Table of Contents

Finite State Machine

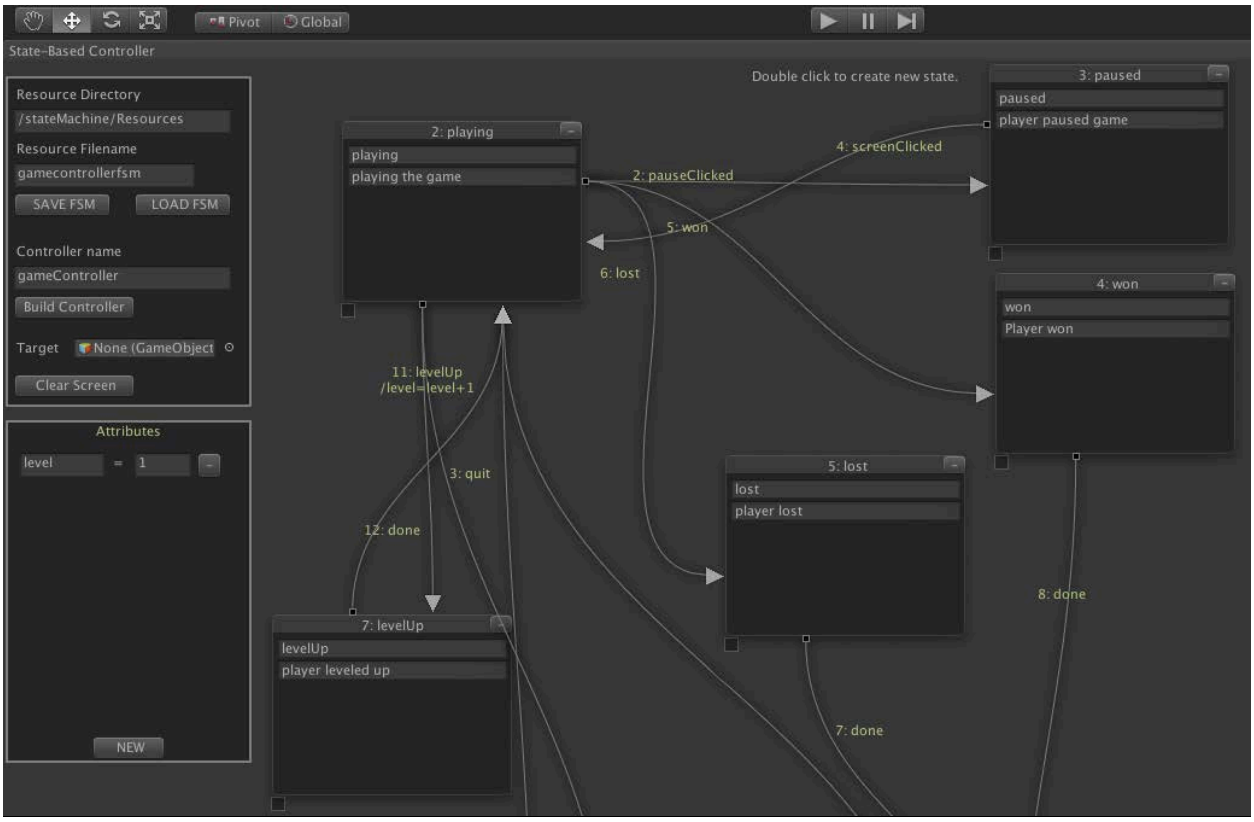
Description:

This is a straight forward and robust state based controller developed to provide programmers with a light weight option for implementing a finite state machine in their applications. It is easy to use and does not require the GUI builder if you do not want to use it.

Please note that this is not a visual scripting tool designed to allow non-programmers to graphically build applications. It is an asset that allows you to quickly and easily build the underlying application structure. As such it is not recommended for people unfamiliar with programing in Unity.

The state controller allows you to build state machine controllers for a wide range of applications. The controllers contain callbacks for each of the states that fire when the state in entered by the controller. The developer can then control the flow of the application simply by including method calls in these callbacks as well as by sending events as needed (such as button clicks) to the state machine.

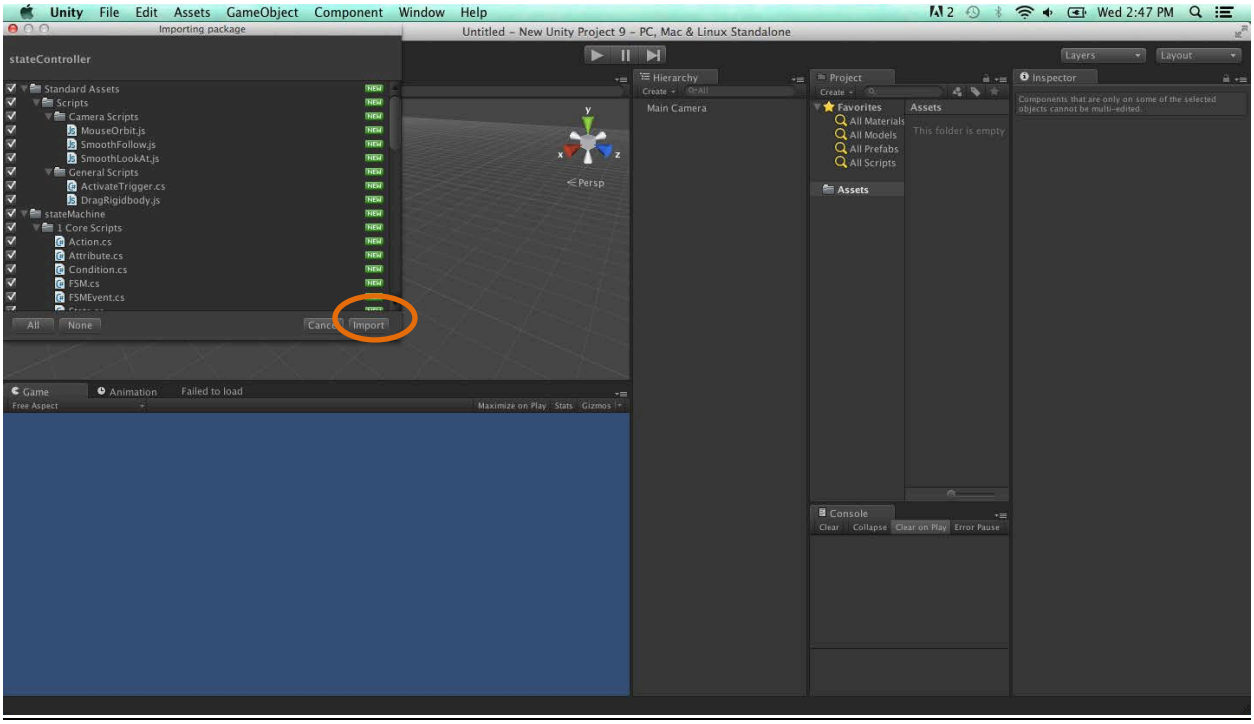
The conditions/attributes and actions greatly increase the flexibility of the controller to allow it to work for almost any type of problem. You can add multiple conditions (separated by a colon) and multiple actions (separated by a colon.) The actions and conditions are based on the attributes in the state machine.



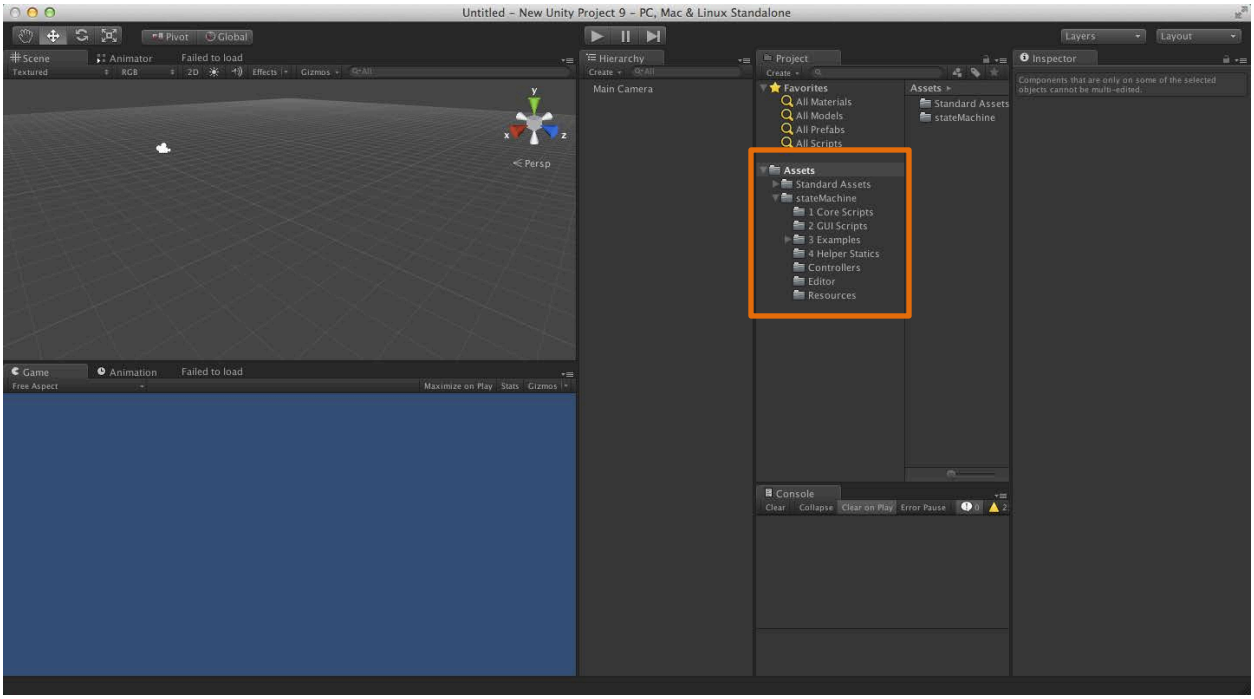
Getting Started:

Initial set-up and navigation

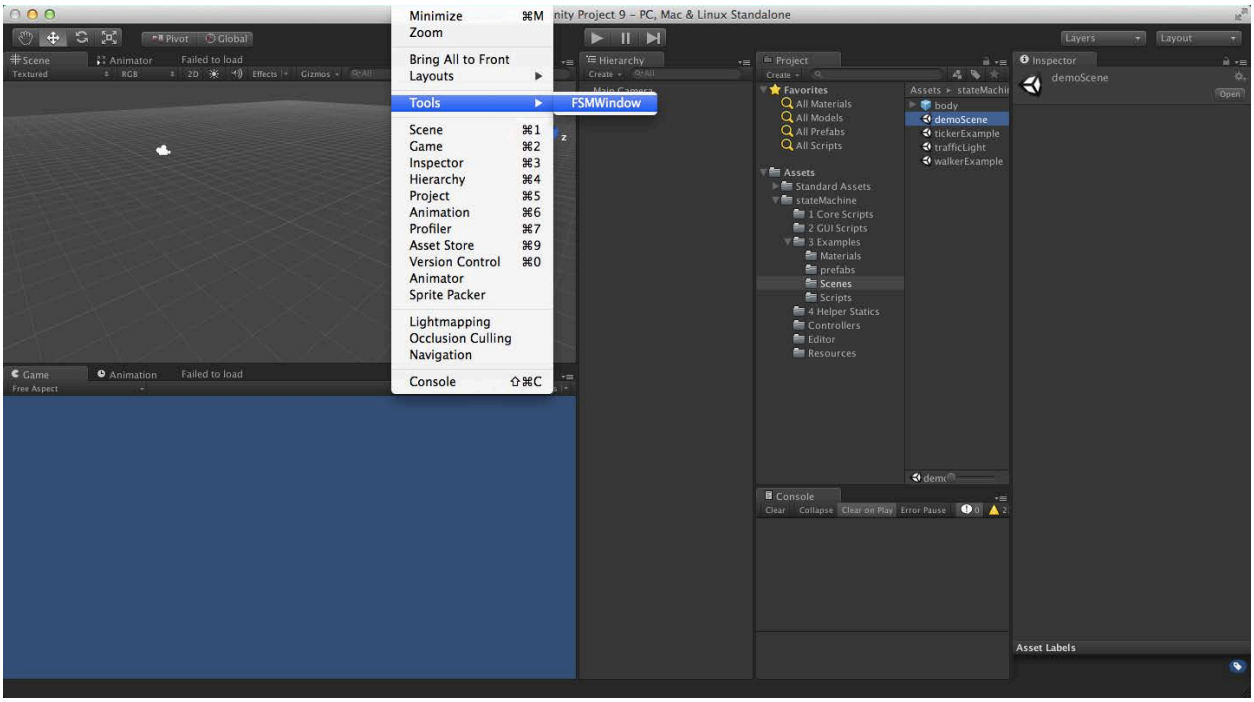
1. After you download the asset click import

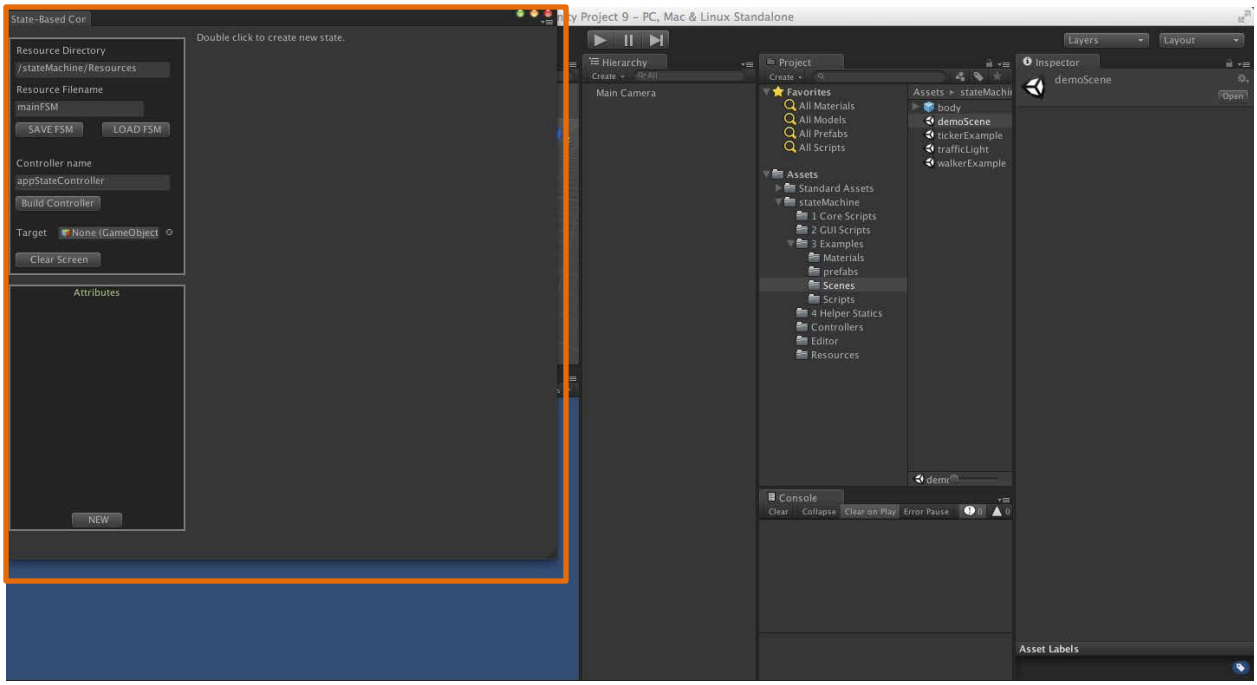


2. After you import the asset you will see two new folders in you assets folder



3. To bring up the state controller console click “Window”, click “Tools”, select “FSMWindow”

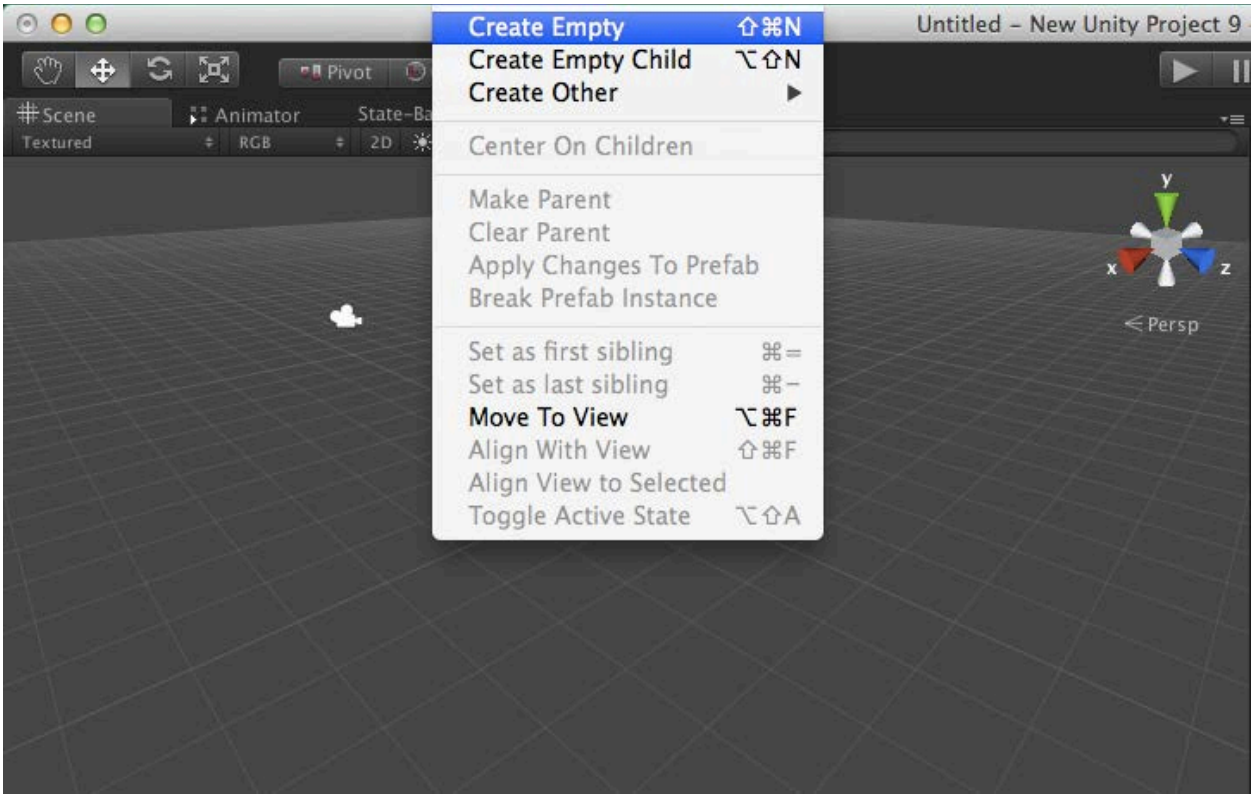




Steps to make a new controller

1. Start a new scene

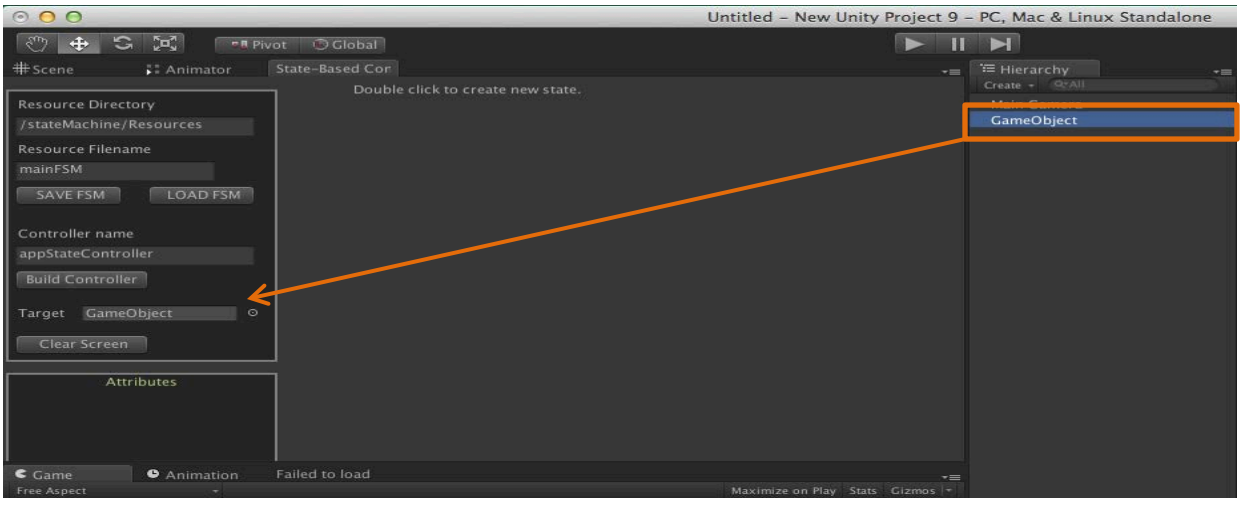
2. Make a blank Game Object that will be your controller



3. Rename your game object to something that makes sense for your domain

4. Select from the menu bar Window->Tools->FSMWindow

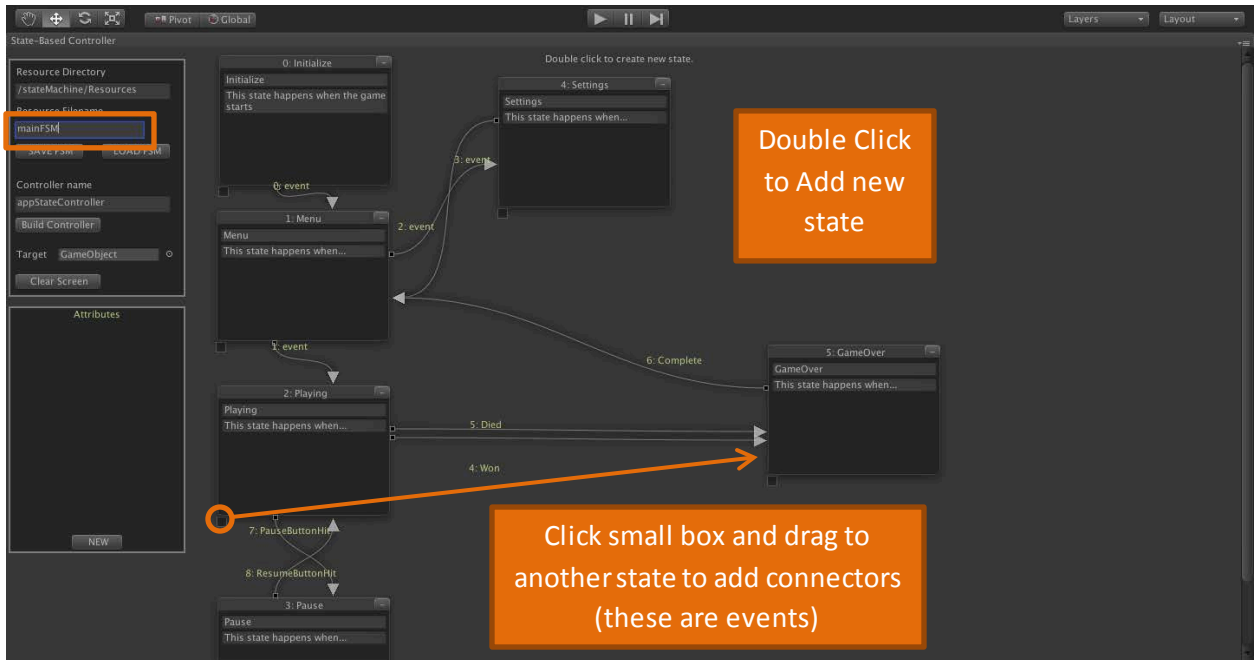
5. Drag your empty game object to the Target field on the left of the FSMWindow



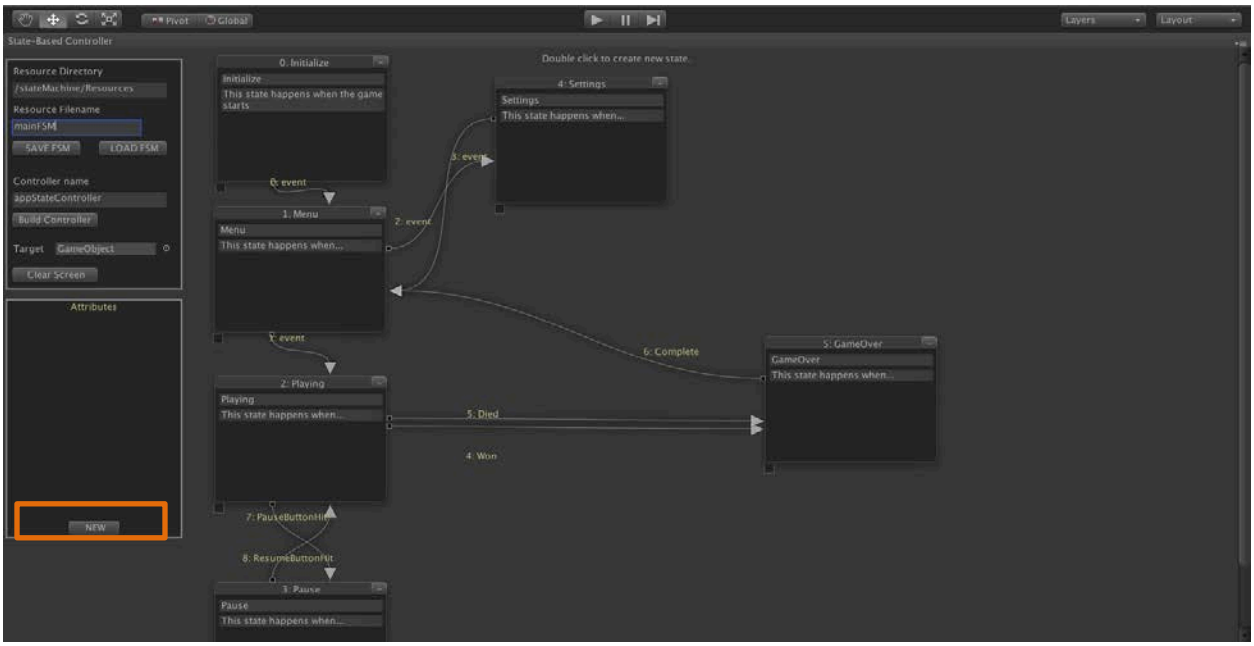
6. Rename Resource Filename to something that makes sense for this controller. This is the resource text file that will define this state machine.

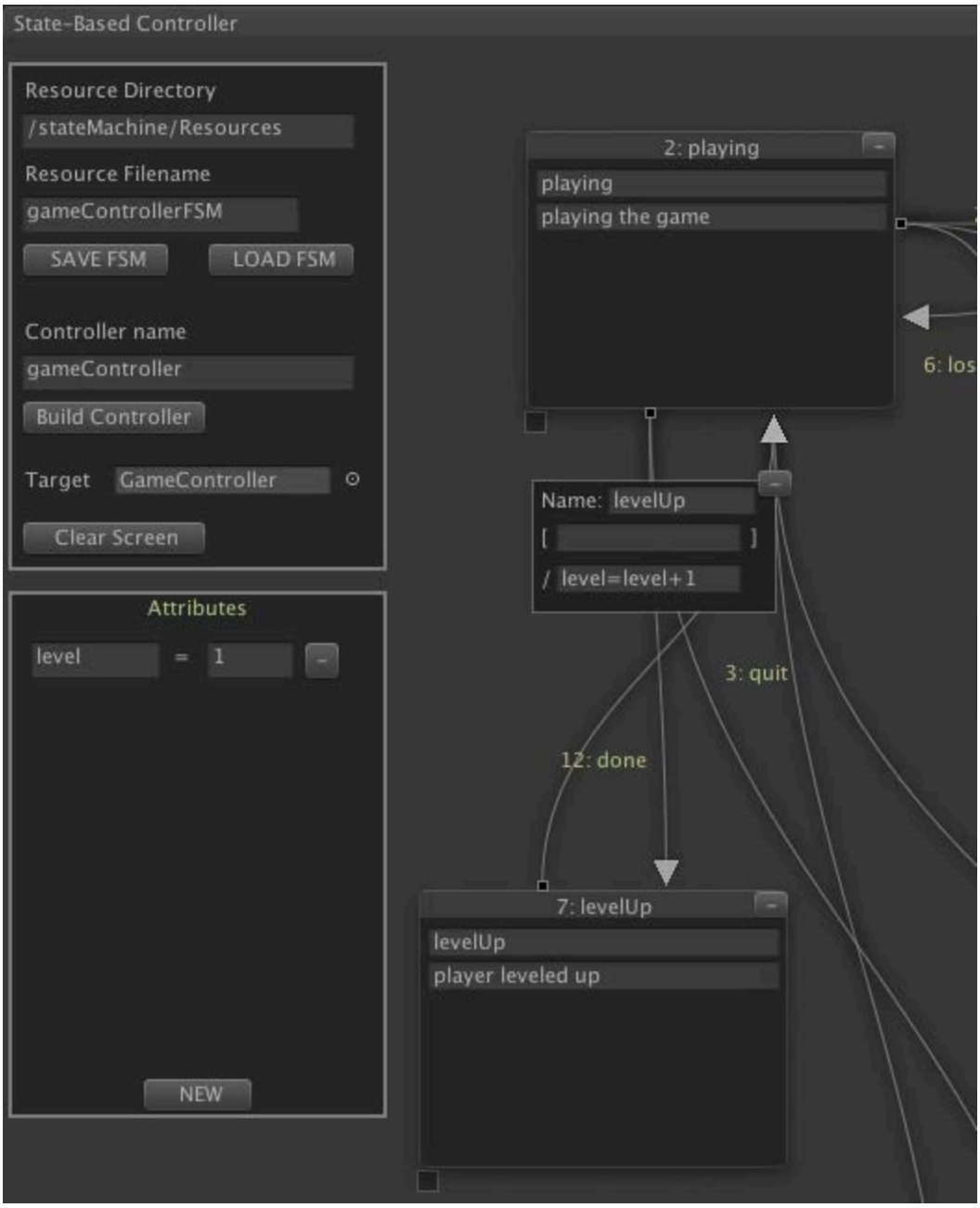
7. Design the state machine by double clicking in the FSMWindow.

8. Add states and dragging from the little nobs in the bottom left of the state to another state to add events.



8. Add new variables by clicking NEW in the Attributes panel. These can be ints, floats, strings, or booleans. You do not need to tell it what type it is, it will figure that out.

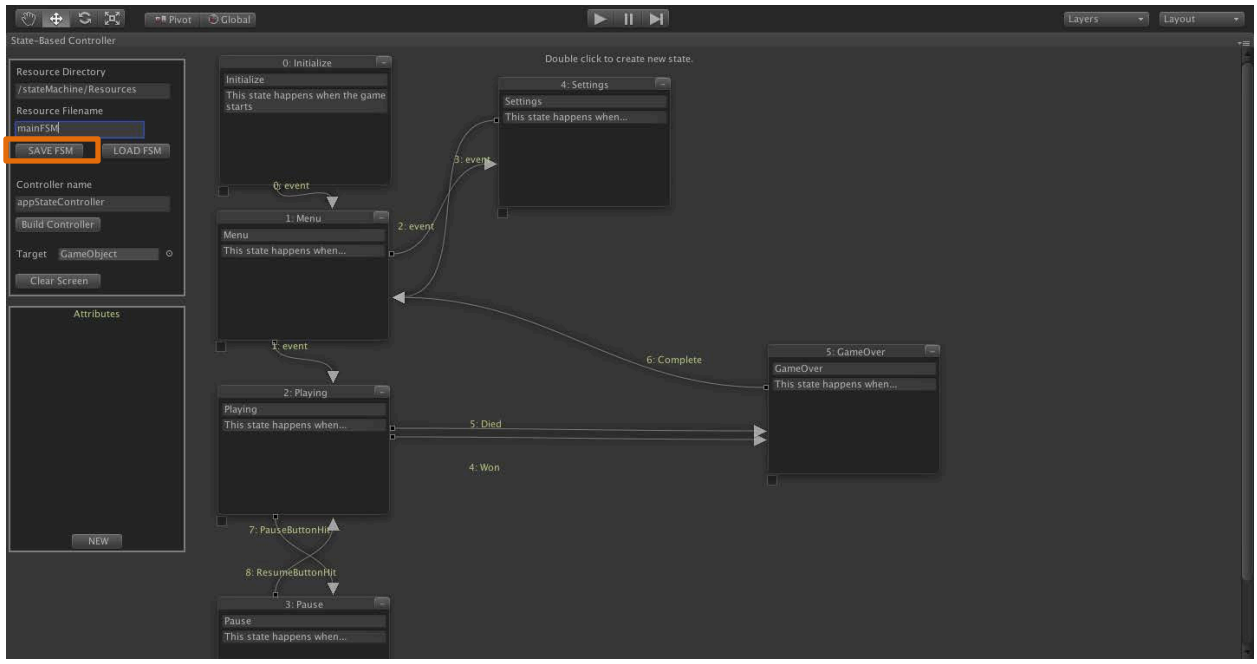


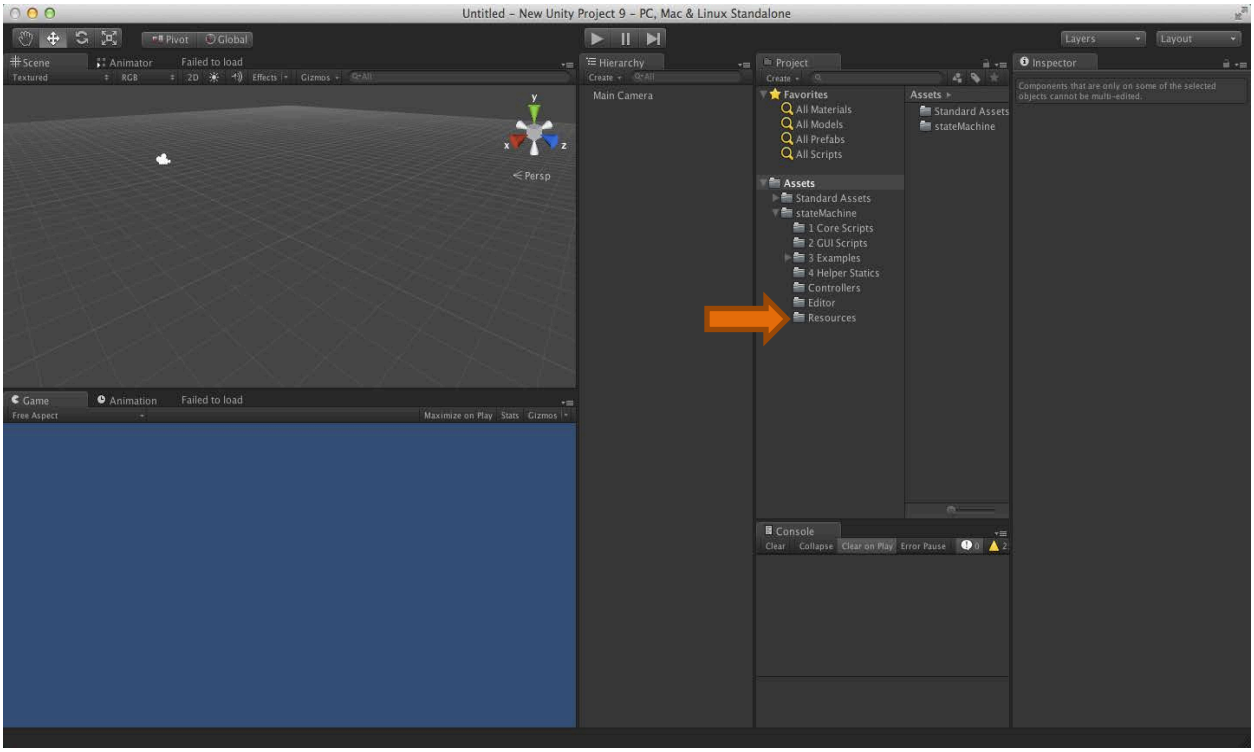


Note: You can change all values in states and events by clicking on the state field that you want to change or click on the event name to convert the event to edit mode.

Note: Don't add spaces or special symbols to event or state names.

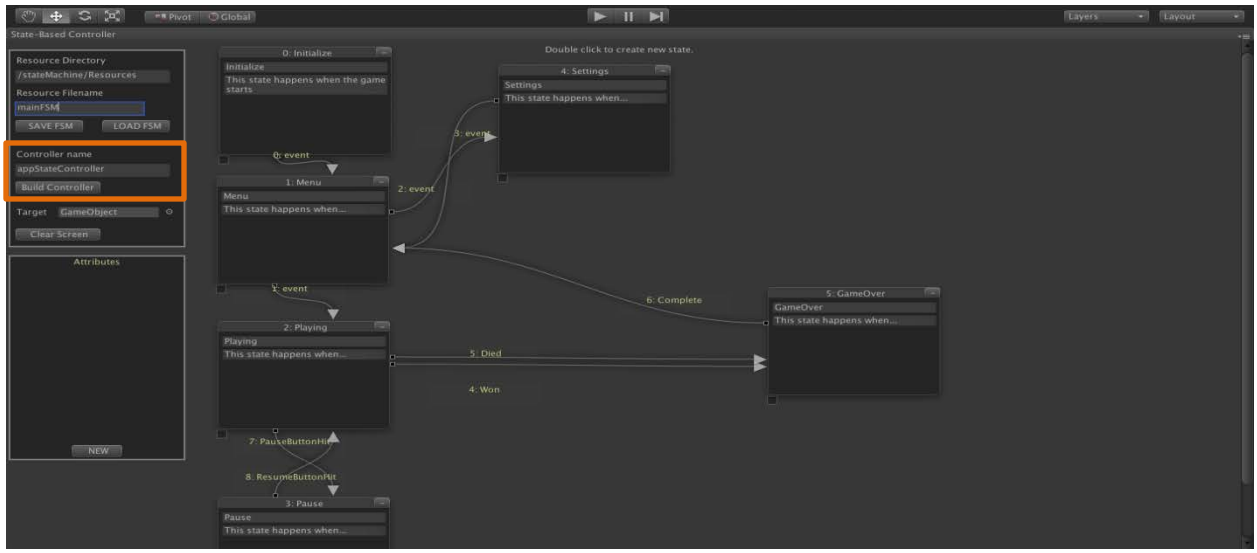
9. Click SAVE FSM. Now your resource file will be loaded in the Resource directory of /Assets/stateMachine/





Note: Clicking "Load FSM" will load the most recent saved version of the state machine for the file name you have in the resource file name in the FSM console.

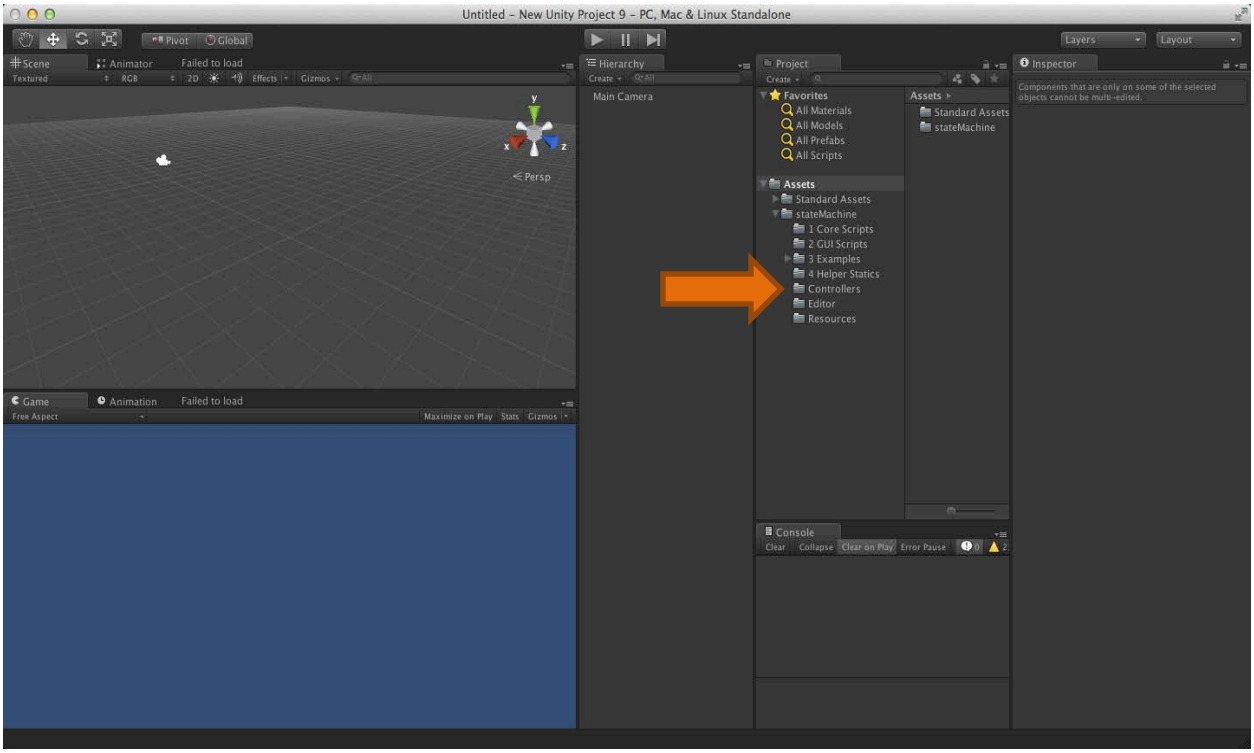
10. Rename "Controller name" to something that makes sense for this application



11. Click "Build Controller" to make a controller script that should attach to the GameObject you dragged into target. If you did not drag a GameObject you can attach it manually like you normally attach a script.

Note: The controller.cs file is created in the stateMachine/Controllers directory

12. After you make your controller it is a good idea to move it out of the controllers directory to avoid over-writing it by accident after you have implemented your handling code.



Steps to edit a controller

1. Type in the name of the FSM resource file in the "Resource Filename" input field. Click "Load FSM"

2. Make your changes to the state machine by pointing clicking, dragging and editing.

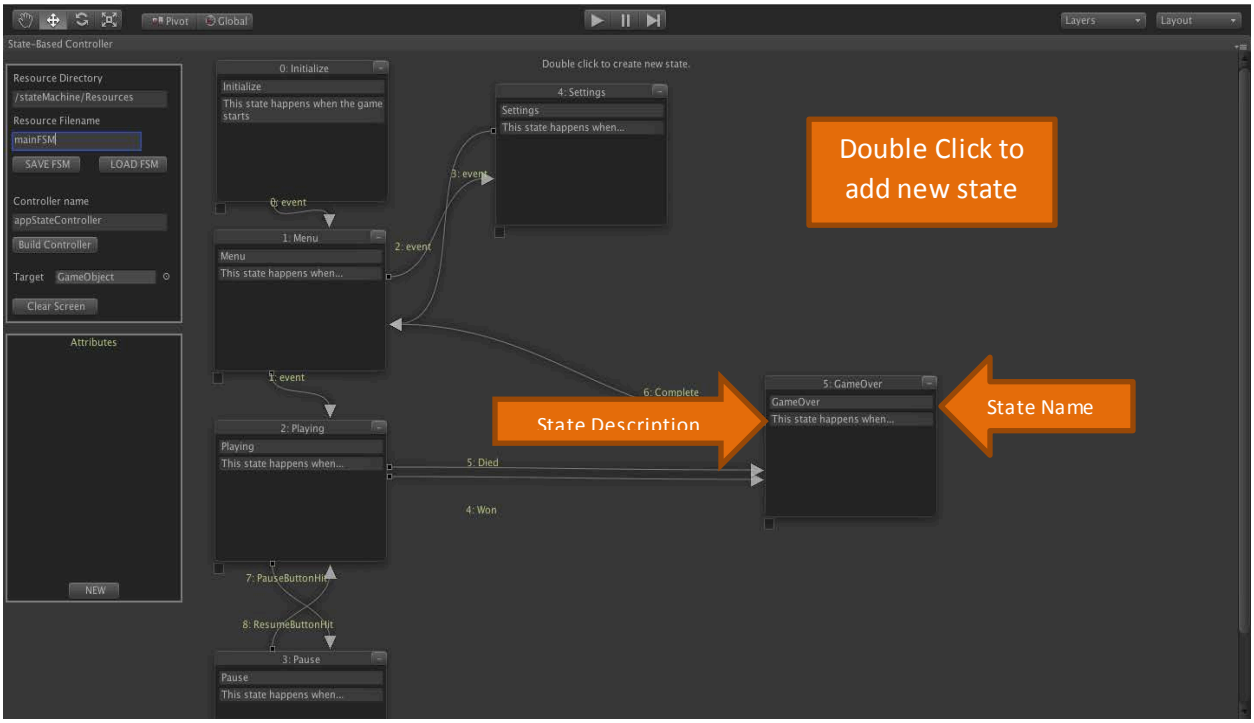
3. Click "Save FSM"

How to make a new state

1. Double click anywhere in the FSMWindow to create a new state. This state can be dragged anywhere you want it to be in the window.

2. Click in the top entry field to change the name of the state. Please do not use spaces or special symbols

3. Click in the second field to provide a state description. This is used in the controller to provide comments about that state



4. You can delete a state by clicking the " – " symbol located on the top right of the state.

> **Note**: If you accidently delete a state, rather than recreating it you can click "Load FSM" and the most recent saved version of the FSM will load. However, if you had not saved the file recently you will lose everything you created prior to the most recent save.

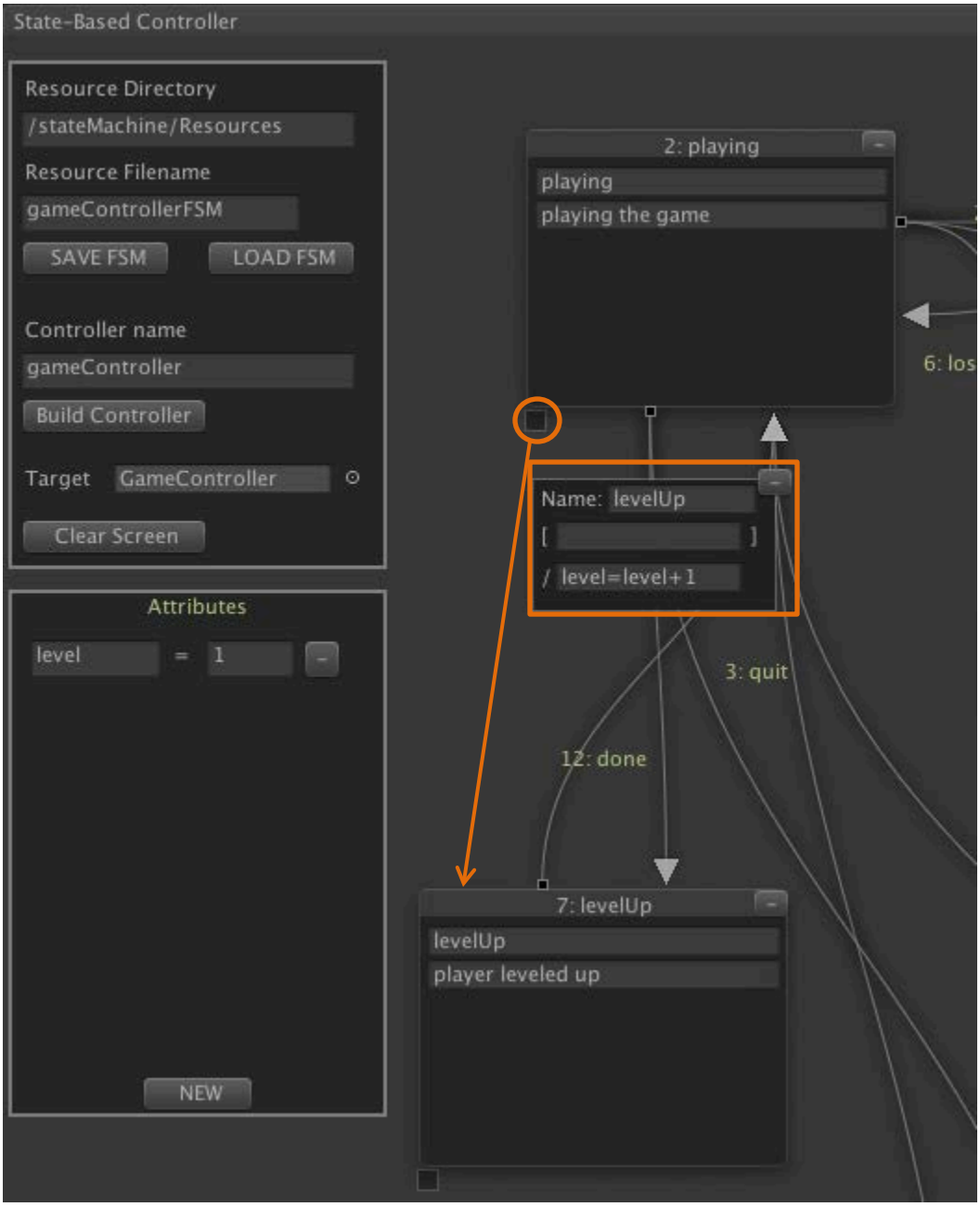
How to make a new event

1. Click and hold on the little nob on the bottom left corner of the start state panel.

2. Drag to the destination state panel. Let go when that panel lights up yellow.

3. Click the title of the event to change the displayed event to an editable panel.

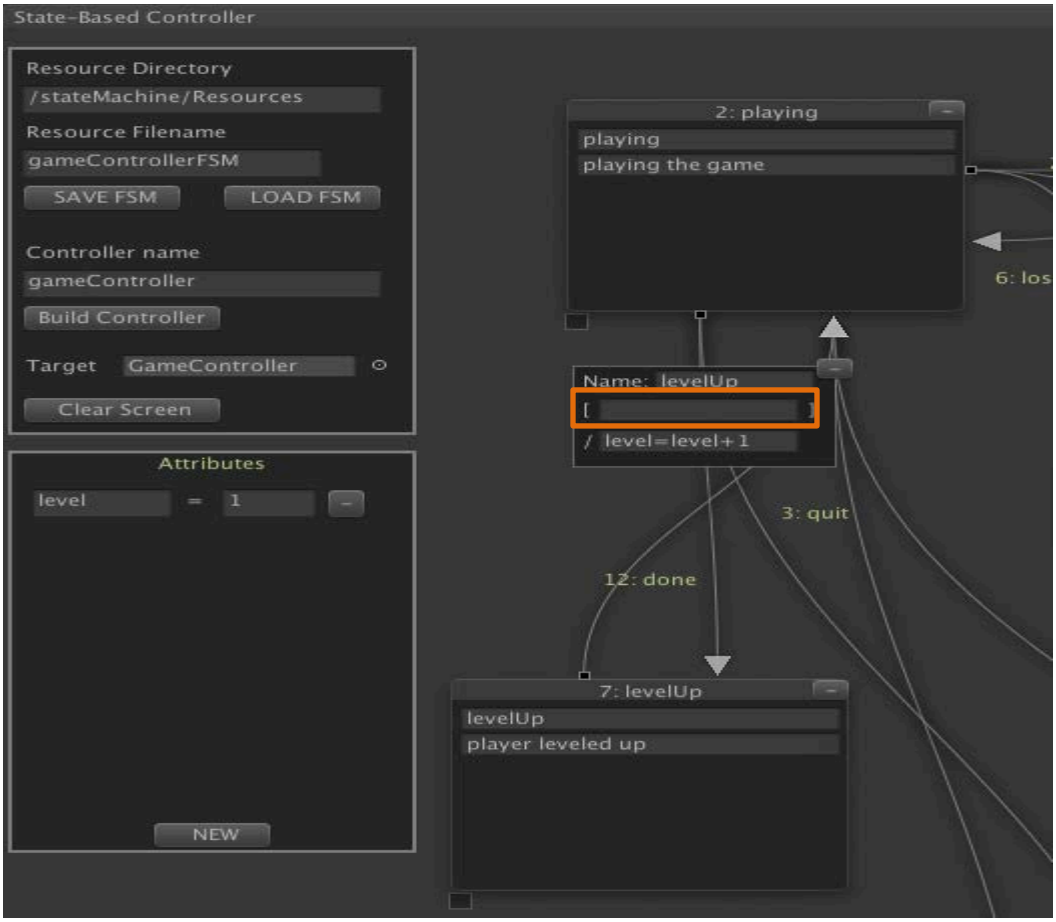
4. Click in the top panel to change the name of the event. Again no spaces or special characters



How to add conditions to an event

1. Click on the title of the event to add conditions to

2. In the second field add your conditions



examples:

state!=Texas

child<=12

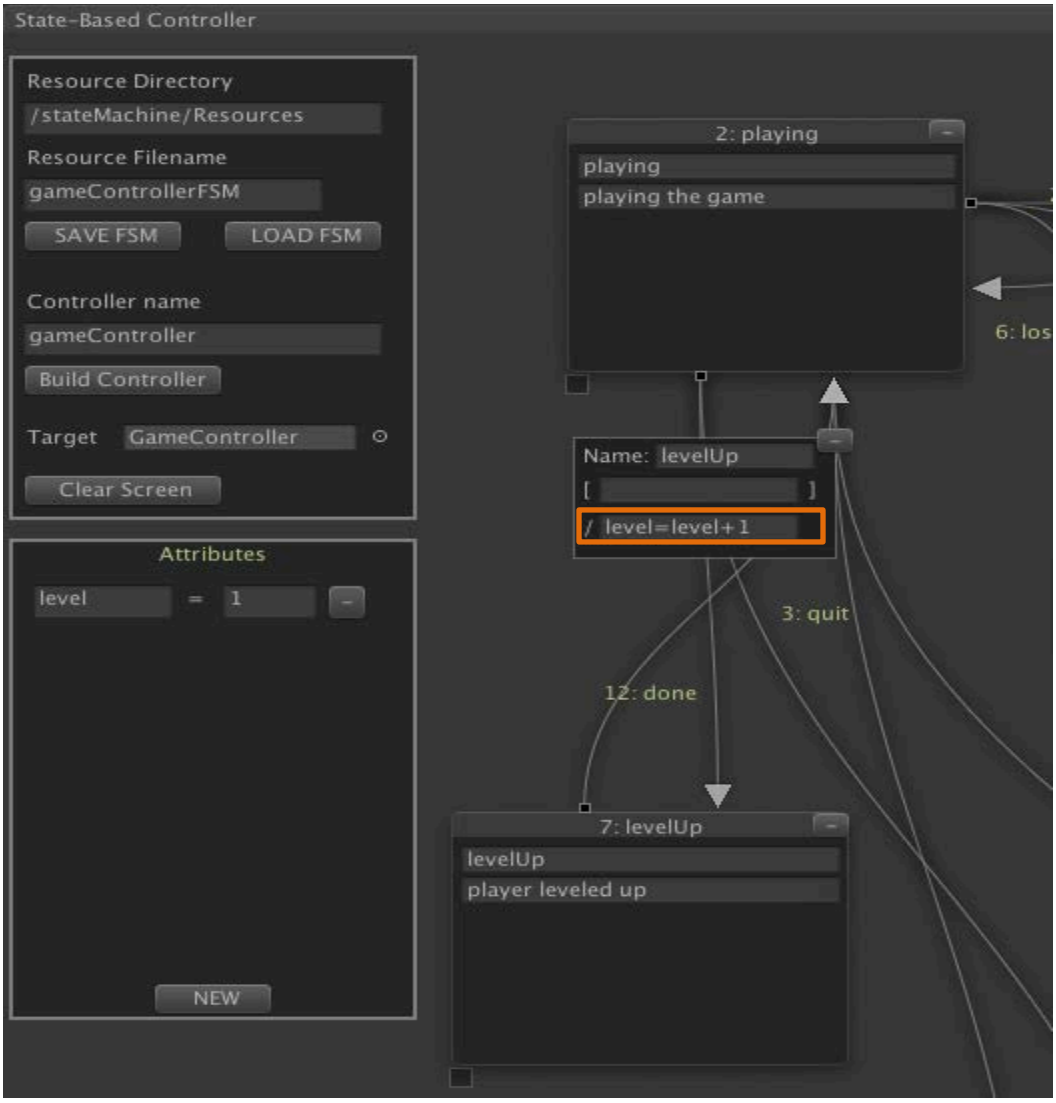
child<=12:child>5

child<=12:child>5:sex=boy

How to add actions to an event

1. Click on the title of the event to add conditions to

2. In the third field add your actions



examples:

age=20

state=Florida

old=true

radius=3.1415

Controller File (inherits from Abstract Class stateController)

To send an event to the state machine use:

```
eventToFSM = "stateName";
```

The current state:

```
myStateMachine.state   or currentState
```

To jump to a specific state bypassing the state graph:

```
myStateMachine.jumpToState(int StateID);
```

Note: the state id is shown next to the state in the state diagram window (FSMWindow)

To get a state attribute:

```
string yourAttribute = myStateMachine.getAtributeValue("attributeName");
```

To convert it to a float:

```
float.Parse(yourAttribute);
```

To convert it to an int:

```
int.Parse(yourAttribute);
```

To convert it to a bool:

```
bool.Parse(yourAttribute);
```

example: `int counter =int.Parse( myStateMachine.getAtributeValue("counter"));`

To save an attribute value:

```
myStateMachine.setAttribute("attributeName",yourAttribute);
```

Note: yourAttribute can be a string, int, bool, long or float

example: `myStateMachine.setAttribute("counter",counter);`

Using the Ticker

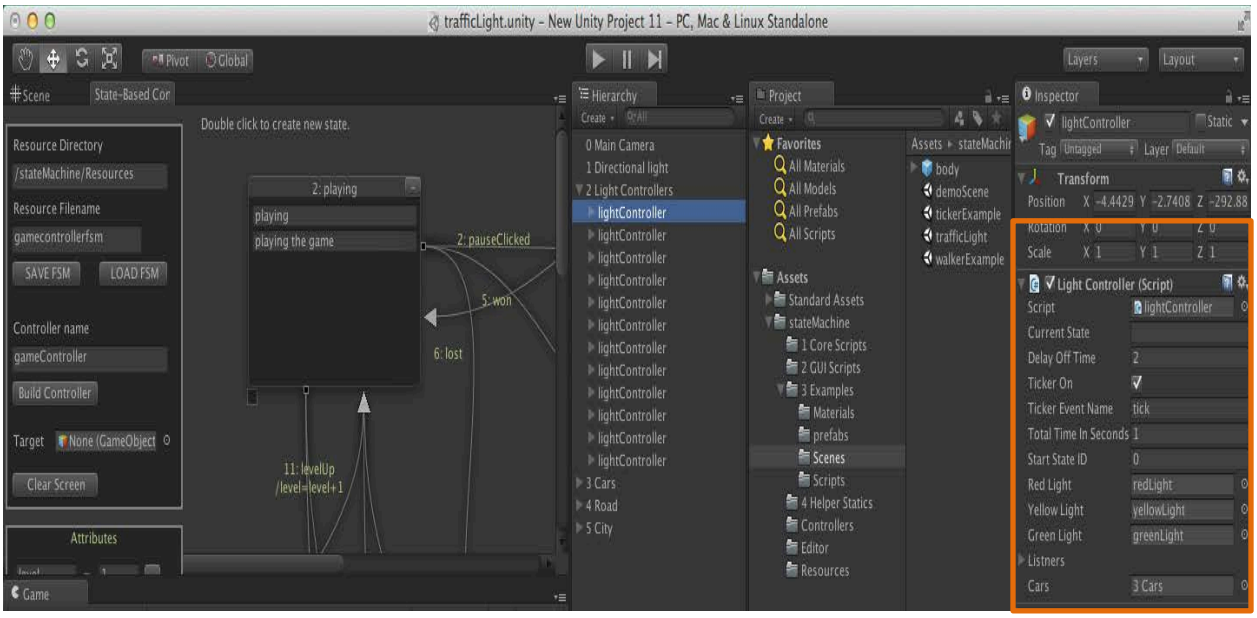
The state controller also includes a ticker which will fire a "tick" event to the state machine every x amount of second, where x is some float.

1. Click the controller game object that holds your controller script.

2. In the inspector, make sure repeating is checked and ticker is check.

3. Set Total Time in Seconds to be how many seconds you want between ticks

4. Your state machine should have events labeled "tick" with whatever conditions and actions you need for your game flow

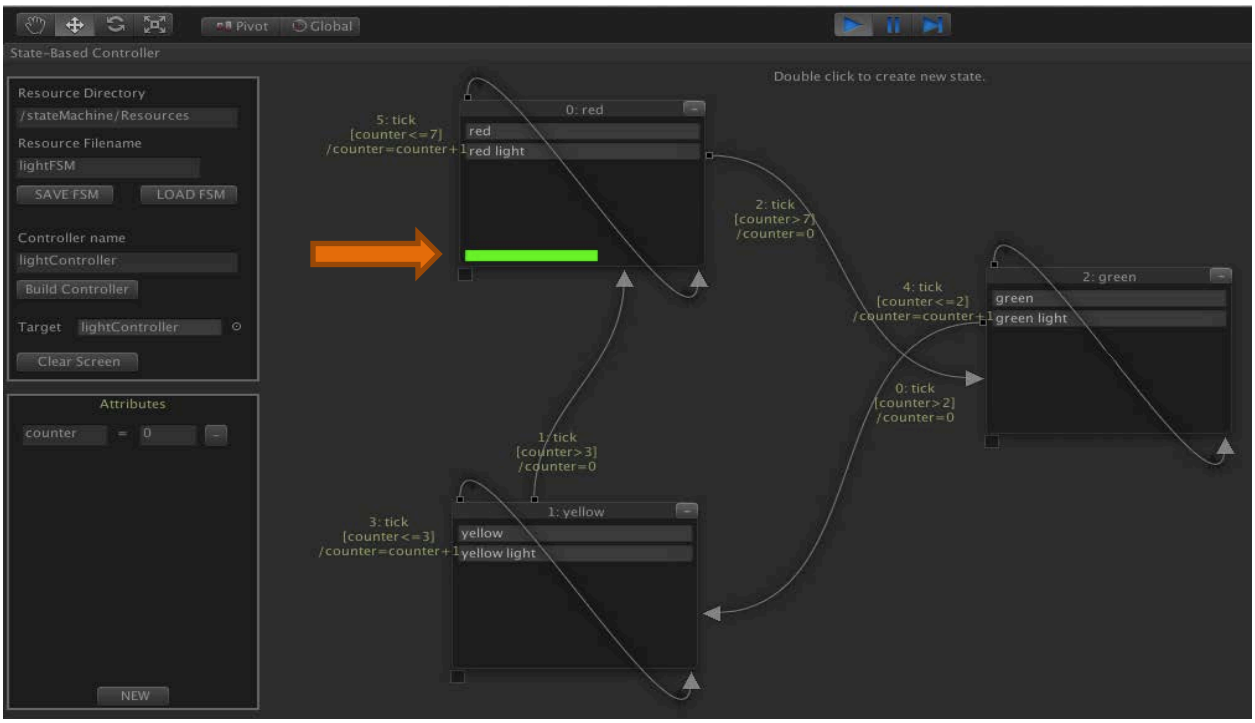


Steps to observe a controller during runtime

1. Type in the name of the FSM resource file in the "Resource Filename" input field. Click "Load FSM"

2. Drag the game object that has the controller script into the "Target" input field

3. Run the program in the editor. You should see a green bar in the active state


Pivot
Global
State-Based Controller
Resource Directory
/stateMachine/Resources
Resource Filename
lightFSM
SAVE FSM
LOAD FSM
Controller name
lightController
Build Controller
Target
lightController
Clear Screen
Attributes
counter
=
0
NEW
Double click to create new state.
0: red
red
red light
5: tick
[counter<=7]
/counter=counter+1
2: tick
[counter>7]
/counter=0
2: green
green
green light
4: tick
[counter<=2]
/counter=counter+1
0: tick
[counter>2]
/counter=0
1: tick
[counter>3]
/counter=0
1: yellow
yellow
yellow light
3: tick
[counter<=3]
/counter=counter+1