Tutorial

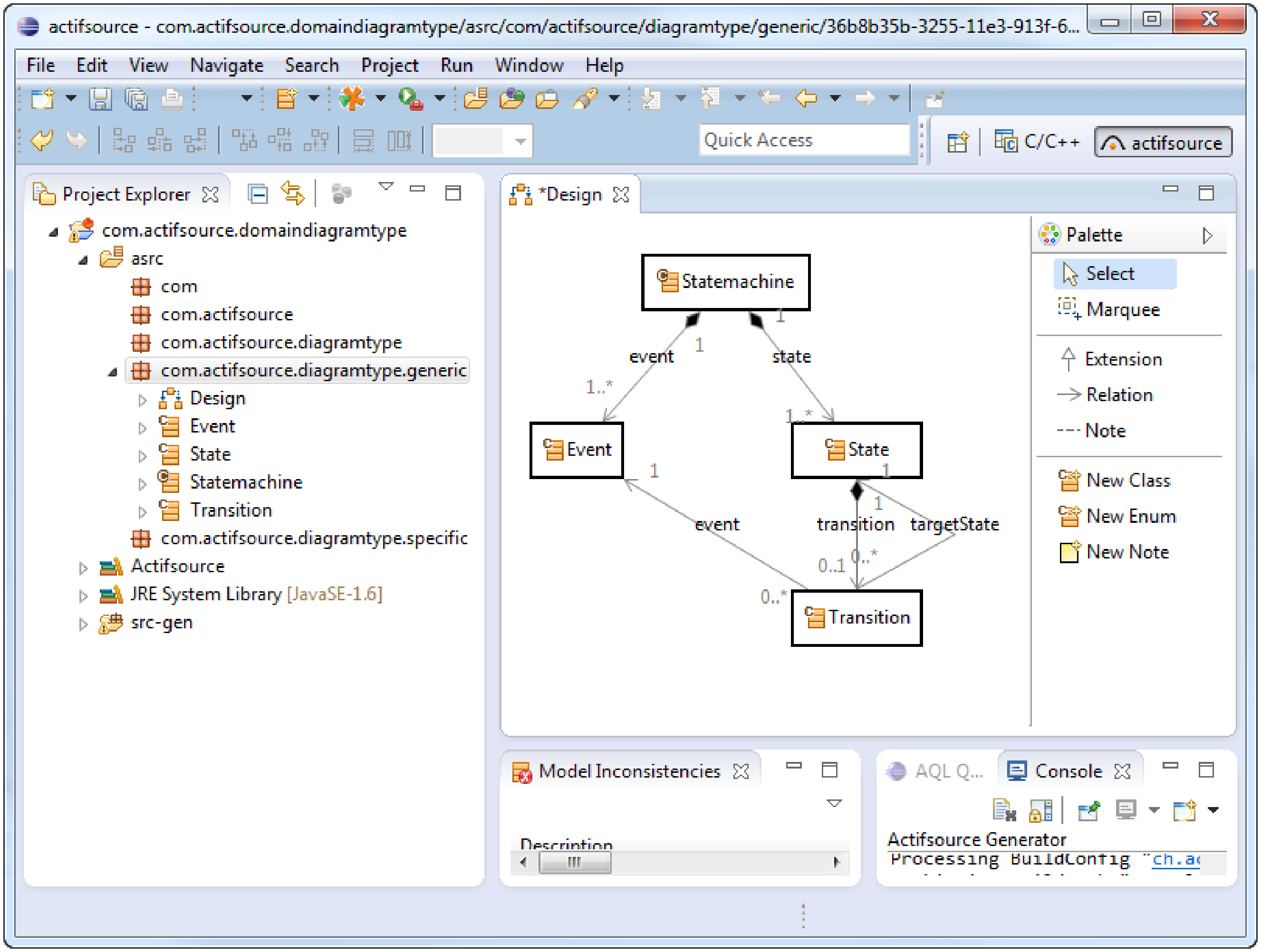
Domain Diagram Type

Tutorial	Actifsource Tutorial – Domain Diagram Type
Required Time	• 30 Minutes
Prerequisites	• Actifsource Tutorial – Installing Actifsource • Actifsource Tutorial – Simple Service • Actifsource Tutorial – Statemachine
Goal	• Create a user defined diagram type to display and edit domain specific diagrams
Topics covered	• Create a user defined diagram type to display and edit domain specific diagrams • Create a domain diagram based on the diagram type to edit the underlying root resource • Define a highlight path from node to node over any edge • Define a tooltip for elements on the domain diagram
Notation	↳ To do • Information • **Bold**: Terms from actifsource or other technologies and tools • **Bold underlined**: actifsource Resources • Underlined: User Resources • *UnderlinedItalics*: Resource Functions • `Monospaced`: User input • *Italics*: Important terms in current situation
Disclaimer	The authors do not accept any liability arising out of the application or use of any information or equipment described herein. The information contained within this document is by its very nature incomplete. Therefore the authors accept no responsibility for the precise accuracy of the documentation contained herein. It should be used rather as a guide and starting point.
Contact	**actifsource GmbH** Täfernstrasse 37 5405 Baden-Dättwil Switzerland www.actifsource.com
Trademark	**actifsource** is a registered trademark of **actifsource GmbH** in Switzerland, the EU, USA, and China. Other names appearing on the site may be trademarks of their respective owners.
Compatibility	Created with **actifsource** Version 5.9.0

Overview

- Create a user defined diagram type to display and edit domain specific diagrams
- Create a domain diagram based on the diagram type to edit the underlying root resource
- Define a highlight path from node to node over any edge
- Define a tooltip for elements on the domain diagram

Part I: Preparation



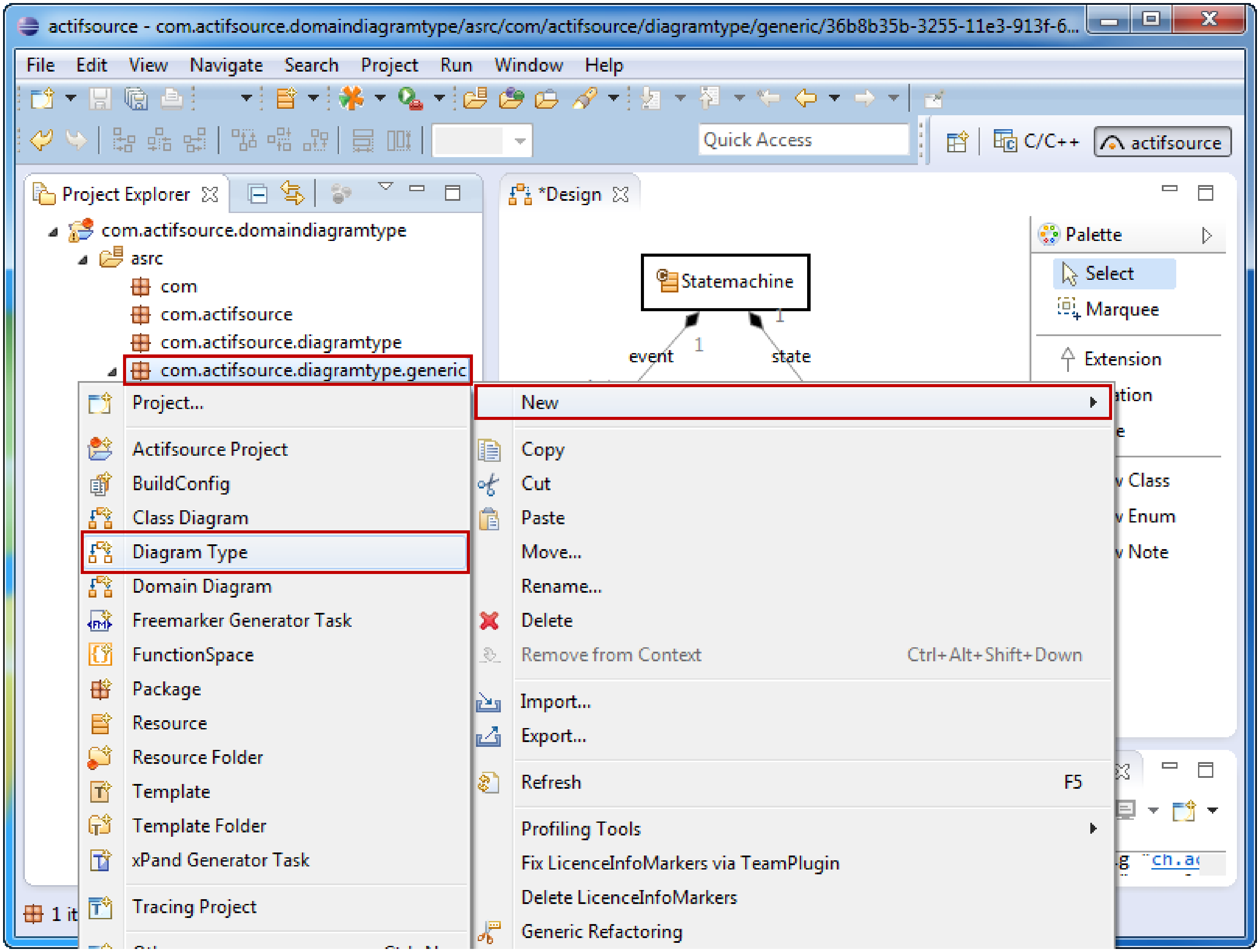
```
Actifsource Generator
Processing BuildConfig "ch.a
```

- Prepare a new actifsource Project as seen in the Actifsource Tutorial – Statemachine
- Create a generic Domain Model as shown above
- State.transition → Decorating Relation with Decorator State.-state.event
- Transition.targetState → Use Relation with RangeRestriction Transistion.-transition.-state.state

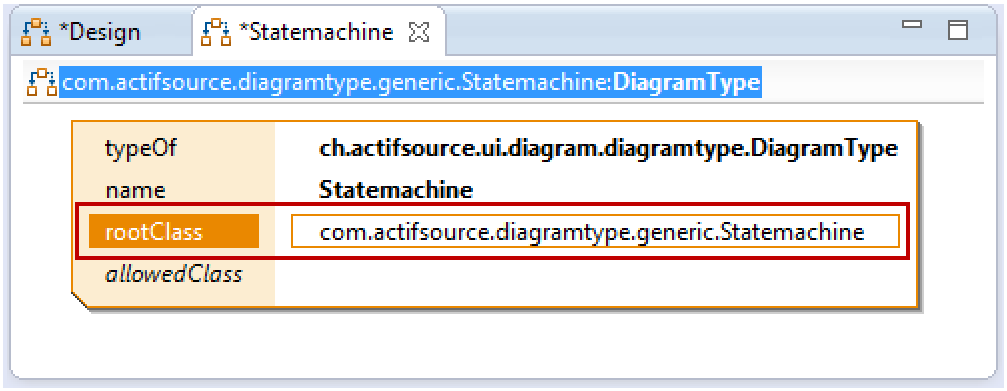
Part II: Create a Diagram Type

- Create a user defined diagram type to display and edit domain specific diagrams

Create a Diagram Type



ⓘ Create a new **<u>DiagramType</u>** named <u>Statemachine</u> in the **Package** *generic*

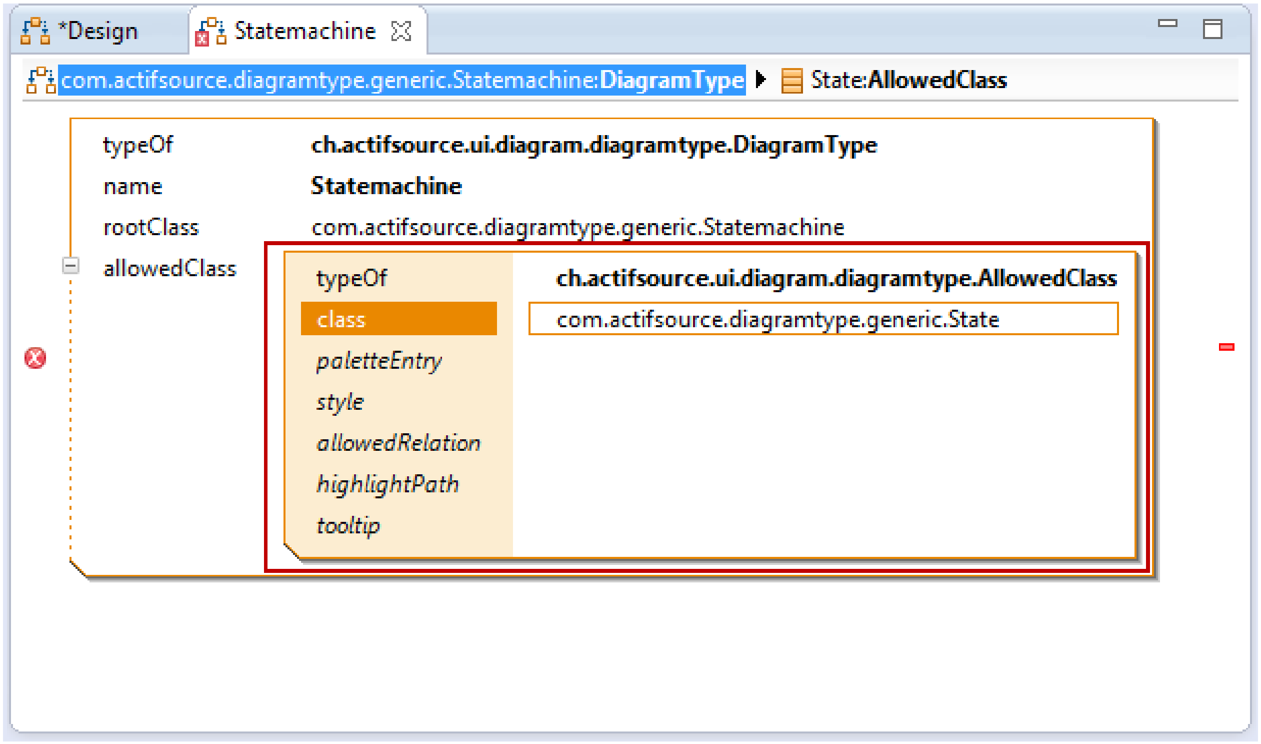


com.actifsource.diagramtype.generic.Statemachine:DiagramType

typeOf	**ch.actifsource.ui.diagram.diagramtype.DiagramType**
name	**Statemachine**
rootClass	com.actifsource.diagramtype.generic.Statemachine
allowedClass	

- The **Root Class** defines the **Resource** which contains the elements that shall be managed on your domain diagram

↳ Select Statemachine as **Root Class**

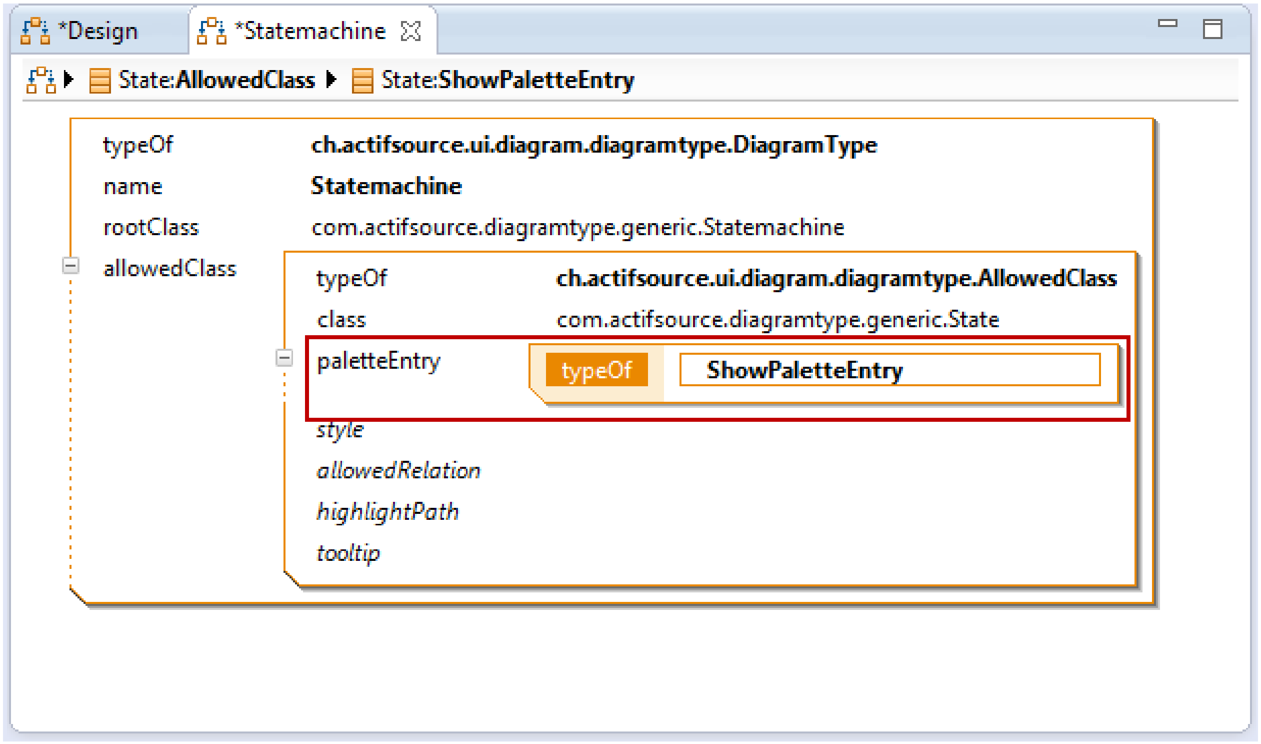

*Design | Statemachine

com.actifsource.diagramtype.generic.Statemachine:DiagramType ▸ State:AllowedClass

typeOf	**ch.actifsource.ui.diagram.diagramtype.DiagramType**
name	**Statemachine**
rootClass	com.actifsource.diagramtype.generic.Statemachine
allowedClass	

typeOf	**ch.actifsource.ui.diagram.diagramtype.AllowedClass**
class	com.actifsource.diagramtype.generic.State
paletteEntry	
style	
allowedRelation	
highlightPath	
tooltip	

- The **Allowed Class** defines all the **Resources** which shall be managed on your domain diagram
- Select State as **Allowed Class** since we want to design a state machine



*Design | *Statemachine

State:AllowedClass ▸ State:ShowPaletteEntry

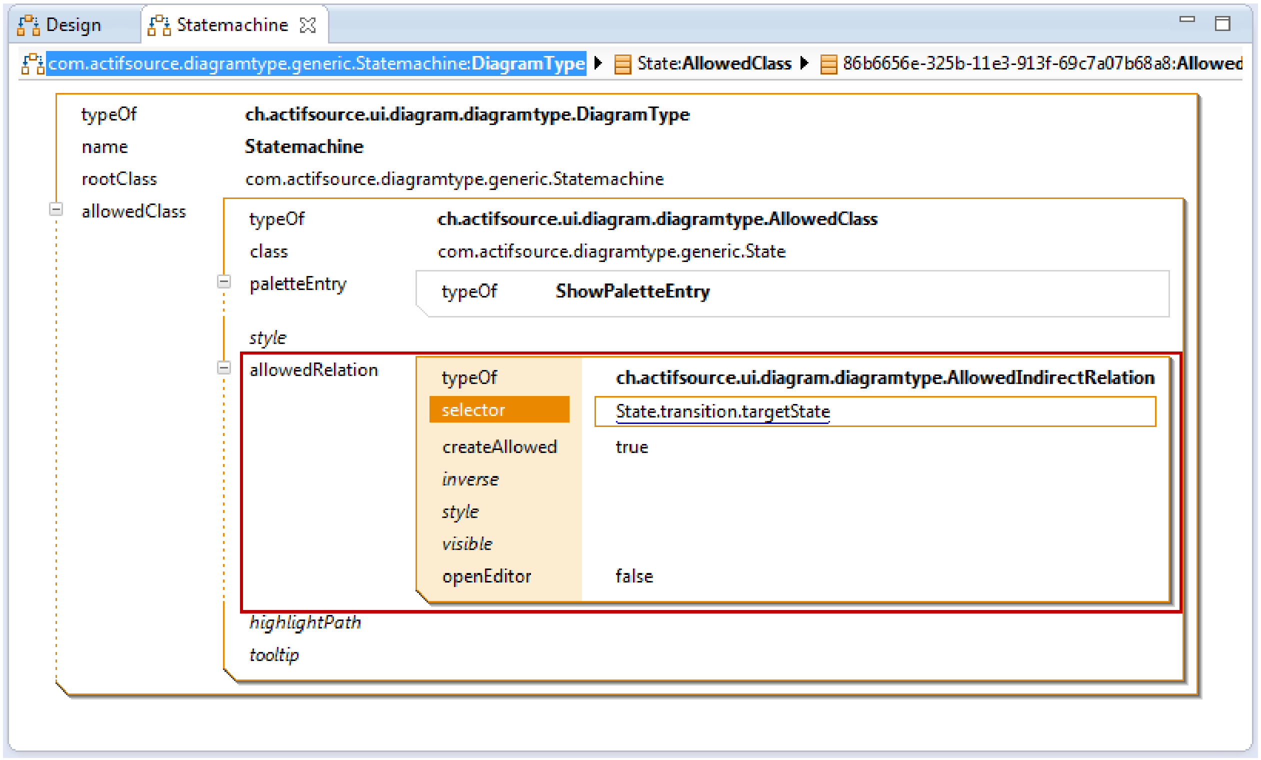
typeOf	**ch.actifsource.ui.diagram.diagramtype.DiagramType**
name	**Statemachine**
rootClass	com.actifsource.diagramtype.generic.Statemachine
allowedClass	

typeOf	**ch.actifsource.ui.diagram.diagramtype.AllowedClass**
class	com.actifsource.diagramtype.generic.State
paletteEntry	typeOf **ShowPaletteEntry**
style	
allowedRelation	
highlightPath	
tooltip	

- Allowed Classes might be created using the domain diagram editor via a Palette Tool
  - HidePaletteEntry: No Palette Entry to create this Allowed Class
  - ShowPaletteEntry: Palette Entry named as the Allowed Class
  - ShowRenamedPaletteEntry: Palette Entry named as defined

↳ Select the **PaletteEntry** **ShowPaletteEntry** or **ShowRenamedPaletteEntry**

Create a Diagram Type



com.actifsource.diagramtype.generic.Statemachine:DiagramType ▸ State:AllowedClass ▸ 86b6656e-325b-11e3-913f-69c7a07b68a8:Allowed

typeOf	ch.actifsource.ui.diagram.diagramtype.DiagramType
name	Statemachine
rootClass	com.actifsource.diagramtype.generic.Statemachine
allowedClass	

typeOf	ch.actifsource.ui.diagram.diagramtype.AllowedClass
class	com.actifsource.diagramtype.generic.State
paletteEntry	typeOf ShowPaletteEntry
style	
allowedRelation	
highlightPath	
tooltip	

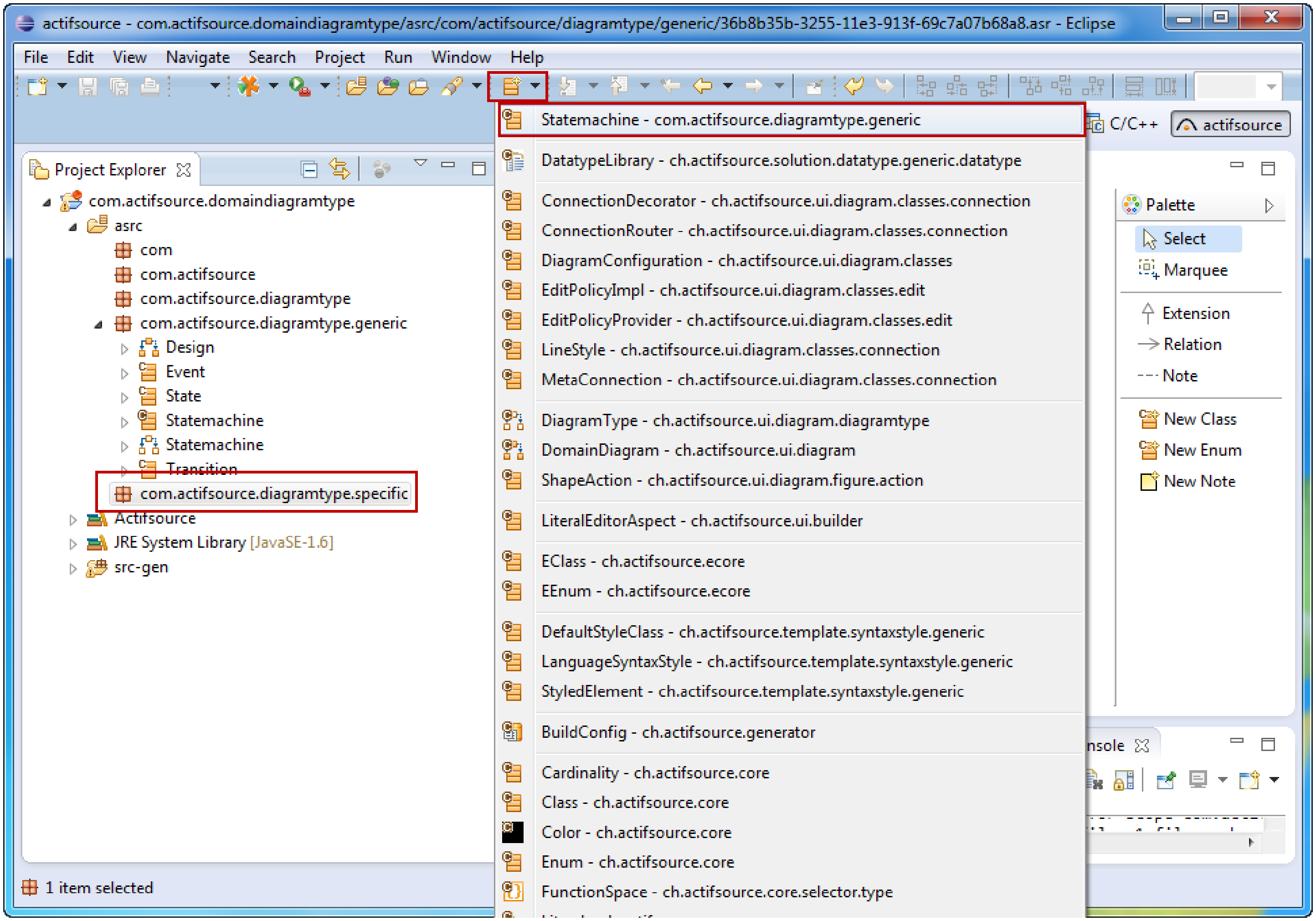
typeOf	ch.actifsource.ui.diagram.diagramtype.AllowedIndirectRelation
selector	State.transition.targetState
createAllowed	true
inverse	
style	
visible	
openEditor	false

- Allowed Relations are relations that shall be displayed on the domain diagram
  - AllowedDependencyRelation: Dotted Line that shows dependencies between components
  - AllowedDirectRelation: Direct relation between resources A and B
  - AllowedIndirectRelation: Indirect relation between resources A and B via X
- We like to see a transition from state to state just as a simple arrow
- Note that the resource Transition is displayed as an arrow
- Define the **Selector** `State.transition.targetState` for the **Indirect Relation**
- Define **openEditor** as false if you do not want to open the **Resource Editor** automatically after creating the transition via domain diagram editor.

Part III: Create a Domain Diagram

- Create a domain diagram based on the diagram type to edit the underlying root resource

Create a Domain Diagram



actifsource - com.actifsource.domaindiagramtype/asrc/com/actifsource/diagramtype/generic/36b8b35b-3255-11e3-913f-69c7a07b68a8.asr - Eclipse

File Edit View Navigate Search Project Run Window Help

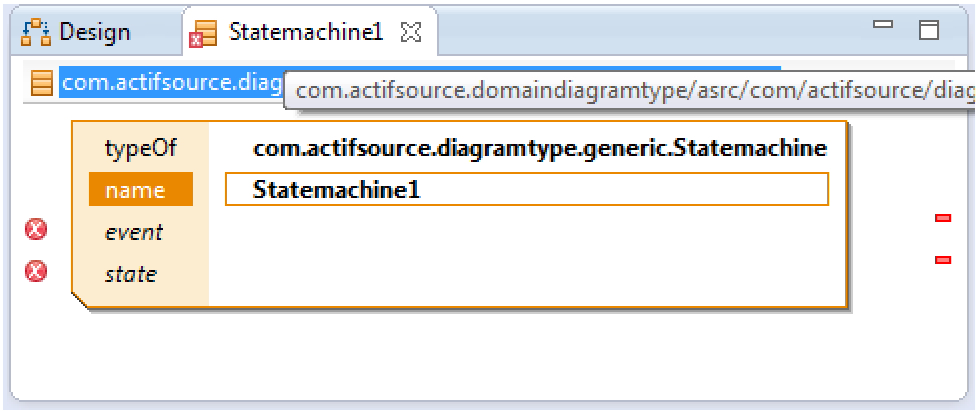
Project Explorer

- com.actifsource.domaindiagramtype
  - asrc
    - com
    - com.actifsource
    - com.actifsource.diagramtype
    - com.actifsource.diagramtype.generic
      - Design
      - Event
      - State
      - Statemachine
      - Statemachine
      - Transition
    - com.actifsource.diagramtype.specific
  - Actifsource
  - JRE System Library [JavaSE-1.6]
  - src-gen

1 item selected

- Statemachine - com.actifsource.diagramtype.generic
- DatatypeLibrary - ch.actifsource.solution.datatype.generic.datatype
- ConnectionDecorator - ch.actifsource.ui.diagram.classes.connection
- ConnectionRouter - ch.actifsource.ui.diagram.classes.connection
- DiagramConfiguration - ch.actifsource.ui.diagram.classes
- EditPolicyImpl - ch.actifsource.ui.diagram.classes.edit
- EditPolicyProvider - ch.actifsource.ui.diagram.classes.edit
- LineStyle - ch.actifsource.ui.diagram.classes.connection
- MetaConnection - ch.actifsource.ui.diagram.classes.connection
- DiagramType - ch.actifsource.ui.diagram.diagramtype
- DomainDiagram - ch.actifsource.ui.diagram
- ShapeAction - ch.actifsource.ui.diagram.figure.action
- LiteralEditorAspect - ch.actifsource.ui.builder
- EClass - ch.actifsource.ecore
- EEnum - ch.actifsource.ecore
- DefaultStyleClass - ch.actifsource.template.syntaxstyle.generic
- LanguageSyntaxStyle - ch.actifsource.template.syntaxstyle.generic
- StyledElement - ch.actifsource.template.syntaxstyle.generic
- BuildConfig - ch.actifsource.generator
- Cardinality - ch.actifsource.core
- Class - ch.actifsource.core
- Color - ch.actifsource.core
- Enum - ch.actifsource.core
- FunctionSpace - ch.actifsource.core.selector.type

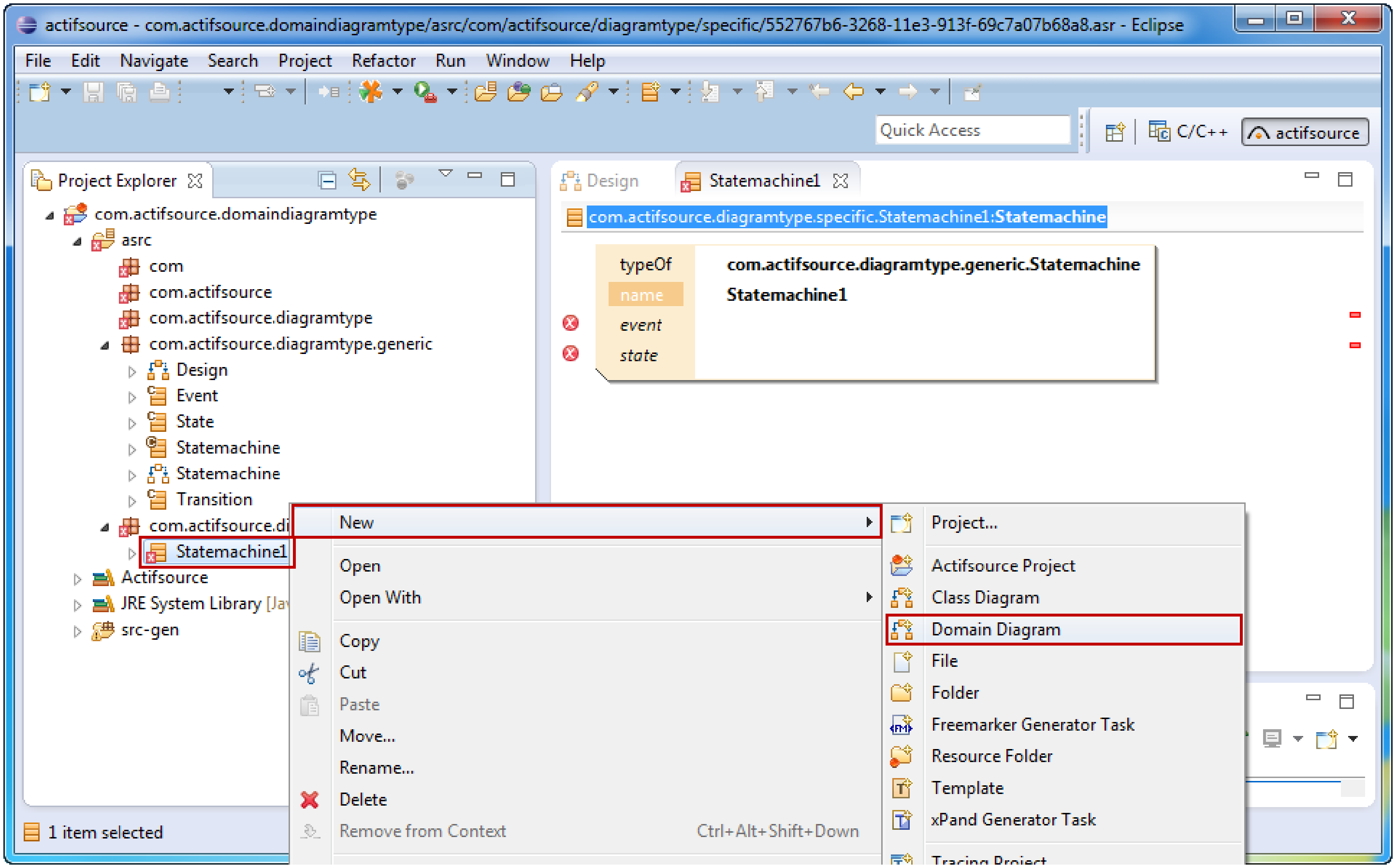
ⓘ Create a new <u>Statemachine</u> named <u>Statemachine1</u> in the **Package** *specific*



com.actifsource.diag
com.actifsource.domaindiagramtype/asrc/com/actifsource/diag

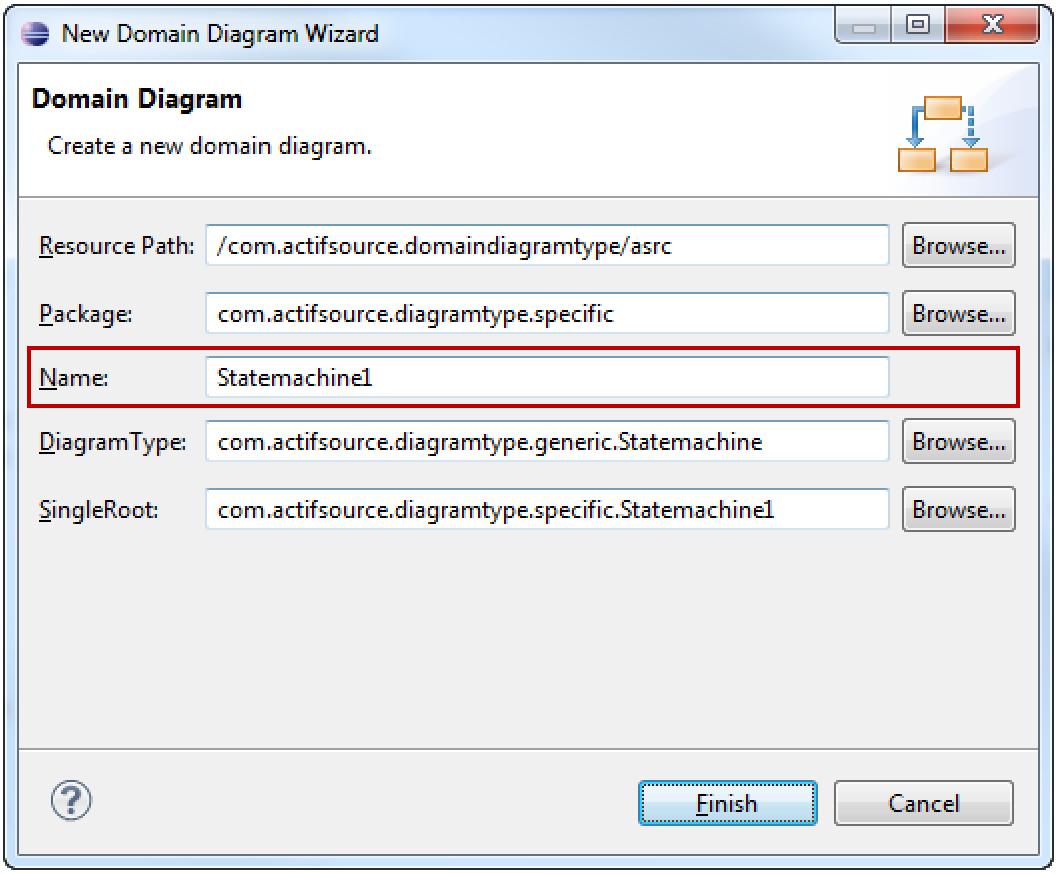
typeOf	com.actifsource.diagramtype.generic.Statemachine
name	Statemachine1
event	
state	

- ⓘ We could now define Event, State and Transition using the Resource Editor
- ⓘ Let’s now try the Domain Diagram Editor


actifsource - com.actifsource.domaindiagramtype/asrc/com/actifsource/diagramtype/specific/552767b6-3268-11e3-913f-69c7a07b68a8.asr - Eclipse
File Edit Navigate Search Project Refactor Run Window Help
Quick Access
C/C++
actifsource
Project Explorer
com.actifsource.domaindiagramtype
asrc
com
com.actifsource
com.actifsource.diagramtype
com.actifsource.diagramtype.generic
Design
Event
State
Statemachine
Statemachine
Transition
com.actifsource.di
Statemachine1
Actifsource
JRE System Library [Ja
src-gen
1 item selected
Design
Statemachine1
com.actifsource.diagramtype.specific.Statemachine1:Statemachine
typeOf
com.actifsource.diagramtype.generic.Statemachine
name
Statemachine1
event
state
New
Open
Open With
Copy
Cut
Paste
Move...
Rename...
Delete
Remove from Context Ctrl+Alt+Shift+Down
Project...
Actifsource Project
Class Diagram
Domain Diagram
File
Folder
Freemarker Generator Task
Resource Folder
Template
xPand Generator Task
Tracing Project

- ⓘ Create a new **Domain Diagram** named Statemachine1 for the Statemachine Statemachine1 in the **Package** *specific*
- ⓘ Use the context menu directly on the Statemachine Statemachine1

Create a Domain Diagram



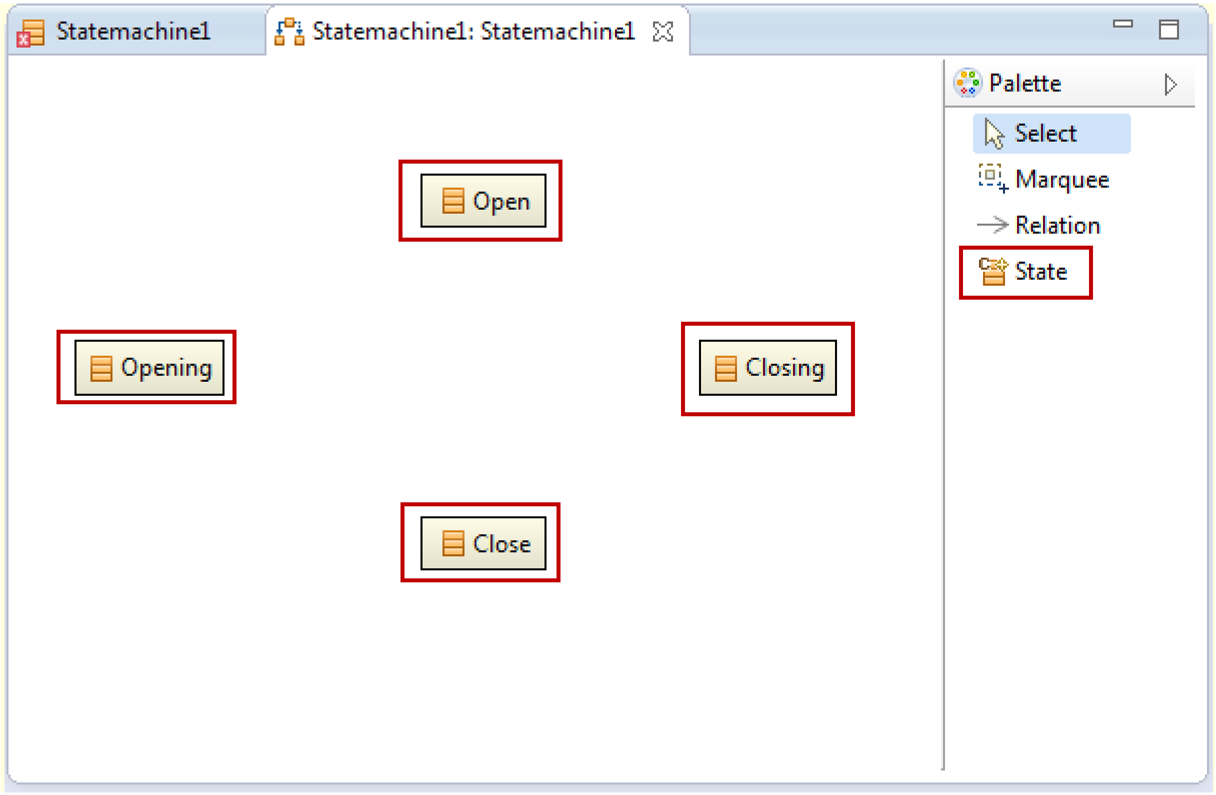
New Domain Diagram Wizard

Domain Diagram

Create a new domain diagram.

Resource Path:	/com.actifsource.domaindiagramtype/asrc
Package:	com.actifsource.diagramtype.specific
Name:	Statemachine1
DiagramType:	com.actifsource.diagramtype.generic.Statemachine
SingleRoot:	com.actifsource.diagramtype.specific.Statemachine1

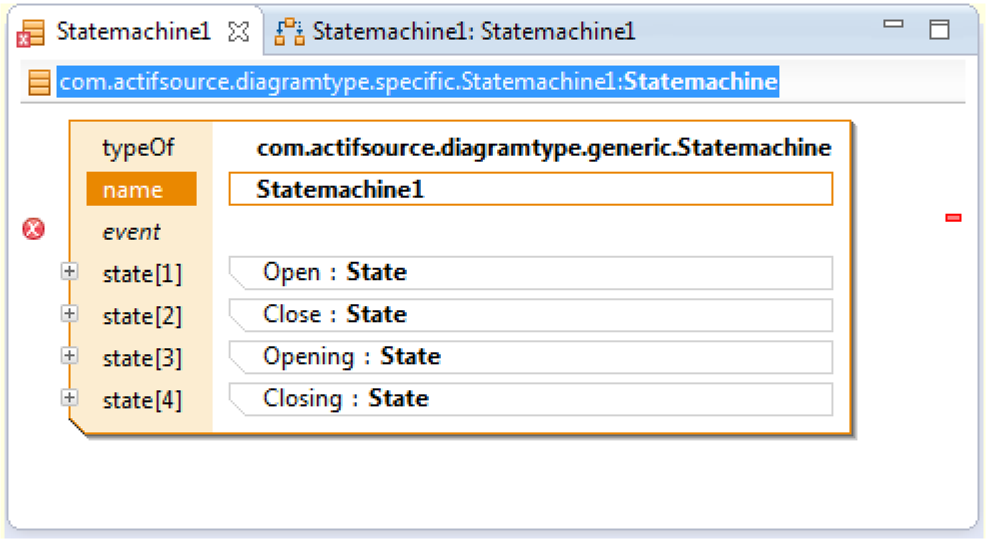
- ⓘ The domain diagram shall be named `Statemachine1`
- ⓘ **DiagramType** is detected automatically based on the **Root Resource**
- ⓘ **Single Root** is filled in automatically based on the context of *new Domain Diagram*



ⓘ Create new States named Open, Close, Opening and Closing using the **State Tool** from the **Palette**

ⓘ Note that the palette entry is influenced by *DiagramType.allowedClass.paletteEntry*

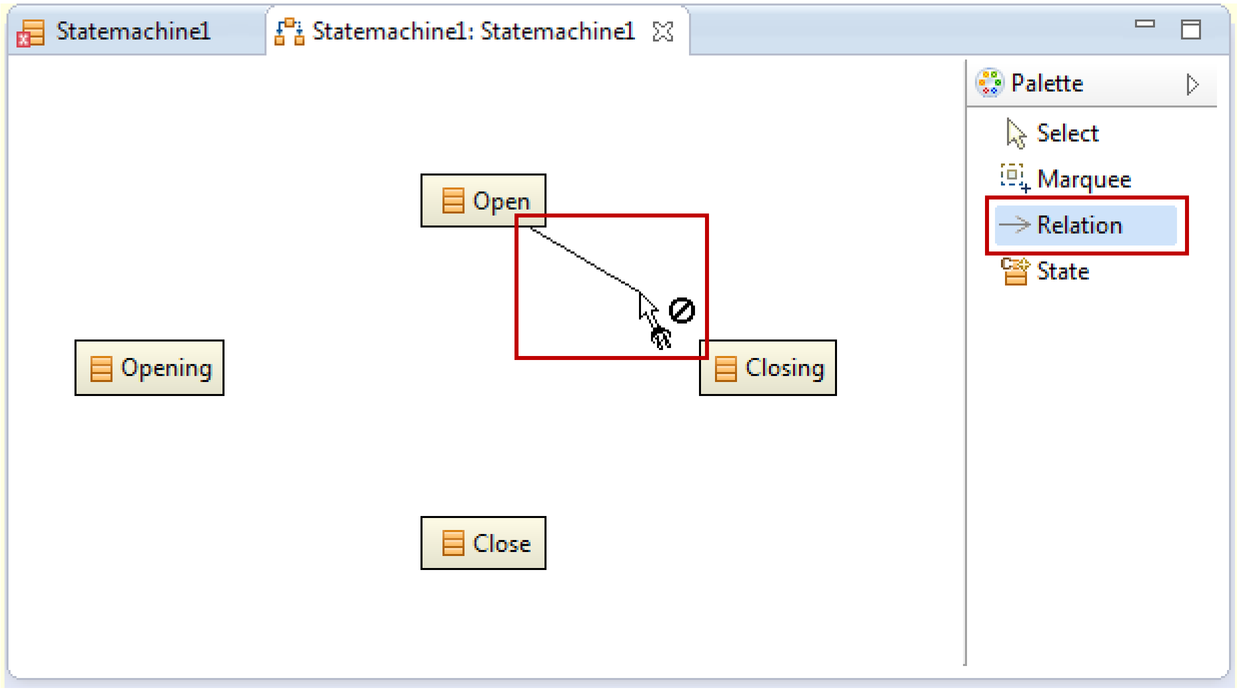
Create a Domain Diagram


Statemachine1 | Statemachine1: Statemachine1

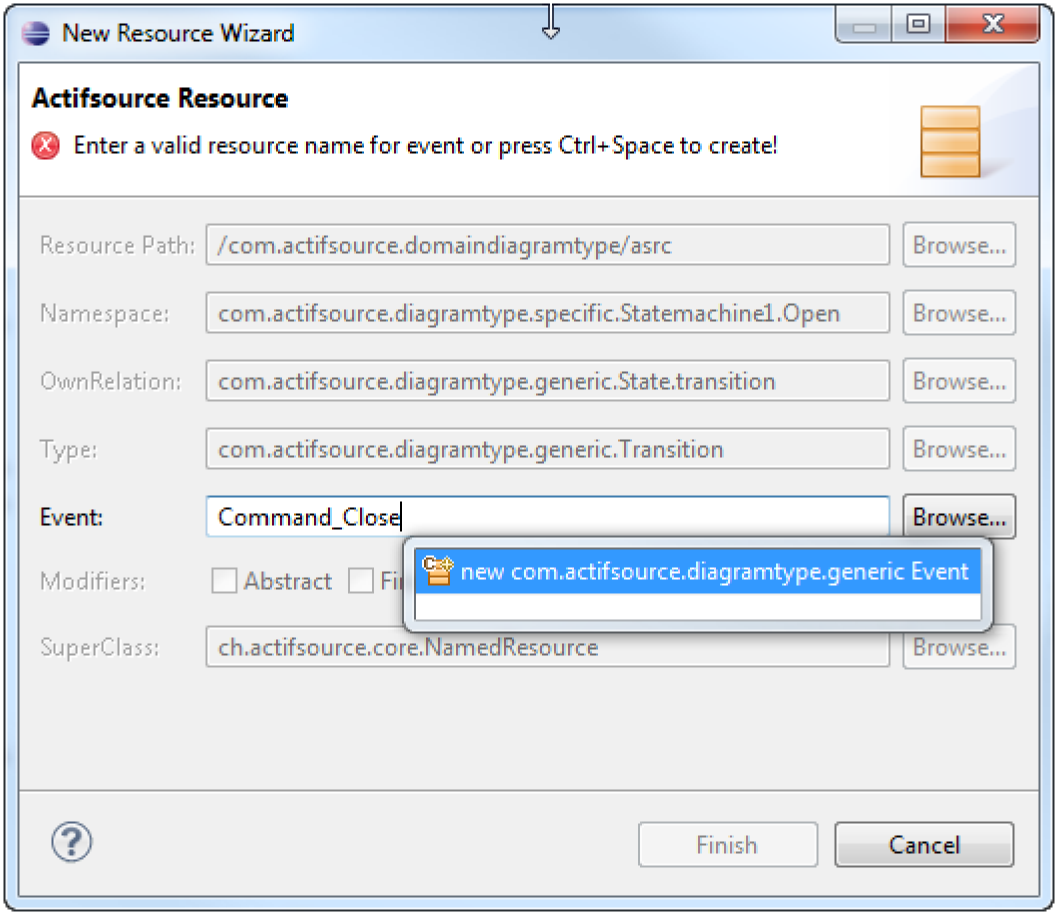
com.actifsource.diagramtype.specific.Statemachine1:Statemachine

typeOf	com.actifsource.diagramtype.generic.Statemachine
name	Statemachine1
event	
state[1]	Open : State
state[2]	Close : State
state[3]	Opening : State
state[4]	Closing : State

ⓘ Note that the dependent Statemachine Statemachine1 is modified accordingly



ⓘ Start adding Transitions using the **Relation Tool** from the **Palette**



Enter a valid resource name for event or press Ctrl+Space to create!

Resource Path: /com.actifsource.domaindiagramtype/asrc

Namespace: com.actifsource.diagramtype.specific.Statemachine1.Open

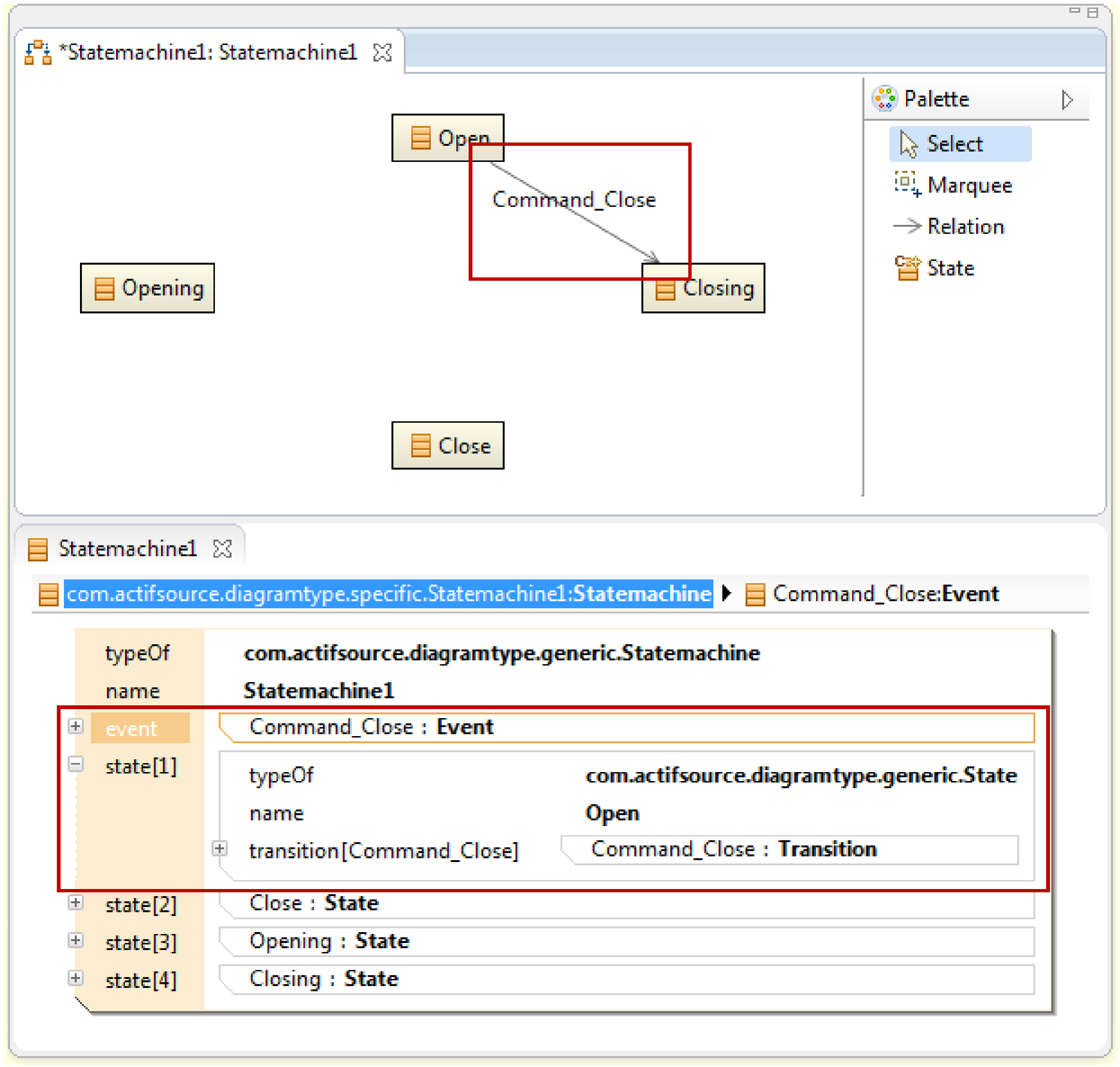
OwnRelation: com.actifsource.diagramtype.generic.State.transition

Type: com.actifsource.diagramtype.generic.Transition

Event: Command_Close

SuperClass: ch.actifsource.core.NamedResource

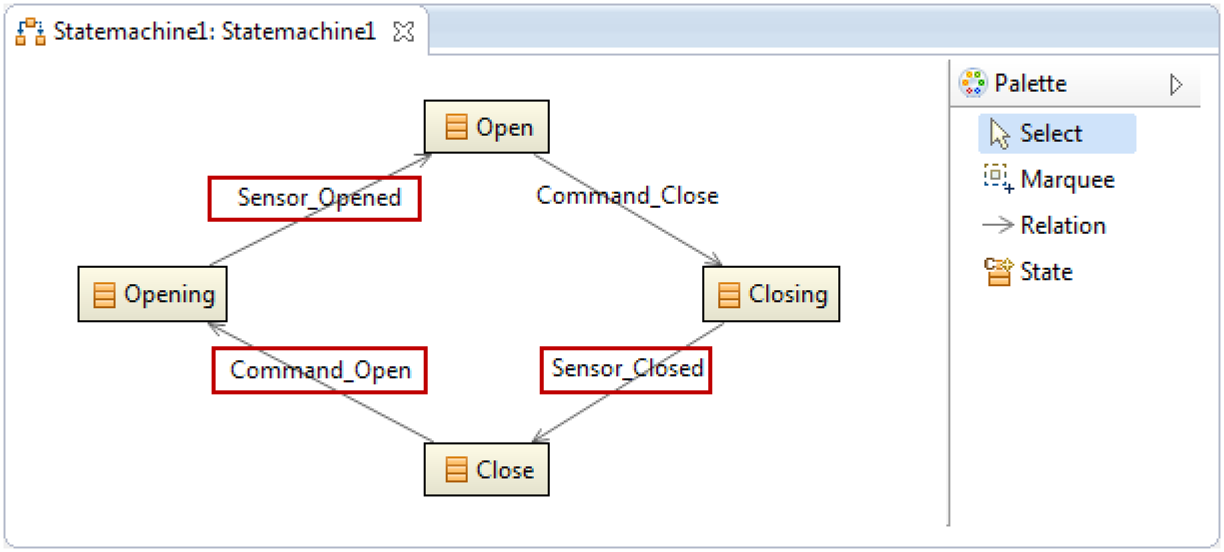
- ⓘ Note that Transitions are based on Events by using the **Decorating Relation**
- ⓘ Therefore you are asked for an Event when creating a new Transition
- ↳ Create a new Event called Command_Close using **Content Assist** (or choose an existing Event)



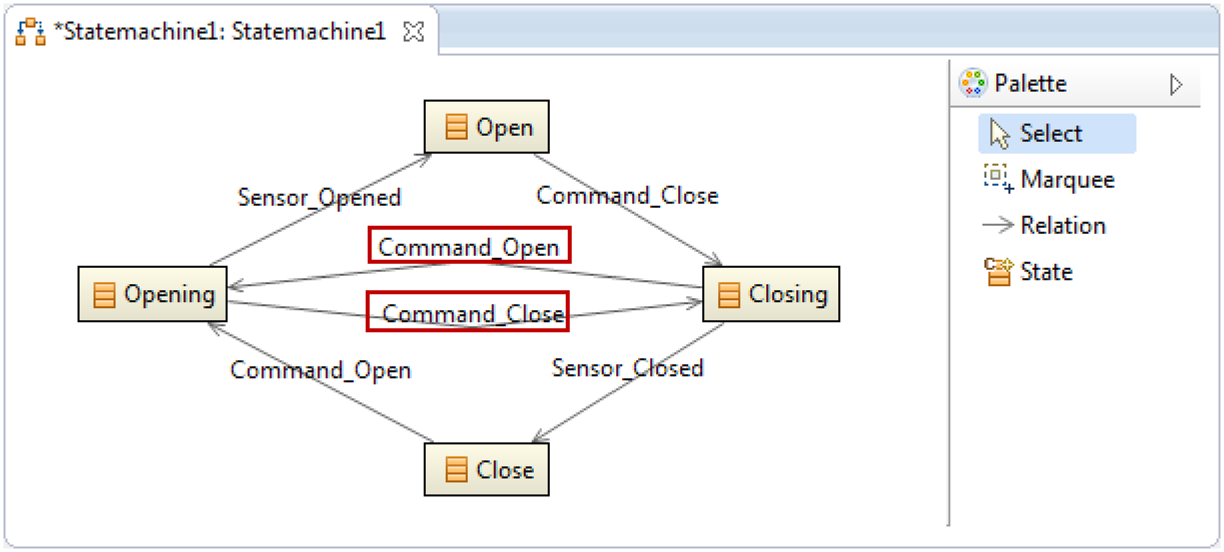
com.actifsource.diagramtype.specific.Statemachine1:Statemachine ▸ Command_Close:Event

typeOf	**com.actifsource.diagramtype.generic.Statemachine**
name	**Statemachine1**
event	Command_Close : **Event**
state[1]	typeOf **com.actifsource.diagramtype.generic.State**; name **Open**; transition[Command_Close] Command_Close : **Transition**
state[2]	Close : **State**
state[3]	Opening : **State**
state[4]	Closing : **State**

- ⓘ The Event Command_Close has been created
- ⓘ The Transition based on the Event Command_Close with targetState Closing has been created



↳ Add some more Transitions based on the *new* Events Command_Open, Sensor_Closed, Sensor_Opened



↳ Add two more Transitions based on the *existing* Events Command_Close and Command_Open

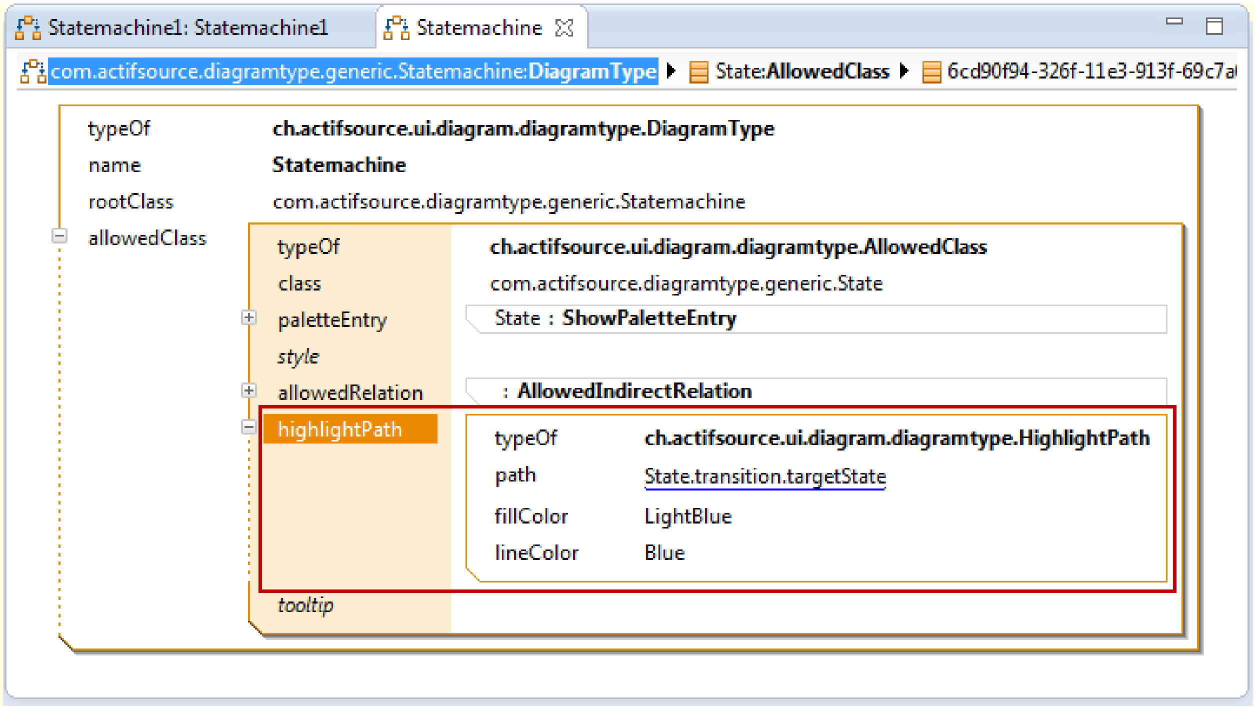
ⓘ Use **Context Assist** to choose the existing Events

Part IV:

Define a Highlight Path

ⓘ Define a highlight path from node to node over any edge

Defining a Highlight Path


Statemachine1: Statemachine1 | Statemachine

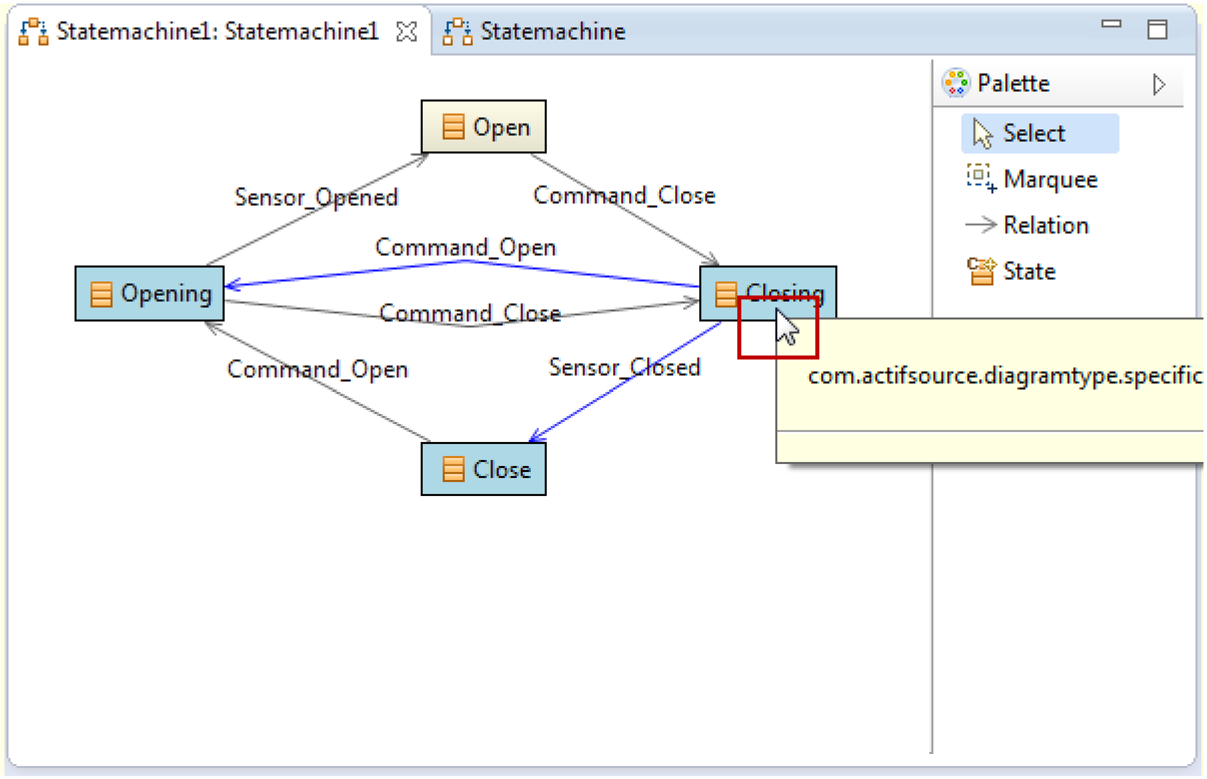
com.actifsource.diagramtype.generic.Statemachine:DiagramType ▸ State:AllowedClass ▸ 6cd90f94-326f-11e3-913f-69c7a

typeOf	**ch.actifsource.ui.diagram.diagramtype.DiagramType**	
name	**Statemachine**	
rootClass	com.actifsource.diagramtype.generic.Statemachine	
allowedClass	typeOf	**ch.actifsource.ui.diagram.diagramtype.AllowedClass**
	class	com.actifsource.diagramtype.generic.State
	paletteEntry	State : **ShowPaletteEntry**
	style	
	allowedRelation	: **AllowedIndirectRelation**
	highlightPath	typeOf **ch.actifsource.ui.diagram.diagramtype.HighlightPath**; path State.transition.targetState; fillColor LightBlue; lineColor Blue
	tooltip	

ⓘ Let’s see where transitions are leading to from source to target state

↳ Define a **HighlightPath** for the **Allowed Class** State as shown in the example above

Defining a Highlight Path



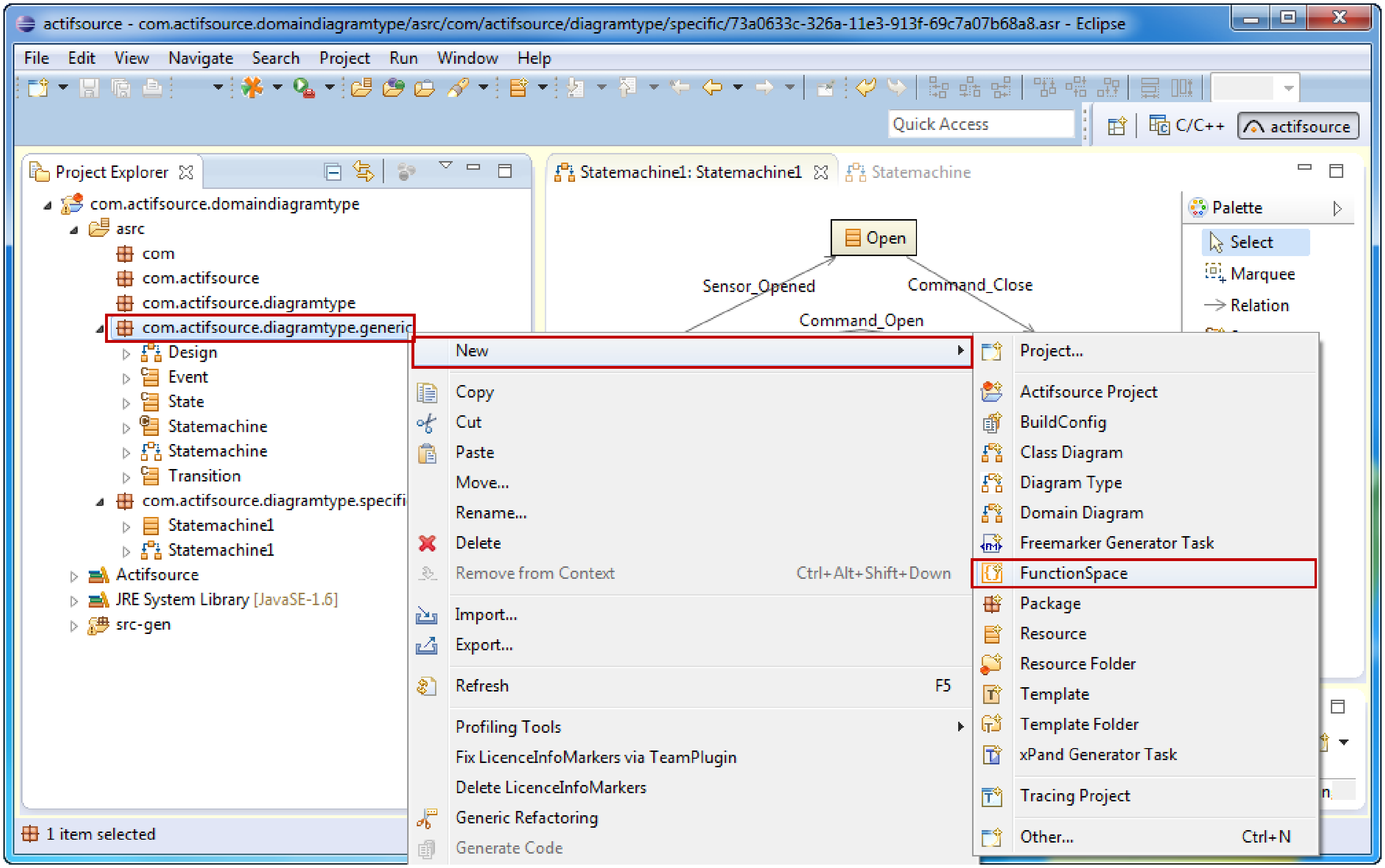
com.actifsource.diagramtype.specific

- Hoover your cursor on any State to activate the **HighlightPath** from State via Transition to TargetState

Part V: Define a Tooltip

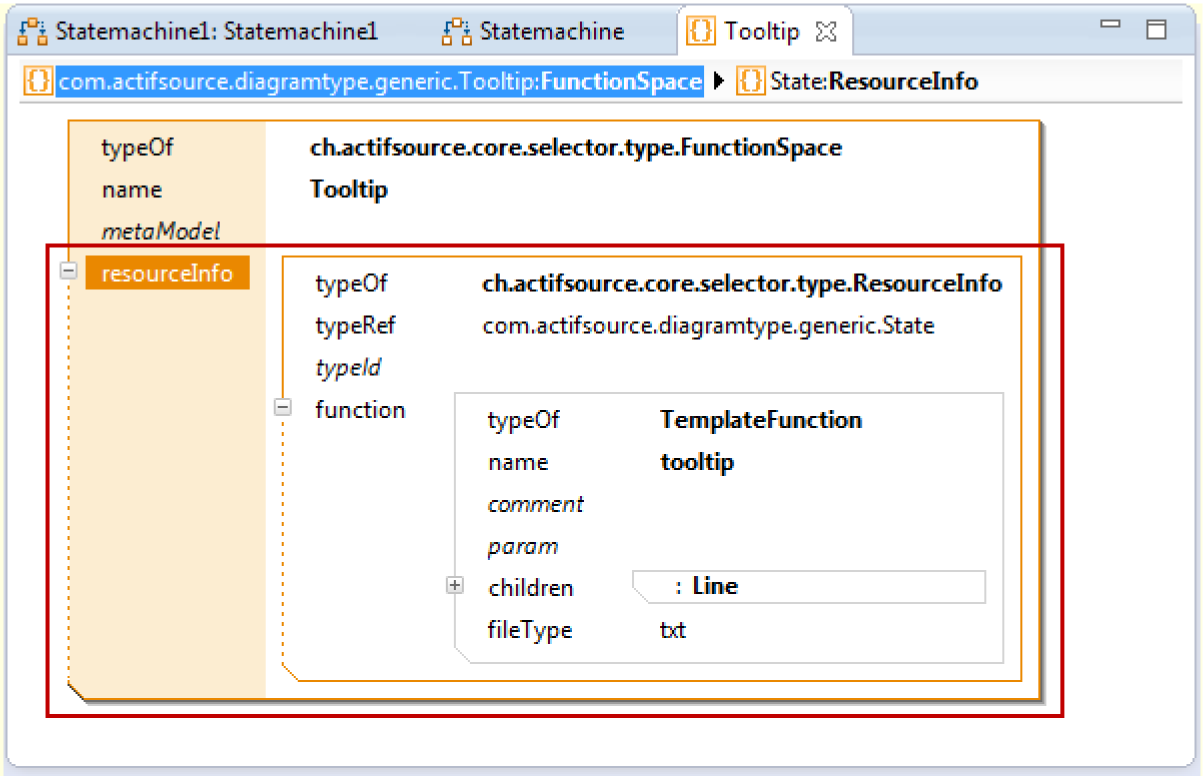
ⓘ Define a tooltip for elements on the domain diagram

Defining a Tooltip



ⓘ Create a new **FunctionSpace** named Tooltip in the **Package** *generic*

Defining a Tooltip



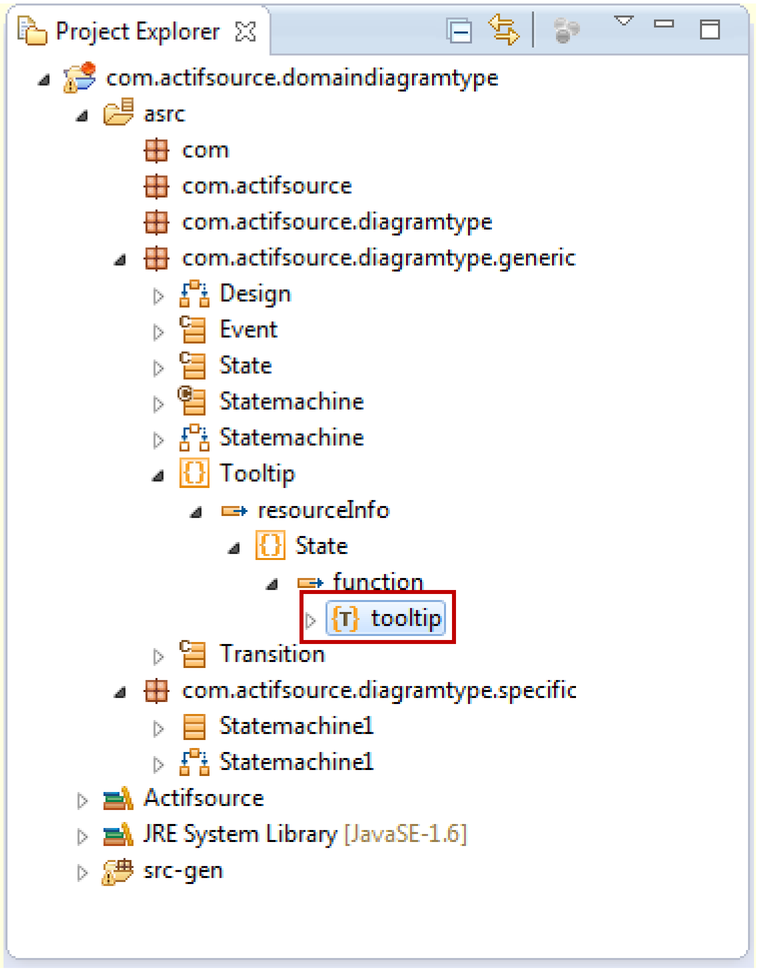
com.actifsource.diagramtype.generic.Tooltip:FunctionSpace ▸ State:ResourceInfo

typeOf	ch.actifsource.core.selector.type.FunctionSpace
name	Tooltip
metaModel	
resourceInfo	

typeOf	ch.actifsource.core.selector.type.ResourceInfo
typeRef	com.actifsource.diagramtype.generic.State
typeId	
function	

typeOf	TemplateFunction
name	tooltip
comment	
param	
children	: Line
fileType	txt

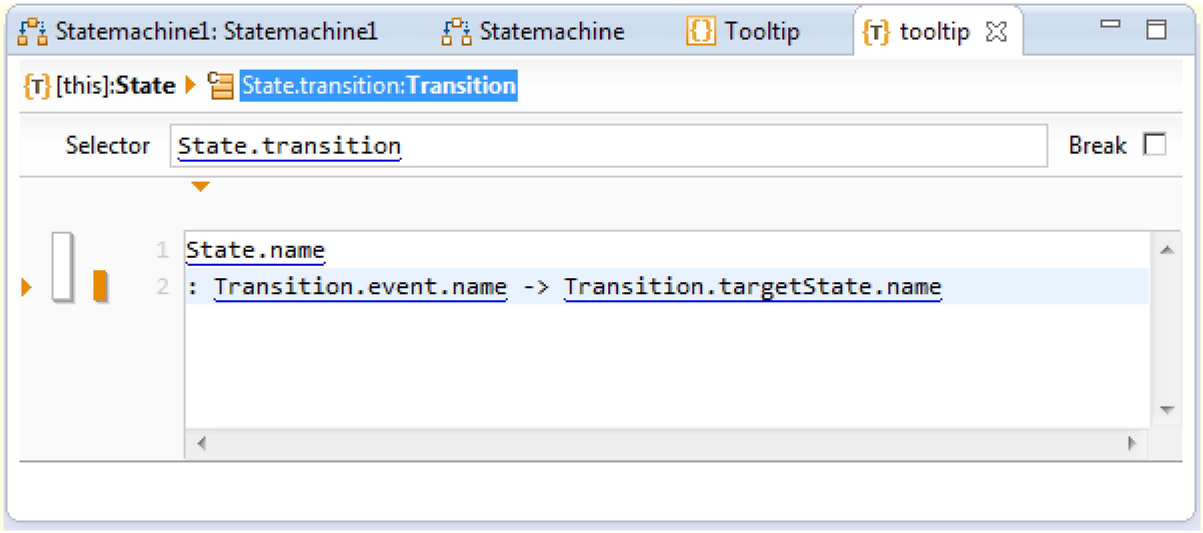
- Create a new **ResourceInfo** with **typeRef** State
- Create a new **TemplateFunction** named `tooltip`


Project Explorer

com.actifsource.domaindiagramtype
- asrc
  - com
  - com.actifsource
  - com.actifsource.diagramtype
  - com.actifsource.diagramtype.generic
    - Design
    - Event
    - State
    - Statemachine
    - Statemachine
    - Tooltip
      - resourceInfo
        - State
          - function
            - tooltip
    - Transition
  - com.actifsource.diagramtype.specific
    - Statemachine1
    - Statemachine1
- Actifsource
- JRE System Library [JavaSE-1.6]
- src-gen

↳ Open the **Template Editor** by double clicking Tooltip.State.tooltip in the **Project Explorer**

Defining a Tooltip

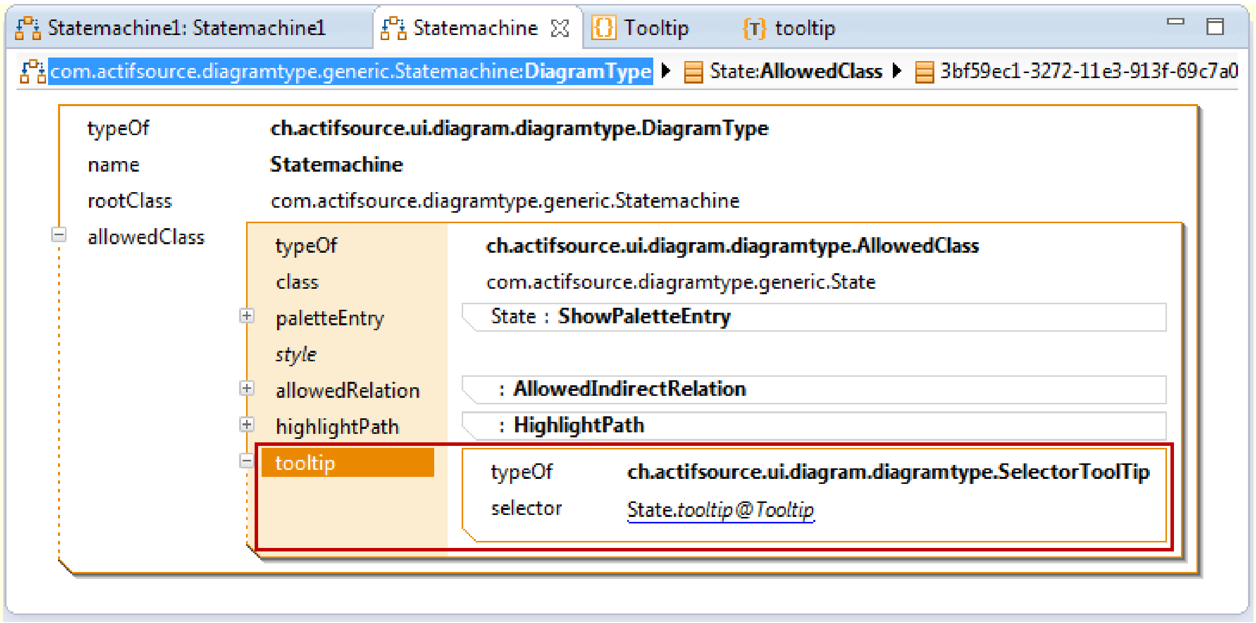


Selector: State.transition

```
State.name
: Transition.event.name -> Transition.targetState.name
```

- Print the State.name
- For every Transition in State: Print Transition.event.name
- For every Transition in State: Print Transition.targetState.name

Defining a Tooltip



com.actifsource.diagramtype.generic.Statemachine:DiagramType ▸ State:AllowedClass ▸ 3bf59ec1-3272-11e3-913f-69c7a0

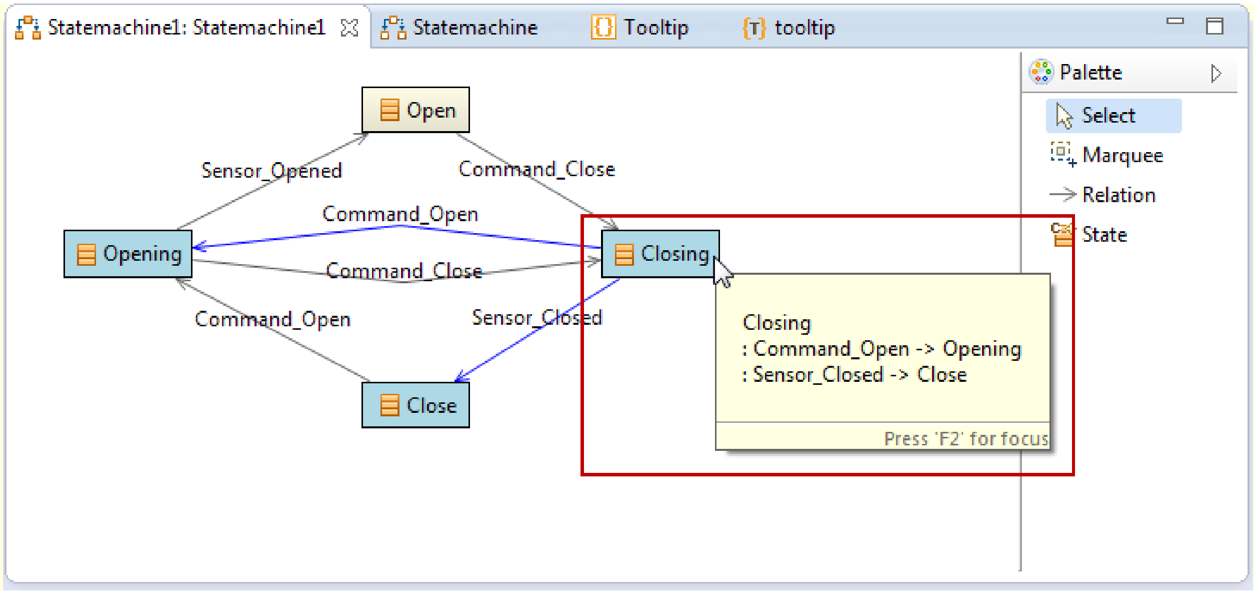
typeOf	ch.actifsource.ui.diagram.diagramtype.DiagramType
name	Statemachine
rootClass	com.actifsource.diagramtype.generic.Statemachine
allowedClass	

typeOf	ch.actifsource.ui.diagram.diagramtype.AllowedClass
class	com.actifsource.diagramtype.generic.State
paletteEntry	State : ShowPaletteEntry
style	
allowedRelation	: AllowedIndirectRelation
highlightPath	: HighlightPath
tooltip	

typeOf	ch.actifsource.ui.diagram.diagramtype.SelectorToolTip
selector	State.*tooltip@Tooltip*

↳ Use the **TemplateFunction** tooltip@Tooltip by using a **SelectorTooltip** in the **Diagram Type**

Defining a Tooltip



Closing
: Command_Open -> Opening
: Sensor_Closed -> Close

↳ Hoover your cursor on any State to activate the **Tooltip**