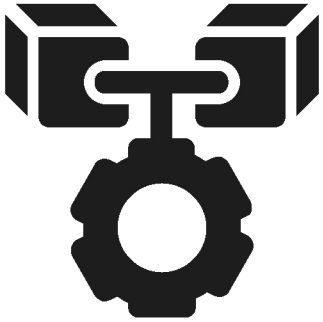


State Machine

1.0.2

Documentation

Table of Contents

1. Introduction

1.1 What is a State Machine?

A state machine is a state switch that uses different kind of conditions to determine in a list of states, what state is currently running.

A state is a part of the state machine that will contain logic on the current state. like a player movement state that will contain logic on how the player moves.

1.2 Why use this State Machine?

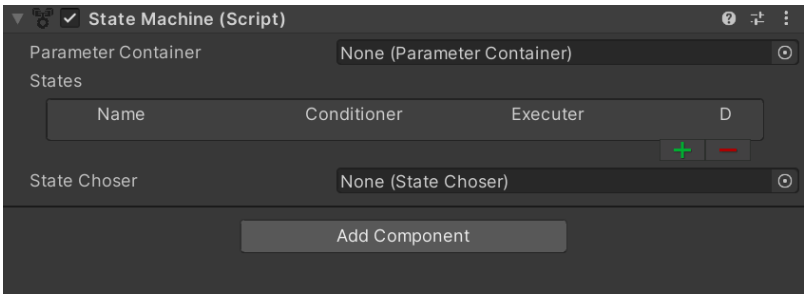
A state machine at its simplest form is just different files that contain the different kind of state, The states will contain logic on how and when the states will switch. This in its base form will get very ugly and hard to read in the long run and that’s why you need a more advanced state machine.

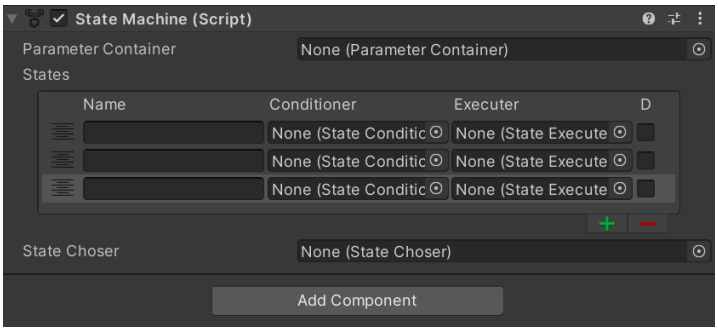
This state machine uses the State Machine, State Choser, State Parameters, State Executer and State Conditioner components to determine what state will run and what a state will do when it is running (More information for this different kind of interfaces down below in chapter 3)

2. Basic Functionality

2.1 Using

When adding a State Machine to an GameObject in the inspector, it will look like something like the picture to the right. Here you can control what logic or parameters to use in your state machine





Under the *"States"* tab you can see a list of states that this state machine have. By default this will be empty, but if you click the green + button on the bottom of the list you will add a state, it will look something like the picture to the left.

You can click the red – button to remove the last state in the list, but if you select a state by clicking on it and press the – button, you will remove the selected state.

When adding a state you will have 4 parameters that define a state.

Name: The name of the state (Used for debugging)
Conditioner: The state conditioner for this state (see, 3.4)
Executer: The state executer for this state (see, 3.5)
D: If this state is the default state (One state at the time can be default)

The Parameter Container and State choser element is where you can have your custom state machine parameters that all states will have access you and define the chose logic that define what state the state machine will chose at a given frame (see, 3.2-3.3)

By default this components will be fill out with the default state parameter and default state choser components when starting the application if none is specified.

3. Interfaces

3.1 State Machine

This component is the main component for this feature. Its job is to control the state choser and the state parameter components. This component will mostly be interacted with unity inspector but can also be controlled by its script interface.

3.2 State Parameter (abstract interface)

The state parameter component is a container component that allows the user to define the states machine parameters that can be access in every state.

By default if not defined the state machine will add a default state parameter component at runtime.

3.3 State Choser (abstract interface)

This script job is to control the choosing, what state will run in a given frame. By defining your own state choser you can easily control what states to run at a given time.

By default if not defined the state machine will add a default state choser component at runtime.

3.4 State Conditioner (abstract interface)

This script defines a condition to a given state that defines when this state will run. A state must have a state conditioner to function if not set to default.

3.4 State Executer (abstract interface)

The state executers job is to allow the user to define the execution logic to a state. A state must have a state executer to function.

For script documentation in more detail see, <not yet implemented>