

Facade Pattern Tutorial

Written Date : October 14, 2009

This tutorial is aimed to guide the definition and application of <u>Gang of Four (GoF)</u> facade <u>design</u> <u>pattern</u>. By reading this tutorial, you will know how to develop a model for the facade 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 *Facade*.



3. Select **Package** from diagram toolbar. Press on the diagram and drag it towards bottom right to form a package representing a subsystem.

🖻 Facade		
🔂 🖑 S 🌒 🗧	<default package=""> 🔍</default>	
Anchor		1
⊕— Containment		
Common 💠		
Tackage 🔹		
🛃 Diagram Overview		
Generic Connector		
🔼 Image Shape		
Screen Capture		
戸 Callout		
📃 Rectangle		
i Oval		
🛆 Polygon		
≶ Line		
📃 Text Box		6
📷 Diagram Info		

4. Name the package *subsystem*.

🕙 Facade	
📐 🔂 🖑 S 🌒 🗧	<default package=""> Q</default>
Anchor	
- Containment	subsystem
Par Common 🗘	
🛅 Package 🔹 👻	
🛃 Diagram Overview	
Generic Connector	
🔁 Image Shape	
Screen Capture	
戸 Callout	
🔲 Rectangle	
i Oval	
🛆 Polygon	
🗲 Line	
🔳 Text Box	
🗔 Diagram Info	

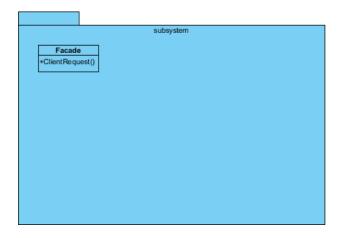
5. Select **Class** from diagram toolbar. Click inside subsystem to create a class. Name it as *Facade*.

	subsystem	
Facade		
racade		

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

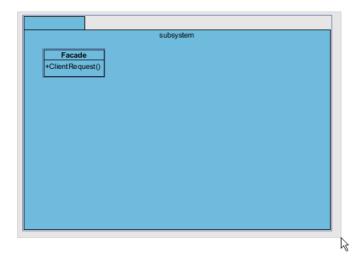
Faca	de	1		
		Add	×.	Attribute Alt+Shift+A
		Open Specification	Enter	Attribute with Getter and Setter
		Stereotypes	•	Operation Alt+Shift+O
		Model Element Properties	•	Constructor
		Sub Diagrams	•	Template Parameter
	– Eclit ∦			-

7. Name the operation *ClientRequest*. Note that it must be a public operation that enables classes external to the subsystem to access it.



Defining Pattern

1. Select everything on the class diagram.



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

				۹ ۵
		subsystem		
Facade		Open Specification	Enter]
+ClientRequest()		Stereotypes	•	
		Model Element Properties	•	
		Sub Diagrams	•	
		Form Subdiagram		
	- Edit	Cut		
		Сору	•	
		Delete	•	
		Duplicate	Ctrl+E	
		Selection	•	
×		Order	•	<u> </u>
		Grouping	•	
		Styles and Formatting	+	
		Alignment	•	
		Presentation Options	•	
		Layer	•	
		Layout	•	
		Define Design Pattern	ar .	
		Related Elements	•	

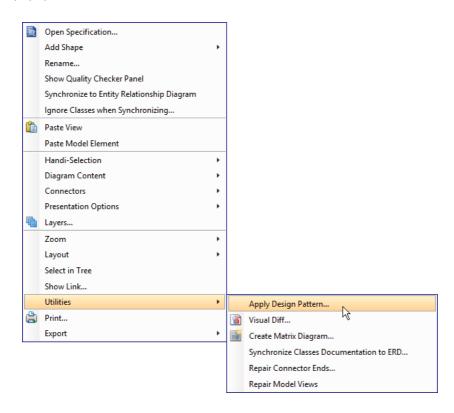
3. In the **Define Design Pattern** dialog box, specify the pattern name *Facade*. Keep the file name as is. Click **OK** to proceed.

\$	Define Design Pattern ×
Name:	Facade
File name:	Facade.pat
Location	
	to workspace:
	to directory:
Directory	y: C:\Users\John\Applications\Visual Paradigm 11.1\bin\vpworkspace\vp_design_pattern_repo v
Destination	n: C:\Users\John\Applications\Visual Paradigm 11.1\bin\vpworkspace\vp_design_pattern_repo\Facade.pat
	OK Cancel

Applying Design Pattern on Class Diagram

In this section, we are going to apply the facade pattern in modeling a code generator.

- 1. Create a new project Code Generator.
- 2. Create a class diagram *Generator*.
- 3. Right-click on the class diagram and select **Utilitiews > Apply Design Pattern...** from the popup menu.



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

\$	Design Pattern	×
Patterns: Facade	subsystem Facade +ClientRequest()	
	Diagram Element <all></all>	~
	Facade Facade	> > >
Add Remove	OK Cancel	

5. Select *subsystem* in the overview pane.

	subsystem	
Facade +ClientRequest()		
	Ŗ	

6. At the bottom pane, rename *subsystem* to *generator*. Rename *Facade* to *CodeGenerator* and *ClientRequest* to *generate*.

🚞 subsystem	generator	¥
📑 Facade	CodeGenerator	~
😂 ClientRequest	generate	~

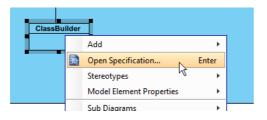
7. Click **OK** to apply the pattern. This is the diagram obtained:

	generator	
CodeGenerator		
+generate()		

8. We need to fill in the subsystem. Move the mouse cursor over the *CodeGenerator* class, and drag out **Aggregation** > **Class** to empty region in the package to create a class. Name the class *ClassBuilder*.

	generator	
Code Generator +generate()		
ClassBuilder		

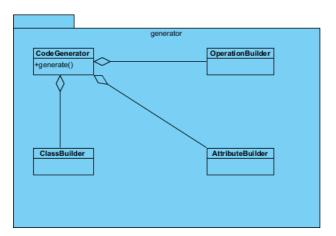
9. Right-click on *ClassBuilder* and select **Open Specification** from the popup menu.



10. In the Class Specification, set Visibility to be package. Click OK to confirm.

	Cli	ass Speci	ificatior	า		
Diagrams Traceabi	ility Reference	es Proje	ct Manage	ement	Quality	Comments
Class Code Details	Java Annotation	ns Stere	otypes	Tagged	Values	Constraints
General Attributes	Operations	Relations	Chart R	elations	Templa	te Parameters
Name: ClassBuilder Parent: generat Visibility: public Documen <unspecified private protected package public</unspecified 						····

11. Repeat steps 8 to 10 to create classes AttributeBuilder and OperationBuilder.



Resources

- 1. <u>Design Patterns.vpp</u>
- 2. Facade.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/)