Interpreter Pattern Tutorial

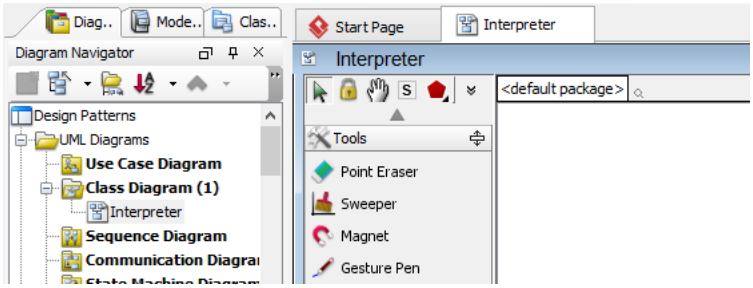
Written Date : October 14, 2009

This tutorial is aimed to guide the definition and application of Gang of Four (GoF) interpreter design pattern. By reading this tutorial, you will know how to develop a model for the interpreter pattern, and how to apply it in practice.

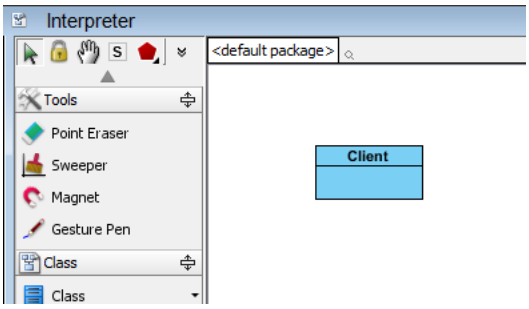
Modeling Design Pattern with Class Diagram

1. Create a new project *Design Patterns*.

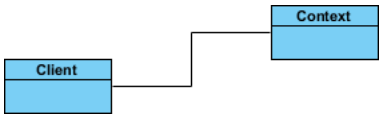
2. Create a class diagram *Interpreter*.



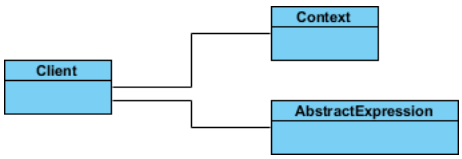
3. Select **Class** from diagram toolbar. Click on the diagram to create a class. Name it as *Client*.



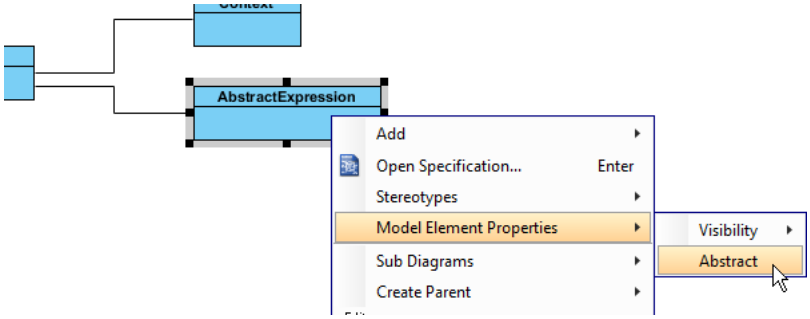
4. Move the mouse cursor over the *Client* class, and drag out **Association** > **Class** to create an associated class *Context*.



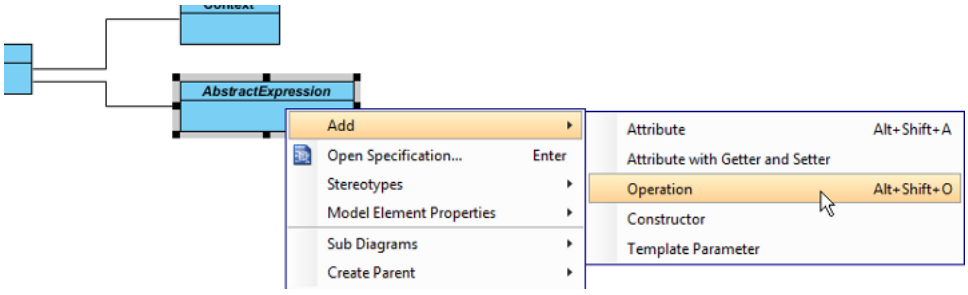
5. Move the mouse cursor over the *Client* class, and drag out **Association** > **Class** to create an associated class *AbstractExpression*.



6. Right-click on *AbstractExpression*, and select **Model Element Properties** > **Abstract** to set it as abstract.

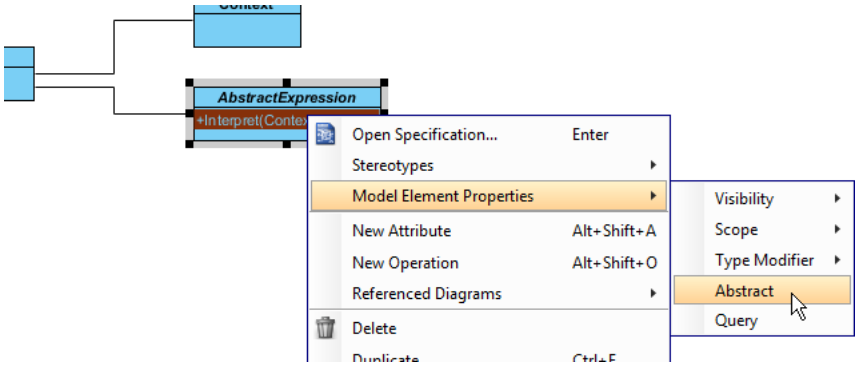


7. Right-click on *AbstractExpression* class, and select **Add** > **Operation** from the popup menu.

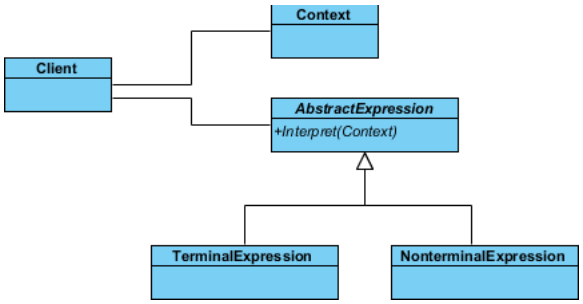


8. Name the operation *Interpret(Context)*.

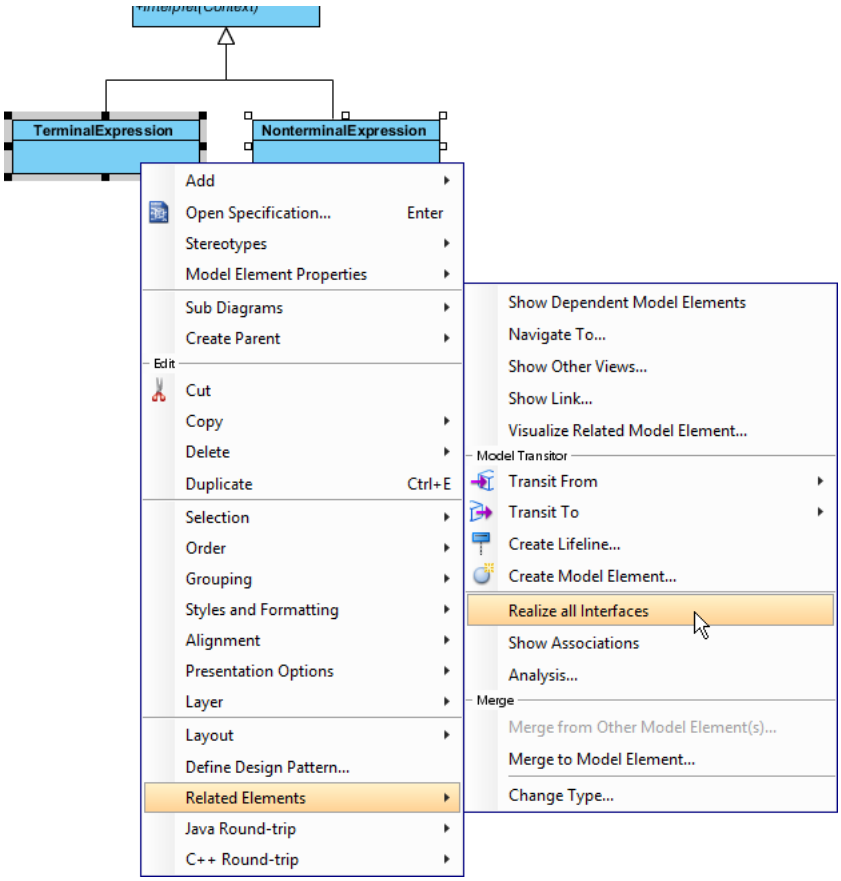
9. Right-click on *Interpret(Context)*, and select **Model Element Properties** > **Abstract** to set it as abstract.



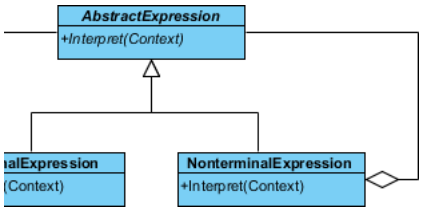
10. Move the mouse cursor over the *AbstractExpression* class, and drag out **Generalization** > **Class** to create a subclass *TerminalExpression*. Repeat this step to create another subclass *NonterminalExpression*, from *AbstractExpression*.



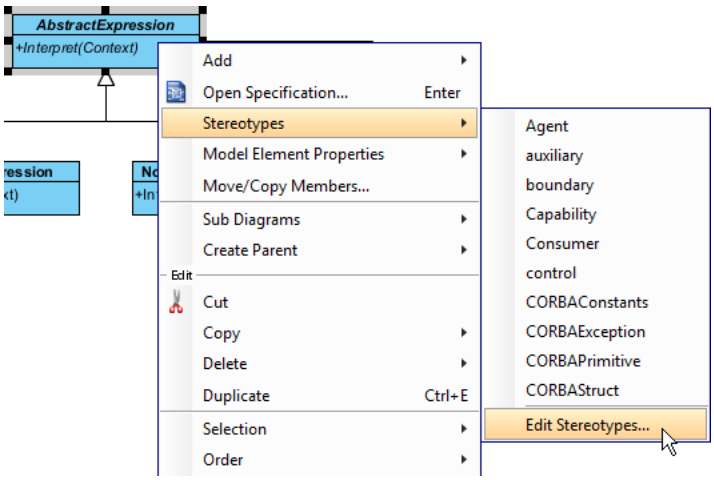
11. *TerminalExpression* and *NonterminalExpression* will inherit the operations from *AbstractExpression*. Select *TerminalExpression* and *NonterminalExpression*, right-click on them and select **Related Elements** > **Realize all Interfaces** from the popup menu.



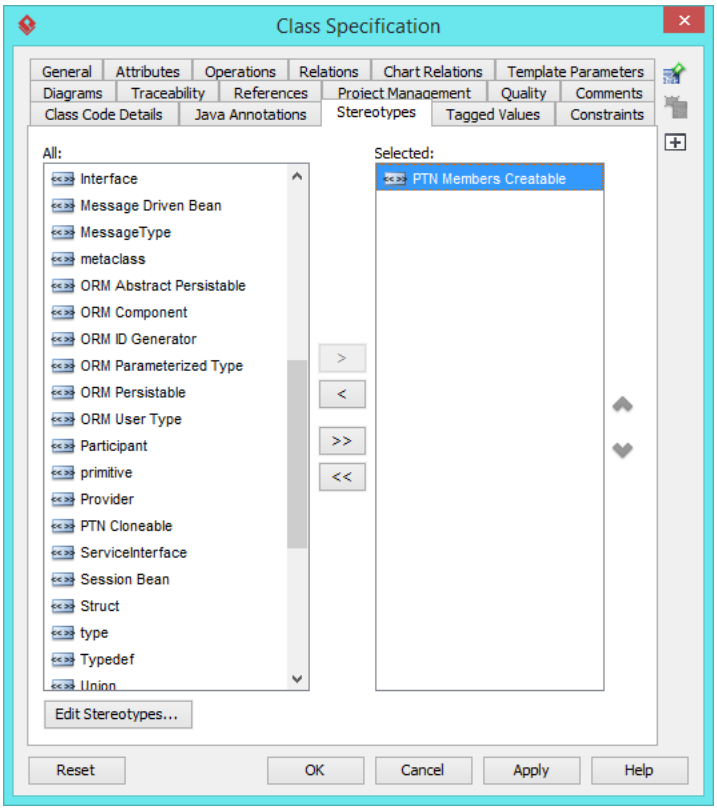
12. Move the mouse cursor over the *NonterminalExpression* class, and drag out **Aggregation** > **Class** to *AbstractExpression*.



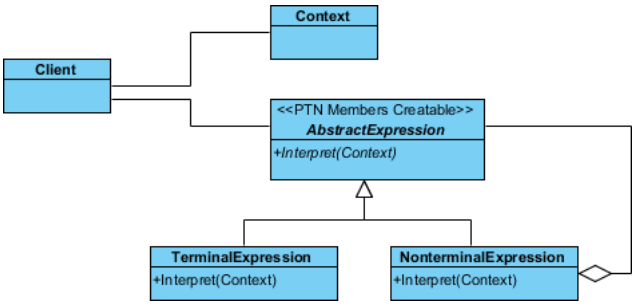
13. In practice, there may be multiple operations in *AbstractExpression*. To represent this, stereotype the class *AbstractExpression* as **PTN Members Creatable**. Right-click on *AbstractExpression* and select **Stereotypes** > **Stereotypes...** from the popup menu.



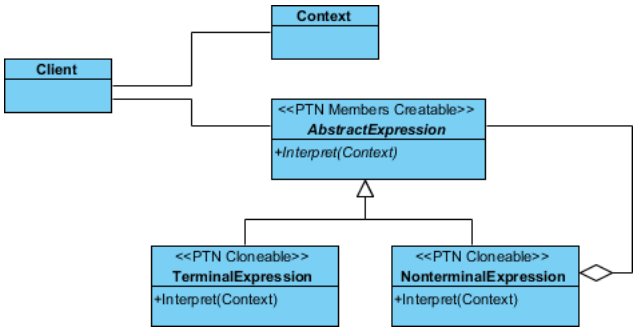
14. In the **Stereotypes** tab of the **Class Specification** dialog box, select **PTN Members Creatable** and click > to assign it to *AbstractExpression* class. Click **OK** to confirm.



Up to now, the diagram should look like this:

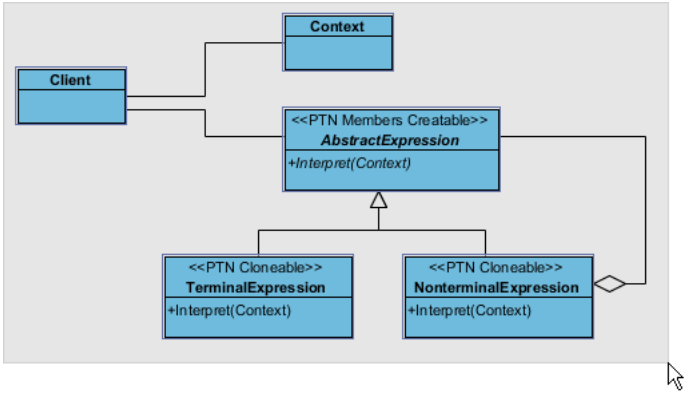


15. In practice, there may be multiple *TerminalExpression* and/or *NonterminalExpression*. To represent this, assign them with **PTN Cloneable**, by following steps 13 and 14.

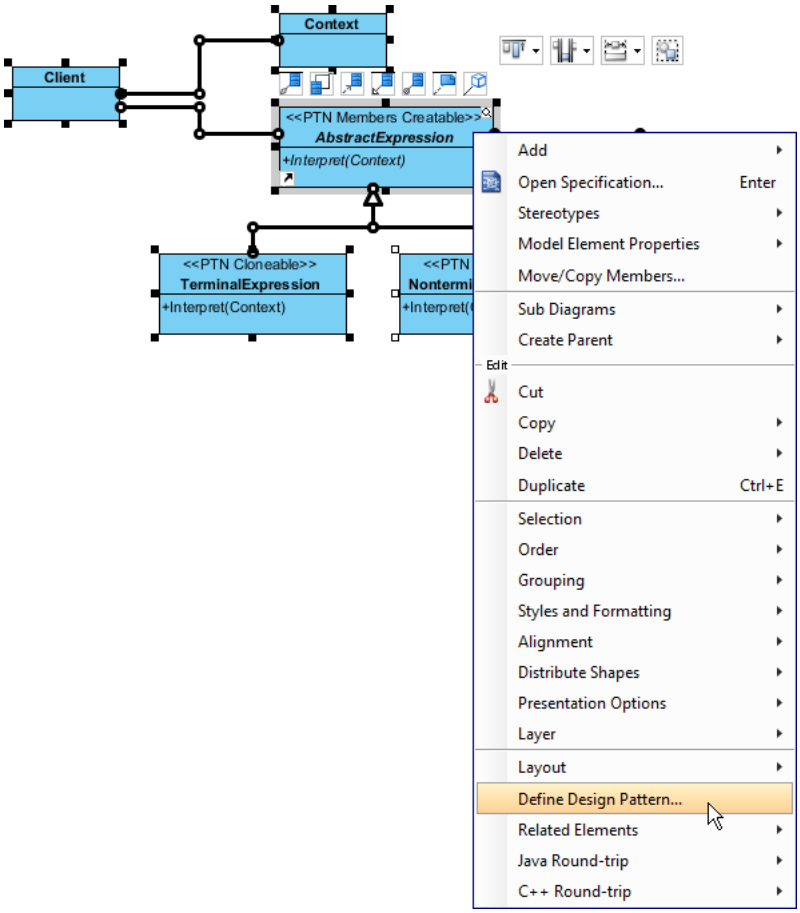


Defining Pattern

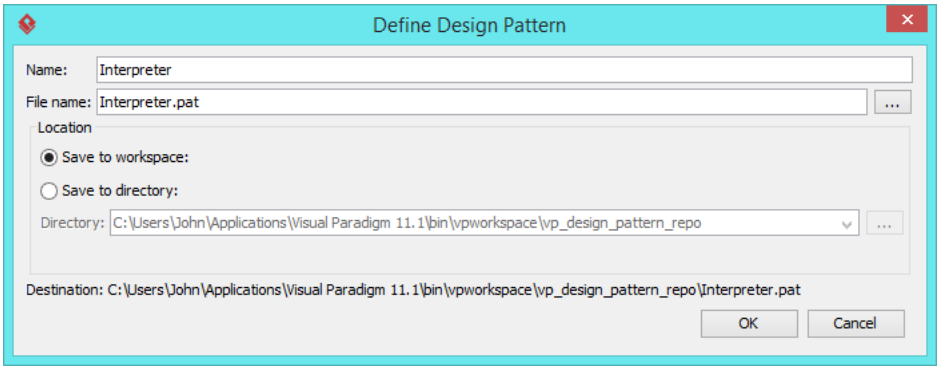
1. Select all classes on the class diagram.



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 *Interpreter*. Keep the file name as is. Click **OK** to proceed.

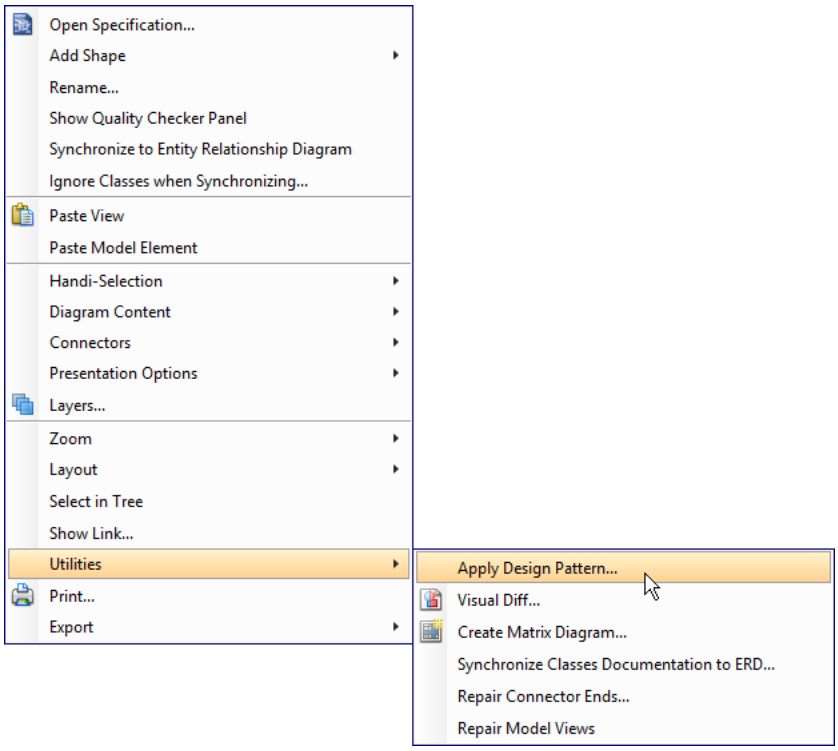


Applying Design Pattern on Class Diagram

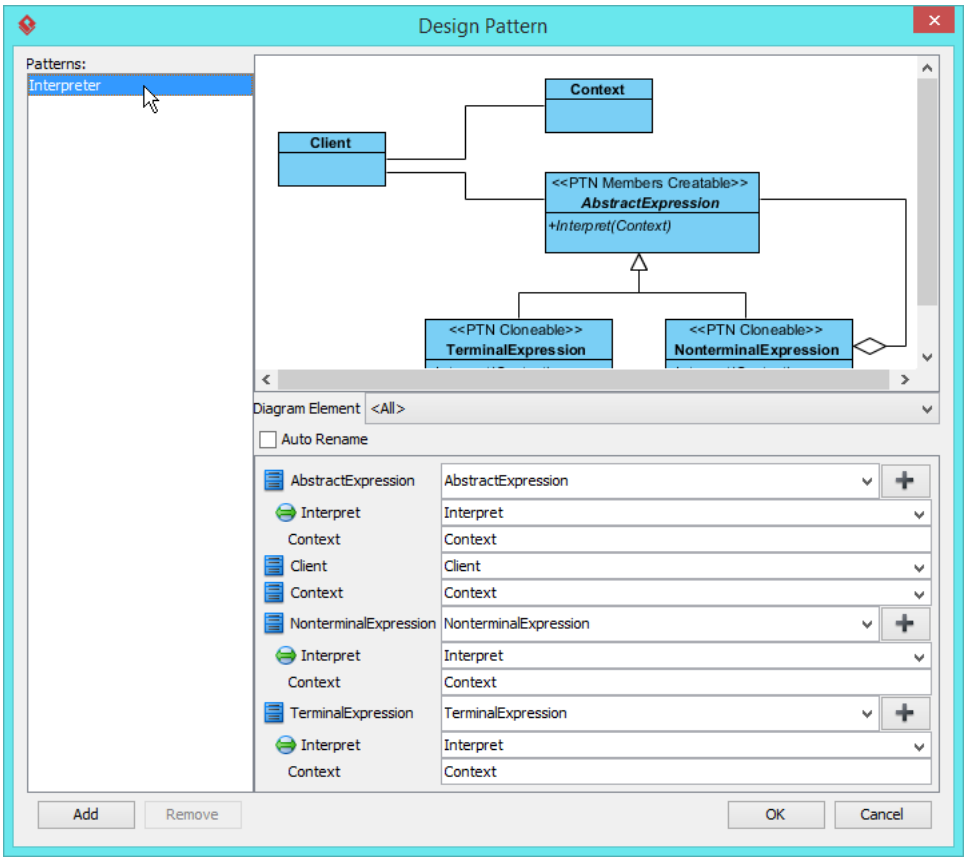
In this section, we are going to apply the interpreter pattern to model the interpretation of expression.

1. Create a new project *Expression*.
2. Create a class diagram *Expression*.

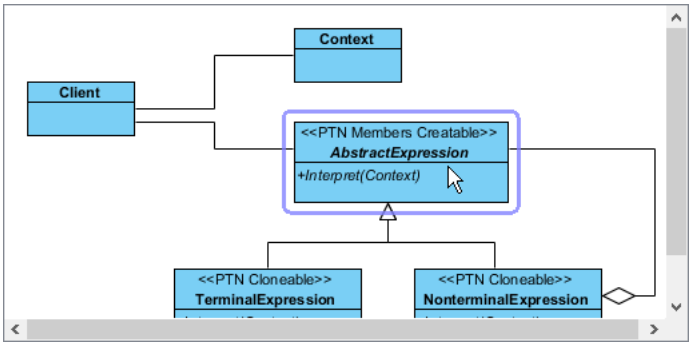
3. Right-click on the class diagram and sleect **Utilities** > **Apply Design Pattern...** from the popup menu.



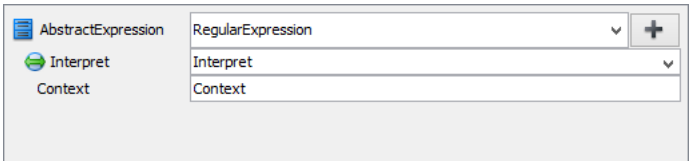
4. In the **Design Pattern** dialog box, select *Interpreter* from the list of patterns.



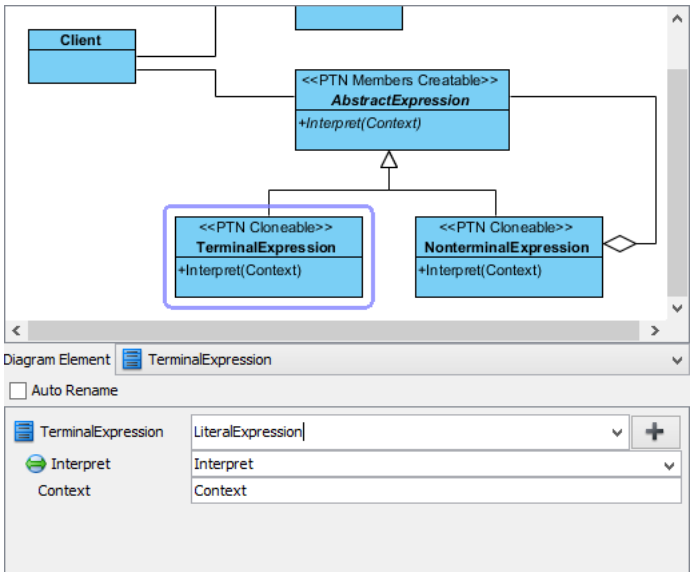
5. Click on *AbstractExpression* in the overview.



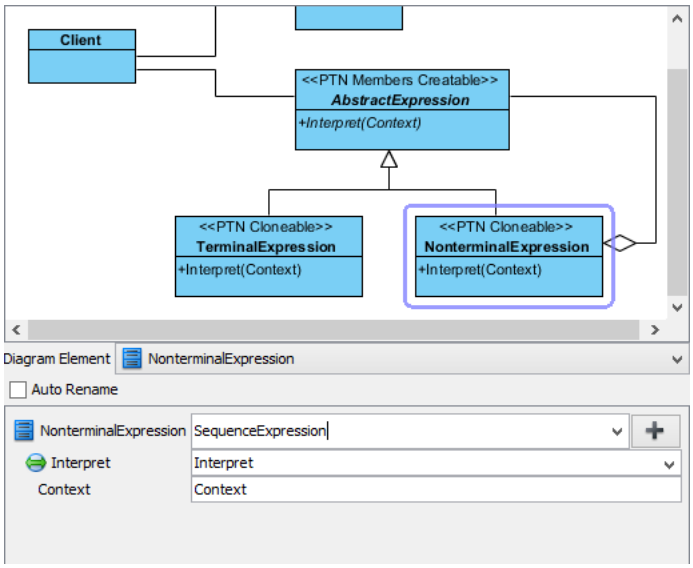
6. Rename *AbstractExpression* to *RegularExpression*.



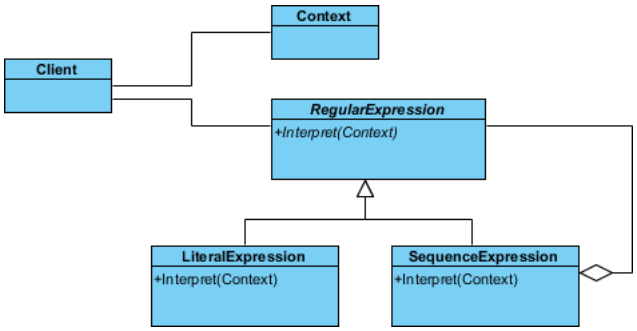
7. Select *TerminalExpression* in overview, and rename it to *LiteralExpression* at the bottom pane.



8. Select *NonterminalExpression* in overview, and rename it as *SequenceExpression* at the bottom pane. Click **OK** to apply the pattern to diagram.



This is the result:



Resources

1. Design Patterns.vpp
2. Interpreter.pat

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/)