FAQs about Visual Paradigm for UML Community Edition

**Visual Paradigm for UML** is the software we will use to draw UML class diagram, and more specifically, we will use the Community Edition, which is free for academic usage. UML stands for Universal Modeling Language.

**Question A:** where to download Visual Paradigm (VP) For UML Community Edition?

This is the download link: <https://www.visual-paradigm.com/download/community.jsp> , and you need to select the correct distribution for your OS: Windows, Linux, or Mac OS X.

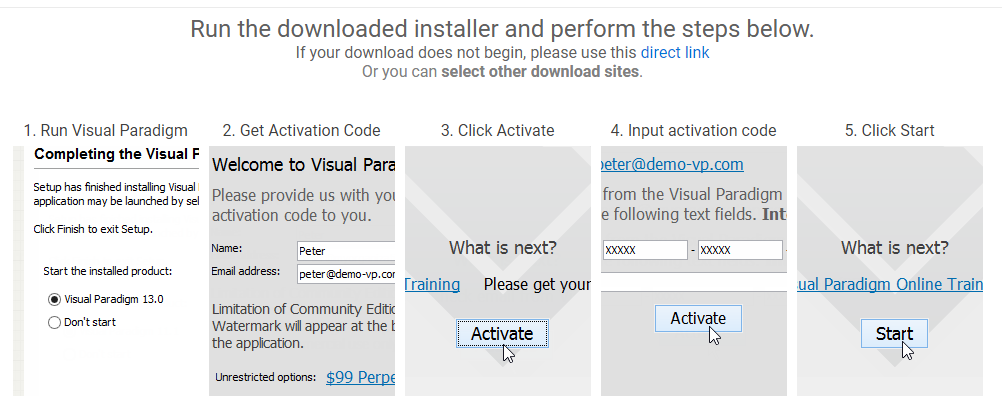
Notice: we use the **Community Edition** (it is free), not the **free trial of Enterprise Edition**. The reason is that the **free trial of Enterprise Edition** has watermark, which does not look nice.

After you download “Visual Paradigm for UML Community Edition”, you need to click the executable file to install the software in your machine.

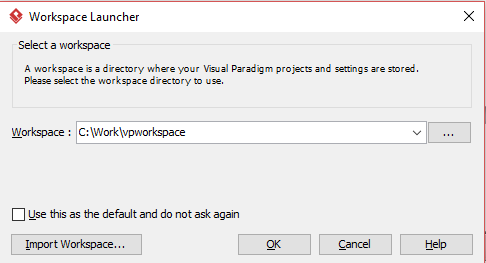
**Question B:** how to obtain the activation key for ***VP for UML Community Edition***?

The VP-UML can run successfully only after you supply a valid activation code.

* After you finish the installation of VP for UML Community Edition, a window will pop up, asking for your name and email address.
* Input your name and email address, and click the button “Get Activation Code” , and then in the next pop up page, click “Activate”.
* Check your email account and locate the activation code sent to you.
* Copy and paste the activation code, and then clicks the Activate button, then in the next page click “Start”. The process are illustrated by the pictures below.

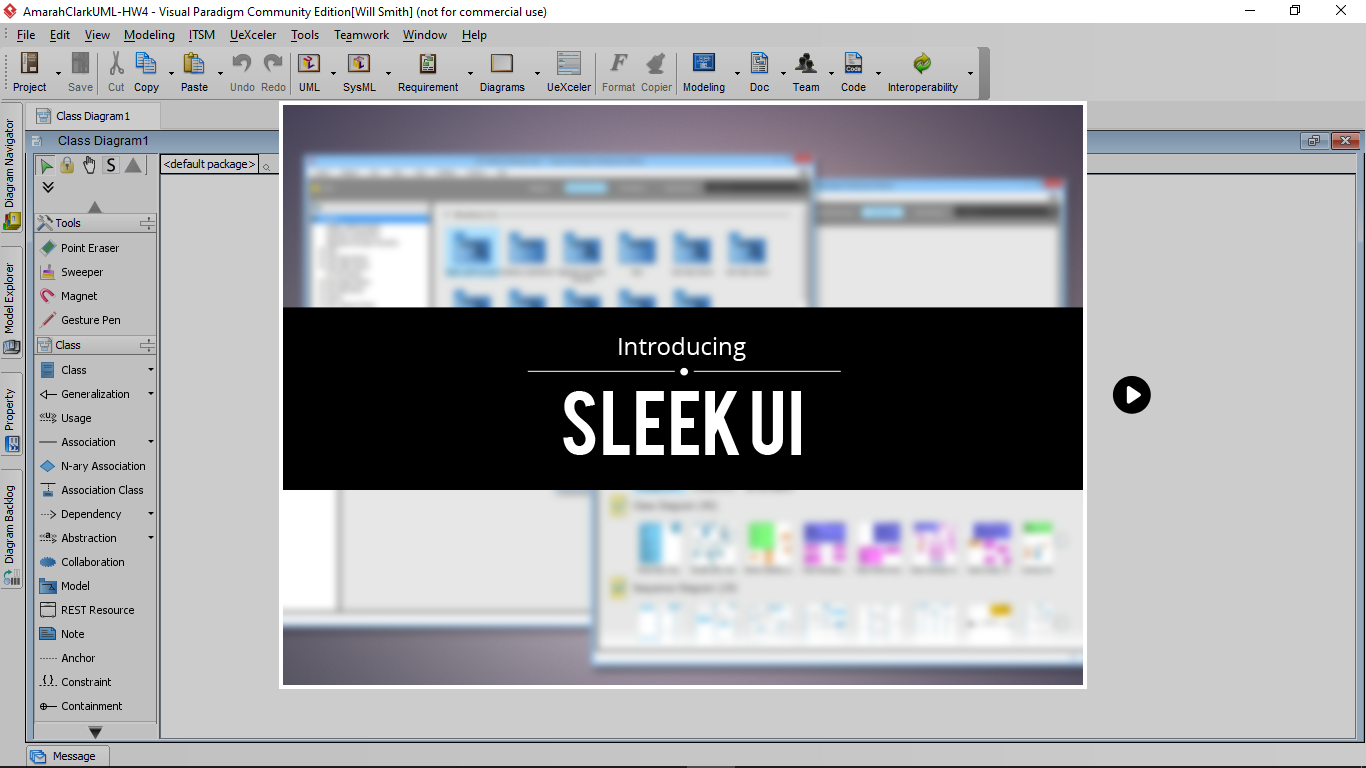


The first time when you launch the Visual Paradigm, it will ask you for the directory of your workspace, where all your UML drawings will be saved to.

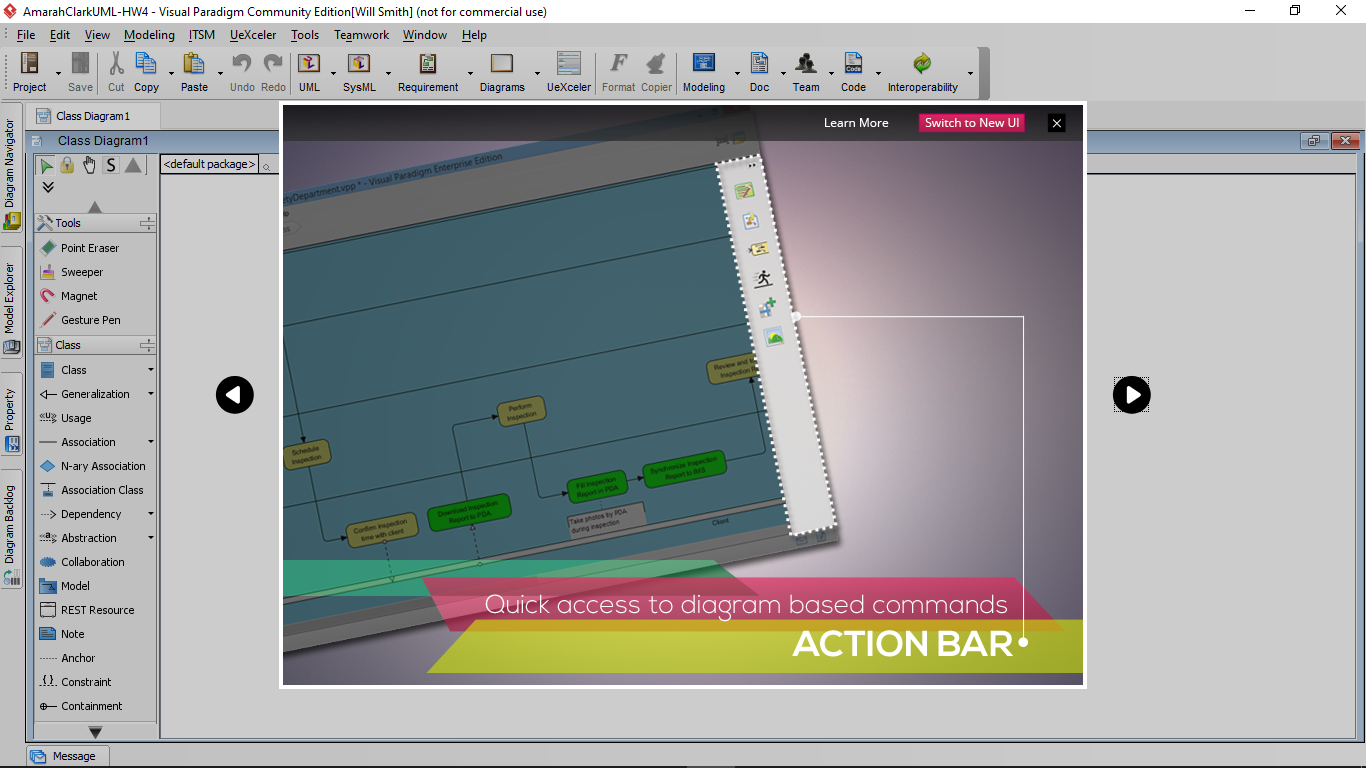


Just specify your preferred folder as the Workspace in the above windows, then click OK.

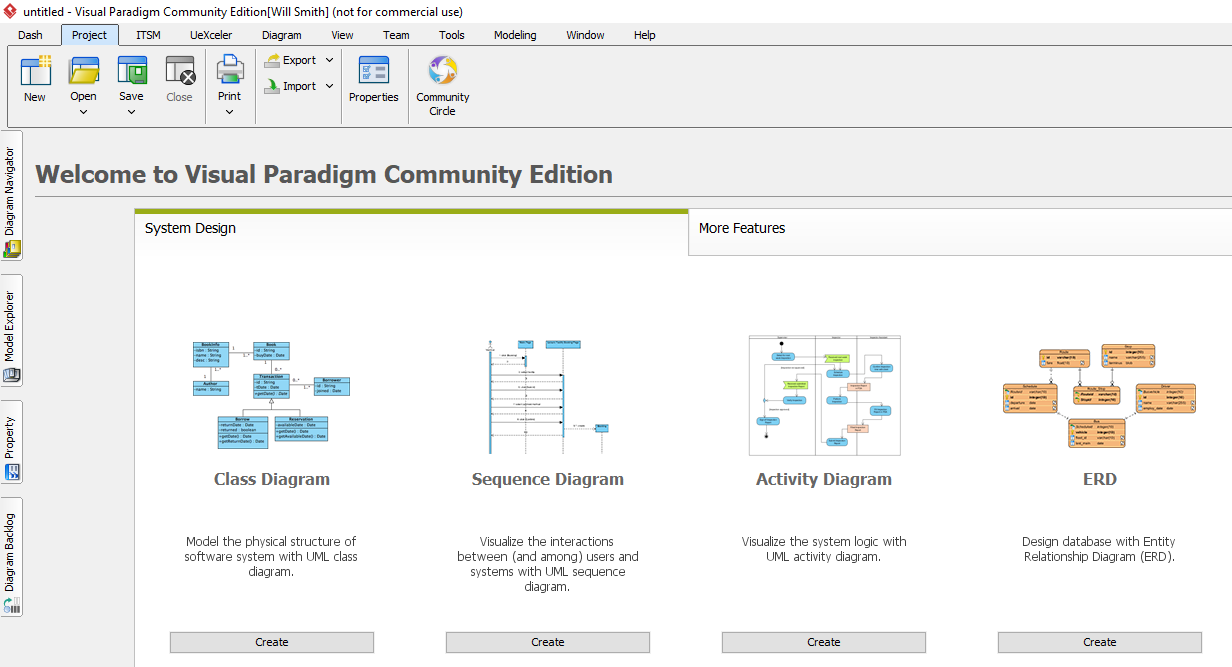
The first time it starts, VP may ask you whether you want to tour the new features of this version in a popup window, just close the popup window for now. And this screen shows up:



Then you clcik the next button (the play button on the right), it will becomes like this:



Then you can see the window close check x mark on the upper right corner. Click it and close the demo window. Now you should see this screen:



And then click the shortcut “Create” button under “Class Diagram”, you can start drawing UML class diagram.

Question C: how to create UML class diagram using VP for UML Community Edition?

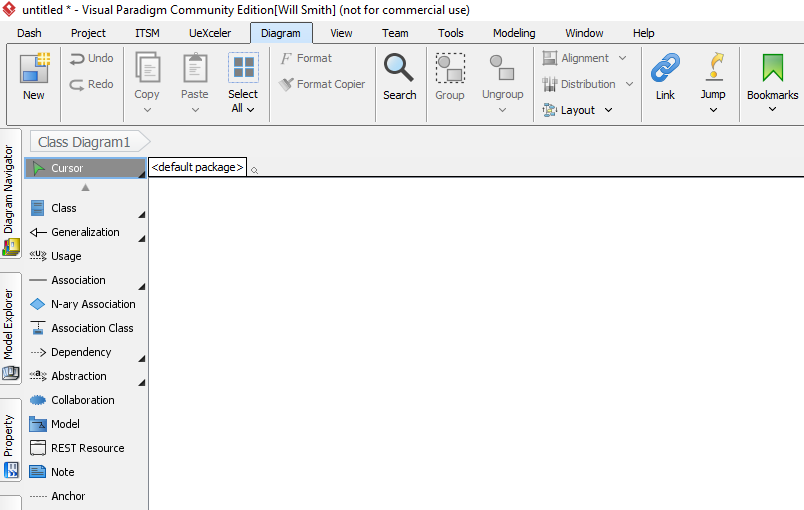
Below is an online tutorial on how to draw UML class diagram using VP:

<https://www.visual-paradigm.com/support/documents/vpuserguide/94/2576/7190_drawingclass.html>

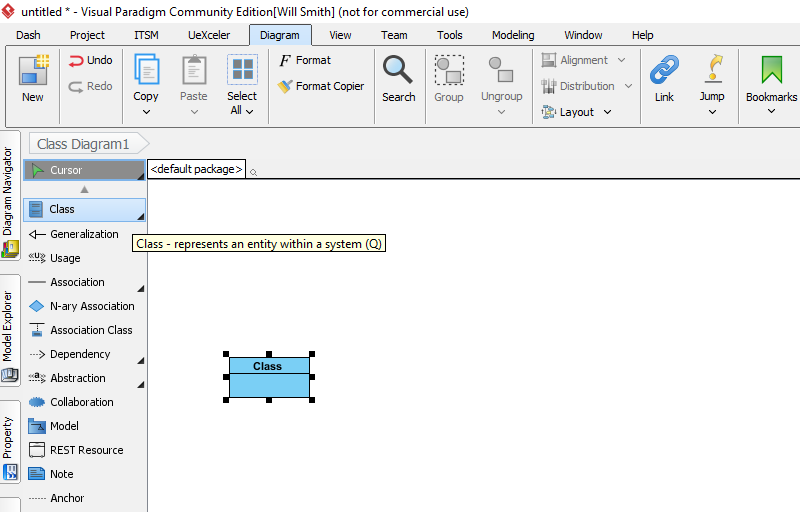
You only need to study section 2.1 “Creating class diagrams” in the above link.

The following is a step-by-step example on how to create the UML class diagram for class BookOrder. File BookOrder.java is provided in Moodle.

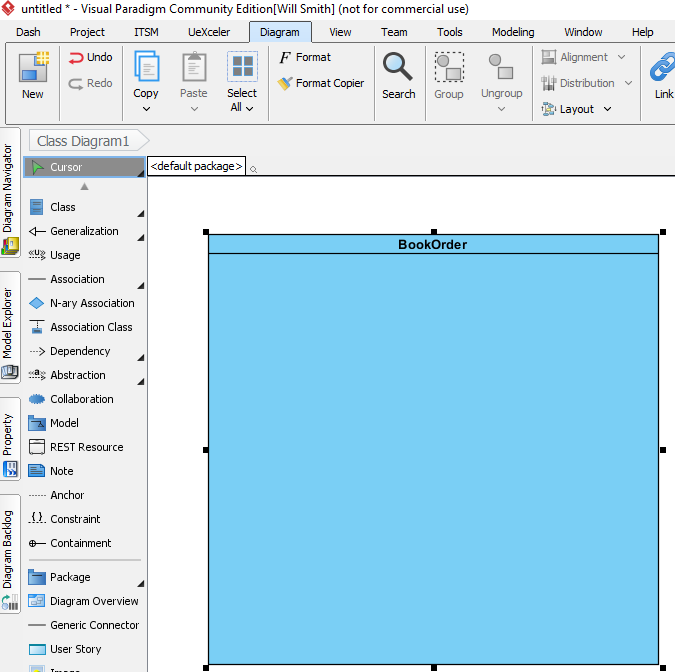
1, after launching Visual Paradigm for UML Community Edition, and supplying a folder as workspace to store VP project files, click “Create” button under “Class Diagram”, and you will see this:



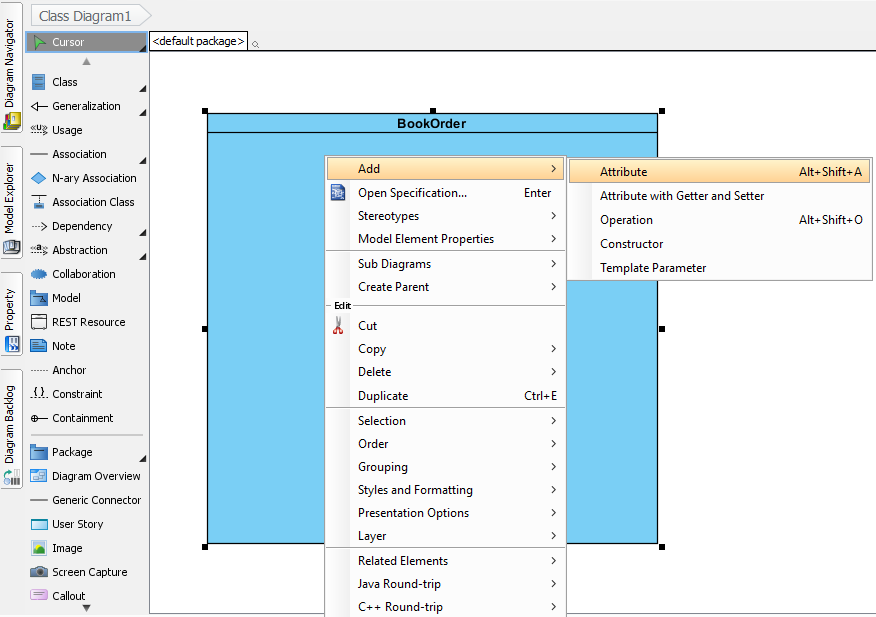
2, in the left panel, you can see icon “Class”. Click this icon and then move your mouse to the empty background, and click the background again, then you will see a new class diagram has been placed there.



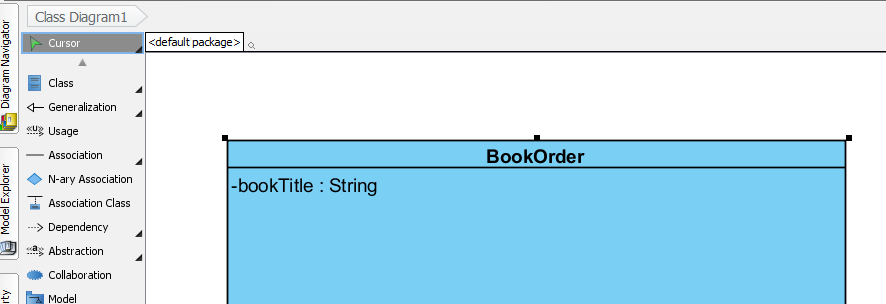
3, type in the class name in the class box: BookOrder , and then drag the rectangle box of class BookOrder and enlarge its size, so that we can place all the private data and public method into this box (the UML class diagram).



5, add an instance variable *bookTitle* of String type, by right-clicking anywhere inside the class rectangle (the blue area), and in the context menu, select Add->Attribute.



, then type in the variable name and type as bookTitle:String , with : separating the variable name and variable type.

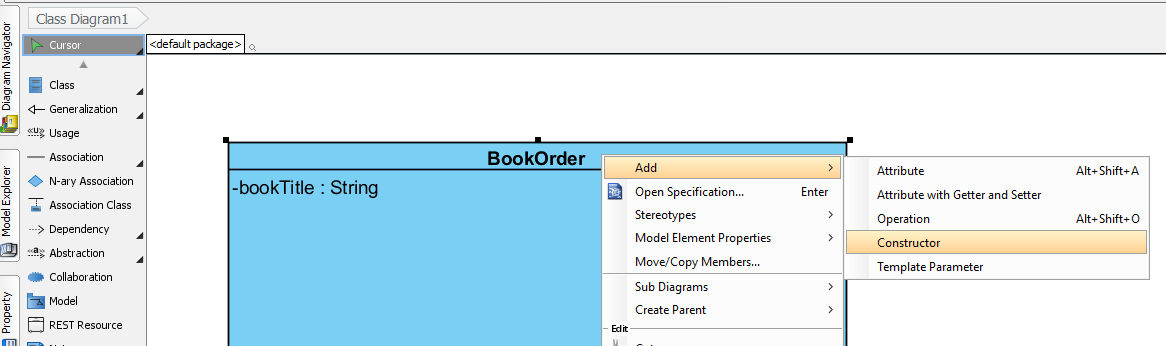


Notice that the minus sign – in front of the variable name, which is added automatically by Visual Paradigm, because by default, VP assumes that the variable is a private variable.

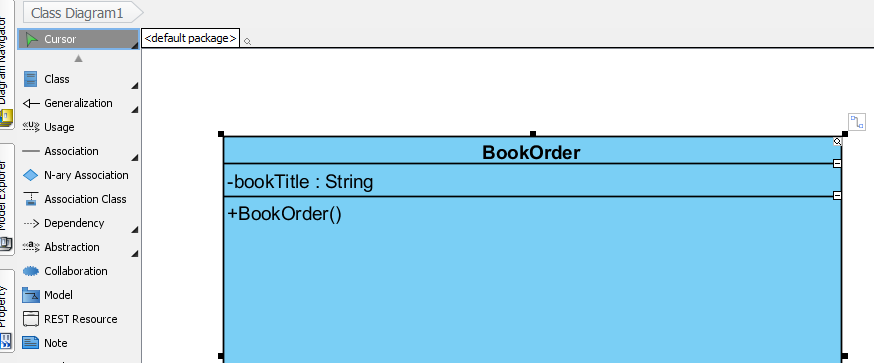
Then click anywhere else inside the class diagram, to finish the first private data member.

If at any point of the UML class creating process, you have an item in the class body that you don’t want, such as a data member or a method member, you just right click the item, and then select “delete” to remove the unwanted item.

6, add a constructor by right-clicking the class diagram body, and in the context menu, select Add->Constructor

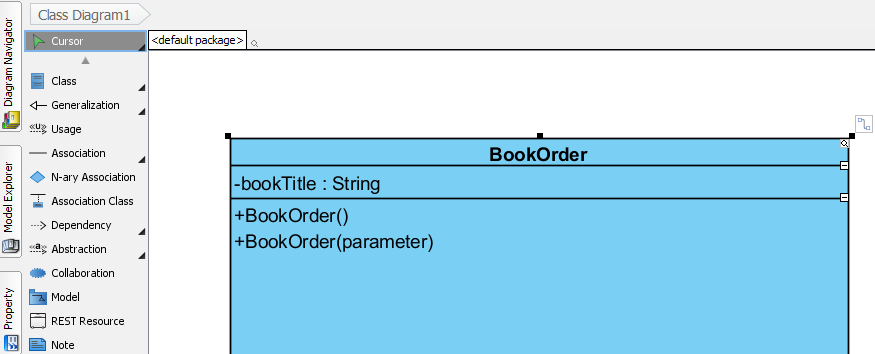


, then the default constructor will be added, as indicated below:

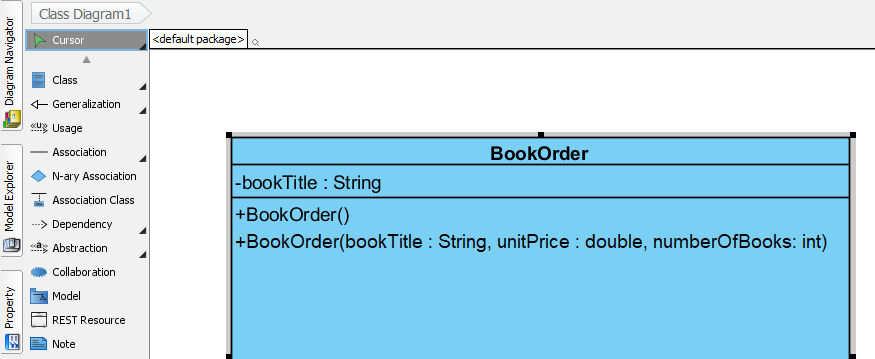


Notice that the plus sign + in front of the constructor indicates that, the method is public by default. Click anywhere else inside the class diagram to accept the default constructor.

Then right-click the class diagram body, and in the context menu, select Add->Constructor, and this time, the string “parameter” shows up in the parenthesis of method BookOrder.



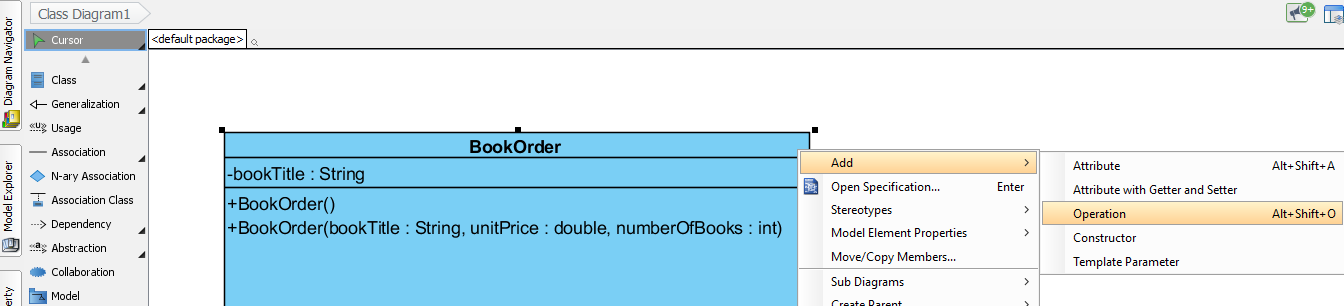
Then type in the list of formal parameters for the overloaded constructor, as shown below:



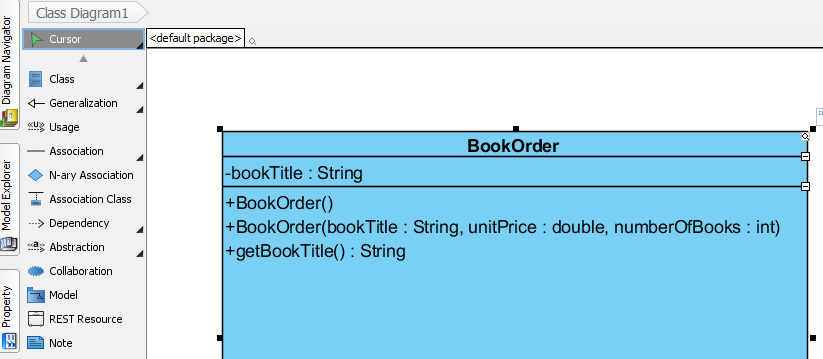
Notice that the parameters are separated by comma, and each parameter follows the syntax of variableName : variableType .

You can always double click any data or method item inside the class diagram, and edit the item.

7, add the getter method getBookTitle(), by right click the class body, and in the context menu, select Add->Operation , as below:

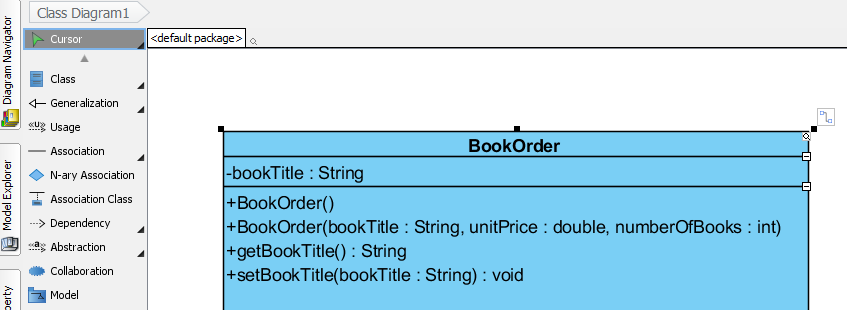


, then type in the correct syntax for the getter, as below:



, and notice that this getter has a return type String. In comparison, the constructors do not have return type, not even void.

8, similarly, we can add the setter method setBookTitle(),by right clicking the class body, and in the context menu, select Add->Operation , and then type in the setter, as below:



, and notice that, the setter method has formal parameter, and its return type is void. In comparison, the constructors do not have void or any other return type.

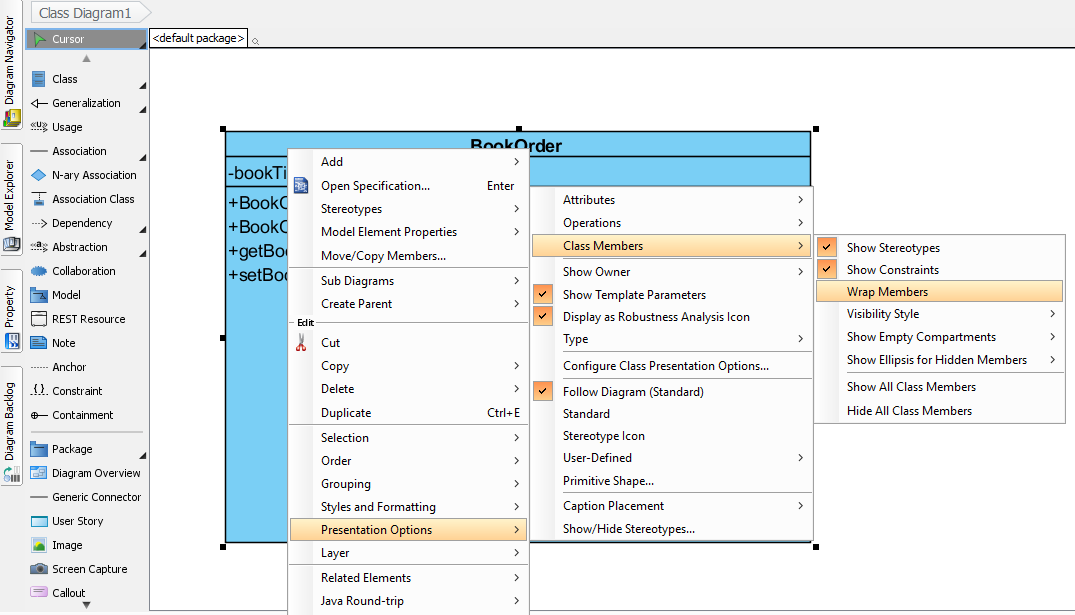
9, repeat the above processes from step 5 to step 8, so that you can add the other private data members, and other public constructors/getters/setters/effectors into class diagram, until the UML class diagram is finished. Attention: when you finish, the class diagram should have all the private data members and all the public methods in class BookOrder.

10, if there is any unwanted item such as data member or method member in the class diagram, you can first select the unwanted item, then right click it, and then choose “delete” in the context menu, so that this unwanted item can be removed.

11, when you feel that the current class diagram size of the current class is not big enough, you can take one of the two actions below:

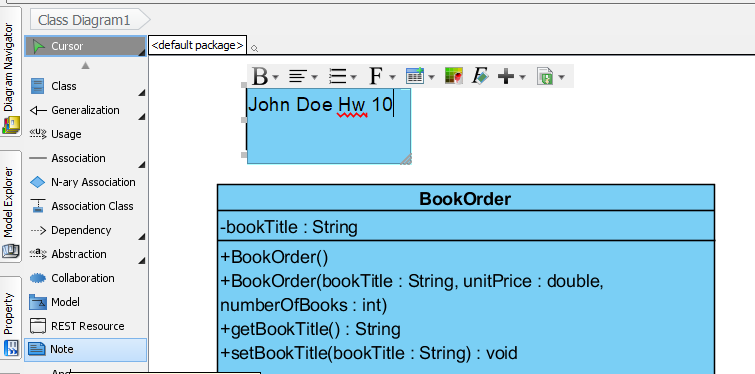
* expand the class diagram size by dragging the border of the class box to the right , or
* set the “wrap” property of class members, so that a long line can wrap instead of cut off, by right click inside the class body, and then select

“Presentation Options” -> “Class Members” -> “Wrap Members”

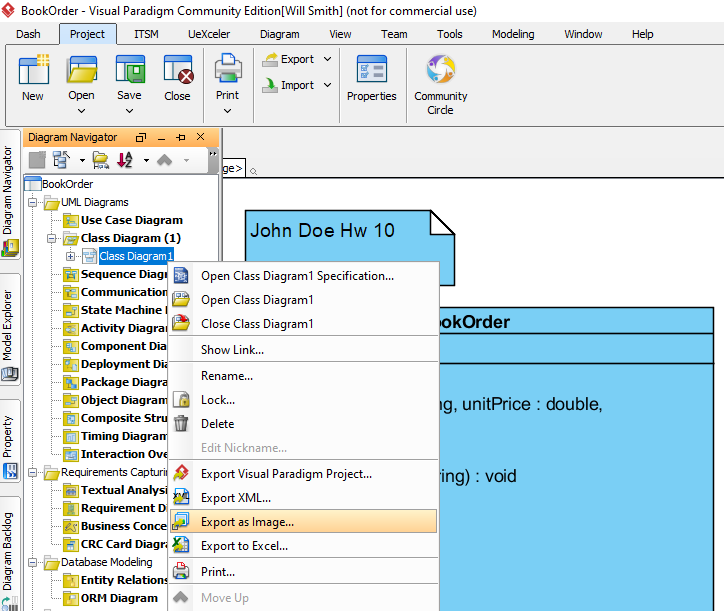


14, when you finish creating the class diagram, you may want to add a label above the class BookOrder, to indicate the author of this class diagram. For example, the label you add can be   
“John Doe Hw10, class BookOrder”, with JohnDoe being your first and last name.

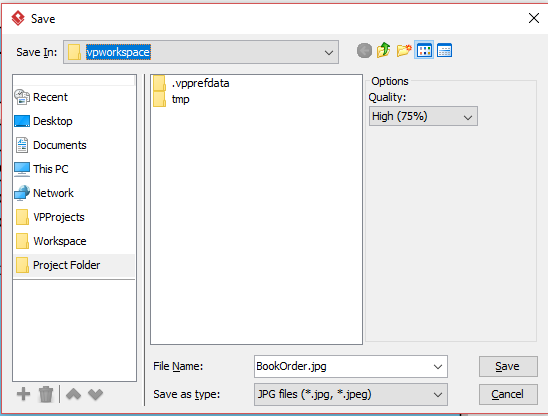
Click the Note item on the left panel, and then click an open area above the class BookOrder, and then a note is added, so that you can type in the notes.

.

15, after you draw all the private data and public methods and add the note, you need to save the project and export the diagram as an image file. To do this, you first need to click the save button to save the project, and then click the “Diagram Navigator”, which is the first vertical item on the left panel, and then right click the “Class Diagram 1” under “Class diagram”, and then select menu item: “Export as Image…”



, and you will see a pop-up window asking for the file name of the exported image:

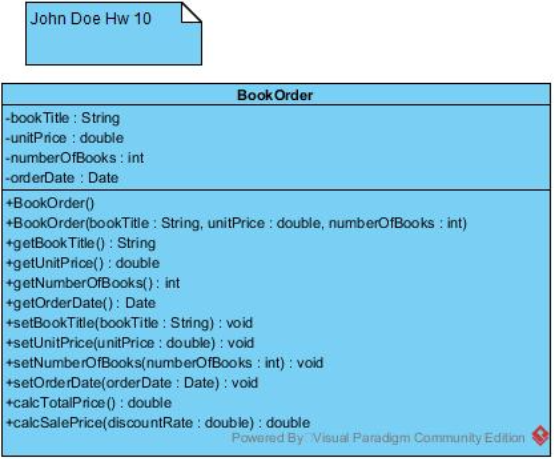


Then navigate to the correct folder, and then type in the correct file name, such as BookOrder.jpg, and click save. The file “BookOrder.jpg” is provided along with the java source code after you unzip “UML-classDiagram.zip”.

Now you can go to the folder where you save the UML class diagram as an image, and view the class diagram as an image file. In the pop-up window above, you can also change the “Save as type” of the exported file, and pdf file is another file format for exportation.

The reason why you want to export your UML class diagram to an image file is because it will be easier for other people to view the class diagram without having to install Visual Paradigm for UML on their computer.

Here is the sample picture of UML class diagram for BookOrder:



, and you can see that, the UML class diagram follows the same sequence of the java source code: **private data, public methods as a list of constructors, getters, setters, effectors.** When you draw your UML class diagram, please follow the same sequence as above,

16, the example above does not have a static variable. Now here is an example that has a static non-final variable, a static final variable, and a static method. You need to learn how to indicate static in UML class diagram.

public class Circle{

private double radius;

private static int totalCircles;

public static final double PI = 3.14;

public static int getTotalCircles(){

// the detail of this method is omitted here.

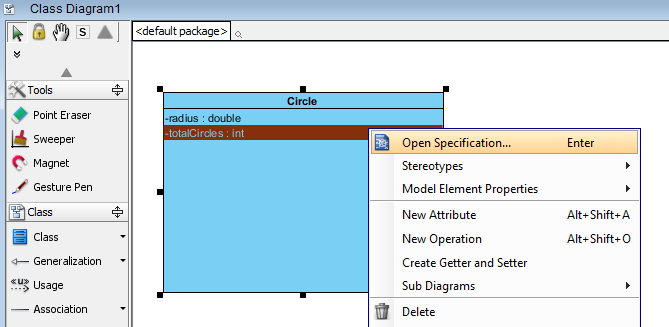
}

// other methods are omitted...

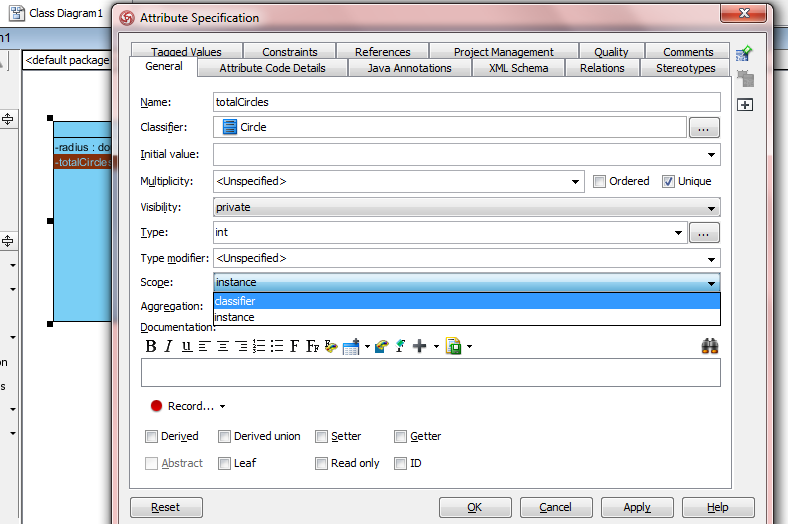
}

When we create the UML class diagram for the above Circle class, private data radius is a non-static variable, thus the process is the same as in example BookOrder class.

For private static data totalCircles, we first add this variable as an item in Circle class, then right click this item and select “Open Specification…”



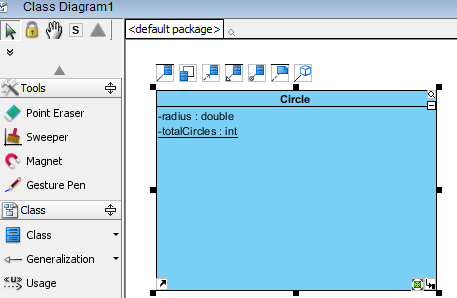
, then in the “Attribute Specification” windows, click the drop down box “Scope”, and change the scope from ***instance*** to ***classifier***, then the job is done!



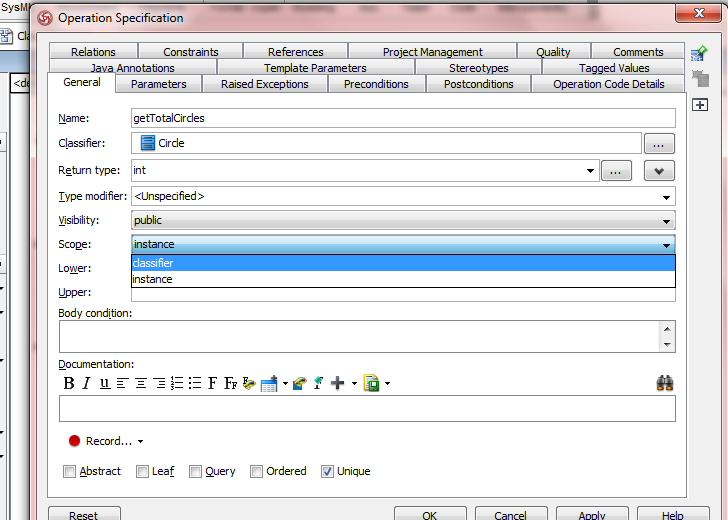
The instance scope is for non-static variable, because non-static variable is associated with each instance.

The class scope is for static variable, because static variable is not associated with a particular instance; instead, the static variable belongs to the entire class, thus referred to as class scope.

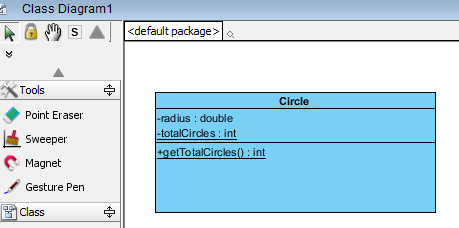
After that , click ok button to close the “Attribute Specification” window, and then you will see an underscore showing up for variable totalCircles. The underscore is the indication of static variable (and static method) in UML class diagram.



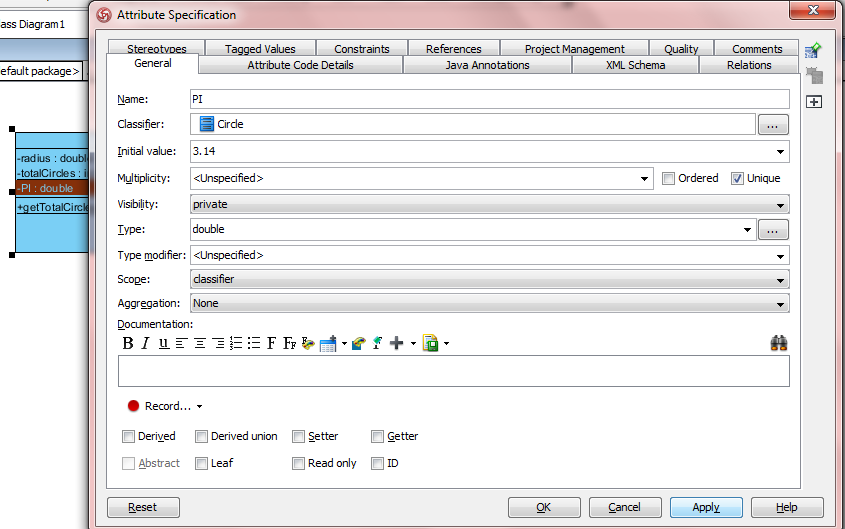
Do the same thing for method **public static int getTotalCircles()**. First we add this method into the class diagram, and then we right click it, select “Open Specification…”, and then in the “Operation Specification” pop-up window, set the Scope value as “classifier”.



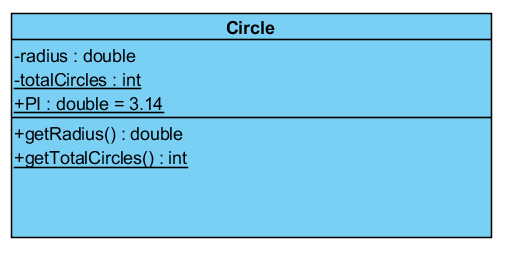
Then click OK to finish the setting for this static method. As you can see from the picture below, the method getTotalCircles also has an underscore, indicating this is a static method.



The next example is the public static final variable PI, and we need to set its scope as “classifier”, and then type in 3.14 in “Initial value” of the “Attribute Specification” window:



Then click OK button. Then double click the PI item to enter edit mode, and change the ***minus sign –*** that stands for ***private*** to ***plus sign +*** that stands for public, because PI is public:



Also added in the above class diagram is the public getter for private data member radius.

From the graph above, we can clear see the following of UML class diagram:

* the difference between static method and non-static method
* the difference between private static variable and private non-static variable
* the difference between private static variable and public static final variable