Run to Cursor (F4).

```
24  System.out.printf("%d\n", i);
25  }
26  }
```

It shows that indeed, i has the value of 3, when it will be displayed to the console.

If you want to stop the debugger, you can select the Red Stop button.
Source Format

You can have NetBeans automatically format the source code for you. From the Source menu, select Format.

It will automatically indent, put spaces in between variables and assignments, and fix other formatting issues.

Before formatting is applied:

```java
public static void main(String[] args) {
    // TODO code application logic here
    int i = 0;
    i++;
    i++;
    i++;
    System.out.printf("%d\n", i);
}
```

After formatting:

```java
public static void main(String[] args) {
    // TODO code application logic here
    int i = 0;
    i++;
    i++;
    i++;
    System.out.printf("%d\n", i);
}
```
Spell Checking

If you get a spelling error, the editor will give a red squiggle under the word to indicate an error. You can correct it by hand, or have the system give you suggestions.

To bring up the suggestions, select the word and press Alt-Enter.
From the **File** menu, select **Print**.
The preview is missing the line numbers. Let us add them. Select Print Options.
Select **Line Numbers** and **Wrap Lines**.
Select the Text Font settings
Select **Size 12.** Then Press **OK.**
Press **OK** to save the changes.
Now the preview shows the line numbers and the text wrapped on the page. Press Print to Print your file.
When submitting your assignment, your instructor might request a PDF version of your program. To do so, print your file, and when the print dialog box comes up, select Microsoft Print to PDF. Some systems will say Adobe PDF and others will say Microsoft Print to PDF as the printer name.

On a Mac, when printing, from the PDF pop-up menu (at the bottom of the print dialog box), select Save as PDF.
Exporting / Importing

If you want to move your project to another machine, you can Export it.

From the File menu, select Export Project, To ZIP.

Select Browse to specify a location to save the zip file.
Specify the location to save and then press **Export**.

To import the project back on a new machine (or if you used the export as a backup and wish to restore back the original), from the **File** menu, select **Import Project, From ZIP**.
The top Browse button will let you select the ZIP file that contains your project. The bottom Browse button selects the folder that contains where your NetBeans projects are stored. Leave the bottom Folder location alone. Select from the top Browse button where to find the ZIP file.

Press Import once things are set properly.

If the project already exists it will replace (overwrite it). Do this only if you are restoring to replace the current project files with the files contained in the ZIP file. Otherwise you can Change Import Folder to place the files in a new folder. Otherwise press Cancel to not do the import sequence.
Multiple Class Files

The project has an expandable/collapsible tree listing of the files in your project.

Click on the + symbol to expand it.

Double click on the file name to open it.

The symbols next to the file name mean:

- Counter.java
- ☑ Needs to be compiled.
- ☑ Contains main class.
You can press **F9** to compile the file (or from the Run menu, select **Compile File**).

**Build/JAR file**

If your project contains multiple files, you can build a **JAR (Java ARchive)** file that contains the all your class files and associated resources (such as media or data files) for your profile into a single file for distribution. To do so, you can select from the Run, **Build Project**.
From the toolbar you can select the Hammer icon to Build as well.

You should get a Build Successful message. If your code contained errors, you would see them below.

Such as for the following:
If you want to Clean (erase the compiled class files) and then Build the project completely, you can select **Clean and Build Project**, from the **Run** menu. You should do this, if you previously built a project and want to assure that newly compiled classes are in the project.

From the toolbar you can select the Hammer/Broom icon to Clean & Build as well.

The JAR file is located in the dist folder of your project’s directory. From a **Command Prompt** you can run it by doing the following command (in our case):

```
java -jar "C:\Users\Staff\Documents\NetBeansProjects\Counter\dist\Counter.jar"
```

This jar file can then be used to run your project from any machine that has a JVM (Java Virtual Machine).

NetBeans will create a README.TXT which gives the precise command that can be used to invoke it. The README.TXT file is best opened in Wordpad (on Windows).
You should get into the habit of writing your comments in the Javadoc format:

- https://www.tutorialspoint.com/java/java_documentation.htm

Traditionally /* and */ are used to indicate normal text comments to the compiler. However, Javadoc comments have an opening tag (also known as the begin-comment delimiter) with two asterisks.

For example:
/** I am a Javadoc comment. */

Here is an example for a method.

```java
/**
 * Add two integers together.
 * @param numA First number to add
 * @param numB Second number to add
 * @return int Sum of numA and numB.
 */
public int addNum(int numA, int numB) {
    return numA + numB;
}
```

When generating Javadoc files, you will get message Warning: Leaving out empty argument 'windowtitle', if you do not set the document title. You can ignore this. However if you want to add more to the title than just the package name, go to the Properties of the project. From the File menu, select Project Properties.
From the Build category, select **Documenting**. Then enter in a **Browser Window Title**.

To generate the Javadoc information for your project, from the **Run** menu select **Generate Javadoc**.
That will open up the Javadoc files for your project. For example with AddNum, it will have a description of the class.

![Class Summary](image)

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddNum</td>
<td>Add Two Numbers!</td>
</tr>
</tbody>
</table>

Click on the class name to get more information on the class.

![Method Detail](image)

```java
public int addNum(int numA,
                   int numB)
Add two integers together.
Parameters:
numA - First number to add
numB - Second number to add
Returns:
int Sum of numA and numB.
```

For example, this is the information was generated from the Javadoc comments. The Javadoc comments became HTML code viewable in a web browser. The Javadoc associated files are located within the `dist\Javadoc` directory of your project. You might be encouraged to design a Java library to for use by other developers around the world. The Javadoc comments you add will be invaluable to others. The more descriptive you are, the easier others (and yourself) will be able to read your code.
Code Completion

Code Assistance in the NetBeans IDE Java Editor: A Reference Guide
https://netbeans.org/kb/73/java/editor-codereference.html

Code Completion will help assist you when writing your programs. It will offer the various methods and objects associated with the code that you are writing and complete the command you are writing.

Make sure that Code Completion is enabled. From the Tools menu, select Options.
Select Editor options, then select **Code Completion**. You can adjust the **Auto Popup** options as you see fit. But for learning purposes, it’s best to keep them set to aid you in coding.
When typing your code and you press a "." a code completion windows popup.

```java
public static final PrintStream err
```

The "standard" error output stream. This stream is already open and ready to accept output data.

Typically this stream corresponds to display output or another output destination specified by the host environment or user. By convention, this output stream is used to display error messages or other information that should come to the immediate attention of a user even if the principal output stream, the value of the variable out, has been redirected to a file or other destination that is typically not continuously monitored.
However, you can press **Ctrl-Space** at any time to pop open the code completion window. Such as if you type `Sys` and then press Ctrl-Space up will come the following.

```java
public static void main(String[] args) {
    System.out.println("Hello World!");
    System.err.println("|");
    System.out.println("className.methodName()"); systrace
    System.out.println("|");
    System.out.println();
    System
```

You can then select which line to insert into your code by using the arrow keys to select it and pressing enter (or by double clicking on it).
Margin

From the Tools/Options window, go to the Editor tab and select Formatting.

Uncheck “Use all Language Settings”

Set Right Margin to 75
Insert Code
You can also have NetBeans generate get and set methods for access to the variables in your class.

Place the cursor at the location in the file you want to insert the code and then press Alt + Insert (or from the Source menu, select Insert Code).
It is also accessible by right clicking at the code location and selecting **Insert Code**.

A menu will appear that allows you to generate a piece of code.

**Example – Getter and Setter**

Generate get and set methods for the private variable *item*.

```java
public class Test {
    private int item;
}
```

Right click in the class that contains the variable(s) you want to generate code for.

Select **Insert Code**.
Select **Getter and Setter**.

Check the variables that you want getters and setters generated. Then press the **Generate** button. If you have multiple variables, you can select them all by clicking the check box next to the class name.
For the variable `item`, NetBeans generated the functions `getItem` and `setItem` for your program to gain access to the variable.

```java
private int item;

public int getItem() {
    return item;
}

public void setItem(int item) {
    this.item = item;
}
```

**Example - Constructor**
Another useful code to insert is a Constructor.

![Constructor options](Generate)
Select the fields you want to be initialized by the constructor and then press **Generate**.

```java
public Test(int item) {
    this.item = item;
}
```

A Tutorial is available here:

[https://platform.netbeans.org/tutorials/nbm-code-generator.html](https://platform.netbeans.org/tutorials/nbm-code-generator.html)
Updates

If you see a popup bubble that indicating “updates found”, you should update. If this bubble disappears, or you want to check for updates go to the Help menu and select Check for Updates.
You will see which updates are available. Select **Next**.

Restart NetBeans to complete the installation of the updates.
You will see the NetBeans Updater window while the updates are in progress.

When the updates are completed, NetBeans will open.

Enjoy using NetBeans...