FAQ

How do I create a machine?

After installing the asset, right-click anywhere in your Assets-folder and click **Create > Graphical State Machine.**

I instantiated GameObjects by code. How do I execute callbacks on them?

You can apply callbacks to any event by code as well. Right after instantiating, call `GraphicalState.RegisterRuntimeCallback(RuntimeCallback)` to register a callback. Here is a quick example:

```
var newObject = Instantiate(prefab);

machine.GetStateByName("Game Running")
      .RegisterRuntimeCallback(
            GraphicalStateMachine.RuntimeCallbackType.OnStateStay,
            newObject.GetComponent<MyComponent>().MyCoolUpdateFunction);
```

This creates a new GameObject called `newObject`. It has a Component `MyComponent` where you want `MyCoolUpdateFunction()` to be called each Update call while the state "Game Running" is active.

Tip: You can always register any method by using anonymous lambda functions!

Why are my runtime callbacks gone when restarting the machine?

When stopping the machine using `GraphicalStateMachine.Stop()`, all callbacks will be deleted. If you want to keep them, just call `GraphicalStateMachine.Stop(false)` instead or uncheck the box “Clear Callbacks on Stop” on the `StateMachineProcessor` Component.

Why are my runtime callbacks gone when restarting the machine even with “Clear Callbacks on Stop” unchecked?

When stopping your machine by code, the **Clear Callbacks on Stop** Checkbox will be ignored. To keep your runtime callbacks, just call `machine.Stop(false)`. The bool determines whether the callbacks should be deleted or not.

What is the difference between OnStateSetActive() and OnStateEntered()?

Both callbacks get invoked when a state is set active. First, OnStateEntered() and then OnStateSetActive(). The difference is, that OnStateSetActive() will also be called when the state is set active by name or when the game starts.

The methods I attached to OnStateStay seem to be laggy. Why?

When editing a state, you can choose which Update-method you want to use (FixedUpdate, Update or LateUpdate). In some cases, one method makes more sense than another. Choosing the wrong one may cause “laggy” behaviours.

The methods I attached to OnStateStay are getting called, even when the state is not active.

Make sure not to name your methods Update(), FixedUpdate() or LateUpdate(). Those functions are provided by Unty itself and will always be executed!

Some methods are not getting called. Why?

Quick solution: Activate “Show Invocation Error” in your machine’s settings. Not invoking a method may have multiple reasons:

Reason No.1:

The machine searches your selected objects of each callback by its name in the current scene (once at startup). Make sure you have opened the right scene. To proof that this is the case, open your machine and select the state where the method should be executed. If you see the message “Unknown Object”, the selected object could not be found in the scene.

Reason No.2:

The method does not exist. This may be the case when the object is deleted or renamed, the script is deleted or renamed, the script is not attached to the object or when the method itself is renamed or deleted.

Why can I only set OnStateSetActive() on terminating states?

When a terminating state is set active, the machine will immediately be stopped after OnStateSetActive() is executed. Therefore OnStateStay() and OnStateLeft() will not make sense. Also in that case, OnStateSetActive() and OnStateEntered() provide the same functionality.

Can I have two or more StateMachineProcessor with the same machine file?

Yes. Since version 1.1.

In that case, both processors share the same machine file. That means, that changing the state for one processor will automatically changes the state in all other (current state is saved within the file itself, which is shared between all processors)