

Eclipse

JVM, main method and using Eclipse

Produced Dr. Siobhán Drohan
by:



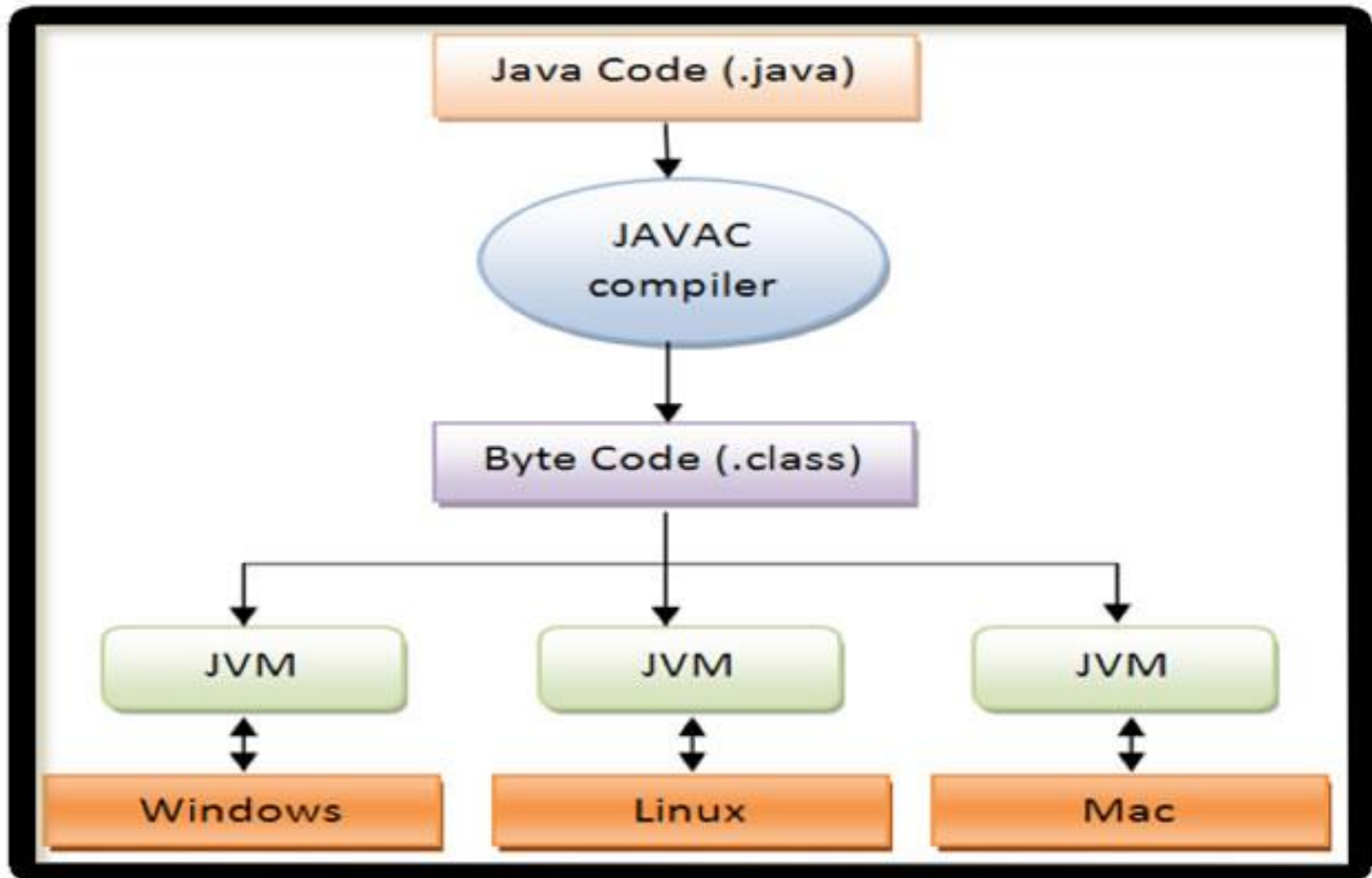
Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Department of Computing and Mathematics
<http://www.wit.ie/>

Topics list

- Files in Java.
- Java Virtual Machine.
- main method.
- Eclipse IDE.
- Shop Project.

Files in Java (1)



Files in Java (2)

- Java code is written in **.java** file.
 - This code contains one or more Java language constructs like Classes, Methods, Variable, Objects etc.
- When you compile this code a **.class** file is generated.
 - A class file is also known as “**byte code**”; it is the input to Java Virtual Machine (JVM).
 - The JVM reads this code, interprets it and executes the program.

Topics list

- Files in Java.
- Java Virtual Machine.
- main method.
- Eclipse IDE.
- Shop Project.

Java Virtual Machine (JVM)

- Java Virtual Machine (JVM) is a “virtual” computer that resides in the “real” computer as a software process.
- The JVM gives Java the flexibility of platform independence.
- The **.class** files can be run on any operating system, once a JVM has been installed (i.e. it is installed when you install the JDK).

Topics list

- Files in Java.
- Java Virtual Machine.
- main method.
- Eclipse IDE.
- Shop Project.

main method

- When you want to run a java project, the Java Virtual Machine (JVM) invokes the **main method** in the project.
- For the JVM to recognise it, the main method must have a specific method signature.

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


main method

- “main” must exist
- “main” must be public
- “main” must be static (class method)
- “main” must have a String array parameter
- Only “main” can be invoked automatically.

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

main method

```
public static void main(String[] args)
{
    Driver driver= new Driver();
    driver.runMenu();
}
```

- The main method should
 - create an object
 - call the first method

Topics list

- Files in Java.
- Java Virtual Machine.
- main method.
- Eclipse IDE.
- Shop Project.

What is Eclipse?

- Eclipse is an integrated development environment (IDE) for developing applications using the Java programming language and other programming languages such as C/C++, Python, PERL, Ruby etc.
- Eclipse is free to download and install.

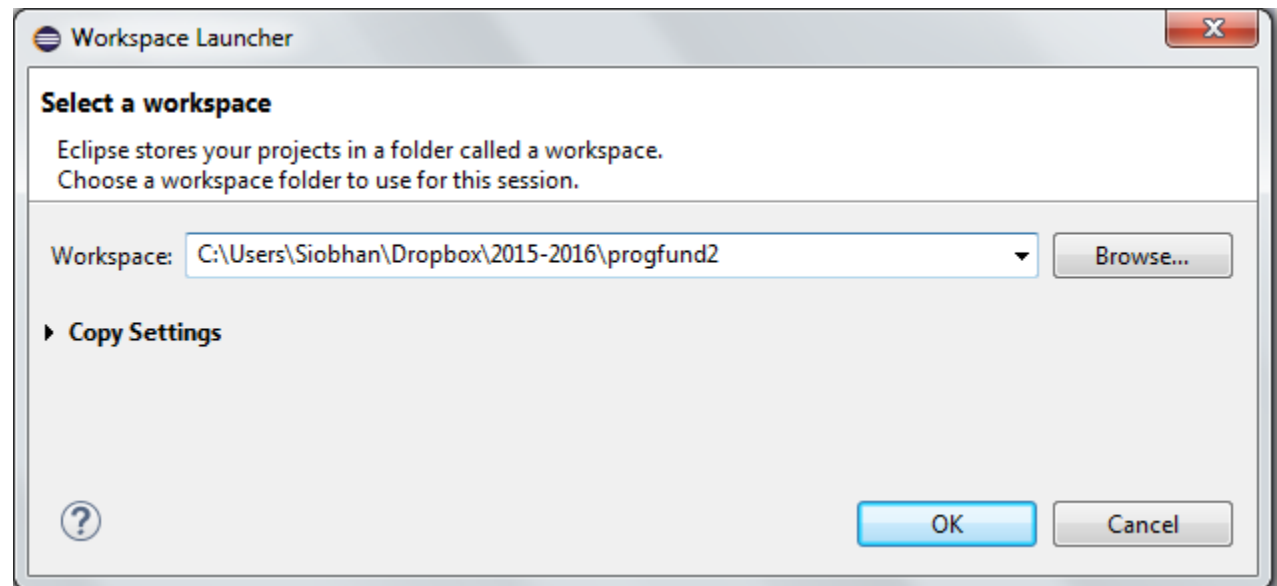
Downloading and Installing Eclipse

- <http://www.eclipse.org/downloads/>

	<h3>Eclipse IDE for Java EE Developers</h3> <p>275 MB 1,850,647 DOWNLOADS</p> <p>Tools for Java developers creating Java EE and Web applications, including a Java IDE, tools for Java EE, JPA, JSF, Mylyn...</p>		Windows 32 bit 64 bit
	<h3>Spring Tool Suite</h3> <p>Complete IDE for enterprise Java, Spring, Groovy, Grails and the Cloud.</p>		★ Promoted Download
	<h3>Eclipse IDE for Java Developers</h3> <p>166 MB 908,477 DOWNLOADS</p> <p>The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven integration and WindowBuilder...</p>		Windows 32 bit 64 bit
	<h3>Eclipse IDE for C/C++ Developers</h3> <p>176 MB 373,036 DOWNLOADS</p>		Windows 32 bit 64 bit

Launching Eclipse

- When eclipse starts up, it prompts you for the location of the workspace folder.
- All your data will be stored in the workspace folder.

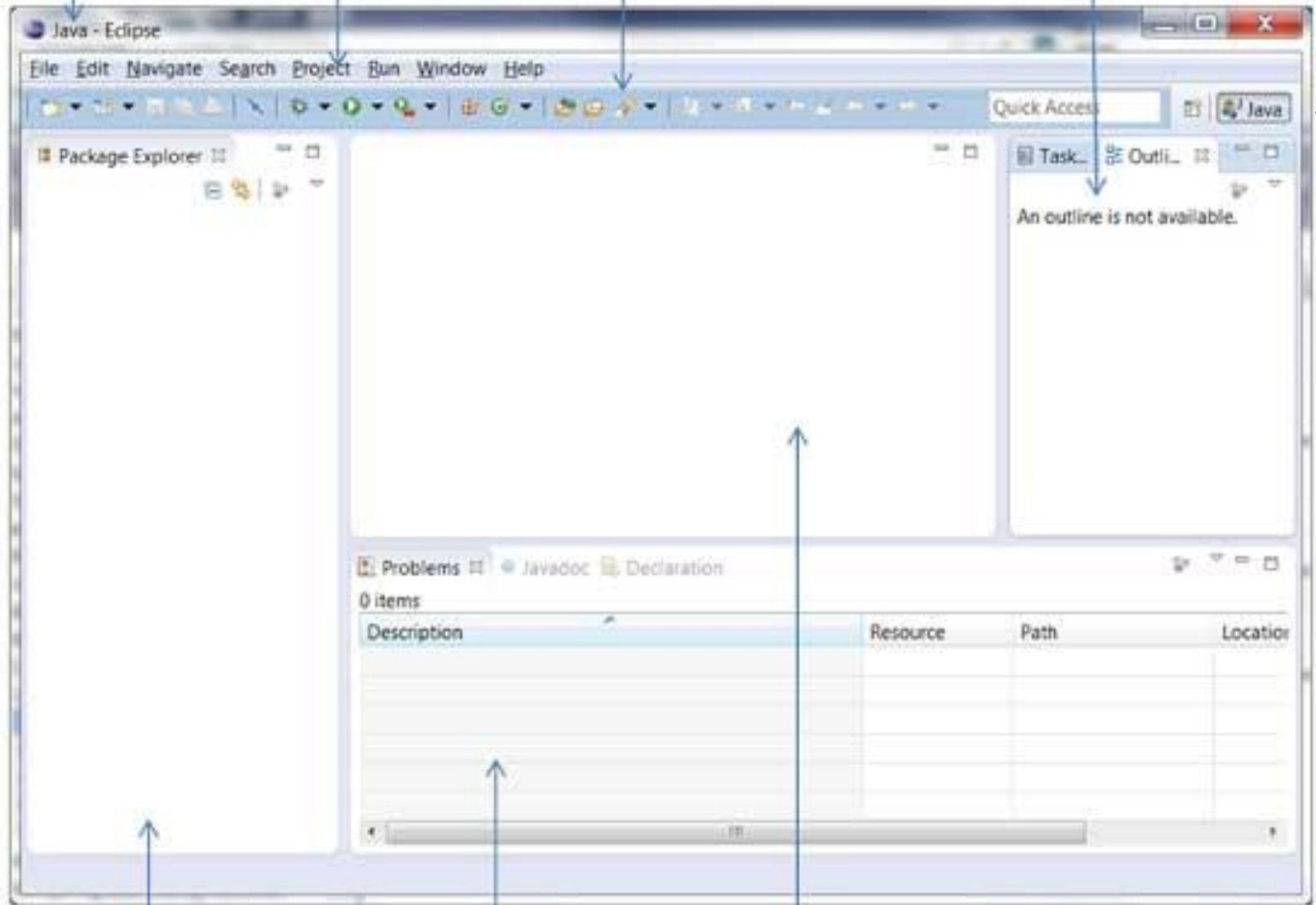


Name of current perspective

Menu Bar

Tool Bar

Outline View



Package Explorer View

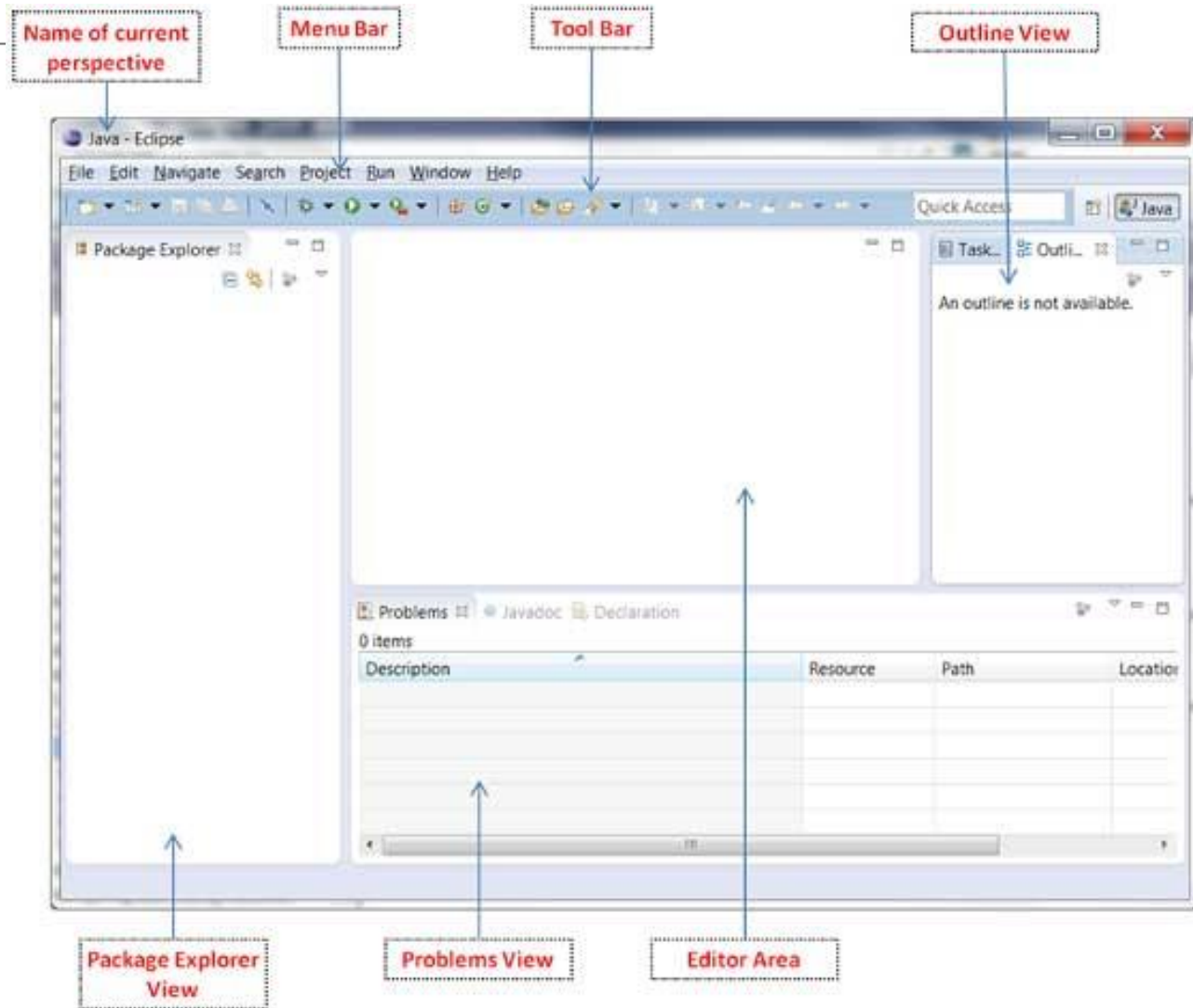
Problems View

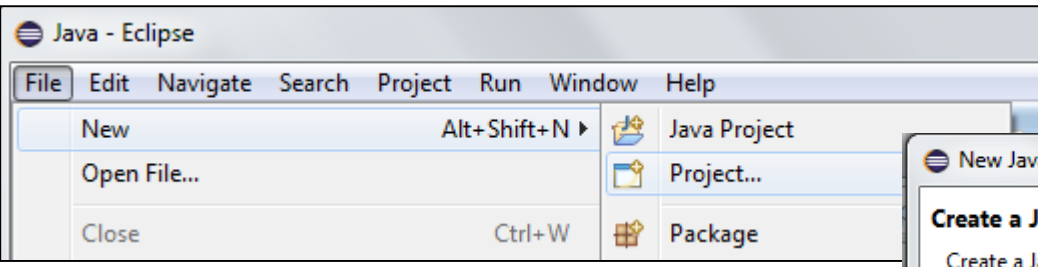
Editor Area

Perspectives

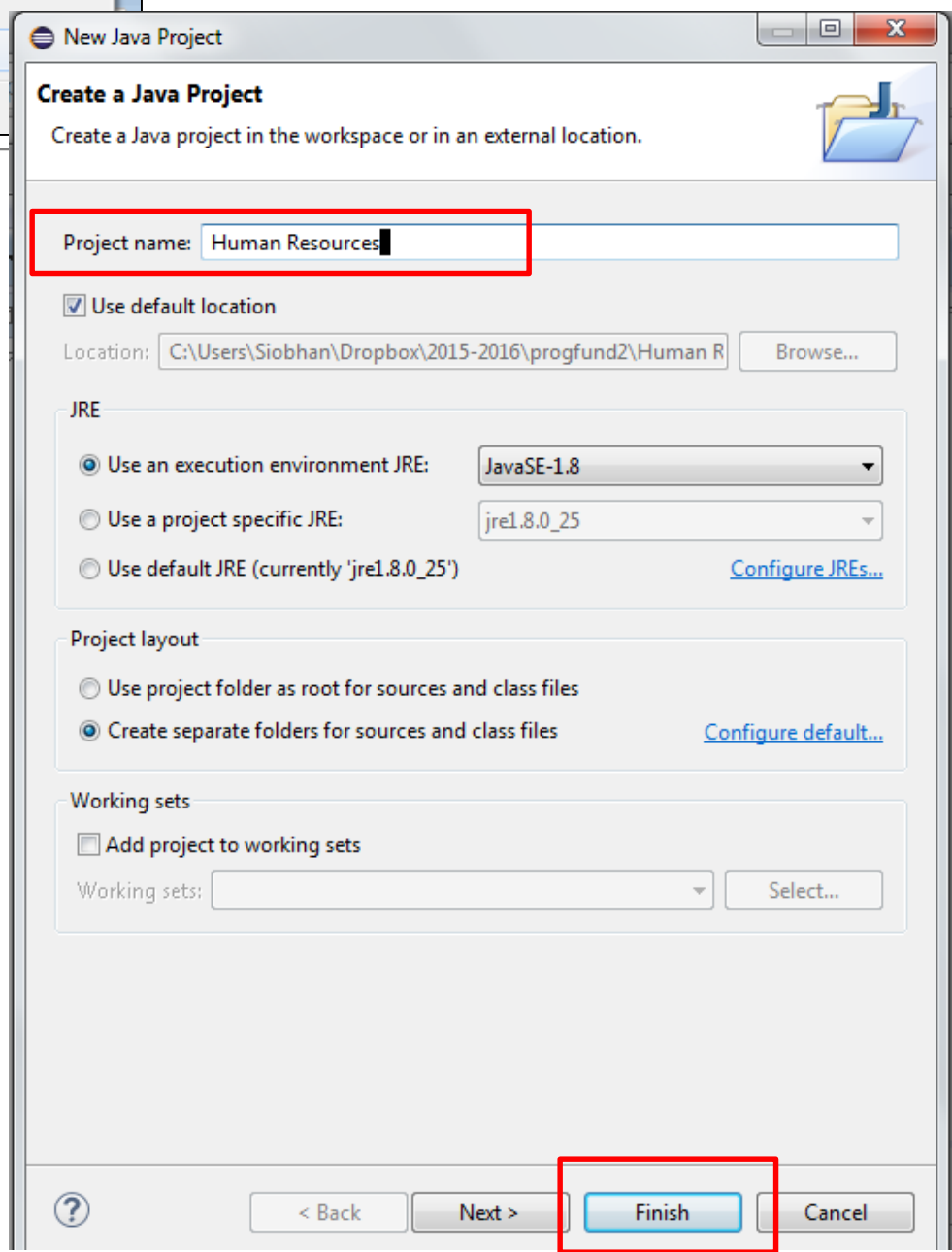
An eclipse perspective is the name given to the arrangement of views and editor area.

The default perspective is called **java**.

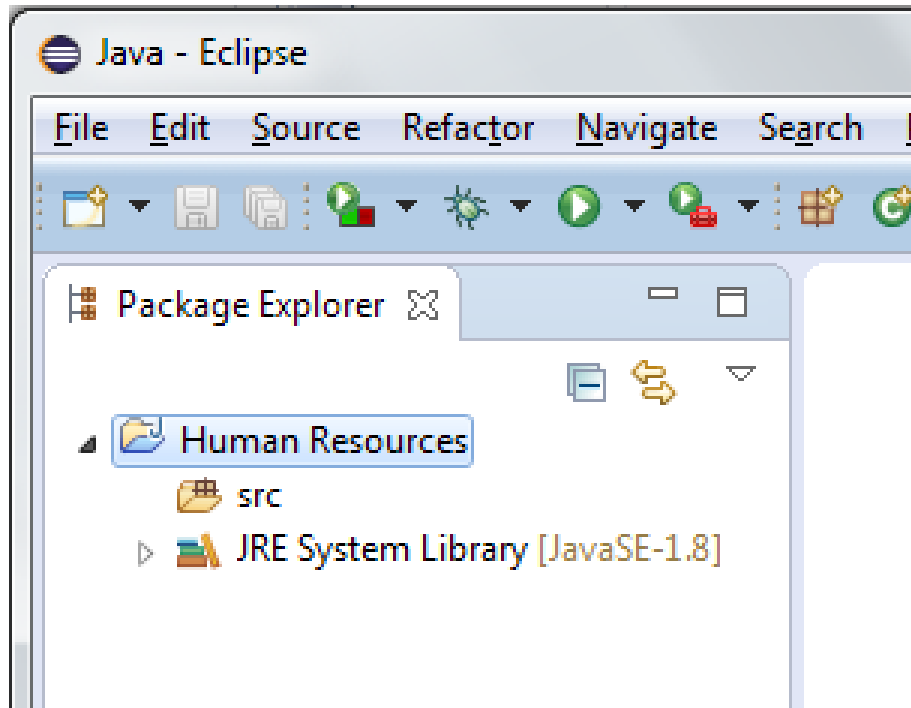




New Project Wizard



Viewing the New Project

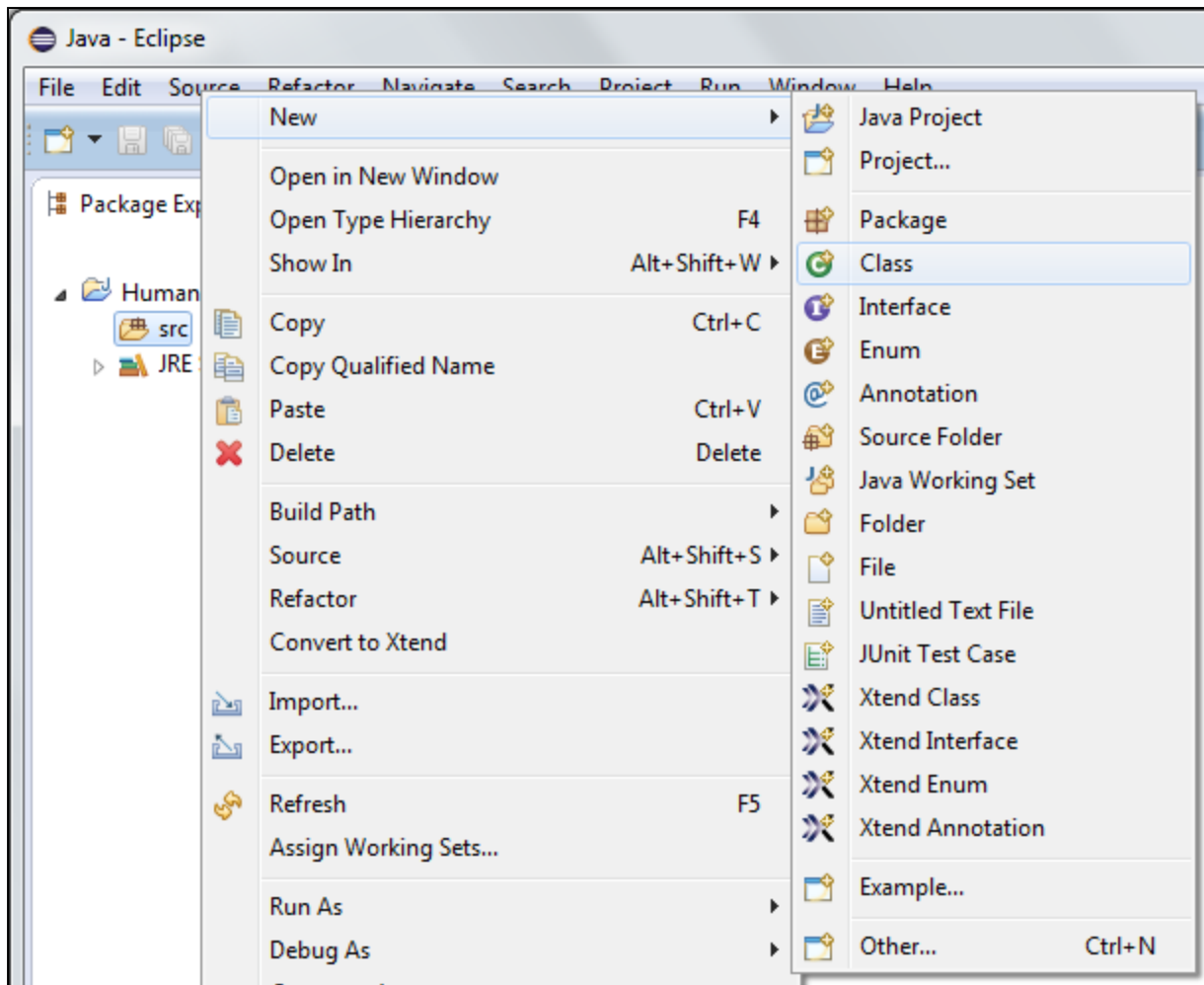


The package explorer shows the newly created Java project.

The icon that represents a Project is decorated with a "J" to show that it is a Java Project.

The folder icon is decorated to show that it is a java source folder.

New Java Class

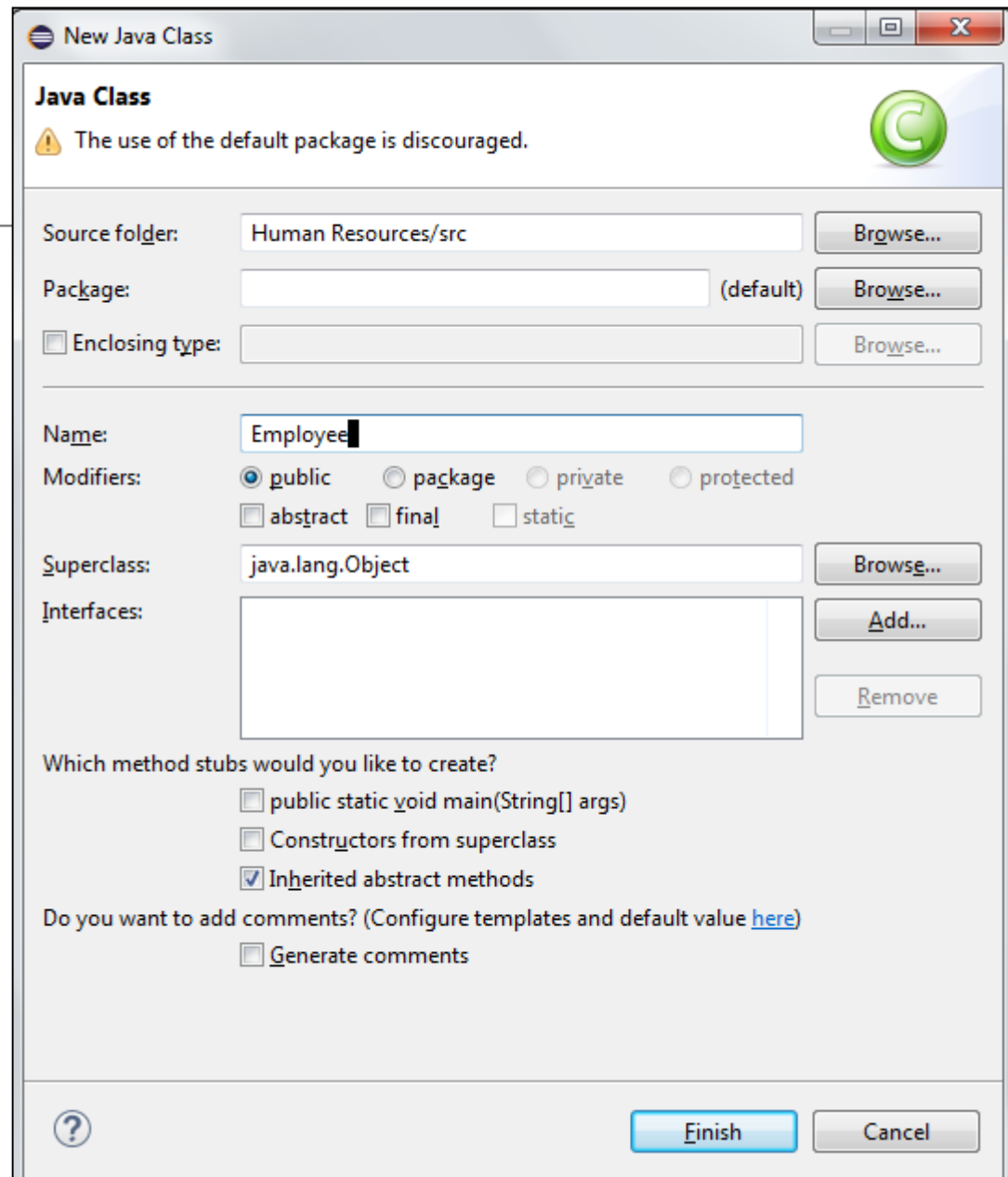


Right click on the **src** folder.

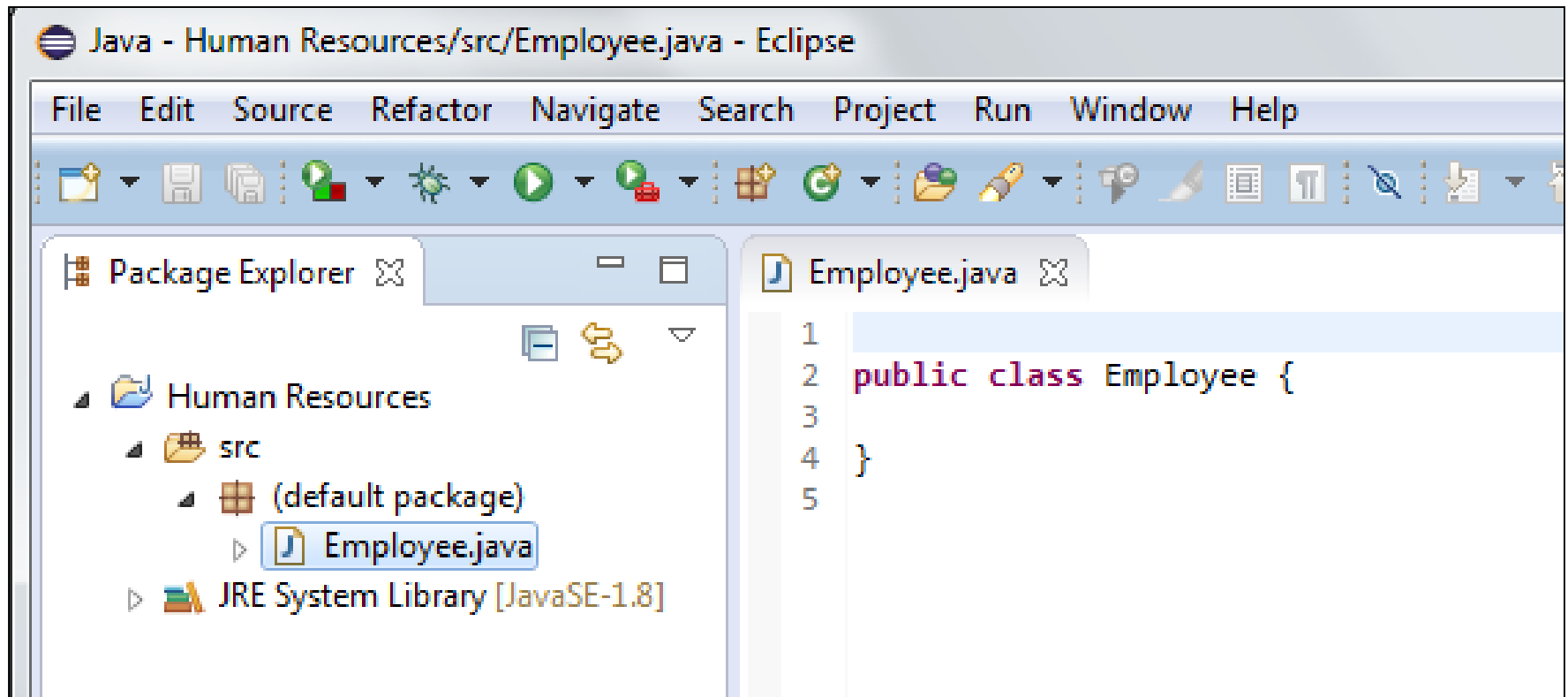
When the context menu appears, select **New**, followed by **Class**.

New Java Class Wizard

- Ensure the source folder and package are correct
- Enter the class name
- If your class will hold the main method, check the appropriate box to include it automatically.
- Click the Finish button when done.

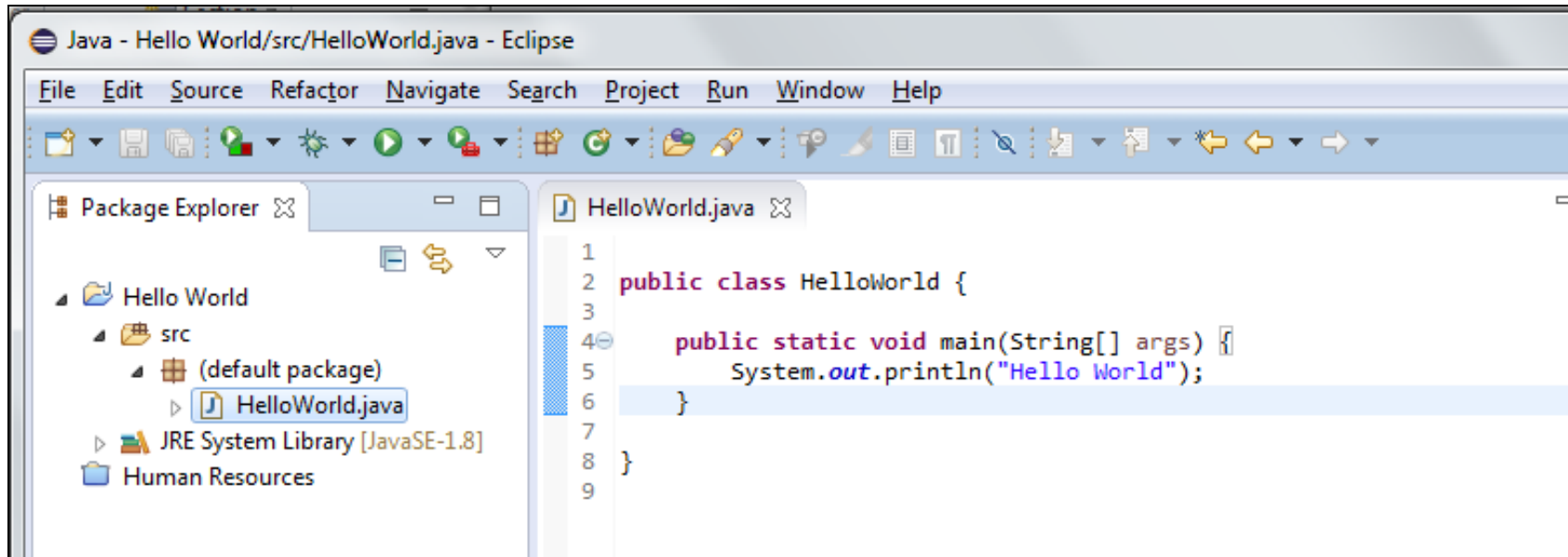


Viewing the New Class



Running a Java Program (1)

- We want to run the following code:



```
1  
2 public class HelloWorld {  
3  
4     public static void main(String[] args) {  
5         System.out.println("Hello World");  
6     }  
7  
8 }  
9
```

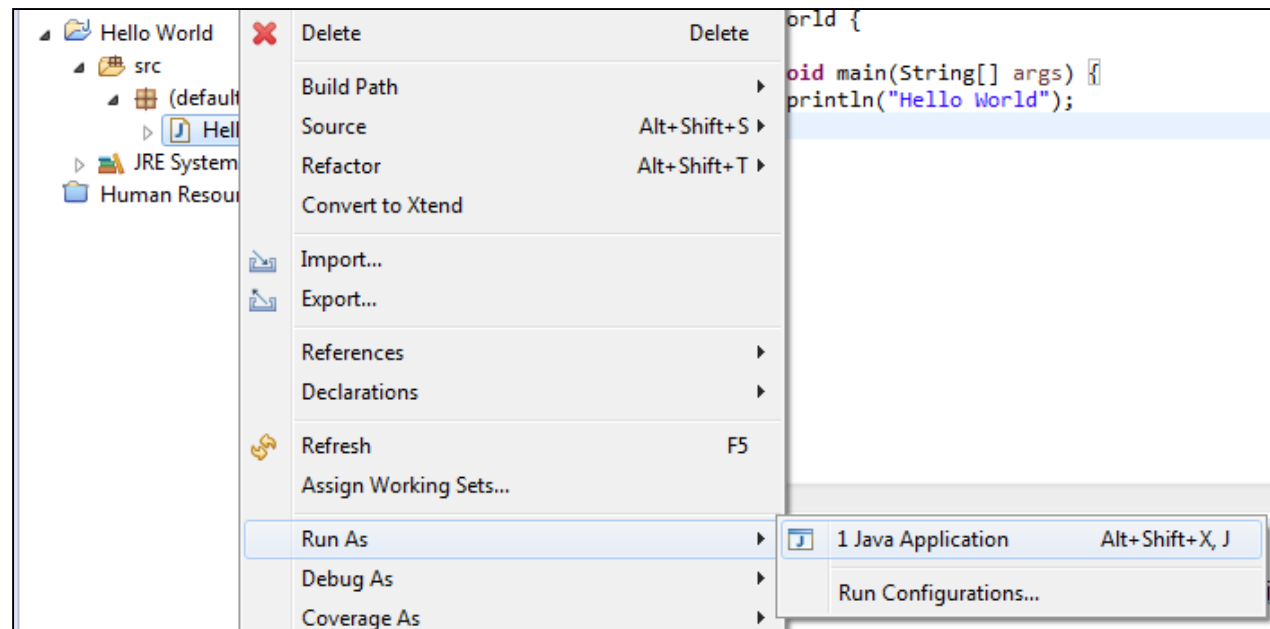
Running a Java Program (2)

- One way to run a Java program is to use the Package Explorer view.

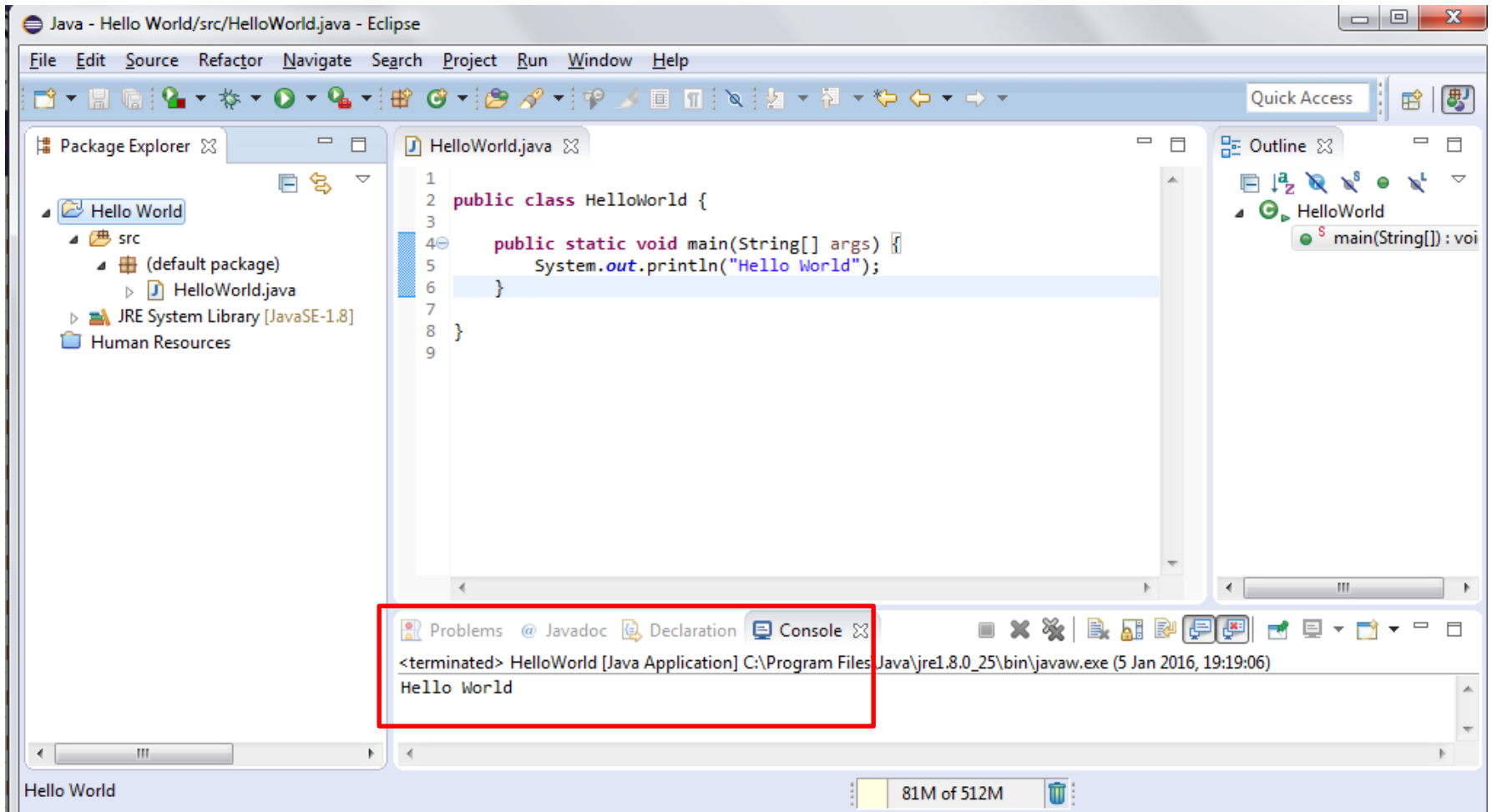
In the Package Explorer view:

Right click on the java class that contains the main method.

Select Run As > Java Application



Running a Java Program (3)

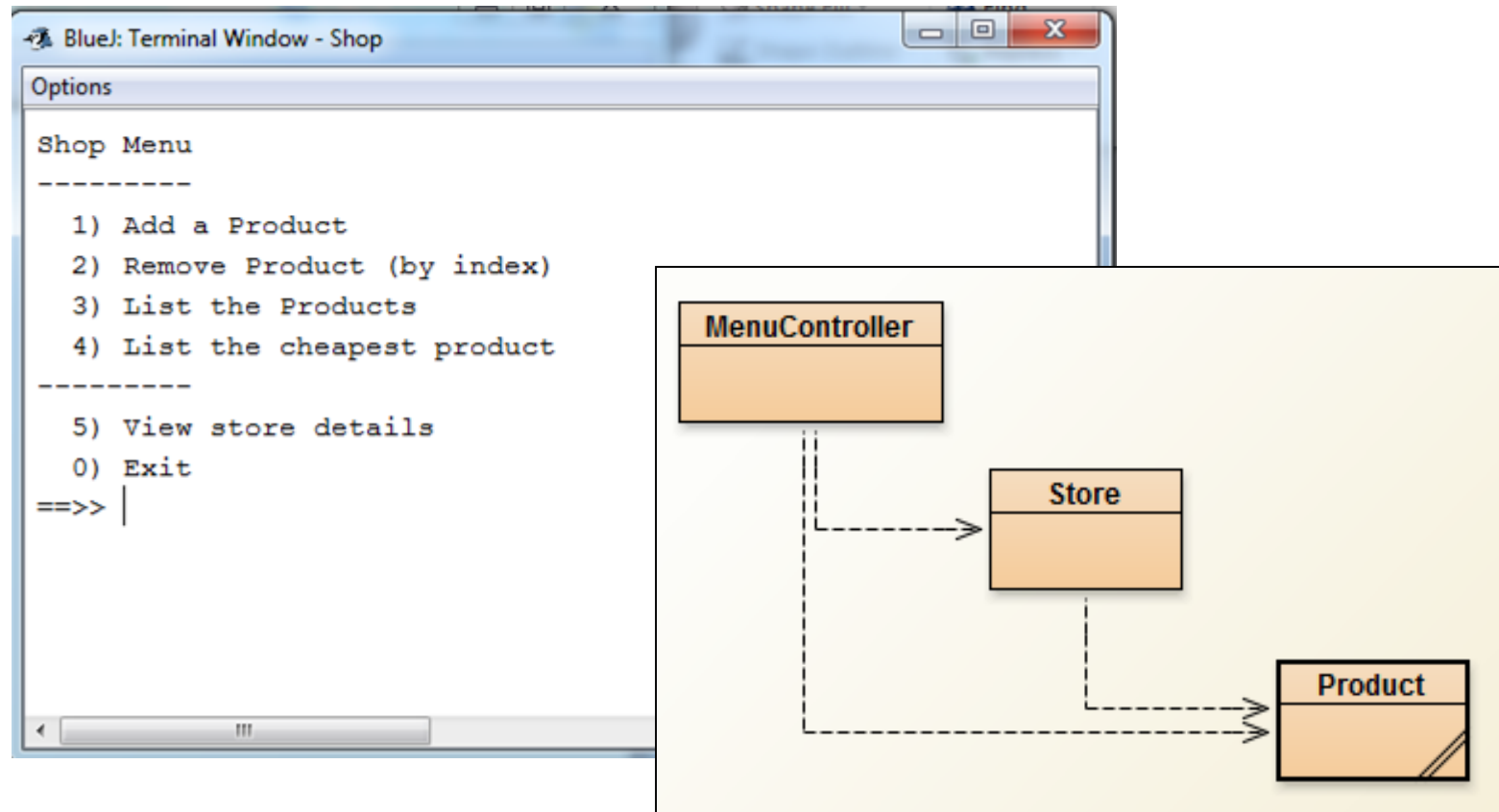


Topics list

- Files in Java.
- Java Virtual Machine.
- main method.
- Eclipse IDE.
- Shop Project.

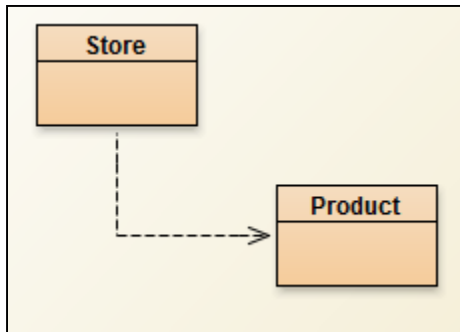
Shop Project

- Our **BlueJ** version of the shop project looks like this:

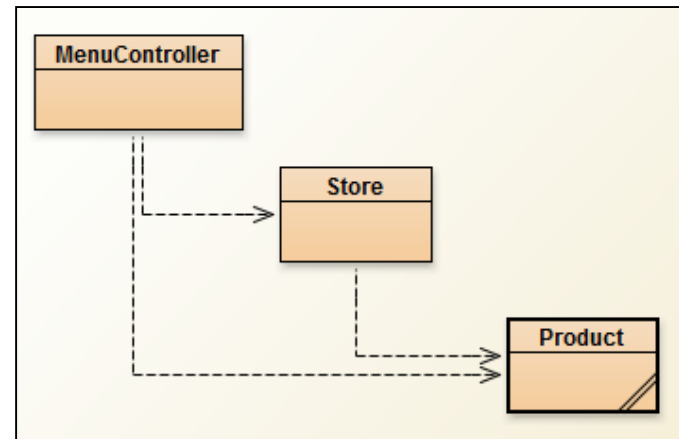


Shop Project

- Let's call the version without the console driven menu, **ShopV1.0**

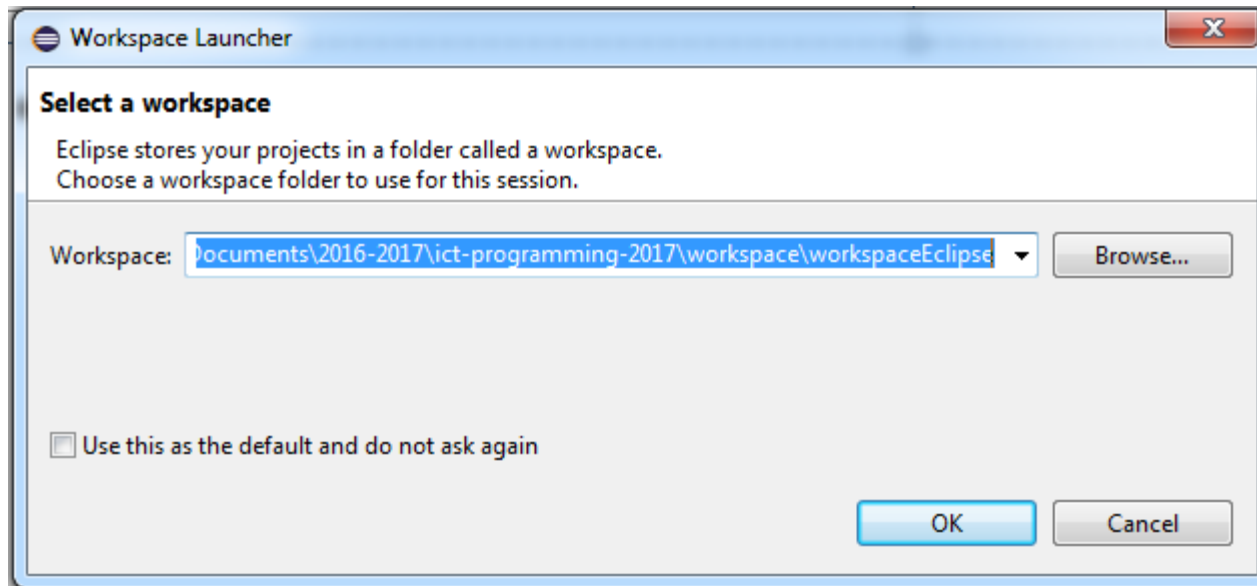


- Let's call the version with the menu, **Shop V2.0**



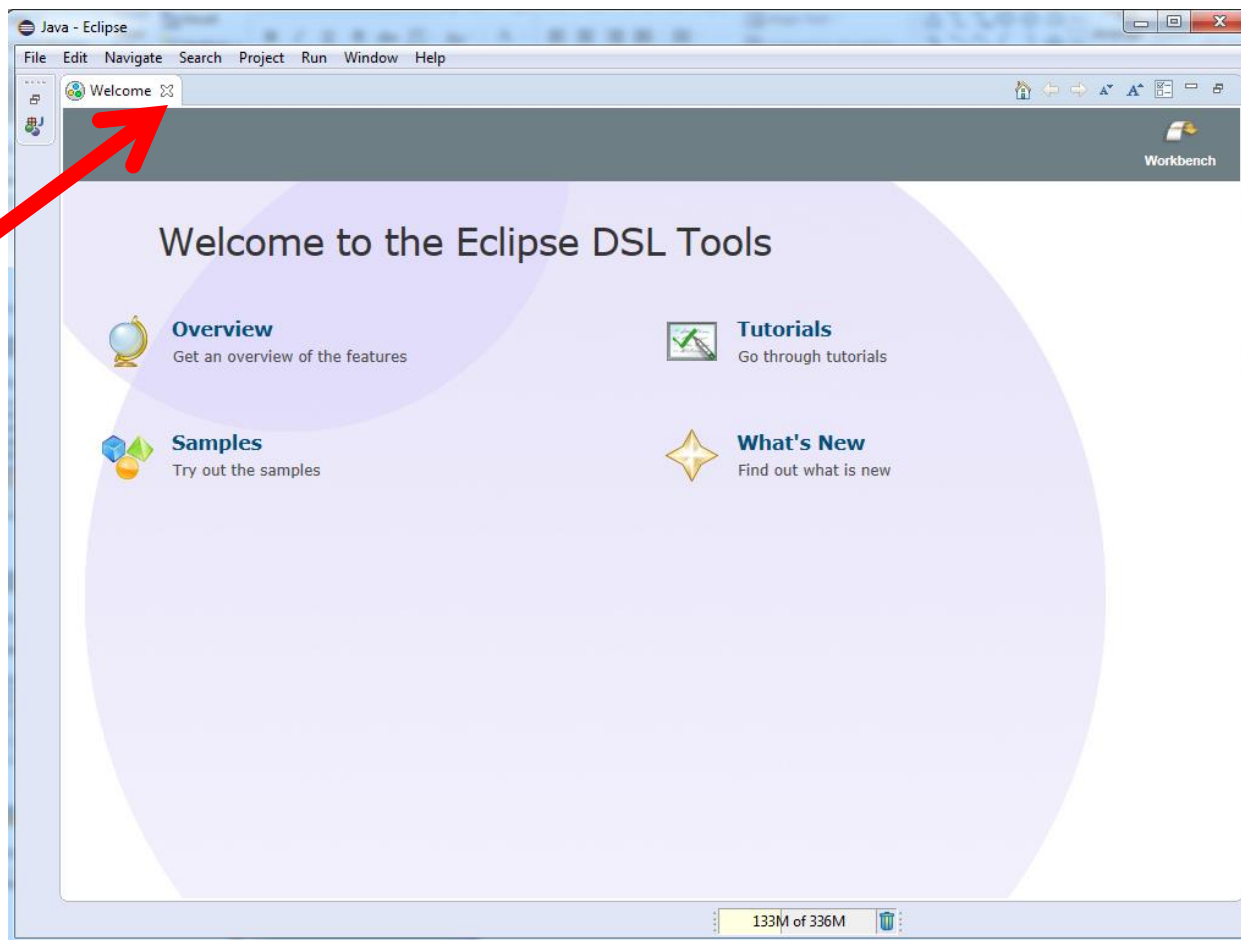
Let's move Shop V2.0 into Eclipse

- Start Eclipse and choose your workspace location



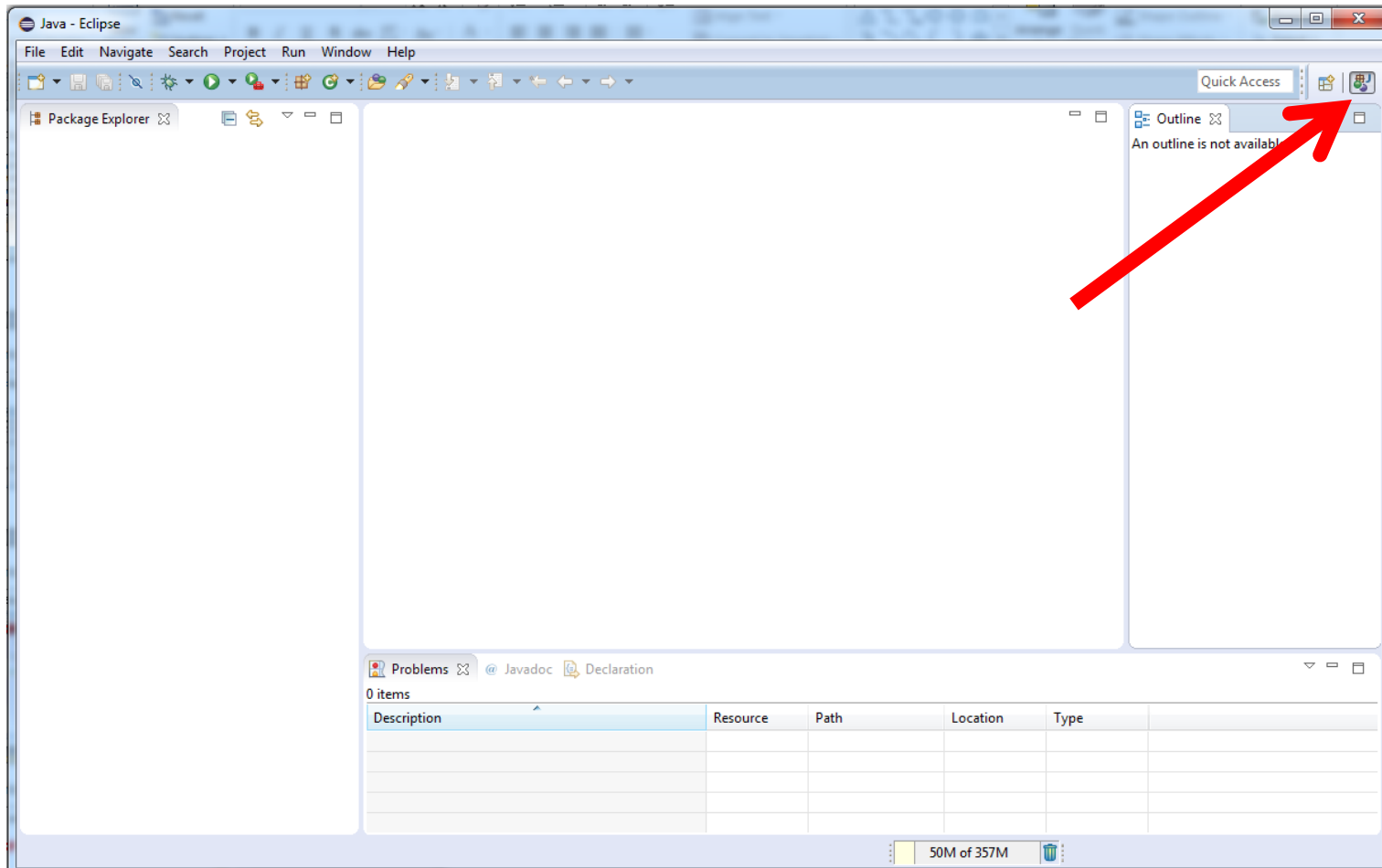
Let's move Shop V2.0 into Eclipse

- When Eclipse opens, close the welcome screen



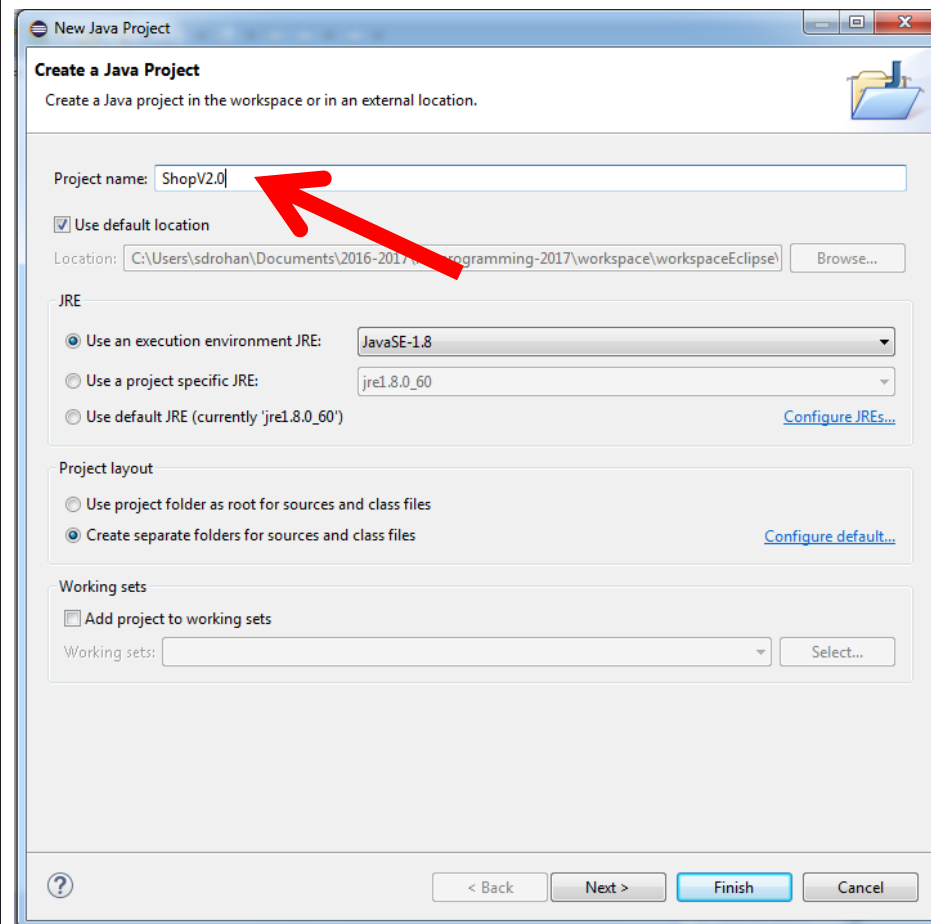
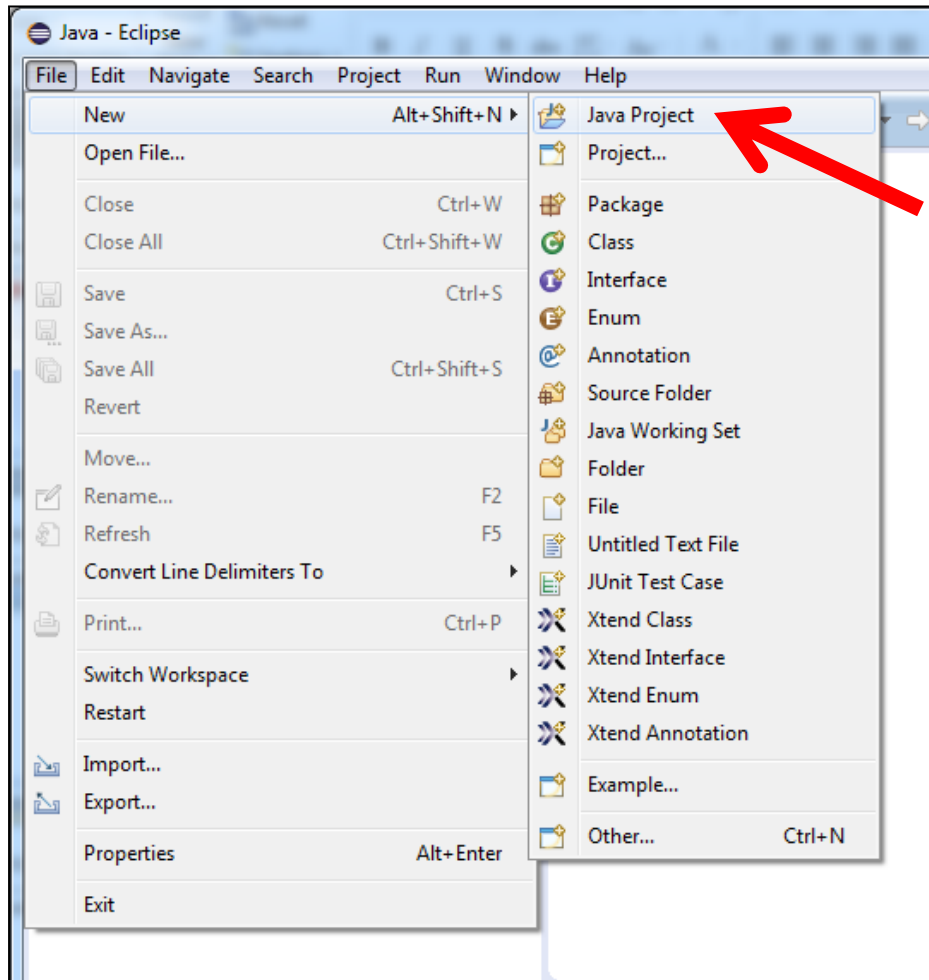
Let's move Shop V2.0 into Eclipse

- Ensure you are in the **Java** perspective



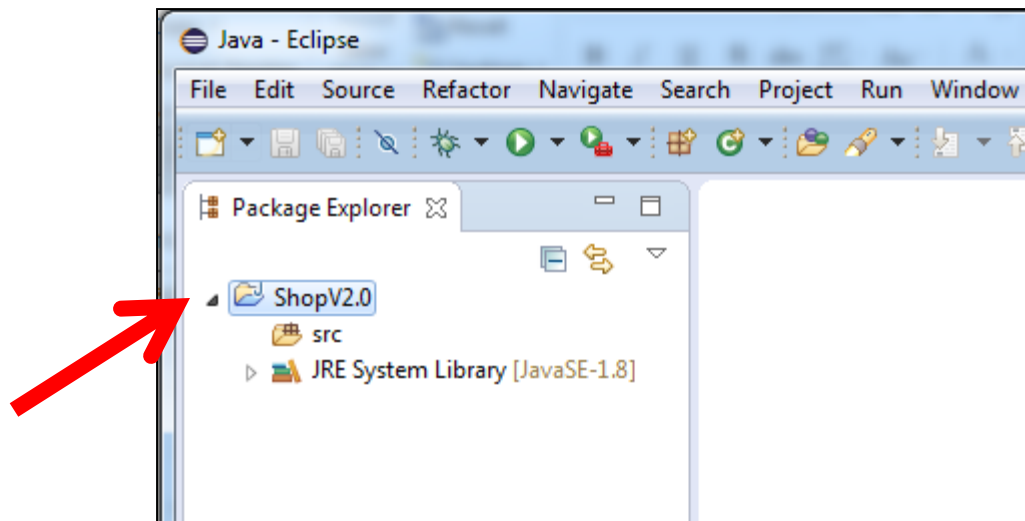
Let's move Shop V2.0 into Eclipse

- Create a new Java Project and call it ShopV2.0



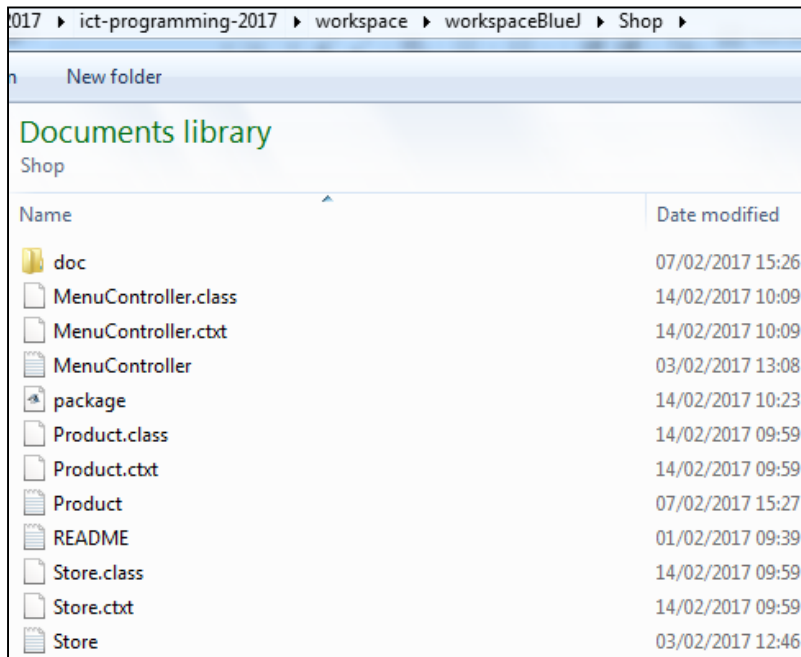
Let's move Shop V2.0 into Eclipse

- The following project will be created in your workspace

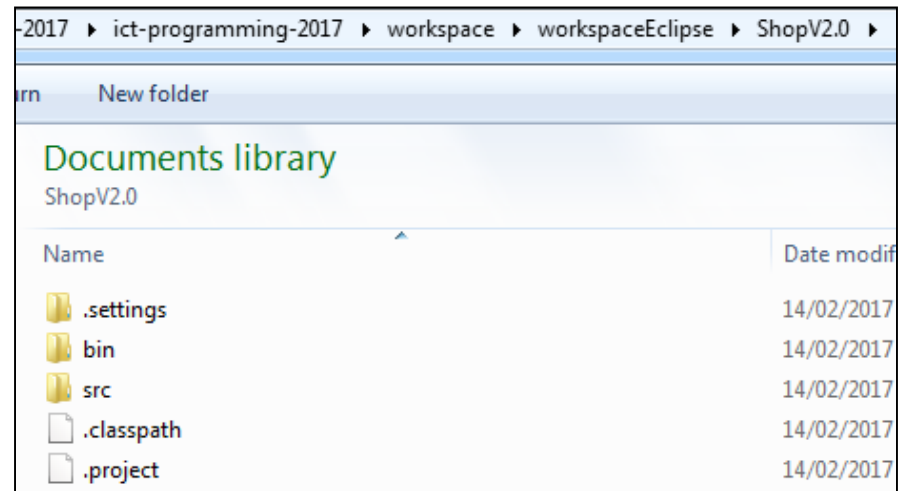


Let's move Shop V2.0 into Eclipse

In Explorer, open the folder where your existing **BlueJ** Shop project is stored:



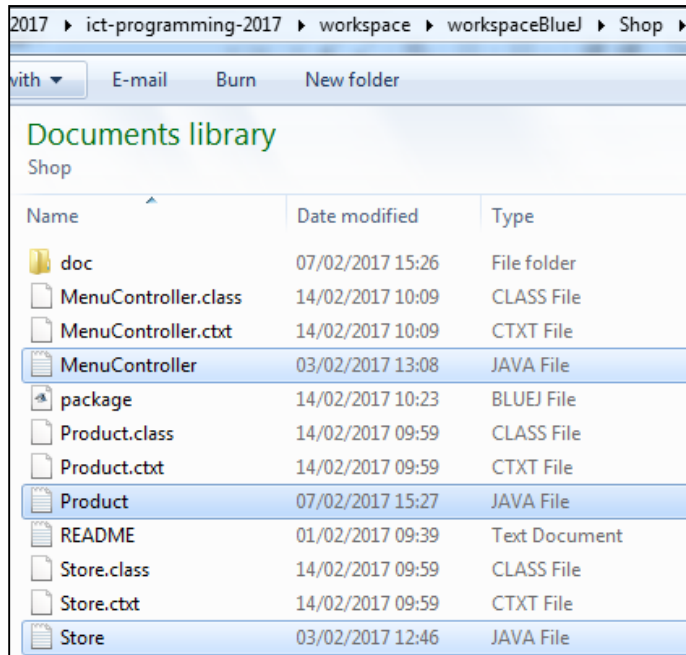
In Explorer, open the folder where your new **Eclipse** ShopV2.0 is stored:



We are going to copy the *.java files from the BlueJ project to the Eclipse project.

Let's move Shop V2.0 into Eclipse

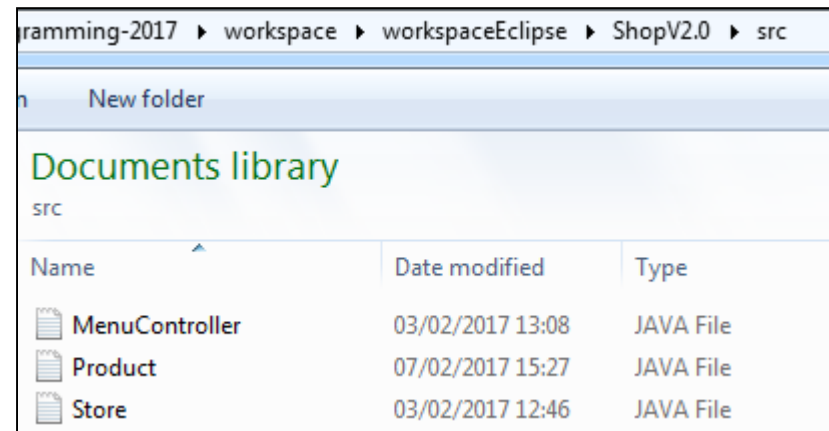
Existing **BlueJ** Shop project:



The screenshot shows a file explorer window with the path: 2017 > ict-programming-2017 > workspace > workspaceBlueJ > Shop. The file list is as follows:

Name	Date modified	Type
doc	07/02/2017 15:26	File folder
MenuController.class	14/02/2017 10:09	CLASS File
MenuController.ctxt	14/02/2017 10:09	CTXT File
MenuController	03/02/2017 13:08	JAVA File
package	14/02/2017 10:23	BLUEJ File
Product.class	14/02/2017 09:59	CLASS File
Product.ctxt	14/02/2017 09:59	CTXT File
Product	07/02/2017 15:27	JAVA File
README	01/02/2017 09:39	Text Document
Store.class	14/02/2017 09:59	CLASS File
Store.ctxt	14/02/2017 09:59	CTXT File
Store	03/02/2017 12:46	JAVA File

New **Eclipse** ShopV2.0 project with the three Java files:



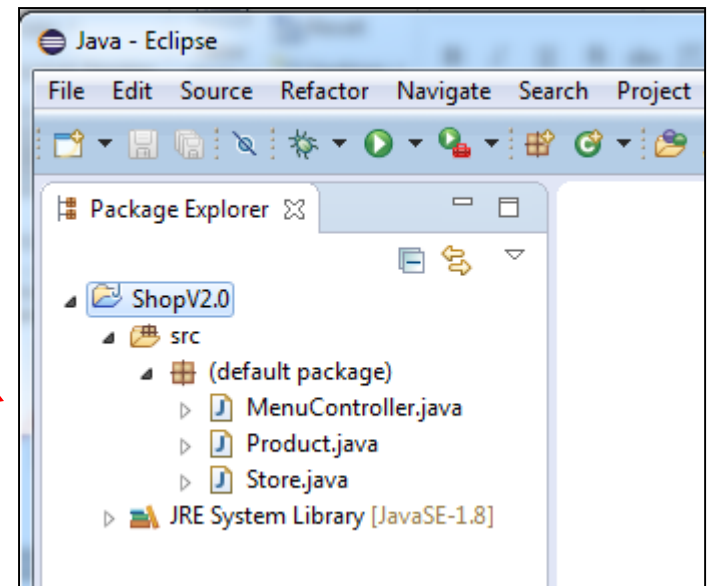
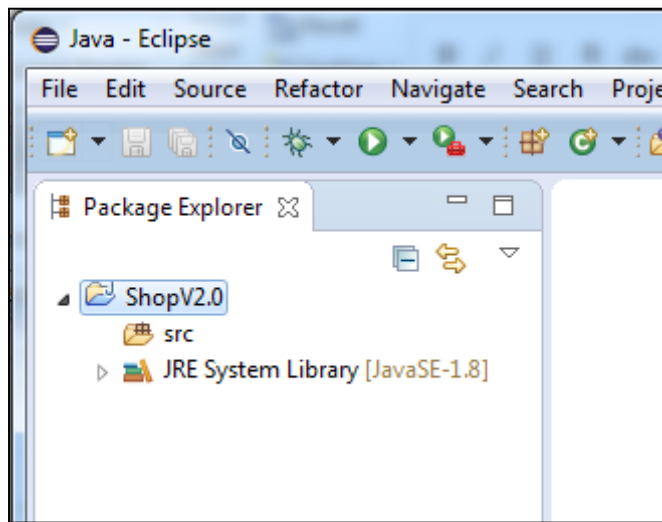
The screenshot shows a file explorer window with the path: ict-programming-2017 > workspace > workspaceEclipse > ShopV2.0 > src. The file list is as follows:

Name	Date modified	Type
MenuController	03/02/2017 13:08	JAVA File
Product	07/02/2017 15:27	JAVA File
Store	03/02/2017 12:46	JAVA File

Copy the *.java files from the BlueJ project and paste them into the /src folder in the new Eclipse ShopV2.0 project.

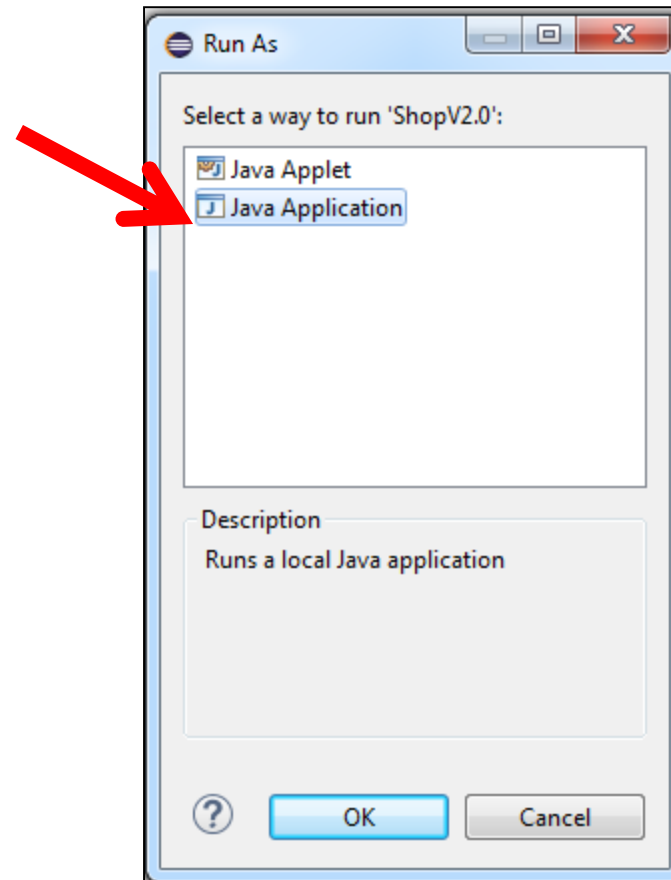
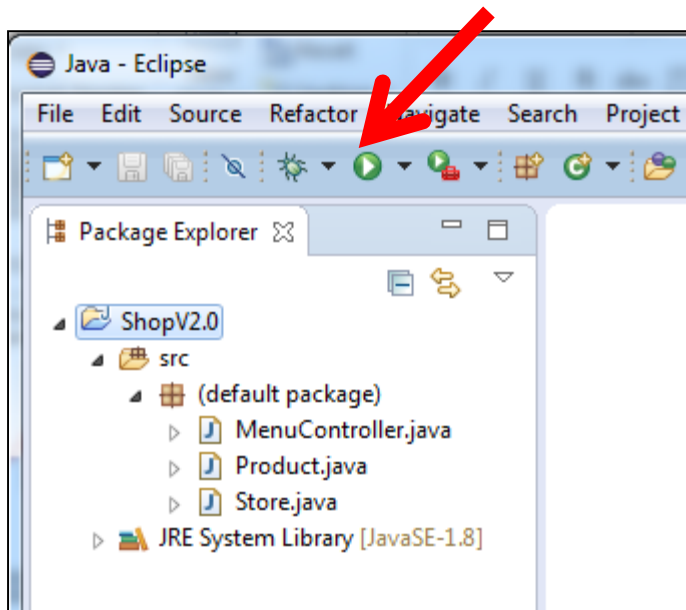
Let's move Shop V2.0 into Eclipse

- Return to Eclipse and refresh the project (F5)



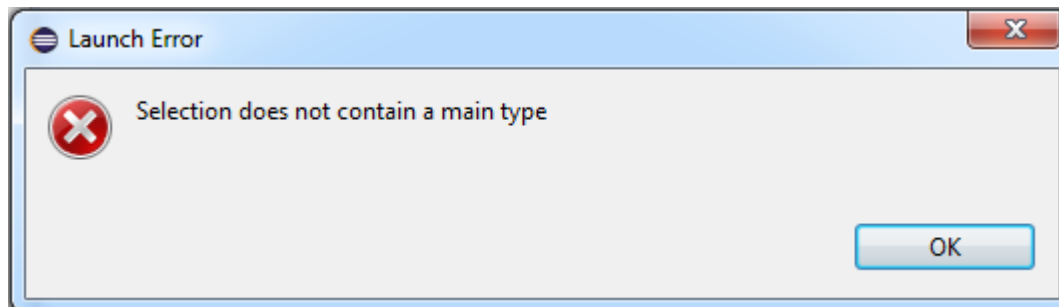
Let's move Shop V2.0 into Eclipse

- To run the application, click the run button and select Java Application

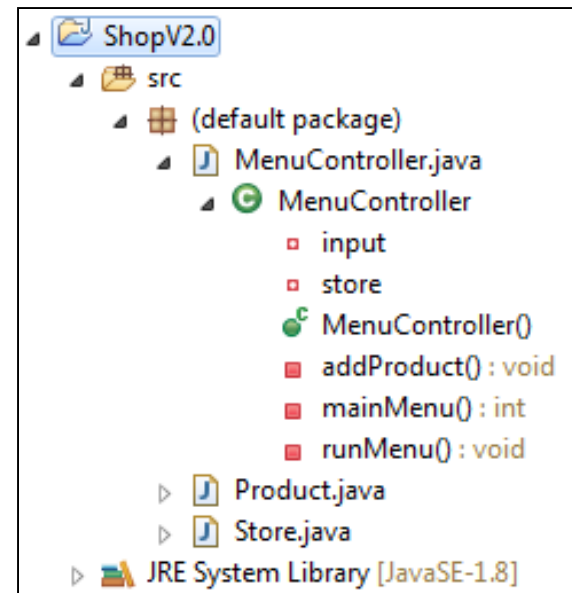


Let's move Shop V2.0 into Eclipse

- You will get the following error:



- We haven't included a **main** method in our MenuController class; the JVM needs this method to start an application!



Let's move Shop V2.0 into Eclipse

- Add the following **main** method into MenuController and run the Application again:

```
public class MenuController{


    private Scanner input;
    private Store store;

    public static void main(String[] args){
        new MenuController();
    }

    public MenuController(){
        input = new Scanner(System.in);

        //read in the details....
        System.out.print("Please enter the store location: ");
        String location = input.nextLine();

        store = new Store(location);
        runMenu();
    }
}
```





Package Explorer

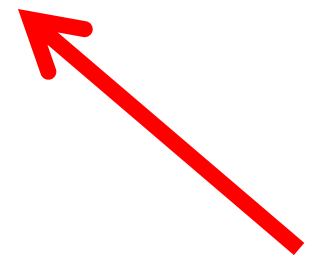
- ShopV2.0
 - src
 - (default package)
 - MenuController.java
 - MenuController
 - main(String[]): void
 - input
 - store
 - MenuController()
 - addProduct(): void
 - mainMenu(): int
 - runMenu(): void
 - Product.java
 - Store.java
- JRE System Library [JavaSE-1.8]

```
MenuController.java
6  * @author >loppan wronan
7  */
8  public class MenuController{
9
10     private Scanner input;
11     private Store store;
12
13     public static void main(String[] args){
14         new MenuController();
15     }
16
17     public MenuController(){
18         input = new Scanner(System.in);
19
20         //read in the details....
```

Outline

- MenuController
 - input: Scanner
 - store: Store
 - main(String[]): void
 - MenuController()
 - mainMenu(): int
 - runMenu(): void
 - addProduct(): void

MenuController [Java Application] C:\Program Files\Java\jre1.8.0_60\bin\javaw.exe (14 Feb 2017, 10:48:42)
Please enter the store location: |





Package Explorer

- ShopV2.0
 - src
 - (default package)
 - MenuController.java
 - MenuController
 - main(String[]): void
 - input
 - store
 - MenuController()
 - addProduct(): void
 - mainMenu(): int
 - runMenu(): void
 - Product.java
 - Store.java
- JRE System Library [JavaSE-1.8]

```

6  * @author >iooan uronan
7  */
8  public class MenuController{
9
10     private Scanner input;
11     private Store store;
12
13     public static void main(String[] args){
14         new MenuController();
15     }
16
17     public MenuController(){
18         input = new Scanner(System.in);
19
20         //read in the details....

```

Outline

- MenuController
 - input: Scanner
 - store: Store
 - main(String[]): void
 - MenuController()
 - mainMenu(): int
 - runMenu(): void
 - addProduct(): void

MenuController [Java Application] C:\Program Files\Java\jre1.8.0_60\bin\javaw.exe (14 Feb 2017, 10:48:42)

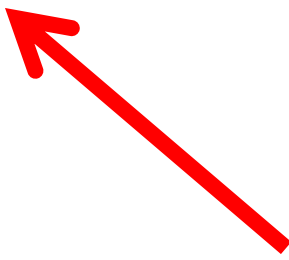
Please enter the store location: **Waterford**

Shop Menu

- 1) Add a Product
- 2) Remove Product (by index)
- 3) List the Products
- 4) List the cheapest product

- 5) View store details
- 0) Exit

====>



Questions?





Except where otherwise noted, this content is licensed under a Creative Commons Attribution-NonCommercial 3.0 License.

For more information, please see <http://creativecommons.org/licenses/by-nc/3.0/>



Waterford Institute of Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Department of Computing and Mathematics
<http://www.wit.ie/>