HexagonNico/Godot-FiniteStateMachine: A plugin for Godot 4 that adds an implementation of the finite state machine pattern

Finite State Machine plugin

A GDScript implementation of the finite state machine pattern.

Adds node types for finite state machines and states.

Installing the plugin in your project

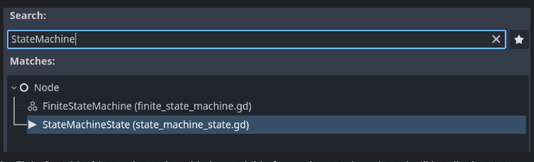
The procedure is the same as other Godot plugins. See the Godot docs for a full explanation.

1. Click the **AssetLib** tab at the top of the editor and look for Finite State Machine.
2. Download the plugin and install the contents of the `addons` and the `script_templates` folders into your project's directory. You don't need the contents of the `example` folder.
3. Go to `Project -> Project Settings... -> Plugins` and enable the plugin by checking "Enable".

It is also possible to install the plugin manually by downloading the zip archive from the Releases section.

Plugin contents

The plugin contains two script classes: `FiniteStateMachine` and `StateMachineState`. Both can be added to your scenes as nodes through the "Create New Node" menu.

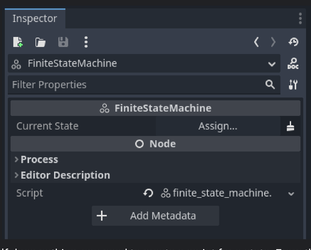


The **FiniteStateMachine** node can be added as a child of your character's node and will handle the state machine's logic.

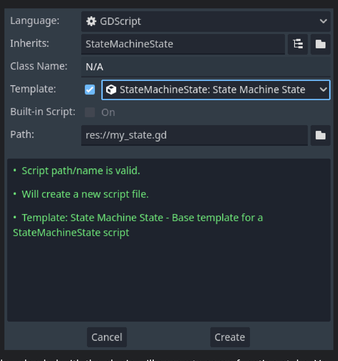
The **StateMachineState** class is an abstract class. You can extend this script to create your states.

Creating a state machine

Open the "Create New Node" menu and search for "FiniteStateMachine". Add this node in your character's scene.



A state machine by itself does nothing, you need to create a script for a state. From the "Create script" menu, create a script that inherits from `StateMachineState`.

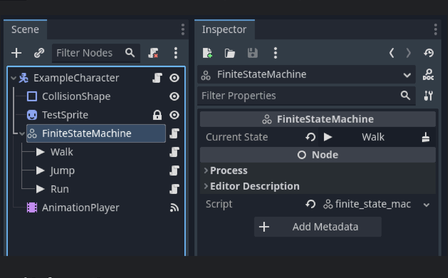


The script template downloaded with the plugin will generate some function stubs. You can now implement these functions to create a state.

```
class_name MyState
extends StateMachineState


# Called when the state machine enters this state.
func on_enter() -> void:
	pass


# Called every frame when this state is active.
func on_process(delta: float) -> void:
	pass


# Called every physics frame when this state is active.
func on_physics_process(delta: float) -> void:
	pass


# Called when there is an input event while this state is active.
func on_input(event: InputEvent) -> void:
	pass


# Called when the state machine exits this state.
func on_exit() -> void:
	pass
```

Once a state is created, you can add it as a child of the FiniteStateMachine node. Assign a state to the state machine from the inspector to set that as the initial state.

You can now change between states from your scripts by setting `current_state` or using the `change_state` function. See the comments in the scripts for a full explanation.



Additional information

Important note: Do not delete `.godot/global_script_class_cache.cfg` when this plugin is installed until Issue 81615 is resolved.