Behavior Demo Documentation

Project Overview

This demo showcases the Unity Behavior package, enabling for the creation of artificial intelligence (AI), interactions, state machines, and behavior trees by assembling nodes.

- Read more about Behavior
- Get answers and support

Features overview and asset mapping

- **Behavior graphs**
  - **Interaction**: Simple behavior reaction to input
    - `B_Teleporter`
    - `B_Crushable_Mushroom`
    - `B_Promenade_Coin`
  - **State machine**: Simple or well-defined state scenarios
    - `B_StateMachine_Game`
    - `B_StateMachine_GameMode`
    - `B_StateMachine_Character`
  - **Behavior tree**: Complex behaviors, ideal for nuanced decision-making and behavior sequencing
    - `B_Brain_Patrol`
  - **Generic behavior/Function**: Implement logic with provided data
    - `B_Load_Level`, `B_Unload_Level`
    - `B_Attack_Simple`, `B_Attack_Chain`, `B_Attack_Combo`
    - `B_Promenade_Start`, `B_Promenade_GameOver`
- **Blackboard assets**
  - **Global collection**: Share global variable accros graphs
    - `BB_Player_Camera_Shake`
    - `BB_AudioSourceReferences`
  - **Interface**: Share common variable accros graphs
    - `BB_GameMode`

        - `BB_Character_Attack`
- **Run subgraph**
    - **Static**:
        - `B_StateMachine_Game` with `B_Load_Level`, `B_Unload_Level`
    - **Dynamic**:
        - `B_StateMachine_GameMode` with `B_Promenade_Start`, `B_Promenade_GameOver`
        - `B_StateMachine_Character` with `B_Attack_Simple`, `B_Attack_Chain`, `B_Attack_Combo`
- **C# binding**
    - `B_StateMachine_Character` with `AgentController.cs`
    - `B_Brain_Patrol` with `AIAgent.cs`

1. Graphs overview

Game state machine (`B_StateMachine_Game`)

> `GameState.cs`: `Init`, `Level_00`, `Level_01`, `Level_02`, `Exit`

Manages overall game flow and level transitions.

Features

- Emulates a state machine `OnStateEnter/Exit`.
- Uses RunSubgraph (Static) to execute `B_Load_Level` and `B_Unload_Level` with different parameters.

Game mode state machine (`B_StateMachine_GameMode`)

> `GameModeState.cs`: `Init`, `Loop`, `GameOver`, `Pause`

Manages game mode rules across levels:

- Handles UI elements (HUD, Menu, Timer).
- Manages level timer and game over conditions.
- Controls agent controllers based on game state.
- Controls gameplay pause/resume.

Features

- Supports additional behaviors through subgraphs for Start, Update, and GameOver states using `BB_GameMode` (as interface).
- provides an `Input Performed` example that listen to project-wide input actions.

Character state machine (`B_StateMachine_Character`)

> `CharacterState.cs`: `Idle`, `Hit`, `Attack`, `Dead`

Integrated with C# scripts, responds to character state changes.

Features

- **Behavior Reaction**: Reacts to `CharacterStateEventChannel` state changes with behaviors like attack actions and hit reactions.
- **Graphs Binding**: Reacts to the global `EventChannel_GamePause`.
- **C# Binding**: Works with `AgentController.cs`.

Character subgraphs overview

- `B_Attack_Simple`: Basic attack logic.
- `B_Attack_Chain`: Sequential attack logic.
- `B_Attack_Combo`: Combo attack sequences logic.
- `B_Promenade_Player_Hit`: Promenades specific logic to drop coin when hit.
- `B_Character_Death_Player`: Sends message over `EventChannel_GameModeState`.

Patrol brain (`B_Brain_Patrol`)

Controls the AI behavior of patrolling guards.

Features

- Controls a character using Navigation nodes.
- Communicates with the relevant `B_StateMachine_Character` through a script-binded `CharacterState EventChannel`.

- Manages two state machines using:
  - `NpcState.cs` (`Idle`, `Investigate`, `Attack`)
  - `NpcCombatState.cs` (`Chase`, `Attack`, `Retreat`)

Trigger Area

Examples of simple graph reacting to `OnTriggerEnter`:

- `B_Teleporter`: Sends message over `EventChannel_GameState` to transition to another level.
- `B_Crushable_Mushroom`: Interactions with mushrooms.
- `B_Promenade_Coin`: Manages coin spawning and collection.

2. Blackboard Assets

- `BB_GameMode`: Interface for game mode subgraphs.
- `BB_Character_Attack`: Interface for attack subgraphs.
- `BB_Player_Camera_Shake`: Camera shake data.
- `BB_AudioSourceReferences`: `AudioSource` references (used by the `PlayAudio` node).
- `BB_Promenade`:Promenade-specific data.

3. Event Channels

- `EventChannel_GameState`: Sends messages to the game state machine.
- `EventChannel_GamePause`: Managed by the GameMode for pause functionality.
- `EventChannel_GameModeState`: Sends messages to the game mode state machine.

4. Scripts binding

`AgentController` and `IStateMachineModifier`

Example binding a BehaviorGraph to C# systems using EventChannel (`AgentController.cs`):

```
/// <summary>
/// Binds the local Character's EventChannel BlackboardVariable to
```

```
propagate state changes to C# systems.
/// </summary>
private void BindCharacterStateChannel()
{
        Debug.Assert(m_StateMachine != null, "AgentController need
a valid StateMachine", this);

        if
(!m_StateMachine.BlackboardReference.GetVariableValue(k_StateChanne
lVariableName, out m_StateChannel))
        {
                Debug.LogError($"{m_StateMachine} expects a
BlackboardVariable of type
'{typeof(CharacterStateEventChannel).Name}' named " +
                        $"{k_StateChannelVariableName}.");
                return;
        }

        m_StateChannel.name = this.transform.parent.name + " State
Channel";
        var components =
transform.parent.GetComponentsInChildren<ICharacterStateChannelModi
fier>();
        if (components == null || components.Length == 0)
        {
                return;
        }

        foreach (var component in components)
        {
                component.StateChannel = m_StateChannel;
        }
}
```

Troubleshooting

Several warnings when importing the project

When you first import the demo, you might encounter seven warnings:

- `File 'Anims M Level01' has animation import warnings. See Import Messages in Animation Import Settings for more details.` (1 occurance)

- `A polygon of Mesh 'rock_4_0x' in Assets/Art/Environments/Generic Landscaping/Rocks/rock_4_0x.fbx is self-intersecting and has been discarded.` (6 occurances)

These import warning can be safely ignored.

Errors when switching projects: `Path could not be found for script compilation file`

If you get this error when switching projects, create an empty project and use the `Import` option to bring in the assets.

`InputEventModifier: Failed to find action System/Pause as part of the InputSystem Project-wide Action asset (Project Settings > Input System Package > Project-wide`

To resolve this issue, ensure that the Project-wide Action asset is set to `Asset/Data/Input/SystemInput_Actions` in your Project Settings.