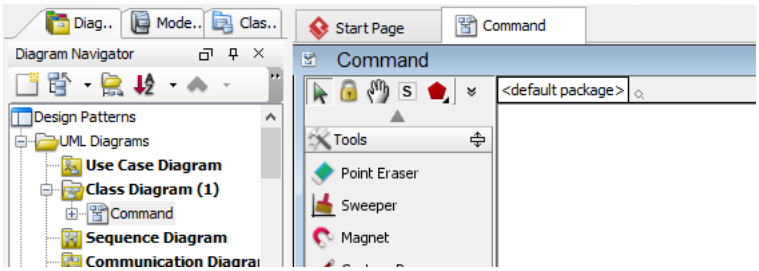
Command Pattern Tutorial

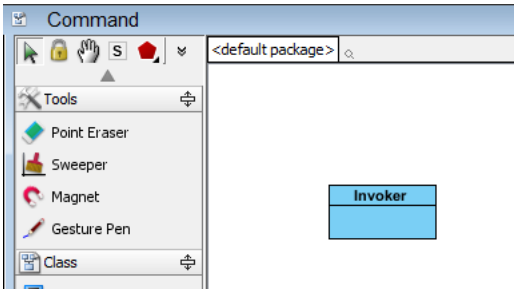
Written Date : October 14, 2009

Modeling a Design Pattern with a Class Diagram

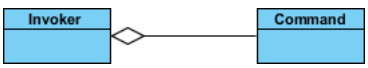
1. Create a new project named *Design Patterns*.
2. Create a class diagram named *Command*.



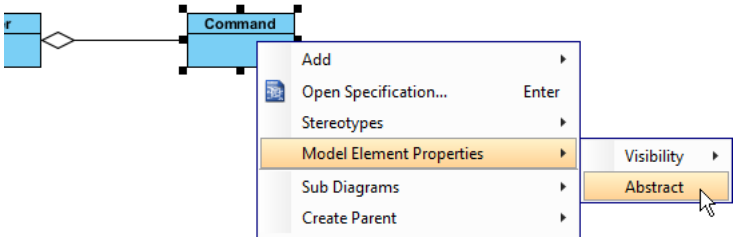
3. Select **Class** from the diagram toolbar. Click on the diagram to create a class and name it *Invoker*.



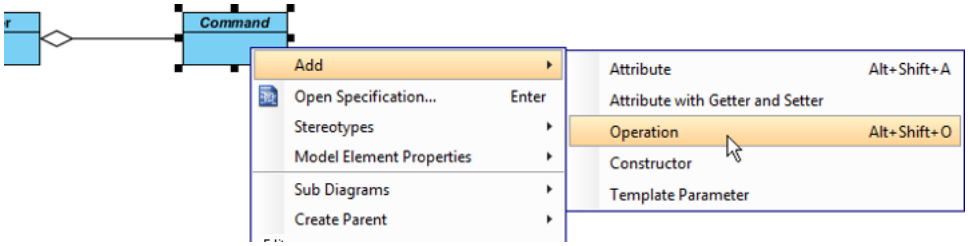
4. Move the mouse cursor over the *Invoker* class and drag out **Aggregation > Class** to create an associated class named *Command*.



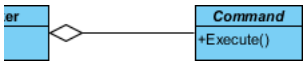
5. Right-click on *Command* and select **Model Element Properties > Abstract** to set it as abstract.



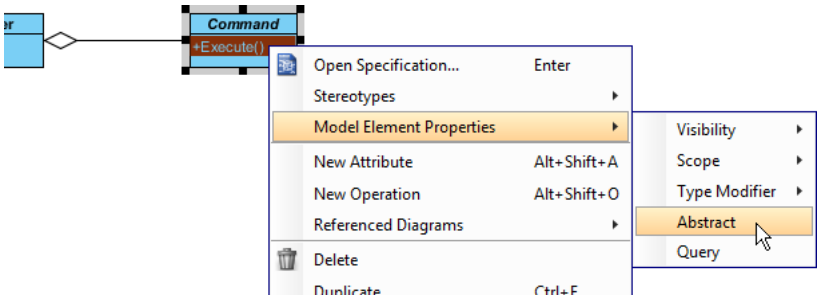
6. Right-click on the *Command* class and select **Add > Operation** from the popup menu.



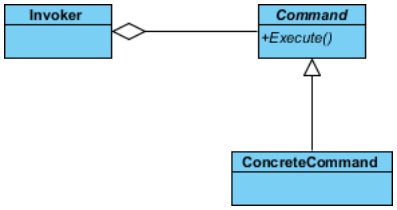
7. Name the operation *Execute()*.



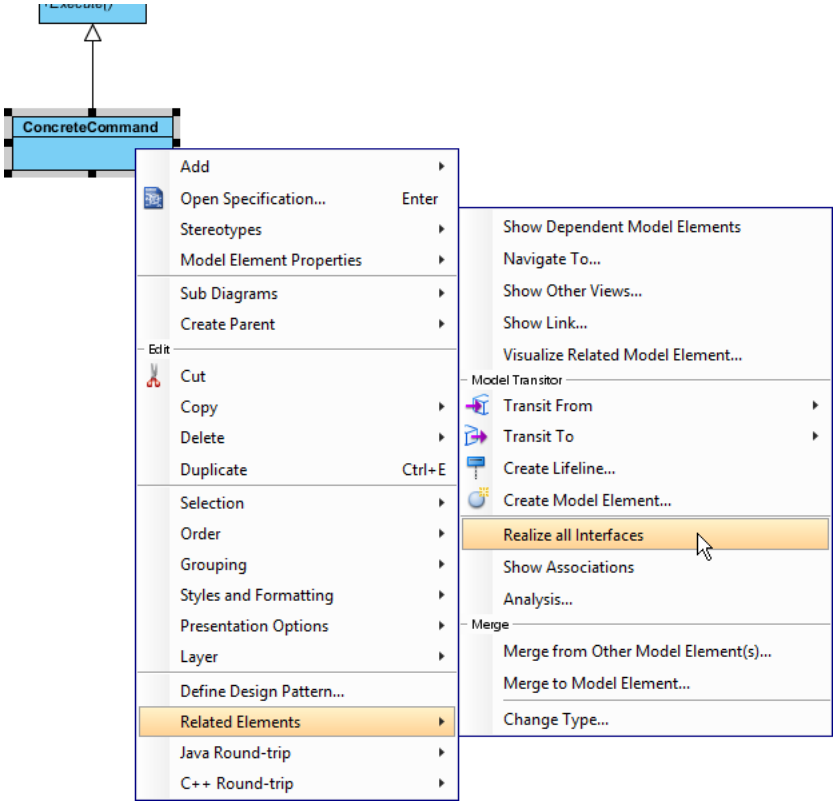
8. Right-click on *Execute* and select **Model Element Properties > Abstract** to set it as abstract.



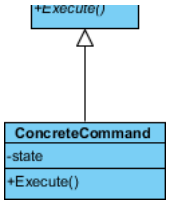
9. Move the mouse cursor over the *Command* class and drag out **Generalization > Class** to create a subclass named *ConcreteCommand*.



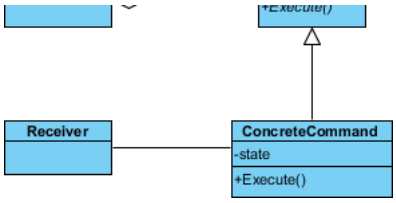
10. You need to make the concrete commands inherit operations from the command class. Right-click on *ConcreteCommand* and select **Related Elements > Realize all Interfaces** from the popup menu.



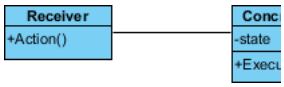
11. Right-click on the *ConcreteCommand* class and select **Add > Attribute** from the popup menu. Enter *state* as the attribute name.



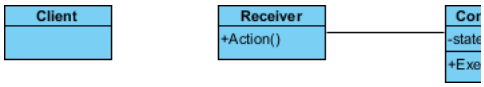
12. Move the mouse cursor over the *ConcreteCommand* class and drag out **Association > Class** to create an associated class named *Receiver*.



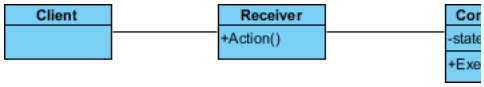
13. Right-click on the *Receiver* class and select **Add > Operation** from the popup menu. Enter *Action* as the operation name.



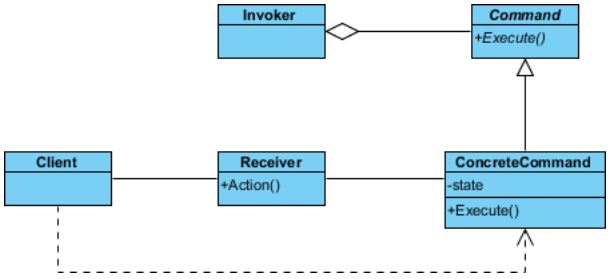
14. Create a *Client* class near the *Receiver* class.



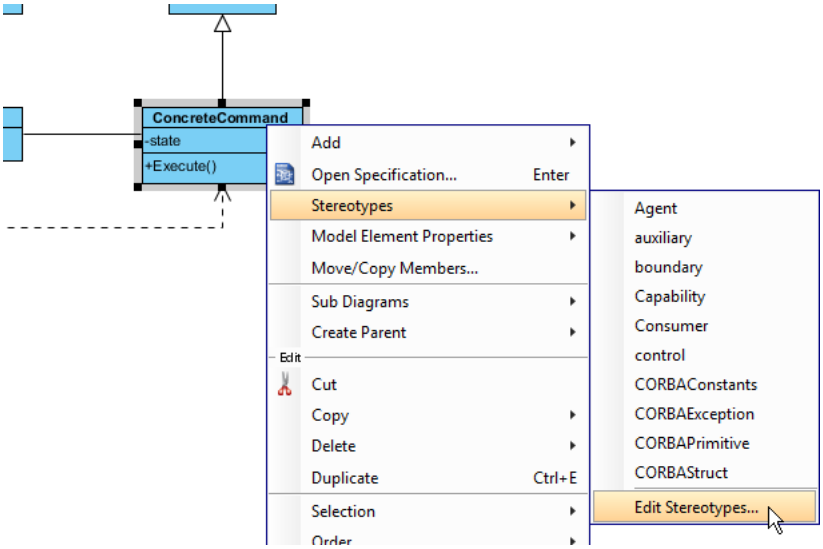
15. Move the mouse cursor over the *Client* class and drag out **Association > Class** to create an associated class named *Receiver*.



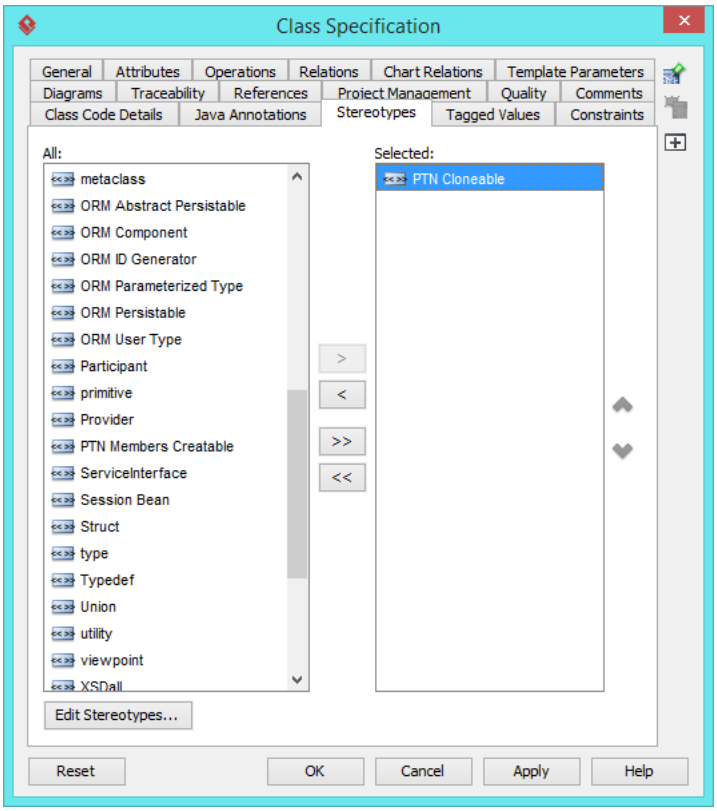
16. Move the mouse cursor over the *Client* class and drag out **Dependency > Class** to create a dependency on the *ConcreteCommand* class. Up to now, the diagram should look like this:



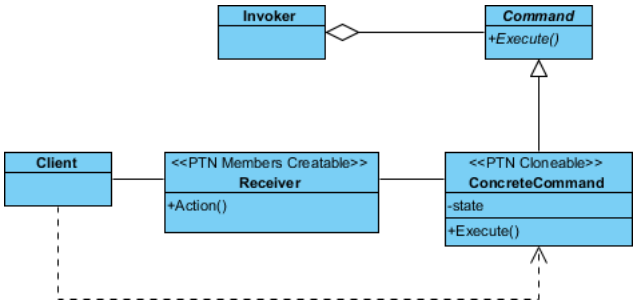
17. In practice, there may be multiple concrete commands. To represent this, stereotype the *ConcreteCommand* class as **PTN Cloneable**. Right-click on *ConcreteCommand* and select **Stereotypes > Stereotypes...** from the popup menu.



18. In the **Stereotypes** tab of the **Class Specification** dialog box, select **PTN Cloneable** and click > to assign it to the *ConcreteCommand* class. Click **OK** to confirm.

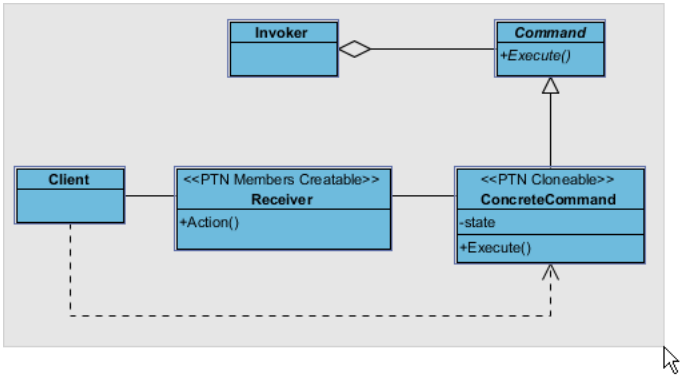


19. There may be multiple actions that the receiver can perform. To represent this, stereotype the *Receiver* class as **PTN Members Creatable**. The diagram should now look like this:

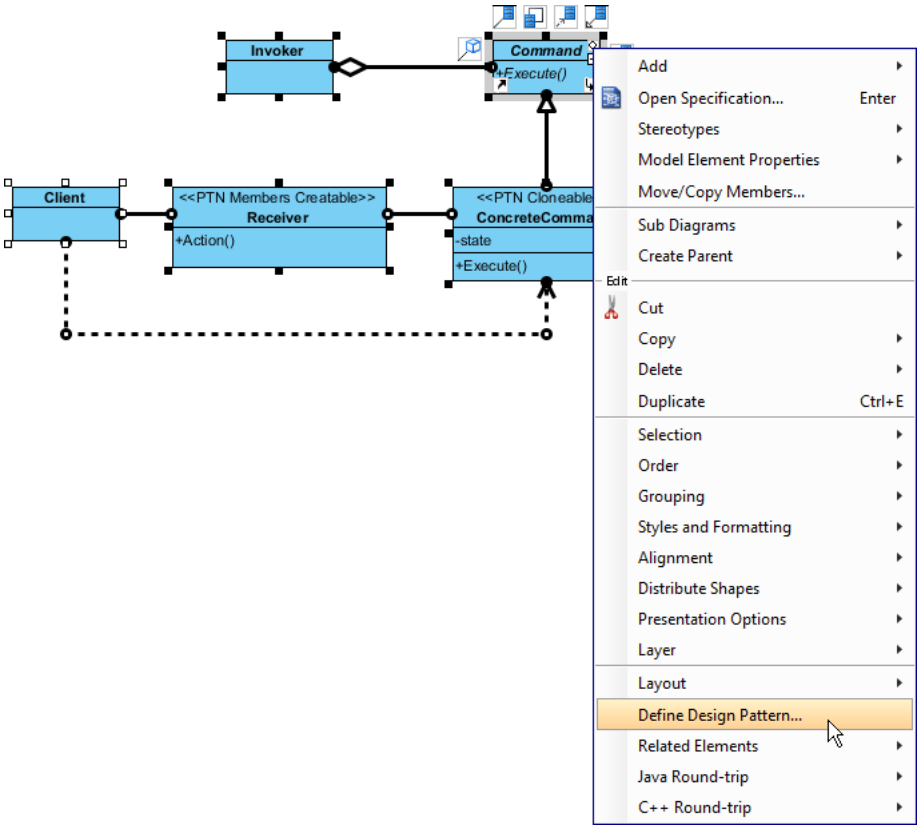


Defining a Pattern

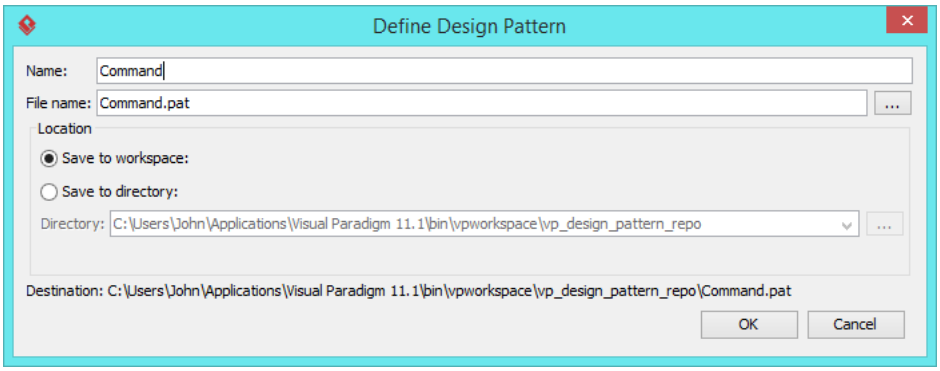
1. Select all classes on the class diagram.



2. Right-click on the selection and select **Define Design Pattern...** from the popup menu.



3. In the **Define Design Pattern** dialog box, specify the pattern name as *Command*. Keep the file name as is and click **OK** to proceed.

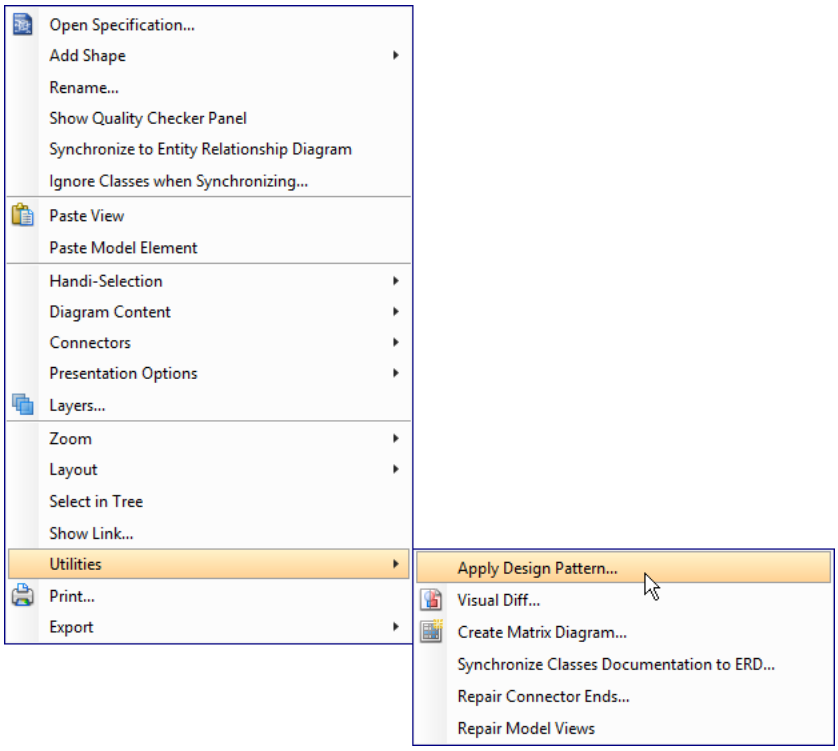


Applying a Design Pattern to a Class Diagram

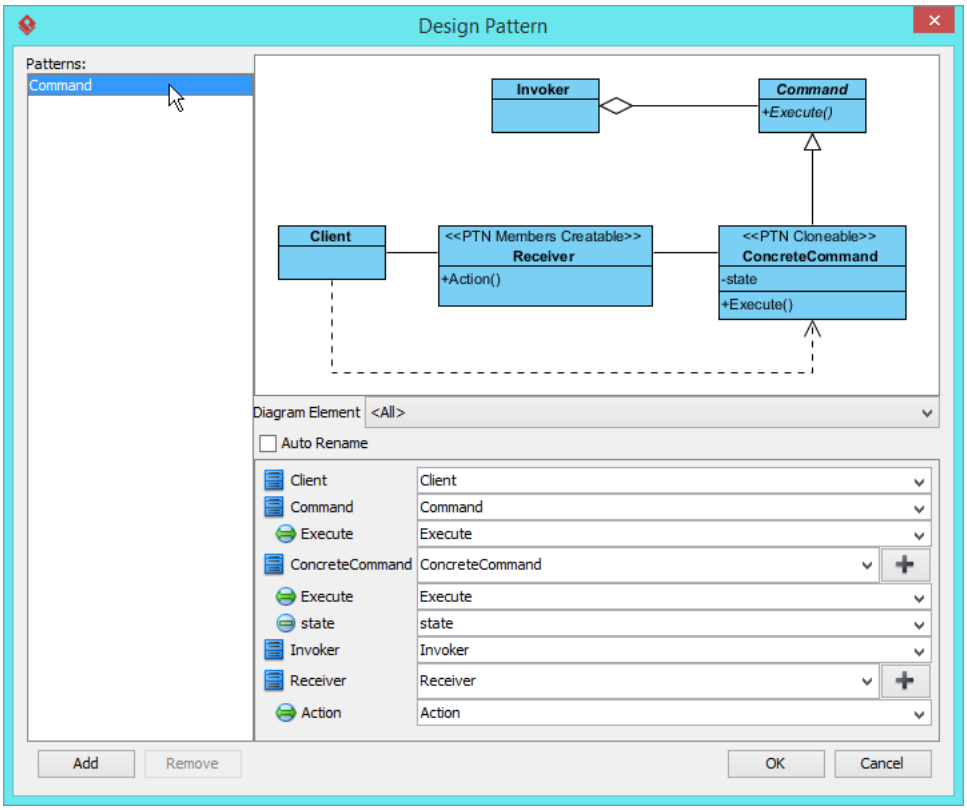
In this section, we will apply the command pattern to model a document editor.

1. Create a new project named *Document Editor*.
2. Create a class diagram named *Domain Model*.

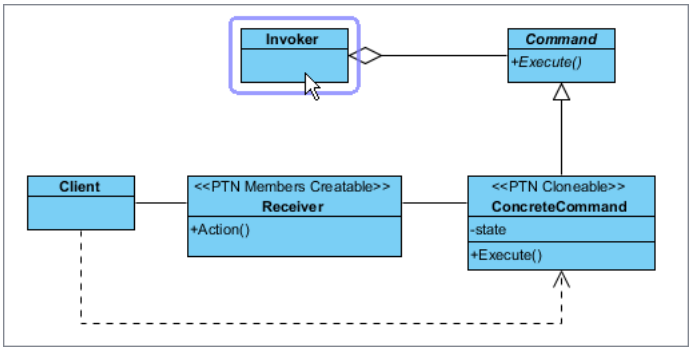
3. Right-click on the class diagram and select **Utilities > Apply Design Pattern...** from the popup menu.



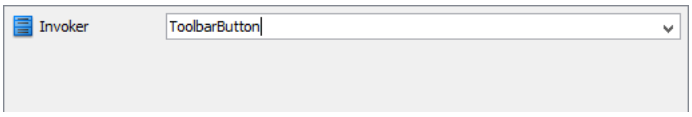
4. In the **Design Pattern** dialog box, select *Command* from the list of patterns.



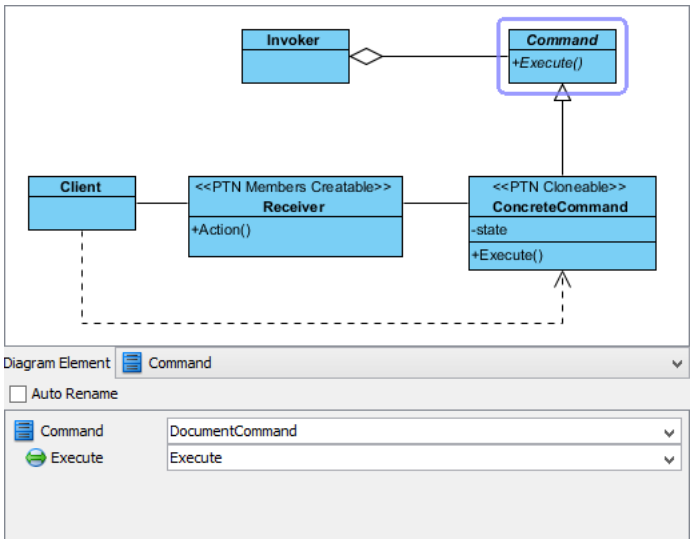
5. Select *Invoker* in the overview.



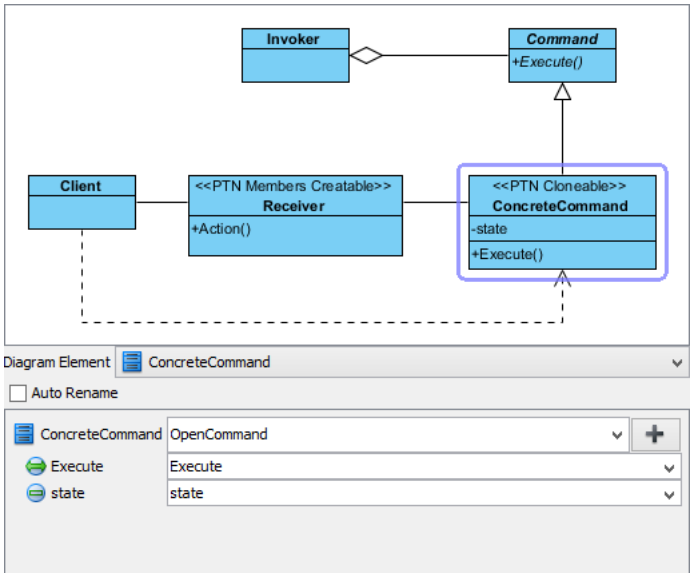
6. In the bottom pane, rename *Invoker* to *ToolbarButton*.



7. Select *Command* in the overview. In the bottom pane, rename *Command* to *DocumentCommand*.

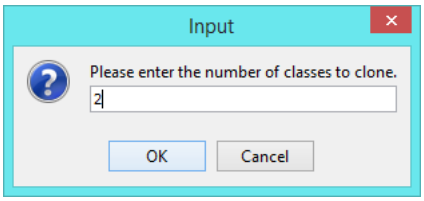


8. Select *ConcreteCommand* in the overview. In the bottom pane, rename *ConcreteCommand* to *OpenCommand*.

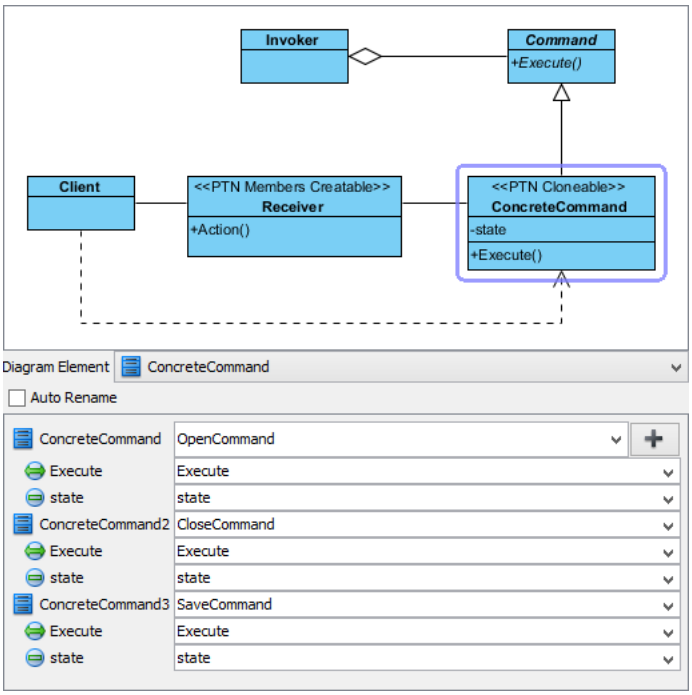


9. You need two more concrete commands for closing and saving a document. Click the + button and select **Clone...** from the popup menu.

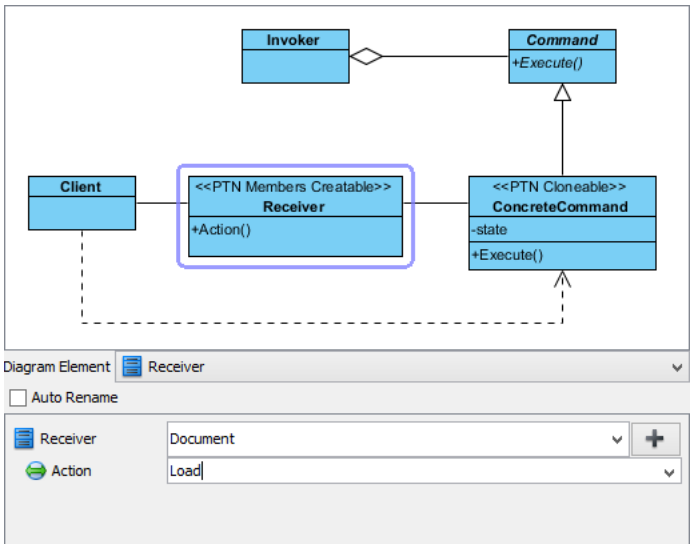
10. Enter *2* as the number of classes to clone.



11. Rename *ConcreteCommand2* to *CloseCommand* and *ConcreteCommand3* to *SaveCommand*.

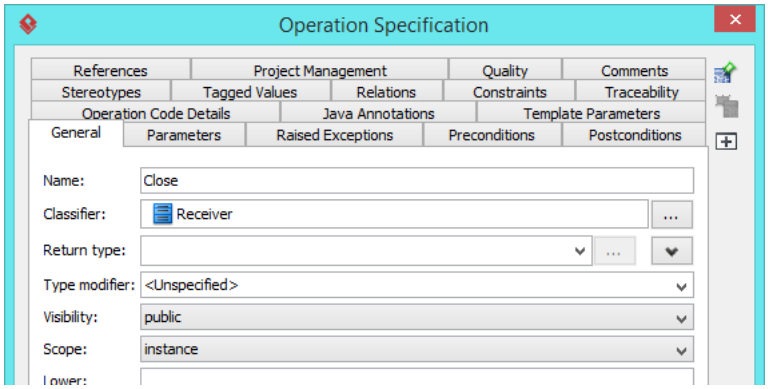


12. Select *Receiver* in the overview. In the bottom pane, rename *Receiver* to *Document* and the *Action* operation to *Load*.



13. Create more operations for closing and saving documents. Click the + button and select **New Operation...** from the popup menu.

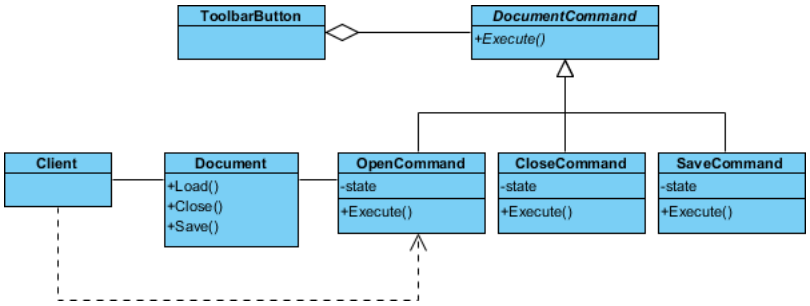
14. In the **Operation Specification**, enter *Close* as the name and click **OK** to confirm.



15. Repeat steps 13 and 14 to create the *Save* operation.

16. Click **OK** to apply the pattern to the diagram.

17. Tidy up the diagram. The result should look like this:



Resources

1. Command.pat
2. Design Patterns.vpp

Related Links

- 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/)