

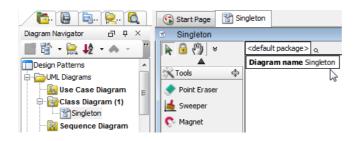
Singleton Pattern Tutorial

Written Date : September 30, 2009

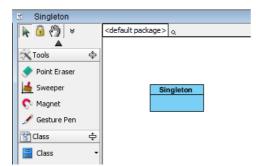
This tutorial is aimed to guide the definition and application of <u>Gang of Four (GoF)</u> singleton <u>design</u> <u>pattern</u>. By reading this tutorial, you will know how to develop a model for the signletion pattern, and how to apply it in practice.

Modeling Design Pattern with Class Diagram

- 1. Create a new project Design Patterns.
- 2. Create a class diagram *Singleton*.



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



-

-

4. Right-click on the *Singleton* class and select **Add** > **Attribute** from the popup menu.

1	Singleto	on					
			Add	+	Attribute	N	Alt+Shift+A
			Open Specification	Enter	Operation	13	Alt+Shift+O
			Stereotypes	+	Template Par	ameter	
			Model Element Properties	۱.			
			Sub Diagrams	•			

5. Name the attribute instance. Set its type as Singleton.



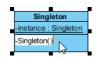
6. The attribute instance need to be static. Right-click on the attribut and select **Model Element Properties > Scope > Classifier** from the popup menu.

iton Sinaleton						
Open Specification	Enter					
Stereotypes	*					
Model Element Properties	•	Multiplicity	F			
New Attribute	Alt+Shift+A	Visibility	►			
New Operation	Alt+Shift+O	Scope	×		classifier	
Sub Diagrams	۲.	Type Modifier	×	~	instance	5
Delete		Setter				
Duplicate	Ctrl+E	Getter				
Selection	•					
	indiation Open Specification Stereotypes Model Element Properties New Attribute New Operation Sub Diagrams Delete Duplicate	Incluton Enter Open Specification Enter Stereotypes > Model Element Properties > New Attribute Alt+Shift+A New Operation Alt+Shift+O Sub Diagrams > Delete Duplicate Ctrl+E	Model Element Properties Multiplicity New Attribute Alt+Shift+A New Operation Alt+Shift+O Sub Diagrams J Delete Setter Duplicate Ctrl+E	Inclusion Enter Open Specification Enter Stereotypes > Model Element Properties > New Attribute Alt+Shift+A New Operation Alt+Shift+O Sub Diagrams > Delete Setter Duplicate Ctrl+E	Inter Enter Stereotypes > Model Element Properties > New Attribute Alt+Shift+A New Operation Alt+Shift+O Sub Diagrams > Delete Getter Duplicate Ctrl+E	Involution Enter Open Specification Enter Stereotypes > Model Element Properties > New Attribute Alt+Shift+A New Operation Alt+Shift+O Sub Diagrams > Delete Setter Duplicate Ctrl+E

7. Create constructor for the *Singleton* class. Right-click on *Singleton* and select **Add** > **Operation** from the popup menu.

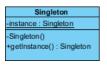
Singleton	_					
-instance : Singleton		Add	×.	Attribute		Alt+Shift+A
		Open Specification	Enter	Operation	N	Alt+Shift+O
		Stereotypes	+	Template Parameter	5	
		Model Element Properties	+			
		Move/Copy Members				

8. Name the operation *Singleton*, which follows the *Singleton* class's name. Change + to - in front of the operation name to indicate that this is a private constructor.



9. Right-click on *Singleton* and select **Add** > **Operation** from the popup menu.

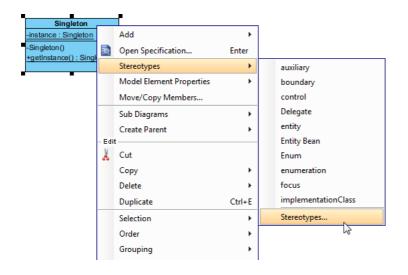
10. Name the operation *getInstance*, and make it return *Singleton*.



11. The operation *getInstance* need to be static. Right-click on the operation and select **Model** Element Properties > Scope > Classifier from the popup menu.

Singlete					
-Singleton() +getInstance()::	Singleton				
	Open Specification	Enter			
	Stereotypes	+			
	Model Element Properties	•	Visibility	×	
	New Attribute	Alt+Shift+A	Scope	×	classifier
	New Operation	Alt+Shift+O	Type Modifier	×	🖌 instance 🗟
	Sub Diagrams	•	Abstract		
Ť	Delete		Query		
	Duplicate	Ctrl+E			

12. In practice, there may be operations for accessing data in the Singleton class. To represent this, stereotype the *Singleton* class as PTN Members Creatable. Right-click on the *Singleton* class and select **Stereotypes** > **Stereotypes...** from the popup menu.



13. In the class specification dialog box, select **PTN Members Creatable** and click > to assign it. Click **OK** to confirm.

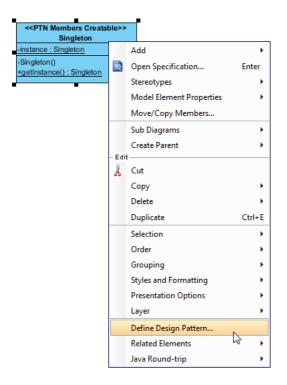
Gen	eral	Attrib	utes	Operat	ions		Relatio	ns	Temp	late Parame	ters
	gged Va			straints	_	Diagra	_		ferences		ments
C	ass Coo	de Details		Java An	notat	ions		ORM	Query	Stereot	ypes
All:					_	5	Selecte	d:			_
<< >>	impleme	entationCl	ass		^		<.>>	TN Me	embers Cre	eatable	
<< >>	Interfac	e									
<< >>	Messag	e Driven	Bean								
<< >>	metacla	SS									
<< >>	ORM AI	bstract Pe	ersistable	e							
<< >>	ORM Co	omponent									
~~>>	orm ID	Generato	or								
<< >>	ORM Pa	irameteriz	ed Type	•		>					
~	ORM Pe	ersistable				<					
<< >>	ORM Us	ser Type									**
	primitive			:		>>					~
	PTN Clo					<<]					
_	Sessior	n Bean									
_	Struct										
~~>>											
	Typede	f									
	utility										
_	XSDall										
		ibuteGrou	qu								
~~~	XSDcho	lice			Ψ.						
Edi	t Stereo	otypes									

Up to now, the diagram should look like this:



# **Defining Pattern**

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



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

🌀 Define	Design Pattern	×					
Name:	Singleton ]						
File name:	Singleton.pat						
Location							
Save	to workspace:						
Save	Save to directory:						
Directory	r: C:\vpworkspace\vp_design_pattern_repo	<b>–</b>					
Destination	: C:\vpworkspace\vp_design_pattern_repo\Singleton.pat						
		OK Cancel					
Destination	: C:\vpworkspace\vp_design_pattern_repo\Singleton.pat	OK Cancel					

## **Applying Design Pattern on Class Diagram**

In this section, we are going to apply the singleton pattern in modeling a class registry.

- 1. Create a new project *Class Registry*.
- 2. Create a class diagram *The Registry*.

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

20	Open Specification			1	1	1	1	7	7	7	7	7	7
395													
	Add Shape	•											
	Rename												
	Synchronize to Entity Relationship Diagram												
	Ignore Classes when Synchronizing												
(în	Paste View												
	Paste Model												
	Handi-Selection	•											
	Diagram Content	•											
	Connectors	•											
	Presentation Options	►											
•	Layers												
	Zoom	×											
	Layout	•											
	Select in Tree												
	A ³ Platform	•											
	Utilities	•		Арр	Apply Desig	Apply Design Pattern	Apply Design Pattern	Apply Design Pattern	Apply Design Pattern	Apply Design Pattern	Apply Design Pattern	Apply Design Pattern	Apply Design Pattern
٢	Print		1	👔 Visu	👔 Visual Diff	Visual Diff	👔 Visual Diff	👔 Visual Diff	👔 Visual Diff	👔 Visual Diff	Visual Diff	Visual Diff	Visual Diff
	Export	•		Syn	Synchronize	Synchronize Classes I	Synchronize Classes Docum	Synchronize Classes Documentat	Synchronize Classes Documentation to	Synchronize Classes Documentation to ERD			

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

🕝 Design Pattern					x
Patterns: Singleton					
	Diagram Element <a< td=""><td>ll&gt;</td><td></td><td></td><td>•</td></a<>	ll>			•
	Singleton	Singleton instance Singleton getInstance		•	+ • •
Add Remove				ОК Са	incel

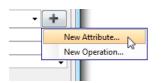
5. Click on *Singleton* in the ovewview.

<< PTN Members Creatable Singleton	6>>		
-instance : Singleton			
-Singleton()			
+getInstance() : Singleton			
	3		

6. Rename the class *Singleton*, as well as the constructor to *ClassRegistry* at the bottom pane.

🗐 Singleton	ClassRegistry + +
😑 instance	instance 👻
😸 Singleton	ClassRegistry 📋 👻
😂 getInstance	

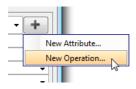
7. We need to add an attribute for holding the classes user register. Click on the + button and select **New Attribute...** from the popup menu.



8. In the Attribute Specification, enter *classMapping* as attribute name. Enter *Map* as type.

Attribute Spe	ecificat	ion			×				
Tagged Valu General A	_	Constraints Code Details	Diagrams Java Annotations	References XML Schema	Comments Stereotypes				
Name:	classMa	apping							
Classifier:	📄 Sin	gleton							
Initial value:									
Multiplicity:	Unspec	ified		▼ Ordered	d 🔽 Unique				
Visibility:	private				-				
Type:	Мар	Ι			▼				
Type modifier:	<unspe< td=""><td>ecified&gt;</td><td></td><td></td><td>•</td></unspe<>	ecified>			•				
Scope:	instanc	e			•				
Aggregation:	None				-				
Documentation HTML	-	IUEE	'≡  \; :::   F H	Fr 🥐 🦿 📑 斗	🍋 🙈 »				
Record									
Derived	Sette	r 🔲 Getter	Abstract						
Reset		ОК	Cancel	Apply	Help				

9. We need to add operations for registering class and retrieving class by type. Click on the + button and select **New Operation...** from the popup menu.



- 10. In the **Operation** Specification dialog box, enter *registerClass* as operation name.
- 11. Open the **Parameters** tab.

(	Operation Specificatio	on					23
	Tagged Values	Co	onstraints	Diagra	ms	References	Comments
	Operation Code Deta	ils	Java Annot	Java Annotations		mplate Parameters	Stereotypes
	General Parameters		Raised Exceptions			Preconditions	Postconditions
i I r	deneral	5	readed a	Acceptions		Treconditions	restconditions

12. Click Add... at the bottom of specification dialog box.

13. In the **Parameter** Specification dialog box, enter *name* as parameter name and set *String* as type.

Tagged V	alues	Constraints	References	Comments	
General	Parame	ter Code Details	Java Annotations	Stereotypes	
Name:	name				
Operation:	registerCla	SS			
Type:	String	Ι		<b>▼</b>	
Type modifier:	<unspecifi< td=""><td>ed&gt;</td><td></td><td></td></unspecifi<>	ed>			
Direction:	inout				
Default value:					
Multiplicity:	Unspecified	1	▼ Ord	lered 🔽 Unique	
Documentation					
Documentation HTML		u   = = =   :=	ः ः= । म म्ह 🥔 🦸 🛋	14 🎠 🔍 »	
1		<u>u</u>  ===	: :=  F Fr ≁ 7 📑	🧤 🎠 🍓 🕷	
1		<u>u   = = =  </u>	::=  F Fr 🛷 I 📑	ų 拝 🧶 »	
1		<u>u === </u>	: :=  F Fr 🛹 f 📑	🧤 🏪	
1		<u>u</u>  = = =	∷≡  F Fr 🛹 I 📑	🌉 🚧 🛞 »	
1		<u>u = = = </u> ;=	∷≡  F Fr 🛹 I 📑	🧤 🊈 🍓 »	
		u  ≣ ≡ ∃  ∰	: :≡   F Fr ≁ <b>f</b> mੈ	əli 🚧 🦗	
		<u>u∣≡ = = </u> ≣	∷≡  F Fr <b>∻ f</b> 📑	ali	
		<u>u  = = =  =</u>	∷≡  F Fr <b>∻ f</b> 📑	an a	
	• B I	<u>u  = = =  </u>	: :≡   F Fr <b>~ f m</b>	2 in the second	

14. Repeat steps 12 and 13 to add parameter *regClass*, and set *Class* as type. Click **OK** to confirm.

				-				
	Tagged Values Constrai				eferences		Comments	
Operation Code Details General Parameters		Java Annotations Ter Raised Exceptions			mplate Parameters Preconditions		Stereotypes Postconditions	
General		Kaiseu E.	xcepuoris	Preco	nunuoris	Postcon	uluons	
Name	Туре		Default Val	ue	Direction			
name	String				inout			
regClass	Class							
							♥	
			Open Spe	cification	Add	Re	move	

- 15. Click on the + button and select **New Operation...** from the popup menu.
- 16. In the **Operation Specification** dialog box, enter *getClass* as name, and set *Class* as return type.

Singleton	ClassRegistry +
😑 instance	instance 🗸
😂 Singleton	ClassRegistry -
😂 getInstance	getInstance 🗸
😑 classMapping	dassMapping 🗸 🗸
😝 registerClass	registerClass 🗸
name	name
regClass	regClass
😝 getClass	getClass 👻

17. Click **OK** to apply the pattern to diagram. This is the result:



### Resources

- 1. <u>Design Patterns.vpp</u>
- 2. Singleton.pat

### **Related Links**

• Full set of UML tools and UML diagrams



(https://www.visual-paradigm.com/)

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