



How to Generate Code from State Machine Diagram?

Written Date : June 24, 2015

A [state machine](#) consists of a number of states and the transition between states. To create a state machine, you start by creating a controller class, and then create a sub-state machine diagram from the controller class. Moreover, you can generate source code based on the sub-state machine diagram.

Overview of this Tutorial

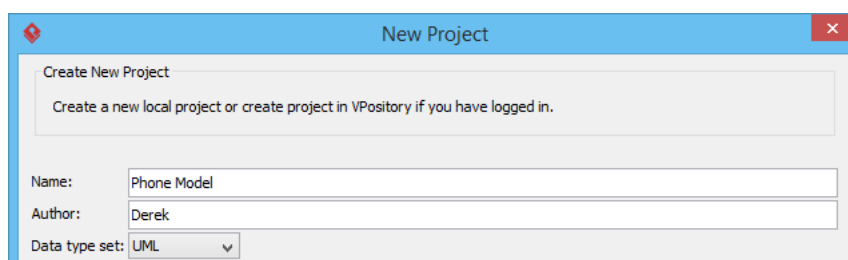
In this tutorial, we will show you how to model a controller class and its state machine. At the end, you will generate state machine code and play with the sample application. You will also export [SCXML](#) from your state machine.

In order to complete this tutorial you must have Visual Paradigm installed. You also need to have basic knowledge in [UML modeling](#) with Visual Paradigm.

Create a Project for This Tutorial

In order not to mess up your production data, we will create a new project for this tutorial. In this section, you are going to create such a project.

1. Select **Project > New** from the toolbar.
2. Enter *Phone Model* as project name.

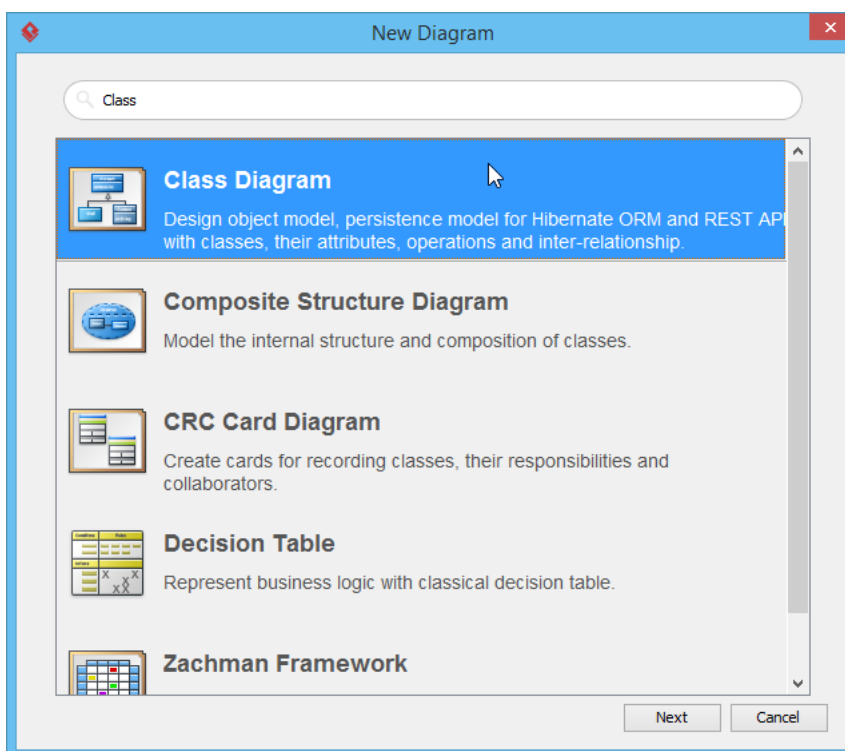


3. Click **Create Blank Project** to confirm the creation.

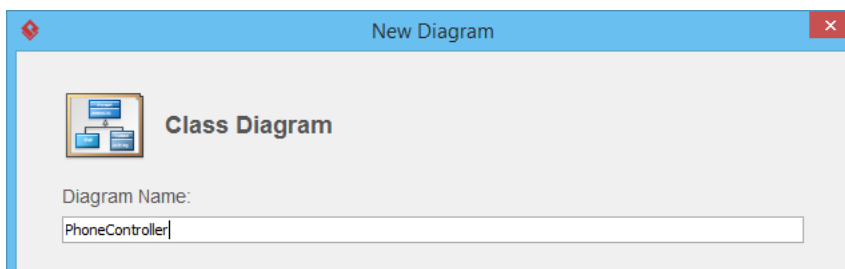
Creating State Machine

1. Create a class diagram first. Select **Diagram > New** from the toolbar.

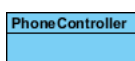
2. In the **New Diagram** window, select **Class Diagram** and click **Next**.



3. Enter *PhoneController* as diagram name.



4. Click **OK** to confirm the creation of class diagram.
5. Draw a *PhoneController* class in the class diagram.

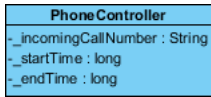


6. Add the following attributes into *PhoneController*. You can add attribute by right clicking on the class and selecting **Add > Attribute** from the popup menu.

Name	Type
_incomingCallNumber	String

_startTime	long
_endTime	long

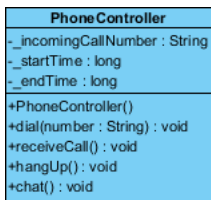
The class should look like this:



7. Add operations to the class. These operations will trigger state change. To add operation, right click on the class and select **Add > Operation** from the pop-up menu.

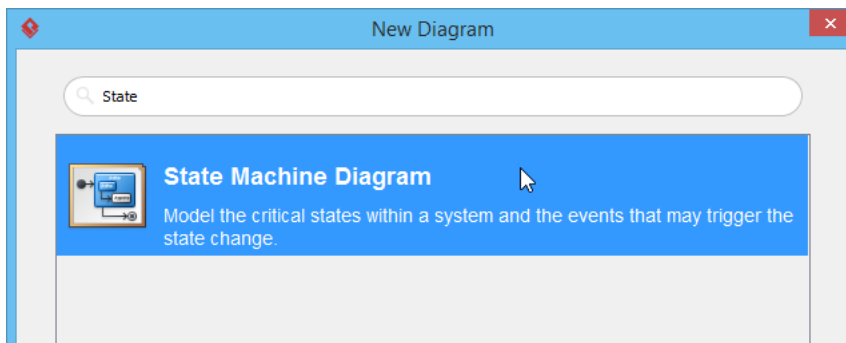
Name	Return Type
PhoneController	
dial(number : String)	void
receiveCall	void
hangUp	void
chat	void

The class should look like this:

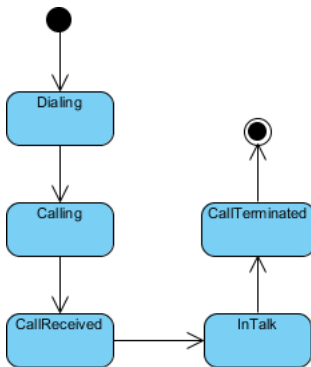


8. Now, we are going to draw the state machine for *PhoneController* class. Right click on *PhoneController* and select **Sub Diagram > New Diagram...** from the popup menu.

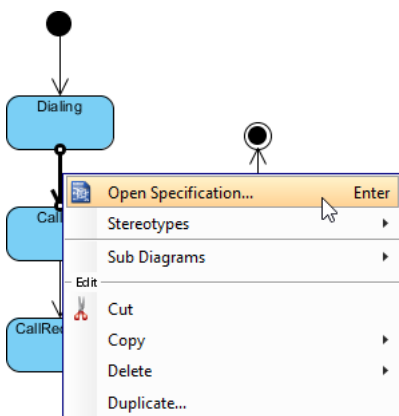
9. Select **State Machine Diagram** and click **Next**.



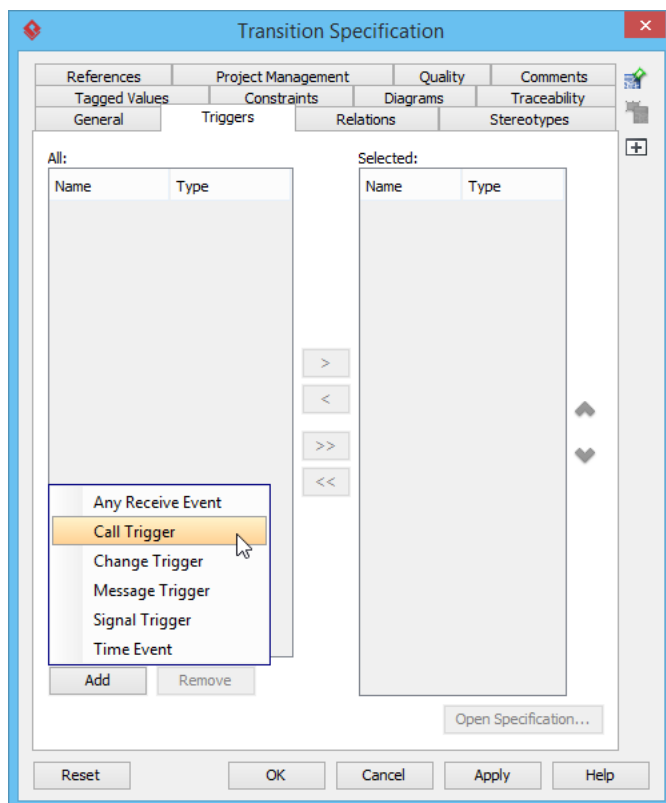
10. Keep the diagram name as-is and click **OK** to confirm diagram creation.
11. A state machine diagram is created with an initial node appears. Complete the diagram by drawing the states show in the following diagram.



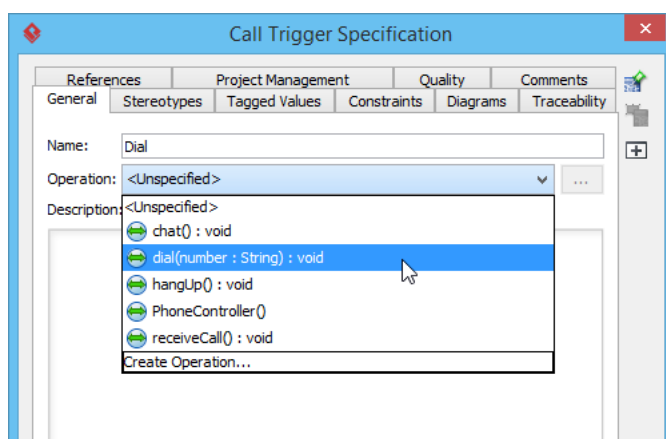
12. Now, add a trigger to transitions. Right click on the transition between *Dialing* state and *Calling* state. Select **Open Specification...** from the pop-up menu.



13. In the **Transition Specification** window, open **Triggers** tab. Click **Add** and select **Call Trigger** from the pop-up menu.



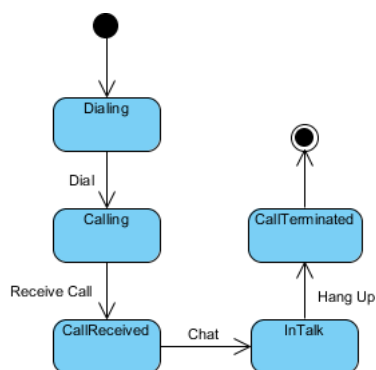
14. In the **Call Trigger Specification** window, enter *Dial* as trigger name. Select *dial(number : String) : void* from the drop down menu of **Operation**.



15. Click **OK**.
16. The trigger is selected for the transition.
17. Click **OK**.
18. Continue to add a few more triggers following the table below:

Transition	Trigger Name
Calling -> CallReceived	Receive Call
CallReceived -> InTalk	Chat
InTalk -> CallTerminated	Hang Up

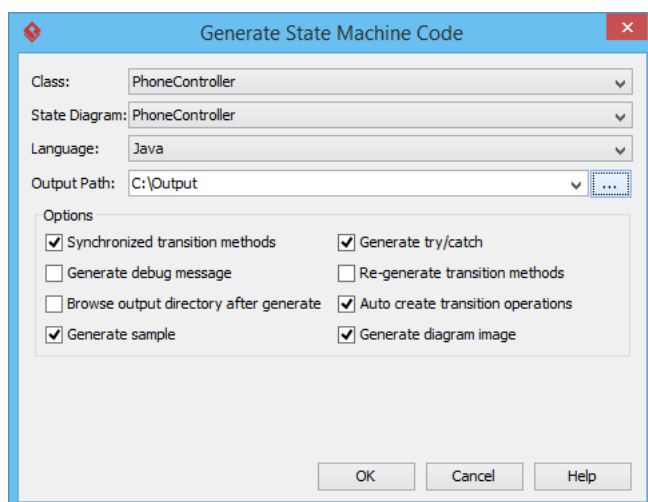
The completed state machine diagram is shown as follows:



Generating State Machine Code

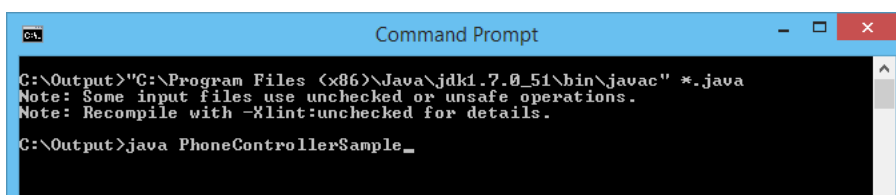
Now, let's generate state machine code from the project.

1. Select **Tools > Code > Generate State Machine Code** from the toolbar.
2. In the **Generate State Machine Code** window, specify the directory to store the generated source file.

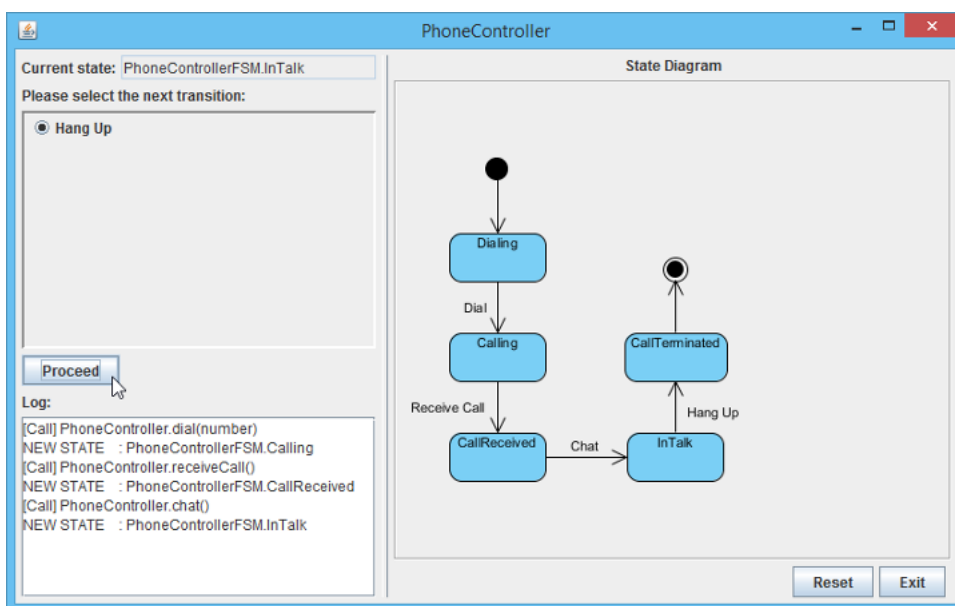


3. Click **OK**.

4. Compile the generated code and run the **PhoneControllerSample** class.



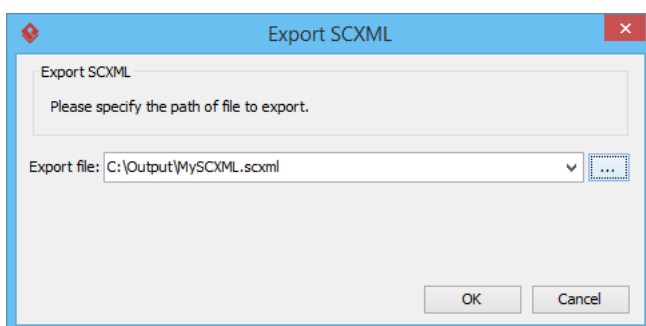
Run the sample application by clicking **Proceed** repeatedly and observe the change of states.



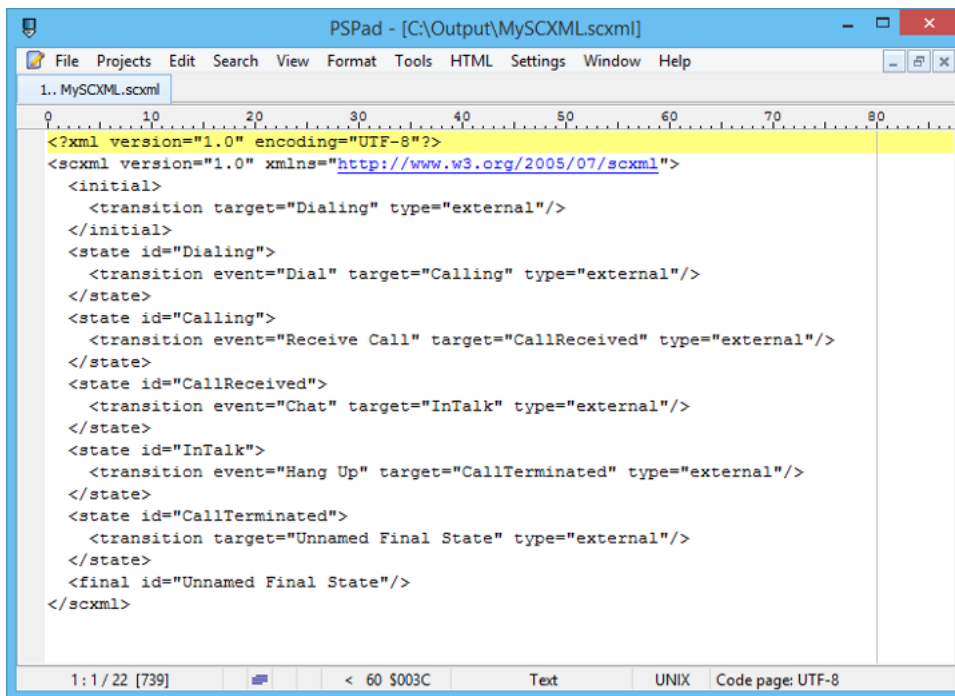
Generating SCXML

Now, let's generate SCXML from your state machine.

1. Right click on the background of your state machine diagram and select **Export ^gt; Export to SCXML...** from the popup menu.
2. In the **Export SCXML** dialog box, specify the filepath of the *.scxml file.



3. Click **OK**. The exported file should look like this one.



The screenshot shows a PSPad editor window titled "PSPad - [C:\Output\MySCXML.scxml]". The editor displays the following SCXML code:

```
<?xml version="1.0" encoding="UTF-8"?>
<scxml version="1.0" xmlns="http://www.w3.org/2005/07/scxml">
  <initial>
    <transition target="Dialing" type="external"/>
  </initial>
  <state id="Dialing">
    <transition event="Dial" target="Calling" type="external"/>
  </state>
  <state id="Calling">
    <transition event="Receive Call" target="CallReceived" type="external"/>
  </state>
  <state id="CallReceived">
    <transition event="Chat" target="InTalk" type="external"/>
  </state>
  <state id="InTalk">
    <transition event="Hang Up" target="CallTerminated" type="external"/>
  </state>
  <state id="CallTerminated">
    <transition target="Unnamed Final State" type="external"/>
  </state>
  <final id="Unnamed Final State"/>
</scxml>
```

The status bar at the bottom indicates "1: 1 / 22 [739]", "60 \$003C", "Text", "UNIX", and "Code page: UTF-8".

Resources

1. [Phone Model.vpp](#)

Related Links

- [Code Engineering features in Visual Paradigm](#)
- [State Machine Diagram feature description](#)



Visual Paradigm home page
(<https://www.visual-paradigm.com/>)

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