# UML Getting Started - UML Modeling in Eclipse

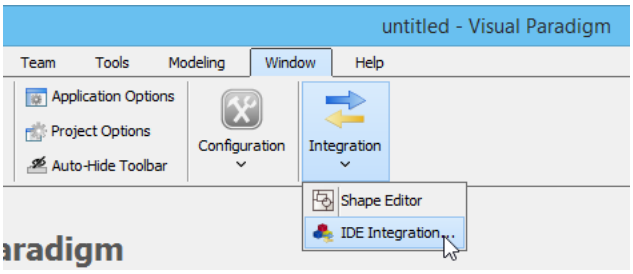
Written Date : March 03, 2016

## Preparation

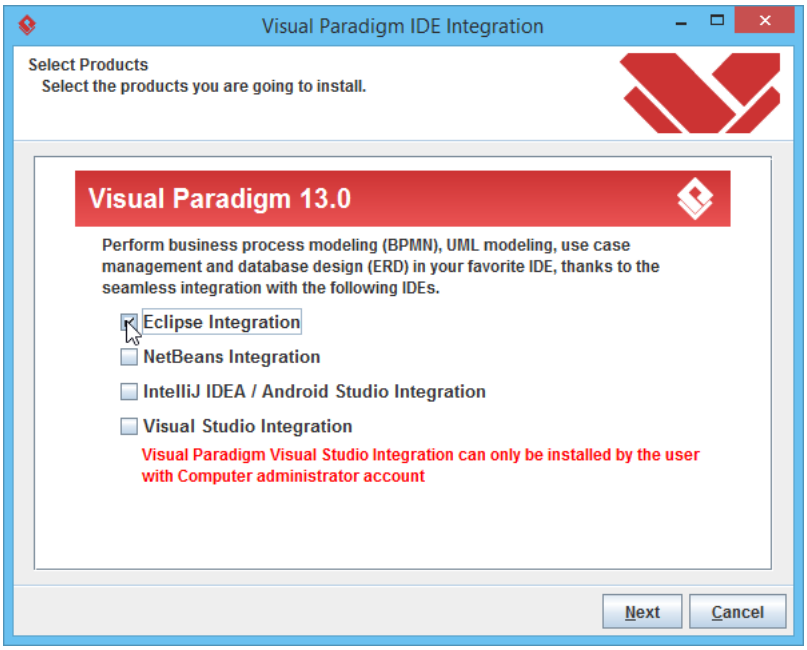
To follow this tutorial, you must have Visual Paradigm installed, which can be downloaded from the Visual Paradigm download page. You also need the Eclipse IDE. We assume you have already installed it, but if you haven't, please download and install it from www.eclipse.org. Additionally, ensure you have installed the Eclipse integration from Visual Paradigm. Finally, to make the tutorial easier to follow, we will not describe every minor step required to draw a class diagram in detail. We assume that you have the basic skills required to draw a UML class diagram in Visual Paradigm.

## Installing the Visual Paradigm Eclipse Integration

1. In Visual Paradigm, select **Window > Integration > IDE Integration...** from the application toolbar.

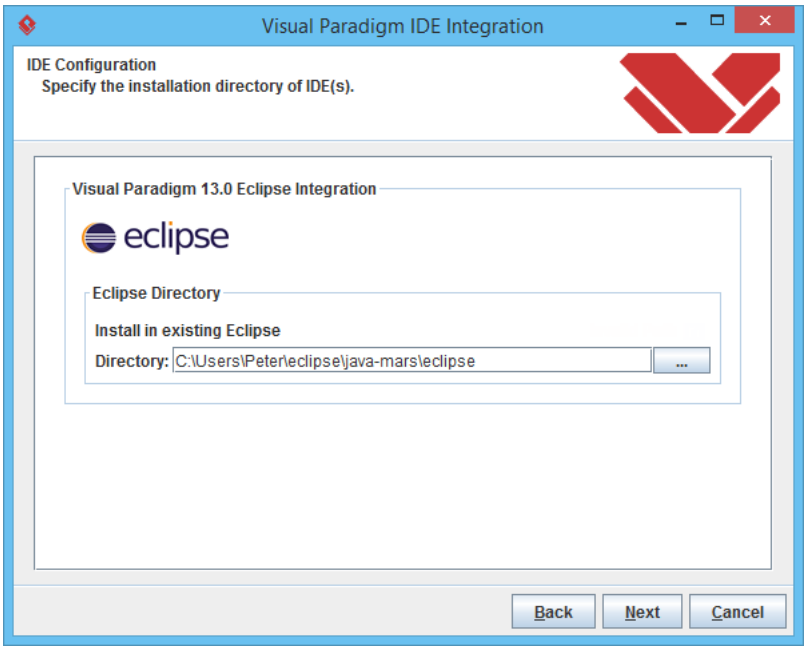


2. In the **Visual Paradigm IDE Integration** window, check **Eclipse Integration**.




3. Click **Next**.

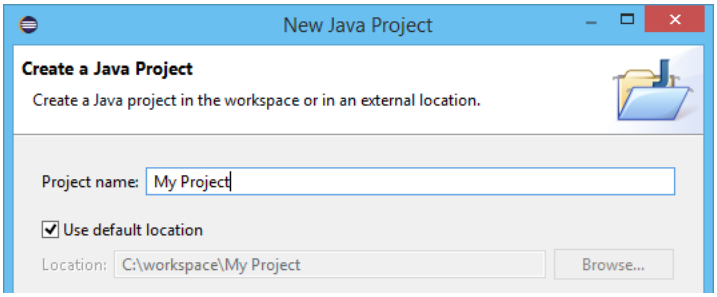
4. Enter the path of your Eclipse installation and click **Next**.




This begins copying files. If you see the error message "java.io.IOException: Cannot make dirs for file...", please restart Visual Paradigm with the **Run as Administrator** option. When file copying is finished, close Visual Paradigm and move on to the next section to see how to create a Java project in Eclipse along with a UML model.

## Creating a Java Project

1. Start Eclipse.
2. Select **File > New > Java Project** from the main menu to open the **New Java Project** window.
3. In the **New Java Project** window, enter *My Project* in the **Project name** field.

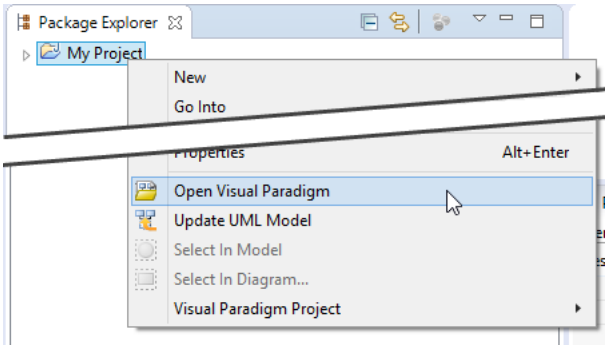



4. Click **Finish**.

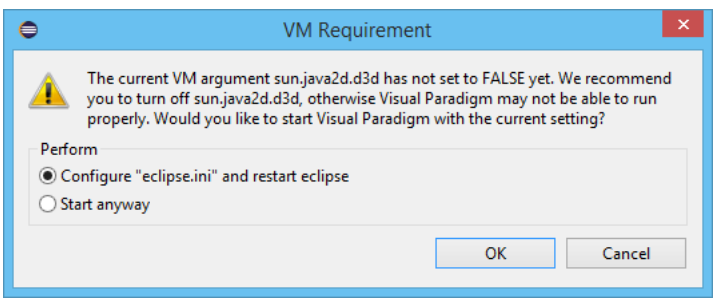
## Creating a UML Model for the Java Project

Now that you have an empty Java project, let's create a UML model for it. To create a UML model:

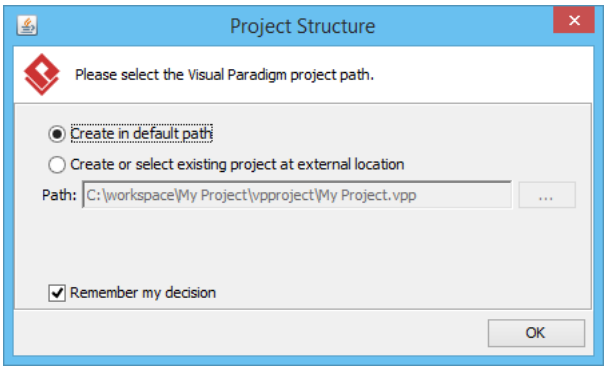
1. Right-click on the project node in **Package Explorer** and select **Open Visual Paradigm** from the popup menu.




2. If you see the **VM Requirement** dialog box, keep the option **Configure "eclipse.ini" and restart eclipse** selected and click **OK** to restart Eclipse. Then, repeat the previous step to open Visual Paradigm.



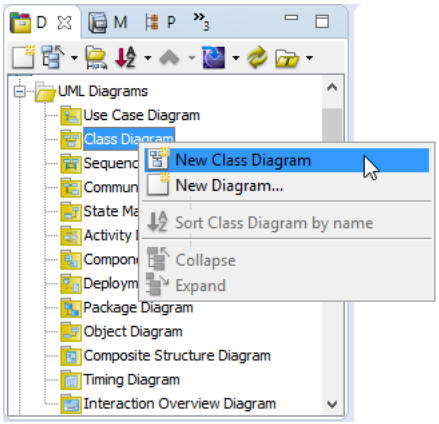
3. Click **OK** when you are prompted to select a path to store the .vpp project file.



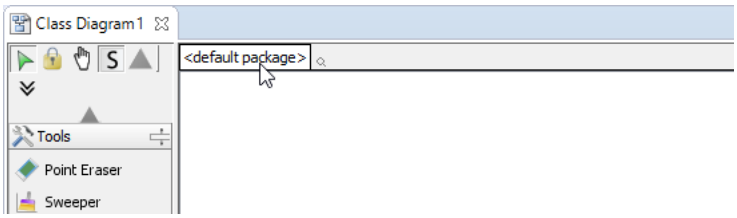

## UML Modeling in Eclipse

Let's draw a simple class diagram. We will generate Java code from it in the next section.

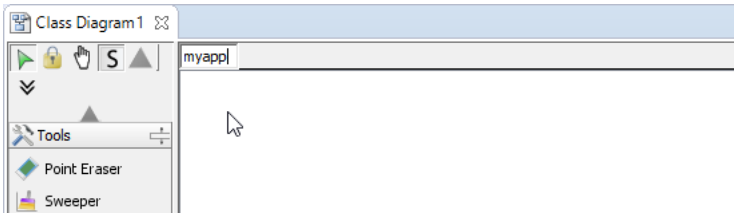
1. In **Diagram Navigator**, right-click on the **Class Diagram** node and select **New Class Diagram** from the popup menu.



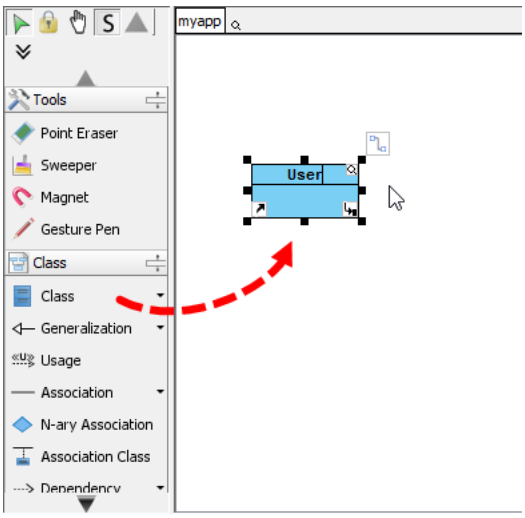

2. A new diagram is created. Double-click on the package header at the top left corner of the diagram, labeled **<default package>**.



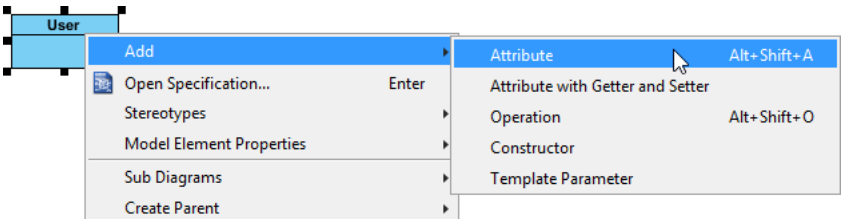
3. Enter *myapp* and press **Enter**. From now on, classes drawn in this diagram will be placed in a new package named *myapp*. In practice, you can also enter a nested package structure like *com.vp.myapp*.




4. Create a class by selecting **Class** from the diagram toolbar. Drag and drop it onto the diagram. Enter *User* as the name and press **Enter** to confirm.

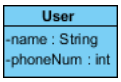



5. A user has two attributes: name and phone number. Let's add them. Right-click on the *User* class and select **Add > Attribute** from the popup menu.

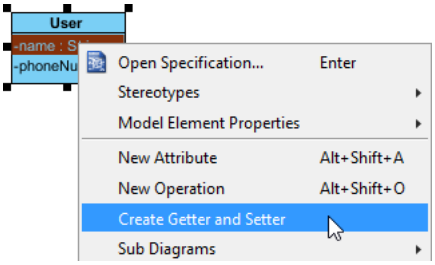



6. Enter *name : String* to create the *name* attribute with a String type. Then press **Enter** to confirm.

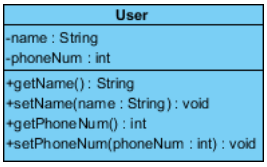
7. Similarly, enter *phoneNum : int* to create the *phoneNum* attribute with an int type. Press **Enter** to confirm, and then press **Esc** to finish adding attributes.




8. We want Visual Paradigm to generate getters and setters for the attributes during code generation. Right-click on the *name* attribute and select **Create Getter and Setter** from the popup menu.

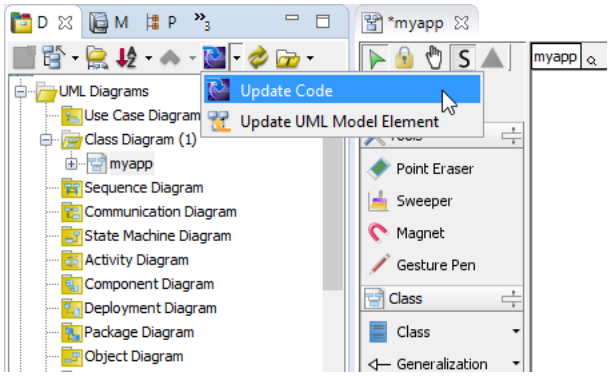



9. Repeat the previous step for the *phoneNum* attribute. At this point, your class diagram should look like this:

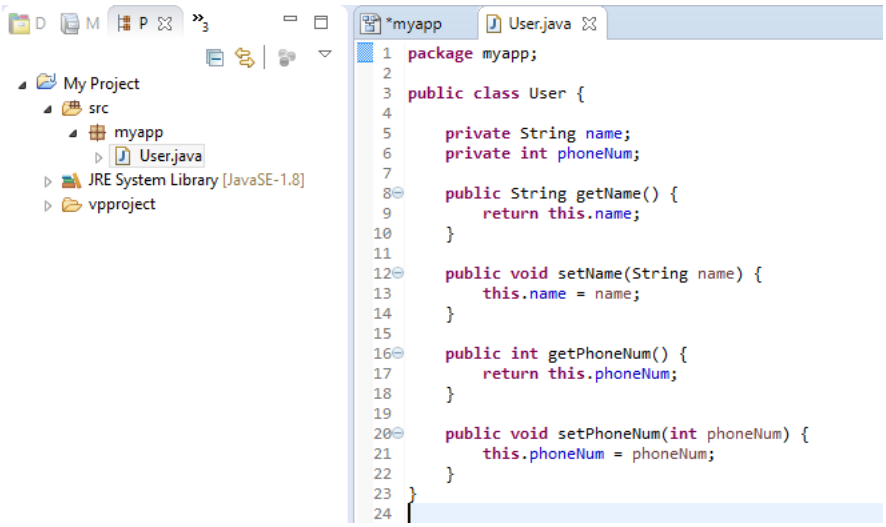



## Generating Java Code from a UML Class

Let's produce Java source code from the UML class. There are several ways to achieve this. Here, we will try the one that generates code for the entire UML model. Click the **Update Code** button at the top of the **Diagram Navigator**.




In the **Package Explorer**, expand the project node and the src folder node. The *myapp* package and the *User* class are there. Open *User.java*. You can see the *User* class populated with attributes and their getters and setters.

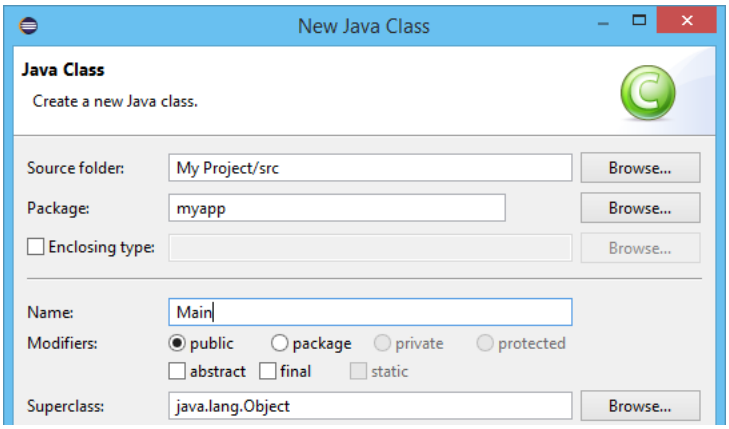



## Writing Code

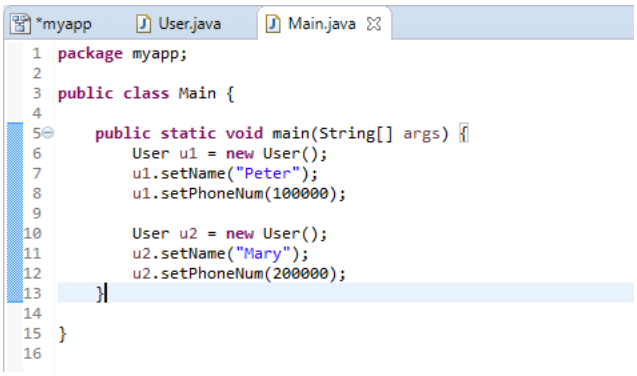
In this section, you are going to build an executable application with the *User* class.

1. Create a *Main* class. In the **Package Explorer**, right-click on the *myapp* package and select **New > Class** from the popup menu.

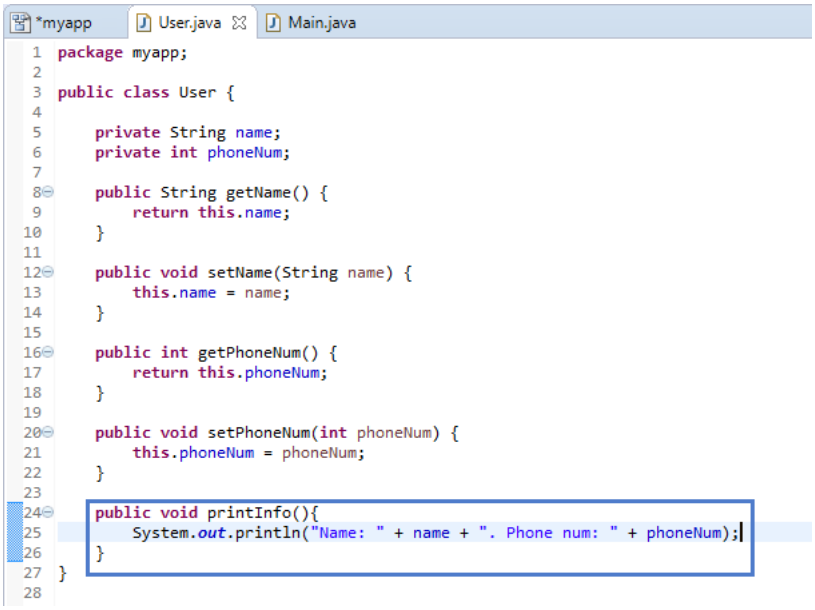
2. In the **New Java Class** window, enter *Main* as the class name and click **Finish**.



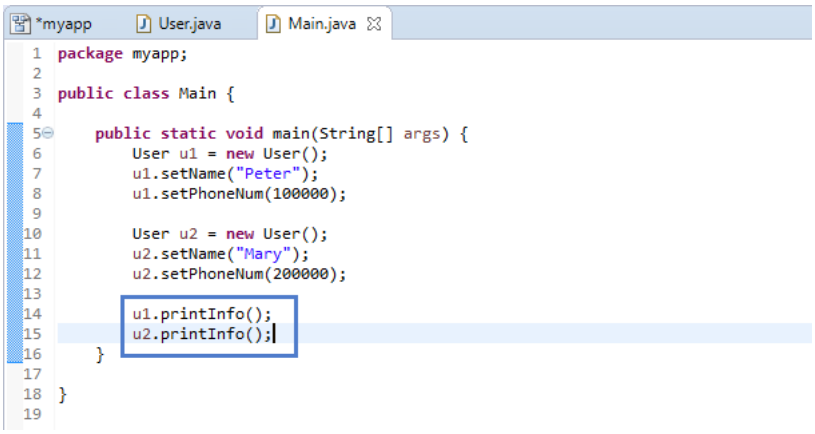
3. Create a static main method in the *Main* class that creates and instantiates two *User* objects. Set their name and phone number through the setters.




4. It would be nice to add a method to the *User* class to print its name and phone number. Add a public method *printInfo()* to the *User* class.



5. Call *printInfo()* from the *Main* class to display the information of the created users.

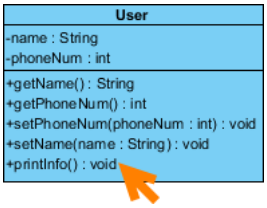



## Updating the UML Model from Java Code

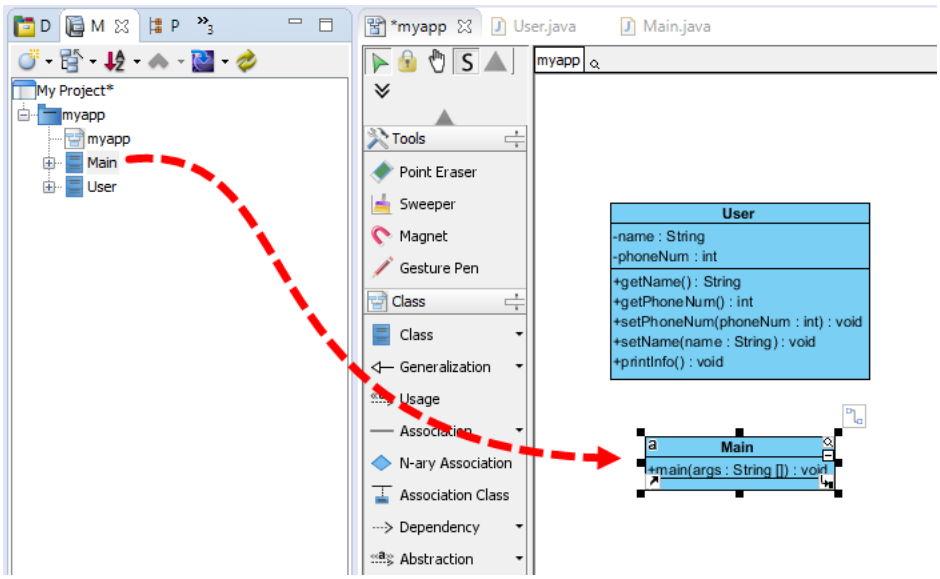
You have made some changes in the source code, such as creating the *Main* class and the *printInfo()* method in the *User* class. To keep the design consistent with your source code, you need to update the UML model from the code. Let's do it now.

1. In the Eclipse toolbar, click the **Update UML Model** button.


2. Open the class diagram. The *printInfo()* method is now present in the *User* class.



3. You can find the *Main* class under the **Model Explorer**. Drag it out and place it below the *User* class on the diagram.




Related Links

- Tutorial - Hibernate in Eclipse
- Tutorial - C# Round-trip engineering
- Tutorial - Control Shape Appearance with Stereotype
- Full set of UML tools and UML diagrams

Visual Paradigm home page
(https://www.visual-paradigm.com/)

Visual Paradigm tutorials
(https://www.visual-paradigm.com/tutorials/)