**FSM User Manual**

This document will explain how to use the Java-based FSM tool for creating a FSM or converting a FSM into a working program.

The program can either be used entirely by command line to convert a FSM file into a working program file or a GUI can be launched via the command line to design a FSM file.

[ Example image of using the command line to pass a file ]

[ Example image of using the command line to launch the GUI ]

**Graphical User Interface**

# Launch Page

The launch page is the first screen presented when the GUI is launched. From this screen, an existing FSM design file can be imported or a new design file can be started.

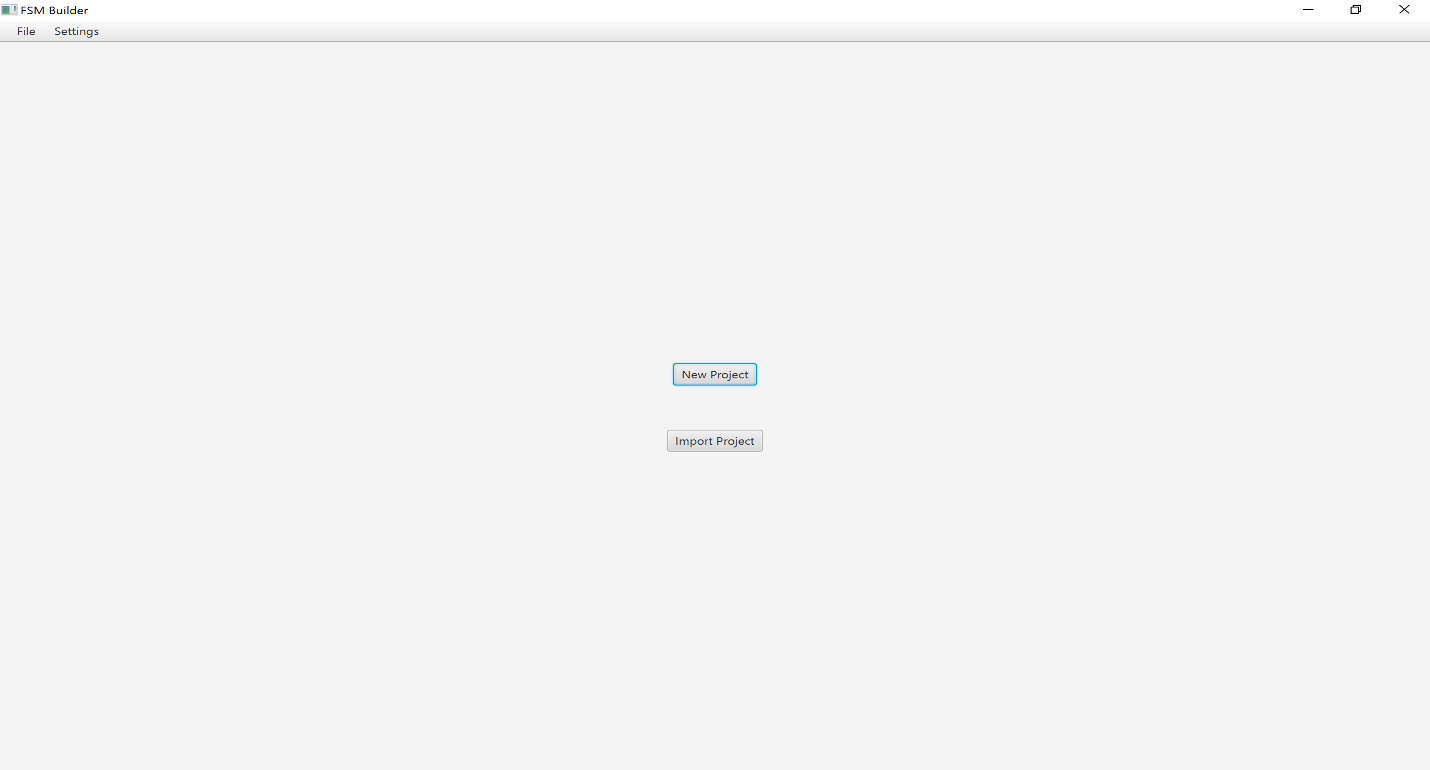
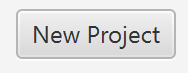
The program is launched in a maximized state and is intended to stay as such for optimal visual development of the FSM. The screen can be made smaller, but it will currently greatly hinder the design ability.

Figure 1: GUI Launch screen

Opens FSM designer view to begin creating a new FSM

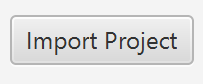
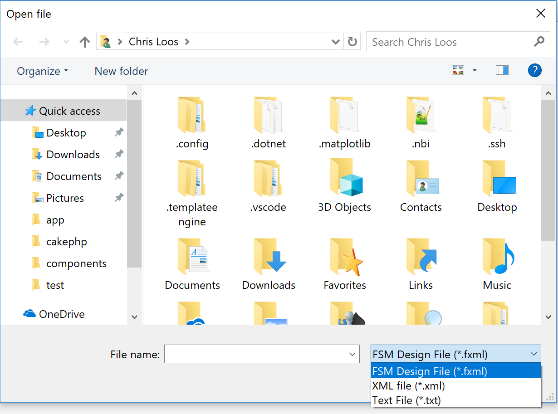


Figure 2: Import Project Dialog

Prompts a “Open File” dialog to open an existing FSM back into the designer view. These files must contain a valid XML design body and may be of the following extensions: \*.FXML, \*.XML, \*.TXT

Menu Bar

Some frequently used functions have been added to the menu bar. These can be accessed from any view within the GUI.



Figure 3: Menu Bar

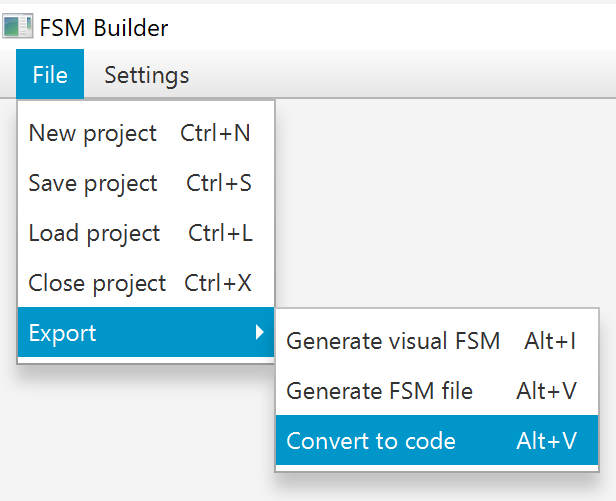


Figure 4: File menu

|  |  |
| --- | --- |
| New Project Close the current designer and start a new FSM. | Load Project Import an existing XML representation of an FSM into the designer |
| Save Project Build an XML representation of the current designer view. | Close Project Close the current designer and return to launch screen. |

# Export

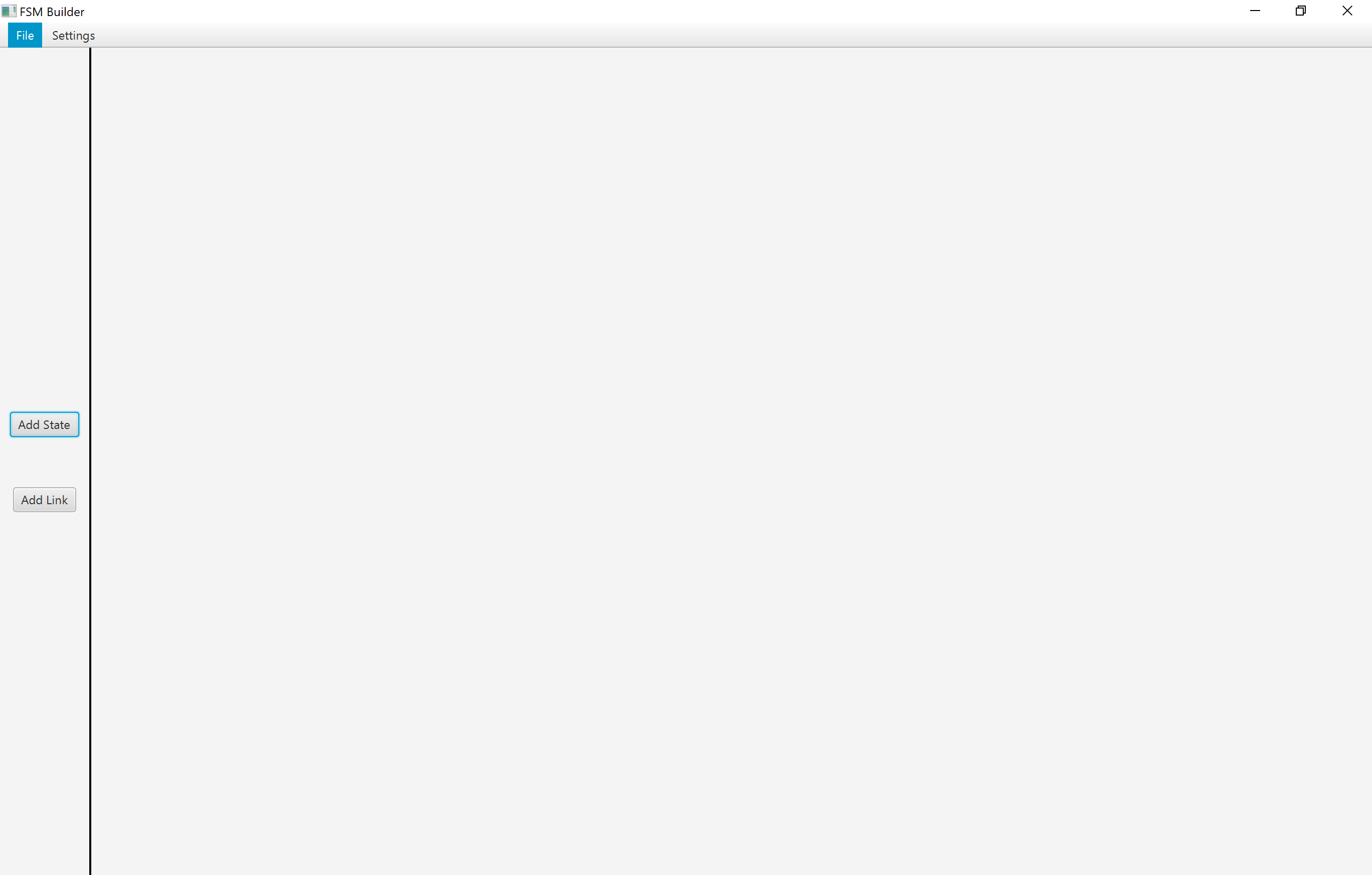
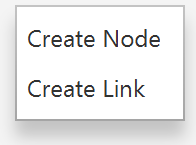
|  |  |
| --- | --- |
| Generate Visual FSM Export a .jpg image representation of the FSM | Generate FSM file Export an FSM file based on the designer view. |
| Convert to code Export a functional program based on designer view. Currently supports Python exports. | |

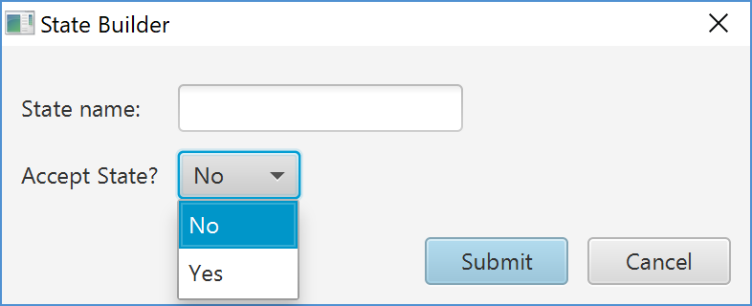
# Settings

Manipulate visual settings for designer view.

# Designer View

Existing or new FSMs can be designed from this screen.



 Figure 5: FSM Designer View

Opens a new state dialog to add a state to the designer. States may be designated as an accept state within this dialog. State names must be unique.

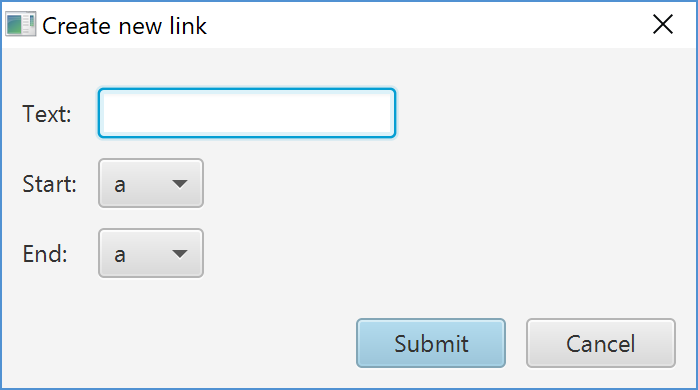


Figure 6: Create new state dialog

Figure 7: Create new link dialog



Opens a new link dialog to add a link to the designer. A single existing state node is required at a minimum to create a link. The absence of any nodes will prompt an error. Each link must be unique.

# Creating a new state

The create state dialog allows the creation of new state nodes. Alternative to the Add State button, right-clicking on the designer view allows access to this prompt as well. All new state nodes are created in the center of the page as a default, but can then be moved into the desired position. All state nodes must be unique.

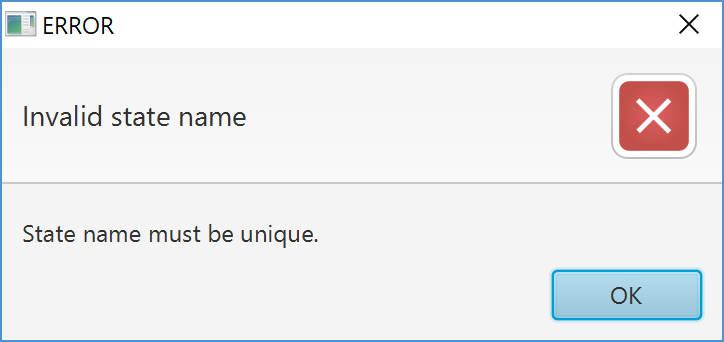
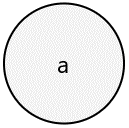
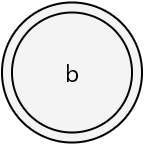


Figure 10: Accept state node

Figure 9: State node

Figure 8: Error - State node exists

# Creating a new link

The create link dialog allows the creation of new links between state nodes. Alternative to the Add Link button, right-clicking on the designer view allows access to this prompt as well. All links must be unique. Creating a new link requires at least one existing state node.

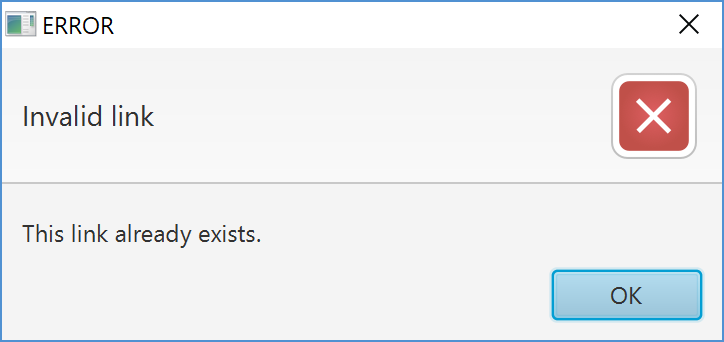
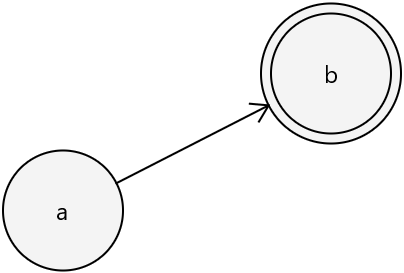


Figure 11: Error - Link exists

Figure 13: Link between node a and node b

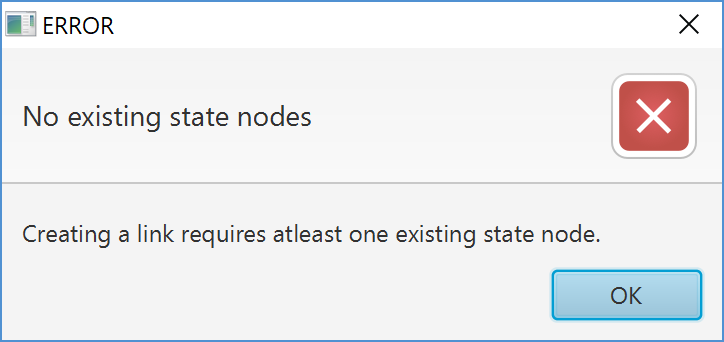
 [ INSERT IMAGE OF SELF LINK ]

Figure 12: Error - No state nodes

# Manipulate an existing node

Right-clicking an existing node offers the ability to make changes to the target node.

Figure 15: State node context menu

# Exporting FSM Data

From the GUI, the FSM can be exported in three formats.

# Generate visual FSM

This creates an image file based on the FSM currently being developed in the developer view by utilizing [?], an external tool from this project via the command line interface.

# Generate FSM file

Figure 16: Sample FSM file output

This creates an FSM file based on the FSM currently being developed in the developer view.

# Convert to code

Generate production code based on a provided FSM file. Currently only produces Python code.

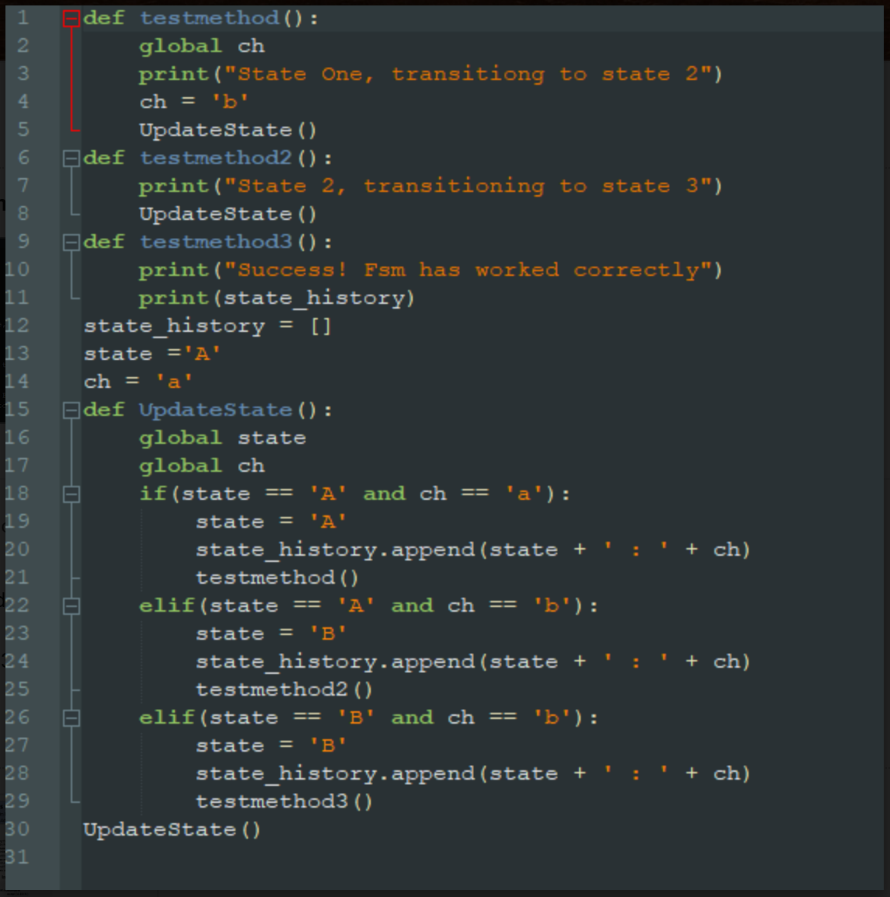


Figure 17: Sample production Python code generated from FSM file

**Command Line Interface**