



Quest Tool complete guide

Description

Quest Tool is a visual graph editor for quest scripts. It provides a powerful quest management system that allows for the control and tracking of the progress of each quest.

Narrative designers can quickly and easily create game quests with the help of quest nodes, which use events and function calls to perform actions in response to gameplay events. Nodes run when they receive a signal and are completed after their task is performed, after which they send the signal further.

There are four types of nodes:

1. Action nodes are executed when an input signal is received. These nodes are the basis of the script.
2. Event nodes enable blueprints to perform a series of actions in response to certain events that occur within the game.
3. Condition nodes – pass the signal when a configured condition is met. It cannot send an execution signal.
4. Container nodes are executed when an input signal is received. These nodes allow running an attached blueprint, thereby encapsulating and structuring the logic of the quests.

There are also editor tools:

- Comment block – a section used to create comments

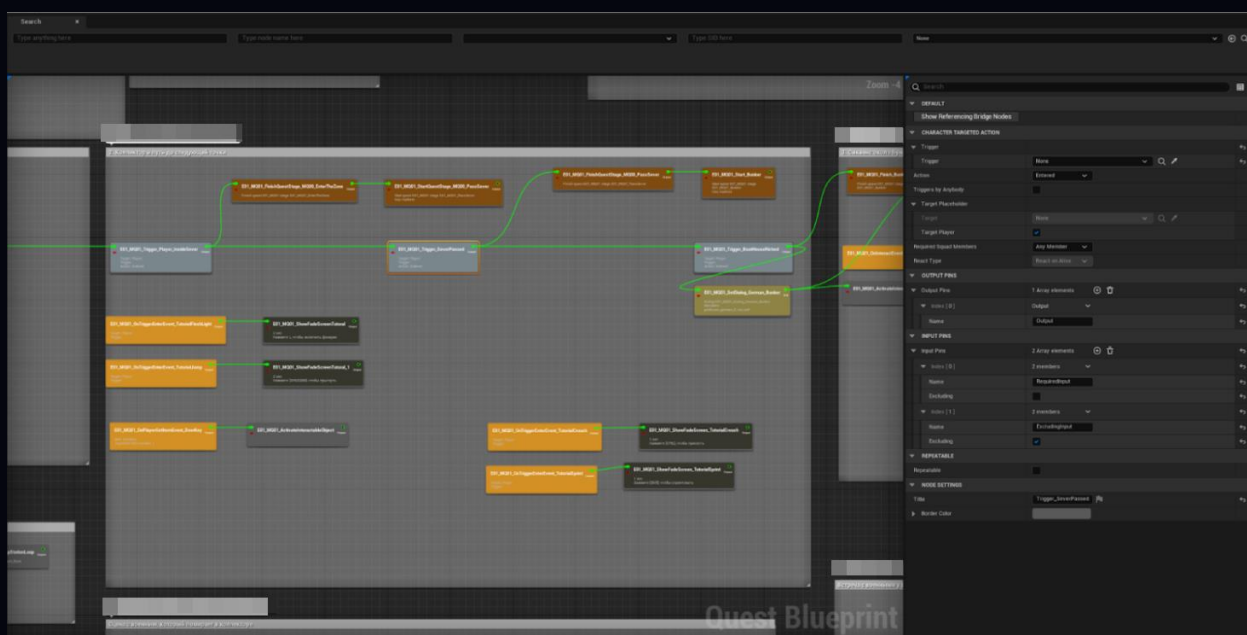
In addition, you can edit node settings:

- Configure node comments

- Edit node names
- Change the color and thickness of node outlines, while the node fill color remains unchanged and corresponds to the node type

References

Quest blueprint example



Any quest blueprint can be saved in two formats:

1. *.uasset – binary files that contain information about the blueprint, including its components and settings.
2. *.cfg – these files contain information about the blueprint's settings in a text format. You can edit these files manually in any text editor, making them useful for configuring blueprints outside of the Unreal Engine environment.

Use case

The Quest Tool functionality allows you to do the following:

- Create new quest blueprints
- Activate other blueprints within existing blueprints using containers
- Add and delete nodes, cut, copy, and paste, including between different blueprints
- Connect nodes
- Configure nodes – it affects how they work in the game
- Run blueprints and individual nodes in the game, including with the help of debug functionality
- Connect different blueprints or different segments of a single blueprint using bridges
- Change the color and thickness of node outlines

Nodes let you do the following:

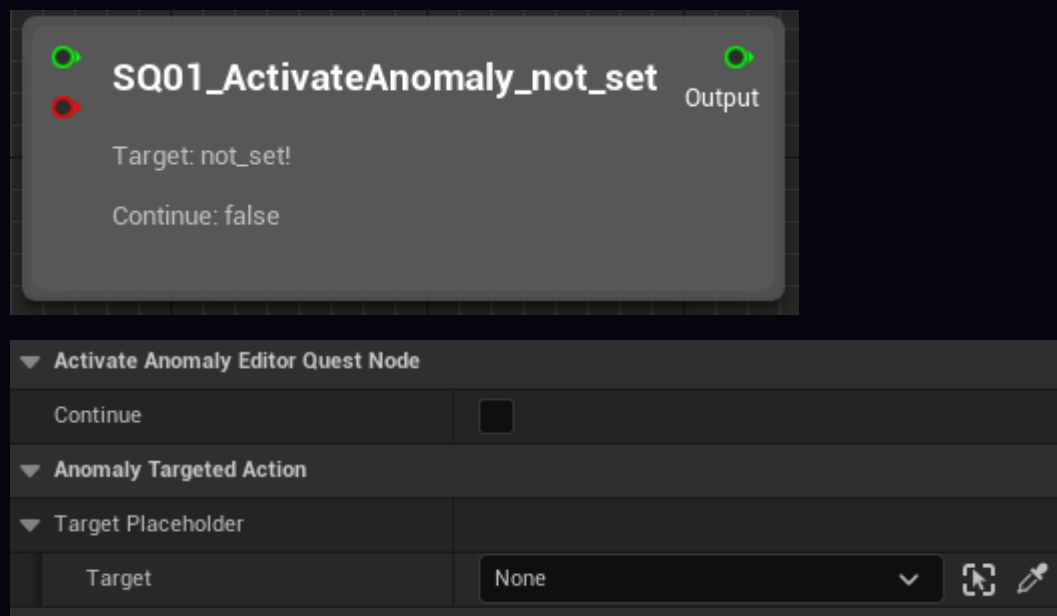
- Activate anomalies
- Send and receive signals to interactive objects
- Add notes and audio logs for the player
- Spawn an NPC, squad, anomaly, mutant, item, or container with items
- Add or remove an NPC or mutant from a squad
- Add upgrades to technicians
- Change an NPC faction
- Change the relationships between characters or factions
- Execute console commands specified when configuring the node
- Activate and deactivate game zones
- Prohibit certain types of player interaction with NPCs – talking, transferring, etc.
- Start the Emission, modify the Emission schedule
- Load and unload Data Layers specified in the node
- Force the player to equip a specific weapon slot, e.g., force them to take out a knife
- Forcefully turn the flashlight on and off
- Simulate player interaction with an interactive object
- Add a marker to a stash
- Add and remove selected items from the inventory
- Rotate the player's camera to a specified point
- Play a specified VFX

Quest Tool nodes

Activate Anomaly

This node lets you start the active phase of an anomaly using a script, as if it had been activated by a player or NPC. This is necessary for directed sequences with anomalies, e.g., the Prologue.

Node view



How to use

Parameter	Description
Continue	The parameter is responsible for the node's operation until the node is excluded. The node will send a signal to the anomaly for one second if this option isn't activated.
Target Placeholder	The anomaly is activated by the node

IMPORTANT

Each anomaly has its own specific activation logic. See the table below for more details:

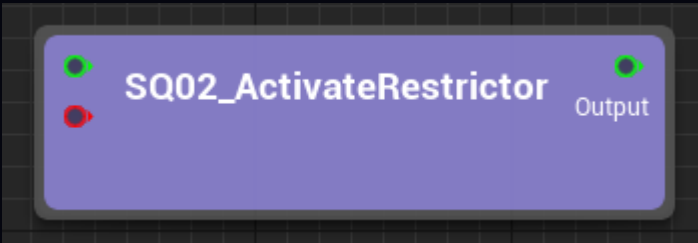
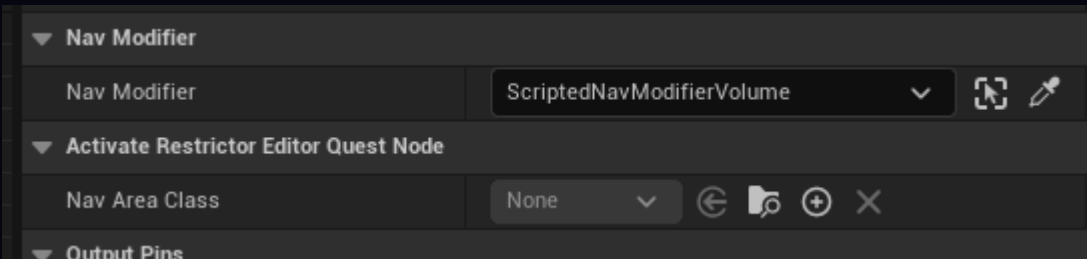
Anomaly	Single reaction	Continue reaction
Expulsion	Activated once	After reloading, it is activated until the node is excluded
Razor	The anomaly is always active, the node doesn't work on it	The anomaly is always active, the node doesn't work on it
Carousel	Activated once	After reloading, it is activated until the node is excluded
Flycatcher	The anomaly is always active, the node doesn't work on it	The anomaly is always active, the node doesn't work on it
Pillow	The anomaly is always active, the node doesn't work on it	The anomaly is always active, the node doesn't work on it
Burner	Activates once and works as if an object has entered the anomaly for 1 second	Activates and burns until the node is excluded
Clicker	Activated once	After reloading, it is activated until the node is excluded
Lava Lamp	Activated once	After reloading, it is activated until the node is excluded
Fruit Punch	Activates once and works as if an object has entered the anomaly for 1 second	Activates and works until the node is excluded
Soap Bubble	Activated once, the bubble bursts into small bubbles	Behaves similarly to Single Activate

Toxic Cloud	Activates once and works as if an object has entered the anomaly for 1 second	Activates and works until the node is excluded
Electro	Activated once	After reloading, it is activated until the node is excluded
Lightning Ball	Doesn't work	Doesn't work
Psy	The anomaly is always active, the node doesn't work on it	The anomaly is always active, the node doesn't work on it

Activate Restrictor

This quest node activates ScriptedNavModifierVolume, aka Space Restrictor.

The node will remain active until excluded. The node's exclusion disables ScriptedNavModifierVolume.

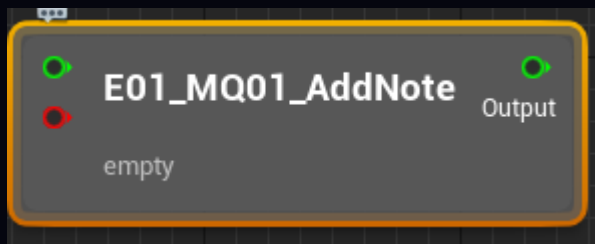
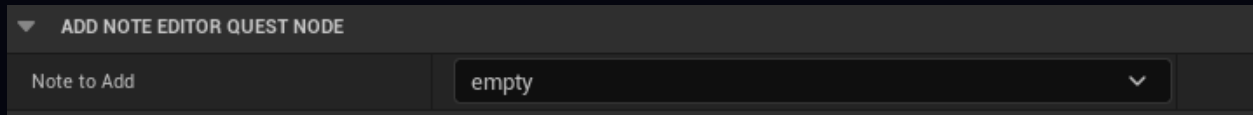


How to use

Parameter	Description
Nav Modifier	ScriptedNavModifierVolume is activated and deactivated by the node. Keep in mind that the restrictor works as long as the node is active. To disable the restrictor, exclude the node.

Add Note

This node allows adding notes and audio files to the player's PDA.

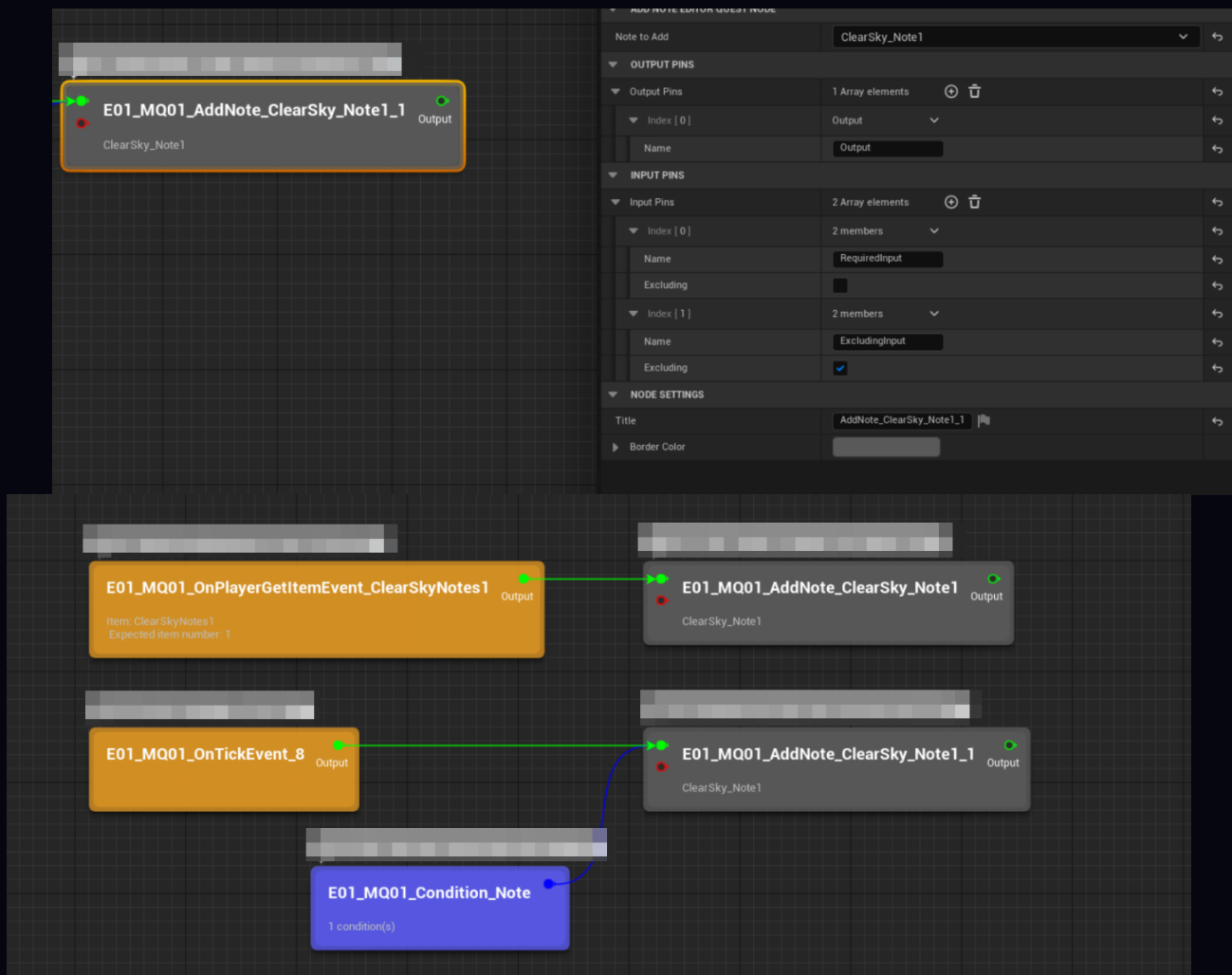


Use case

A narrative designer wants a new note to be added to the player's Notes after picking up a specific PDA. To do so, they create a check to see if the player has picked up the PDA and then activate the Add Note node. Once the node is activated, the note appears in the player's Notes.

How to use

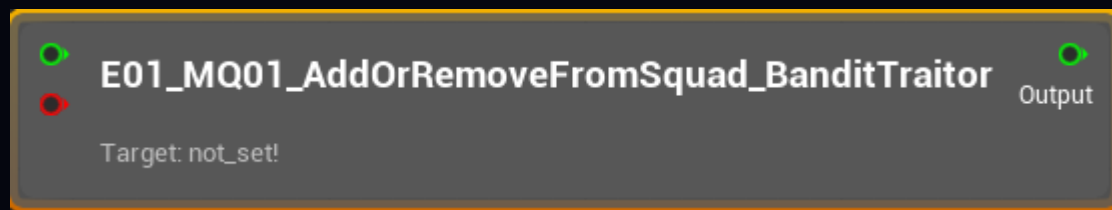
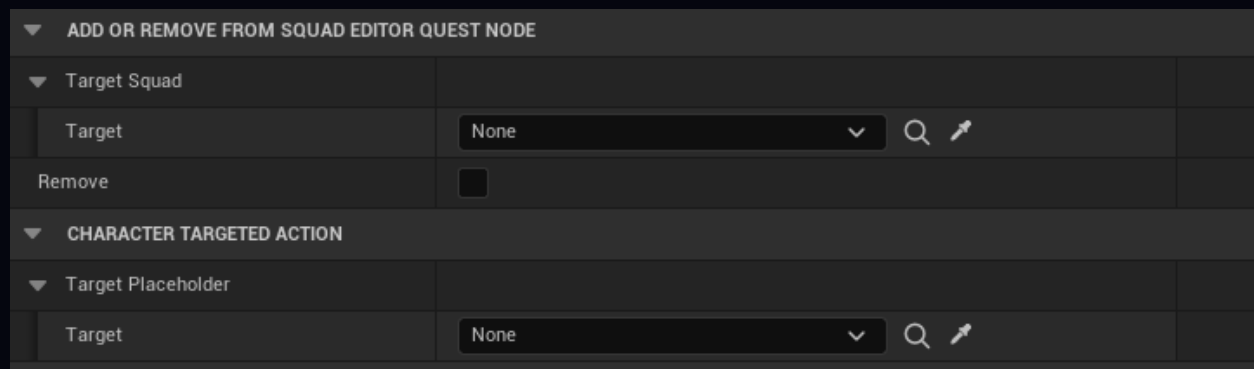
1. Create a note.
2. Create the config so that the note is added to the engine.
3. After the order has been placed in the general branch, be sure to update it.
4. After updating, restart the engine, create the node, and select the ordered note config in the **Note to Add** field.



Add Or Remove From Squad

This node allows adding and removing NPCs from the NPC squad spawner.

References



Use case

A narrative designer wants to create a sequence in which characters from one squad split into two rival squads. To do this, they remove half of the characters from the first squad and immediately add them to the other squad, after which they give both squads the goal of killing each other.

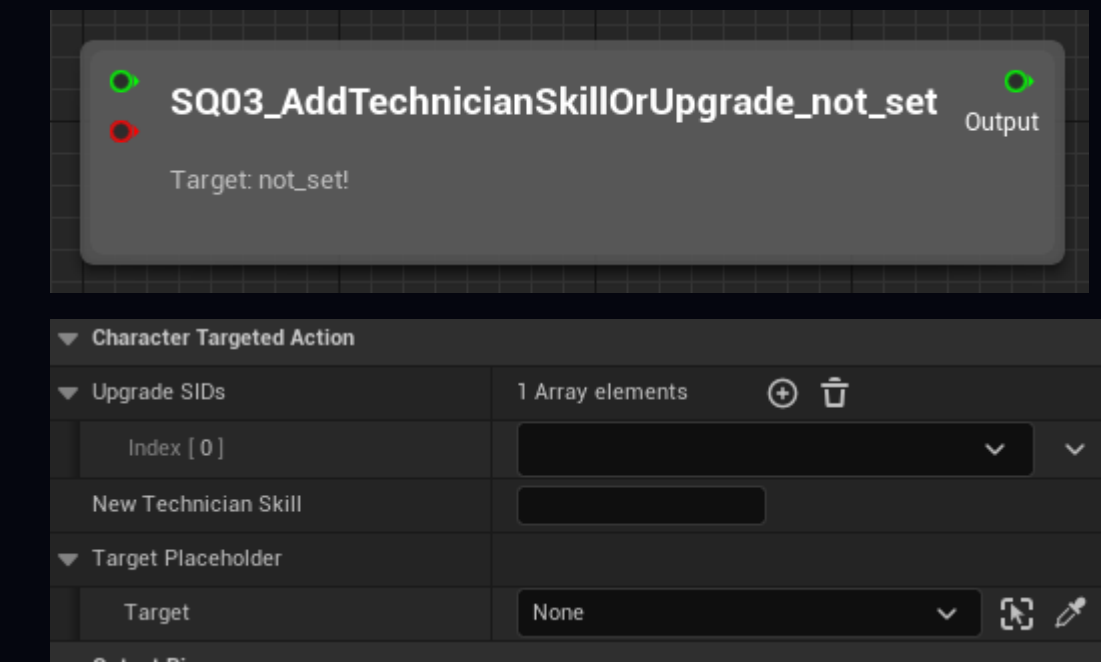
How to use

- 1. Configure the squad placeholder and NPC placeholder in these squads. Be sure to use a combination of two types of placeholders, as the node refers to specific NPCPlaceholders.
- 2. Select targets in **Target Squad** and **Target Placeholder** using the pipette or by selecting the desired targets from the options.
- 3. Select or clear the **Remove** checkbox based on whether you need to remove or add NPCs from the squad.
- 4. Check it in the PiE.

Add Technician Skill Or Upgrade

This quest node allows a technician to install the selected upgrade.

The node ends immediately after activation. Node exclusion doesn't cancel its effect.



How to use

Parameter	Description
Upgrade SID	Upgrade SID, which is available for installation. The information is taken from the config.
Target Placeholder	An NPC, which is given an ability

NOTE

Any NPC, not just a technician, can be a target placeholder. However, make sure that the target of the node is a technician because assigning an ability to a character who is not a technician will not allow them to use that ability.

A-Life Director Zone Switch

This node changes the status of AlifeZoneTrigger from active to inactive and vice versa.

In the **Action** dropdown list, select the action to be performed with the node – **Enable** or **Disable**.

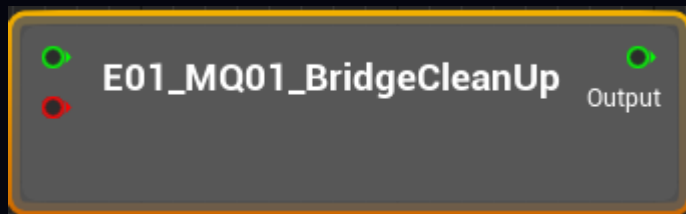
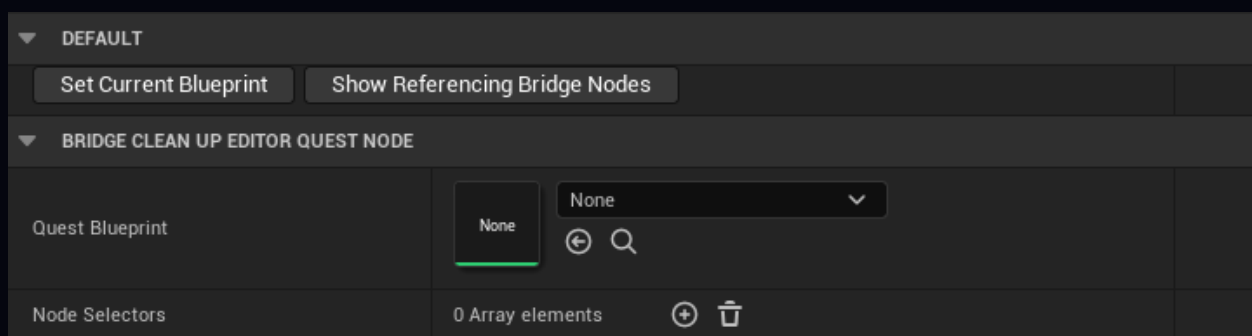
Specify the AlifeZoneTrigger whose status you want to change. It's also possible to overwrite the Scenario Group SID of the AlifeZoneTrigger zone.

How to use

1. Create the node.
2. Select the needed AlifeZoneTrigger.
3. Optionally, overwrite Scenario Group SID.
4. Save the changes.

Bridge CleanUp

This node clears information from the game's memory about whether any node has previously worked, allowing the bridge to be reactivated along with that node. This is necessary for repeatable quests.



Use case

A narrative designer wants to use bridges in a repeatable blueprint. To do this, they create a structure with bridges and, before completing the container, run the Bridge CleanUp node, which refers to nodes that are referenced by bridges.

How to use

Select the blueprint where the node to be cleaned up is located.

Quest Blueprint	<div>None</div>	<div>None</div>
-----------------	-----------------	-----------------

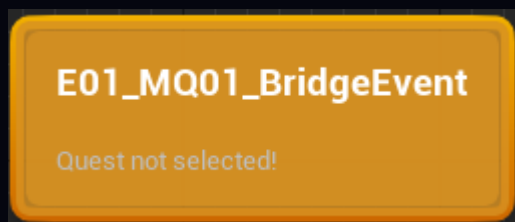
Select the nodes to be cleaned up, not the bridge itself.

Node Selector	<div></div>
---------------	-------------

Bridge Event

This node allows checking whether another node has completed, including the one in another blueprint. It is essentially a virtual connection between nodes.

Bridges have two key limitations: they cannot be used with dialog nodes that do not complete when a signal arrives at End (see *How to use* below for details), and they cannot be used without the Bridge CleanUp Quest Tool node in containers that will be restarted.



▼ DEFAULT		
<div>Generate Output Pins</div>		
<div>Go to Bridged Node</div>		
<div>Set Current Blueprint</div>		
<div>Show Referencing Bridge Nodes</div>		
▼ BRIDGE EVENT EDITOR QUEST NODE		
Quest Blueprint	<div>None</div>	<div>None</div>
Node Selector	<div></div>	

Use case

A narrative designer wants to convey information that a node has completed from blueprint A to blueprint B.

To reduce the number of twisted connections between nodes, the narrative designer adds bridge events to hide the connections.

How to use

In **Quest Blueprint**, select the quest blueprint whose node you would like to track.

In **Node Selector**, select the node you want to follow.

The node contains a **Go To Bridged Node** button for quick navigation to the node it refers to.

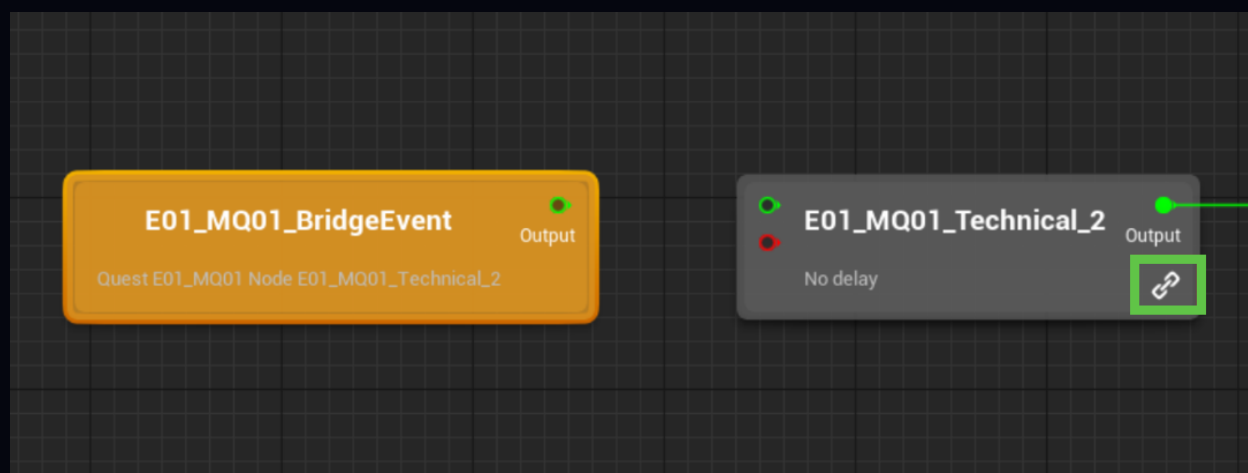
At the same time, each node contains a **Show Referencing Bridge Nodes** button.

If there are no bridges on the node, selecting this button will show an empty window.

If there are bridges:

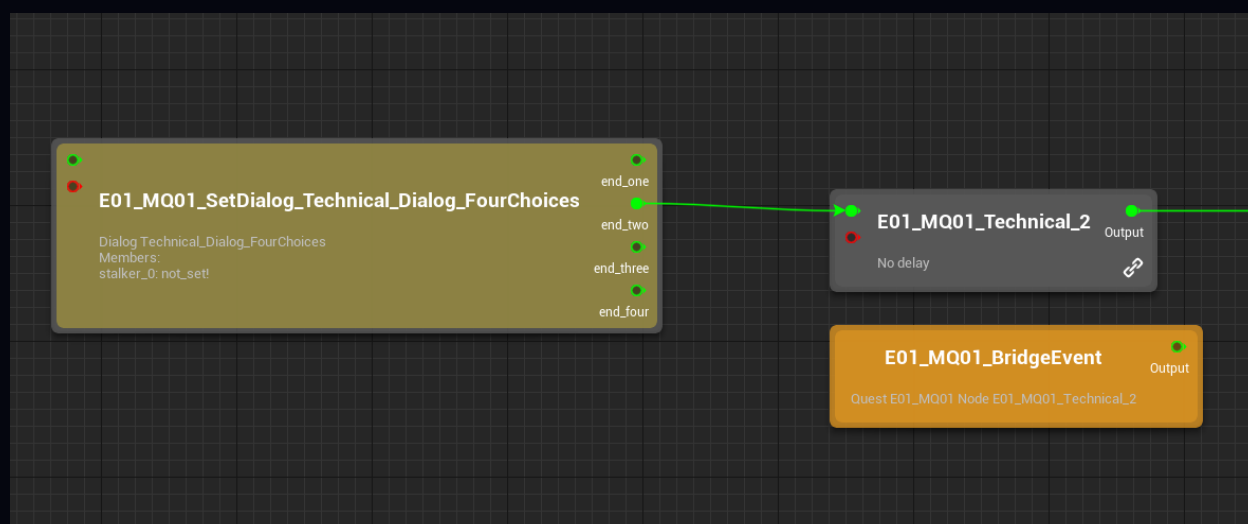
1. A chain links icon will appear on the node in the lower right corner. The icon appears, and the list becomes non-empty after saving the blueprint with the new bridge.
2. Selecting the **Show Referencing Bridge Nodes** button will open a window with a Bridge Nodes list, or immediately redirect to the Bridge Node if it is the only one referring to this node. This list will also include Bridge Conditions and IF Nodes from dialogs.

The node to which the Bridge was attached becomes locked. You cannot change the Output pins and fields that affect their change, rename, or delete them. You can copy, change connections, and move them around the blueprint.



IMPORTANT

You cannot use bridges with those dialog nodes that must send a signal without completion (using a special end). Since the node does not complete, the bridge will not work. Instead, create a connection between such a node and a technical node and bridge it.



Cancel All Side Quests

This quest node uses Cancel to complete all active logs with the **Is Main Quest** parameter set to **False**.

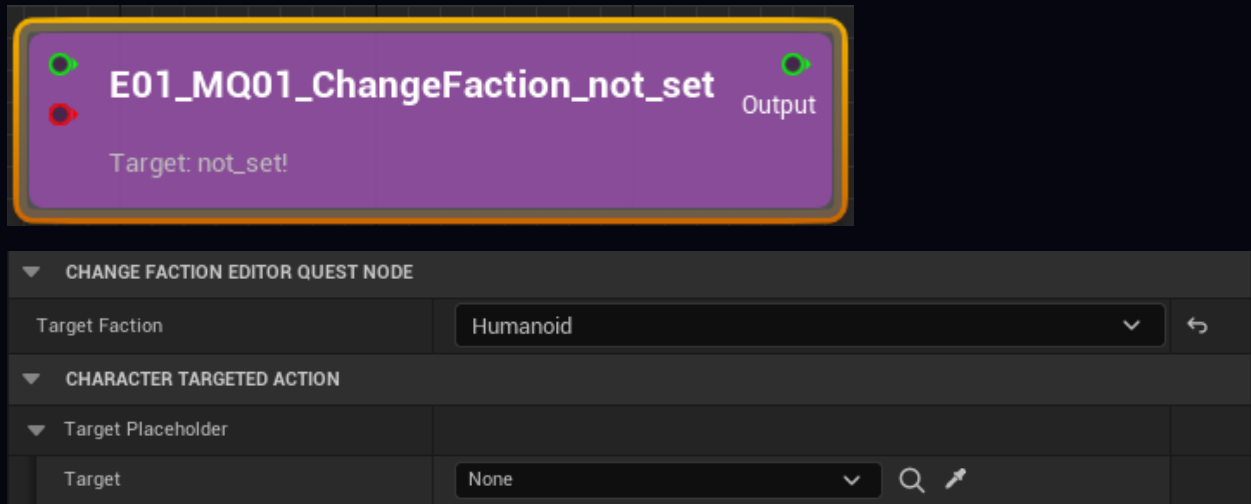
It is used after the player passes the No Return Point.

The node completes after execution. The node exclusion won't cancel its actions. The node has no parameters.

Change Faction

The node allows changing a character's faction. It is necessary for workarounds and plot moments so that the gameplay supports the narrative.

References



Use case

After the Signal, the narrative designer wants the character from Noontide to change the faction to Monolith. They use a node to change the character's faction.

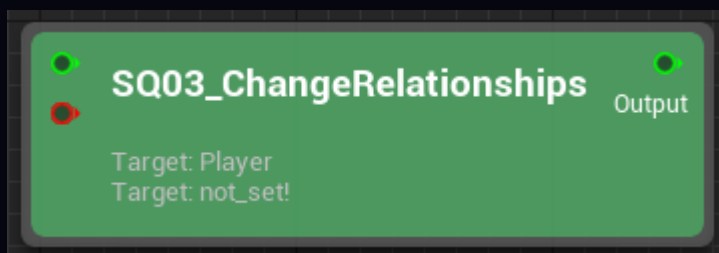
How to use

1. In the **Target Faction** field, select the faction to which the character's faction should change.
2. Select the desired character in the **Target** field with the pipette or select from the drop-down list of all actors.

Change Relationships

This quest node changes the relationship between two targets (NPC, squad, faction, player).

The node ends immediately after triggering. Node exclusion will not cancel its effect.



▼ Default

Show Referencing Bridge Nodes

▼ Relationships Target 1

▼ First Target Character or Squad

Target

None

↺

↻

Target Player

☐

↺

First Target Faction

☐

First Target Faction

Humanoid

↺

▼ Relationships Target 2

▼ Second Target NPCOr Squad

Target

None

↺

↻

Second Target Faction

☐

Second Target Faction

Humanoid

↺

▼ Relationship Value

Change Relationships Mode

Set Absolute Value

↺

Changed Value

0

Relation Preset

Neutral

↺

Set Faction Relationship as Personal

☐

How to use

Parameter	Description
First Target	<p>This is the first target of the node.</p> <p>Target Player – the player will be the first target of the node.</p> <p>First Target Faction – the selected faction or subfaction will be the first target of the node.</p> <p>The Target Player checkbox has higher priority and overrides all other target settings.</p>
Second Target	<p>This is the second target of the node.</p> <p>Second Target faction – the selected faction or subfaction will be the second target of the node.</p>

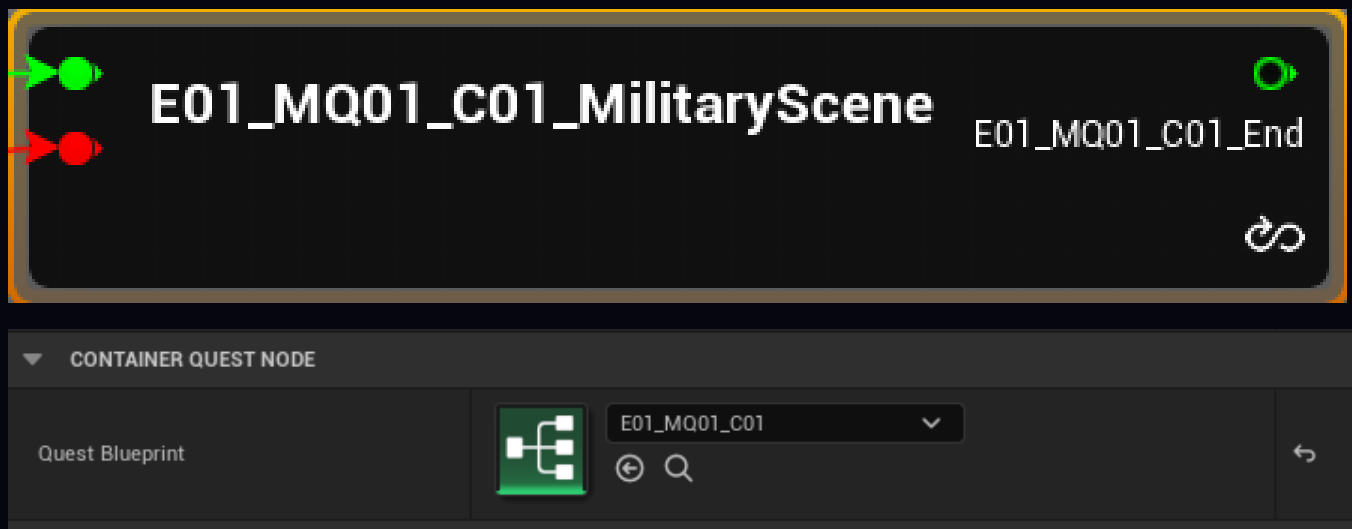
Change Relationships Mode	<p>This is the node operation mode.</p> <p>You can set the following:</p> <ul style="list-style-type: none"> • Absolute value – set an absolute value • Increase – add to the current value • Decrease – subtract from the current value • Use Preset – use one of 4 presets: <ul style="list-style-type: none"> ○ Enemy (-800) ○ Disaffection (-400) ○ Neutral (0) ○ Friend (800)
Use Faction Relationship as Personal	<p>This is the variable that converts factional relationships to personal ones.</p> <p>By default, personal relationships are of higher priority than factional ones. Activating the checkbox allows you to ignore the personal relationships between two targets, converting all relationship checks to factional ones.</p>

IMPORTANT

The node changes relationships both ways. This means that situations when the first target of the node is hostile to the second, and the second is neutral to the first, are impossible.

Container

The node allows running an embedded quest blueprint, thereby encapsulating and structuring the logic of quests.



How to use

Parameter	Description
Quest Blueprint	Selecting a blueprint launched by a node. Attempting to select a blueprint in which a specific node is located is impossible -- such an action is accompanied by a window informing you of the corresponding error.

A repeated parallel signal to the node restarts the quest blueprint selected in the node.

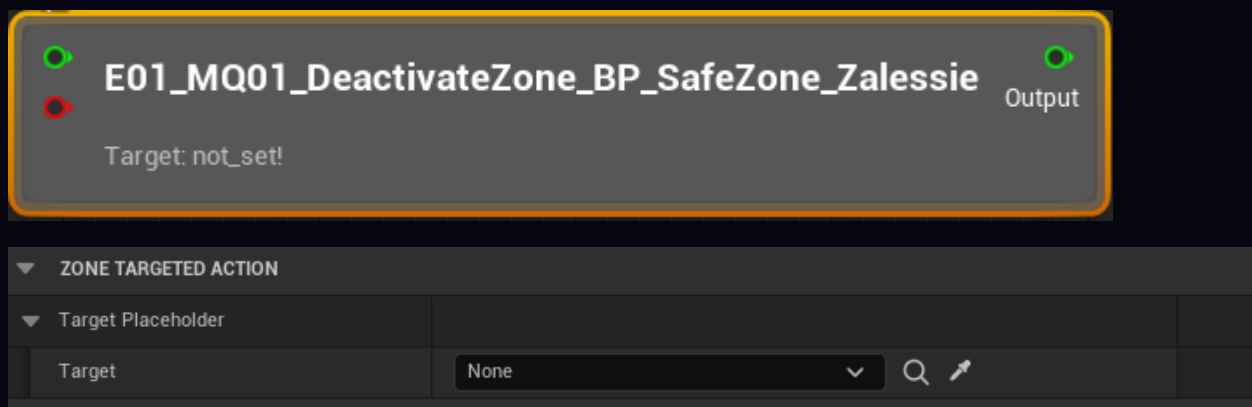
Deactivate Zone

This node allows pausing zones – SafeZone, Shelter, Hub, SpaceRestrictor.

When this node is started, all triggers that were part of this zone stop working as this zone. For example, if one trigger was part of both a safe zone and shelter, when the shelter deactivation node is started, that trigger stops functioning as a shelter but continues to function as a safe zone.

The closure logic is also deactivated for hubs during combat.

References



Use case

A narrative designer wants the hub to become a dangerous area where battles can occur after certain events in the game, for example, the Noontide Base after the Signal.

How to use

1. Select the configured and loaded Zone Placeholder in the node settings.
2. When the signal arrives at the node, the selected zone will be deactivated.
3. If you need to reactivate the zone, exclude the node.

The specificity of this exclusion means that such nodes must be placed in the region's quest blueprint so that the node is not excluded along with the quest container.

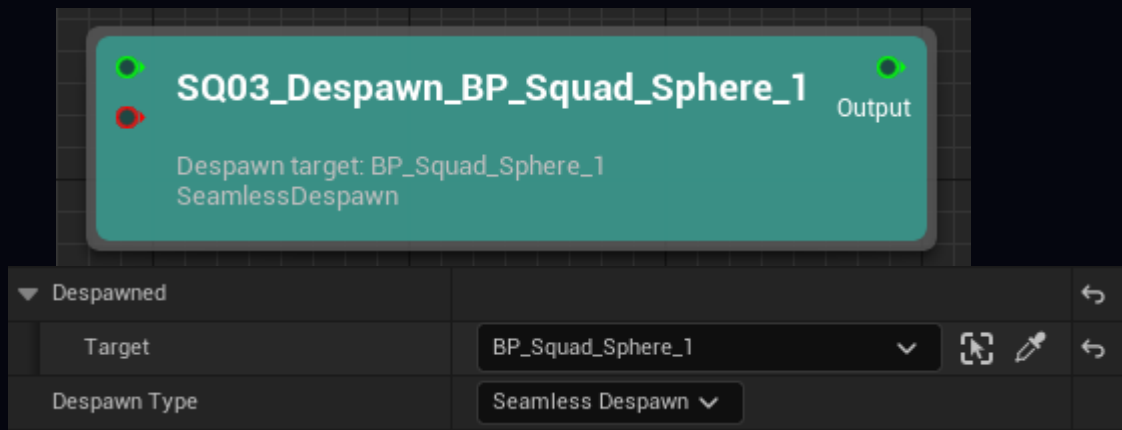
Despawn

This quest node allows despawning the selected character. Its functionality is similar to the Spawn node exclusion.

The node ends immediately after the selected character is despawned. If the selected character has not yet spawned, the node will remain active and will trigger after the character spawns.

The node exclusion does not cancel its effect.

Node view



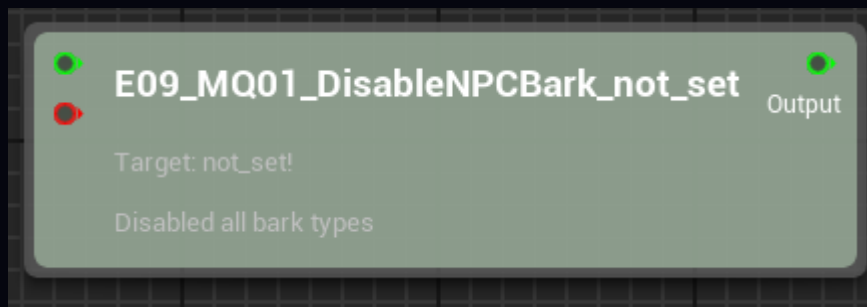
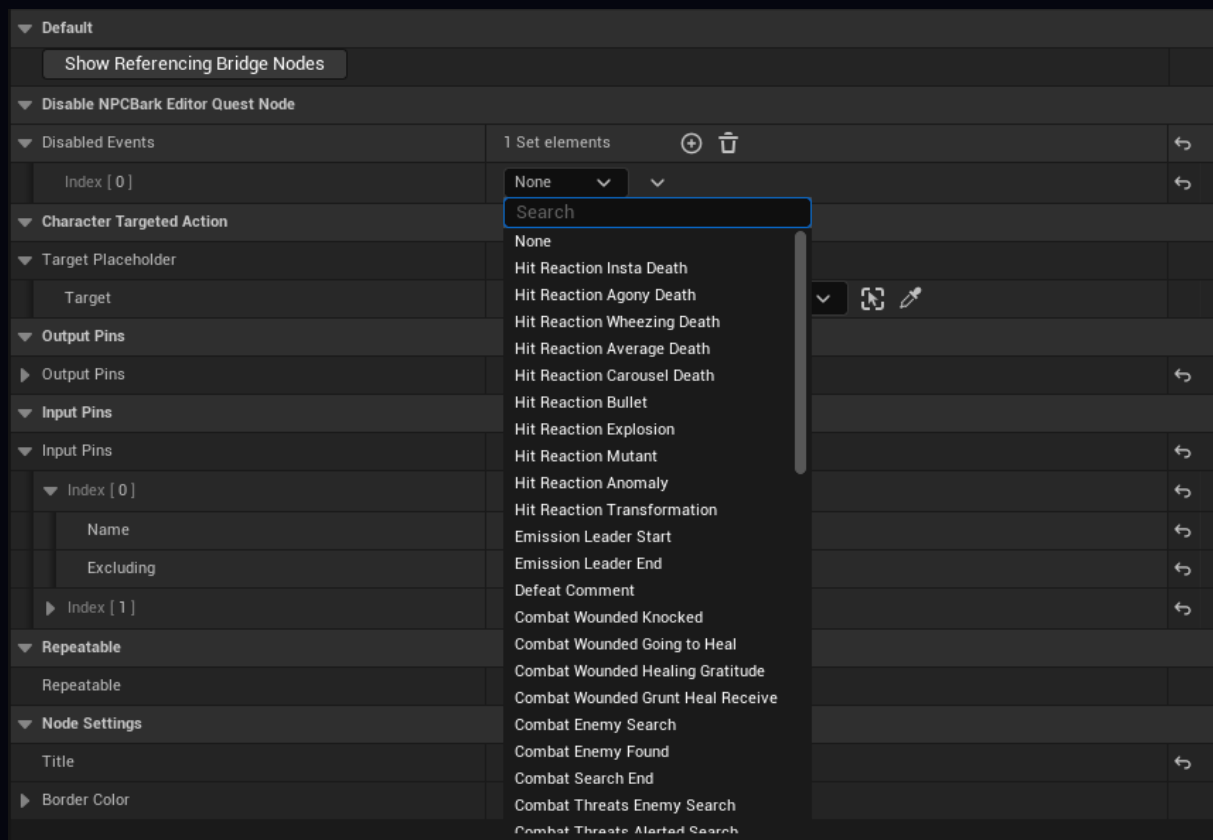
How to use

Parameter	Description
Target	The target to be despawned: NPC, squad, item, item container, anomaly placeholder, anomaly spawner, artifact spawner.
Despawn Type	<p>Ways of despawning the selected object:</p> <ul style="list-style-type: none">• Seamless Despawn – invisible despawn for the player, happens not in front of the player.• Force Despawn – despawn, may happen in front of the player.• Kill – kill, which works for NPCs, does not work for containers.• Specific Item Despawn – despawn of an item already spawned by the Spawn node.• Specific Item Despawn – despawn of the item wherever it is: in the player's inventory, in the player's hands, in the world, in a container, or in a corpse. <p>The placeholder that spawned this specific item is selected as the target.</p>

Disable NPC Bark

The node disables a specific bark for the selected NPC. The node disables the event that triggers the bark, not the dialog asset itself.

Node view



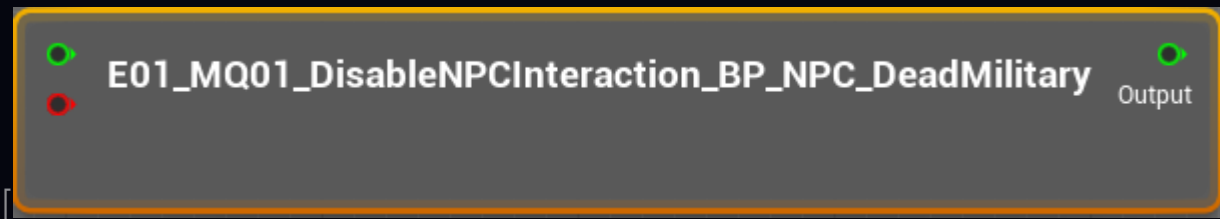
How to use

Parameter	Description
Target Placeholder	This is the NPC whose bark will be disabled. If the target is a squad, it will be disabled for all members. The node has a lower priority than the bark system. If, at the moment the node is triggered, the NPC is saying a bark, the node waits for the bark to end.
Disabled Events	This is a voiceover (bark) disabled by a node. One node can disable multiple events.

Disable NPC Interactions

This node can prohibit certain types of interaction with a living NPC or corpse.

References



DISABLE NPCINTERACTION EDITOR QUEST NODE		
Target	None	
Restrict Dialog Interactions	☒	
Restrict Defeat State Interaction	☐	
Restrict Defeat State Movement Interaction	☐	
Restrict Dead Body Movement Interaction	☐	
Restrict Dead Body Loot Interaction	☐	

Use case

- A narrative designer wants to prevent players from talking to a certain NPC until a certain point in the game. Using a node, the narrative designer prohibits conversation with this NPC.
- A narrative designer wants to prevent players from carrying away the body of an NPC they have killed. To do this, the narrative designer uses a node to prohibit the transfer of this NPC's corpse.
- A narrative designer wants to prohibit looting the NPC's corpse, but allow the player to move it. To do this, they prohibit looting the selected corpse using this node.

How to use

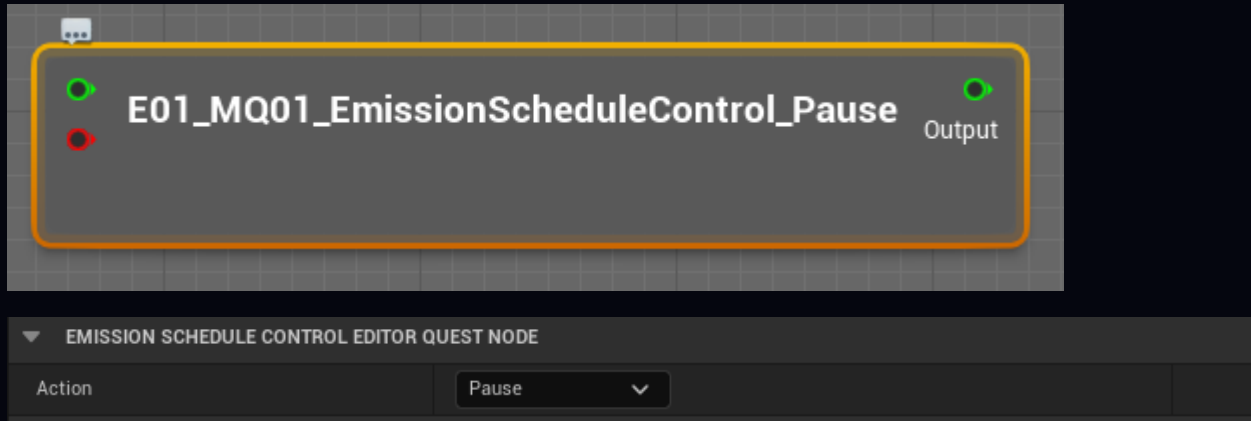
Setting	Type	Default	Function
Target	Reference	None	Allows selecting NPC placeholder, squad placeholder, or dead body placeholder as a reference.
Restrict dialog interactions	Bool	True	Prevents the ability to start a dialog with an NPC, except for force dialog, as described below. Instead of the default behavior, the character plays their Busy topic.
Restrict defeat state interaction	Bool	False	Prohibits interaction with NPCs in the wounded state. In this case, the corresponding Defeat state topic is not triggered. The behavior is

			equivalent to the wounded state for hostile characters.
Restrict unconscious state movement interaction	Bool	False	Prevents moving NPCs who are unconscious.
Restrict dead body movement interaction	Bool	False	Prevents moving the NPC's corpse. When interacting, Skif plays the prohibition interaction with corpses topic.
Restrict dead body loot interaction	Bool	False	Prevents looting of the NPC's corpse.
Restrict dead body despawn	Bool	False	Prevents the corpse from despawning. The node's values override the values set in the corpse's placeholder.

Emission Schedule Control

The node allows manipulating the schedule: pause and activate it again. It is needed to implement directed episodes of the main storyline and solve the issue of negative scenarios in side quests.

References



Use case

During the Duga attack, a narrative designer wants to prevent the Emission from interrupting the quest sequence. To do this, they pause the Emission schedule, restoring it only after the quest is completed.

How to use

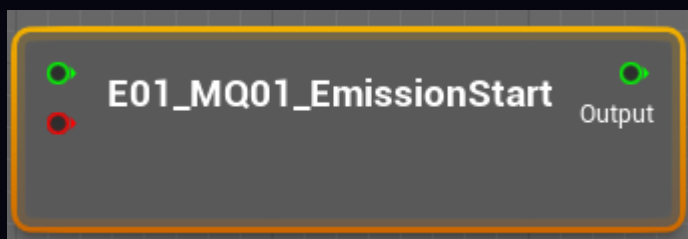
To pause, activate the node with the Pause setting. To continue the sequence, activate it with the Resume setting. If you pause the Emission somewhere, be sure to reactivate it at the end of the script. Otherwise, there will be no more emissions in this game.

Emission Start

The node allows you to start the Emission according to a script, bypassing the emission schedule. Also, activate unique quest emissions with special behavior in addition to the default Emission.

In the node settings, there is the **Force Emission** checkbox, which, when activated, allows starting the Emission immediately after a signal is received by the node, bypassing the weather blend rules.

References



EMISSION START EDITOR QUEST NODE		
Emission Prototype SID	empty	
Default Prototype	☐	
Start Quest	☐	

Use case

- A narrative designer wants a scripted Emission to start at a certain point in the quest. They use this node to activate it at the right moment in the script.
- A narrative designer wants to create a situation where the player's actions deceive the Emission alert system, making the characters believe that the Emission is about to begin. To do this, they order configurations for a fake Emission from game designers and then activate it using this node.
- A narrative designer wants a standard Emission to start in the middle of the quest, but without a separate Find Shelter quest appearing, so as not to disrupt the flow of the quest. They stop the Emission schedule and use the node settings to launch a default Emission where the quest does not start.

How to use

The list of available Emissions is specified in the config.

`Z:\git\ls2gamebuild\Content\GameLite\GameData\EmissionPrototypes.cfg`

The **Default Prototype** setting allows selecting the default emission as the Emission. The Start Quest setting allows choosing whether or not to start the *Find shelter from emission* quest.

When using the node, keep in mind that a regular Emission needs some time to get going due to the gradual weather blend.

In the node settings, there is the **Force Emission** checkbox, which, when activated, allows you to start the Emission immediately after the signal is received by the node, bypassing the weather blend rules.

Equip Item

The node allows you to forcefully equip a player/NPC/squad with an item in the selected slot and forcefully give that item to them.

References

●
●

E01_MQ01_EquipItemInHands

●
Output

Target: Player

EMainHandEquipmentType::None

empty

CHARACTER TARGETED ACTION		
Equipment Type Filter	Pistol	↩
Item SID	GunPM_HG	↩
Target Placeholder		
Target	None	🔍 ✎
Target Player	☒	

Use case

A narrative designer needs the player or NPC to pick up a specific weapon, detector, etc., at a certain point in the story. To do this, they set this item in the node settings, and it is forcefully equipped in the slot in the game.

How to use

Set a slot type in the **Equipment Type Filter** field: **Pistol**, **Primary Weapon**, **Secondary Weapon**, **Knife**, **Grenade**, **Bolt**, **Binoculars**.

CHARACTER TARGETED ACTION		
Equipment Type Filter	Bolt	↩
Target Placeholder		
Target		🔍 ✎
Target Player		
OUTPUT PINS		
Output Pin	Bolt	🔍 ✎

After the slot is selected, the **Item SID** field appears in the node.

Equipment Type Filter	Grenade	↩
Item SID	GrenadeRGD5	↩
Target Placeholder		
Target	None	🔍 ✎
Target Player	☒	

In **Item SID**, select the specific item that you want the player/NPC to equip instead of the current one. Also, in the player's case, the equipped item is immediately selected as the active weapon. If the player does not have a suitable item in their inventory, the node will be automatically excluded.

Item SID cannot be set for the **Knife**, **Bolt**, and **Binocular** slots, so selecting them will cause the player to forcibly take out the selected item.

Flashlight On\Off

This node allows activating and deactivating the player/NPC's head flashlight.



How to use

Parameter	Description
Target	<p>The character whose flashlight will be turned off.</p> <p>The Target Player checkbox lets you select the player as the target.</p>
Flashlight Status	<p>The type of interaction with the selected character's flashlight:</p> <ul style="list-style-type: none">• On – turn on the flashlight, one-time action.• Off – turn off the flashlight, one-time action.
Disable Flashlight Control	<p>This checkbox determines whether the selected character can manually change the flashlight status.</p> <p>Activating this checkbox means that the node will not end on its own, and therefore, its effect will not end until the node is manually excluded. Without activating this parameter, the node will end immediately.</p>

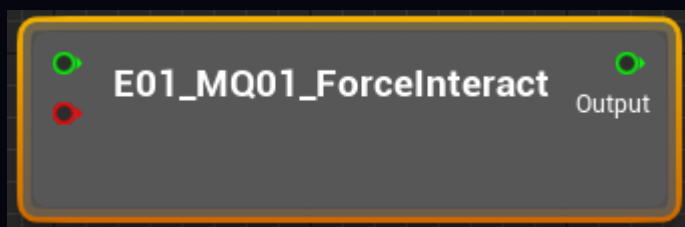
- If the node target is a squad, flashlight settings are configured for all its members. A node running in parallel and configured for one of the squad members will have higher priority and will change the character's settings. Exclusion of this node will return the character to their squad's flashlight setup.
- Only one node of a certain type (Single/Squad) can be active at a time for a specific target. Activating any new node of the same type (Single/Squad) for the same target will complete the work of the previous one, regardless of its parameters, i.e., activating a single-moment node (DisableFlashlightControl == false) can complete a long-running node (DisableFlashlightControl == true).
- Flashlight settings are multi-layered in terms of priority:
 - Basic level – a flashlight is controlled by an AI/player.
 - High level – forced setting by the Squad node.
 - Highest level – forced setting by the Single node.

That is, the flashlight state at a given moment is determined by the highest priority active level at that moment, regardless of the order in which these levels are activated.

Force Interact

The node simulates player interaction as if the player approached an interactive object and pressed F.

References



FORCE INTERACT QUEST NODE		
Interactable		
Interactable Actor	None	
Interactable Object Guid	{00000000-0000-0000-0000-000000000000}	
Ignore Enabled Check	☒	

Use case

In a quest that uses the corpse transfer mechanic, a narrative designer uses Force Interact on the transferred corpse to force the player to lower the body.

How to use

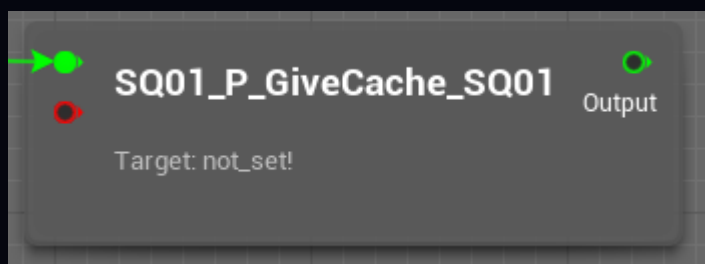
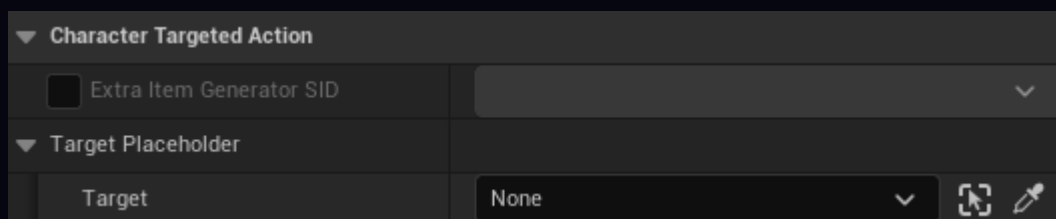
Field	Description
Interactable	The selected actor on the map. When selecting and saving, checks should be performed to see if this actor has an interactable component (as in the Activate Interactable quest node) in case it is a lever/door/button/etc. located on the map OR if the actor is a placeholder of one of the following types: NPC placeholder, dead body placeholder, item placeholder, item container placeholder.
blgnoreEnabledCheck	If this parameter is active, interaction occurs even if the object is not available for normal interaction via F. If the parameter is inactive, interaction occurs when the object becomes available for interaction or immediately if it is already available.

Give Cache

This quest node sets a stash marker on the map.

It completes after the marker is displayed. Exclusion of the node will not cancel its effect.

Node view



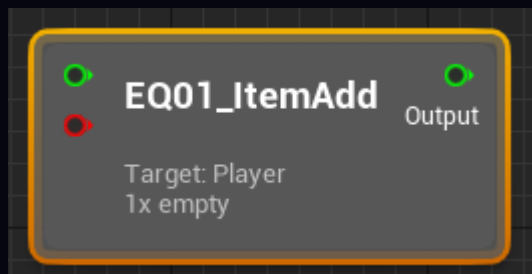
How to use

Parameter	Description
Target	Item Container, the coordinates of which will be given to the player.
Extra Item Generator	The ability to add an extra Item Generator to the stash.

Item Add

The node adds the selected item to the player's inventory, NPC, dead body placeholder, or container. It is required for issuing rewards, spawning quest items in containers, or transferring items from the player's inventory to the NPC's inventory.

References



CHARACTER TARGETED ACTION		
Item SID	A045A	↩
Items Count	1	
Target Placeholder		↩
Target	BP_NPCPlaceholder_Dima	↩
Target Player	☐	↩

Use case

- A narrative designer needs to give the player a PM as a reward for completing the quest. To do this, the narrative designer creates an Item Add node in the blueprint, selects GunPM_HG в качестве ItemSID, sets **Items Count** to 1, and sets the **Target Player** checkbox in **Target Placeholder**. When the node receives a signal, the PM appears in the player's inventory.
- During quest development, a narrative designer wants to show that the stalker managed to find an artifact in the anomaly before dying. To do this, the narrative designer creates an Item Add node in the blueprint, selects the required artifact as **Item SID**, and selects the desired dead body placeholder in **Target Placeholder**. When the node receives a signal, the selected artifact will appear in the dead stalker's inventory.

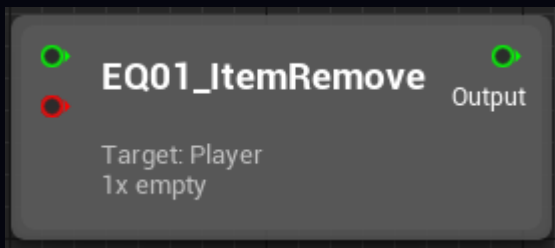
How to use

- Create the node.
- Select the required value from the list in **Item SID**.
- Select the quantity in **Items Count**.
- Select **Target Placeholder**.
- Save the changes.

Item Remove

The node removes the selected item from the player's inventory, NPC, dead body placeholder, or container. It is necessary to remove quest items from the player, simulate the loss of items from NPCs, and simulate the transfer of items from one inventory to another.

References



CHARACTER TARGETED ACTION	
Item SID	Empty
Items Count	1
Target Placeholder	
Target	None
Target Player	☒

Use case

- A narrative designer needs to create a case where the player opens the door with a key, after which the key disappears from the inventory. The narrative designer creates an Item Remove node, selects Quest_Key_Example in **Item SID**, sets the value to 1 in **Items Count**, and selects **Target Player** as the **Target Placeholder**. After the signal reaches the node, the key will be removed from the player's inventory.
- A narrative designer has come up with a scenario in which the player can obtain an item from an NPC either peacefully or by killing them. In the first case, the item must be removed from the NPC's inventory. The narrative designer first adds the item to the NPC, then creates an Item Remove node in the blueprint, selects the item in **Item SID**, sets the value in **Items Count**, and selects the NPC in **Target**. After the signal reaches the node, the item will be removed.

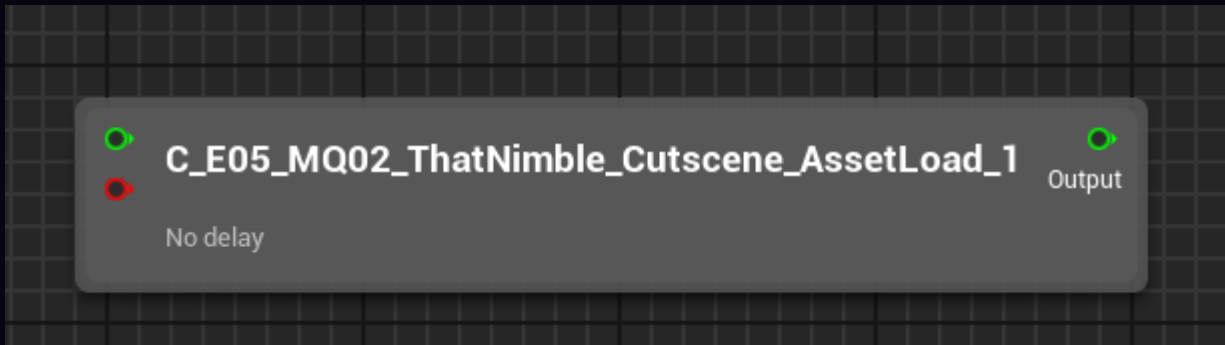
How to use

- Create the node.
- Select the required value from the list in **Item SID**.
- Select the quantity in **Items Count**.
- Select **Target Placeholder**.
- Save the changes.

Load Asset

The node loads assets and holds them in memory. It can be used for massive assets that will be used in the game in some period of time.

As an example, using this node before entering a cutscene with a LevelSequence asset of a cutscene will prevent waiting for the LevelSequence asset to load, so the cutscene will be started immediately. The node should be excluded when the asset is no longer required.



How to use

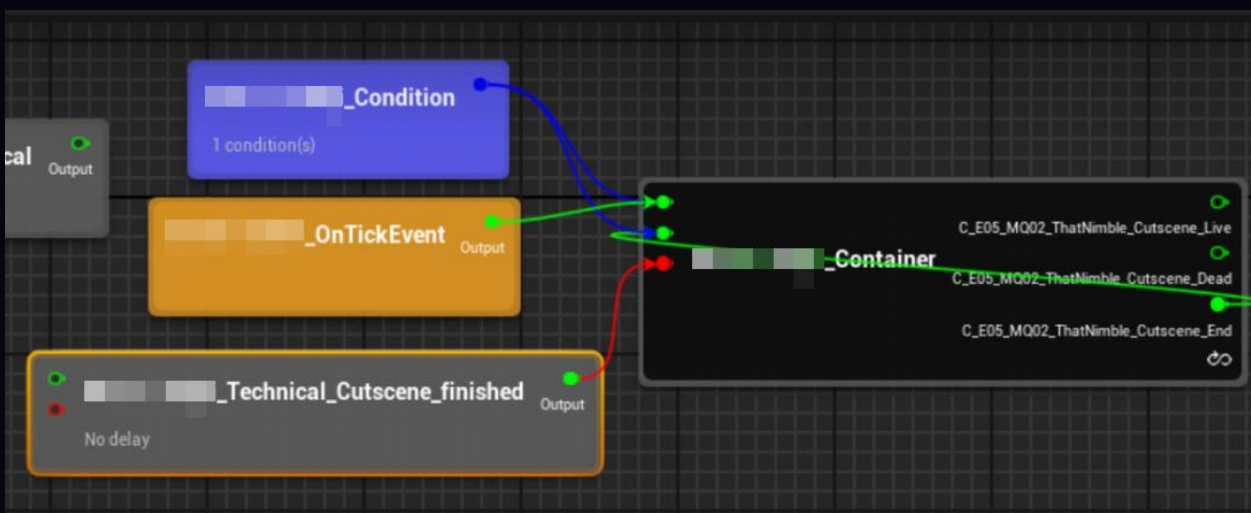
Place the node in a graph a few seconds before the event where AssetToLoad will be used. Massive assets take more time to load – please keep it in mind while scripting with this node.

Use the node under the following conditions:

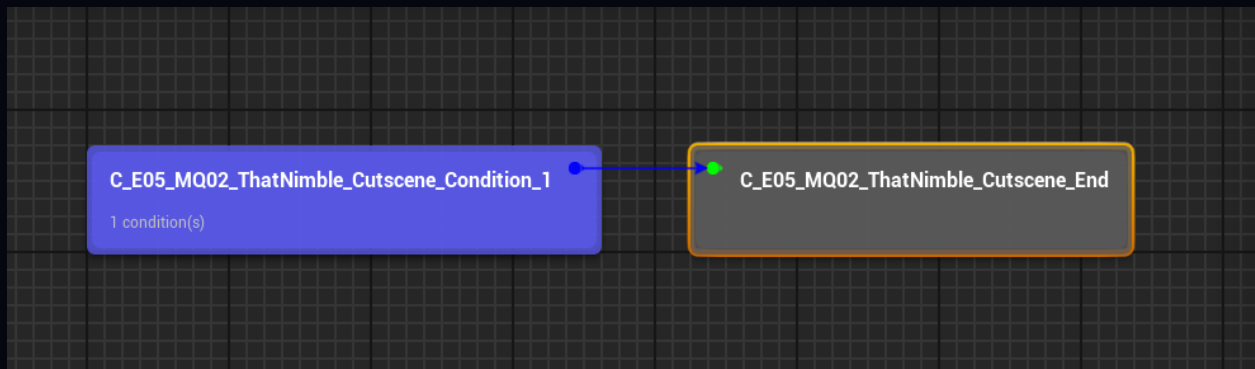
1. Create a repeatable container that has the **ExcludeAllNodesInContainer** parameter set to **True**.
2. Connect LoadAssetQN to start or to another node if needed.
3. Connect ContainerQN and start and exclude it by some condition. As an example, if the player is in a trigger, start ContainerQN. Else, exclude ContainerQN.

As an example, this way you will be able to preload the cutscene before activating it.

The *player is in a trigger* condition is used outside the container.



The *player is not in a trigger* condition is used inside the container.

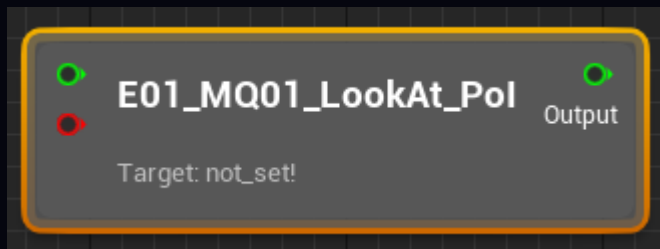


Parameter	Description
AssetsToLoad	Assets that will be loaded at the start of the node.
AudioLocalizationAssetsToLoad	Assets that can be loaded at the start of the node by the localization condition used in the current player's game.

Look At

A node that allows a narrative designer to forcefully direct the player's camera or NPC's sight line to a selected object using the Look At system.

This node has a higher priority than turning the player's camera toward the speaker during a dialog.



▼ Common Settings

Attraction Point TypeStatic Location ▼

▼ Default

Show Referencing Bridge Nodes

▼ Character Targeted Action

▼ Target Placeholder

TargetNone ▼ [icon] [icon]

Target Player☒

▼ Look at Actor/Coordinates

Target LocationNone ▼ [icon] [icon]

▶ Target Location Coordinates0.00.00.0

▼ NPC Settings

Reaction Time0.0

Priority0

Collision ChannelWorldStatic ▼

Apply Restriction TypeLeave Same ▼

▼ Player Settings

Player Entering Duration1.0

Player Look at Preset NameNone

▼ Custom Preset

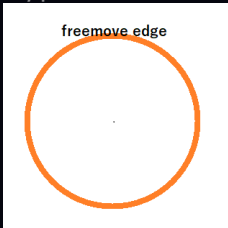
Rotation Freemove Edge20.0

Rotation Stop Edge60.0

Should Repeat Target Movement☐

How to use

Parameter	Description
	Basic node function Static location – look at a static object Follow Object – look at a dynamic object Follow Character Bone – look at the selected Skeletal Mesh (NPC) bone Exit Look At – end all Look At actions for the character Look at Static Object – bring an Empty Actor onto the stage and set the node on it. Look at NPC – use Follow Character Bone with the setting bone name = jnt_head, so that the NPC's head will be the center of attention.

Target Placeholder	The character who is looking
Look At Actor/Object	The character or object being looked at
Reaction Time	<p>The time after which the node will activate</p> <p>Used for situations where the character's reaction is not immediate, for example, when you need to look at an explosion.</p>
Priority	<p>Priority for situations where several nodes are set on a character.</p> <p>IMPORTANT Please avoid such situations and remember that dialogs already have a built-in camera turn to the speaker.</p>
Apply Restriction Type	Please do not use this parameter
Player Entering Duration	The time it takes for the player's camera to turn to the point
Player Look At preset Name	The name of the Look At sector settings preset for the player
Rotation Freemove Edge	<p>The sector in which the camera rotation is not limited when the node is active</p> <p>The sector is measured from the center of the mass target. Below is a hypothetical example from the player's perspective.</p> 
Rotation Stop Edge	<p>The sector in which a “rubber band” is attached to the camera rotation when the node is active. It can be less than the Freemove Edge value.</p> <p>The sector is calculated from the boundaries of the Freemove Edge sector. Below is a hypothetical example from the player's perspective.</p> 

IMPORTANT

Please use the node only when absolutely necessary, when the player needs to be prompted on where to look.

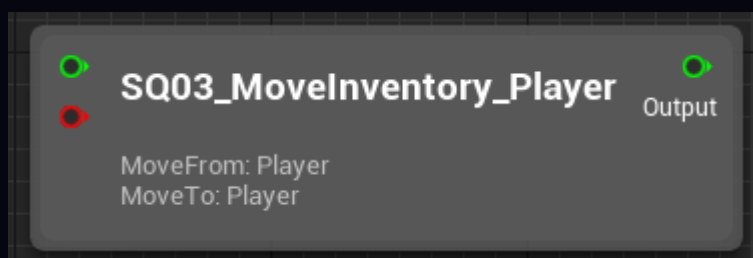
Also, keep in mind that if the player is initially turned away from the point, the camera movement will be very noticeable and inconvenient if the correct camera rotation time is not set.

Move Inventory

This quest node allows moving items from one container to another.

It completes after the items are moved. Node exclusion does not cancel its effect.

Node view



▼ Character Targeted Action		
▼ Move From		↶
Target	None ▼	🔗 ✎
Target Player	☐	↶
▼ Move To		↶
Target	None ▼	🔗 ✎
Target Player	☐	↶
With Equipped	☒	

How to use

Parameter	Description
Move From Target	The container from which the item will be transferred. If the target is a squad, items will be taken from all its members. The Player Inventory checkbox takes priority over the placeholder. Activation overrides Placeholder Target.
Move To Target	The container to which the item will be moved. If the target is a squad, items will be moved to the squad leader.
With Equipped	A variable that allows items to be moved from equipment slots if Move From Target is an NPC or the player.
Move Quest Items	It indicates whether items with the IsQuestItem set to True will be moved. The default value is False.

IMPORTANT

A container in the context of a node is any actor that has an inventory, namely: player, NPC, squad, item container, etc. Do not overuse moving items from the player's inventory.

NPC Bark

A node that triggers the selected Voiceover (bark) for the selected NPC.

How to use

Parameter	Description
Target Placeholder	<p>The NPC for which the bark will be triggered.</p> <p>If the target is a squad, the bark will be triggered for the squad leader.</p> <p>The node has a lower priority than the bark system. If, at the moment the node is triggered, the NPC is saying a bark, the node waits for the bark to end.</p> <p>If the selected NPC does not have a selected bark, the node will simply end, and no bark will be played.</p>

Bark Event

Voiceover (bark) triggered by a node.

The list below contains current generic barks for NPCs, divided into voiceover categories. The higher the voiceover category in the list, the higher its priority.

- NPC interaction barks

Event name: *General_CharacterSID_NPC_Interact_****

- Dialog system

- HelloTopic
 - ByeTopic
 - BusyComment

- NPC reaction on approach

- NPC_Interact_Hide_Weapon
 - NPC_Interact_Friendly
 - NPC_Interact_Neutral
 - NPC_Interact_NonFriendly
 - NPC_Interact_Bump

- Squad leaders

- NPC_Interact_SpeakToLeader

- Emission barks

Event name: *General_CharacterSID_NPC_Emission_****

- AI reaction to emission

- NPC_Emission_LeaderStart
 - NPC_Emission_LeaderEnd

- Combat barks

Event name: *General_CharacterSID_NPC_Combat_****

- Wounded state

- NPC_Combat_Wounded_Knocked
 - DefeatComment
 - NPC_Combat_Wounded_GoingToHeal
 - NPC_Combat_Wounded_HealingGratitude
 - NPC_Combat_Wounded_Grunt_HealReceive

- General_zombie_0_DefeatComment_Group_Agnostic

- AI enemy search

- NPC_Combat_EnergySearch
- NPC_Combat_EnergyFound
- NPC_Combat_SearchEnd

- Threats

- NPC_Combat_Threats_EnergySearch
- NPC_Combat_Threats_AlertedSearch
- NPC_Combat_Threats_AlertedSearchEnd
- NPC_Combat_Threats_ThreatDetected

- Combat speech

- NPC_Combat_Start
- NPC_Combat_Over
- NPC_Combat_EnergyRetreat
- NPC_Combat_SelfRetreat

- Combat speech + combat actions

- NPC_Combat_Action_Cover
- NPC_Combat_Action_Move
- NPC_Combat_Action_Flank
- NPC_Combat_Action_Detour
- NPC_Combat_Action_EnergyHit
- NPC_Combat_Action_FriendlyHit
- NPC_Combat_Action_FriendlyFire
- NPC_Combat_Action_Reload
- NPC_Combat_Action_FireSuppression
- NPC_Combat_Action_FriendlyGrenade
- NPC_Combat_Action_EnergyGrenade
- NPC_Combat_Action_FriendlyDead
- NPC_Combat_Action_EnergyDead

- General_zombie_0_NPC_Combat_Zombie_Attack
- General_zombie_0_NPC_Combat_Zombie_Moan
- General_zombie_0_NPC_Combat_Threats_ThreatDetected_Group_Agnostic

- Hit reaction barks

Event name: *General_CharacterSID_NPC_HitReaction_****

- Death

- NPC_HitReaction_InstaDeath
- NPC_HitReaction_AgonyDeath
- NPC_HitReaction_WheezingDeath
- NPC_HitReaction_AverageDeath
- NPC_HitReaction_CarouselDeath
- General_zombie_0_NPC_HitReaction_InstaDeath

- NPC hits

- NPC_HitReaction_Bullet
- NPC_HitReaction_Explosion
- NPC_HitReaction_Mutant
- NPC_HitReaction_Anomaly
- General_zombie_0_NPC_HitReaction_Transformation
- General_zombie_0_NPC_HitReaction_Bullet
- General_zombie_0_NPC_HitReaction_Anomaly

- Quest contextual action barks

- Peaceful reaction barks

Event name: *General_CharacterSID_NPC_Reaction_****

- Drag the dead body

- NPC_Peaceful_CorpseHubComment
- NPC_Peaceful_DropCorpse

- Corpse looting

- NPC_Peaceful_LootingEnemyCorpse

- NPC_Peaceful_LootingFriendlyCorpse
- AI call player
 - CallPlayer
- Daily schedule and needs
- Chatter

- Contextual action barks

Event name: *General_CharacterSID_NPC_Contextual_****

- Contextual action
 - NPC_Contextual_Monolog
 - NPC_Contextual_Dialog
- Daily schedule and needs
- Chatter
- Grunts
 - NPC_Contextual_Negative_Reaction
 - NPC_Contextual_Positive_Reaction
 - NPC_Contextual_Depression
 - NPC_Contextual_Failure
 - NPC_Contextual_Physical_Exercise
 - NPC_Contextual_Happiness
 - NPC_Contextual_Excitement
 - NPC_Contextual_Laugh
 - NPC_Contextual_Stretch
 - NPC_Contextual_Heavy_Exhale
 - NPC_Contextual_Fart_Reaction
 - NPC_Contextual_Wake_Up
 - NPC_Contextual_Moan_Pain
 - NPC_Contextual_Moan_Punch
 - NPC_Contextual_Thought
 - NPC_Contextual_Unsure
 - NPC_Contextual_Relief

	<ul style="list-style-type: none"> ■ NPC_Contextual_Humming ■ NPC_Contextual_Drunk ■ NPC_Contextual_Vodka ■ General_zombie_0_NPC_Contextual_Zombie_Slurred_Speech ■ General_zombie_0_NPC_Contextual_Zombie_Speech <ul style="list-style-type: none"> ● Radio, loudspeaker barks Event name: <i>HubName_Hub_PropTopic_***</i>
--	--

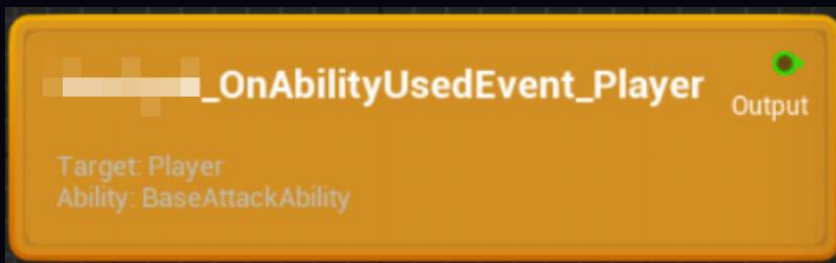
On Ability Start/End Event

These event nodes send a signal after the selected character starts or finishes using the selected skill.

Only works with NPCs or squads.

The node can be Repeatable.

Node view



▼ Targeted Event			
Ability SID	BaseAttackAbility	▼	↶
▼ Target NPCPlaceholder			
Target	None	▼	🔍 ✎
Target Player	☒		

How to use

Parameter	Description
Target Placeholder	The NPC or squad that should use the skill. The node will work for a squad when any member of the squad uses the skill.
Ability Used	The skill, which, when used, will cause the node to send a signal.

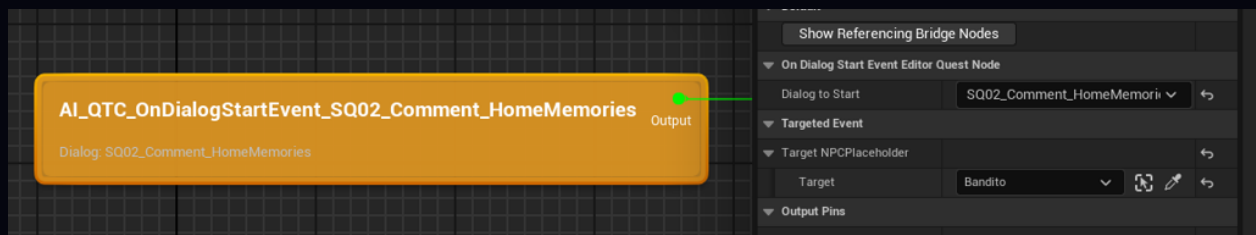
IMPORTANT

If an NPC cannot use the selected skill (it is not in the list of skills available to them), the node will linger indefinitely and wait until the character uses that skill.

On Dialog Start Event

An event node that tracks the moment when the selected NPC begins to speak the selected dialog or comment. This is either the moment when forced dialog (with ignore global WFR) or comments are triggered, or the moment when a topic is selected in global WFR. The Ignore global WFR checkbox, when activated, allows ignoring WFR when starting a dialog.

Node view

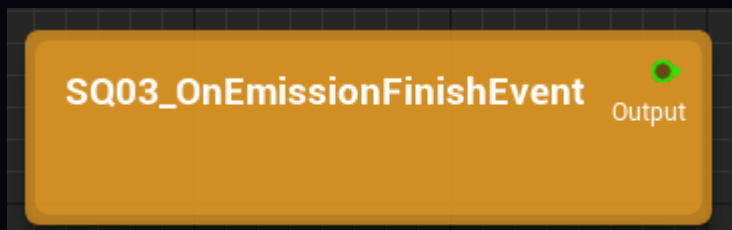


How to use

Parameter	Description
Dialog to Start	Dialog SID that should start
Target	<div>The NPC with whom the dialog should start:</div> <ul style="list-style-type: none">• This is NPC To Start Dialog if the event tracks a forced dialog.• This is the NPC with whom there was interaction if the event tracks the topic selection in Global WFR.• This is the first non-empty (with the selected NPC placeholder) member if the event tracks comments. If there are no speaking NPCs in the comment, for example, everyone is on the walkie-talkie, this field should be left blank.

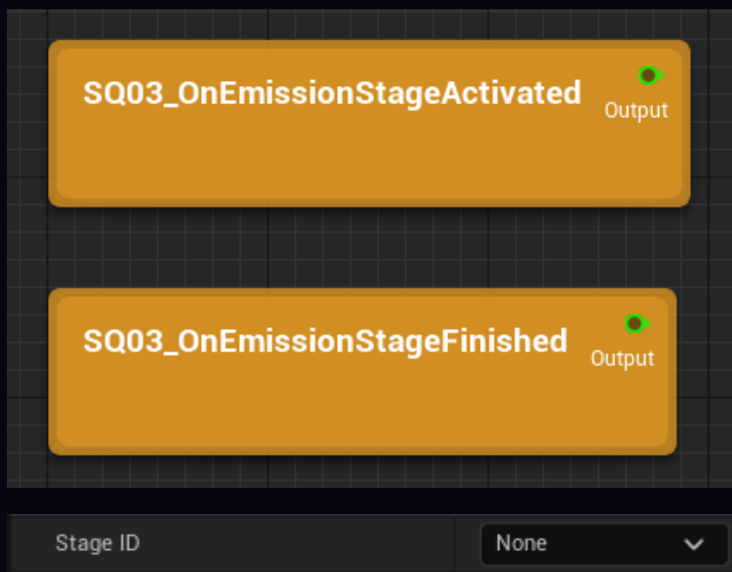
On Emission Finish Event

This quest event node sends a signal after the Emission ends.



On Emission Stage Activated / Finish Event

These quest event nodes trigger at the beginning or end of a specific Emission stage.



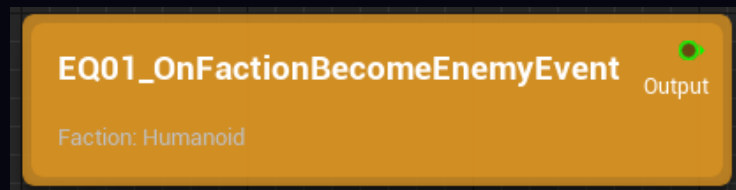
How to use

Parameter	Description
Stage ID	<p>The Emission stage at which the nodes will trigger.</p> <p>Before the Storm – the beginning of the Emission</p> <p>Activate Quest – receiving a quest to find a shelter</p> <p>Shock Wave – the first blast wave of the Emission</p> <p>Active – active stage</p> <p>After The Storm – a short post-storm weather transition</p>

On Faction Become Enemy Event

This event node sends a signal after the selected faction becomes hostile to the player.

References



ON FACTION EVENT EDITOR QUEST NODE		
Target Faction	Humanoid	↩
EVENT EDITOR QUEST NODE		
Track Before Active	☐	

Use case

A narrative designer wants Freedom members on Rostok to praise the player if Duty is hostile to them. To do this, the narrative designer creates the On Faction Become Enemy Event node, selects Duty in Target Faction, and sets the Track Before Active checkbox to **True**.

How to use

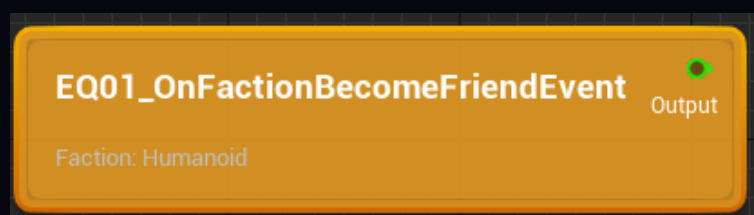
1. Create the node.
2. Select the desired Target Faction.
3. Optionally select the Track Before Active checkbox.
4. Save the changes.

The Track Before Active checkbox determines whether the node treats the event conditions as fulfilled if the faction became hostile before the blueprint in which this node is created was launched.

On Faction Become Friend Event

This event node sends a signal after the selected faction becomes friendly to the player.

References



▼ DEFAULT		
Show Referencing Bridge Nodes		
▼ ON FACTION EVENT EDITOR QUEST NODE		
Target Faction	Humanoid	↩
▼ EVENT EDITOR QUEST NODE		
Track Before Active	☐	

Use case

A narrative designer wants Freedom members on Rostok to insult the player if Duty is friendly towards them. To do this, the narrative designer creates the On Faction Become Friend Event node, selects Duty in Target Faction, and sets the Track Before Active checkbox to **True**.

How to use

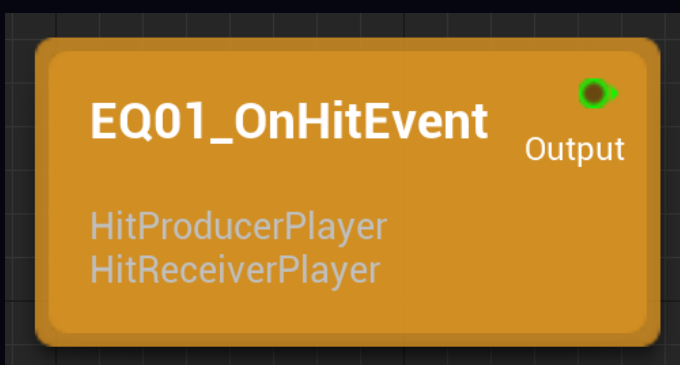
1. Create the node.
2. Select the desired Target Faction.
3. Optionally select the Track Before Active checkbox.
4. Save the changes.

The Track Before Active checkbox determines whether the node treats the event conditions as fulfilled if the faction became hostile before the blueprint in which this node is created was launched.

On Hit Event

This event node sends a signal if the selected character hits another selected character.

References



ON HIT EVENT EDITOR QUEST NODE			
▼ Hit Producer			↩
	Target	None ▼ 🔍 📌	
	Target Player	☐	↩
▼ Hit Receiver			↩
	Target	None ▼ 🔍 📌	
	Target Player	☐	↩

Use case

A narrative designer needs to create a shooting range where the player shoots at rodents. To do this, they use the On Hit Event node.

How to use

1. Create the node.
2. Select Hit Producer target – the character who makes the hit. If the hit is made by the player, set Target Player to **True**.
3. Select Hit Receiver target – the character who receives the hit. If the hit is received by the player, set Target Player to **True**.
4. Save the changes.

On Infotopic Finish Event

This event can be used to activate quest sequences tied to the player hearing a specific infotopic.

References

E01_MQ01_OnInfoTopicFinishEvent_General_Neutrals_Infotopic_5
●

Output

CHARACTER TARGETED ACTION			
Dialog to Set	None	None ▼ 🔍	

Use case

A narrative designer wants the encounter to become available only after the player has listened to a specific infotopic. They set up a node that activates the encounter after the player has talked to the NPC and received the corresponding topic.

A narrative designer wants the player to receive a location marker after learning about it in the infotopic. They set up a node and open the location marker after the player listens to the corresponding topic.

How to use

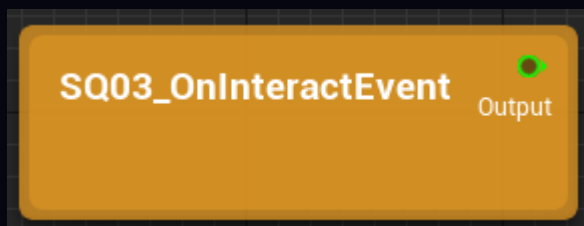
This node can be used with any dialogs, but please note that this node cannot be used to check what the output pin of the dialogs was - it cannot be used as a replacement for bridges.

To activate encounters, also add a distance condition: the player can receive many infotopics from A-Life actors, which means that they can be very close to the location of the encounter.

On Interact Event

This quest event node sends a signal after the player interacts with the selected Quest Object. It does not work with Interactable Objects.

Node view



▼

On Interacted Event Quest Node

Interact Event

On Start Interact ▼

▼

Interactable

Interactable Actor

None ▼

🔗

✎

Interactable Object Guid

{00000000-0000-0000-0000-000000000000}

How to use

Parameter	Description
Interact Event	The stage of interaction at which the node will trigger (start or finish)
Interactable Actor	The actor with whom the interaction occurs. It can be an NPC, squad, container, item, pack of items. Does not work with Interactable Objects.

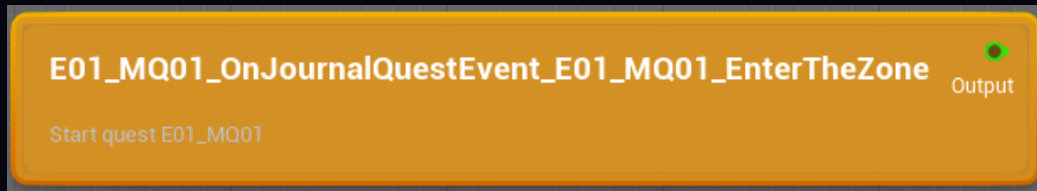
IMPORTANT

The node should not trigger on Talk Through Radio dialogs.

On Journal Quest Event

This event allows tracking any journal events: the start and end of a quest or quest stage, a change in description, etc.

References



ON JOURNAL QUEST EVENT EDITOR QUEST NODE		
Entity	Quest Stage	
Action	Start	
Journal Quest	E01_MQ01	↩
Quest Stage SID	E01_MQ01_EnterTheZone	↩
EVENT EDITOR QUEST NODE		
Track Before Active	☐	

Use case

A narrative designer wants to check that a certain stage of another quest has already been completed in order to add a new topic to the NPC. To do this, they use this event with the Track Before Active setting, which allows them to check that the event's trigger conditions have been met before the quest blueprint itself starts.

A narrative designer wants to activate the quest only if another quest has failed. They use this event (again, with the Track Before Active setting) to activate the desired quest blueprint.

How to use

After creating the node, you can choose one of two settings: Entity - Quest or Quest Stage.

ON JOURNAL QUEST EVENT EDITOR QUEST NODE		
Entity	Quest Stage	
Action	Quest	
	Quest Stage	
Journal Quest	E01_MQ01	↩
Quest Stage SID	E01_MQ01_EnterTheZone	↩

The Quest setting allows checking the events that occurred with the quest by selecting the appropriate Action setting:

- Start – receiving a quest
- Finish – completing a quest
- Fail – failing a quest

- Cancel – canceling a quest
- Change Description – changing the quest description

▼ ON JOURNAL QUEST EVENT EDITOR QUEST NODE

Entity	Quest	↩
Action	Start	
Journal Quest	Start	↩
Quest Stage SID	Fail	
▼ EVENT EDITOR QUEST NODE	Cancel	
Track Before Active	☐	
▼ OUTPUT PINS		

The screenshot displays the 'Quests' tab in the game editor. On the left, a scrollable list contains numerous quest entries, including 'empty', 'DKTO_2', 'E01_MQ01' (highlighted in blue), 'E02_MQ01', 'E02_MQ02', 'E02_MQ02_C01', 'E02_MQ02_C02', 'E02_MQ02_C03', 'E02_MQ03', 'E02_MQ03_C01', 'E02_MQ03_C02', 'E02_SQ01_C01', 'E02_SQ01_C02', 'E02_SQ01_C03', 'E02_SQ01', 'E03_MQ01', 'E03_MQ02', 'E03_MQ03_Svara', 'E03_MQ03_Voentorg', 'E03_MQ03_Varan', 'E03_MQ04', 'E03_MQ05', 'E03_MQ06', 'E03_SQ01_C1', 'E04_MQ01', 'E04_MQ02', 'E04_MQ03', 'E04_MQ04', 'E04_MQ05', 'E04_MQ06', 'E05_MQ01', 'E05_MQ02', 'E05_MQ03', 'E05_MQ04', 'E06_MQ01', 'E06_MQ02_C01', 'E06_MQ02_C01_E02', 'E06_MQ02_C01_E03', 'E06_MQ02_C02', 'E06_MQ02', 'E06_MQ03', 'E06_MQ04', 'E07_MQ01', 'E07_MQ02', 'E07_MQ02_C01', 'E07_MQ03', 'E08_MQ01', 'E08_MQ02', 'E08_MQ03', 'E08_MQ04', 'E08_MQ05', 'E09_MQ01', 'E09_MQ02', 'E09_MQ03', 'E09_MQ04', 'E09_MQ05', 'E10_MQ01', 'E10_MQ02', 'E10_MQ03', and 'E11_MQ01'.

The right panel provides a detailed view of the selected quest, 'E01_MQ01'. It includes a 'Quest' dropdown menu set to 'E01_MQ01', a 'Start' dropdown menu set to 'E01_MQ01_EnterTheZone', and an 'Output' dropdown menu set to 'OnJournalQuestEvent_E01_MQ01_EnterTheZone'. A yellow bar is visible at the bottom of the quest details panel.

You can perform the same checks with a specific quest stage. To do this, select Quest Stage in the entity and additionally specify the required stage in the Quest Stage SID setting.

▼
DEFAULT

Show Referencing Bridge Nodes

▼
ON JOURNAL QUEST EVENT EDITOR QUEST NODE

Entity	<div> Quest Stage ▼ </div>	
Action	<div> Start ▼ </div>	
Journal Quest	<div> E01_MQ01 ▼ </div>	↶
Quest Stage SID	<div> E01_MQ01_EnterTheZone ▼ </div>	↶
▼ EVENT EDITOR QUEST NODE	<div> E01_MQ01_EnterTheZone </div>	
Track Before Active	<div> E01_MQ01_GoToThePowerLine </div>	
▼ OUTPUT PINS	<div> E01_MQ01_GoToTheBridge </div>	
▼ Output Pin	<div> E01_MQ01_PutTheScanner </div>	↶
Name	<div> E01_MQ01_Hide </div>	↶
▼ REPEATABLE	<div> E01_MQ01_TakeTheScanner </div>	
Repeatable	<div> E01_MQ01_PowerLineTakeScanner </div>	
▼ NODE SETTINGS	<div> E01_MQ01_PowerLineHide </div>	
Title	<div> E01_MQ01_Bunker </div>	
► Border Color	<div> E01_MQ01_CaveScanningPoint </div>	
	<div> E01_MQ01_RestorePower </div>	
	<div> E01_MQ01_Wait </div>	

On Killer Check Event

This event node sends a signal if a specific character or squad kills the selected character or squad.

References

EQ01_OnKillerCheckEvent_not_set

Output

Killer: Player
ShouldBeKilled: not_set!

▼
TARGETED EVENT

▼ Should be Killed

Target	<div> None ▼ </div>	<div> Q </div>	<div> </div>
Full Squad	<div> </div>		
▼ Target NPCPlaceholder			
Target	<div> None ▼ </div>	<div> Q </div>	<div> </div>
Target Player	<div> ☒ </div>		

Use case

A designer needs to script Safari. To do this, they need to make sure that it is the NPC that kills the boars. To do this, the narrative designer uses the On Killer Check Event node. After a specific NPC kills the boars, the node sends a signal further.

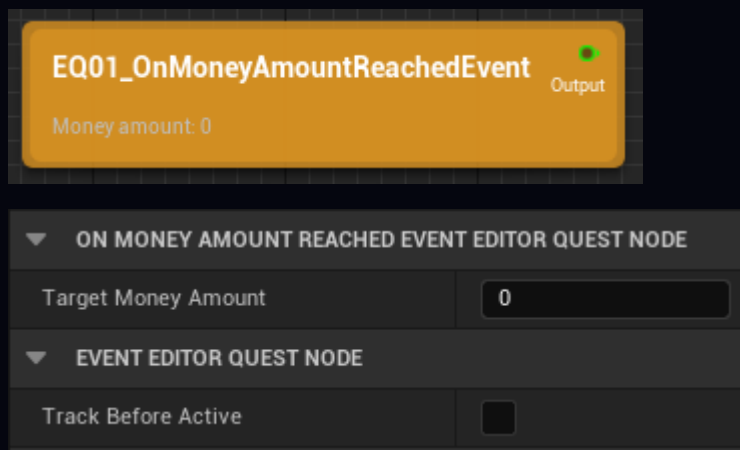
How to use

1. Create the node.
2. Select the target to be killed in the Target field in Should be Killed.
3. In Target NPC Placeholder, select the killer.
4. Save the changes.

On Money Amount Reached Event

This event node sends a signal after the player has accumulated a certain amount of game currency.

References



Use case

A narrative designer decided to allow the player to purchase SV Gvintar from a merchant after the player had accumulated 73 coupons. The narrative designer creates an On Money Amount Reached Event node, enters 73 in the Target Money Amount field, and sets the Track Before Active checkbox to **True**. After the player has accumulated 73 coupons, the node will send the signal further.

How to use

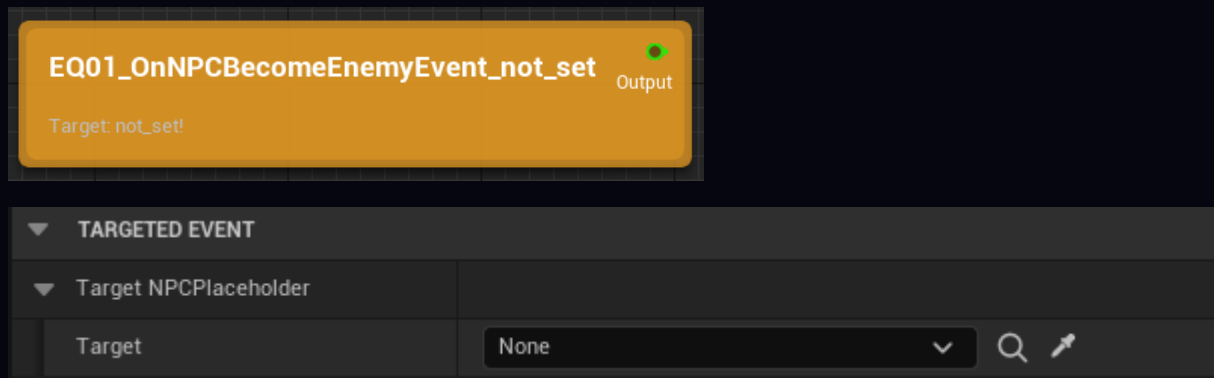
1. Create the node.
2. Enter the required amount of money in Target Money Amount.
3. Configure Track Before Active according to the design.
4. Save the changes.

The Track Before Active checkbox determines whether the node will treat the event conditions to be fulfilled if the amount of money was accumulated before the blueprint in which this node is created was launched.

On NPC Become Enemy Event

This event node sends a signal after the selected NPC becomes hostile to the player.

References



Use case

A narrative designer wants the quest to be canceled if the player starts shooting at the quest giver. To do this, the narrative designer uses the On NPC Become Enemy Event node.

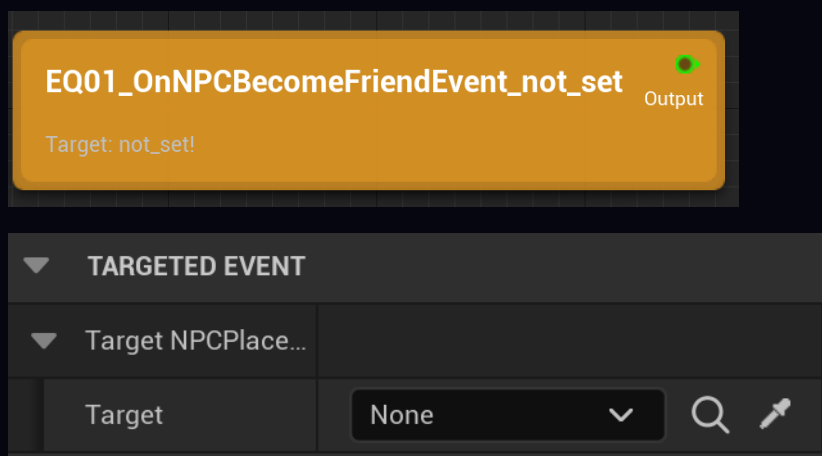
How to use

1. Create the node.
2. In the target field, select the NPC with which the node will interact.
3. Save the changes.

On NPC Become Friend Event

This event node sends a signal after the relationship between the selected NPC and the player reaches friendly status.

References



Use case

A narrative designer wants the stalker to give the player a quest if he has not previously conflicted with the neutral faction and, for example, has given this stalker a first aid kit or performed another action that increases his reputation with a specific NPC. To do this, they use the On NPC Become Friend Event node.

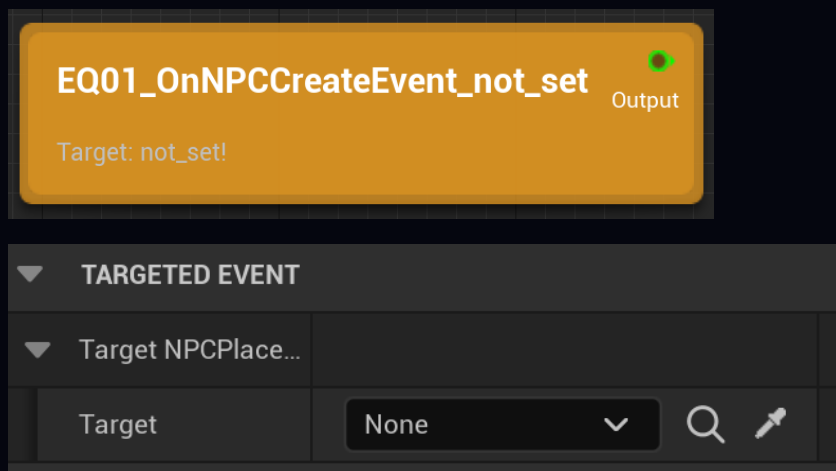
How to use

1. Create the node.
2. Select the desired NPC in Target.
3. Save the changes.

On NPC Create Event

This event node sends a signal when the selected NPC spawns on the map.

References



Use case

A narrative designer needs to launch a cutscene with Korshunov, but first, they need to make sure that Korshunov exists on the World Map. In this case, it is best to use the On NPC Create Event node.

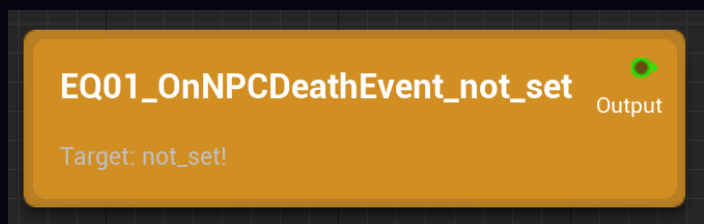
How to use

1. Create the node.
2. Select the desired NPC in the Target field.
3. Save the changes.

On NPC Death Event

This event node sends a signal after the selected NPC dies.

References



▼ TARGETED EVENT	
▼ Target NPCPlace...	
Target	<div>None ▼</div> <div>🔍 📌</div>

Use case

The player needs to change the NPC's infotopic if their partner dies. To do this, you use the On NPC Death Event node.

How to use

1. Create the node.
2. Select the desired NPC in the Target field.
3. Save the changes.

On NPC Defeat Event

This event node sends a signal after the selected NPC enters Defeat State (an NPC falls and curls up on the ground).

References

EQ01_OnNPCDefeatEvent_not_set

Output

Target: not_set!

▼ TARGETED EVENT	
▼ Target NPCPlace...	
Target	<div>None ▼</div> <div>🔍 📌</div>

Use case

A narrative designer needs to start a final dialog with a stalker who has been wounded by the player. To do this, they use the On NPC Defeat Event node.

How to use

1. Create the node.
2. Select the desired NPC in the Target field.
3. Save the changes.

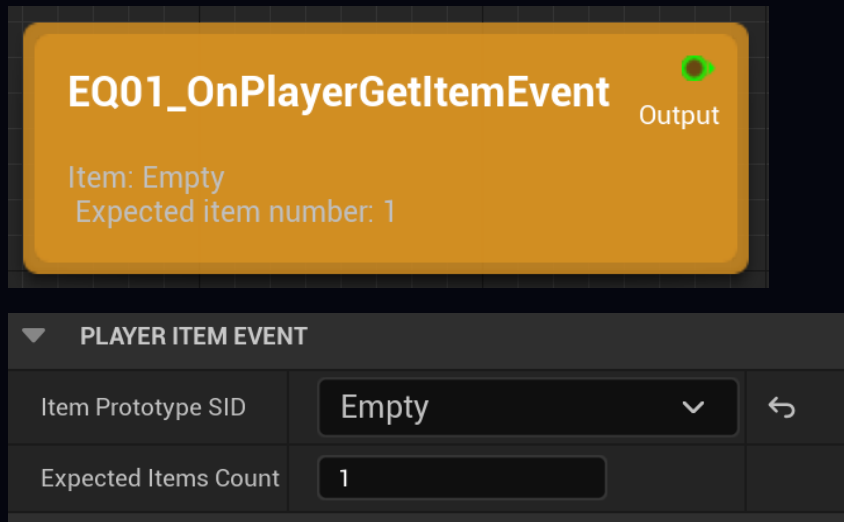
On Player Get Item Event

This event node triggers a signal if the selected item appears in the player's inventory in the selected quantity.

In the node, you can configure its activation even if the item does not go to the inventory, but to the Equip Slot.

The node only works on items visible in the inventory (UI).

References



Use case

A narrative designer wants to make the NPC react to the presence of a Gauss gun in the player's inventory. To do this, they use the On Player Get Item Event node.

How to use

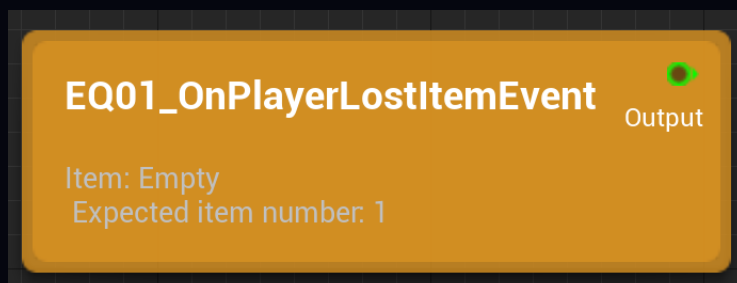
1. Create the node.
2. Select the desired item in Item Prototype SID.
3. Select the desired number of copies of this item in **Expected Items Count**.
4. Save the changes.

On Player Lost Item Event

This event node triggers a signal if the selected item disappears from the player's inventory.

The node only works on items visible in the inventory (UI).

References



▼ PLAYER ITEM EVENT		
Full Amount	☐	
Item Prototype SID	Empty ▼	↩
Expected Items Count	1	

Use case

A narrative designer needs Skif to react to the fact that he has run out of vodka. To do this, the designer uses the On Player Lost Item Event node.

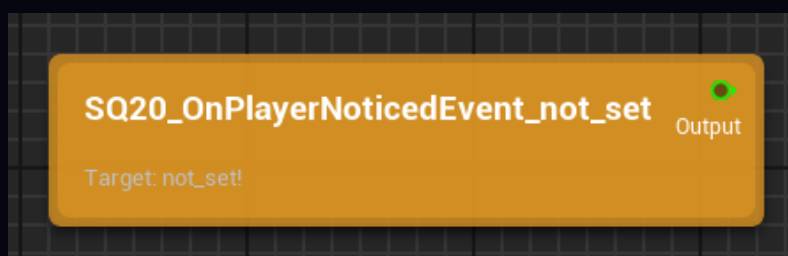
How to use

1. Create the node.
2. Select the desired item in Item Prototype SID.
3. Set the desired number of units of this item:
 - If you need to check for the loss of all copies of the item, set the Full Amount checkbox to **True**.
 - If you need a specific number of copies, manually enter it in Expected Items Count.
4. Save the changes.

The full amount allows the node to work if the player has lost all copies of the selected item.

On Player Noticed Event

This quest event node sends a signal after the selected NPC sees and recognizes the player as an enemy, or sees them with their eyes.



▼ Default	
Show Referencing Bridge Nodes	
▼ Targeted Event	
▼ Target NPCPlaceholder	
Target	None ▼ [Icons]

How to use

Parameter	Description
-----------	-------------

Target NPC Placeholder	NPC or squad that must notice the player for the node to trigger
------------------------	--

On Player Rank Reached Event

This quest event node sends a signal after the player reaches a certain rank.



▼ On Player Rank Reached Event Editor Quest Node

Target Rank

Newbie ▼

▼ Track Before Active

Track Before Active

☐

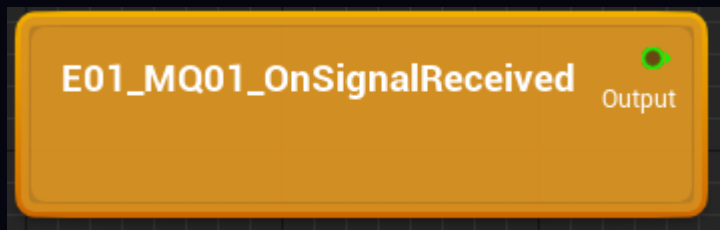
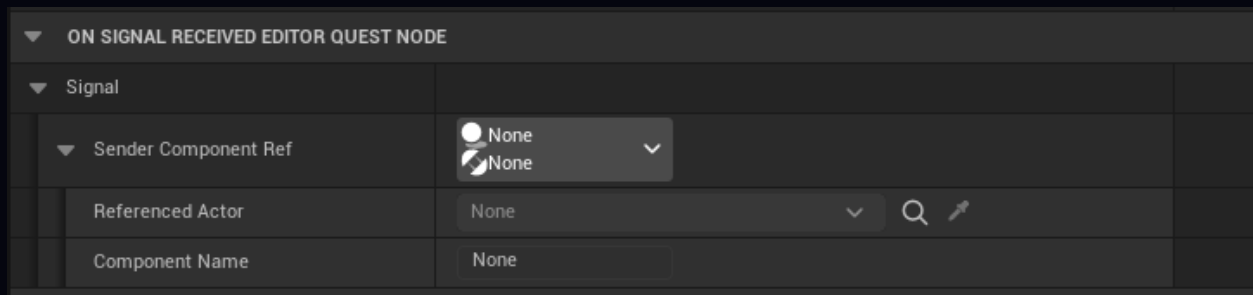
How to use

Parameter	Description
Target Rank	The rank at which the node sends a signal. Ranks: Newbie, Experienced, Veteran, Master
Track Before Active	The checkbox that activates the node from the start of the game, rather than from the start of the quest blueprint. Using this parameter is almost always mandatory with this node, because the player's rank may change during an active quest.

On Signal Received

This node allows receiving information from interactive objects in Quest Tool.

References



Use case

A narrative designer wants the NPC to play a comment after a certain interactive object is triggered. To do this, they specify the interactive object and component in the node, and then in the game, when this interactive component sends a signal, the node will be triggered.

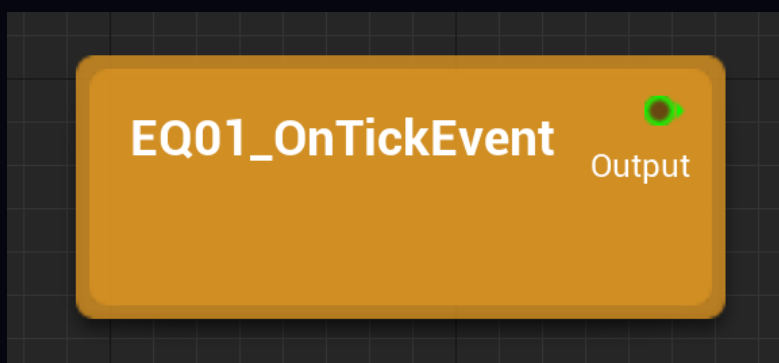
How to use

1. Select the button in the **Sender Component Ref** field.
2. Find the interactive object in the list – the node will “listen” to its Signal Sender.
3. Select the component.
4. Save the changes.

On Tick Event

This node sends a signal on the first tick after the quest blueprint is launched, after which it ends immediately, sending the signal further.

References



Use case

A designer needs to connect any node that does not depend on the quest flow. On Tick Event is suitable for this.

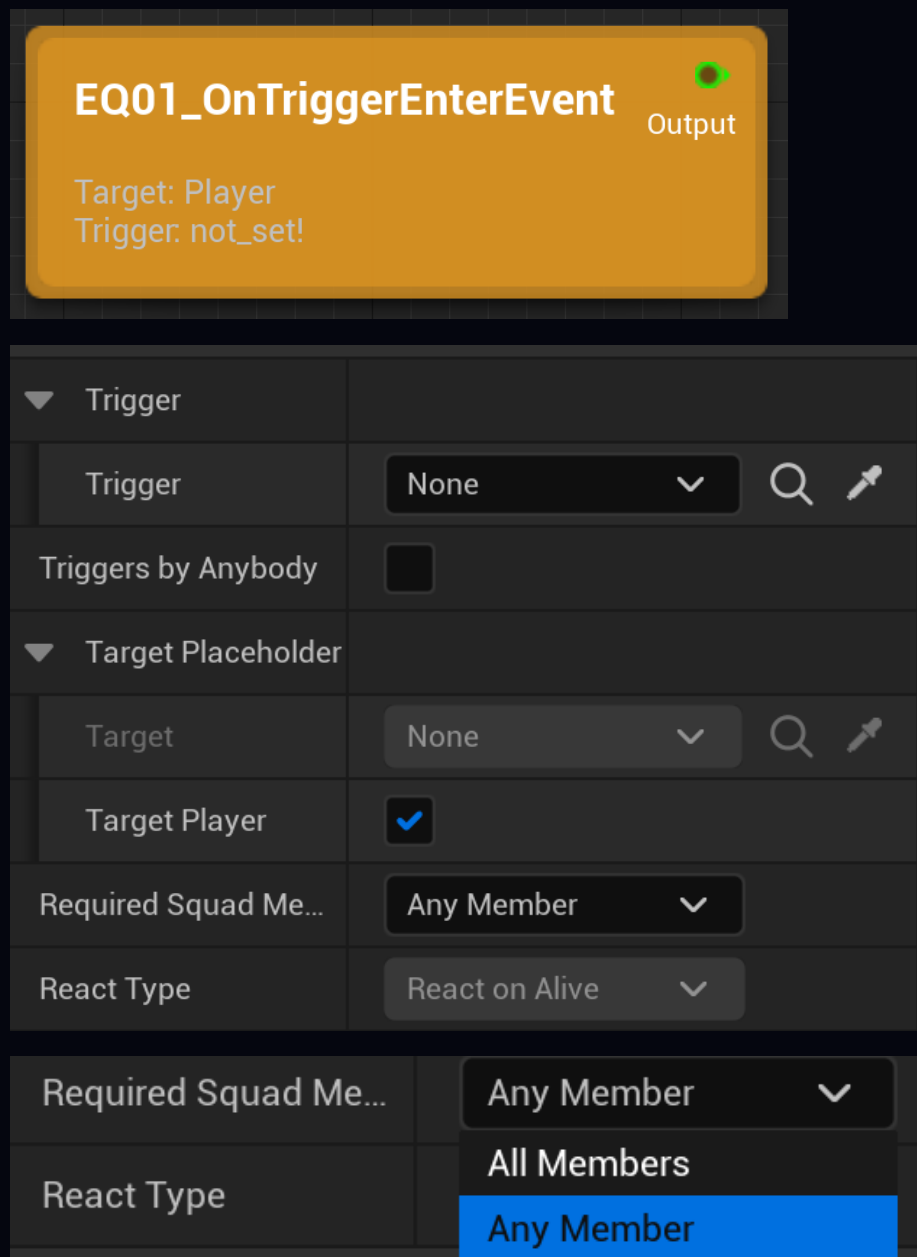
How to use

1. Create the node.
2. Save the changes.

On Trigger Enter Event

This event node sends a signal if the selected character enters the selected trigger.

References



The image shows the 'EQ01_OnTriggerEnterEvent' node in a game engine. The node is orange and has a green output pin. Below the node name, it shows 'Target: Player' and 'Trigger: not_set!'. Below the node is a configuration panel with the following settings:

▼ Trigger	
Trigger	None ▼ 🔍 🎯
Triggers by Anybody	☐
▼ Target Placeholder	
Target	None ▼ 🔍 🎯
Target Player	☒
Required Squad Me...	Any Member ▼
React Type	React on Alive ▼

Below the configuration panel, there is a dropdown menu for 'Required Squad Me...' with the following options:

- Any Member
- All Members
- Any Member (highlighted)

Use case

1. A designer wants bloodsuckers to attack the player when they enter the trigger. In this case, use On Trigger Enter Event.
2. A designer wants the NPC to make a comment when entering the trigger. In this case, use On Trigger Enter Event.

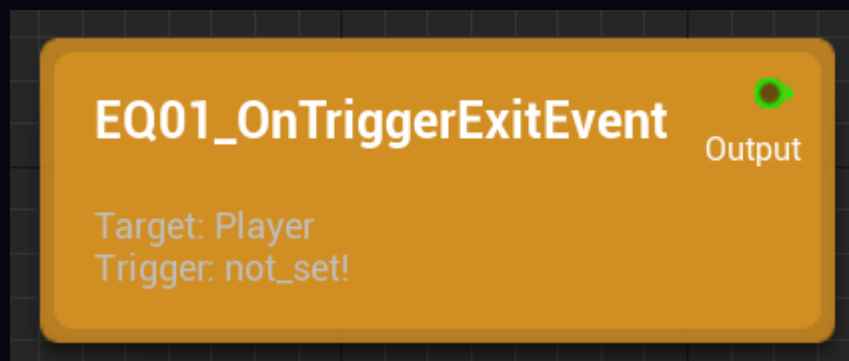
How to use

1. Create the node.
2. In the **Trigger** field, set the required trigger placeholder.
 - a. The trigger will only work if it completely covers the player or NPC capsule in height.
3. In the **Target Placeholder** field, configure the character on which the node will work.
 - a. If the target is a player, select the **Target Player** checkbox.
 - b. If you want the node to activate when any character overlaps with the trigger, select the **Triggers by Anybody** checkbox.
 - c. If the target is a squad, configure how the node will work: when the first squad member enters the trigger, or when all squad members are in the trigger.
4. In **React Type**, select whether the node will work if the character entering the trigger is dead.

On Trigger Exit Event

This event node sends a signal if the selected character leaves the volume of the selected trigger.

References



▼ Trigger	
Trigger	None ▼ 🔍 📌
Triggers by Anybody	☐
▼ Target Placeholder	
Target	None ▼ 🔍 📌
Target Player	☒
Required Squad Me...	Any Member ▼
React Type	React on Alive ▼

Use case

1. A narrative designer wants the player to be shot at if they leave Korshunov's office.
2. A narrative designer wants the player's journal to be updated when the selected NPC leaves the temporary camp area.

How to use

1. Create the node.
2. In the **Trigger** field, set the required trigger placeholder.
 - The trigger will only work if it completely covers the player or NPC capsule in height.
3. In the **Target Placeholder** field, configure the character on which the node will work.
 - If the target is a player, select the **Target Player** checkbox.
 - If you want the node to activate when any character leaves the trigger, select the **Triggers by Anybody** checkbox.
 - If the target is a squad, configure how the node will work: when the first squad member leaves the trigger, or when all squad members leave it.
4. In **React Type**, select whether the node will work if the character leaving the trigger is dead.

On Upgrade Install

This event node sends a signal after the player installs a specific upgrade at the technician's.

How to use

Parameter	Description
-----------	-------------

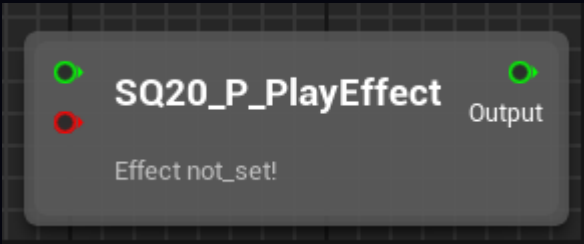
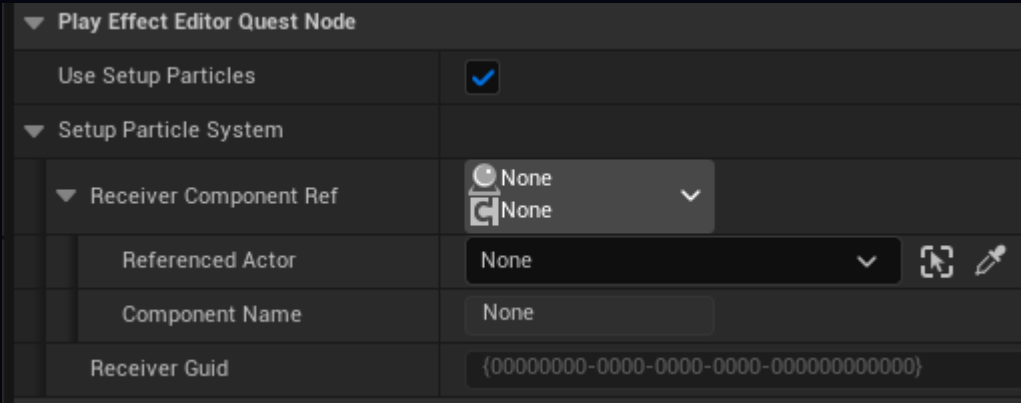
Upgrade sid	An upgrade that activates the node when installed. If you do not select a SID, the node will activate when any upgrade is installed.
-------------	---

Play Effect

This node activates the selected Niagara System Actor (VFX).

The node will end immediately after sending a signal to the selected actor.

Node view



How to use

Parameter	Description
Use Setup Particles	The checkbox responsible for which particle will be played and how. Using the node without this checkbox is prohibited!
Setup Particle System	BP_QuestVFX actor with Niagara component with effects configured

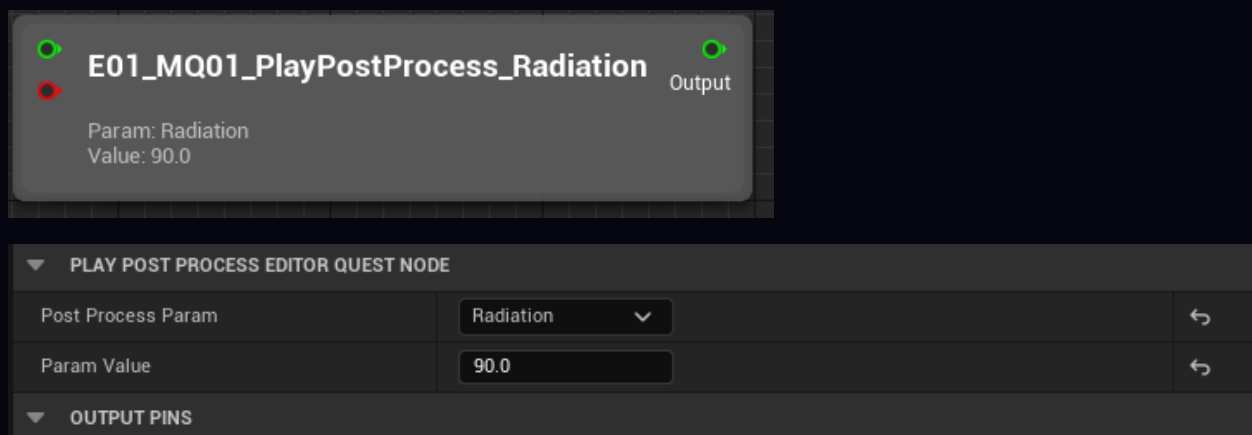
How to set a VFX actor for use with the node

1. Place the actor (blueprint s2gamebuild/Plugins/GameDataEditor/Content/BP_QuestVFX.uasset) on the map. By default, it has a Niagara System component with the auto-activate set to **False**. Make sure to configure effects in this actor.
2. In the node, select the BP_QuestVFX actor as the target. Activating the node activates the Niagara component attached to it, while node exclusion deactivates it.

Play Post Process

This node allows applying post-processes from a predefined list.

References



Use case

A narrative designer wants to simulate the effect of the PSY effect on the player for dramatic effect without the actual PSY effect. To do this, they apply a post-process to the player using this node.

How to use

After creating the node, select a specific Post Process in the settings.

A simple way to calculate the optimal Param Value range is by trial and error, and it usually varies between 10 and 100.

A more complex method is to find the post-process material of the same name in the Content browser and see what parameters and values it accepts.

Play Sound

This quest node is used by sound and narrative designers in quests to play sound events.

The node ends immediately after the event is triggered.

Node exclusion will not cancel its effect or stop the sound event.

How to use

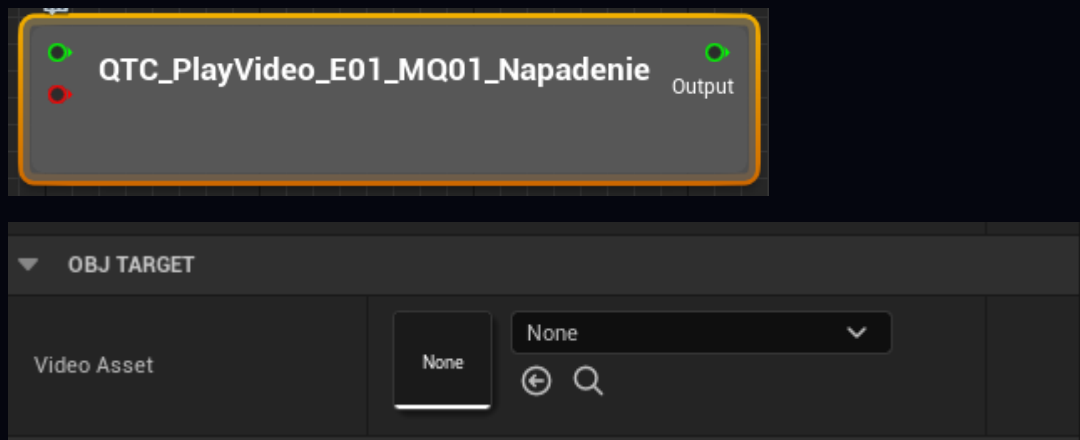
Parameter	Description
Sound Event	The sound event that will be played by the node
MasterAkEvent	The event triggered after a save load. Only after this event is triggered will the base event be activated. If a master event is not selected, the quest logic will work as usual during a save load, and the base event will be activated.

Play Video

This node allows launching cutscene drafts and other videos.

It uses the default Unreal media player and the .mp4 format.

References



Use case

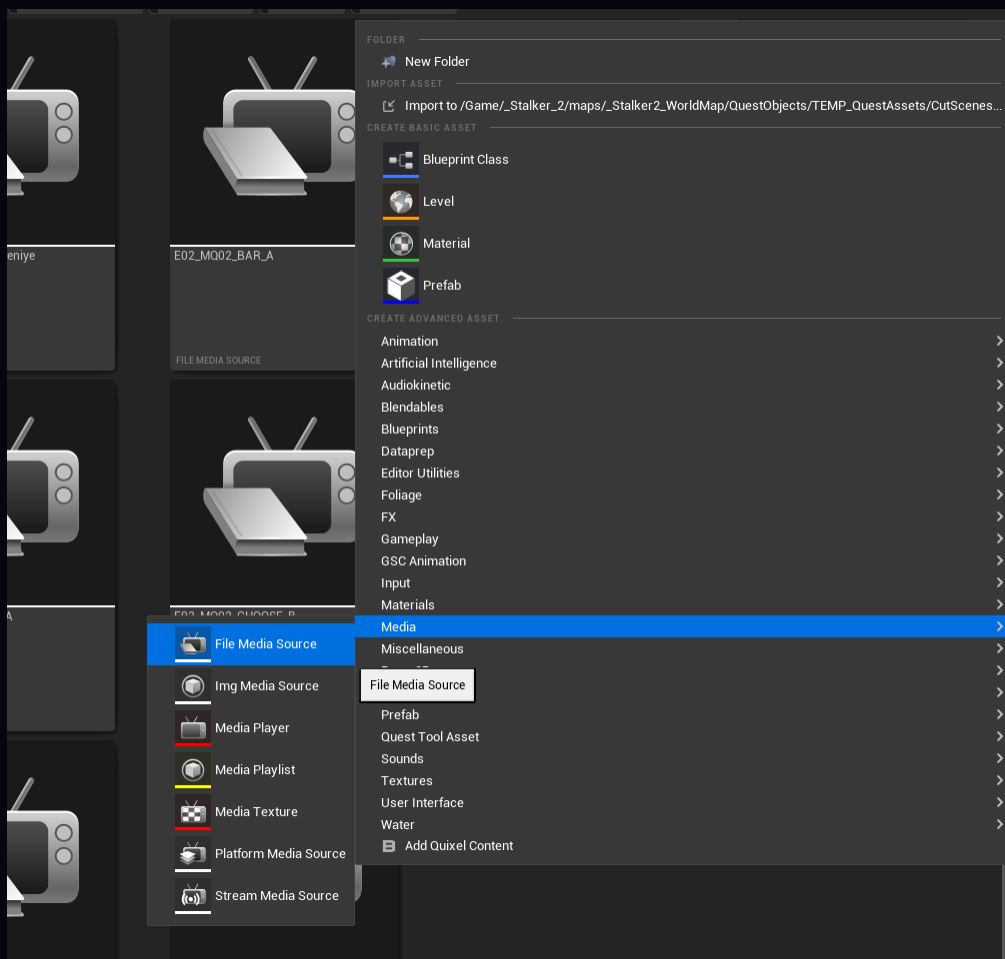
A narrative designer wants to use a draft video with a mockup of a cutscene. They configure this node and run it so that the cutscene can be seen in the game and better understand how it works in the context of the quest.

How to use

Assets that have already been created and can be added to the node are located here:

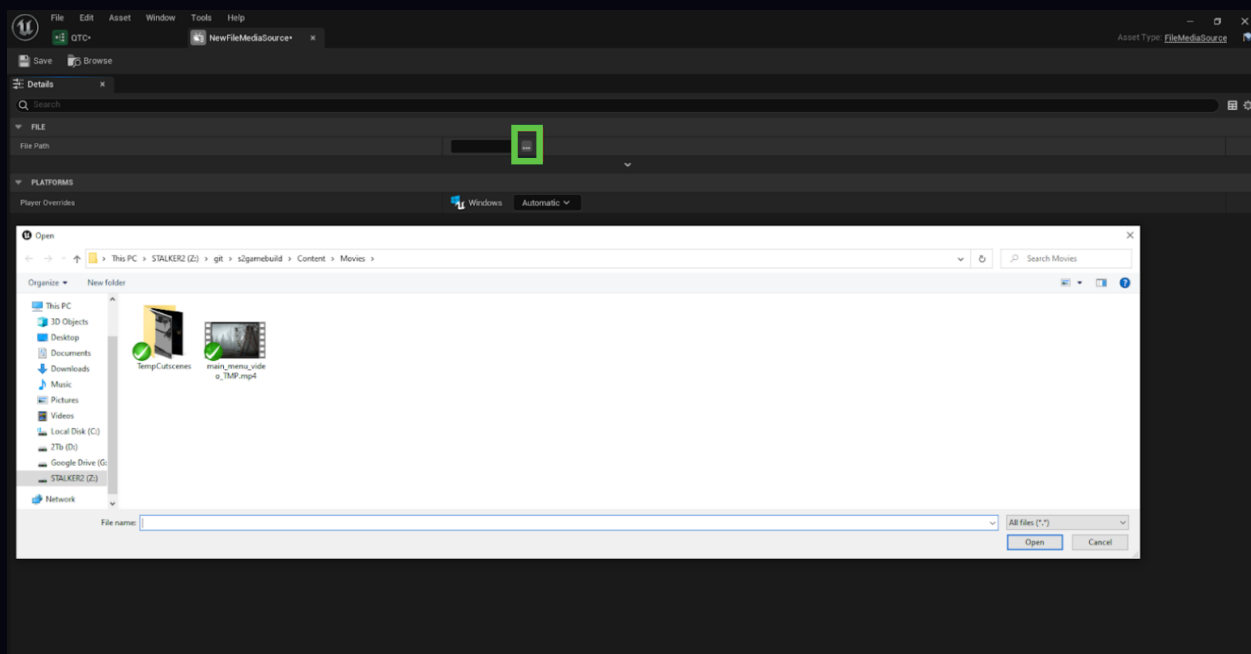
Z:/git/s2gamebuild/Content/_Stalker_2/maps/_Stalker2_WorldMap/QuestObjects/TEMP_QuestAssets/CutScenes/

1. To create a new asset, place the .mp4 file in the *Z:/git/s2gamebuild/Content/Movies/TempCutscenes* folder.
2. Go to *Z:/git/s2gamebuild/Content/_Stalker_2/maps/_Stalker2_WorldMap/QuestObjects/TEMP_QuestAssets/CutScenes/* and right-click there to create an empty File Media Source asset.



3. Name the new asset the same as the .mp4 file and save it.

4. Open the newly created asset and set the File Path.



5. Save changes again.

6. Select a file in the node. Do not forget to commit and push the .mp4 file. Otherwise, other developers will not be able to use this cutscene!

Protect Placeholder

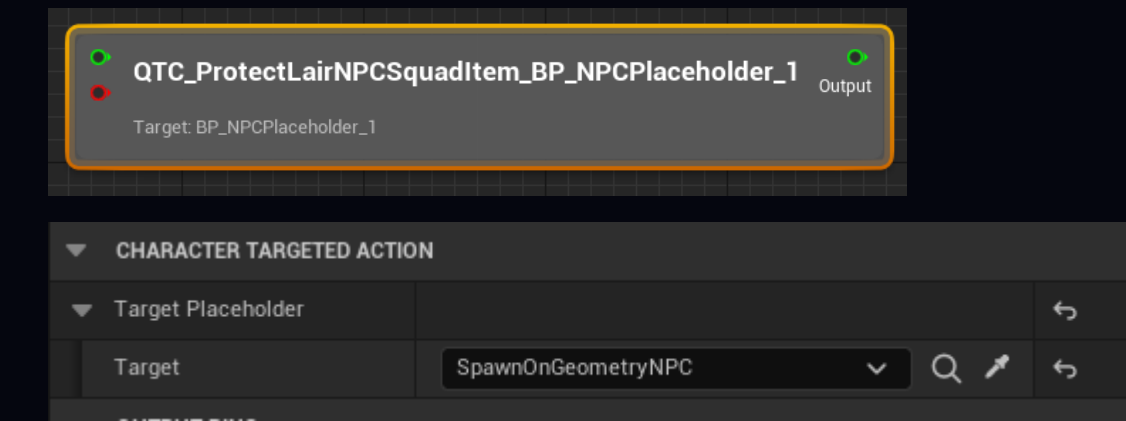
This node helps narrative designers protect the quest NPC from A-Life.

The node gives the selected target immunity to damage caused by A-Life agents. This means that combat between A-Life and quest agents can occur, but quest agents have a clear cheating advantage in this battle.

The node also gives NPCs immunity to the controller's zombification.

Similar protection can also be built into the NPC config.

References



How to use

Parameter	Description
Target	The NPC, squad, or lair protected by the node. Use the node carefully and consider its necessity. An NPC protected from A-Life can cut out the entire zone if scripted incorrectly.

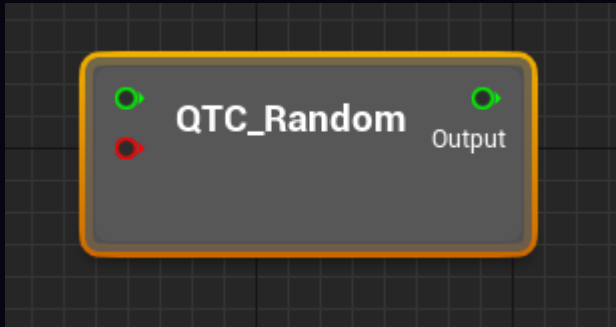
IMPORTANT

Once a node is launched, it remains active until Exclude is executed for it. Therefore, it is very important to configure the node's exclusion conditions and not forget to set the **Exclude all nodes in Container** in the End node settings.

Random

This node selects its random output pin and sends a signal through it. It allows adding randomized sequences to quests.

References



RANDOM EDITOR QUEST NODE		
▼ Pin Weights	1 Array elements	
Index [0]	<input type="text" value="0.0"/>	
▼ OUTPUT PINS		

Use case

A narrative designer wants to randomize the number of enemies that the player will meet in an encounter. To do this, they add the Random node that spawns either a squad of three enemies or a squad of five, depending on how the signal goes.

How to use

In the node settings, add several output pins depending on how many options the node has to choose from.

Weight settings are automatically added for each pin.

RANDOM EDITOR QUEST NODE			
▼ Pin Weights		3 Array elements	
	Index [0]	0.0	
	Index [1]	0.0	↩
	Index [2]	0.0	↩
OUTPUT PINS			
▼ Output Pins		3 Array elements	⊕ 🗑 ↩
	▼ Index [0]	Output	↩
	Name	Output	↩
	▶ Index [1]	None	↩
	▶ Index [2]	None	↩
INPUT PINS			
▼ Input Pins		3 Array elements	⊕ 🗑 ↩

Pins and their weights are correlated by index. For example, for a pin with index [0], the weight is set in Index [0].

RANDOM EDITOR QUEST NODE			
▼ Pin Weights		3 Array elements	
	Index [0]	30.0	↩
	Index [1]	0.0	↩
	Index [2]	0.0	↩
OUTPUT PINS			
▼ Output Pins		3 Array elements	⊕ 🗑 ↩
	▼ Index [0]	Output	↩
	Name	Output	↩
	▶ Index [1]	None	↩
	▶ Index [2]	None	↩

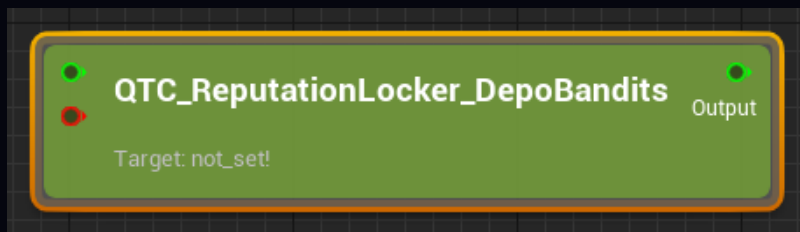
You can set the weight to any positive number above 0. The higher the number relative to the numbers set for other pins, the higher the chance that the signal will go through this pin.

A weight of 0 is synonymous with a disabled pin, which can be used for debugging.

Reputation Locker

This node fixes the level with a specific NPC/squad or faction, preventing systems or player actions from changing it.

References



REPUTATION LOCKER EDITOR QUEST NODE		
Target Faction	☒	↩
Target Faction	DepoBandits ▼	↩
Minimal Reputation	Neutral ▼	
CHARACTER TARGETED ACTION		
Target Placeholder		
Target	None ▼ 🔍 ✎	

Use case

A narrative designer wants to create an encounter where bandits talk to the player. To do this, they change their reputation to neutral and fix it using local reputation. Thus, they can be sure that the player will not be able to influence the reputation with the bandits by interacting with the systems, and the encounter will not be ruined.

How to use

Reputation can be fixed for both an entire faction and an individual character. In the case of a character, select their NPC placeholder or squad placeholder as the target. In the case of a faction, select the **Target Faction** checkbox and select the desired faction from the drop-down list.

There are four types of relationships:

- Friend
- Neutral
- Disaffection
- Enemy

The node does not end after starting, and its effect remains active until the node is excluded. Please keep this in mind when scripting.

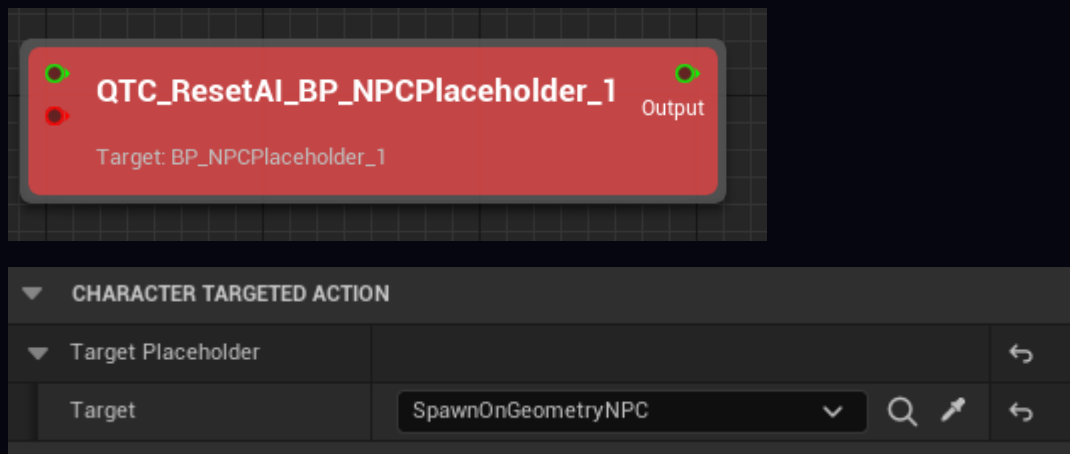
NOTE

The faction reputation fixed using this node will remain unchanged throughout the game. Use this node with extreme caution with the main factions: Varta, Spark, Neutrals, and Bandits.

Reset AI

This node resets the quest goals of NPCs or squads, turning them into A-Life actors.

References



Use case

After completing a quest, a narrative designer wants the NPCs who participated in that quest to go into A-Life and become actors like the other NPCs in the game. To do this, they reset the NPC quest goals before completing the blueprint.

How to use

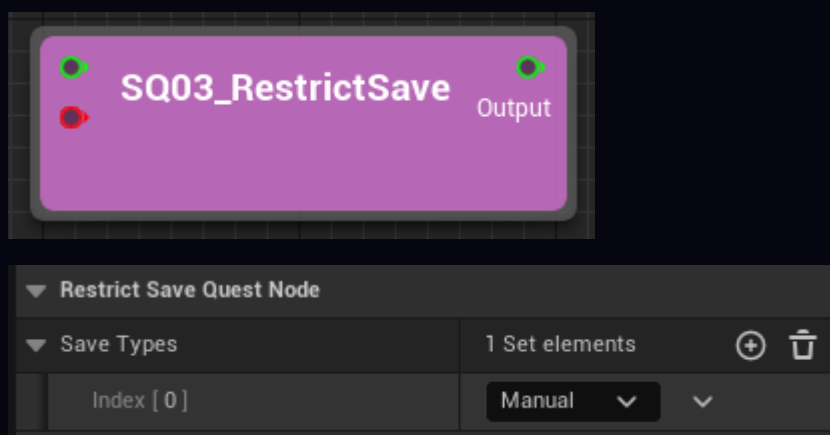
Resetting goals is a mandatory action to be performed for NPCs after completing an encounter or quest, except for the cases when you want them to disappear from the map – in this case, they must be despawned.

To do this, create the Reset AI node and select the desired placeholder – NPC or squad – in the **Target** field. After the node is completed, the goals will reset, and the character will start acting according to the A-Life rules.

Restrict Save

This node prohibits saving the game. While the node is active, saving of the selected type is prohibited.

The node remains active until it is manually excluded.



How to use

Parameter	Description
Save Types	The number and types of prohibited saves: Manual – manual saving via the menu Auto – auto-saving Quick – quick saving via a hotkey Quest – saving via QBP

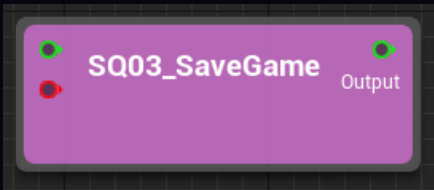
IMPORTANT

Make sure to configure the conditions of the node exclusion so that the save prohibition does not break the game for the player.

Save Game

- This node saves the game.
- The node ends immediately after the save is created.
- The node exclusion will not cancel its effect.

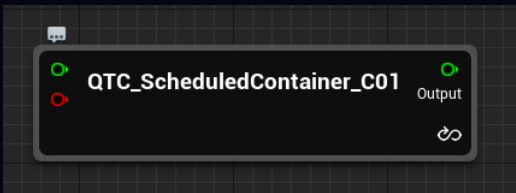
Node view



Scheduled Container

A container that runs for a specific time interval and then excludes itself, restarting when the next time condition begins to be fulfilled. Otherwise, it is identical to a standard Container node.

References



▼ DEFAULT

Generate Output Pins

Show Referencing Bridge Nodes

▼ SCHEDULED TIME

Time Interval

Day ▼

From Hour

0

To Hour

1

▼ SCHEDULED DAY

Monday	☒
Tuesday	☒
Wednesday	☒
Thursday	☒
Friday	☒
Saturday	☒
Sunday	☒

▼ CONTAINER QUEST NODE

Quest Blueprint

None

None ▼

Use case

A narrative designer wants to create an encounter that only runs at night. To do this, they configure a Scheduled Container in which they set the container activation time to nighttime, after which the encounter will only work during this time period.

How to use

As with regular containers, create a quest blueprint and specify it in the node settings. Then, in the settings, you can select a time preset or set a specific time interval in hours.

Presets:

Morning: 06:01 - 12:00

Day: 12:01 - 18:00

Evening: 18:01 - 00:00

Night: 00:01 - 06:00

If you need to set a specific time interval, use the Custom setting and set the interval from hour to hour in the format from 0 (midnight) to 23 (11 p.m.).

Send Signal

The node allows sending a signal to an interactive object component.

References



SEND SIGNAL EDITOR QUEST NODE		
Signal		
Receiver Component Ref	<div><div>None</div><div>None</div></div>	
Referenced Actor	<div>None</div>	<div><div></div><div></div></div>
Component Name	<div>None</div>	
Receiver Guid	<div>{00000000-0000-0000-0000-000000000000}</div>	

Use case

A narrative designer wants to launch an interactive object from a quest blueprint. They create and configure a node. When a signal arrives, it forwards it to the corresponding interactive object.

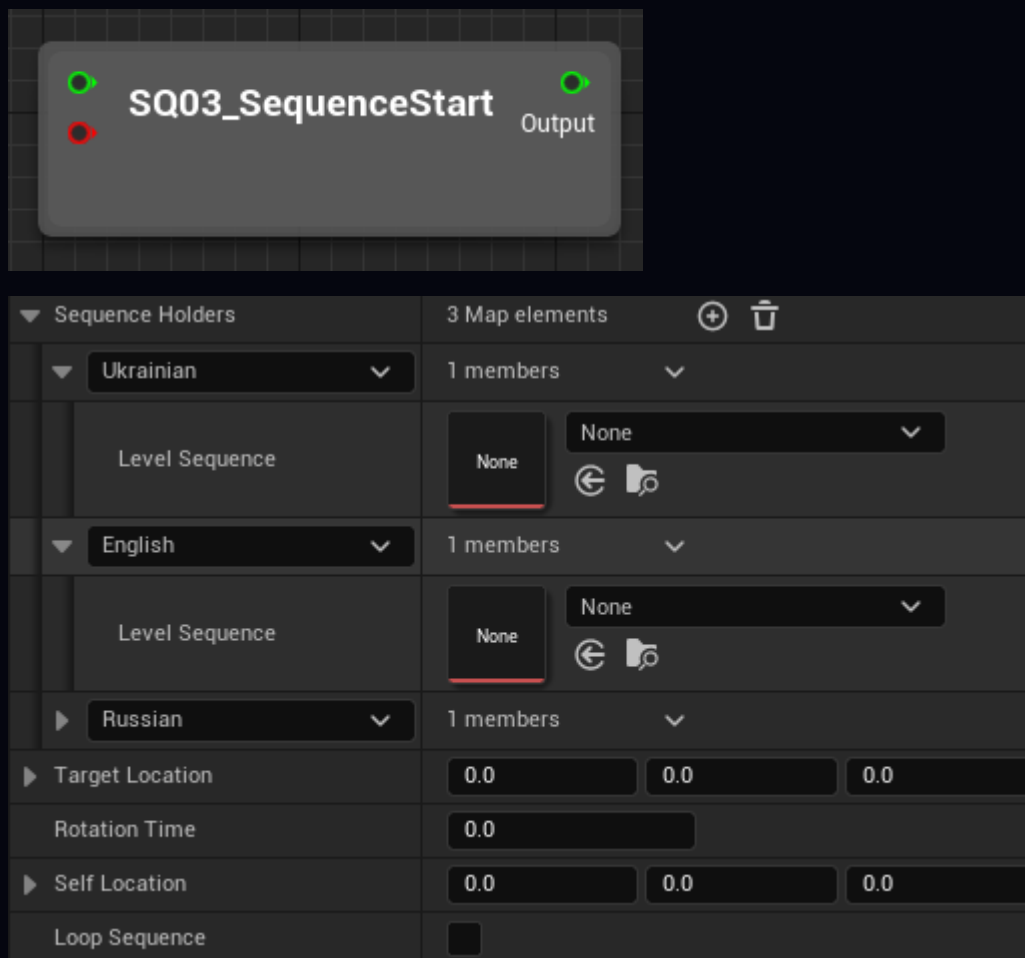
How to use

1. Select the button in the **Receiver Component Ref** field.
2. Find the interactive object in the list to which the Signal Receiver node should send a signal.
3. Select the Signal Receiver component.
4. Save the changes.

Sequence Start

This quest node launches a sequence asset.

The node ends after the sequence is complete, unless the **Loop Sequence** checkbox is selected. With this setting, configure the node's exclusion conditions.



How to use

Parameter	Description
Sequence Holder	The sequence asset launched by the node. The number of sequence assets is currently tied to the number of game localizations.
Loop Sequence	The checkbox that enables automatic sequence repetition until the node is excluded. It is important to configure the node exclusion conditions when this parameter is activated.

Set AI Behavior

Set AI Behavior is a set of quest nodes that allows setting a specific behavior goal for an NPC's AI, which it will continue to perform until the node is active or until the goal is achieved or failed.

This is a multifunctional combiner node. It allows assigning specific behavior to an NPC or squad.

Current behavior types:

- Berserker
- Escort
- Follow
- Guard
- Kill
- Move To
- Patrol Fields
- Stay
- Target Shoot Fields
- Use Ability

All behavior types can be configured to one of the following priorities:

1. Higher than A-Life – an NPC ignores A-Life events within a certain radius.
2. Higher than Anomaly Fields – an NPC ignores anomalies and attempts to bypass them, following the configured behavior pattern.
3. Higher than Radiation Fields – an NPC ignores radiation and follows the configured behavior pattern.
4. Higher than Emission – an NPC ignores emissions and attempts to find a shelter, following the configured behavior pattern.
5. Higher than Combat – an NPC ignores combat and follows the configured behavior pattern.
6. Ignore Threat – an NPC ignores danger and follows the configured behavior pattern.

How to use

Parameter	Description
Priority	<p>Behavior execution priority. Choose from the following:</p> <ul style="list-style-type: none">• Higher than A-Life – an NPC ignores A-Life events. This is the basic and primary priority.• Higher than Anomaly Fields – an NPC ignores anomalies and attempts to bypass them, following the configured behavior pattern• Higher than Radiation Fields – an NPC ignores radiation, following the configured behavior pattern• Higher than Emission – an NPC ignores emissions and attempts to find a shelter, following the configured behavior pattern• Higher than Combat – an NPC ignores combat, following the configured behavior pattern• Ignore Threat – an NPC ignores danger, following the configured behavior pattern
Can be interrupted	<p>This setting is responsible for the ability to interrupt behavior by starting a dialog with the player.</p> <p>When this setting is disabled, attempting to start a dialog will result in the NPC saying a Busy Comment and continuing to perform the goal.</p>
Weapon State	<p>A setting that determines how an NPC will handle weapons while performing behavior.</p> <p>No Weapon – a character uses weapons only in accordance with the rules for responding to threats.</p> <p>Relaxed – a character holds the weapon in their hands but does not aim at the target. An intermediate state between no weapon in hand and a weapon at the ready.</p> <p>Combat – a character performs the behavior with the weapon at the ready.</p> <p>Keep Current – a character will keep their current weapon state.</p> <p>The setting works with Stay Idle, Move To, Escort, and the Look At node. A character will walk or stand with their weapon at the ready.</p>
Fields	<p>The type of behavior assigned to the selected character</p>
Ignore	<p>An array of settings that allows for more customization of the threats ignored by the character:</p> <ul style="list-style-type: none">• Anomaly Fields• Radiation Fields• Emission• Combat• Threat

React on Approach with Weapon	A parameter that determines whether an NPC will react to a player with a weapon in their hands if they pass by or approach the NPC. Initially disabled for all quest NPCs. NPC reaction – aiming at the player until they put away their weapon.
React on Non-enemy sound	A parameter that determines whether NPCs will react to various actions (shooting, grenades, etc.) performed by allied or neutral characters, including the player. Initially disabled for all quest NPCs. NPC reaction – search for the source of the threat around them and return to Behavior after the threat is over.
FailureBy	An array of settings that allows for more customization of the conditions, which fail behavior execution: <ul style="list-style-type: none"> • Emission • Combat • Threat
Target Placeholder	The node target is an NPC or squad performing a specified type of behavior
Success Pin	The pin that sends a signal after successful node execution
Failure Pin	The pin that sends a signal after the node fails
Dead Pin	The pin that sends a signal after the death of the target placeholder

AI Behaviour fields

Field	Description
Escort	<p>AI Behaviors: Escort</p> <p>Behavior that forces the selected character to follow the player to a specific point.</p> <p>The following parameters can be set in the node:</p> <ul style="list-style-type: none"> • Movement type – movement speed • Waiting player – whether the character will wait until the player enters the configurable radius • Radius – the radius from the character in which the player must be for the escort to work (the NPC follows the route) • Waiting Time – the time during which the player must enter the radius for the character to start moving. If the player does not

	<p>enter the radius during this time, the node will end via a Failure pin.</p> <ul style="list-style-type: none"> • Locations and comments: <ul style="list-style-type: none"> • Target Location – the location where the character will go. Upon reaching the final location, the node will end via the Success pin. • Reach Point Comment – the dialog that starts after arriving at the desired point. Optional setting. • Dialog Members – participants of the dialog, if the Reach Point Comment is configured. • Comments – comments that can be played in certain game situations: <ul style="list-style-type: none"> • Player Leaves While Moving • Player Leaves While Combat • Starts Battle • Finish Battle • Reach Anomaly Zone • Leave Anomaly Zone <p>The node ends via the Success pin after the character reaches the Target location.</p>
Follow	<p>Behavior that allows setting a specific behavior goal for an NPC – to follow a specific object or player.</p> <p>The following parameters can be set in the node:</p> <ul style="list-style-type: none"> • Movement Pose – full height or crouching • Movement type – movement speed • Min Acceptable Distance – the minimum distance to the object for the node to be considered completed (via a success pin) • Max Acceptable Distance – the maximum distance to the object at which the character stops following the object • Goal Target – the point, character, or object to which the node's target should move <ul style="list-style-type: none"> • Follow Player – use it if you need to follow the player <p>Unlike Move To, Follow cannot complete itself. To complete this node, you need to exclude it.</p>
Guard	<p>Guard is a type of AI behavior used to guard specific objects or areas on the game map.</p> <p>The following parameters can be set in the node:</p> <ul style="list-style-type: none"> • Guard Type – select what needs to be protected <ul style="list-style-type: none"> • Target – select the NPC or squad as Guard Target • Player – select a player as the protected object • Zone – select the Guard Zone <ul style="list-style-type: none"> • Guard Zone Shapes – select specific Guard Zone Placeholders • Factions – select the factions from which the Zone needs to be protected

	<ul style="list-style-type: none"> • Include Player As Goal – select the player as the character from whom the Zone needs to be protected
Kill	<p>Behavior that forces the node target to kill the selected character. The following parameters can be set in the node:</p> <ul style="list-style-type: none"> • Kill Target Type – setting responsible for who needs to be killed <ul style="list-style-type: none"> • NPC • Player • Area – killing all characters within a Radius from the Target Location center • Simulate Battle • Ignore Cover – ignoring cover by the node target (those who kill) <p>The node ends via the Success pin after the character kills the node target.</p>
Move To	<p>AI Behaviors: Move To Behavior that forces the NPC to move along a specified route. The following parameters can be set in the node:</p> <ul style="list-style-type: none"> • Movement Type – movement speed • Move To Success Range – radius from the final point sufficient to complete the node via the Success pin • Movement Target – type of character navigation in motion <ul style="list-style-type: none"> • Path Legacy – select any object as the Movement Target, and the character will go to its coordinates • Path – Move To Placeholder and the checkpoints configured in it will be used to specify the coordinates of the place to move to • Dynamic Object – the character will move towards a target whose coordinates change (it moves around the map) • Move To Failure Range – the distance from the target at which the node will end via the Failure pin <p>The node ends via the Success pin after the character reaches the Success Range around the final point of the route.</p>
Patrol Fields	<p>AI Behaviors: Patrol Behavior that sends NPCs to patrol specific points. The following parameters can be set in the node:</p> <ul style="list-style-type: none"> • Target – Patrol Placeholder, which is responsible for the patrol area • Patrol Repeat Count – the number of patrol cycles • Patrol Graph Point – the point in the placeholder from which the patrol will start <p>The node ends via the Success pin after the character has completed the configured number of patrol cycles. Interrupting and resuming patrols is similar to how contextual actions work: the character leaves the patrol when a threat is detected and returns to the patrol after the threat is eliminated.</p>

Stay

AI Behaviors: Stay

Behavior that allows NPCs to stand at a given point with a given animation or perform a contextual action.

The following parameters can be set in the node:

- Movement Type – the speed at which the character will move towards Contextual Action
- Look At Player – whether the character will turn their head towards the player within the configured radius
 - Is Aiming at Player – whether the character will aim at the player when they enter the Look At Player radius
- Stay Type
 - Animation – the character will stop at the point where they are currently located
 - Idle Animation – setting the selected animation on the target node
 - Contextual Action – the character will go to the Contextual Action entry point and stand there
 - Search in Radius – the character will search for any suitable contextual action within the configured radius
 - Target – selection of a specific contextual action for the target
 - Use Custom Path – the ability to set the path to the contextual action via Move to Placeholder
 - Idle – the character will stand at the selected point in the Idle animation
 - Idle Position – the point where the character will stand
 - Look At – the point where the character will look or aim

Using Stay Type = Animation is not recommended. Most AI actions are tied to working with a Contextual Action.

Use Animation when you need to play a specific animation on a character, but the contextual action or scripter sequencer tools are not suitable.

How NPCs behave with different combinations of Weapon State and Look At Player:

- No Weapon + Look At Player – the character looks at the player
- No Weapon + Look At Player + Is Aiming At Player – the character takes out a weapon and aims at the player
- Relax Weapon + Look At Player – the character looks at the player with a weapon in their hands, but does not aim at the player
- Relax Weapon + Look At Player + Is Aiming At Player – the character raises their weapon and aims at the player
- Combat Weapon + Look At Player – the character lowers their weapon (Relax Weapon) and looks at the player

	<ul style="list-style-type: none"> • Combat Weapon + Look At Player + Is Aiming At Player – the character aims at the player
Target Shoot Fields	<p>AI Behavior: Target Shoot</p> <p>A type of NPC narrative behavior in which the NPC shoots at a specified target.</p> <p>The following parameters can be set in the node:</p> <ul style="list-style-type: none"> • Shoot From Location – the shooter's position • Movement Type – the speed at which the character will move to the shooter's position • Shoot At Location – the target at which the character will shoot • Shots Queue Count – the number of attacks required to complete the node via Success Pin. If it is 0, the node will never complete on its own.
Use Ability	<p>The node that forces the selected character to use an Ability available to them.</p> <p>The following parameters can be set in the node:</p> <ul style="list-style-type: none"> • Ability – a drop-down list of abilities available to the character • Time to Use – the time during which the character must use the ability. After the time is over, the node completes via Failure Pin. If it is 0, the node will not fail, and the character will try to use the ability. • Ability Target – the target to be attacked with the skill. If it is empty, the character uses the skill on the first encounter when the conditions for the skill to trigger are met. <p>A character can use an ability that has not been added to their active pool via Set NPC Sequential Ability if the ability is suitable for the character.</p>
Looting (WIP)	<ul style="list-style-type: none"> • TargetPlaceholder – an NPC that should perform looting (the looter) • Fields – Loot – selection field for looting behavior • LootTarget – NPC\Stash\corpse that needs to be looted, you can select one target (the loot) • Ignore, Failure by, and React on fields are identical to those currently presented in the node • ExcludeItem contains a drop-down list of items that NPCs will NOT loot (i.e., items will remain in the corpse/stash). On the one hand, this will allow you to play more with the looting narrative (for example, a stalker leaves canned meat in their inventory because it's disgusting), and on the other hand, it will allow you to control the moment of overload in NPCs. • LootGoalItem - contains a drop-down list of items that NPCs will loot selectively, i.e., they will only take these items from the entire inventory. If the required item is not present in the specified placeholder, the signal goes to the fail input. <p>NOTE</p> <p>If ExcludeItem and LootGoalItem contain the same placeholders, then</p>

when saving the blueprint, a warning should be issued: "ExcludeItem and LootGoalItem shouldn't contain the same item." If the warning is ignored, priority is given to the LootGoalItem parameter when the node is launched

After the node starts, the following actions occur:

- The corpse is reserved for searching, making it unavailable for searching by other NPCs or the player.
- The NPC starts moving towards the corpse.
- When approaching the corpse at a sufficient distance, the looting animation plays.
- If there are no items in the corpse, the signal goes to fail.
- If the specified placeholder does not become a corpse (the NPC survives), the signal goes to fail.
- If the placeholder does not contain the selected items in the LootGoalItem field, the signal goes to fail.
- If the search is interrupted by combat, the signal will go to the fail output.
- If the search is interrupted by an emission, the signal goes to the fail output.
- If the player interacts with NPCs during looting, the busy topic is played.
- When searching, NPCs take items from the corpse and transfer them to the inventory.
- Looting the corpse is not interrupted by the Set Dialog node – NPCs can talk while looting.
 - The Interact Trigger widget for the corpse is displayed as locked for the player.
- After searching the corpse, the inventory state must be preserved for both the searcher and the searched in the visualization mode and offline.
- Reserving a corpse by a quest NPC has higher priority than a non-quest NPC. If a quest NPC reserves a corpse, the goal and reservation of a non-quest NPC for this corpse should be canceled.
- Looting should work with dialogs in the same way as other nodes of the Set AI behavior type. That is, a dialog node can be launched while looting is running.
- Quest NPCs DO NOT speak during looting, unless these lines are manually set via Set Dialog.

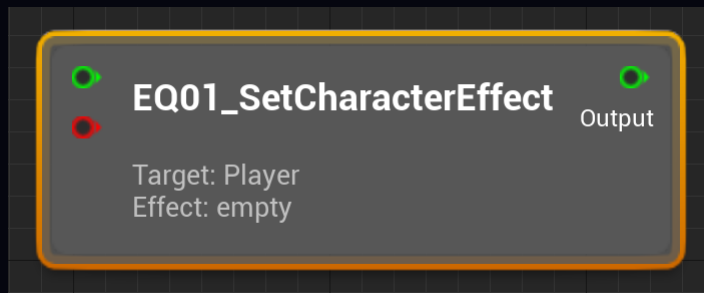
When starting a conversation with an NPC affected by the Move To, Follow, and Escort behaviors, these behaviors are paused and resumed after the dialog is complete.

The same logic applies to the Kill victim (the one being killed, Kill Target Type. If you start a dialog with the victim, the Kill behavior for the killer (the one who kills, **Target Placeholder**) is paused. The behavior resumes after the dialog is complete.

Set Character Effect

This node allows applying a selected effect to NPC_placeholder or the player.

References



CHARACTER TARGETED ACTION	
Effect SID	Empty
Target Placeholder	
Target	None
Target Player	☐

Use case

- A narrative designer needs to apply an intoxication effect to Skif after he drinks vodka in a dialog or scene. Set Character Effect is suitable for this.
- A narrative designer has planned for the player and NPC to pass through a field of chemical anomalies. To ensure the NPC's survivability, you can apply an effect that protects against chemical anomalies using the Set Character Effect node.

How to use

1. Create the node.
2. Select **Target Placeholder**.
 - a. If the target is a player, set the target player checkbox to **True**
3. Select an effect in Effect SID.
 - a. Not all effects work correctly or work at all on NPCs and players. Below are some of the effects that are available and potentially the most frequently used:
 - i. Bleeding
 - ii. Drunkenness

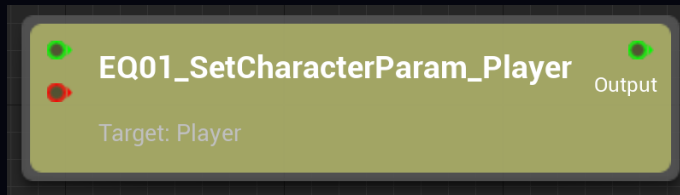
- iii. Radiation
- iv. Strike Damage
- v. Sleepiness
- vi. Hunger
- vii. Concussion

4. Save the changes.

Set Character Param

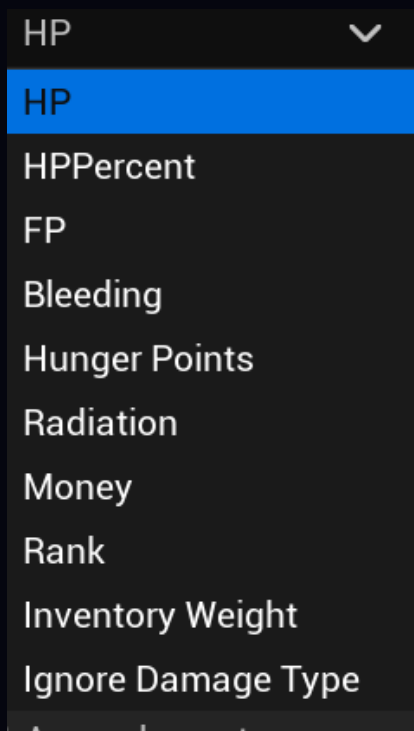
This node allows changing parameters for a player or selected NPC.

References



CHARACTER TARGETED ACTION	
Params	0 Array elements + 🗑️
▼ Target Placeholder	
Target	None ▼ 🔍 ✎
Target Player	☐

▼ Params	1 Array elements + 🗑️
▼ Index [0]	▼
Modified Character...	HP ▼
Change Value Mode	Set ▼
New Value	0.0



Use case

- A narrative designer needs to add Skif money as a reward for the quest. Set Character Param with the Money setting allows you to do this.
- A narrative designer needs to make the mainline character immortal. The Ignore Damage Type node parameter will help with this.

How to use

Parameter	Description
Health (HP)	Absolute value of health points setting
Health Percent (HPPercent)	Percentage of the maximum setting
Stamina (FP)	Absolute value of stamina setting
Bleeding	Bleeding effect attribution
Hunger (Hunger Points)	Hunger setting
Radiation	Radiation exposure setting

Money	Money setting
Rank	Rank change
Weight (Inventory weight)	Maximum carry weight change
Ignore Damage type	Immunity to all incoming damage setting

1. Create the node
 2. Select the Target whose parameters the node will change
 3. Configure one or more parameters that need to be adjusted. A parameter is added by pressing the plus button: .
- Configuring the Health (HP) parameter:
 - Select the type of HP change: set, add, subtract, multiply, divide.
 - Enter the desired value in the **New Value** field.
 - Configuring the Health Percent (HPPercent) parameter:
 - Select the type of HP change: set, add, subtract, multiply, divide.
 - Enter the desired value in the New Value field.
 - Configuring the Stamina (FP) parameter:
 - Select the type of FP change: set, add, subtract, multiply, divide.
 - Enter the desired value in the **New Value** field.
 - Configuring the Bleeding parameter:
 - Select the type of Bleeding change: set, add, subtract, multiply, divide.
 - Enter the desired value in the **New Value** field.
 - Setting the Hunger (Hunger Points) parameter:
 - Select the type of change for Hunger Points: set, add, subtract, multiply, divide.
 - Enter the desired value in the **New Value** field.
 - Setting the Radiation parameter:
 - Select the type of Radiation change: set, add, subtract, multiply, divide.
 - Enter the desired value in the **New Value** field.
 - Setting the Money parameter:
 - Select the type of Money change: set, add, subtract, multiply, divide.
 - Enter the desired value in the **New Value** field.
 - Setting the Rank parameter:
 - Select the rank to be set: Newbie, Experienced, Veteran, Master.
 - Setting the Weight (Inventory weight) parameter:
 - Select the type of change for Inventory weight: set, add, subtract, multiply, divide.
 - Enter the desired value in the **New Value** field.

- Setting the Ignore Damage type parameter:
 - Select the desired damage reaction type: None – standard reaction, Invulnerable – cannot be wounded or killed, Unkillable – can be wounded but cannot be killed.

4. Save the changes.

Set Dialog

This node starts a dialog. The node obtains dialog information from the selected Dialog Asset (TextTool Asset) in the **Dialog To Set** field of the node. Dialogs are created in TextTool, where the number of actors participating in the dialog is also configured. After configuring the dialog asset in TextTool, you need to create it in the engine and then script the dialog logic.

References



Dialog to Set	None	None
▼ Dialog Members	1 Array elements	
▼ Index [0]	Remove topic target	↩
Member Name	Remove topic target	
▶ Member		↩
Talk Through Radio	☐	↩
NPCTo Start Dialog	Any member	↩
Start Forced Dialog	☐	
Ignore Global WFR	☐	
Can Exit Anytime	☒	
Call Player	☐	
Call Player Radius	1000.0	
Seek Player	☐	
Is Comment	☐	
Override Dialog Topic	None	

Use case

- A narrative designer needs a dialog for the player to take the quest.
- A narrative designer needs to script the reaction of the NPC guard to the player who tries to interact with him.

- A narrative designer needs to script the Ward fighters' radio communication with the player if he approaches the Ward checkpoint.

How to use

Parameter	Description
Dialog to Set	<p>Selection of the dialog asset played by the node. There are two checkboxes inside the asset:</p> <ul style="list-style-type: none"> • On the Go – dialog that does not block the player's movement and limits the number of responses in WFR to two (Q-E). • Can be interrupted – when this checkbox is <i>включено</i>, the dialog can be interrupted by external sources (combat, anomaly, emission) during playback. After the threat is eliminated, the dialog will continue from the last phrase that was interrupted. <p>IMPORTANT This setting also prohibits interaction with any other NPC while a line with the Can be interrupted set to False is playing. Please use it with caution.</p>
Talk Through Radio	Allows starting a dialog without specifying an NPC placeholder (the sound will come out of Skif).
Voice Modulator	Allows assigning specific post-processing for the sound of a particular actor: radio, loudspeaker, etc.
NPC To Start Dialog	Responsible for which dialog participant to interact with to start the dialog. The Any member option allows starting a dialog when interacting with any of its participants.
Start Forced Dialog	When selected, it starts the dialog immediately after a signal is received at the input pin. If the checkbox is cleared, the dialog starts when interacting with the selected NPC in NPC To Start Dialog after the signal arrives at the node.
Ignore Global WFR	When selected, it allows ignoring WFR when starting a dialog. If the checkbox is cleared, the player will be taken to the WFR window when the dialog starts.
Can Exit Anytime	When selected, allows the player to exit the dialog during it by pressing Esc. If the checkbox is cleared, the player can only leave the dialog via the end.
Call Player	Is obsolete and not recommended for use – it is better to use Is Comment + forced dialog and the Condition node. However, the functionality works. When the checkbox is selected, the dialog starts as soon as a signal is received, and the player enters Call Player Radius (in

	units), which is configured by the scripter when the Call Player option is activated.
Is Comment	Does not block the player's movement, nor does it launch the UI dialog window. In case of interruption, it excludes the node.
Override Dialog Topic	Allows overwriting one of the N default dialog topics

Creating a dialog asset

1. Create a dialog in Quest Tool.
2. Script the dialog blueprint in the engine

Node configuration options

Creating a comment

1. Create a Set Dialog node.
2. Select the desired option in **Dialog to Set**.
3. Select all dialog members.
4. Select the **Is Comment** checkbox.
5. Configure **Start Forced Dialog**.
6. Configure **Talk Through Radio**.

Creating forced dialog

1. Create a Set Dialog node.
2. Select the desired option in **Dialog to Set**.
3. Select all dialog members.
4. Select the **Start Forced Dialog** checkbox.
5. Configure **Is Comment**.
6. Configure **Talk Through Radio**.

Replacing dialog topics

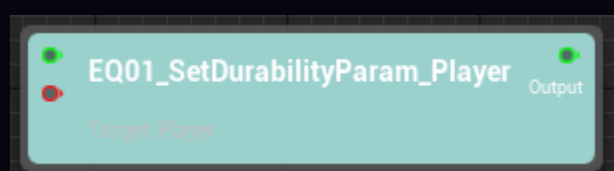
1. Create a Set Dialog node.
2. Select the desired option in **Dialog to Set**.
3. Select all dialog members.
4. Select the desired topic in **Override Dialog Topic**.

If the first or last node in the dialog blueprint is deleted/changed, select the **Update Node** button for the dialog node in the quest blueprint and resave the quest blueprint.

Set Durability Param

This node changes the weapon state in the hands of the selected character.

References



▼ CHARACTER TARGETED ACTION	
Should Target Primary Weapon	☐
Primary Weapon Durability	<input type="text" value="0.0"/>
Should Target Secondary Weapon	☐
Secondary Weapon Durability	<input type="text" value="0.0"/>
Should Target Pistol	☐
Pistol Durability	<input type="text" value="0.0"/>
▼ Target Placeholder	
Target	<input type="text" value="None"/> <input type="button" value="Q"/> <input type="button" value="↗"/>
Target Player	☐

Use case

- In the context of the quest, a narrative designer needs to show that a bandit does not know how to use a sniper rifle. To do this, the rifle must be rendered unusable using the Set Durability Param node.
- During the quest, a mutant attacks the player, damaging their machine gun. A narrative designer uses the Set Durability Param node to reduce the rifle.

How to use

1. Create the node.
2. Select **Target Placeholder**.
3. To set a player, select the Target Player checkbox.
4. Select the weapon slot in which the weapon will be affected by the node:
 - Primary Weapon – the first weapon slot
 - Secondary Weapon – the second weapon slot
 - Pistol – the slot for pistols
5. Set durability – 100 is the maximum value, 0 is the minimum.
6. Save changes.

Set Emotional State

This quest node sets the NPC's facial emotion.

You can configure the following emotional masks in the node:

- Idle – NPC facial animation when idle, moving, or performing other activities that do not require separate facial animations
- Surprised – emotional mask of a surprised NPC
- Angry – emotional mask of an angry NPC

- Scared – emotional mask of a frightened NPC
- Sad – emotional mask of a sad NPC
- Happy – emotional mask of a happy NPC
- Some more emotions – emotions can be updated

One node can be set for one NPC at a time. Setting a new node ends the previous one. Node exclusion returns the NPC to the default emotional mask.

The transition of the selected NPC to Wounded State or their death ends the active node. Transitioning to combat does not affect the node's state.

Emotional masks that are set through dialog *override* the emotions that the node sets. At the same time, the node is not ended or excluded. It will continue to work after the dialog that overrode the emotional masks is completed.

If the node target is a squad, emotions are set for all its members. At the same time, a parallel node set for one of the squad members will have higher priority and will change the NPC's emotion. Node exclusion will return the NPC their squad's emotion.

Setting a new node to the same target will cause the older node to end. It works for NPC-NPC and squad-squad, but does not work for NPC-squad. If a character without facial emotion becomes a member of a squad with emotion, that character will be affected by the squad's emotion.

Resetting the AI of the selected NPC ends the node, as it ceases to be a quest agent.

Use case

The node is used anywhere in the game where you need to show emotions.

How to use

1. Create the node.
2. Select the NPC target
3. Select the emotion.
4. Set up the node exclusion condition.
5. Save the quest blueprint.

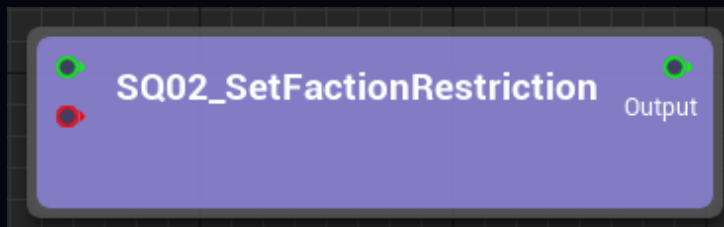
Set Faction Restriction

This quest node activates ScriptedNavModifierVolume, aka Space Restrictor, for a specific faction.

The node can be used to set restrictions in **Nav Area Class** without changing the project parameters.

The node will be active until excluded. Node exclusion disables the restrictions it sets.

Node view



▼ Nav Modifier

Nav Modifier
None
▼

▼ Set Faction Restriction Editor Quest Node

Nav Area Class
None
▼

▼ Restricted Factions

Faction
Humanoid
▼

Restriction Type
None
▼

How to use

Parameter	Description
Nav Modifier	ScriptedNavModifierVolume, activated and deactivated by the node. Keep in mind that the restrictor will work as long as the node is active. To disable the restrictor, exclude this node.
Nav Area Class	Select 1 of 64 engine presets for navigation modifiers. Used when a specific faction needs to set restrictions across the entire map. Applies restrictions for the selected preset and corresponding navigation color. Does not change the basic settings of the project, which makes it convenient. Despite this, use the setting with extreme caution so as not to break anything in the game.
Restricted Faction	The faction to which the restriction will be applied
Restriction Type	The type of restrictions in the area: <ul style="list-style-type: none"> In – prohibited to enter Out – prohibited to exit

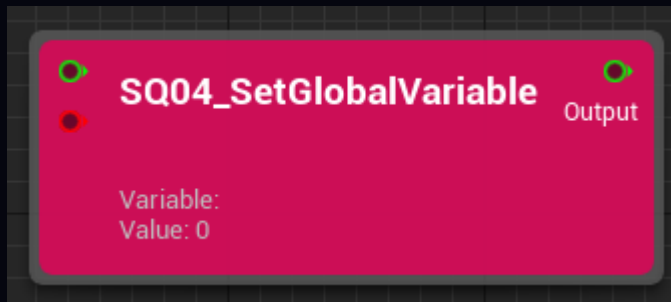
Set Global Variable

This node is required to control Global Variable.

With the help of the node, you can increase, decrease, multiply, divide, or set values in global variables.

The node ends immediately after the variable is changed. Node exclusion will not cancel its effect.

Node view



Set Global Variable Editor Quest Node	
Variable to Set	<input type="text" value=""/> ↕ ↶
Int Value	<input type="text" value="0"/>
Change Value Mode	<input type="text" value="Set"/> ▼

How to use

Parameter	Description
Variable to Set	Changeable Global Variable. You can change both Bool and Int.
Int Value	Numeric value changeable in Int Variable. If Global Variable = Bool Type, then this setting changes to the Bool Value checkbox.
Change Value Mode	Method of changing Int Global Variable: <ul style="list-style-type: none">• Set• Add• Subtract• Multiply• Divide

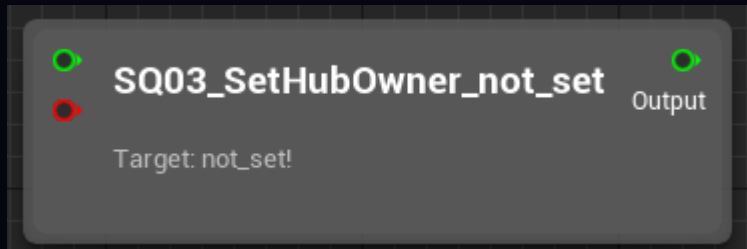
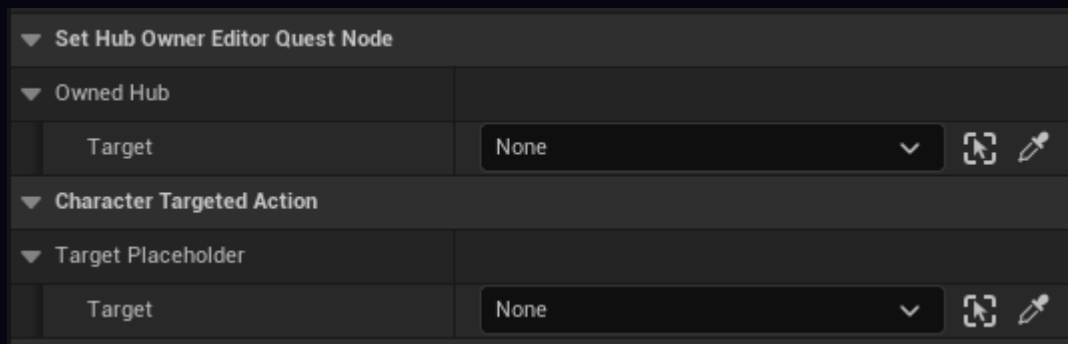
⚠ Path to the configuration file with variables:

Content\GameLite\GameData\GlobalVariablePrototypes.cfg

Set Hub Owner

This quest node sets the selected NPC as the Hub Owner of the selected hub. This status adds a special UI icon to the NPC.

Node exclusion removes the status from the character. The death of this NPC also removes this status.



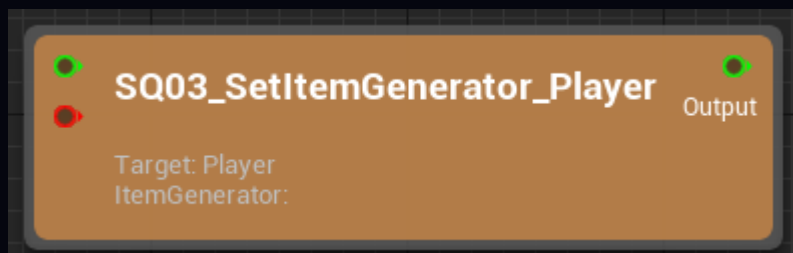
How to use

Parameter	Description
Owned Hub	The hub whose owner will be set by the node
Target Placeholder	The NPC that will become the hub owner. Only works with NPC Placeholder.

Set Item Generator

This quest node allows placing the selected Item Generator in the inventory of the selected entity with or without replacement.

The node ends after the Item Generator is configured. Node exclusion will not cancel its effect.



▼ Character Targeted Action	
Replace Inventory	☐
Item Generator	<div></div> ▼
▼ Target Placeholder	
Target	<div>None</div> ▼  
Target Player	☐

How to use

Parameter	Description
Replace Inventory	Activation of this parameter is responsible for replacing the inventory of the selected entity with the Item Generator. True – the container will be cleared, after which items from Item Generator will be added to the container. False – items from Item Generator will be added to the container without prior clearing.
Item Generator	Item Generator provided by the node
Target Placeholder	Item Generator will be added to this entity's inventory. This can be: NPC, squad, item container, dead Body, player.
Equip Items	The checkbox is responsible for whether items from Item Generator will be equipped in the corresponding slots: <ul style="list-style-type: none">• Equip WeaponPrimary – equip the primary weapon in the first slot• Equip WeaponSecondary – equip the weapon in the second slot• Equip WeaponPistol – equip the pistol• Equip Detector – equip the detector• Equip Armor – equip the armor• Equip Helmet – equip the helmet If the slot is occupied, the equipment will not be made.

	<p>If there are several items in the generator for one slot, the first one (the top one) will be equipped.</p> <p>If the armor slot is free, the helmet slot is occupied, and the armor being equipped has a built-in helmet, the equipped helmet will be removed in favor of the one built into the suit.</p> <p>If the helmet slot is filled with a helmet built into the armor, attempting to equip a helmet will not result in it being equipped.</p>
--	---

IMPORTANT

Be sure to use the node carefully with a squad! When **Target Placeholder** is set to **Squad**, the node applies its effect to all members of the squad, which means that items will be added to all characters in that squad.

Set Location Name

This narrative node is required to change location names. For example, at the beginning of the game, the location is called “A,” and after a certain quest, it is renamed to “B.”

How to use

Parameter	Description
Location	Marker prototype SID to select changed location
New Title	SID for a new title
New Description	SID for a new description. Can be left empty for an empty description.

Set Mesh Generator

This quest node changes the selected NPC's Mesh Generator to the one selected in the node.

The node ends after changing the character's mesh generator.

Node exclusion will not cancel its effect.

The node does not affect the Item Generator.

How to use

Parameter	Description
Target	<p>The character whose Mesh Generator will be changed.</p> <p>Carefully change Mesh Generator, so the change is not noticeable.</p> <p>The target can be an NPC, dead body, or squad.</p>

New Mesh Generator	The new Meshgenerator applied to the target
--------------------	---

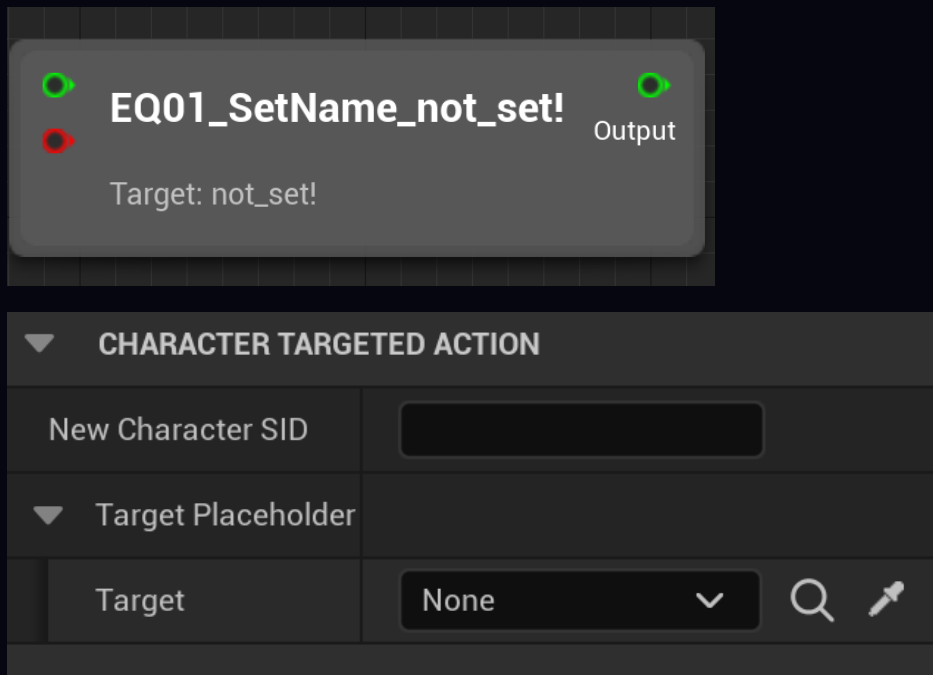
IMPORTANT

Sometimes the node needs to be combined with the Set Item Generator node so that the NPC's defensive parameters change along with the visuals of their costume.

Set Name

This node allows changing the displayed name of the selected NPC to the one existing in TextTool. If the node received a signal before the selected NPC was spawned, the node will wait for this NPC to spawn. Renaming the NPC completes the node.

References



Use case

A narrative designer needs to change a stalker's name, whom the player could meet in the Lesser Zone, to another name if the player meets this character in Prypiat. To do this, a narrative designer creates characters with the appropriate names in TextTool and uses the Set Name node in the script.

How to use

1. Create the node.
2. In TextTool, create characters for the starting NPC name and for the new NPC name.
3. Update TextTool in the engine.
4. In the **Target Placeholder** field, select the NPC to be renamed.
5. In the **New Character SID** field, enter the SID created in TextTool for the character with the new name.
6. Save the changes.

Set NPC Sequential Ability

This node allows removing or setting combat Sequential skills for the selected NPC or mutant.

Attempting to set unsuitable skills for the selected target is theoretically impossible, but if such an attempt occurs, the engine will crash with a broken game date when saving the blueprint.

Information about skills is taken from the SID config of the selected character. The desired skill is selected from a drop-down list.

When configuring a skill, you can set its weight – a parameter that determines the chance that the character will use this skill in combat.

The node ends immediately after changes are made to the NPC's skill list.

Node exclusion will not cancel its actions.

Squad cannot be selected as the target of the node since squad members may be NPCs/mutants with different skills.

Use case

A narrative designer needs to prevent a bloodsucker from becoming invisible for a specific combat sequence script.

How to use

1. Create the node.
2. Select **Target**.
3. Select Set Type:
 - a. **Add** – add a skill or set its weight
 - b. **Remove** – remove a skill
4. Select **Sequential Ability**.
 - a. Set the weight.
5. Save the changes.

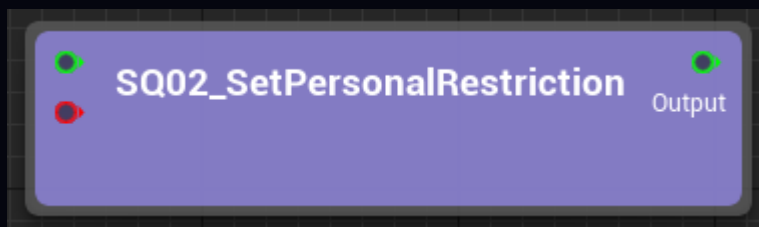
Set Personal Restriction

This quest node activates ScriptedNavModifierVolume, aka Space Restrictor, for a specific NPC or squad.

The node can be used to set restrictions in Nav Area Class without changing the project parameters.

The node will be active until excluded. Node exclusion disables the restrictions it sets.

Node view



▼ Nav Modifier

Nav Modifier
None
▼

▼ Set Personal Restriction Editor Quest Node

Nav Area Class
None
▼

▼ Personal Restriction

▼ Target

Target
None
▼

Restriction Type
None
▼

How to use

Parameter	Description
Nav Modifier	ScriptedNavModifierVolume, activated and deactivated by the node. Keep in mind that the restrictor will work as long as the node is active. To disable the restrictor, exclude this node.
Nav Area Class	Select 1 of 64 engine presets for navigation modifiers. Used when a specific character needs to set restrictions across the entire map. Applies restrictions for the selected preset and corresponding color. Does not change the basic settings of the project, which makes it convenient.
Target	NPC or squad to which the restriction will be applied
Restriction Type	The type of restrictions in the area: <ul style="list-style-type: none"> In – prohibited to enter Out – prohibited to leave

Set Journal

This node controls the quest journal, quest stage, and marker. It is necessary for displaying information about current quests in the player's UI journal (PDA), namely:

- Journal Quest Name
- Journal Quest Description
- Quest Stage
- Stages Description
- Placed Markers

The node can set a marker on the map. There are two types of markers:

- Marker – any object on the map can be selected as a target, or a point can be specified by XYZ coordinates. It is static. If an object that can move around the WorldMap is selected as the target, the marker will not follow it, but will remain at the point specified in the node settings.
- Quest Marker – only quest actors (NPC placeholder, item, stash, actor, teleport placeholder, etc.) can be selected as targets. The quest marker is dynamic. If its target can move around the map, the marker will be updated and follow it. If the target of this marker has not yet been loaded (is far from the player), the marker points to the starting coordinates of the selected object.

Both types of markers can be set as Zone markers, which have a customizable radius, measured in units. The center of such a marker is the coordinates of the selected target.

The node ends after performing the action configured in it.

The node exclusion, after it has already received a signal and executed, does not complete or cancel the actions performed by the node.



▼

Set Journal Editor Quest Node

Entity

Quest Stage

▼

Action

Start

▼

Journal Quest

empty

▼

Quest Stage SID

empty

▼

▼

Placed Markers

1 Array elements

⊕

🗑

▼

Index [0]

9 members

▼

Use Quest Object Location

☐

▶

Quest Target

▼

Target Location

Target Location

None

▼

📐

✎

▶

Target Location Coordinates

0.0

0.0

0.0

Is Zone Marker

☐

▼

Marker Conditions

Conditions

0 Array elements

⊕

🗑

NOTCondition

☐

Condition Check Type

And

▼

Add on Condition

☐

Remove on Condition

☐

How to use

Parameter	Description
Entity	<p>The journal entity that a specific node works with:</p> <ul style="list-style-type: none"> Quest Quest Stage
Action	<p>Action performed by a specific node:</p> <ul style="list-style-type: none"> Start – start the journal entity Finish – complete the journal entity Fail – fail the journal entity Cancel – cancel the journal entity Change Description – change the description of a task in the journal <p>If there is a Quest or Quest Journal in QBP that has a Start Action but no Finish, the engine will warn you that the finish is missing when you save QBP.</p>
Journal Quest	Journal Asset launched by the node
Quest Stage SID	Quest stage launched by the node
Placed Markers	Markers displayed in the interface when a specific quest stage is activated.

	<p>A regular marker cannot change its position. A Quest marker is used for this.</p> <p>If you restart an already active Quest Stage, but with different marker settings, the information about the marker will be updated in the stage.</p>
Use Quest Object Location	<p>A parameter that allows you to set a marker on a quest object or NPC.</p> <p>Quest Marker will change its location along with the object to which it is attached.</p> <p>Quest Marker on an item will disappear without additional conditions when the item is picked up.</p> <p>When setting Quest Marker on a squad, the marker appears on all squad members.</p>
Target	<p>The object on which the marker will be set.</p> <p>For Use Quest Object Location = true, only Quest Target (NPC, item, container, interactable, etc.) can be used.</p>
Remove on Death	<p>A parameter that allows you to remove the marker from quest objects without additional conditions if they have died.</p> <p>Visually, it is a checkbox with a default value of False, located in the Quest Target section.</p> <p>When Quest Marker is set and Remove on Death is selected for a squad, the marker appears on all squad members and disappears without additional conditions when a squad member dies. The death of one squad member does not affect the markers of other squad members, i.e., the marker on each member remains until their death or the Remove on Condition condition is met.</p>
Is Zone Marker	<p>A parameter that allows the marker to highlight an area with a configurable radius. The center of the area is the Target.</p>
Map Radius	<p>The radius of the marker zone. Measured in units (1 unit = 1 cm).</p>
Zone Sub Markers	<p>Submarkers inside the zone, activated when the player approaches the Sub Marker Target Location at the configured Visibility Radius</p>
Add on Condition	<p>A parameter that allows adding a marker when the Condition is met</p>

Remove on Condition	A parameter that allows removing the marker when the Condition is met
Condition	The condition under which the marker appears or disappears. If you select Remove on Condition and Add on Condition simultaneously, the marker will only be displayed when the Condition is met. The marker appears dynamically when the condition is met and disappears dynamically when the condition is not met.

The node exclusion, after it has already received a signal and executed, does not complete or cancel the actions performed by the node.

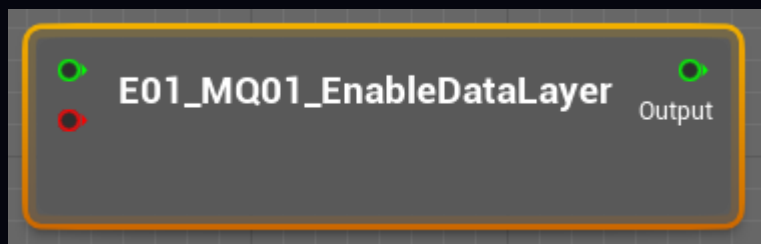
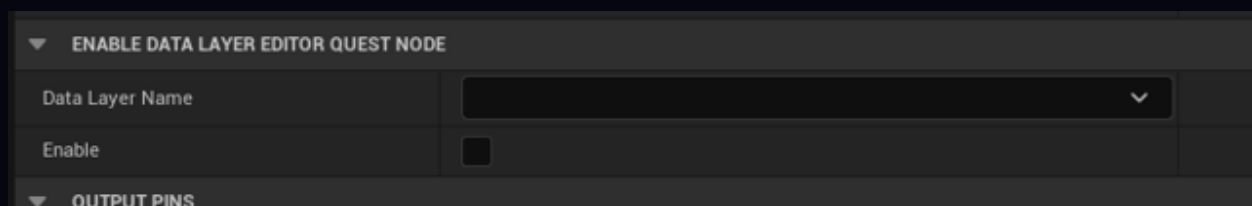
Features of node completion:

- If the Quest is completed via Finish, but there are still uncompleted Quest Stages, these stages are completed via Cancel.
- If the Quest is completed via Fail, but there are still uncompleted Quest Stages, these stages are completed via Fail.
- If the Quest is completed via Cancel, but there are still unfinished Quest Stages, these stages are completed via Cancel.
- If the Quest is completed via Cancel or Fail, and all Quest Stages are completed via Finish, the last stage will be changed to Fail status. And when saving the QBP, the engine will warn the scripter about an incorrect script.

Enable Data Layer

This node allows loading and unloading a specific Data Layer via script. This is necessary for directed changes in level geometry, for example, after the fall of Duga or the opening of the passage from the Lesser Zone.

References



Use case

A narrative designer wants to visually show the destruction brought by the Monolith members in Zalissya after the Signal. To do this, they unload the Data Layer containing the geometry of “Zalissya before the Signal” and load the Data Layer with “Zalissya after the Signal.”

How to use

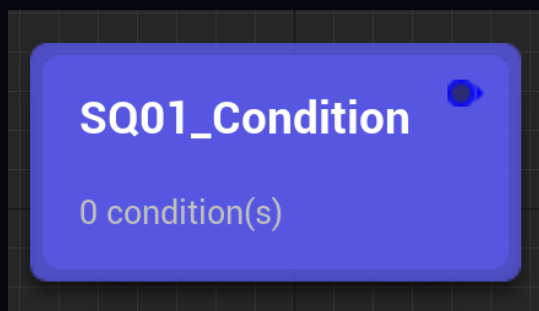
Creating a new data layer and placing assets on it is only possible with the support of programmers and level designers – the narrative designer cannot use this functionality without them.

If the required data layer is ready, select it in the **Data Layer Name** setting and select **Enable** if it needs to be loaded, or clear the checkbox if Data Layer should be unloaded. Usually, such nodes are used in pairs: one unloads existing Data Layer, and the other loads the one that has not been loaded before.

Condition/If

These nodes unlock the signal flow when certain conditions are met. The Condition node cannot trigger the signal on its own – it can only be connected in parallel with another node. The node will only pass the signal if the configured condition is met. The node does not end after passing the signal further, but continues to work and can be connected to other nodes. It is possible if the same check is needed there, and it does not spoil the readability of the blueprint.

The If node connects to other nodes in series. Unlike the Condition node, If instantly passes the signal through itself. If the conditions set in the node are met, the signal will go to the true pin, and the node will end. If the conditions are not met, the signal will go to the false pin, after which the node will end. In all condition settings, you can select the **NOTCondition** checkbox. This feature reverses the conditions for triggering the condition, making them the opposite of how they are configured. For example, if the Is Alive condition is configured for an NPC, but the **NOTCondition** checkbox is selected in the node, the condition will trigger if the selected NPC is dead. In the If node, this checkbox swaps **True** and **False**.



▼ Condition Editor Quest Node		
▼ Conditions List		
Conditions	0 Array elements	⊕ 🗑
NOTCondition	☐	
Condition Check Type	And	▼

▼ Index [0]	▼
Condition Type	HP ▼
Condition Compara...	Equal ▼
Numeric Value	0.0
▶ Target Character Pl...	

How to use

Parameter	Description
Conditions	Check used in a specific node.
NOT Condition	Setting that causes the check to trigger when it is NOT met
Condition Type Check	Method of checking an array of conditions: <ul style="list-style-type: none"> • And – to activate when all conditions are met • Or – to activate when one of the conditions is met

Parameter	Description
HP	Check for HP of the selected character
HPPercent	Check for the HP percentage of the selected character
Stamina	Check for the stamina points of the selected character. 100 stamina = the character is not tired.
Radiation	Check for radiation points of the selected character. 0 radiation points = the character is healthy.
Bleeding	Check for bleeding in the selected character. 0 bleeding points = the character is not bleeding.
Psy	Check the selected character's psy health. 0 psy points = the character is healthy.
Hunger Points	Check the selected character's hunger points. 0 hunger points = the character is full.
Money	Check the selected character's amount of money
Inventory Weight	Check the selected character's total inventory weight. Does not work on containers.
Is Alive	Check if the selected character is alive. If the Bool Value checkbox is cleared, the condition will work in case the character is dead.
Is Online	Check if the character is visualized. Visualization distance (online entry) is 15K units. Unloading distance (offline entry) is 18K units.
Faction Relationship	Check for the relationship between the selected character and the selected faction. In Condition Comparance, select the relationship comparison method, and in Relationship, select the relationship preset for comparison.

Personal Relationship	Check for the relationship between two selected characters. In Condition Comparance, select the relationship comparison method, and in Relationship, select the relationship preset for comparison.
Rank	<p>Check for the rank of the selected character. Most often used on the player to control their progression.</p> <p>The player's rank is changed manually via the Set Rank node.</p>
Effects	Check for an effect hanging on the selected character
Awareness	<p>Check for an NPC's awareness level. In Condition Comparance, select the method for comparing awareness with the reference in Awareness.</p> <p>Awareness types must correspond to the threat levels:</p> <ul style="list-style-type: none"> • Calm – 0 threat points • TurnHead – 100 threat points • SearchEnemy – 300 threat points • MovetoLocation – 800 threat points • CallAllies – 800 threat points • Battle – 1000 threat points
Note	Check if the player has a note
Item In Inventory	<p>Check whether the selected item is in the inventory of the selected character or container. Enter the number of items in Numeric Value and select the comparison method with the reference value in Condition Comparance.</p> <p>The With Equipped and With Inventory checkboxes are responsible for checking items in equipment slots and in the backpack, respectively.</p> <p>If Item Prototype sid is empty, the check will work for the presence of any item in the inventory.</p> <p>The check only works for items visible in the inventory (UI).</p>
AI Tagret	Check whether the selected character is the target for killing by another character
Global Variable	Please see Set Global Variable for more details.

Trigger	<p>Check whether the selected character overlaps with the selected trigger.</p> <p>If the condition is aimed at an NPC, you can configure the trigger's reaction to both living and dead NPCs in the React Type tab.</p> <p>When activated, the Triggers by Anybody checkbox allows the condition to trigger if any character interacts with the trigger.</p>
Distance to NPC	<p>Check for the distance from the selected character to another selected character. Works if Target Location changes its coordinates. Condition Comparance is responsible for the type of comparison with the reference value.</p> <p>Numeric Value is the reference value.</p>
Distance to Point	<p>Check for the distance from the selected character to the selected point. Condition Comparance is responsible for the type of comparison with the reference value.</p> <p>Numeric Value is the reference value.</p>
Look At Angle	<p>Check whether the player looked in the selected direction while standing in a specific place. Condition Comparance is responsible for the type of comparison with the reference value in Numeric Value in degrees.</p> <p>Target location is the object to be looked at.</p> <p>Trigger is the trigger in which the player must be located for the check to be triggered.</p>
Bridge	<p>Check for completion of the quest node. In Quest Blueprint, the blueprint from which the node information will be taken is set. The node itself is selected in Node Selector.</p>
Item In Container	<p>Check for the presence of a specific item in the selected container. Condition Comparance is responsible for the type of comparison with the reference value in Numeric Value. Item Prototype SID is responsible for the item to which the node will respond. Target Item Container is the container that the node checks.</p> <p>If Item Prototype SID is empty, the check will work for the presence of any item in the inventory.</p> <p>Does not work with NPCs and the player.</p>

Equipment in Hands	<p>Check for equipment used by the player. Item Prototype SID is responsible for the item that must be equipped. Equipment is responsible for the slot that the node will check.</p> <p>When you select the Equipment but do not select Item Prototype SID, the check will work if the player has any suitable item in their hands.</p>
Is Aiming	<p>Checks whether the player is looking at the sight. The selected Bool Value checkbox means that the node will work when aiming. If you clear the checkbox, the node will work in case the player is not aiming. Aim Duration is responsible for the amount of time the player must spend aiming for the check to trigger.</p>
Corpse Carry	<p>Check if the selected character is carrying a corpse. In Target Character Placeholder, select the character who should carry the corpse. The Target Player checkbox sets the player as that character. In Target Corpse Placeholder, select the corpse to be carried. The selected Any Body checkbox allows the node to work with any corpse.</p>
Journal State	<p>Check the status of the selected journal. Journal Entity sets the choice between a quest and a quest stage. Journal State – the selection of the quest or quest stage status that the node will trigger. Journal Quest – the selection of the asset journal. Quest Stage SID – the selection of the quest stage on which the node will work – this field is only active if Journal Entity is set to Quest Stage.</p>
Is Created	<p>Check whether the selected character or item has physically appeared in the world. If Bool Value is set to True, the node works in case the character exists. False – if the character does not exist. Target Placeholder – the character to which the node reacts.</p> <p>Make sure not to confuse it with Is Online. Is Created works even if the character went offline after spawning.</p>
Emission	<p>Check of the current emission status. If Bool Value is set to True, the node will work when the emission is active. False – it will work when there is no emission.</p> <p>Specific Emission – setting for the node to work on a specific script emission.</p>
Player Overload	<p>Check for the player's inventory overload. Bool Value sets whether the node will trigger if the player is overloaded, or vice versa. Include Partial Overload sets whether the node will trigger if the player is partially overloaded (above the base limit, but still able to walk).</p>

Is Wounded	<p>Check whether the selected NPC is in the wounded state.</p> <p>When Player Target is selected, the check will never trigger – the player cannot be in the wounded state.</p>
Contextual Action	Check whether the selected NPC is in the selected Contextual Action
Is Dialog Member Valid	<p>Check whether Optional Dialog Member is alive/spawned. If the check fails, the dialog label connected to this condition will be skipped.</p> <p>Only works within Dialog Blueprint.</p>
Weather	<p>Check for the current weather.</p> <p>Does not work with Emission – there is Emission, a separate check type.</p>
Has Item Type	<p>Check whether a certain type of item is present in the player/NPC's container or inventory.</p> <p>Item types that can be checked:</p> <ul style="list-style-type: none"> • Gun • Ammo • Artifact • Armor and Gasmask • Medicine • Food • Gun Attach • Quest Item <p>The check also works on equipped slots if the check target is the player.</p>
Sleepness	<p>Checks for the player's sleepiness points:</p> <ul style="list-style-type: none"> • 0 – the player is well-rested • 100 – the player is totally sleepy

Node State	<p>Check the state of a given node. Can be True or False for one of four state types:</p> <ol style="list-style-type: none"> 1. Not started 2. Active 3. Finished 4. Excluded <p>If the node that did not start in time was excluded, the check will return True only for the excluded, and False for the not started.</p>
IO state	<p>Check the Current States of a given interactive object.</p> <p>After selecting the target, an array appears with the option to add a new element, within which you can select the following Current States:</p> <ul style="list-style-type: none"> • Enable • Unbreakable • Broken • SelfSufficient • PrerequisiteRequired <p>Unique for doors:</p> <ul style="list-style-type: none"> • IsLocked <p>Unique for scanners:</p> <ul style="list-style-type: none"> • Inactive • Active • Done • Collected <p>Unique for light bulbs:</p> <ul style="list-style-type: none"> • Flicker • AutoLightOnNight <p>Unique for beds:</p> <ul style="list-style-type: none"> • Is Interaction active <p>You can select True or False for each Current State. If the object's Current State matches the one specified in the conditions, the node returns True. In all other cases, it returns False.</p>

Set Search Point

This node sets the Search Points logic on the selected object or NPC.

Node exclusion deactivates Search Point.

How to use

Parameter	Description
Target	The object on which the search point will be set
Use Quest Target	Checkbox that activates the use of the Quest Target field. Selecting this checkbox makes the Target field inactive.
Quest Target	<p>The quest object on which the search point will be set.</p> <p>If the target is an NPC, the search point will move with the NPC. If the NPC dies, the search point will be set on their corpse.</p> <p>If the target is a squad, the search point will be set on the squad leader.</p>

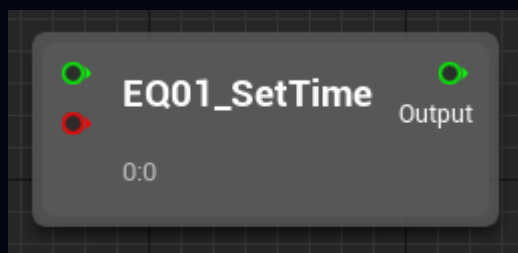
Set Time

This node sets the time in the game. The time can change instantly in one frame or gradually rewind to the set time.

If the difference between the current time and the target time is less than 30 minutes (counted downwards, for example, from 20:29 to 20:00), the time will not be smoothly rewind.

The node does not change the weather.

The node ends after the time is rewound to the selected value.



▼ Time Quest Node

Hours	<input type="text" value="0"/>
Minutes	<input type="text" value="0"/>
Use Transition Time	☐
Transition Time	<input type="text" value="0.0"/>

How to use

Parameter	Description
Hours	The time value that will be set by the node. Values 0-23 are available.
Minutes	The minute value that will be set by the node. Values 0-59 are available.
Use Transition Time	<p>The parameter responsible for the smoothness of the rewind. If False, the time is rewound instantly, in a single frame. If True, the time is rewound smoothly.</p> <p>Smooth time rewinding is not recommended in 99.9% of cases. Use this parameter with extreme caution. Smooth rewinding can potentially cause bugs.</p> <p>Rewinding will not work if the time difference between the current time and the rewind target is >23h 30m. In this case, rewinding will occur in a single frame.</p>
Transition Time	<p>The value in seconds for which time rewinding occurs when Use Transition Time is True.</p> <p>It is better not to use values below 1 and above 60 seconds, as weather changes are blocked during the transition time. This can create bugs or blocking situations.</p>

Please use the node only when absolutely necessary and in very moderate amounts. Do not forget that changing the time does not change the weather.

The node does not affect immersive mechanics like hunger and sleep. If the scripter needs to change these parameters, the Set Character Param node must be used.

Also, the node does not affect inputs and AI in any way.

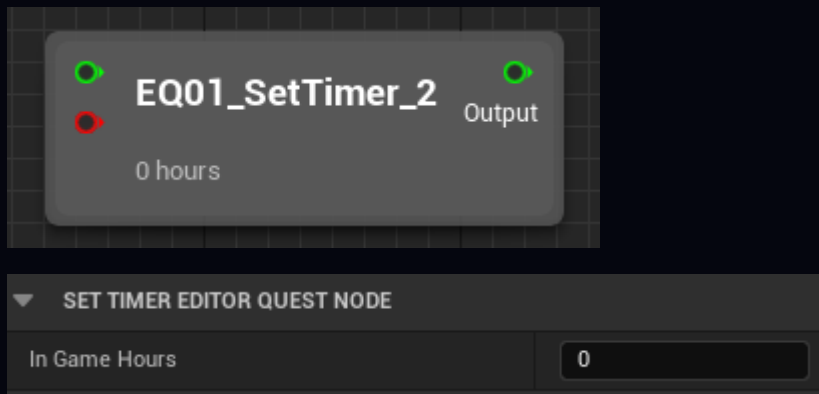
Set Timer

This node sends a signal after a certain amount of game time has passed.

If an exclusion signal enters the node while the timer is running, the node will immediately exclude and will not send the signal further.

IMPORTANT

To avoid bugs, do not use nodes with timers.



How to use

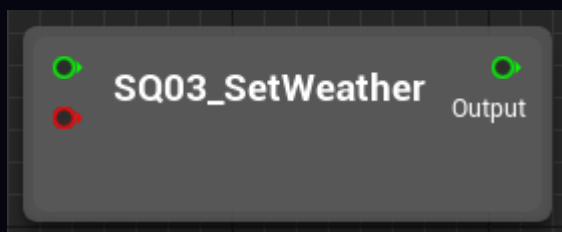
Parameter	Description
In Game Hours	<p>The number of game hours after which the node will send the signal further and end.</p> <p>The node considers all possible ways of skipping or accelerating in-game time.</p> <p>Please keep in mind that game time passes more slowly during the day than at night. This means that to complete a node on a timer at night, you need 10 real minutes, and 15 during the day (figures are for example only).</p>

Set Weather

This node changes the weather as if it was changed by the Weather controller. This means that the weather change is smooth and completely systematic.

When triggered, the node prevents the Weather controller from changing the weather. Node exclusion allows the Weather controller to change the weather.

In the node settings, you can change the transition time from one weather to another.



▼ Weather Quest Node

Weather

Clearly ▼

▼ Output Pins

Clearly

▶ Output Pins

Cloudy

▼ Input Pins

Foggy

▼ Input Pins

Stormy

▼ Index [0]

Light Rainy

▼

Rainy

▼

Thundery

▼

Emission

▼

Calm Before Emission

How to use

Parameter	Description
-----------	-------------

Weather	<p>The weather to which the transition will be made:</p> <p>Clearly</p> <p>Cloudy</p> <p>Foggy</p> <p>Stormy</p> <p>Light Rainy</p> <p>Rainy</p> <p>Thundary</p> <p>Emission</p> <p>Calm Before Emission</p>
Force weather	<p>Bool</p> <p>Instantly switches the weather to the selected one</p>
Use Custom Transition Time	<p>Bool</p> <p>Allows activating a custom transition time between weather conditions</p>
Custom Transition Time	<p>Custom transition time between weather conditions. May cause artifacts, please use with caution!</p>

Set Wounded

This quest node gives the selected character the Wounded state.

When the node is triggered, its target will curl up on the ground, and the target's HP will become 10. The regeneration of the selected character will become zero.

There is a Healing Type setting in the node parameters.

If Hold Component is selected, the character can be healed by holding the interaction button (F) near the character.

If Dialog is selected, the character can be healed through a dialog detected using the Set Dialog node with the Override Defeat Topic or Ignore Global WFR parameters. In this case, the dialog must include the Dialog Action Give medkit.

If Quest is selected, the character can only be healed by another SetWounded node with **IsWoundedOn** set to **False**.

If the node target is the Squad node, all squad members will lie down as wounded.

The node can also heal the selected character if the corresponding setting is activated. In this case, the node's target will exit the Wounded State, its HP will rise to 100%, and regeneration will return to normal.

Exclusion of the node that has already been executed will not cancel its effect.

The node ends immediately after its target has reached the wounded state.

If the node target has not yet loaded/spawned, the node will remain active until the character appears/spawns on the world map.

Use case

A narrative designer needs the character, who was captured according to the quest storyline, to curl up on the ground.

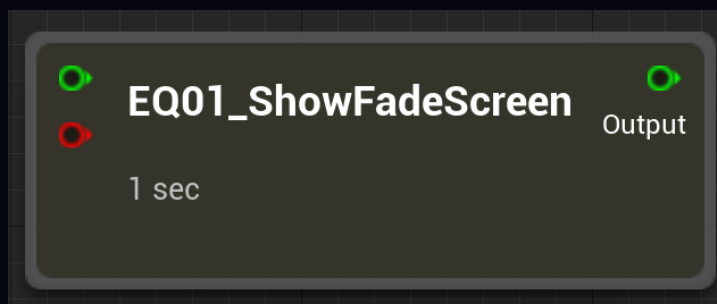
How to use

1. Create the node.
2. Select the target.
3. Configure Heal With Dialog.
4. Save the changes.

Show Fade Screen

This node displays a text or an image on the screen. The node ends when the configured timer expires. Node exclusion immediately interrupts the fade.

References



FADE QUEST NODE	
Fade Time	1.0
Screen Text	
Image	<div>None</div> <div>None</div> <div>←</div> <div>🔍</div>

Use case

- A narrative designer decided to use a game design feature that is not yet ready. The node can be used at least to demonstrate the quest idea.
- A narrative designer needs to show the scenery of the game, but the art is not ready in the engine. However, the concept of this place exists. The Show Fade node can play this concept to show what the narrative designer meant.

How to use

1. Create the node.
2. In **Screen Text**, write the text you want to display. If necessary, select the desired image in **Image**.
3. Set the duration in seconds in **Fade Time**.
4. Save the changes.

Show Marker

This node changes the status of a location marker on the map.

Each marker has its own configuration, and to configure a specific marker, select the appropriate one in the node.

A marker can have one of these statuses:

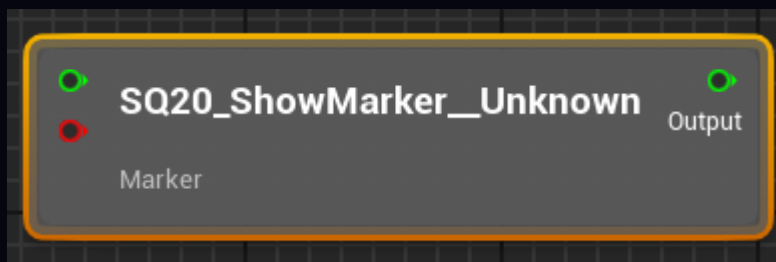
1. Default – the marker is not displayed on the map
2. Discovered – the marker is displayed as a question mark
3. Explored – the marker is displayed as explored

The status of markers can only change in the following order: Default > Discovered > Explored. Attempting to lower the status of awareness of a marker will not change it.

The node ends immediately after changing the status of the marker.

Exclusion of a completed node will not cancel its effect.

Node view



How to use

Parameter	Description
-----------	-------------

Marker SID	Marker SID, whose status will be changed
Explored	<p>Checkbox responsible for whether the marker status will be changed to Explored.</p> <p>Without selecting this checkbox, the marker will be changed to the Discovered status, and the location will be displayed as a question mark.</p>

If a character briefly mentions the location, it is better to display the marker in the Discovered status. If a character sends the player to a specific location, providing contextual information about it, the marker should be changed to Explored.

If the player does not have open information about the region, opening the location marker from that region in the Explored status also opens the marker for the region in which that location is.

Spawn Object NPC Monster

This node spawns NPCs, Squads, Items, Stashes, etc., into the world. If an exclusion signal enters an inactive node, the node will be excluded and will not work.

If an exclusion signal enters an active node, it will behave according to the configured scenario described in How to use.

The node ends in the following cases:

- Target NPC, squad, or mutant is dead
- Target Item – picked up by the player or NPC inventory
- Target Container – emptied

References

EQ01_SpawnObjectNPCMonster_not_set

Output

Spawn target: not_set!

None

▼ Spawned

Target

None

▼

🔍

📍

Ignore Damage Type

None

▼

▼ Spawned	
Target	BP_Squad_Mercs ▼ 🔍 ↗
Ignore Damage Type	None ▼
Spawn Hidden	☐
Exclude Type	Seamless Despawn ▼

Use case

- The quest requires an NPC.
- The quest requires a stash with loot.
- The quest requires hiding an artifact.

How to use

1. Create the node.
2. Place one of BP_Instance_Placeholder that needs to be spawned on the World Map:
 - BP_AnomalySpawnerPlaceholder – anomalous field or single anomaly
 - BP_ArtifactSpawnerPlaceholder - artifact that works according to the rules of the world and is only visible with a detector
 - BP_DeadBodyPlaceholder - NPC corpse
 - BP_ItemContainerPlaceholder - container in which you can put items
 - BP_ItemPlaceholder - single item
 - BP_NPCPlaceholder - NPC or mutant
 - BP_SquadPlaceholder - Squad of NPCs or mutants
3. Set selected BP_Instance_Placeholder in Spawn Target
4. If the target can be killed (NPC, squad, mutant), select **Ignore Damage Type**:
 - None – can be killed and wounded
 - Invulnerable – cannot be killed or wounded
 - Unkillable – cannot be killed, but can be wounded
5. Set **Spawn Hidden** if the selected object allows it and it is necessary.
6. Configure **Exclude Type** if the selected object allows it:
 - Seamless despawn – seamless despawn when the player cannot see the Target
 - Force despawn – forced unloading of the Target
 - Kill spawned obj – kill the Target if it is killable (NPC, squad)
7. Save the changes.

Switch Quest Item State

This node allows a narrative designer to change the Quest Item flag for the item prototype, setting or removing it depending on the node settings.

Available settings:

- **Item SID**, the list of which is formed from the contents of QuestItemPrototype.cfg.
- **IsQuestItem** checkbox, which determines whether the flag will be set or cleared. Set to **False** by default

If the flag is already set to the appropriate position, the node does nothing.

Use case

A narrative designer uses a PDA quest item in their quest, which is not used further in the quest. To dispose of the item, the narrative designer clears **IsQuestItem** for the item so that the player can sell or discard it after completing the quest.

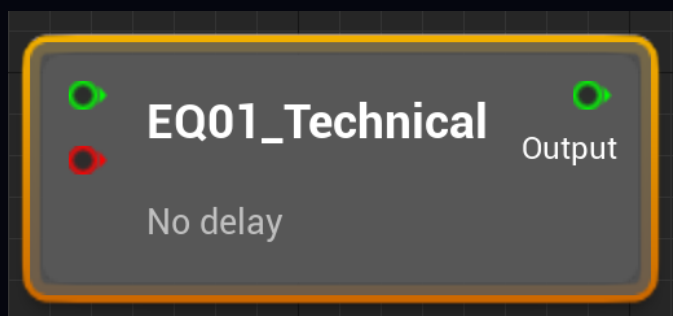
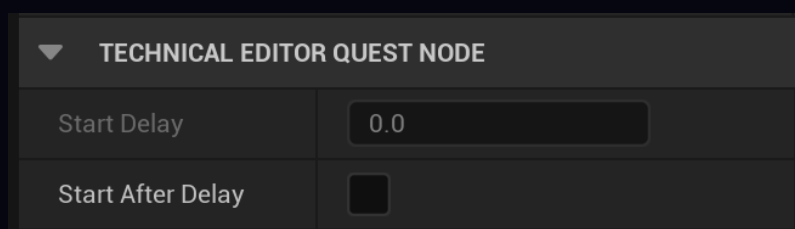
How to use

A narrative designer finds the desired item in the selector and chooses whether to add or remove the flag.

Technical

This node is designed to improve scripting comfort, reduce connection clutter, increase readability, and send a signal after a configured pause. If an exclusion signal is received by the node during the delay, the node will be excluded and won't send the signal further.

References



Use case

- A narrative designer needs to combine several strings into one node. A technical node is ideal for this.

- A narrative designer needs to trigger the signal after a 60-second pause. To do this, they use this node.

How to use

1. Create the node.
2. Set up a delay before sending the signal.
3. Save the changes.

Teleport Character

This node moves the selected character to a specified point. It can also turn the character in a specified direction during teleportation.

The node ends after teleportation is complete (the teleport fade-out ends).

Node view

EQ01_TeleportCharacter_Player

Output

Target: Player

▼ Character Targeted Action			
▼ Target Location and Rotation			
Target Location and Rotation	None ▼ [icon] [icon]		
▼ Target Location Value	0.0	0.0	0.0
X	0.0		
Y	0.0		
Z	0.0		
▼ Target Rotation Value	0.0	0.0	0.0
X	0.0		
Y	0.0		
Z	0.0		
Teleport Type	None ▼		
▼ Target Placeholder			
Target	None ▼ [icon] [icon]		
Target Player	☒		

How to use

Parameter	Description
-----------	-------------

Target Location and Rotation	<p>The point to which the character will be teleported.</p> <p>The character's face direction will be similar to the direction of the actor to whom the teleportation point is assigned.</p>
Teleport type	<p>The type of teleportation used by the node:</p> <ul style="list-style-type: none"> • None – teleportation without effects, use only for teleporting NPCs or objects. • Classic – player teleportation with a loading screen. For example, it can be used for guider mechanics. • Immersive – player teleport without a loading screen, but with post-processing and sound identical to teleportation through the teleport anomaly. It includes the effects and sound of entering the teleport, the looped part, and exiting the teleport. <ul style="list-style-type: none"> ◦ Quest – a subtype of Immersive, made specifically for boss fights and a cutscene with Faust. It features unique post-process visuals. • Instant (not yet implemented) – instant teleportation without post-processing or sound, should look seamless. Used for teleport huts and other situations where teleportation should be invisible to the player. <p>Please do not use for NPC or Item teleportation any type other than None.</p>
Target Placeholder	The character that will be teleported

IMPORTANT

Please use BP_TeleportPlaceholder for the teleportation point.

Time Lock

This node freezes time.

As soon as the node is activated, time stops. Along with time, the system weather change also stops. The Set Weather node can still change the weather.

The Set Time node can also set the time, but its flow will still be blocked.

The flow of time will be unblocked after the node is excluded.

Use case

A narrative designer needs to lock the time in the prologue in case it takes the player too long to explore the prologue passage.

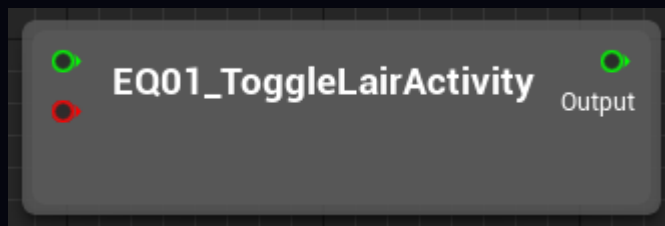
How to use

1. Create the node.
2. Set up an exclusion for the node.
3. Save the changes.

Toggle Lair Activity

This node allows controlling the selected lair

References



TOGGLE LAIR ACTIVITY EDITOR QUEST NODE	
▼ Target Lair	
Target	None ▼ 🔍 ↗
Force Despawn	☐
Activate	☐

Use case

- After the quest, a narrative designer needs to activate the mutant lair in its place so that the location is not empty.
- Before the quest began, there was a lair in its place, and a narrative designer needs to remove it so that it does not interfere with the quest.

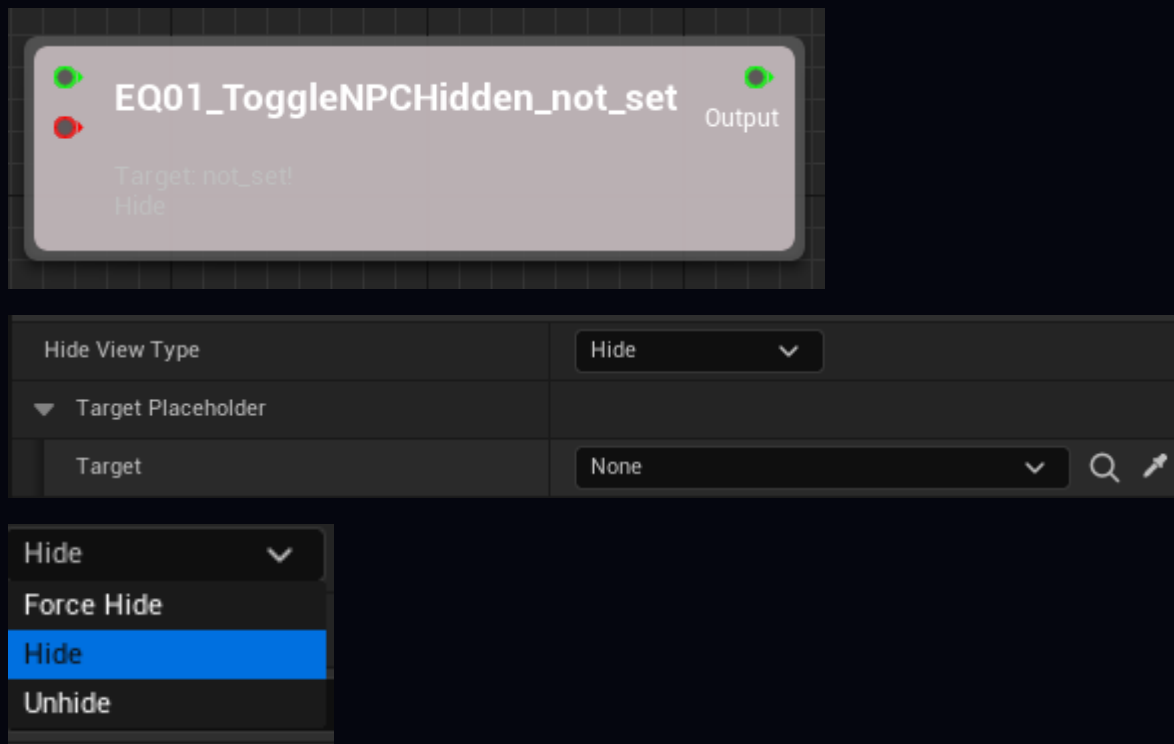
How to use

1. Create the node.
2. Select **Target Lair**.
3. Configure the type of interaction with the lair:
 - a. **Force Despawn** – despawn the lair
 - b. **Activate** – activate the lair
4. Save the changes.

Toggle NPC Hidden

This node allows hiding or unhiding a spawned NPC.

References



Use case

A narrative designer has come up with a quest where the player needs to kill a mutant that only hunts at night. To do this, they can use the Toggle NPC Hidden node.

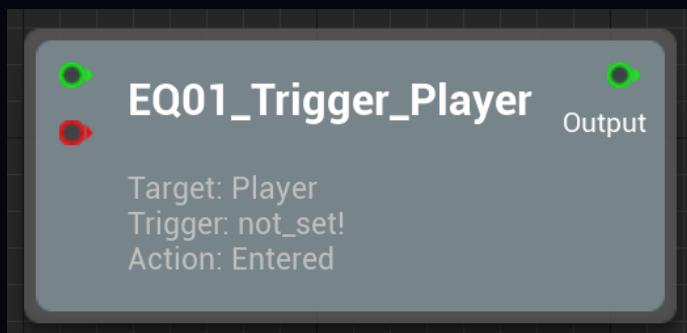
How to use

1. Create the node.
2. Select the desired NPC in **Target Placeholder**.
3. Select the desired Hide View Type:
 - a. Hide – hide the Target when the player cannot see it
 - b. Force Hide – instantly hide the Target
 - c. Unhide – show the character
4. Save the changes.

Trigger

This node sends a signal after the selected character interacts with the selected trigger. The character can be the player, NPC, dead body, item, squad, etc. The node can be triggered both when entering the trigger and after exiting it.

References



CHARACTER TARGETED ACTION	
▼ Trigger	
Trigger	None ▼ 🔍 ✎
Action	Entered ▼
Triggers by Anybody	☐
▼ Target Placeholder	
Target	None ▼ 🔍 ✎
Target Player	☒
Required Squad Members	Any Member ▼
React Type	React on Alive ▼

Entered ▼

Entered

 Exited

Use case

- A narrative designer wants the player to hear a scary sound when entering the building. They can track whether the player has entered the building with the Trigger node.
- A narrative designer wants the NPC to say a comment when passing by the Bazaar within Rostok. To do this, they can use a trigger node configured for this NPC.

How to use

1. Create the node.
2. In the **Trigger** field, select the desired trigger to which the node will respond. The trigger will only work if it completely covers the player or NPC capsule in height.
3. In the **Action** field, select whether the node will work when the character enters the trigger or when they exit it.
4. In the **Target Placeholder** field, select the character whose interaction the node will track.
 - a. If a squad is selected as the target, select in **Require Squad Members** how the node will work: when the entire squad interacts, or when any of its members interact.

- b. If the target is a Player, select the **Target Player** checkbox.
 - c. If the target is any character interacting with the trigger, select the **Triggers by Anybody** checkbox.
 - d. Configure **React Type** if the character interacting with the trigger may be dead at that moment.
5. Save the changes.

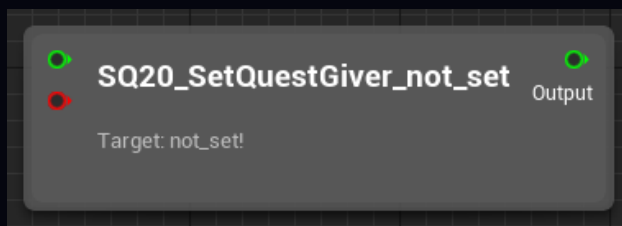
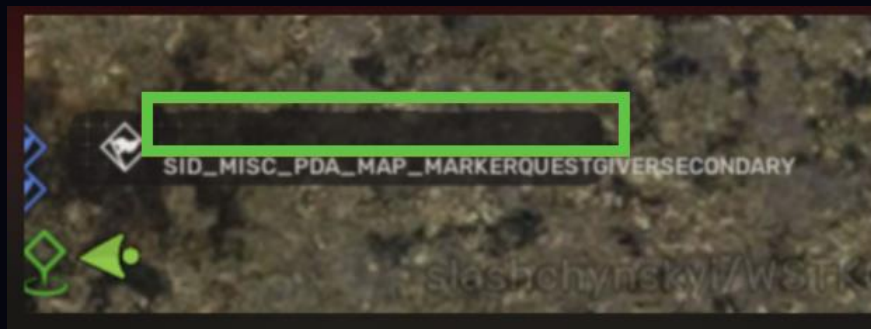
Set Quest Giver

A node that places a Quest Giver marker on an NPC. The marker displays the name of the character selected in **Target Placeholder**. If it is a squad, the marker displays the name of the squad leader. If there are two characters in the squad, names of both characters are displayed.

In addition, the node target can be any placeholder, blueprint, or empty actor. The marker will follow this object.

The marker starts displaying once the player gets as close as 5000 units to the target.

Node exclusion disables the marker.



How to use

Parameter	Description
-----------	-------------

Main Quest	Parameter responsible for marker type: main or side -quest
Target Placeholder	<p>The in-game object to mark</p> <p>If it is a squad, the marker displays the name of the squad leader. If there are 2 characters in the squad, the names of both characters are displayed.</p>

Activate Data Layer Combination

A quest node that activates a specific combination of enabled and disabled Data Layers. Which layers will be enabled and which will be disabled is determined in the combination settings.

The node replaces the outdated Enable Data Layer node.

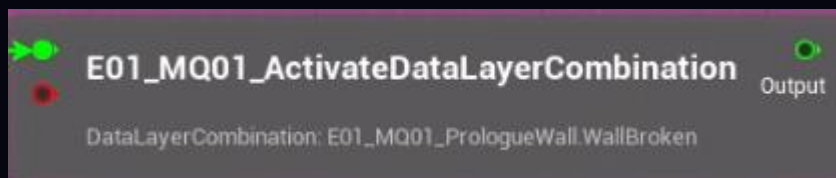
Where to get Data Layer Combinations

You can request a new Data Layer Combination from a Level Designer via a Jira task.

NOTE

Combinations must be ordered at the stage when it is clear exactly which combinations are needed.

Node view



▼ Default

Show Referencing Bridge Nodes

▼ Activate Data Layer Combination Editor Quest Node

▼ Combination Handle

Group

E01_MQ01_PrologueWall

▼

Combination

WallBroken

▼

▼ Output Pins

▶ Output Pins

1 Array elements

⊕

🗑

↶

▼ Input Pins

▼ Input Pins

2 Array elements

⊕

🗑

↶

▶ Index [0]

2 members

▼

↶

▼ Index [1]

2 members

▼

↶

▶ Name

ExcludingInput

↶

▶ Excluding

☒

↶

▼ Repeatable

Repeatable

☐

▼ Node Settings

Title

ActivateDataLayerCombination

📄

↶

▶ Border Color

How to use

Parameter	Description
-----------	-------------

Group	A group of data layers combination, a logical way to organize combinations. For example, all data layer combinations used in E01_MQ01 will be in a group named E01_MQ01_GroupName.
Combination	A specific data layers combination that should be activated by the node.

Broken Game Data Validator

Description

Broken Game Data Validator is a tool that allows you to easily identify errors in configuration files used in quests. It validates configurations when the scripter launches the play mode.

It displays Broken Game Data via the standard UE Message Log after closing the play mode.

In case of crashes and “detective debugging”, the tool creates text files with logs of broken quest assets in these folders: Z:\git\s2gamebuild\Saved\QuestBrokenGameData (errors from quest BP) and Z:\git\s2gamebuild\Saved\BrokenGameData (errors from placeholders and GD configs).

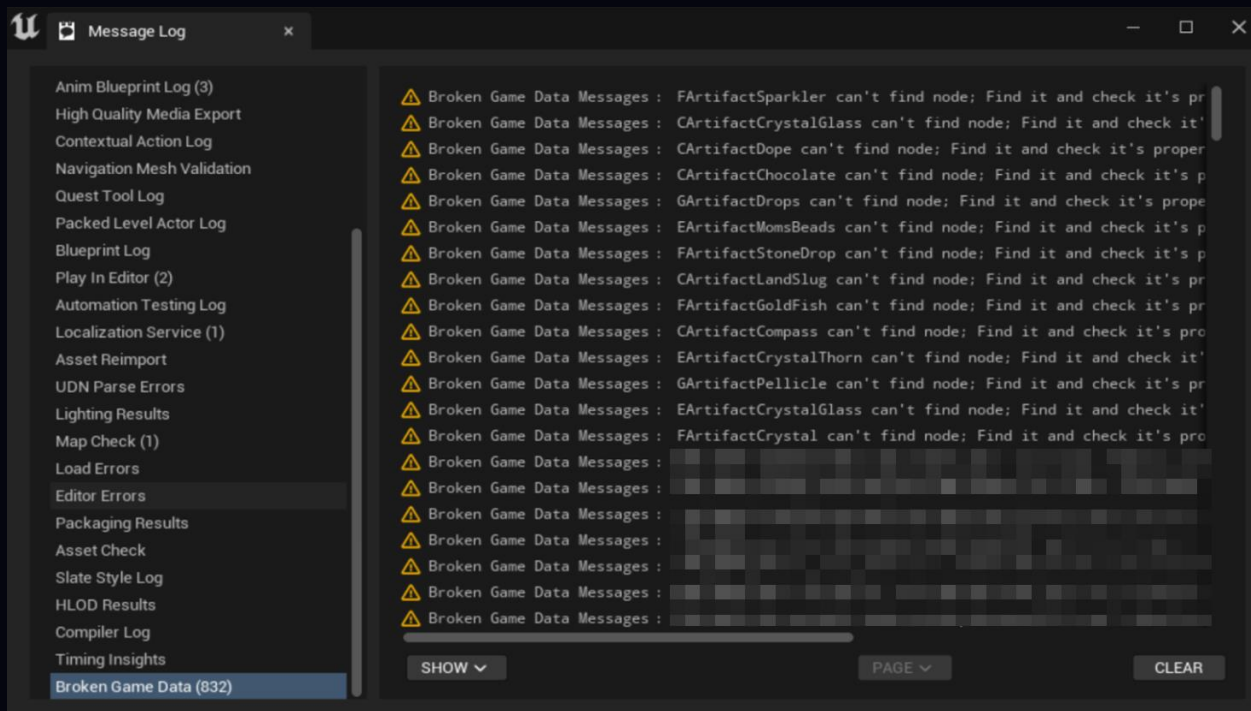
Broken Game Data Validator points to the problematic object in this text file in the *errortype EQXX_NodeName* format, e.g., Invalid trigger placeholder set for trigger node EQ74_OWL_OnTriggerEnterEvent.

What is Broken Game Data?

To put it simply, it is a reference to an object that has been deleted, damaged, or radically reworked after the quest was scripted. Example:

- A quest was created, in which the player receives SMG Gaduka as a reward. At the time of working on the quest, such an item existed in the configs, but after two months, it was deleted by a game designer for some reason. In this case, each time the quest attempts to refer to the SMG Gaduka item, the broken game data validator will create a log indicating the nodes that refer to this asset and the issue itself.
- A quest has a spawn node that spawns a squad. At the time of scripting, the squad existed on the level instance, but was later removed. Each time an attempt is made to reference this squad, the broken game data validator will create a log indicating the nodes that reference this asset and the issue itself.

References



Use case

- Search for assets that cause crashes, ensures, and freezes.
- Quest validation after a long downtime.
- Quest debug during refactoring.

How to use

1. Start the engine.
2. Start PIE.
3. Start the required quest.
4. Test the quest until the end or until it crashes:
 - a. If no crash occurred:
 - i. Exit PIE.
 - ii. In the window that opens, find the **Broken Game Data** tab and search for the quest being tested.
 - iii. If you can't find it, follow the instructions as if your engine crashed
 - b. If the engine crashed:
 - i. Minimize the engine window and go to `Z:\git\s2gamebuild\Saved\BrokenGameData`.
 - ii. Sort the folders by date and open the one that was created last.
 - iii. Sort the files by date and open the last one in any text editor.

- iv. Find the game data errors in the file. You can use the search function (Ctrl+F) and enter the quest blueprint name of the tested quest in the search field.
5. If there are errors, memorize the name of the node and correct the errors:
 - a. Go to the quest blueprint.
 - b. Find the problematic node.
 - c. Correct (reconfigure) the target/seed/content that the validator has issues with. For a more detailed guide on fixing errors, see the instructions below.
 - i. If content that you consider critically important has been deleted, consult with your colleagues, and if they agree with your concerns, report the incident to the tech lead.
 - d. Save the blueprint.
6. Go to step 3 and repeat all procedures until the broken game data from your quest is completely gone.

Fixing Broken Game Data in the most common cases

After any fix, save QBP. If you changed the Level Instance, save and commit it too.

1. Problem in the Spawn node:
 - a. Check if the spawn target is selected in it. If not, select it.
 - b. Check if the spawn target's SID is selected. If not, select and save the Level Instance and QBP.
2. Problem in Item Add/Remove nodes or condition on Item:
 - a. Check if Item Type is selected in the node. If not, select it and save QBP.
3. Problem in Change Faction/Relationships nodes and others where faction can be selected in the target:
 - a. Check if Faction Type is selected in the node. If not, select it.
4. Problem in the Set Journal node:
 - a. Check if Journal Asset is selected. If not, select it.
 - b. If quest markers are set in the node, check if the corresponding target is set. If not, fix it.
5. Problem in the Play Sound node:
 - a. Check if Sound Asset is set. If not, fix it.
6. Problem in the Trigger node:
 - a. Check if the correct trigger is selected. If not, correct it.

7. Problem in the Set AI Behaviour node:

- a. Check if NPC/Squad is selected. If not, select it.
- b. Check if Behaviour is selected. If not, select it.
 - i. If Behaviour requires a target location, be sure to set it.
- c. Check if the animation for behavior is selected. If not, set it.

Solving some problems with real-life examples:

1. Solving Broken Game data in Item Add Node with an example:

- a. Node error: EQ88_ItemAdd_ArtefactToSpichka is trying to add an empty item. Please set a valid item and resave that node.
- b. Problem: in the Item Add node from the EQ88 blueprint, the item is not selected, most likely because it was forgotten to be configured.
- c. Solution: open EQ88, find the ItemAdd_ArtefactToSpichka node, and configure the Item in it.

2. Solution for Broken Game data in Spawn Node using an example:

- a. Node error: SmallZone_SQ_L_SpawnObject_AutoRifle node links on empty(placeholder) set valid placeholder and resave BP.
- b. Problem: the target item placeholder has disappeared from the SpawnObject_AutoRifle node in the SmallZone_SQ_L blueprint. Someone may have accidentally deleted the placeholder or simply forgotten to configure it.
- c. Solution: open SmallZone_SQ_L, find this node, configure the target placeholder in it by selecting from existing ones on the map, or create a new one, remembering to set the required prototype sid. Save QBP.

3. Solution for Broken Game data in the Condition node with an example:

- a. Node error: EQ01_ConditionArtifact node can't find SID: GArtifactChunkMeat. Try to check the selected Item SID validity. : Count : 1
- b. Problem: in the Condition node in the EQ01 blueprint, the check for the GArtifactChunkMeat artifact was set, but something happened to it. Possibly it was renamed or deleted. Now the engine cannot find its prototype.
- c. Solution: open EQ01, find the ConditionArtifact node, and replace the check from GArtifactChunkMeat to the current artifact prototype.

There may be other cases, but the typical algorithm for all nodes of all types is approximately as follows:

1. Check Target Item/Location/NPC/Squad. If it crashed, reset it.
2. Check the asset that the node works with (Sound, Animation, Note, etc.). If it crashed, reset it.

3. For prevention, select **Reassign Pointers** and resave QBP.

Broken Game Data of deleted nodes

Sometimes, when nodes are deleted from QBP, they remain in the configuration of that QBP, and Game Data Validator warns about this with a message like this:

EQ13_Condition_KilledNPC can't find a node; Find it and check its properties, then resave it manually.

What to do

1. Go to the broken QBP – in our case, it's EQ13.
2. Move any node literally by one pixel.
3. Save the QBP and check if the Broken Game Data issue has been resolved.

What to do if the problem remains

You can solve it by manually deleting information about these nodes from the QBP config:

1. Close the engine.
2. Go to Z:\git\s2gamebuild\Content\GameLite\GameData\QuestNodePrototypes.
3. Use the search to find the config for the QBP you need, e.g., EQ13.
4. Open the file using Notepad++ or any other text editor.
5. Use the search to find the node that has not been completely deleted, e.g., EQ13_Condition_KilledNPC.
6. Manually delete all lines from the node name in the EQXX_NodeName : struct.begin (EQ13_Condition_KilledNPC : struct.begin) format to the last struct.end line, the line after which contains the name of another node in the same format.
7. Save the file.
8. Start the engine.
9. Retest the quest and make sure there are no errors.

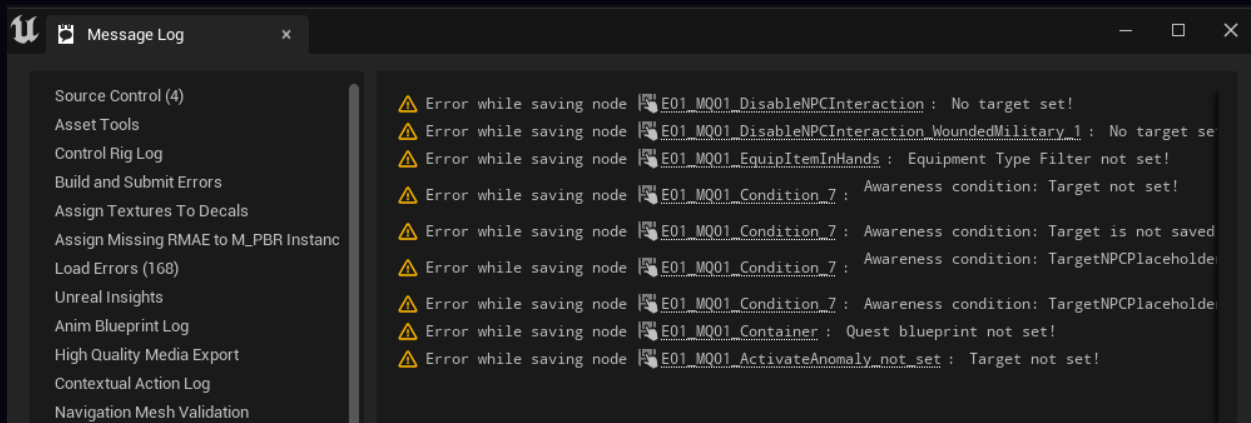
Quest Tool Validator

Description

A Quest Tool Validator launches when the quest blueprint is saved and displays a list of errors with links to the problematic nodes and an explanation of what the error is.

References

Message log example



Use case

A user creates a quest blueprint with incorrect data and saves the blueprint. As a result, a window appears with links to all problematic nodes and a brief description of issues.

How to use

Follow these steps:

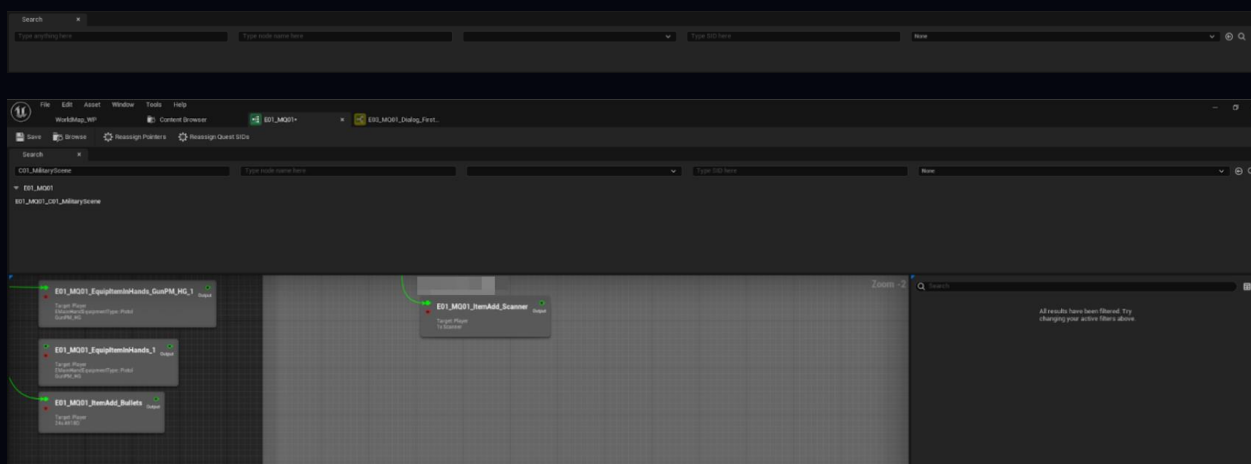
1. Do the work.
2. Allow an error.
3. Save the changes.
4. See issues.
5. Fix issues.

Quest blueprint search

Description

A quest blueprint search is a tool for searching all quest blueprints. Allows you to find nodes by specified parameters like name, node type, etc. among all quest blueprints.

References



Use case

- A user wants to find a node by its name. They enter the node name, possibly not in full, and receive a list of all nodes that match the name.
- A user wants to see all nodes of a certain type. They select the appropriate node from the drop-down menu and receive a list of all nodes of that type.
- A user wants to find a node that uses a specific sid. They enter the sid text in the field and receive all nodes that refer to it.
- A user wants to find a node in a specific blueprint. They select a specific blueprint and receive all nodes that are in that blueprint.
- A user wants to find a node with specific parameters. They enter several parameters at once, e.g., name and type, after which they receive a list containing only nodes that match these conditions.

How to use

Follow these steps:

1. In the Window tab, find the **Search** window and enable it if it is not already enabled.
2. Fill in the available fields.
3. Press **Enter**.
4. Enlarge the **Search** window by dragging the edge of the window if the search results do not fit in it.
5. View the results.
6. Select any result.
7. See the node found by the search.

Runtime quest reload

Description

A runtime quest reload allows you to add, delete, and change QuestTool node settings during playtesting and reload the quest blueprint with the changes applied.

The PIE loading process takes a significant amount of time. This system helps narrative specialists increase the speed of making changes.

Operation logic

There are four operations that you can perform at runtime:

1. Reload the quest configuration while preserving the state of the quest nodes at the time of reloading.
2. Add a node and connect it. If the node joins nodes that have already worked during playtesting, it will work according to the usual rules when restarted, i.e., as if the nodes that have already worked sent a signal to it.

3. Change node parameters, including adding links to characters. When changing a node parameter and reloading the blueprint, for previously existing nodes, it will work as if the node with the previous settings was excluded, and a new node worked instead.

NOTE

For Spawn nodes, the spawned entity is always destroyed as a force despawn in order to immediately respawn a new entity.

Example

You change the Spawn node by activating its Spawn Hidden option. When the blueprint is reloaded, this will cause the NPC spawned by this node to despawn, after which it will spawn again, but hidden. It will spawn according to its placeholder settings and in the same position.

4. Delete nodes. For nodes that have already worked or that have not yet worked when reloading, this will not cause any changes in the world. This will be equivalent to an exclusion for nodes that are currently active.

Example

You delete an activated Spawn node with the Kill on Exclude=true setting. This causes the node to disappear from the blueprint, and the character it spawned dies.

Quest Tool workflow for audio

Introduction

This document aims to provide a clear and informative pipeline for efficient work with existing Quest Tool within the audio function. It defines internal logic creation rules, usage standards for the tool, and explores various use cases relevant to the development process.

Specifically, the document covers:

- The structure and working principles of Quest Tool in the context of audio integration.
- Unified standards and rules for logic creation to ensure consistency and maintainability.
- Practical usage scenarios, including sound event integration, working with triggers, conditional logic links, and dynamic audio changes.
- Workflow optimization to improve team productivity and reduce the risk of errors.

This document serves as a primary guide to help both beginners and experienced sound designers ensure proper and efficient use of Quest Tool in creating in-game audio scenarios.

What is Quest Tool, and how does it interact with audio

Quest Tool is a custom visual scripting system (blueprints) built in Unreal Engine 5.1.1, specifically developed for GSC Game World projects. It is used by narrative designers to create quest logic, allowing rapid implementation of gameplay scenarios without deep programming knowledge.

However, Quest Tool's capabilities go beyond just gameplay logic – it also has built-in support for audio integration. The system allows triggering audio events according to predefined scenarios using specialized functional nodes, which can be used for:

- Playing sounds in specific in-game situations.
- Triggering sounds based on internal conditions within quest logic (e.g., events, NPC states, object interactions).
- Combining audio with other gameplay systems through Quest Tool's shared mechanisms.

NOTE

Quest Tool is a flexible, actively supported instrument by the dev team. This means its audio functionality can be expanded in the future, depending on project needs and requests from the sound design team.

Core principles of audio integration

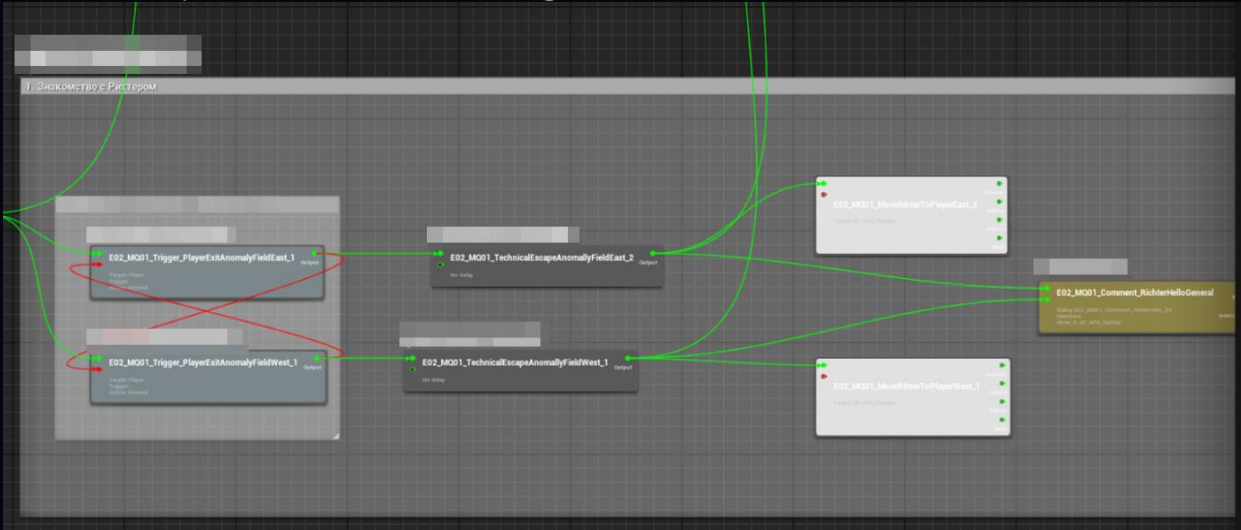
The main principle of audio integration is that all audio logic must be clear, well-structured, and accessible to all team members who are currently working on it or may need to work with it in the future. To achieve this, it is essential to follow the "best practices" of quest logic design.

Key requirements for visual organization of logic

- Readability and neatness
Nodes should be logically ordered and not placed randomly or chaotically.
- Clear connections
Avoid intersecting or tangled wires that make the logic hard to follow.
- Comments and labels
Always use comment blocks and label each node to explain its function.

Examples:

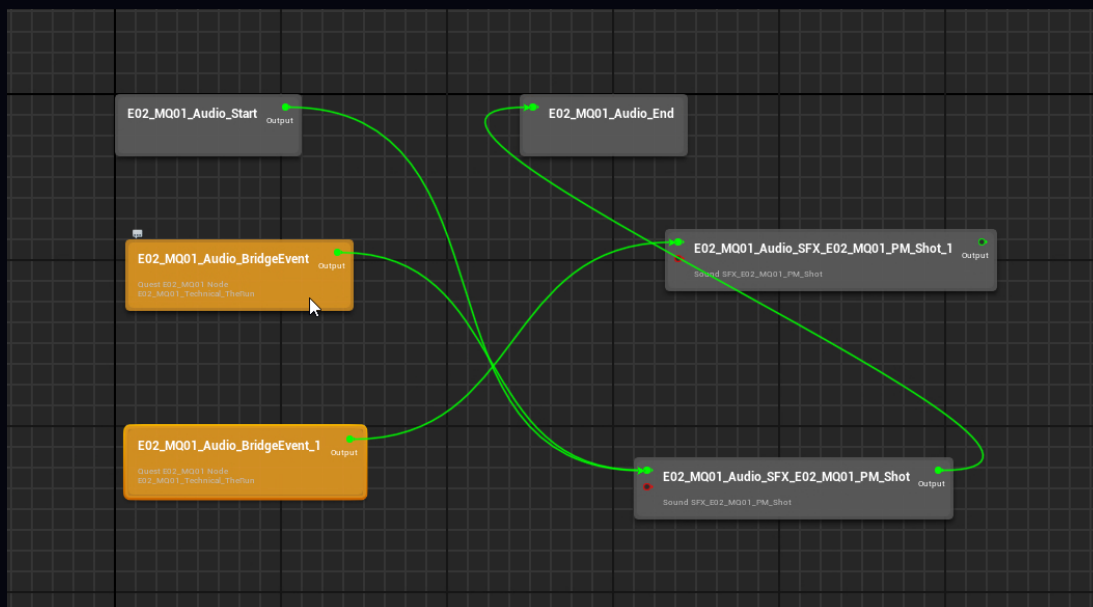
A visually clean structure with neatly arranged nodes, no overlapping wires, and clearly labeled components is considered a good standard.



Example of a poorly readable structure:

- Chaotic node placement
- Confusing or overlapping connections
- No comments or explanations

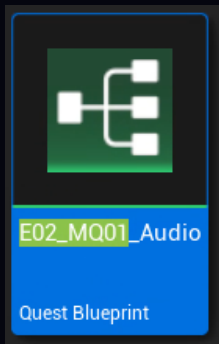
Such setups significantly reduce clarity and make future support or collaboration difficult. Please make sure to avoid this style when integrating audio logic.



Organizing audio logic in Quest Tool

Each quest's audio logic should be placed in its own separate Quest Blueprint, and its name must clearly indicate that it belongs to the audio logic.

The naming convention uses the suffix `_Audio`, for example:



- This naming convention makes it clear that the asset belongs to the quest's audio logic and can be edited, saved, and committed without prior approval.
- All other Quest Blueprints that do not contain `_Audio` in their name must not be modified or committed without the quest owner's approval.

Rules for working with Quest Blueprints

- Audio logic must have a clear structure, closely tied to the main quest logic and relevant only to the specific quest it supports.

Avoiding merge conflicts

- Work on a single Quest Blueprint sequentially – one person at a time.
- Do not start making changes until the previous contributor has merged their changes into the production branch.

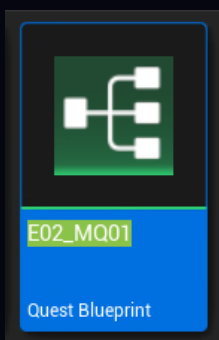
These rules establish unified working standards for the audio department, helping to avoid errors, improve collaboration between teams, maintain a structured and consistent integration of audio into the game's quest system.

Basics of working with audio in Quest Tool

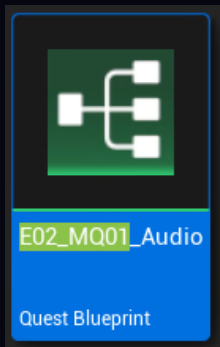
Assets and their structure in the project

To work with quest logic in Unreal Engine, a special type of asset is used – Quest Blueprints. These assets form the core gameplay logic developed by narrative designers.

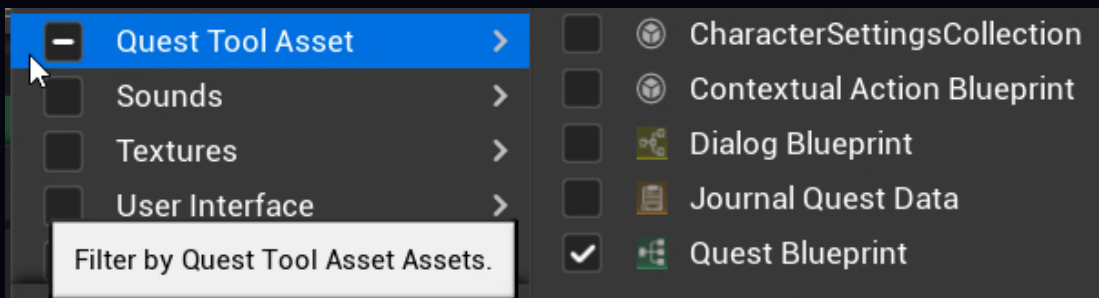
Each Quest Blueprint is named according to the quest it belongs to, reflecting the quest's title in its name.



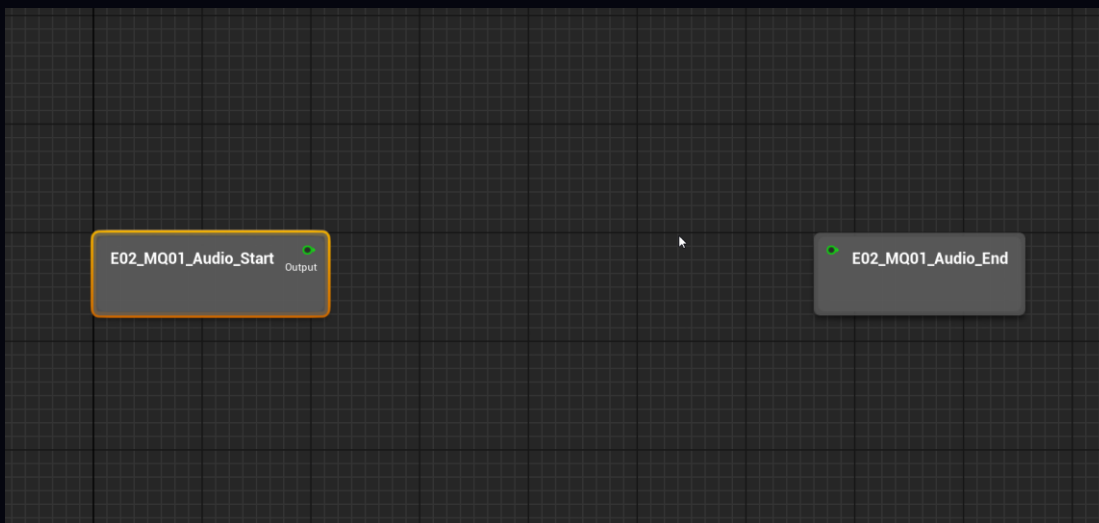
A separate Blueprint is used for handling audio logic, which shares the same name as the main quest but includes the `_Audio` suffix.



To quickly locate these assets in the project, use search filters.



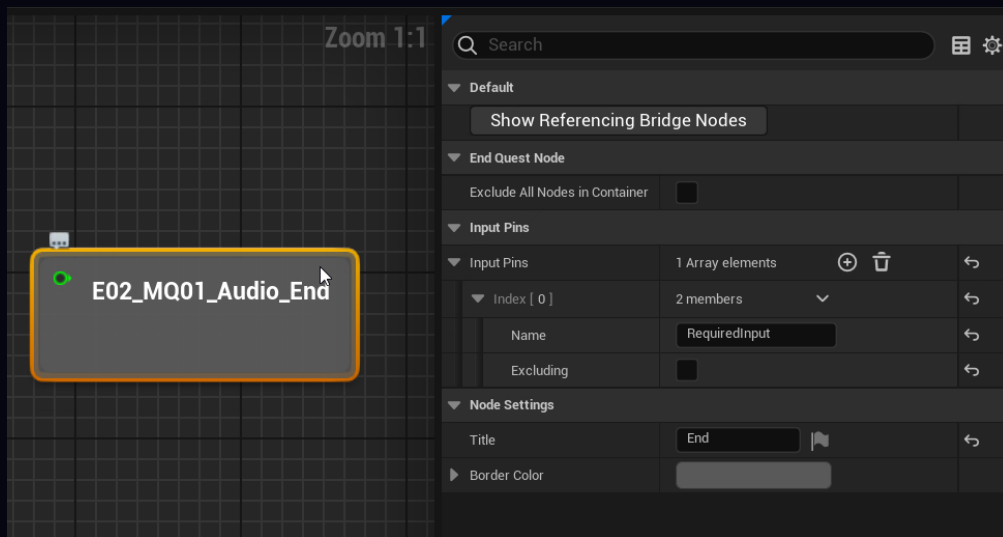
When opening this asset, you'll see its default structure, which consists of two basic nodes – Start and End, each with a single pin.



- The Start node activates all the logic defined within this Blueprint. The logic can be triggered either automatically when the quest starts or as a result of a call from another Blueprint. This node has no extra parameters, except for the ability to assign it a custom name.
- The End node ends the execution of the logic in this Blueprint. Once the signal reaches this node, all running logic stops.

Additional properties of the End node:

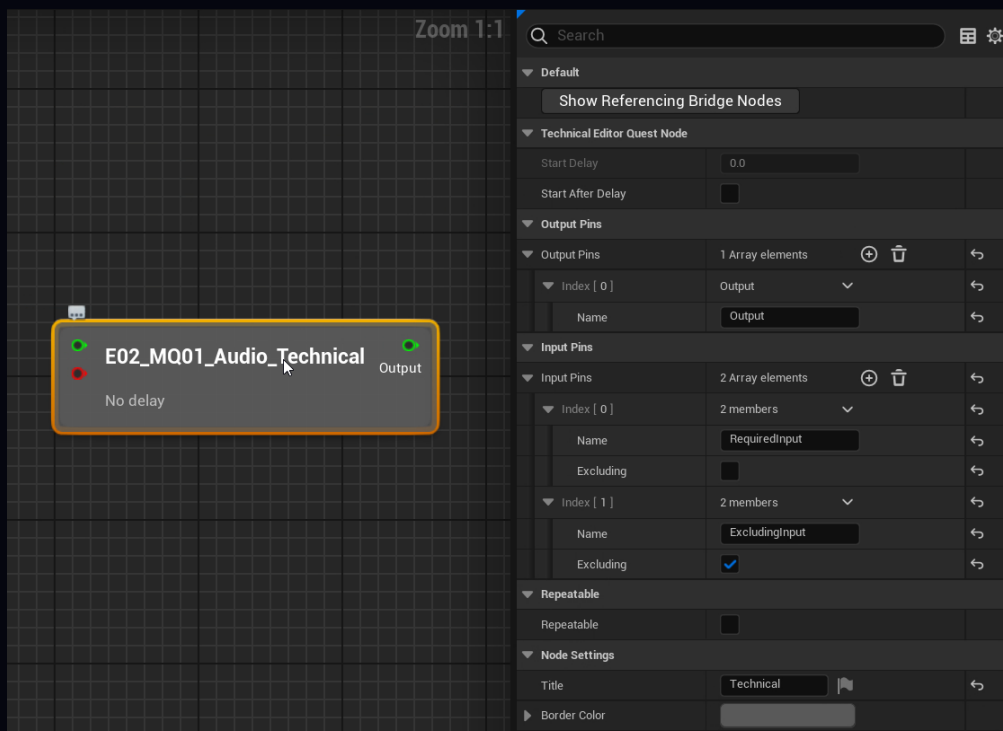
- **Exclude All Nodes in Container** – if set to **True**, all nodes within this Blueprint will be disabled upon execution. This helps clean up memory and fully stop the logic.
- **Input Pins** – you can create additional input pins to build A/B logic paths (alternative branches).



Connecting quest stages with audio

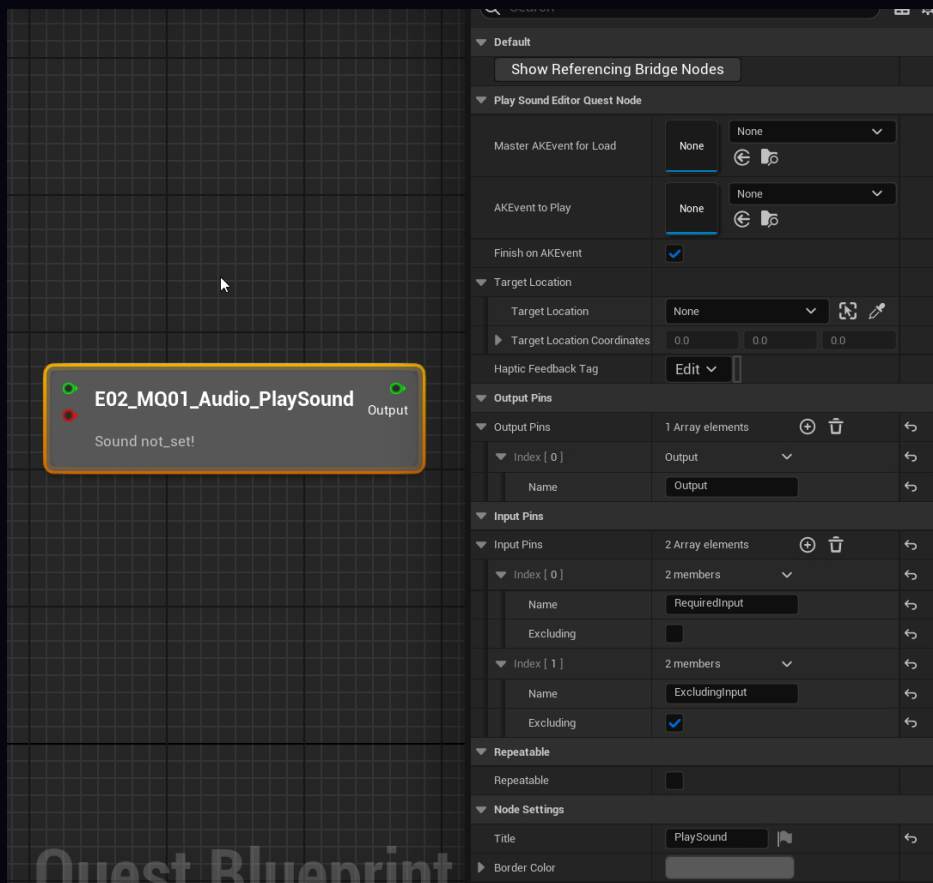
Logic in Quest Blueprints is constructed using nodes. These nodes have various properties and are used to build complex logic chains. All nodes share a set of basic properties, e.g., as seen in the Technical node:

- **Input/Output Pins** allow the addition of extra pins for receiving and passing signals, enabling the creation of more complex logic structures.
- **Excluding** disables the node and stops its execution. This parameter is tied to the Input Pins:
 - A green pin indicates an active input signal.
 - A red pin signals an exclusion, which halts the node's function.



In an audio Blueprint, the result of executing quest logic is typically a PlaySound node, which triggers a Wwise audio event.

You can add this node via the context menu available by right-clicking and searching for *Play Sound*.



Key properties of the PlaySound node:

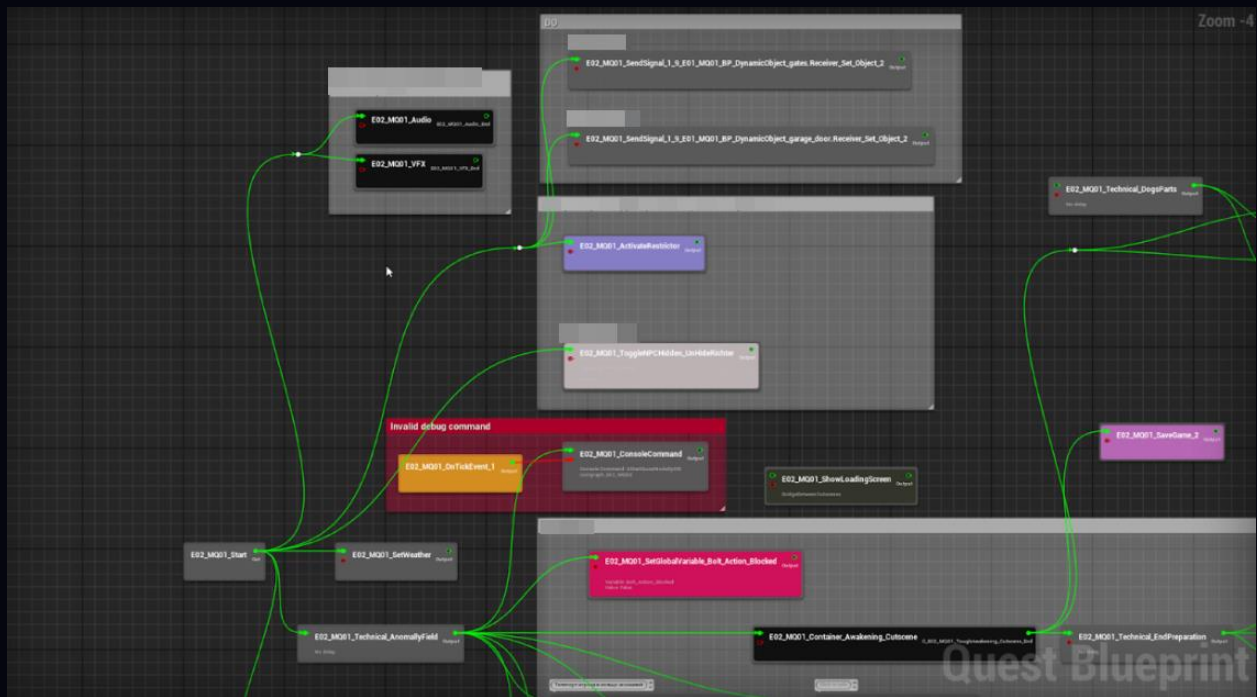
- **Master AKEvent for Load** allows you to set a Wwise event that will play during level loading. This is useful in cases where a guaranteed sound trigger is required.
- **AKEvent to Play** is the main field where you select the Wwise event that will be triggered when the node receives a signal.
- **Target Location** defines where in the game world the sound should play, e.g., an AK Ambient Actor or another actor placed in the level. If the actor is located inside a Level Instance, it must be unpacked in the Outliner to correctly select the playback point.
- **Output Pins/Input Pins** allow for the addition of extra pins to route signals after the PlaySound node is executed.
- **Repeatable** allows the event to be triggered repeatedly if the PlaySound node receives the signal more than once.

Main methods for triggering audio events

Working with an audio Blueprint involves setting up additional conditions to trigger sound logic from within the main quest Blueprint.

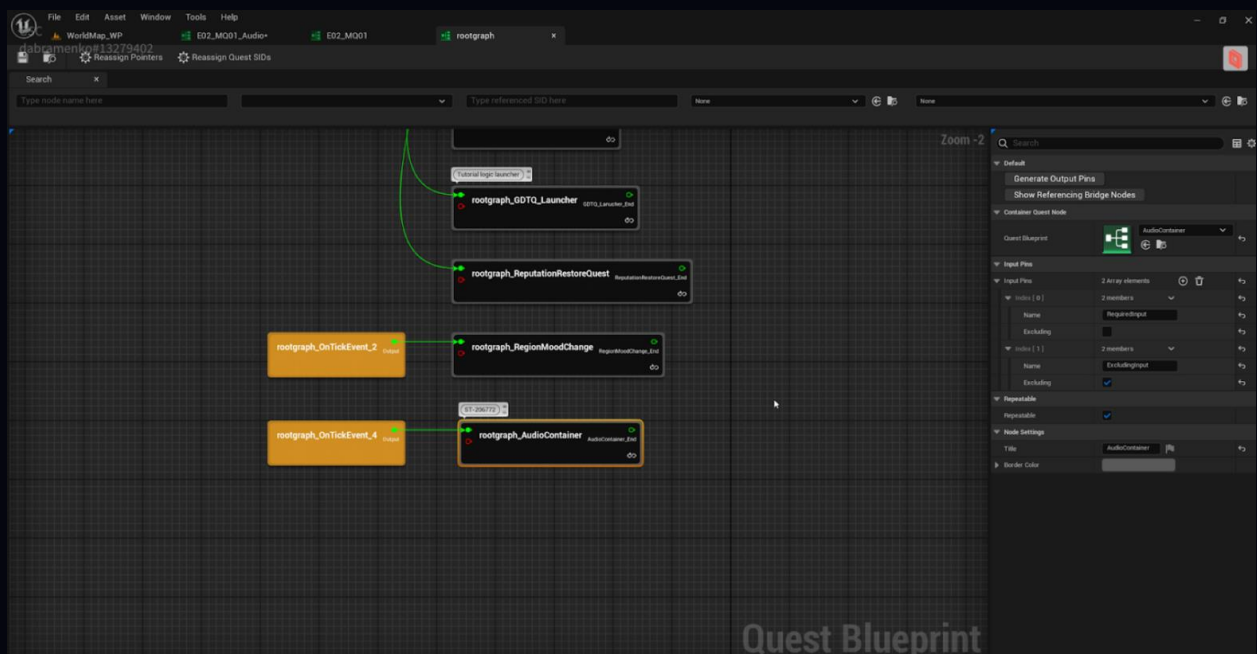
It's important to ensure that the Quest Blueprint contains the Container node that references and activates your audio Blueprint when it receives a signal from its Start node.

If the signal reaches the Container node that points to your audio Blueprint, it means the Blueprint will be activated, the Start node will fire, and your audio logic will proceed as expected.



Different logic structures can be used to trigger the PlaySound node, depending on the specific scenario.

It's also possible to run audio logic independently of the main quest Blueprint by using an audio container placed in the Root Graph and set to activate at game start. This allows for the creation of autonomous audio logic that functions throughout the game, regardless of specific quest progress.



Practical use cases and examples

Example: Integrating sound for a standard quest

Let's look at a simple logic structure that triggers the playback of a sound event.



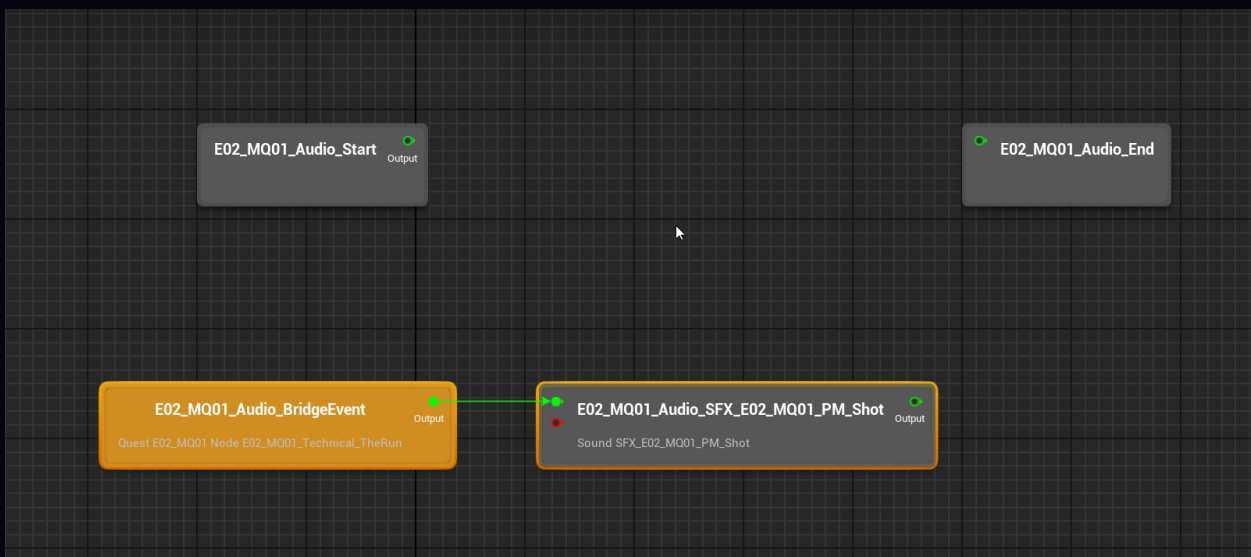
When the Start node is triggered, it sends a signal to the PlaySound node, which in turn activates the specified Wwise event and plays the sound. Once playback is complete (or immediately if not set to wait), the signal continues to the End node, which finalizes the execution of the entire Blueprint's logic.

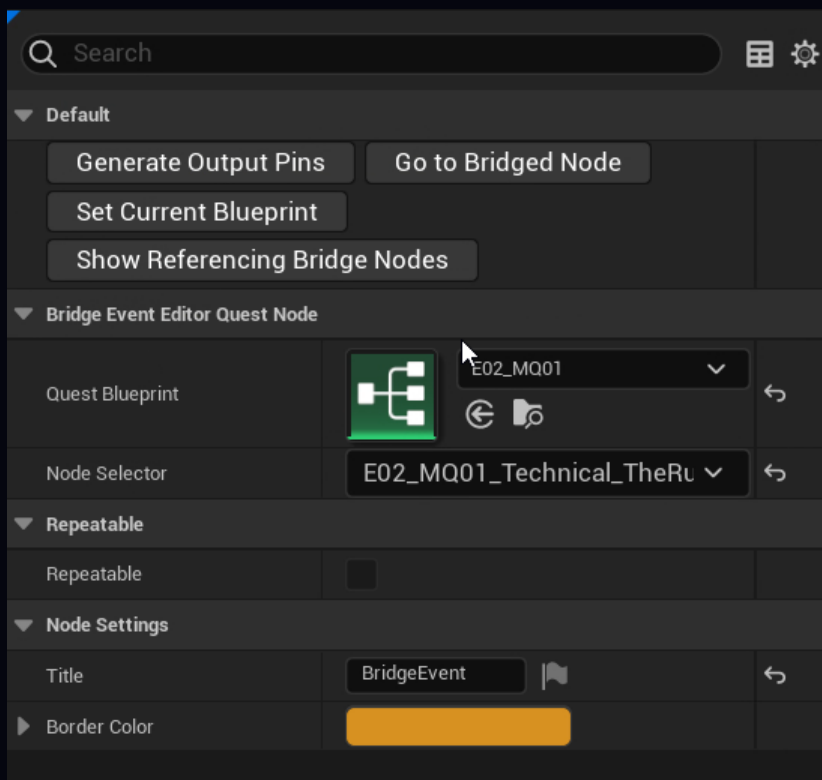
This example demonstrates a basic use case and the core principle of scripting audio logic in Quest Tool.

Next, let's look at several commonly used logic structures for different in-game events and quest scenarios.

Triggering a sound event using a bridge event

When designing audio logic for a quest, there is often a need to reference a specific node in the main quest Blueprint to trigger a sound from a particular point in the quest flow. The Bridge Event node is used for this purpose.





In the properties of the Bridge Event node, you can specify any Blueprint and the specific node within it that should act as a trigger. Once that node is executed, the Bridge Event activates and can pass the signal to a PlaySound node or any other logic.

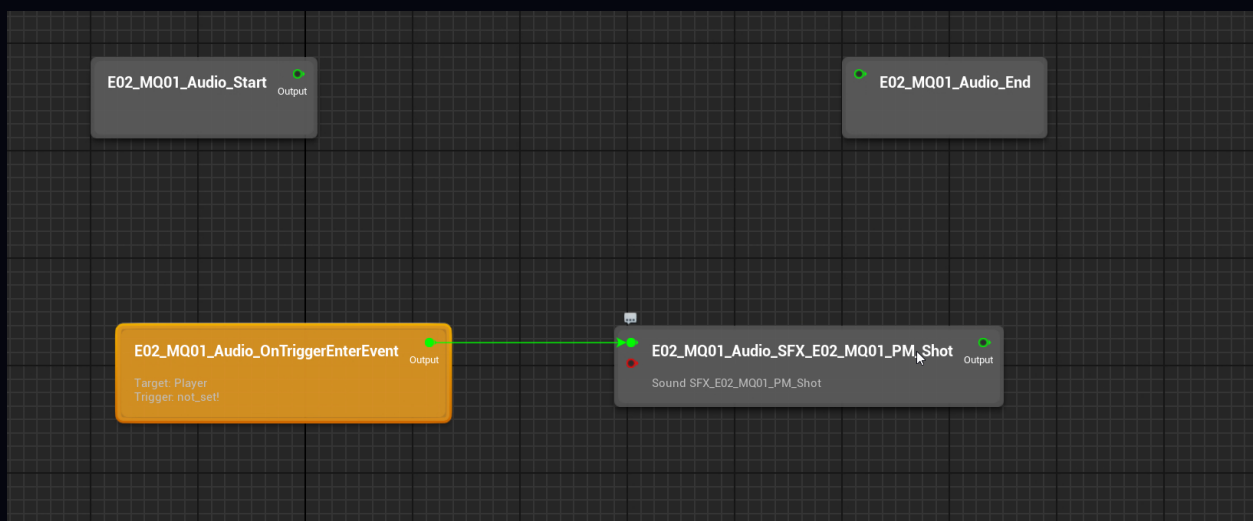
NOTE

The Bridge Event only continues execution after the selected node has finished its process.

Triggering a sound event using audio or narrative triggers

With the help of the On Trigger Enter Event and On Trigger Exit Event nodes, you can create logic that activates a PlaySound node when the player enters or exits a trigger zone.

This allows for spatial or situational audio playback tied directly to the player's movement through specific in-game areas.



Setup

1. Create the node – either On Trigger Enter Event or On Trigger Exit Event, depending on the desired behavior.

2. In the node's properties, under the **Trigger** section, specify the name of the audio trigger or narrative placeholder (these appear in red in the Viewer).
3. Connect the On Trigger Enter/Exit Event node to the PlaySound node to trigger the sound when the condition is met.

Search

▼ Default

Show Referencing Bridge Nodes

▼ On Trigger Event Editor Quest Node

▼ Trigger

Trigger: None

Triggers by Anybody: ☐

▼ Target Placeholder

Target: None

Target Player: ☒

Required Squad Members: Any Member

React Type: React on Alive

▼ Track Before Active

Track Before Active: ☐

▼ Output Pins

▼ Output Pin

Name: Output

▼ Repeatable

Repeatable: ☐

▼ Node Settings

Title: OnTriggerEnterEvent

Border Color: Yellow

Use case

Triggering sounds at specific points on the map as the player moves along their path.

This method is especially useful for:

- Ambient transitions
- Environmental cues
- Narrative moments tied to location-based triggers – it enhances immersion by dynamically reacting to the player's position in the world

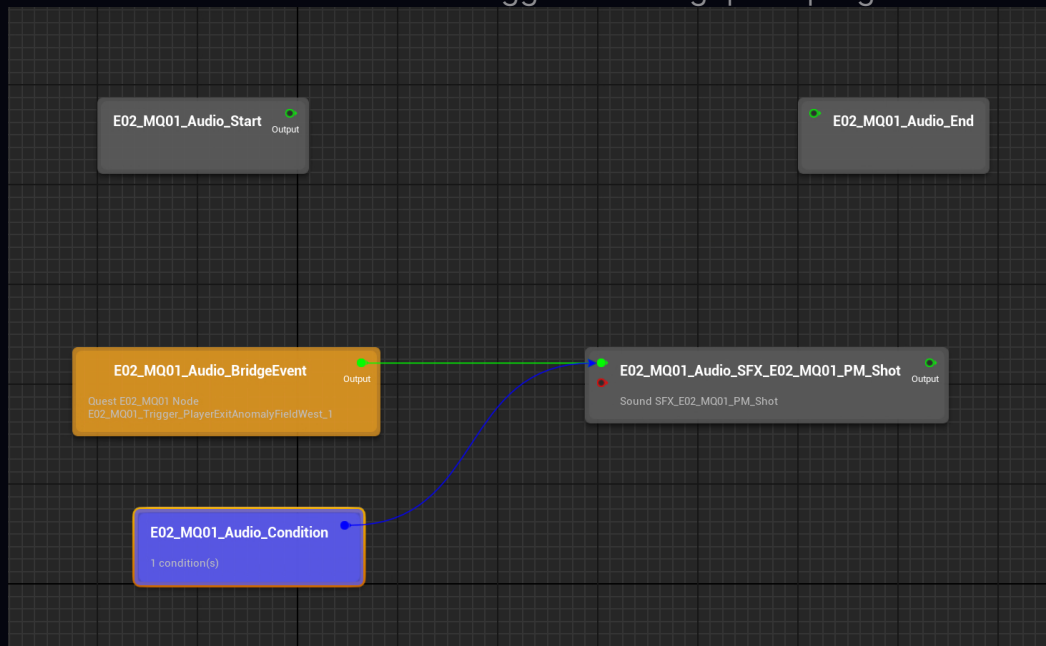
Executing a PlaySound node via a Condition node

The Condition node is used to control whether a signal is allowed to pass through. It has only an output pin and works in combination with other logic nodes.

Use case

If you want to trigger a Bridge Event and a PlaySound, but only if the player is within a certain distance, you can use a Condition node to gate the logic.

This setup allows for the creation of complex conditional structures, giving you fine control over when and how sounds are triggered during quest progression.



Search

Default

Go to Bridged Node

Show Referencing Bridge Nodes

Condition Editor Quest Node

Conditions List

Conditions

1 Array elements

Index [0]

Condition Type

Distance to Point

Condition Compara...

Equal

Numeric Value

1000.0

Target Location

Target Location

1_2_E01MQ01_BP_Generz

342424.601 648412.891 739.144317

Target Character Pl...

Required Squad Me...

All Members

NOTCondition

Condition Check Type

And

Repeatable

Repeatable

Node Settings

Title

Condition

Border Color

Using the OnSignalReceived node

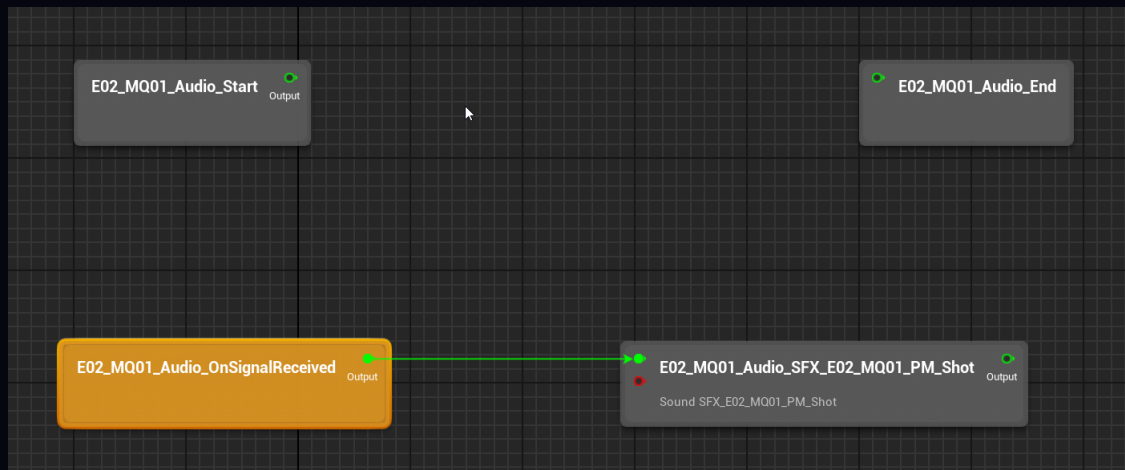
The OnSignalReceived node is used to receive a signal from interactive objects in the scene and pass it on to trigger a PlaySound event.

Use case

If there's a button in the game and you want a sound to play when it's pressed, follow these steps:

1. Add the OnSignalReceived node.
2. Specify which object should send the signal.
3. Define the component type that the signal comes from, e.g., button, trigger, switch.
4. Connect the output of OnSignalReceived to the PlaySound node.

This is one of the most common use cases for interactive audio in gameplay scenarios.



Show Referencing Bridge Nodes

On Signal Received Editor Quest Node

Signal

Sender Component Ref

None

None

Referenced Actor

None

Component Name

None

Sender Guid

{00000000-0000-0000-0000-000000000001}

Output Pins

Output Pin

Name

Output

Repeatable

Repeatable

☐

Node Settings

Title

OnSignalReceived

Border Color

On Player Noticed Event node

The On Player Noticed Event node is used to track whether a specific NPC has detected the player.

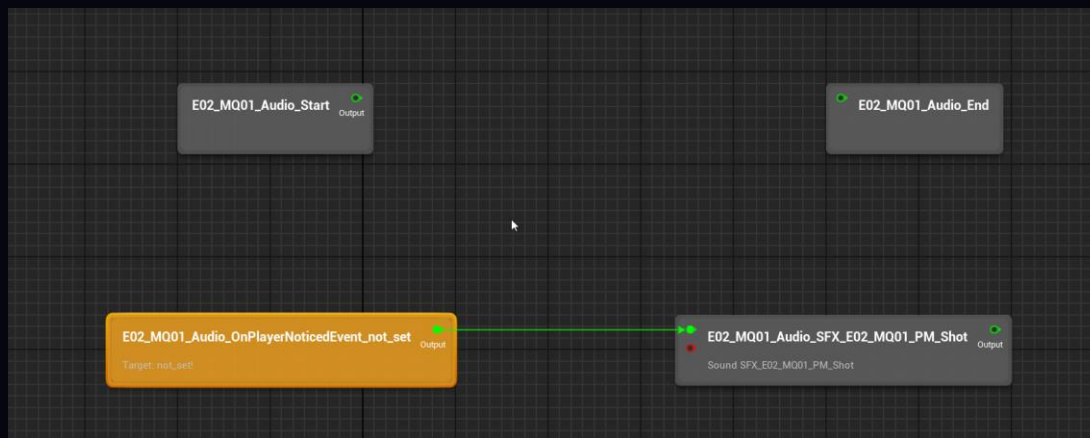
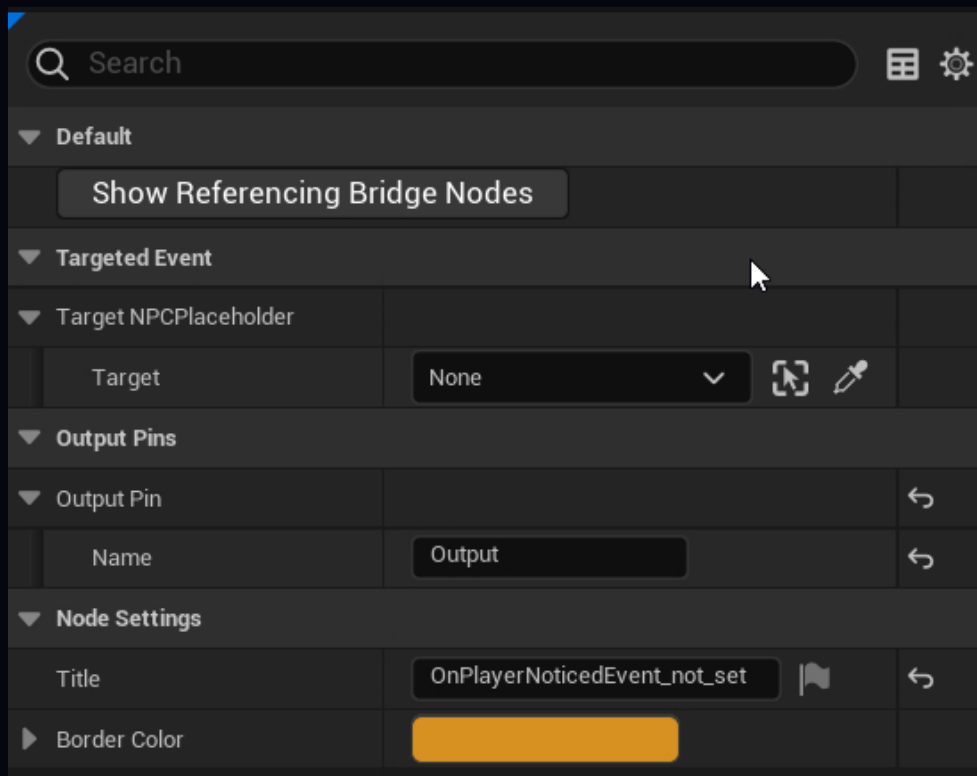
It sends a signal forward only when the selected NPC spots the player.

Use case

Tracking enemy NPCs during stealth missions enables dynamic audio responses, such as:

- Changing the music to reflect tension
- Triggering sound effects when the player is discovered

These kinds of implementations are powerful tools for enhancing immersion and building responsive audio logic for quests and interactive gameplay moments.



Testing and debugging

To ensure the correct functioning of audio logic within a quest, make sure to carry out testing according to the following criteria:

Full quest playthrough

To verify all audio events and their interaction with game logic, do the following:

- Ensure that all nodes in both the main quest Blueprint and the audio Blueprint function correctly.
- Run the quest under the proper in-game conditions required for its full completion.
- Use the start command specified in the QSO documentation for that particular quest.
- If no such command is available, consult the quest owner to clarify the correct way to launch it.

Testing audio logic separately from the quest blueprint

If the audio logic is not directly tied to the main quest Blueprint, e.g., it uses a Bridge node, you can test it independently:

- Run the Start node in the audio Blueprint to trigger all sound-related nodes inside.
- Confirm that all required audio events play under the correct in-game conditions.

Testing audio logic connected to the quest blueprint

If the audio logic is directly dependent on the main quest Blueprint:

- Be sure to launch the Start node in the main quest Blueprint to allow proper interaction.
- Check that all sound events trigger at the correct moments in the game, in sync with quest progress.

Testing in the PIE mode

When testing and debugging audio logic in PIE mode, keep in mind:

- The process may take more time than working with other audio systems.
- It's important to plan ahead and consider this factor in your workflow scheduling.

Conclusions and recommendations

Key best practices

Before starting the development of audio logic and content creation, please carefully plan the sound logic within the context of the quest. This impacts:

- The content creation process
- Configuration in Wwise
- Scripting within Blueprints

The main structure of the audio logic should be formed in Wwise, as its functionality allows you to build a solid foundation of sound events, which simplifies the Blueprint integration.

Key points:

- Use Wwise's built-in tools to manage events. Events can not only play sounds but also stop them directly in Wwise, reducing the need for extra scripting in Blueprints.
- Pre-planning your audio logic helps avoid unnecessary rework and ensures smoother interaction between audio and quest systems.

Common mistakes and how to avoid them

During development, there are a few frequent issues to be aware of:

- **Missing Stop event for looping sounds**
Every looping sound event must have a corresponding Stop event, so the sound can stop at the right moment and doesn't stay active in the game uncontrollably. E.g., if a radio emits static noise during a quest, ensure it stops when the radio is turned off.
- **Incorrect scope for Stop events**
If a looping sound in Wwise isn't tied to a specific object, set the Stop event's Scope to Global. Otherwise, the sound may continue playing even after the stop condition has occurred.
- **Incorrect spatial placement**
Always double-check that the playback location is correctly specified in the PlaySound node's properties.

Quest Tool for GD Sequences

Description

IMPORTANT

This document is intended for users who already know how to work with interactive objects, logs, anomalies, object spawners, mutant spawners, and NPC spawners.

The limited functionality of Quest Tool is used to create unique scenes in locations and as an addition to the basic gameplay functionality and *cannot* be used instead of it.

For example, enemy ambushes can be integrated into an *already* existing lair, but for the first visit, enemy ambushes *cannot* replace the lair, as the script is one-time only. However, it is *possible* to create a situation with an ambush on the player when there is already a lair at the location. Then this situation will happen to the player on their first visit to the lair.

FAQ

- Are there any script examples?

You can find them on WorldMap_WP:

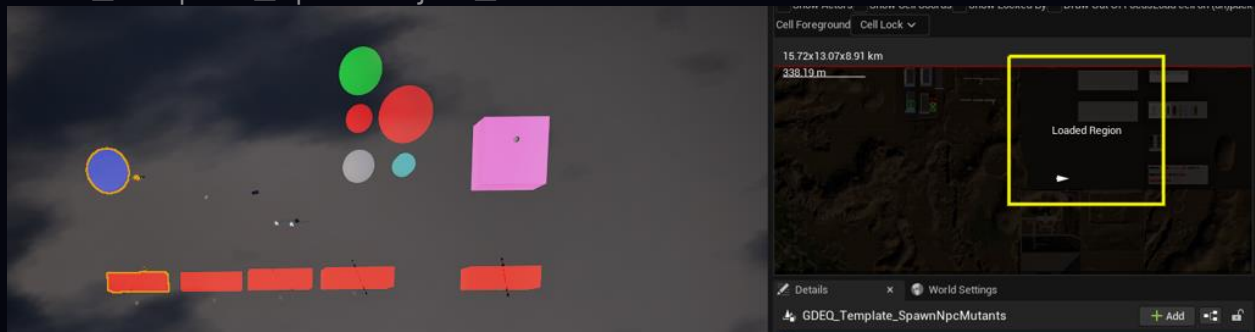
GDEQ_Template_SpawnNpcMutants_BP

GDEQ_Template_ActivateLair_BP

GDEQ_Template_AnomalyAct_BP

GDEQ_Template_ActivateIntObj_BP

GDEQ_Template_SpawnObject_BP



- How can I test the script I created?
 - Launch PiE.
 - Launch the `XStartQuestBySID [QUEST_BP_NAME]` console command.
- Why do spawned NPCs not go to the contextual actions?

All NPCs/mutants spawned via BP_SquadPlaceholder BP_NPCPlaceholder do not occupy contextual actions by default. If you want them to occupy contextual actions, see [Helpful nodes](#).

Cases

This section contains a list of cases that can be created using GDEQ:

- A village on the edge of the Zone. The player breaks the boards in the doorway, enters the house, and a mutant runs out of the next room and starts attacking.
- The player enters the village and, passing by a house, hears an object fall inside. He enters the house and the events from the previous bullet point occur.
- The player approaches the location, which is empty, and begins to explore it. He finds a good stash in the middle of the room and approaches it. Bandits spawn around it and start attacking the player. After that, the lair is activated at the location.
- The player enters the room, sees an anomaly ahead, goes around it to another room, and the anomaly is suddenly activated by a trigger.

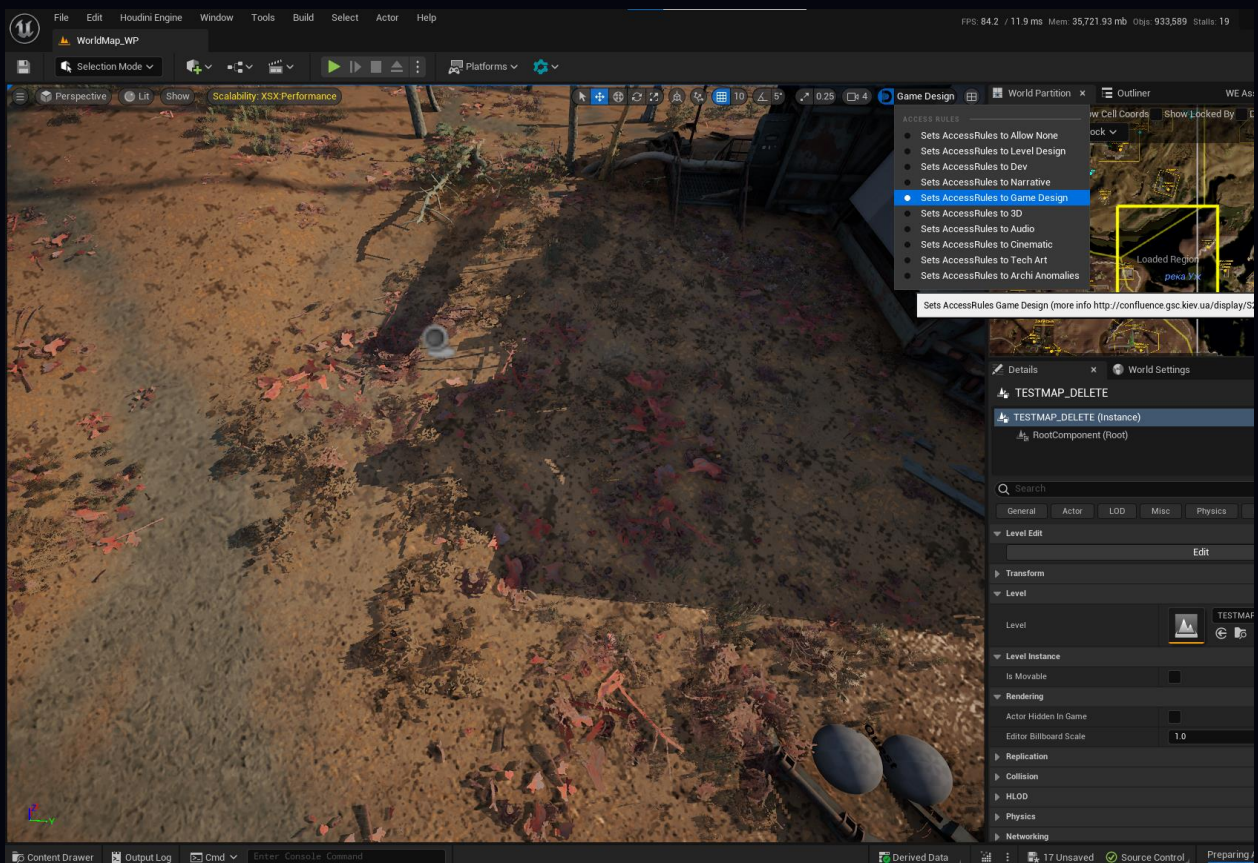
Creating a map and quest BP

Follow recommendations below to create a point that will be the first element and starting point for creating a level:

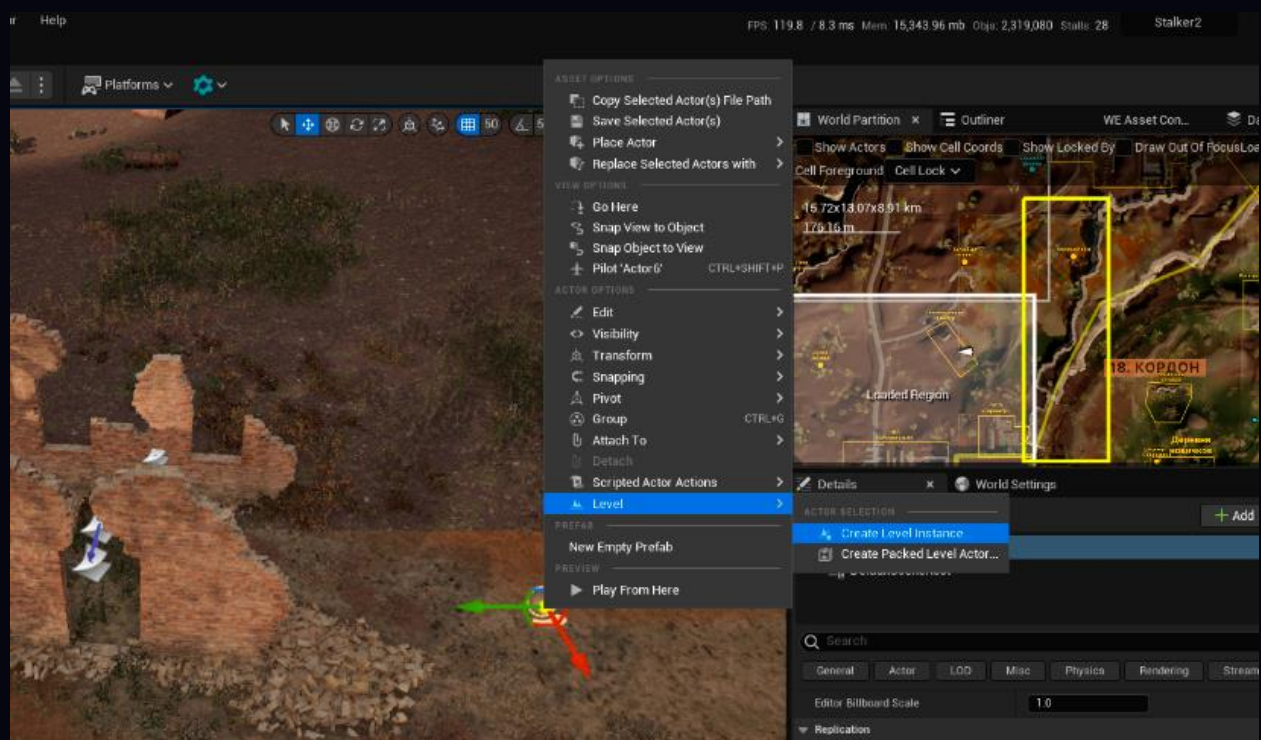
1. Go to **Basic** → **Actor** and add an empty actor to the map, as shown in the screenshot.
2. Place the actor within a radius of 1000 units from the place where GDEQ will be placed.



3. Before creating a level, switch your permissions. In the upper right corner of the window you are working in, select **Sets AccessRules to Game Design**.

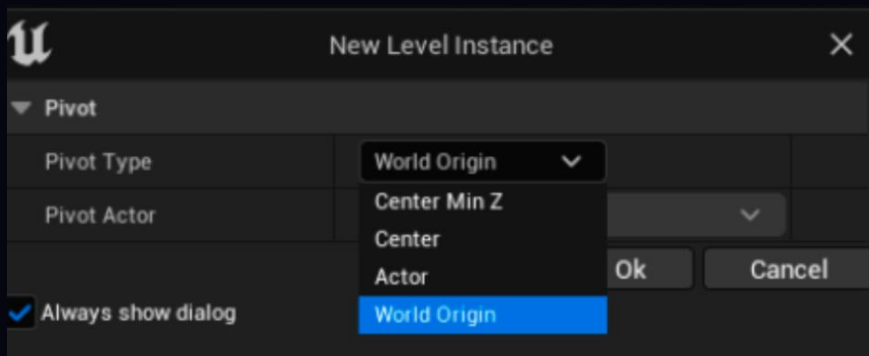


4. Right-click on the actor and select **Level → Create Level Instance**.

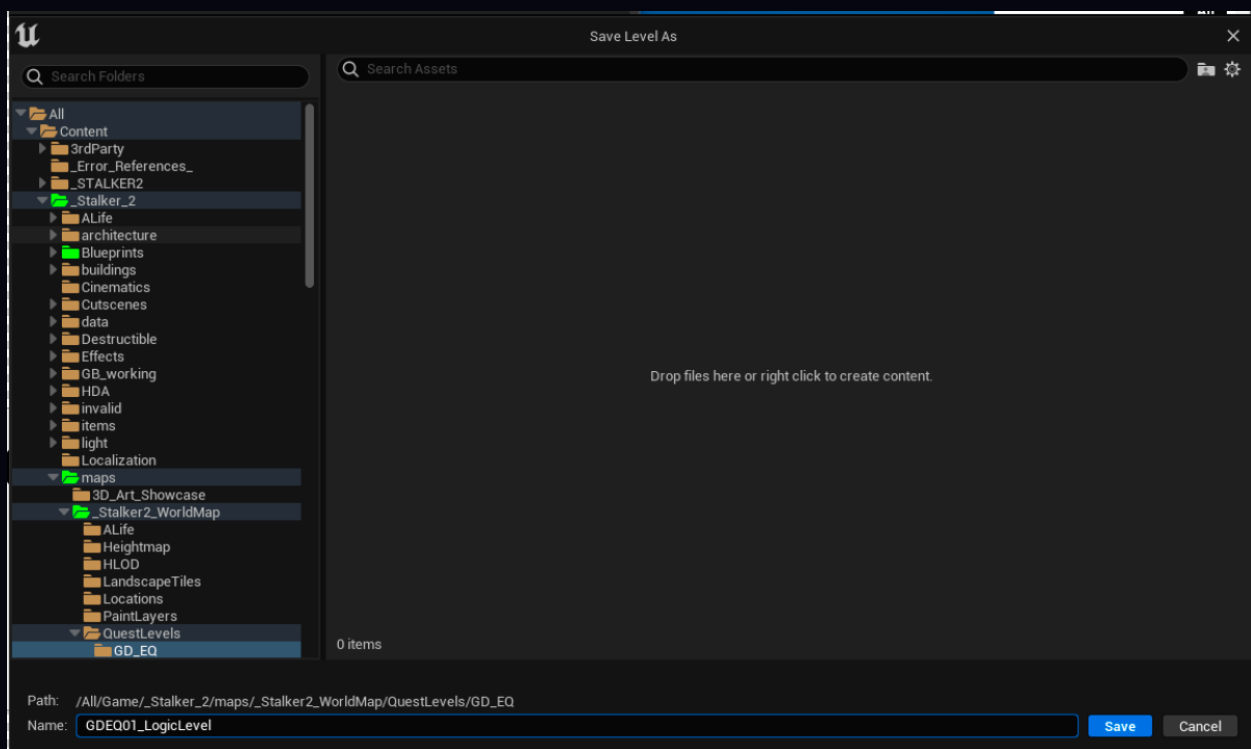


5. In the window, select **Pivot Type → World Origin**.

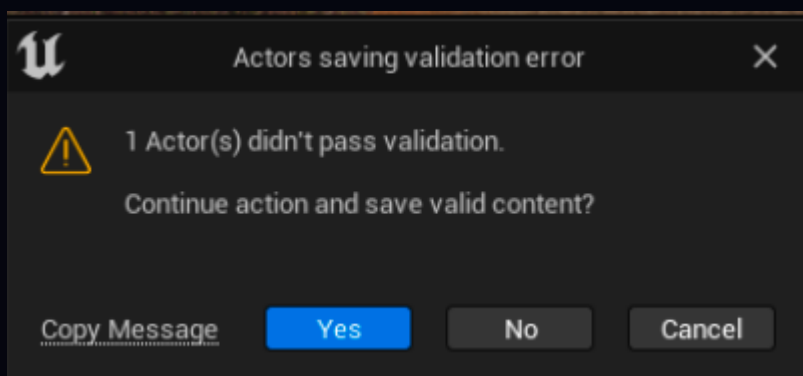
6. Select the **Ok** button.



7. In a new window, select the path for the level you are creating. The path to the folder where the levels should be saved is `_Stalker_2/maps/_Stalker2_WorldMap/QuestLevels/GD_EQ/`.
8. Enter the correct name of your level: `GDEQ[your ID]_LogicLevel`. Example: `GDEQ1001_LogicLevel`.

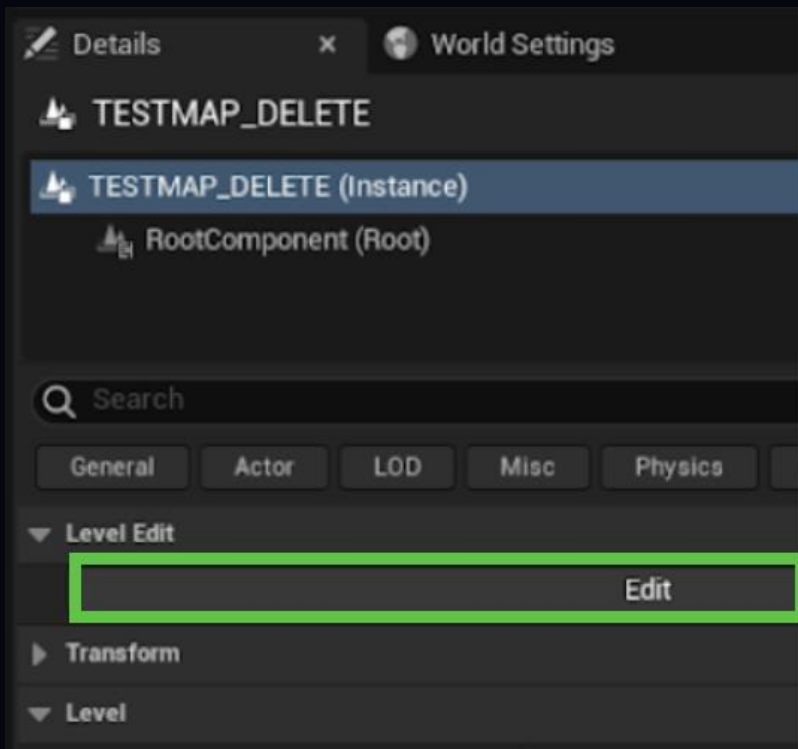


9. If you see a warning message, it means you do not have the rights to create levels, and you haven't done the step above. Select **Yes** and check the files for commit and StageTool to see if all the necessary elements are there or not. You can also roll back all uncommitted changes on the branch so you can work with a clean branch.



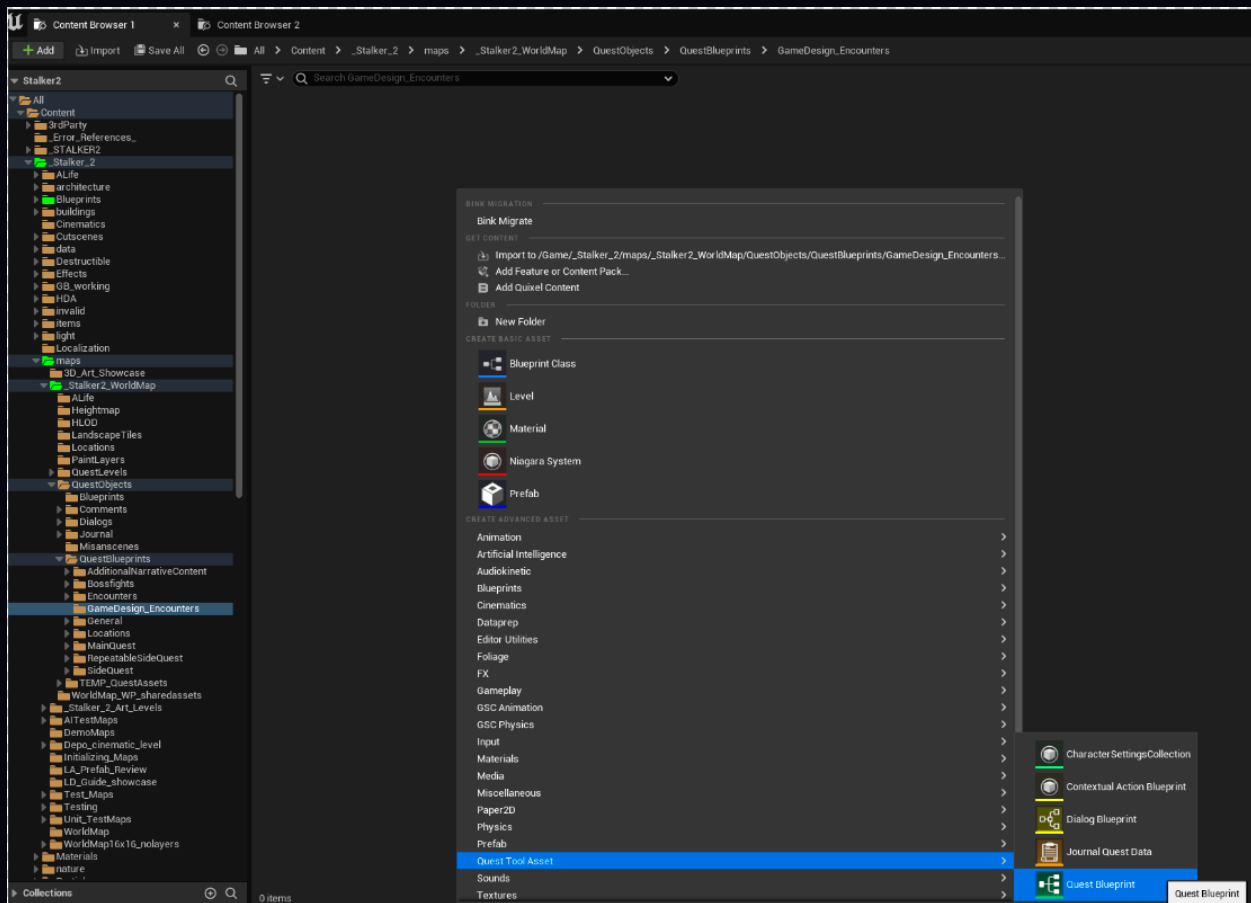
IMPORTANT

Pay attention to which level you are in. The green frame on the screenshot shows the button that lets you start the work in the level. When you finish working on this level, select the same button.

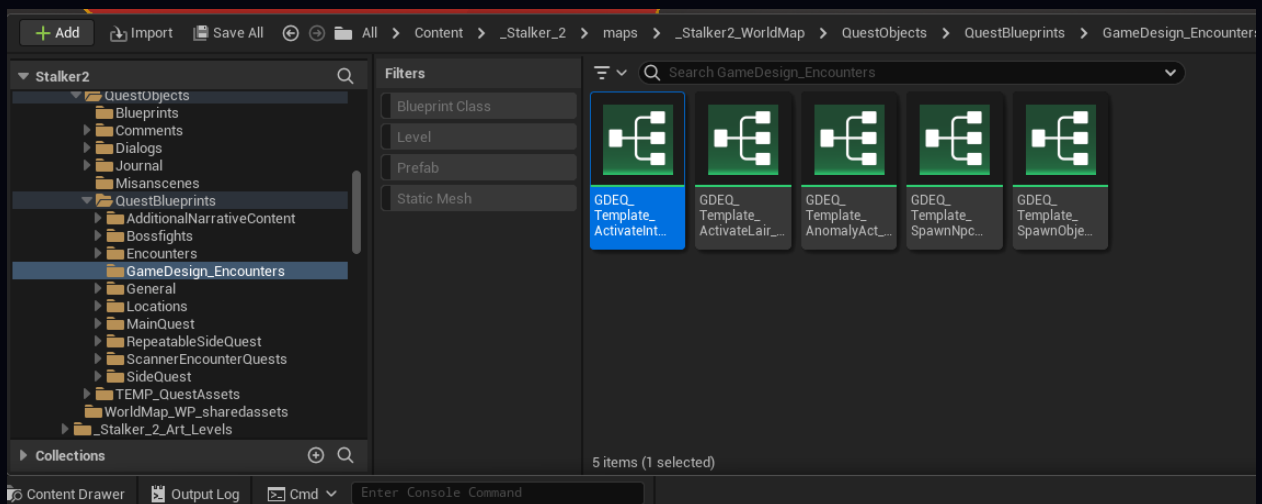


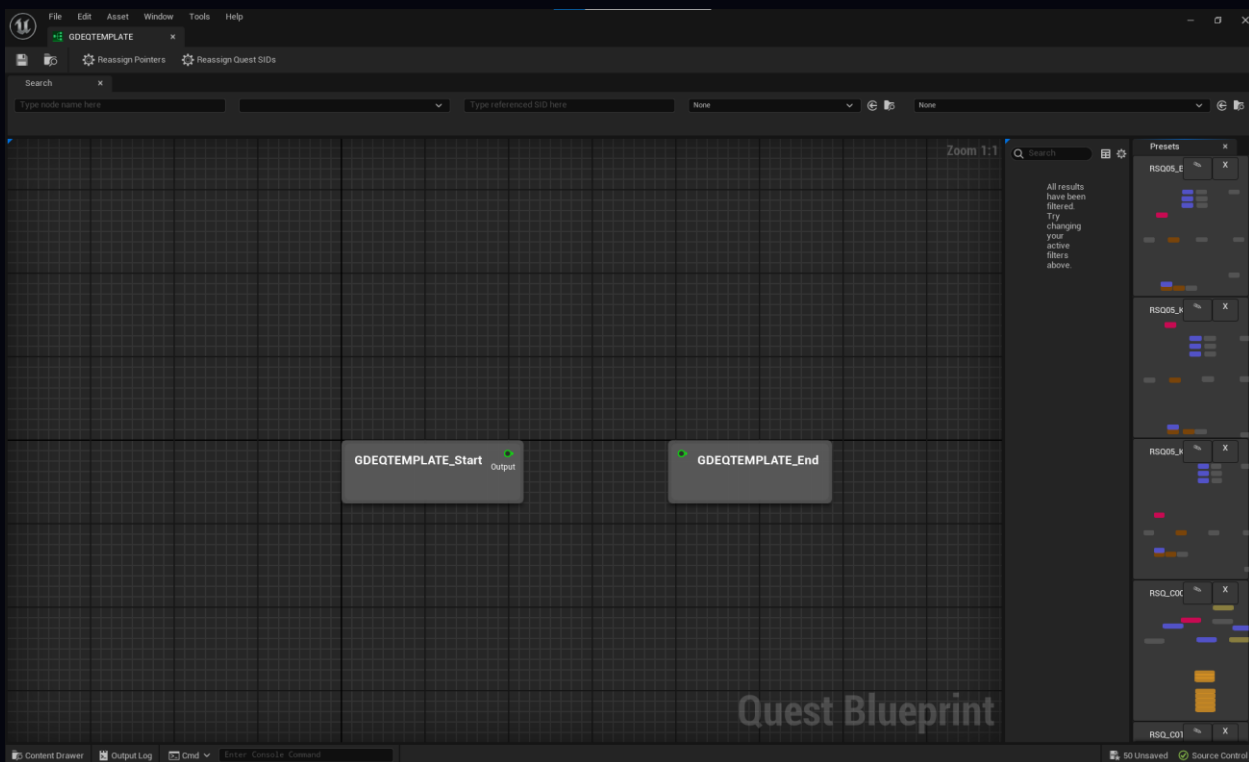
Now, let's create a quest blueprint. Follow these steps:

1. The path to the folder where you need to save the quest blueprint is:
`_Stalker_2/maps/_Stalker2_WorldMap/QuestObjects/QuestBlueprints/GameDesign_Encounters/`.
2. Create the blueprint name using your ID: `GDEQ[your ID]`. Example: `GDEQ1001_BP`.

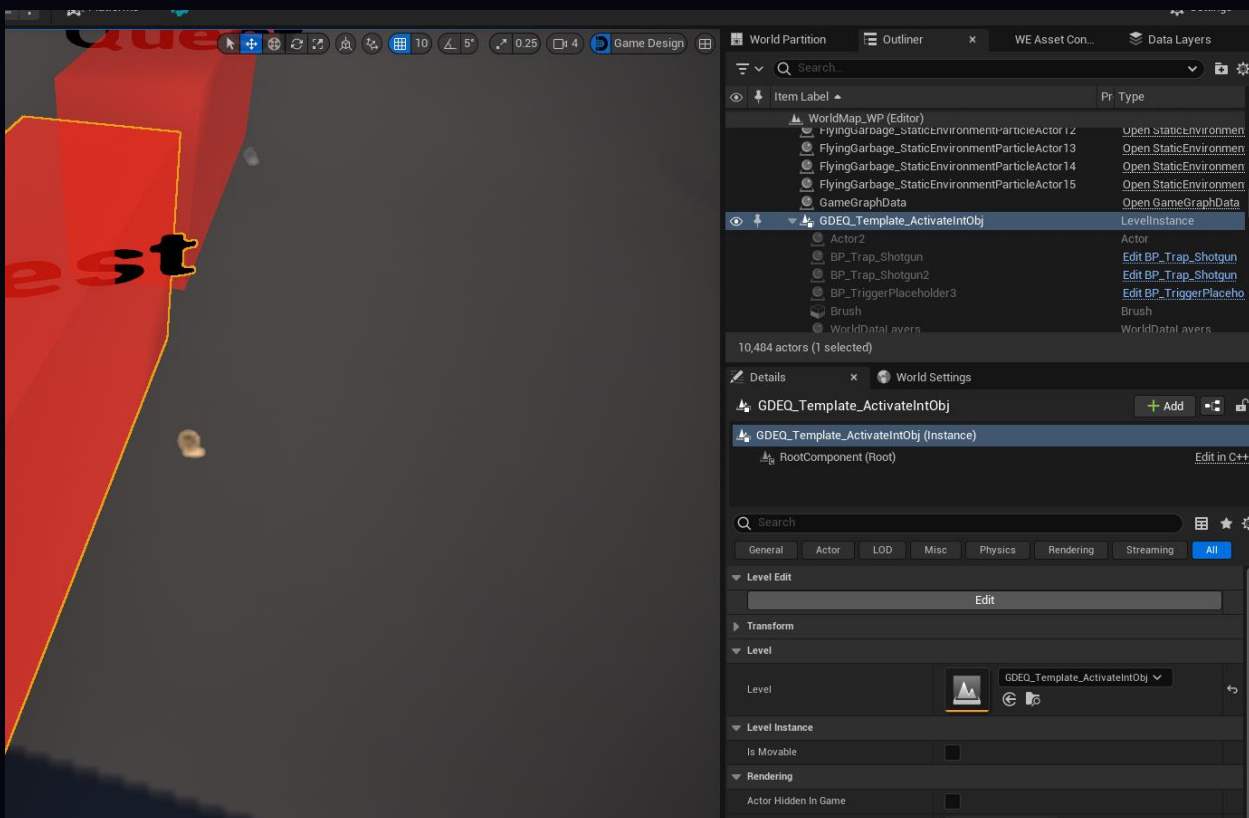


- To start working with a quest, double-click on the quest blueprint you created in the content browser, and you will be transferred to its editing window.



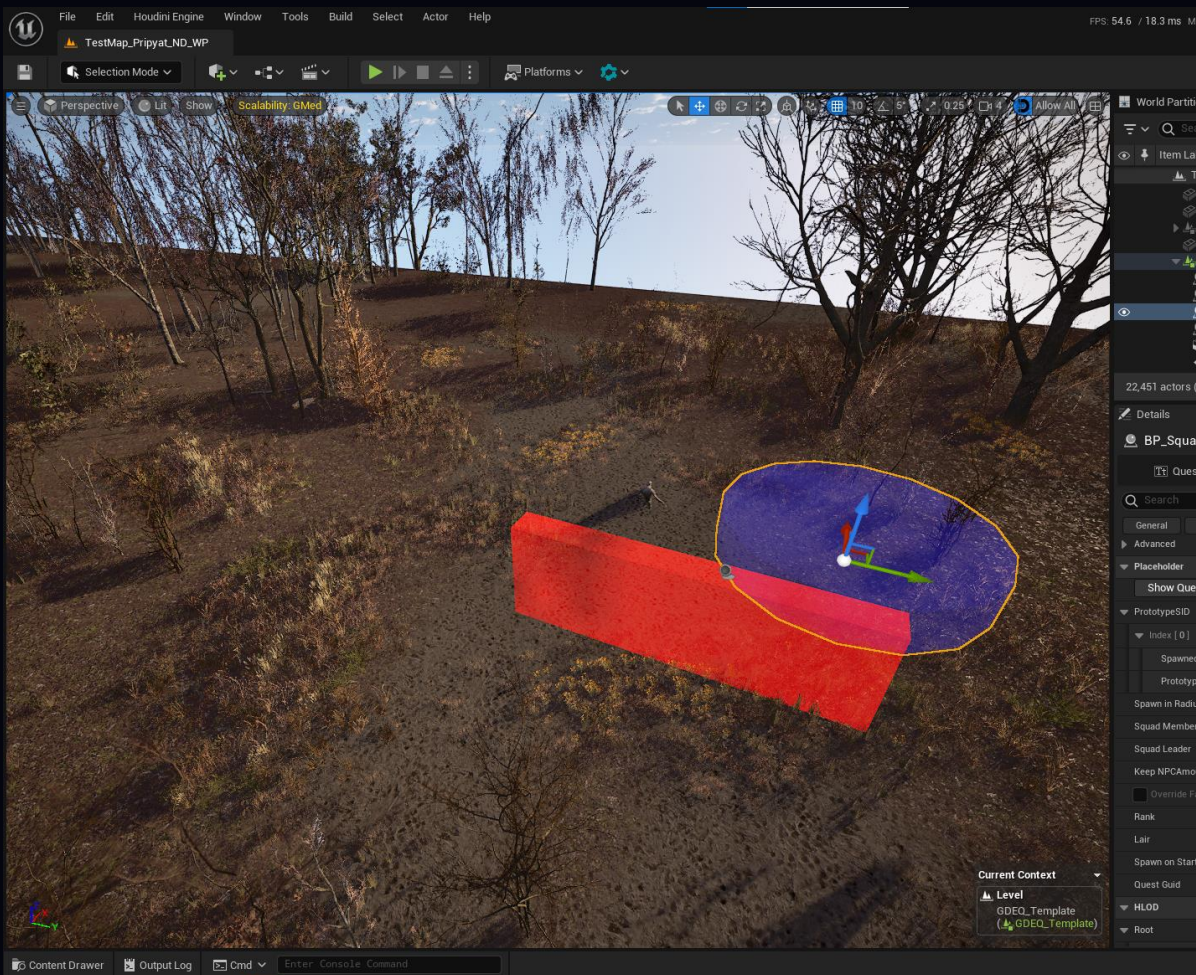


- Make sure you work only within the level you have created.
- To get to the right level, select your created actor on the stage and select the **Edit** button in the window on the right.



- In the lower right corner of the screen you are working on, you can see what level you are on.

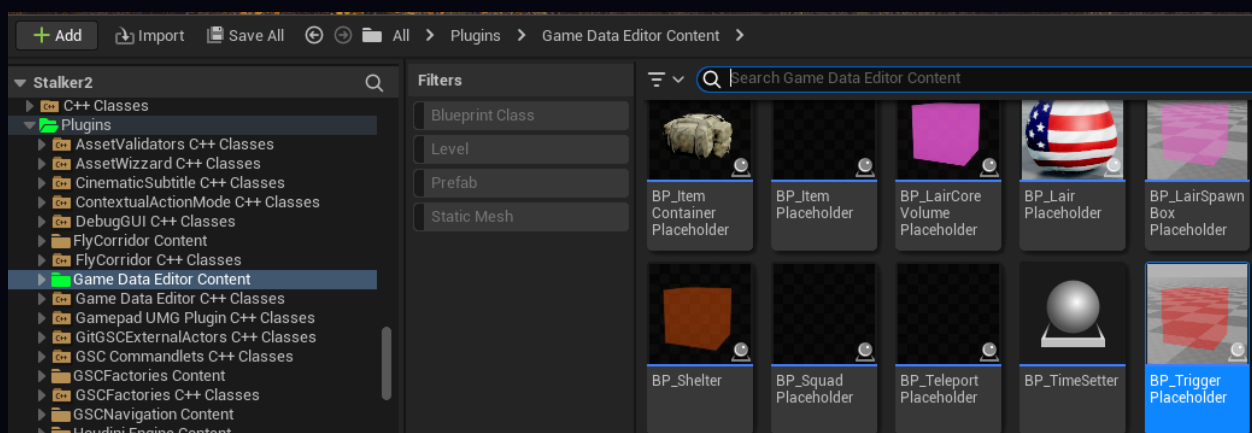
Make sure you are always in the quest level you created and place all elements in it.



NPC/mutant spawn

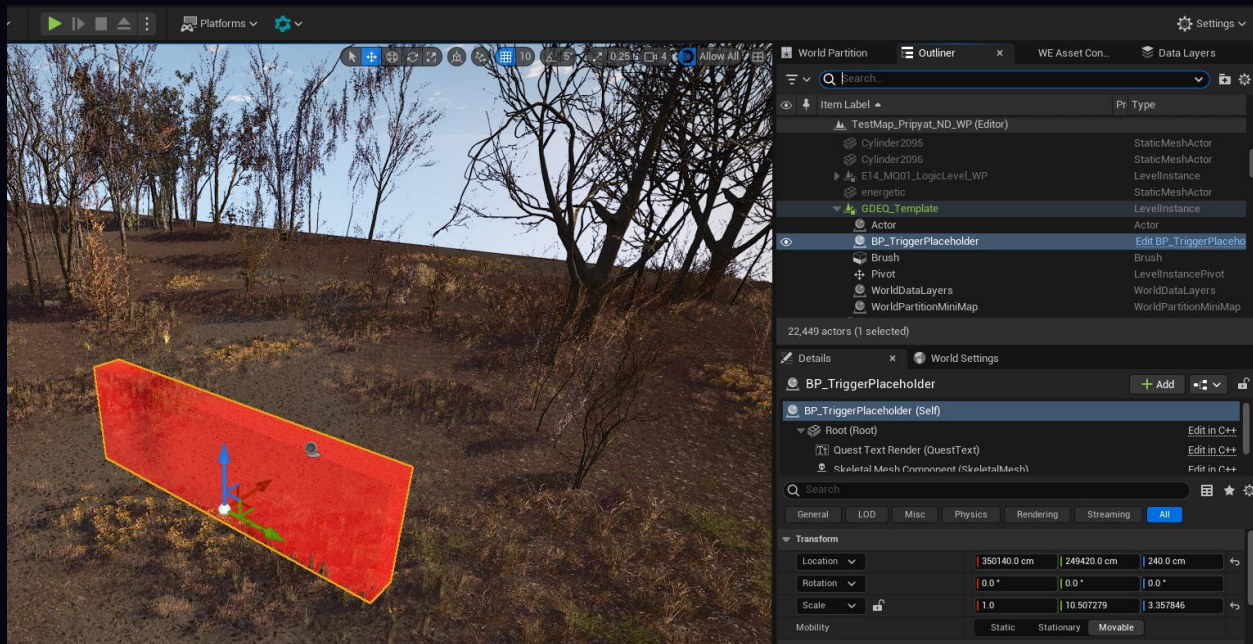
To spawn a mutant\NPC according to the script, follow these steps:

1. Find the trigger to activate the script. The path to the folder where the trigger placeholder is located: *Plugins/GameDataEditorContent*. Also, you can find it by the name of the actor: *BP_TriggerPlaceholder*.

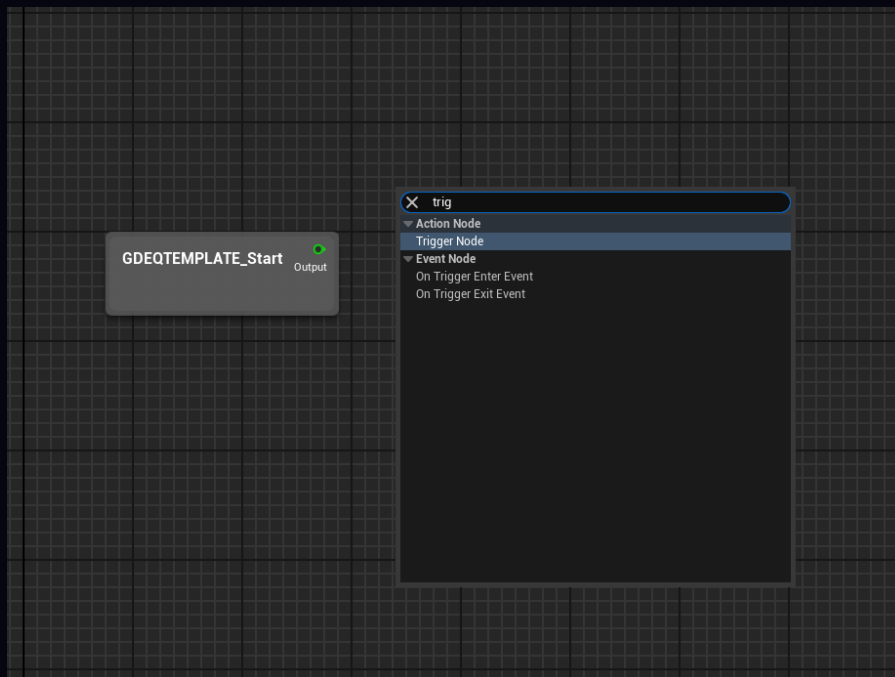


2. Drag the trigger from the context browser to the stage and stretch it over the desired area.
 - The volume thickness must be at least 100 units (1 meter) and no more than 200 units (2 meters).
 - The width and height depend on the environment.

- The volume should not extend beyond the boundaries of the room or opening in which it is installed. This is made so that it cannot be activated through walls, floors, and ceilings.

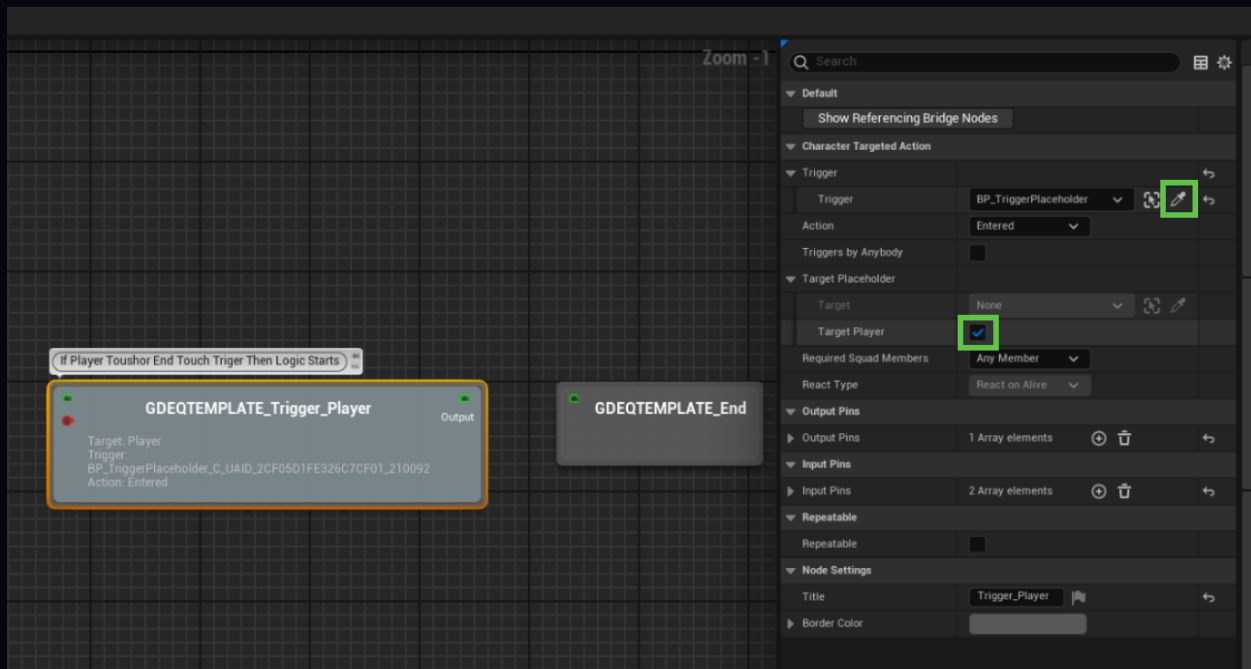


3. Right-click inside the quest blueprint and select **Trigger Node**.



4. Use the pipette tool to select the previously set trigger volume. This way, the node will refer to a specific actor in the world.

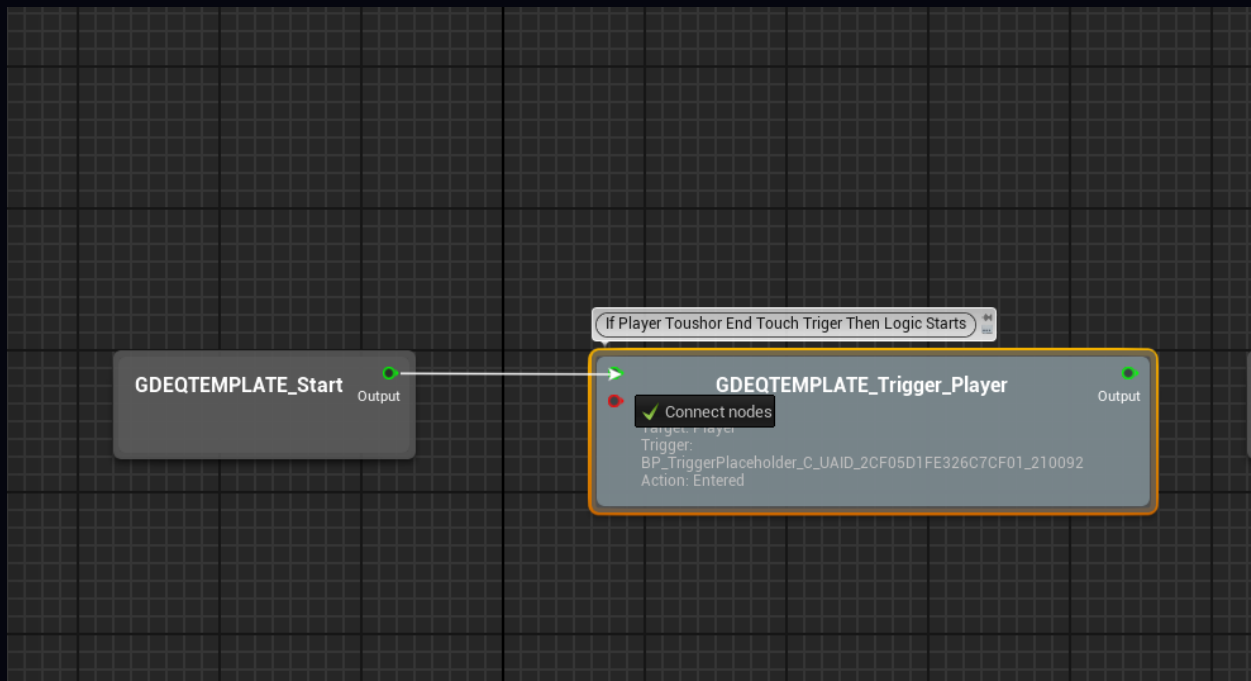
5. Select the **Target Player** checkbox so that the trigger only works on the player.



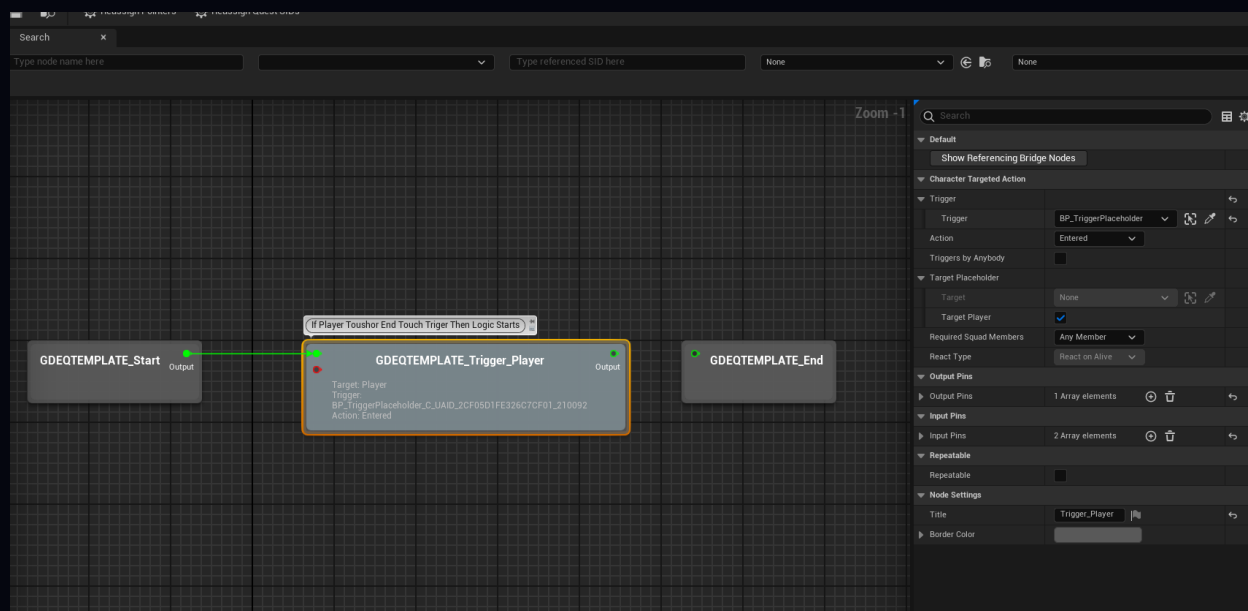
6. Extend the spline from the green quest start pin to the node with the trigger. Do this for each node.

IMPORTANT

Please be careful when monitoring where you are dragging from and where you are inserting the spline.

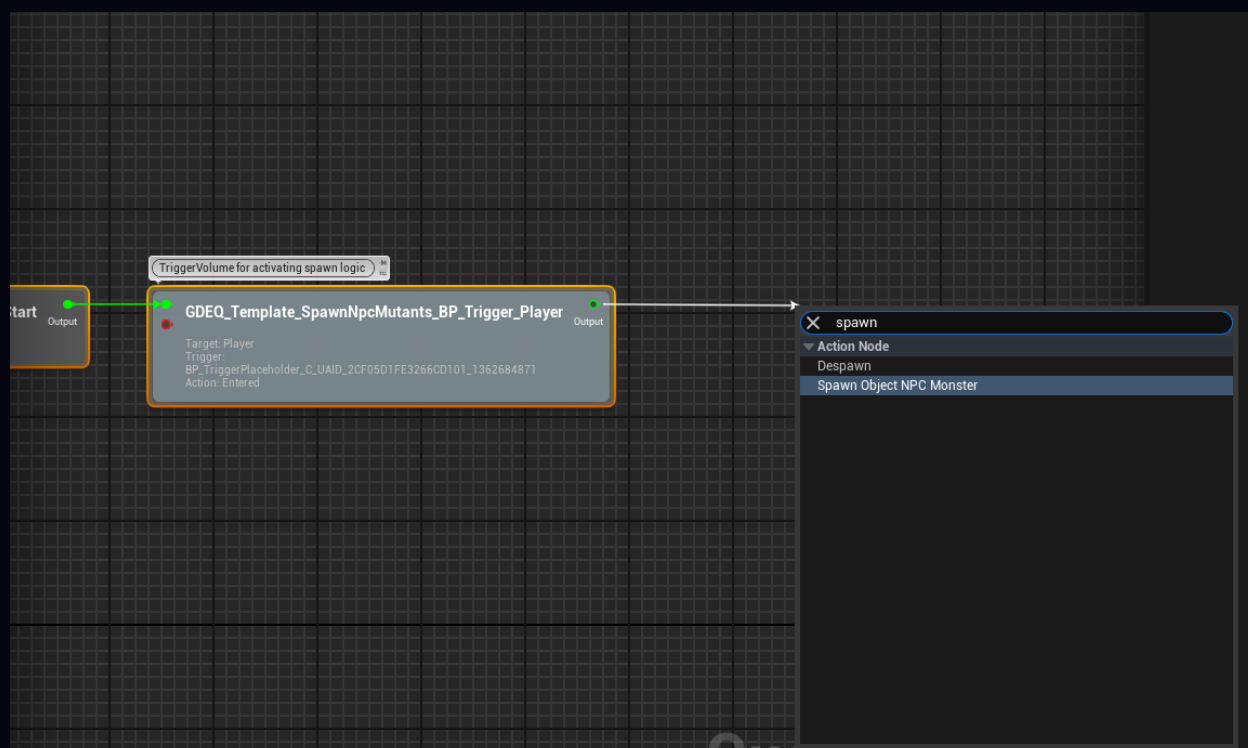
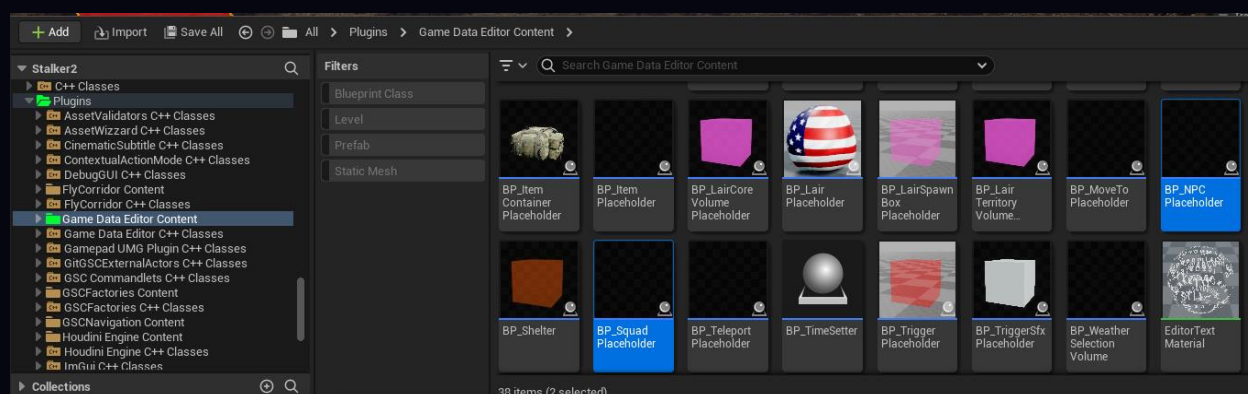


The final result should look like this:

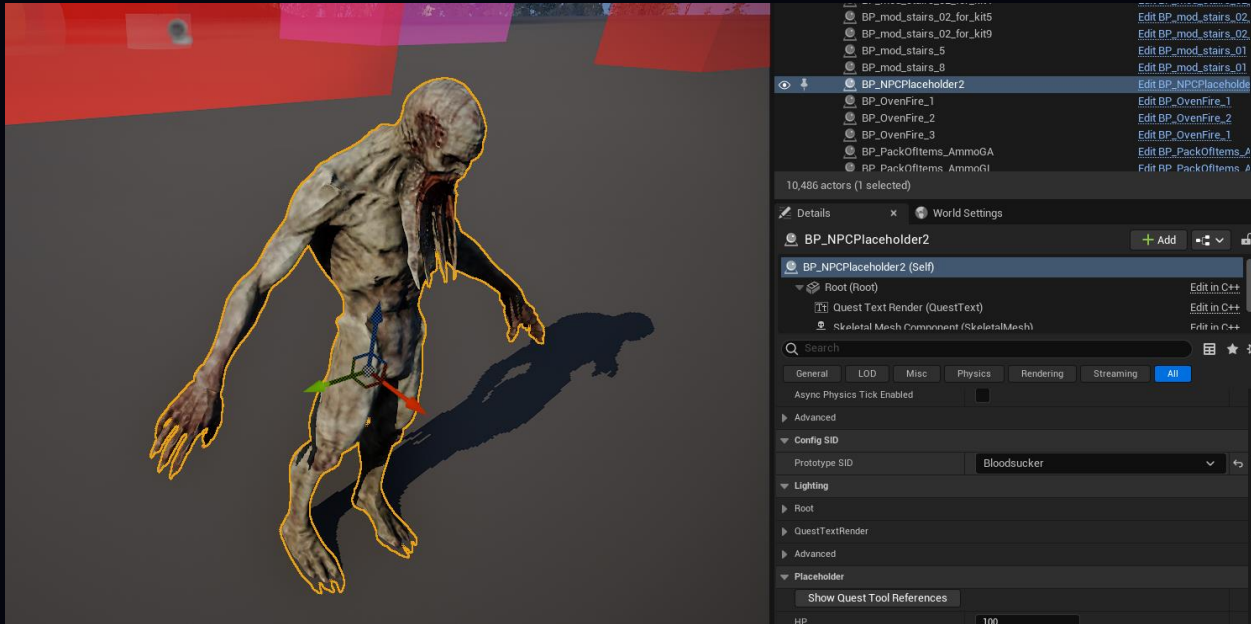


To add a node for a mutant\NPC spawn, follow these steps:

1. Find mutant/NPC spawn actors in *Plugins/GameDataEditorContent*. Also, you can find it by the name of the actor: *BP_SquadPlaceholder*, *BP_NPCPlaceholder*.
2. Right-click inside the quest blueprint and select **Spawn Object NPC Monster**.

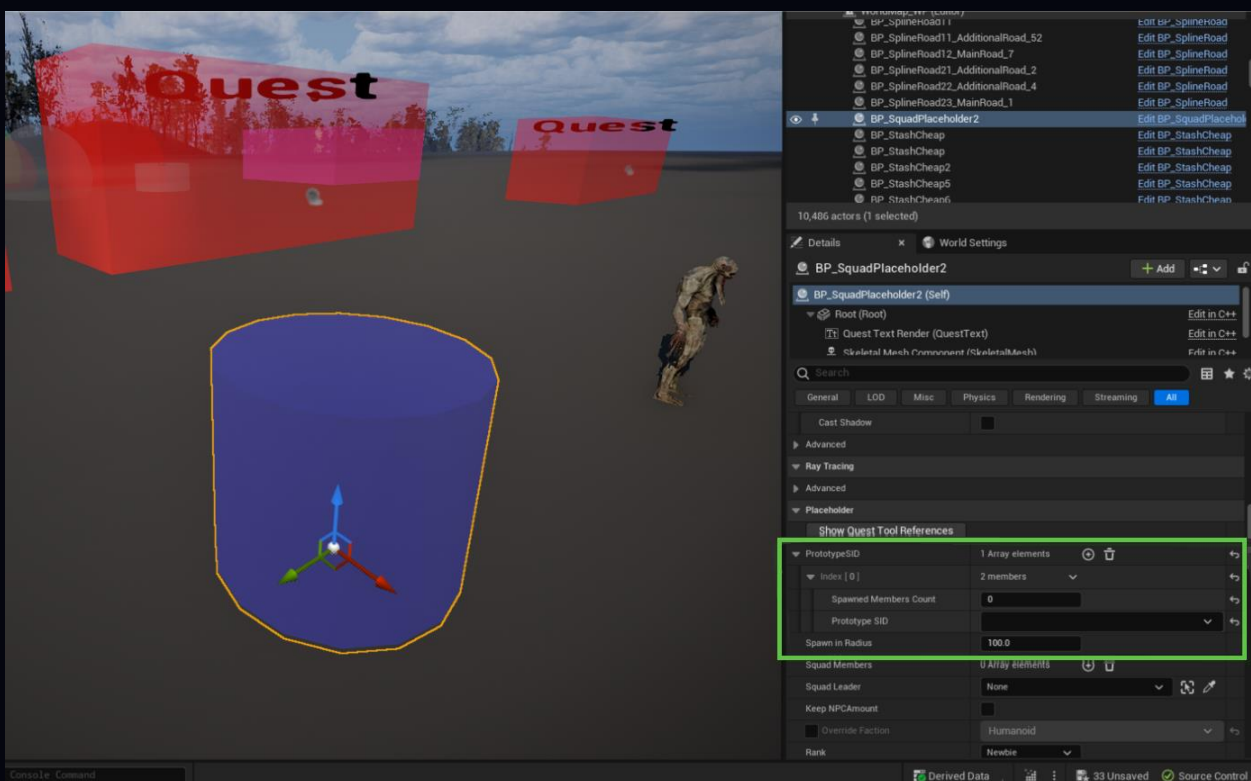


3. To spawn one character, place BP_NPCPlaceholder on the stage. Make sure to specify the SID of the character you want to spawn. If it is an NPC, you can only select SIDs with the word *General* in the SID name. Selecting others is prohibited.

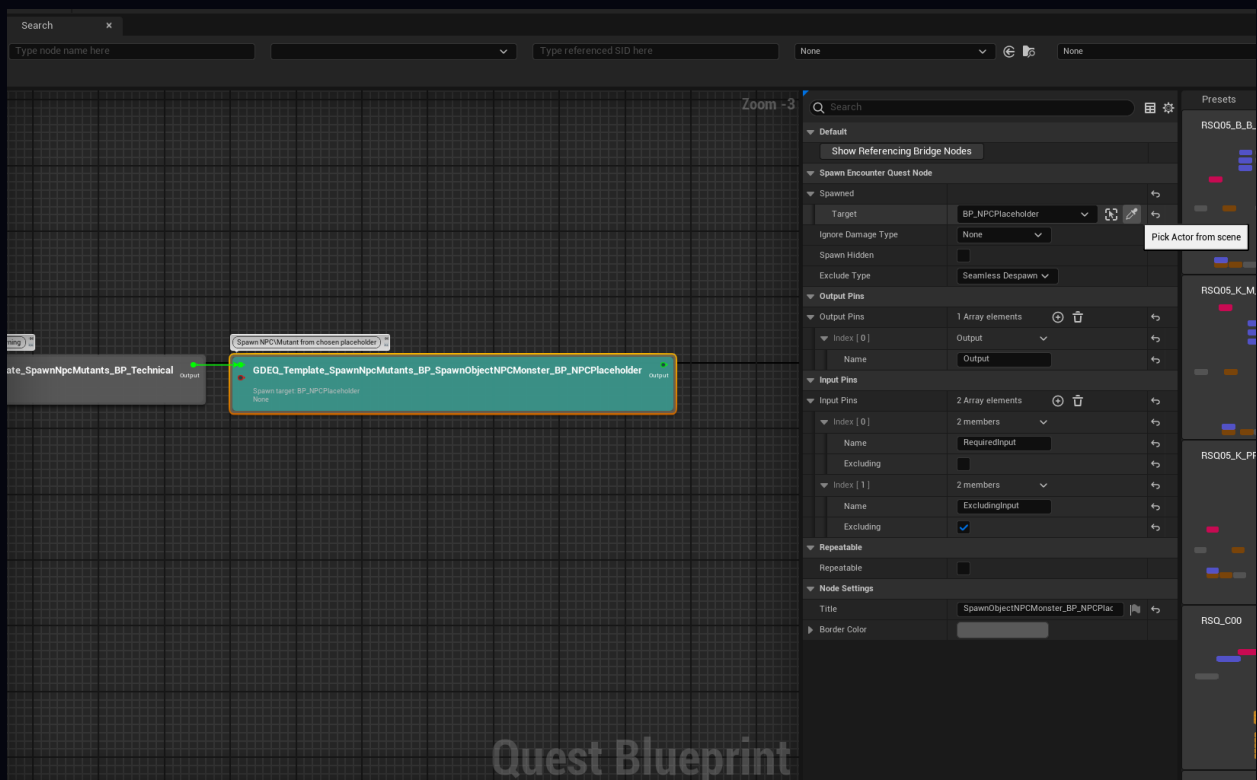


To spawn a group of characters, use BP_SquadPlaceholder, which you should place on the stage. Make sure to set the following parameters:

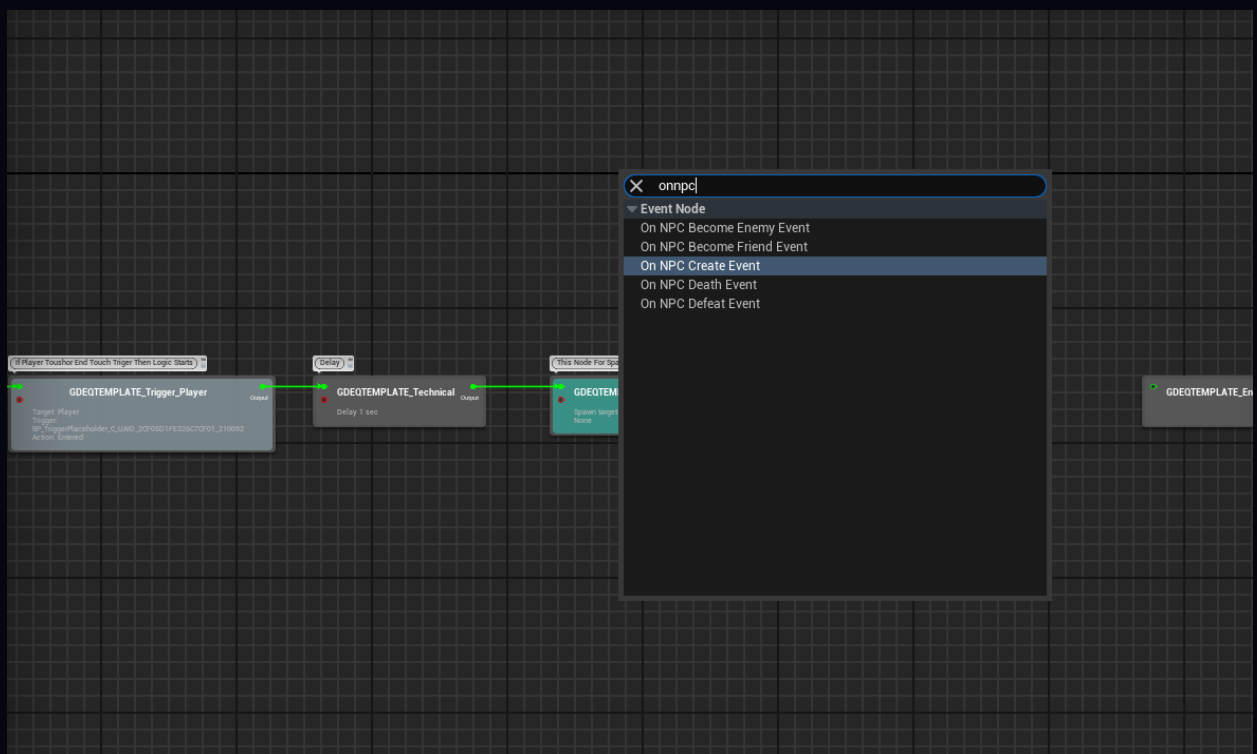
- **Spawned Members Count** – Number of characters
- **Prototype SID** – SIDs of the characters you want to spawn
- **Spawn in Radius** – radius in which NPCs/mutants specified above will spawn



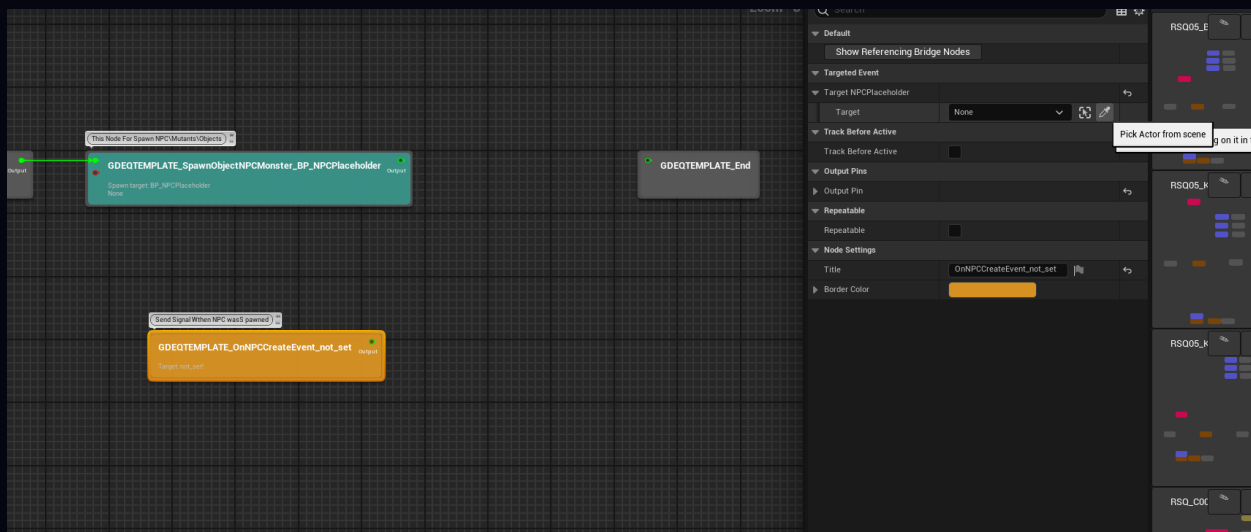
4. Use the pipette tool and select the spawn (single or group) that was previously placed on the map to connect the node to the previously created one with a spline. This way, the node will refer to a specific actor in the world.



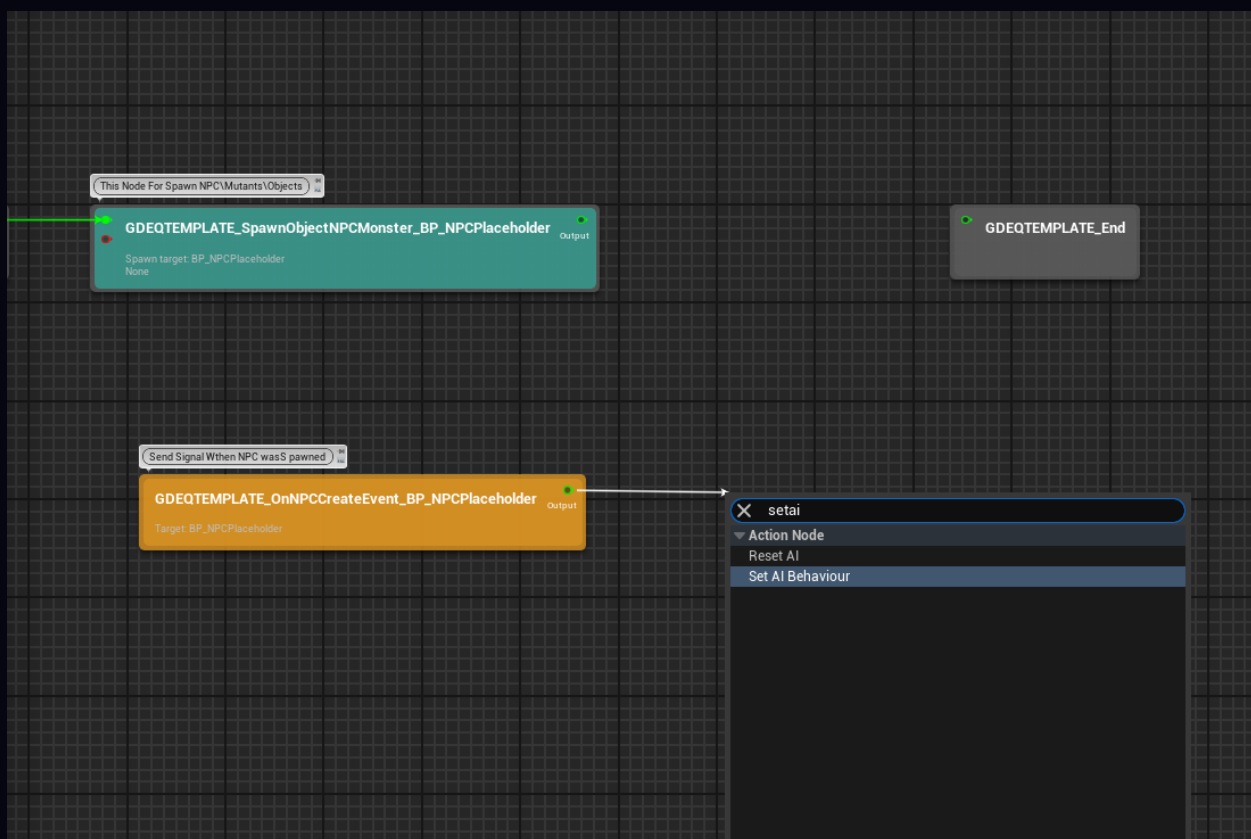
- Right-click inside the quest blueprint and select **On NPC Create Event** to receive a signal that the character has spawned.



- Use the pipette tool to select the spawn (single or group) previously placed on the map to link it to the actor placed on the map. This way, the node will refer to a specific actor in the world.

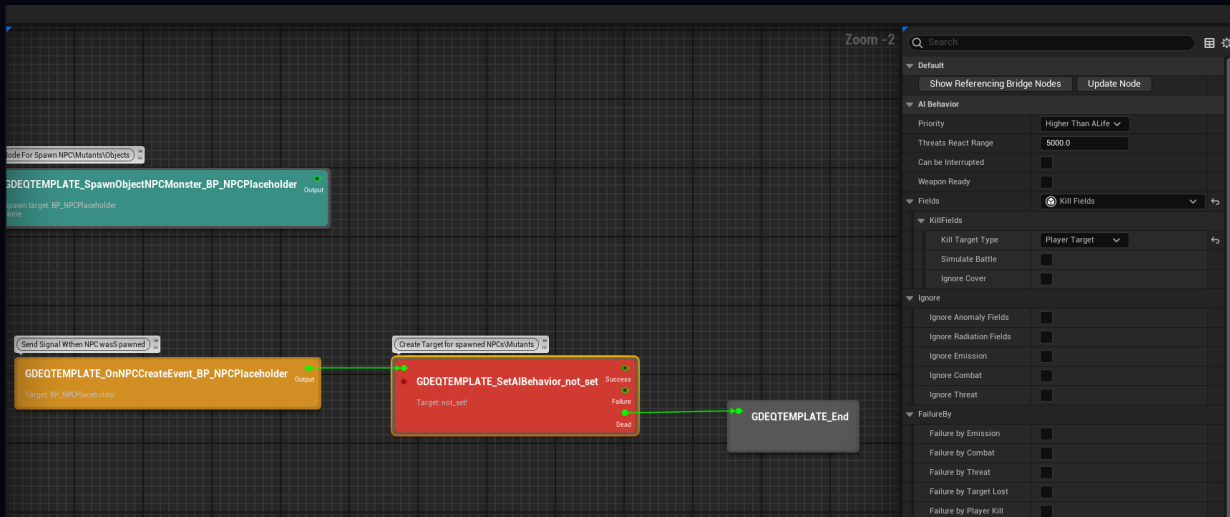


7. Right-click inside the quest blueprint and select **Set AI Behaviour** to set the behavior of the spawned character.

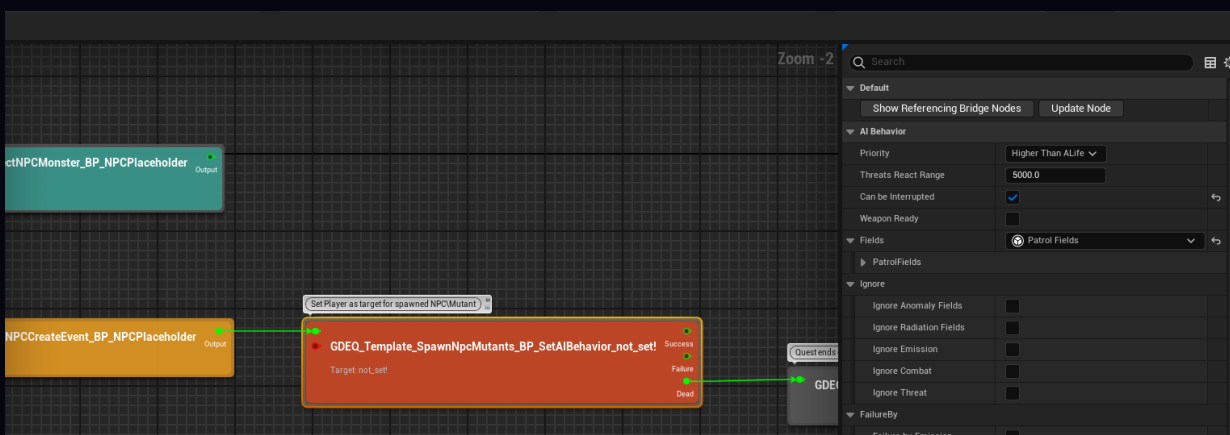


To connect the AI behavior setting node to the node that receives the signal when the character is spawned, do the following:

- If you need spawned characters to attack the player, select **Player Target** in **Kill Target Type**.
 - When setting the AI Behavior node, be sure to set the target placeholder of the NPC/mutant that will perform the behavior.
 - With this setting, spawned characters will immediately know where the player is and will not lose sight of them until they die.
 - Use it at short range.



- If you need spawned characters to attack the character more accurately, select **Patrol Fields in Fields**.
 - With this setting, spawned characters will honestly notice the player.
 - Use it at medium range.

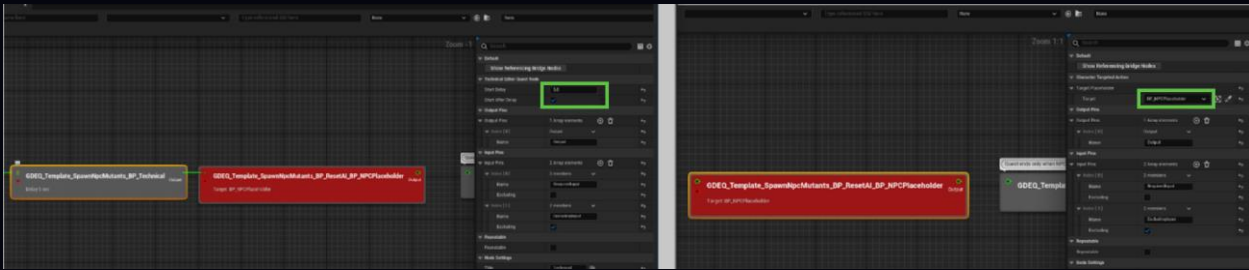


- If you need spawned characters to react according to their visual/auditory/olfactory sensors, do not select anything in **Kill Target Type**. This is only necessary when you spawn neutral NPCs.

To connect ResetAI, follow these steps:

1. If the player can avoid combat with a spawned character, then after spawning:
 - Add the ResetAI node in case the character does not have any special behavior.
 - Set a trigger that activates the ResetAI node in case you need the character to follow the player within the space.
 - Add a Technical node after the AI Behavior node and then connect the ResetAI node in case you need the character to run up to or target the player.

- In the Technical node, set the timer to 2-5 seconds. This limitation is necessary so that the spawned character has time to reach the player or a convenient position for attack. After that, their senses will work correctly on their own.



2. Connect the Set AI Behaviour\ResetAI node to the End node of the quest blueprint. Connecting the node with a spline is only allowed from the Dead pin, as shown in the screenshot below.

Example of GDEQ implementation on WorldMap: *GDEQ_Template_SpawnNPCMutants_BP*



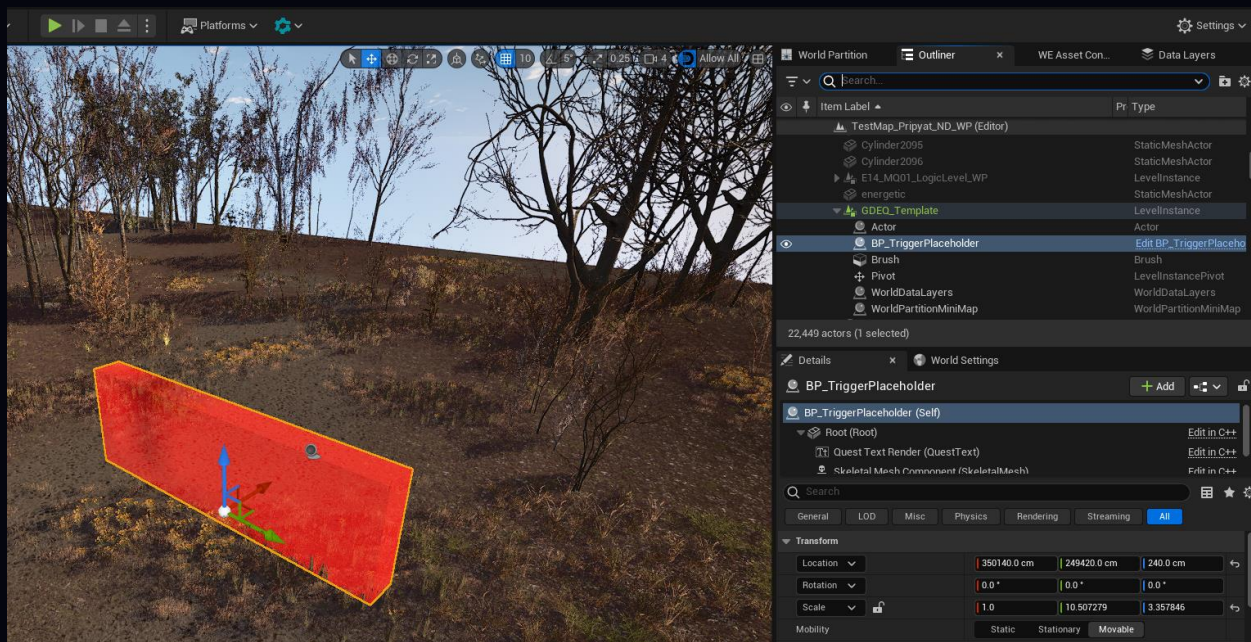
Object spawn

The path to the folder where the trigger placeholder is located:

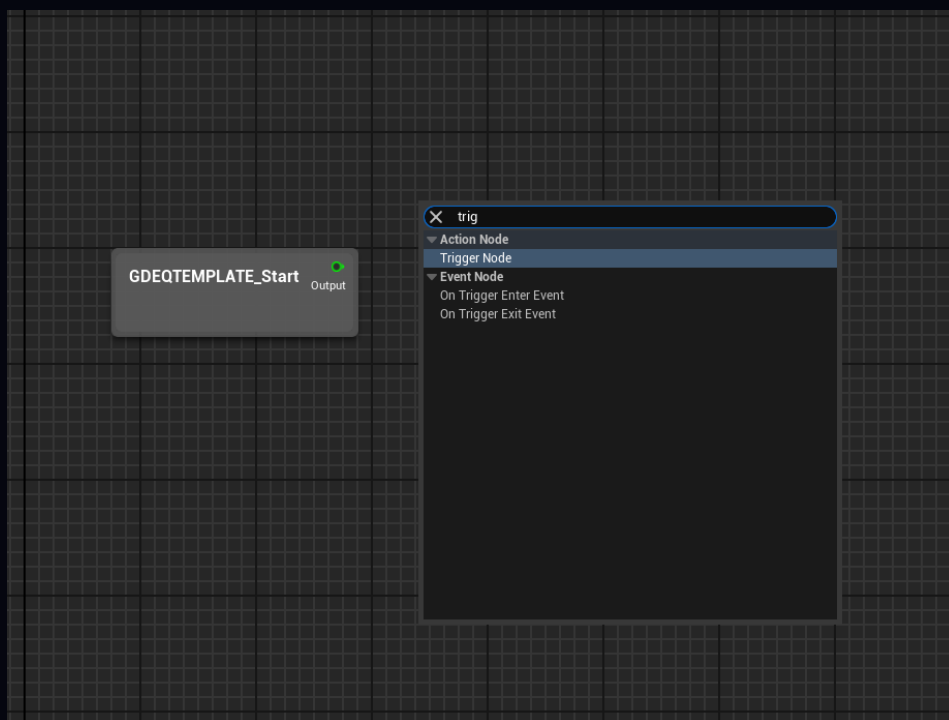
Plugins/GameDataEditorContent. Also, you can find it by the name of the actor: *BP_TriggerPlaceholder*.

To spawn a mutant/\NPC using a script, follow these steps:

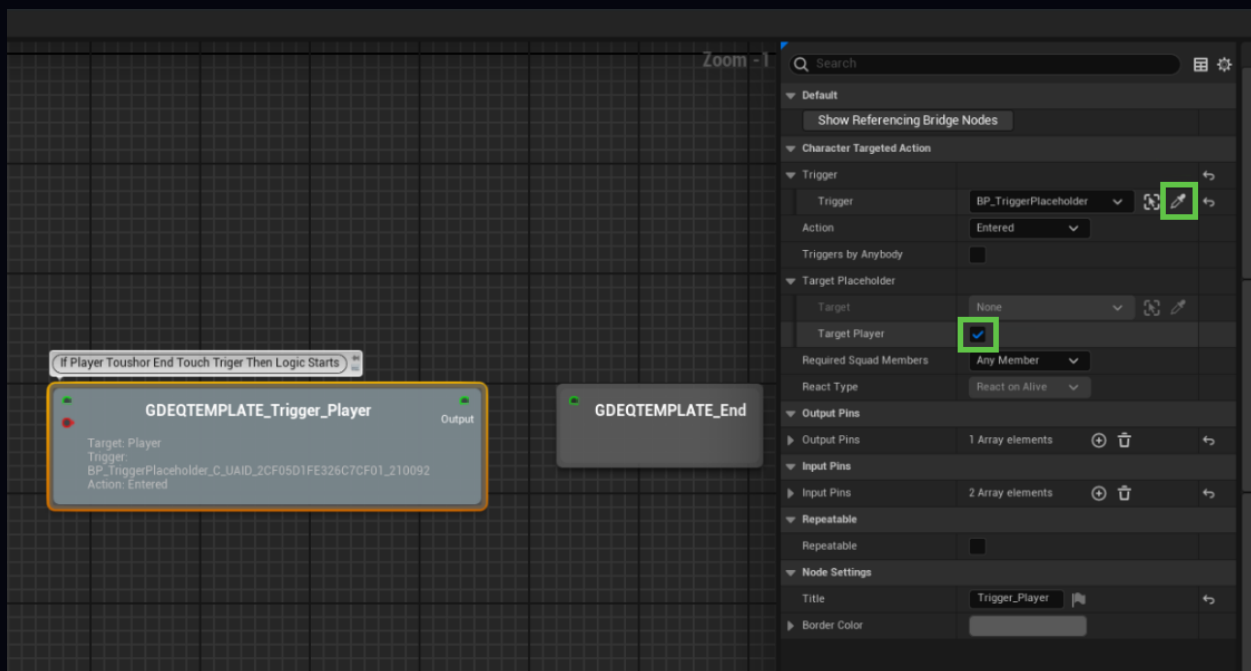
1. Drag the trigger from the context browser to the stage and stretch it over the desired area.
 - The volume thickness must be no less than 100 units (1 meter) and no more than 200 units (2 meters).
 - The width and height depend on the environment.
 - The volume should not extend beyond the boundaries of the room or opening in which it is installed so that it cannot be activated through walls, floors, and ceilings.



2. Right-click inside the quest blueprint and select **Trigger Node** to connect the trigger to your quest.



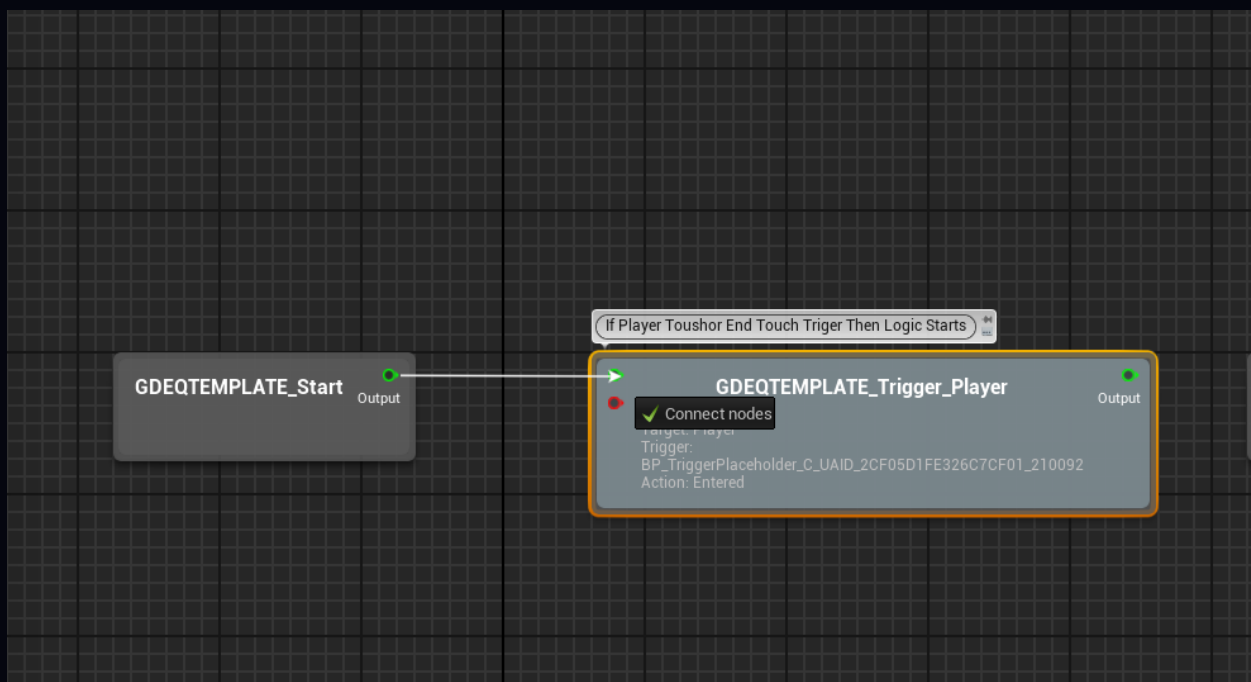
3. Set up the trigger:
 - Use the pipette tool to select the previously set trigger volume. This way, the node will refer to a specific actor in the world.
 - Select the **Target Player** checkbox.



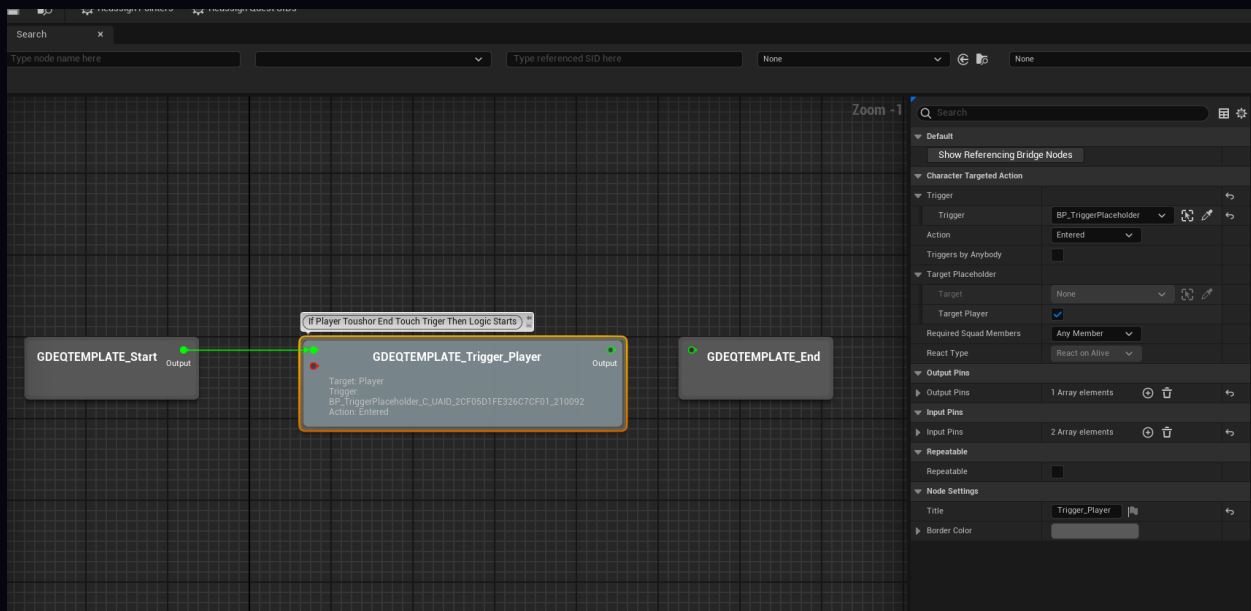
4. Extend the spline from the green start pin of the quest to the node with the trigger. Do this for each node.

IMPORTANT

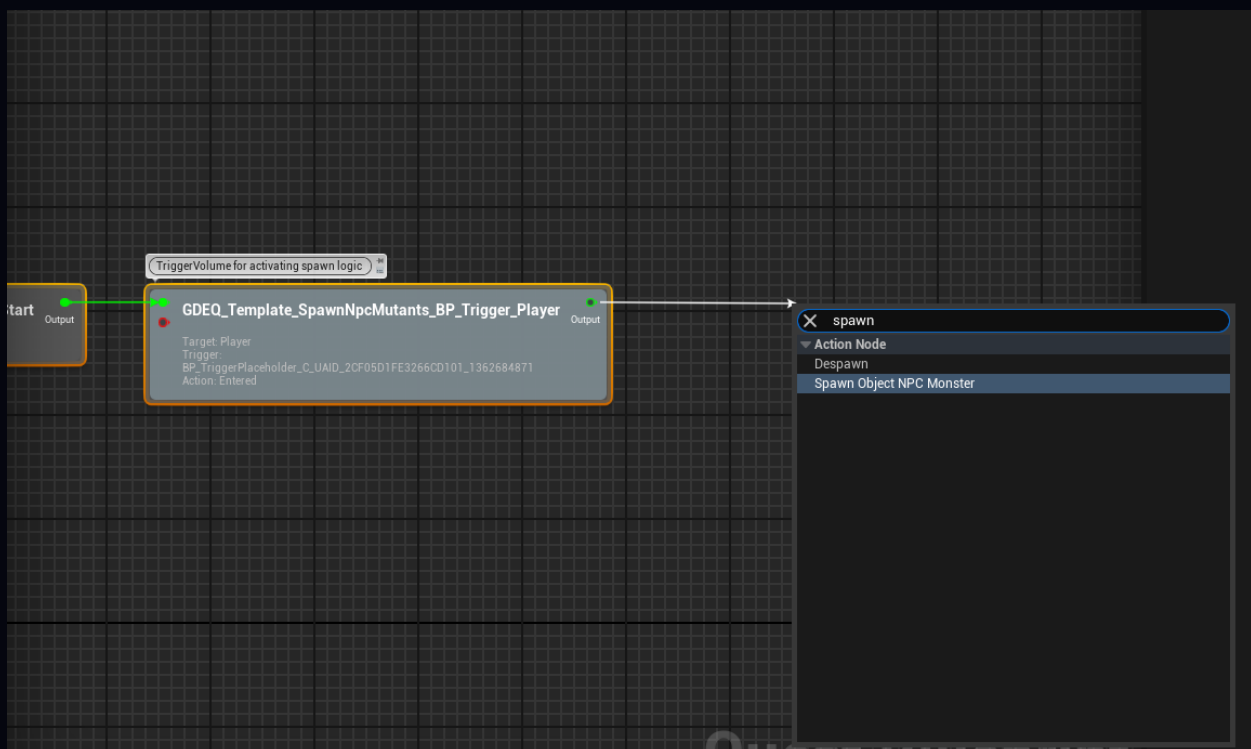
Please be careful when monitoring where you are dragging from and where you are inserting the spline.



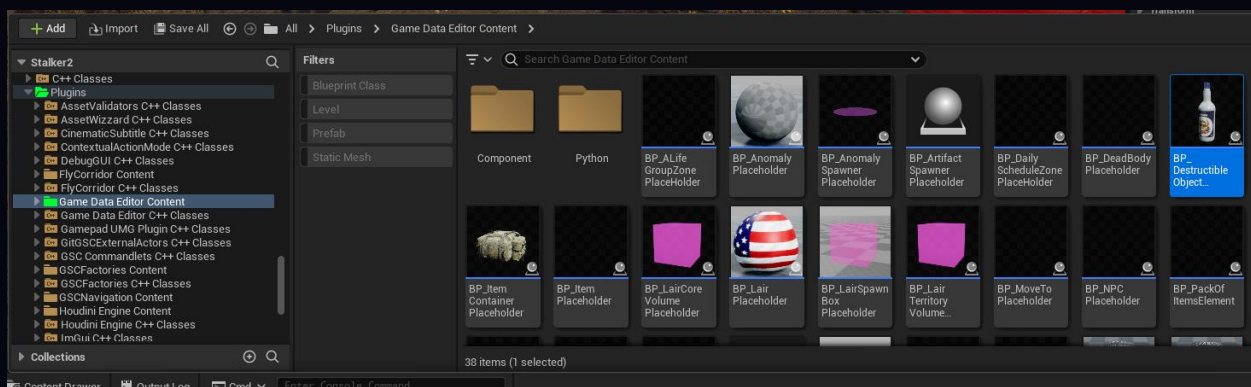
The final result should look like this:



- Right-click inside the quest blueprint and select **Spawn Object NPC Monster** to add a node for spawning objects.

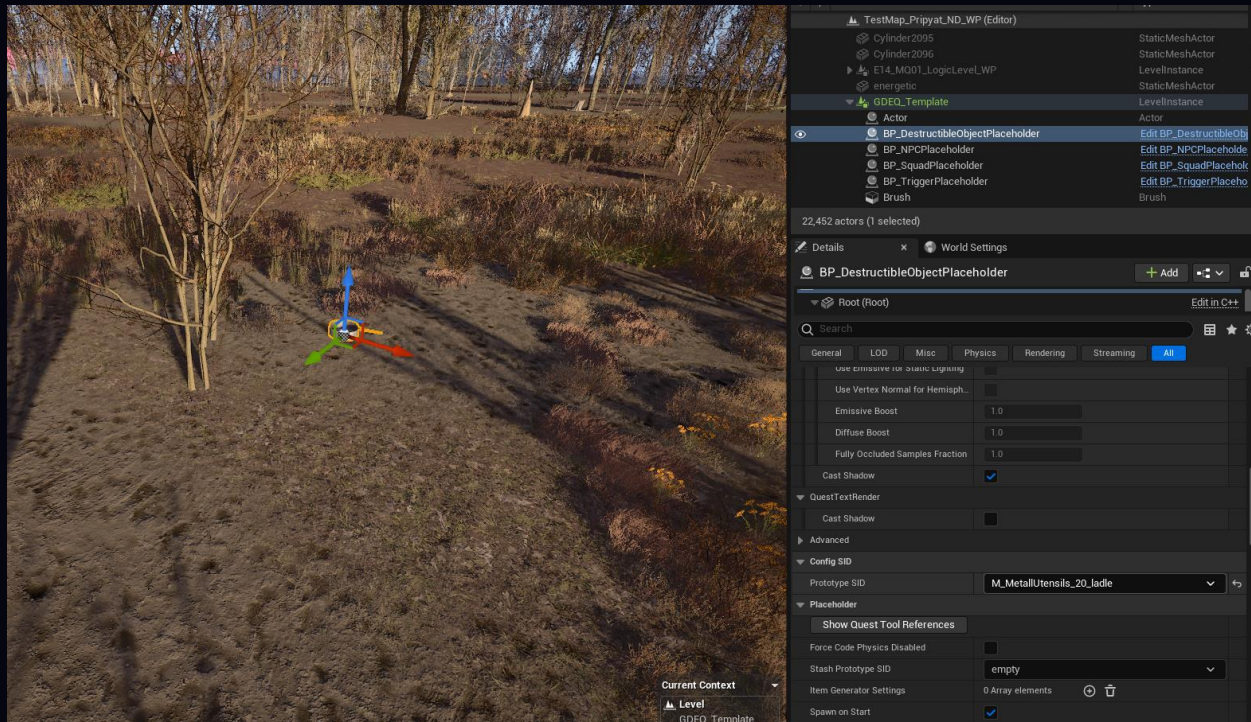


You can find the spawn actors for destructible objects in *Plugins/GameDataEditorContent*. Also, you can find it by the name of the actors: *BP_DestructibleObjectPlaceholder*.

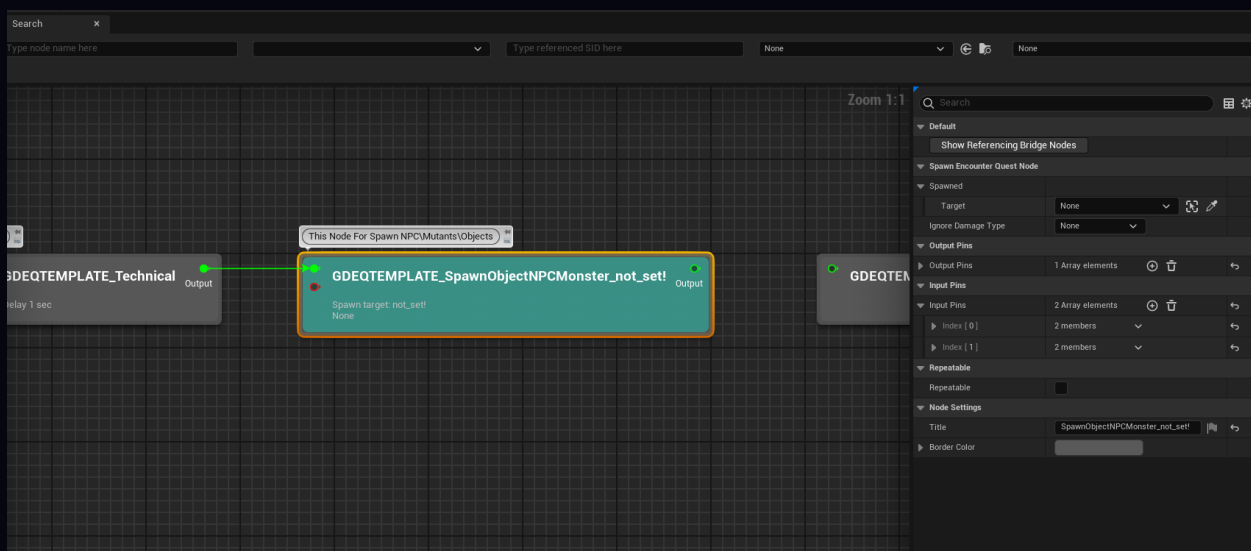


To spawn one object, do the following:

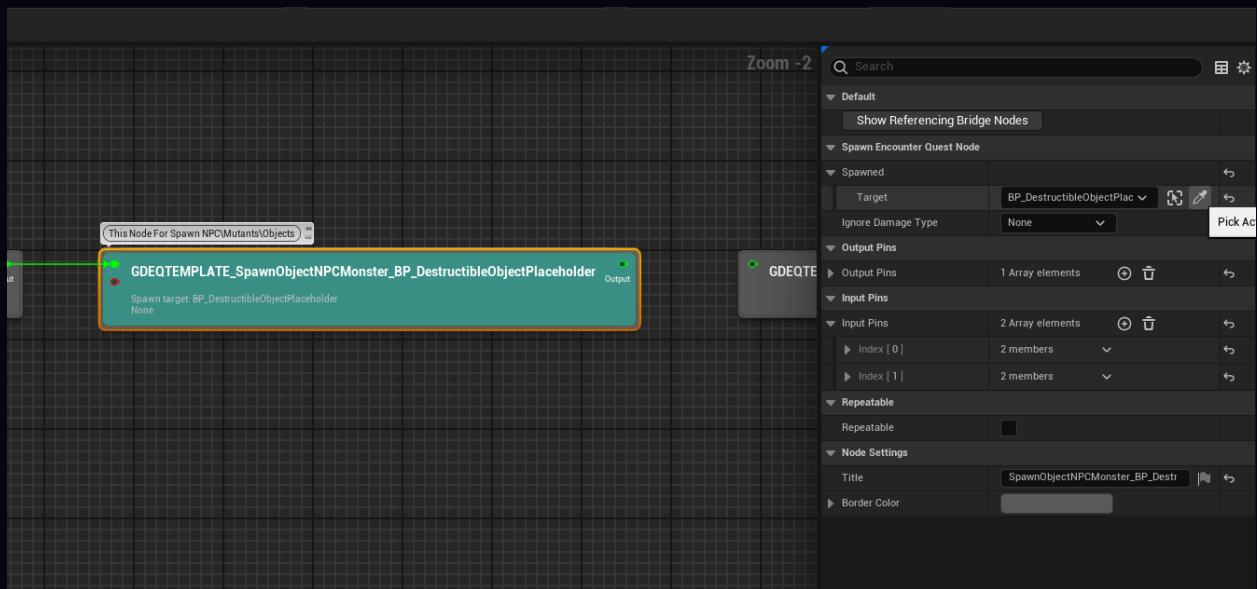
1. Place *BP_DestructibleObjectPlaceholder* on the stage.
2. Specify the SID of the object you want to spawn.
3. Clear the **Spawn On Start** checkbox.



4. Use the pipette tool to select the actor of the destructible object that was previously placed on the map. This way, the node will refer to a specific actor in the world.



5. Connect the *BP_DestructibleObjectPlaceholder* node to the End node of the quest blueprint.



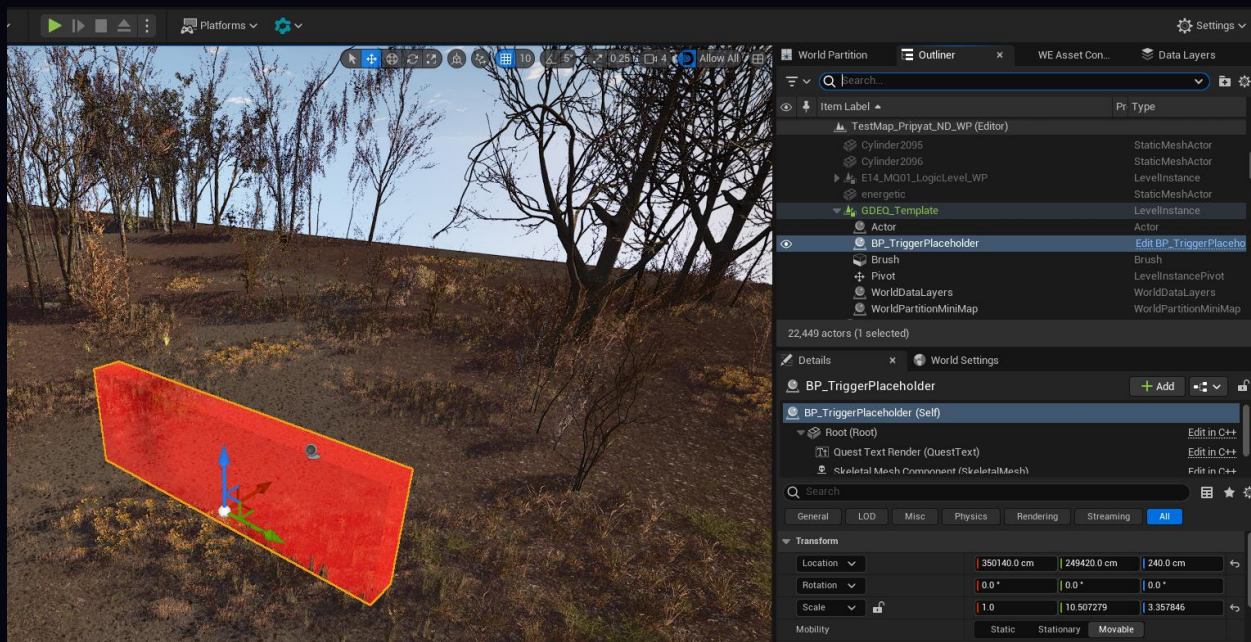
Example of GDEQ implementation on WorldMap: *GDEQ_Template_SpawnObject_BP*



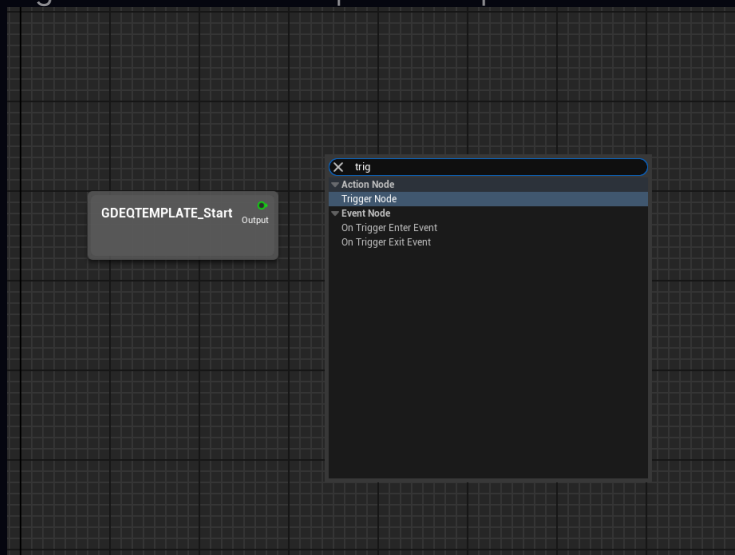
Sending/receiving signal to interact

You have already placed interactive objects on the map and configured the connections between them. Now, to activate the interactive object using a script, follow these steps:

1. Find the trigger to activate the script – the trigger placeholder is located here: *Plugins/GameDataEditorContent*. Also, you can find it by the name of the actor: *BP_TriggerPlaceholder*.
2. Drag the trigger from the context browser to the stage and stretch it over the desired area to mark the trigger for the script activation.
 - The volume thickness must be no less than 100 units (1 meter) and no more than 200 units (2 meters).
 - The width and height depend on the environment.
 - The volume must not extend beyond the boundaries of the room or opening in which it is installed so that it cannot be activated through walls, floors, and ceilings.

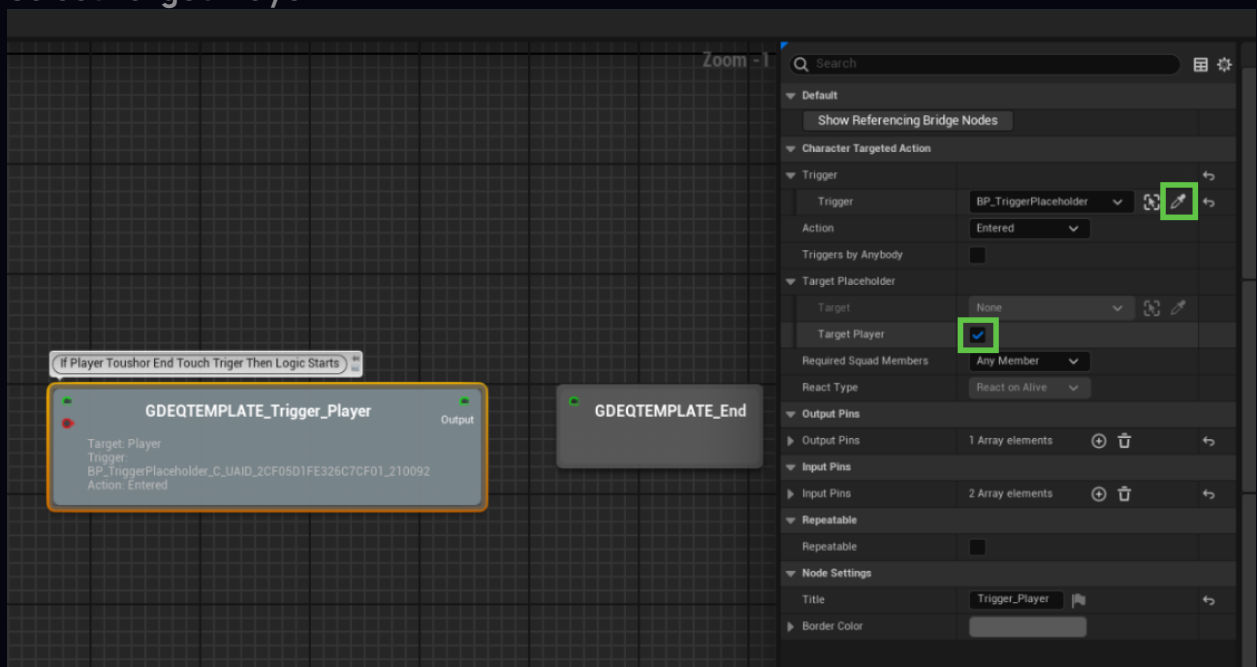


3. Right-click inside the quest blueprint and select **Trigger Node**.



4. Use the pipette tool to select the previously set trigger volume. This way, the node will refer to a specific actor in the world.

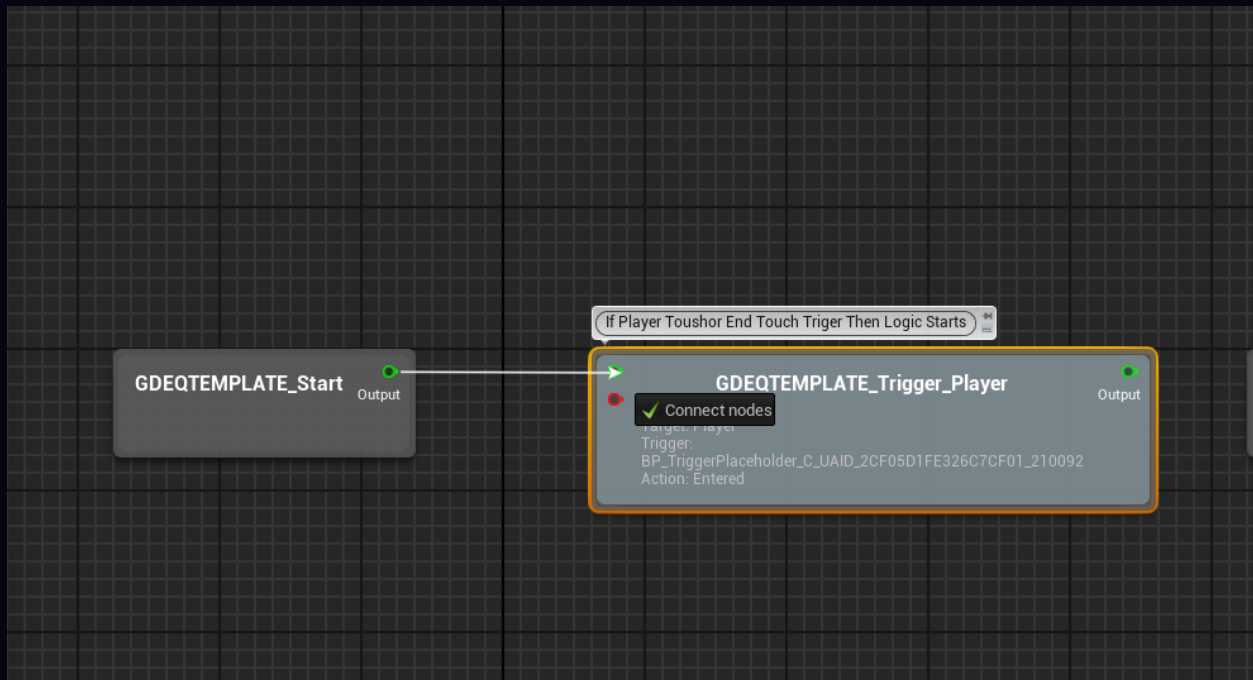
5. Select **Target Player**.



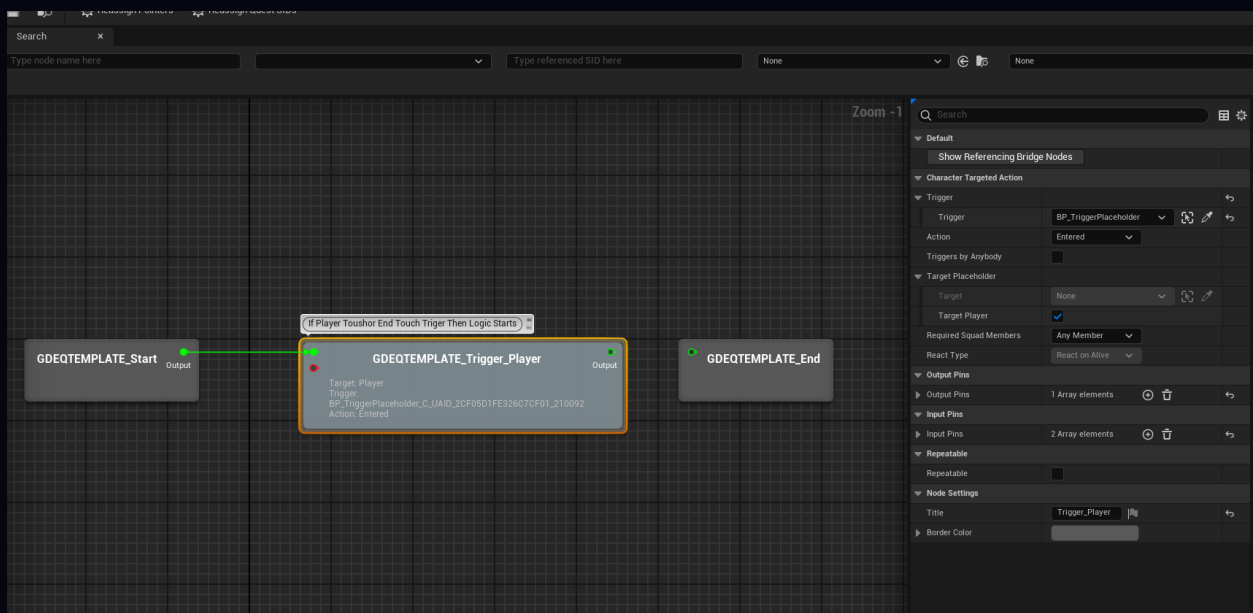
- Pull a spline from the green start pin of the quest to the node with the trigger. Do this for each node.

IMPORTANT

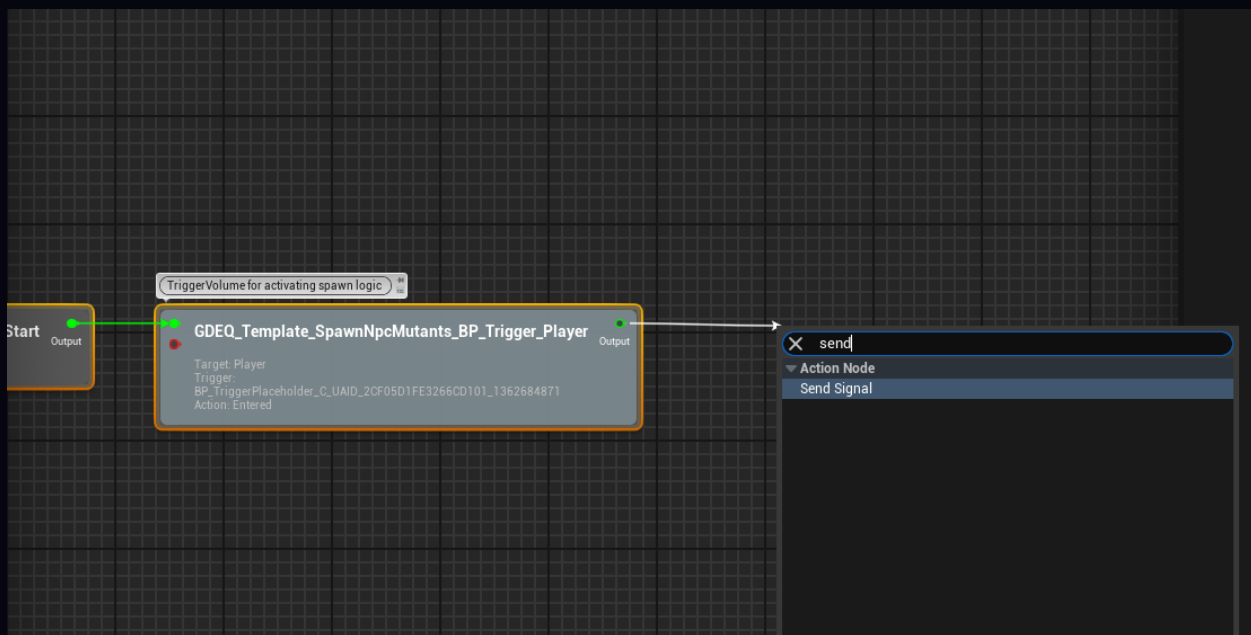
Please be careful when monitoring where you are dragging from and where you are inserting the spline.



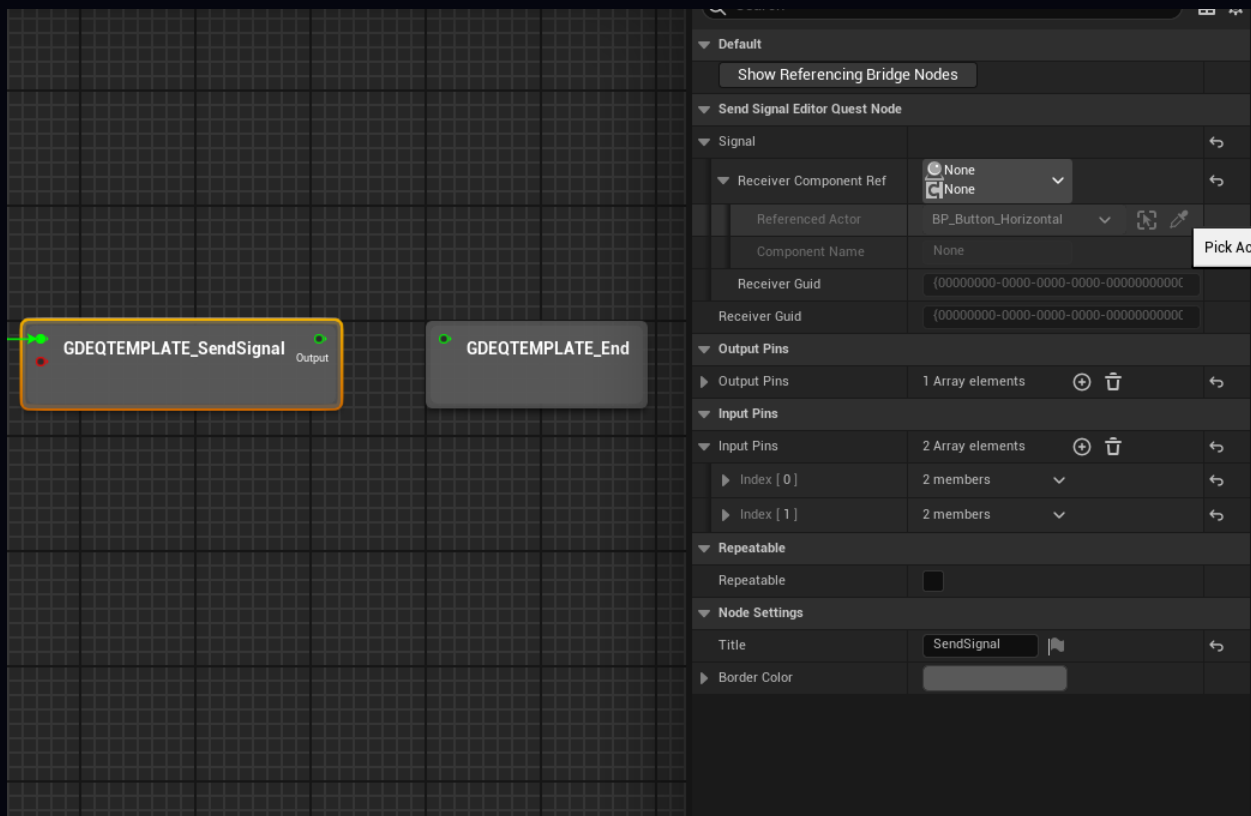
The final result should look like this:



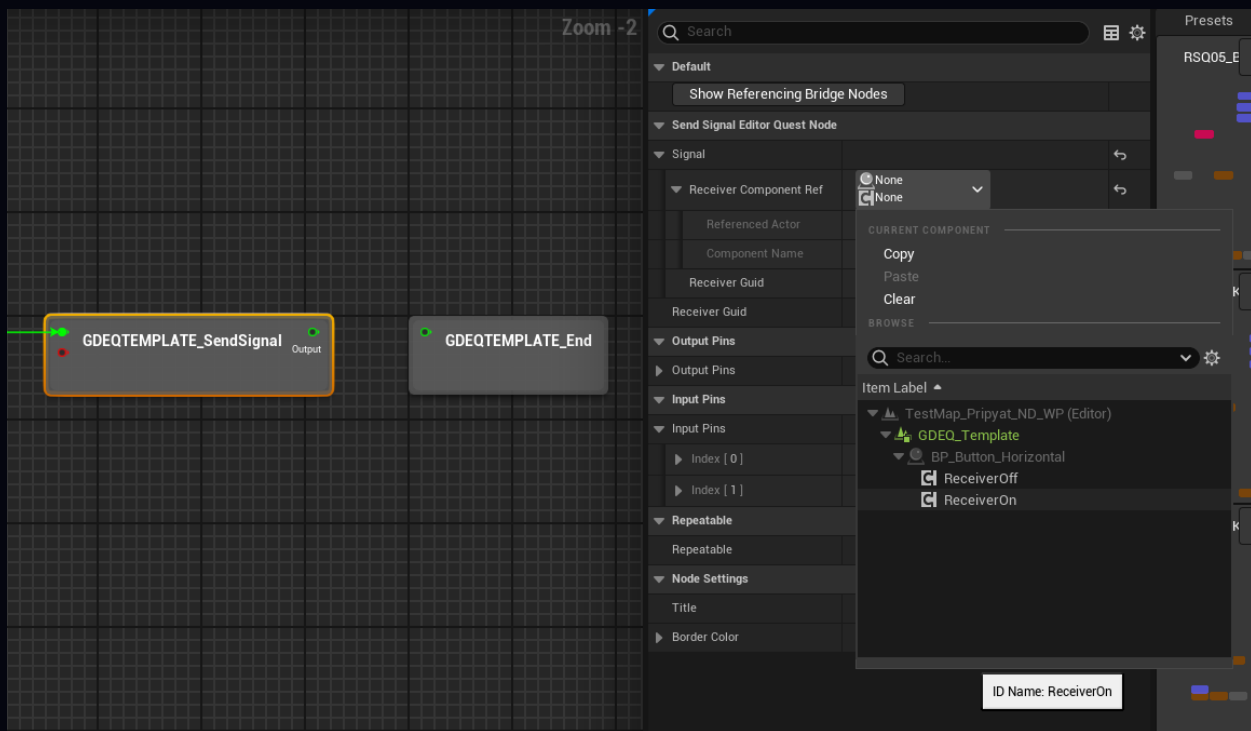
- Right-click inside the quest blueprint and select **Send Signal** to send a signal to an interactive object.



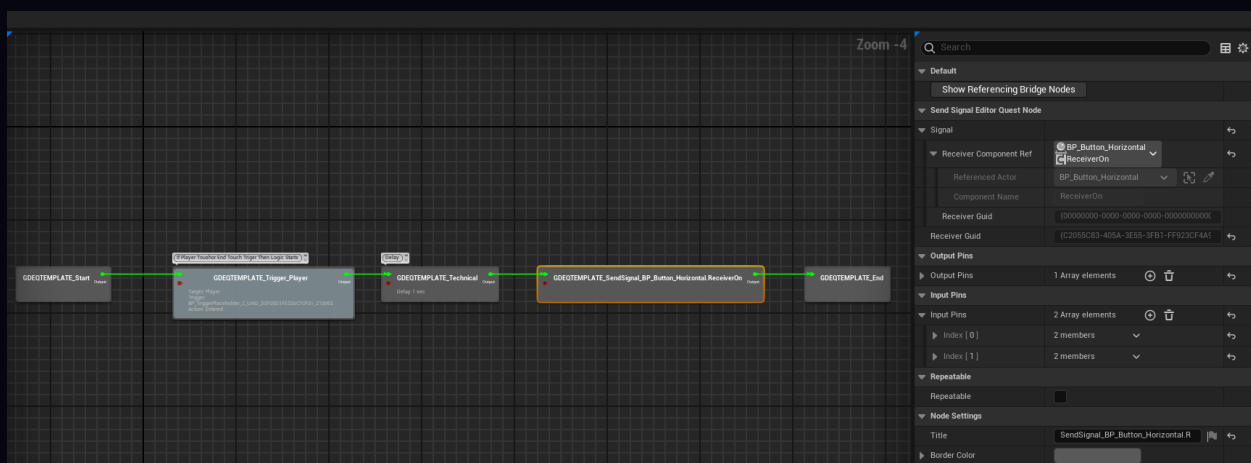
8. Use the pipette tool to select the interactive object actor that was previously placed on the map. This way, the node will refer to a specific actor in the world.



9. Open the menu as shown in the screenshot below and select the required item.
10. Connect the Send Signal node to the End node for the script to complete successfully when the Send Signal node finishes working.

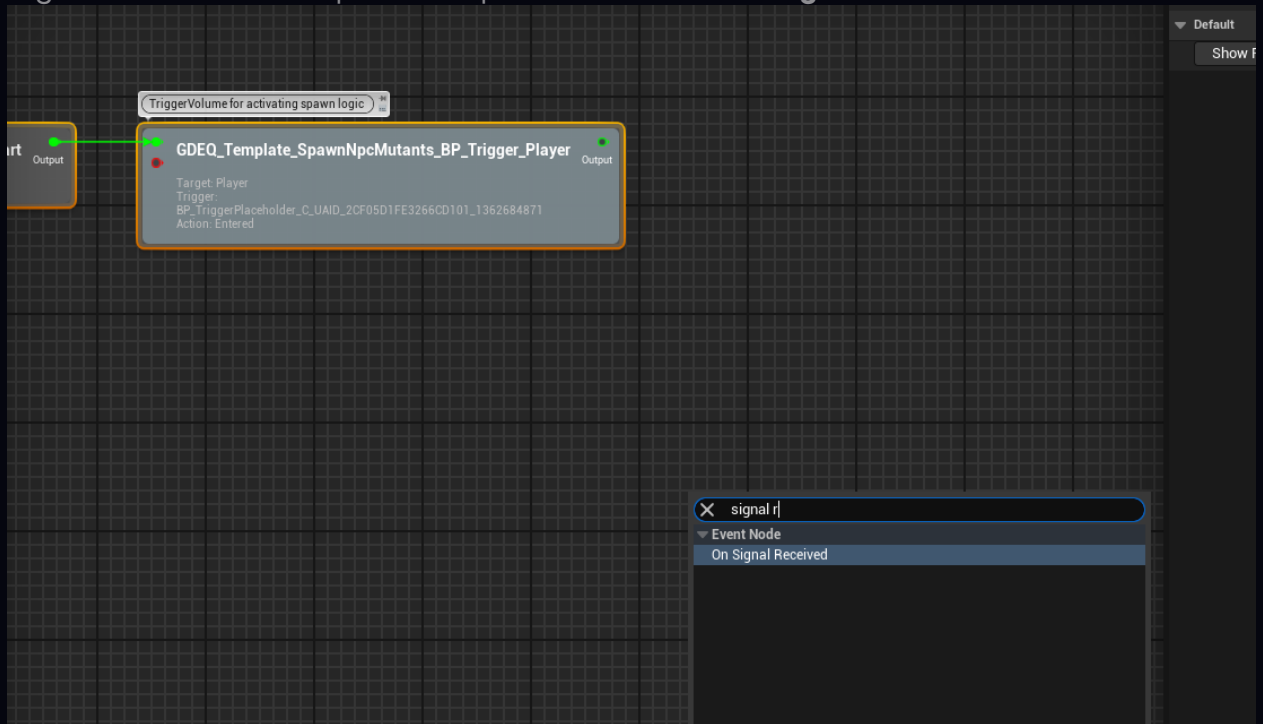


Your settings and blueprint should look exactly as shown in the screenshot below. Please pay attention to the Send Signal node settings.

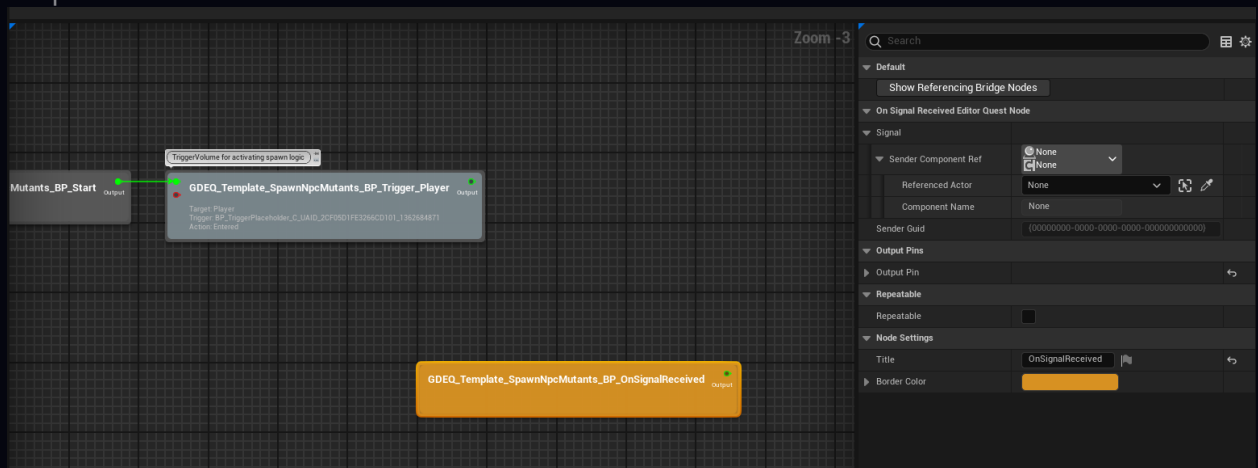


Let's take a look at an example of what to do if you need to receive a signal from an interactive object. Follow these steps:

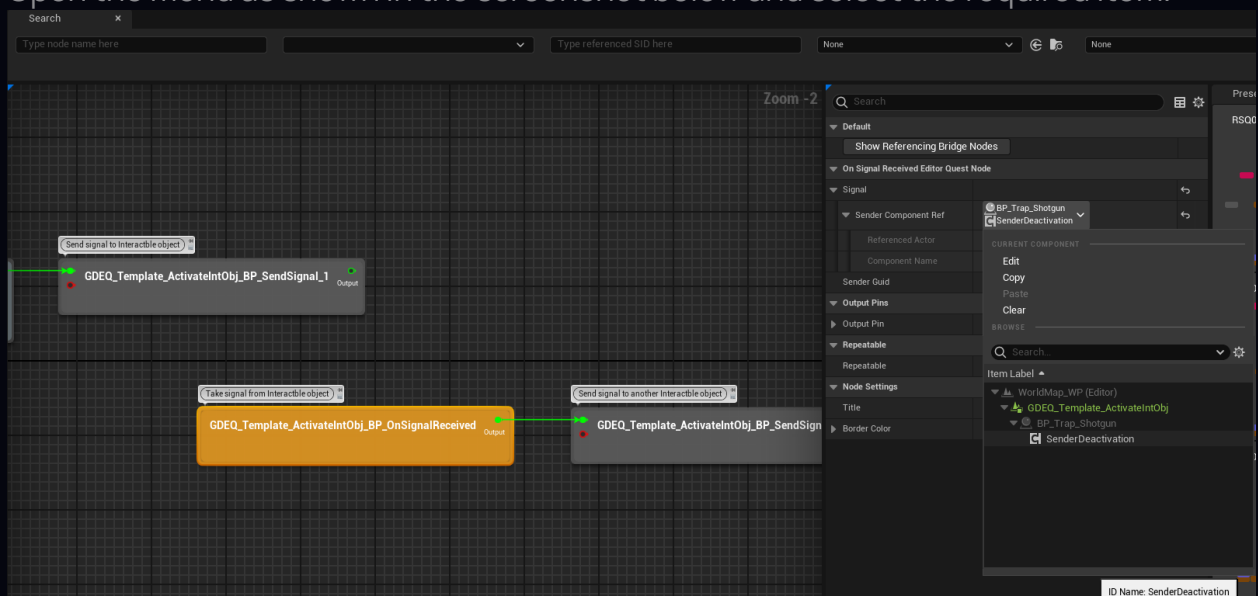
1. Right-click inside the quest blueprint and select **On Signal Received**.



2. Use the pipette to select the interactive object actor that was previously placed on the map.



3. Open the menu as shown in the screenshot below and select the required item.



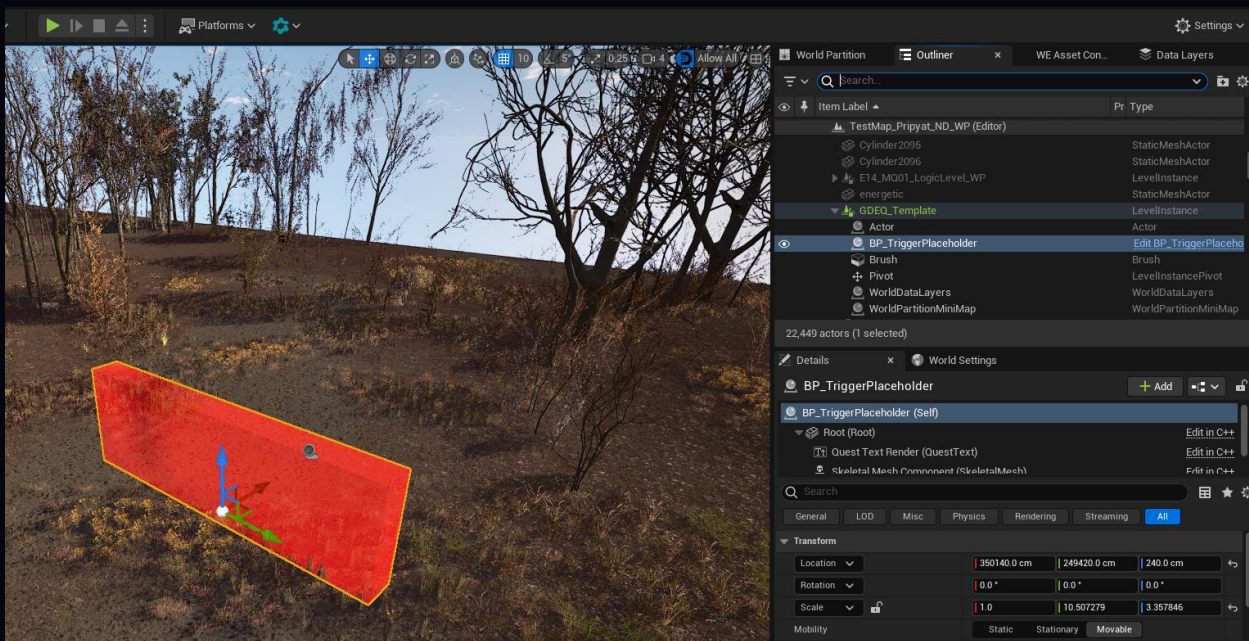
4. Connect the On Signal Received node to the End node. This will ensure that the script completes successfully when the On Signal Received node finishes working.



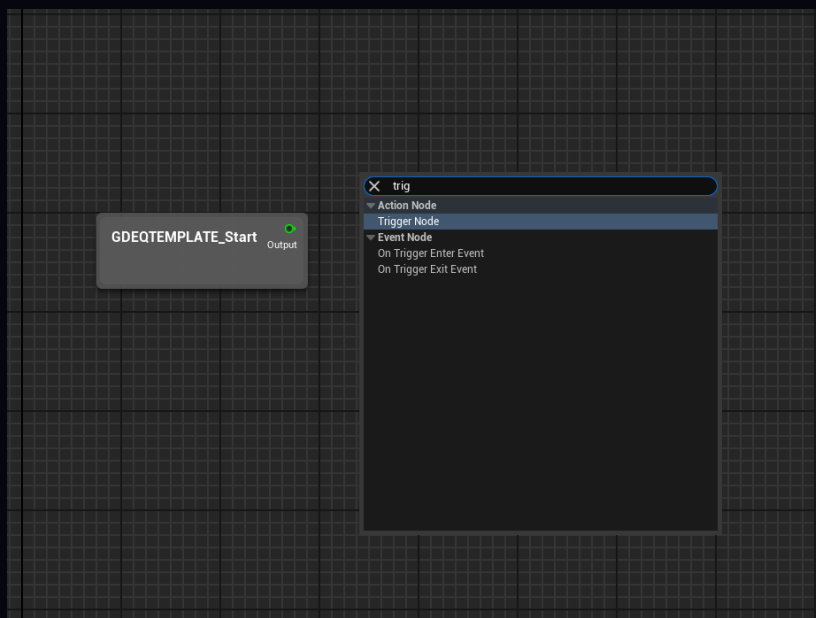
Anomaly usage

To activate the anomaly using a script, follow these steps:

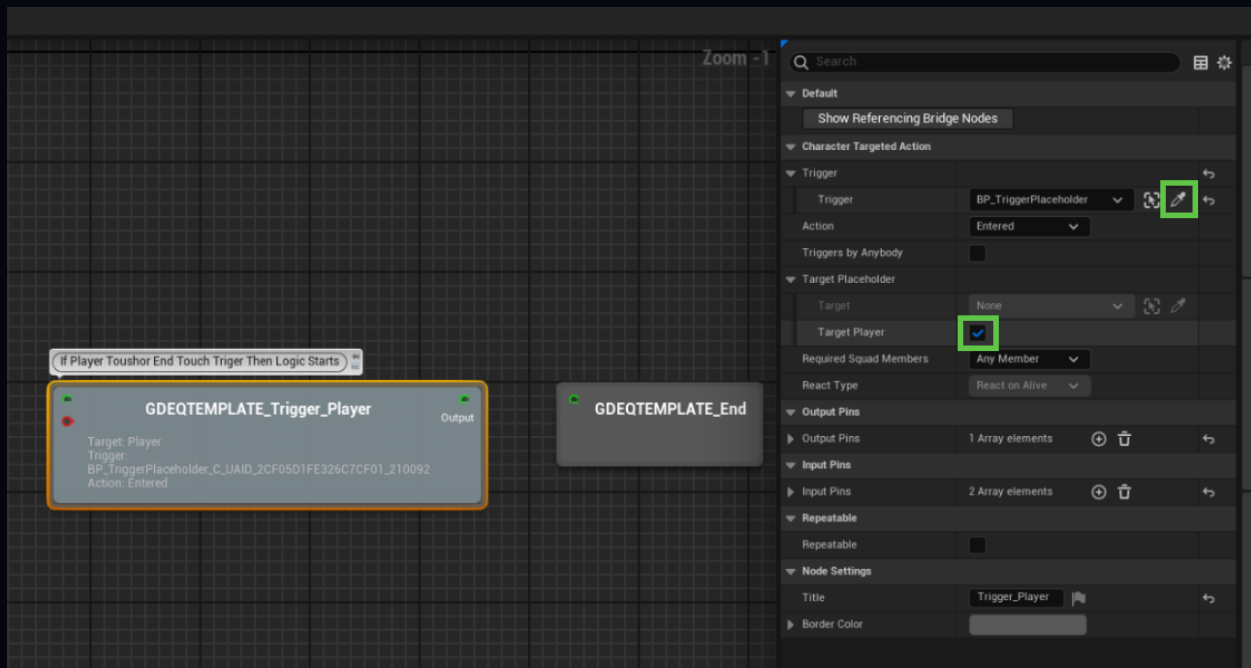
5. Find the folder where the trigger placeholder is: *Plugins/GameDataEditorContent*. Also, you can find it by the name of the actor: *BP_TriggerPlaceholder*.
6. Drag the trigger from the context browser to the stage and stretch it over the desired area.
 - a. The volume thickness must be no less than 100 units (1 meter) and no more than 200 units (2 meters).
 - b. The width and height depend on the environment.
 - c. The volume must not extend beyond the boundaries of the room or opening in which it is installed so that it cannot be activated through walls, floors, and ceilings.



7. Right-click inside the quest blueprint and select **Trigger Node**.



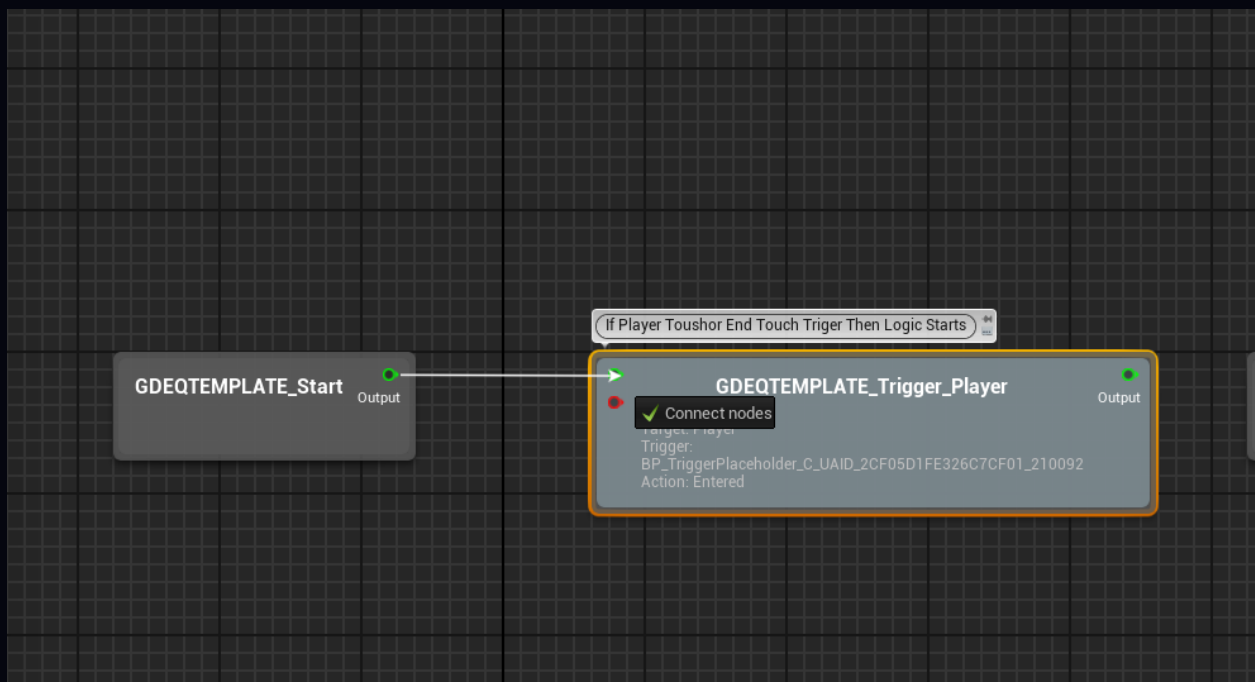
8. Use the pipette tool to select the previously set volume trigger. This way, the node will refer to a specific actor in the world.
9. Select **Target Player**.



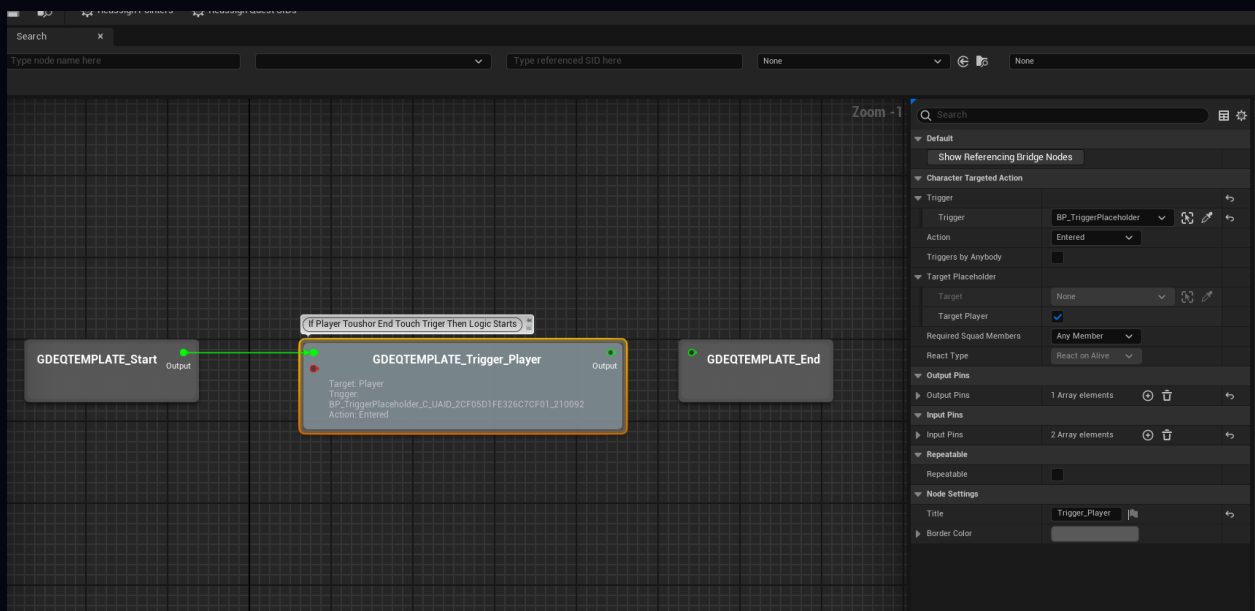
10. Drag a spline from the green quest start pin to the node with the trigger, as shown in the screenshot below. Do this for each node.

IMPORTANT

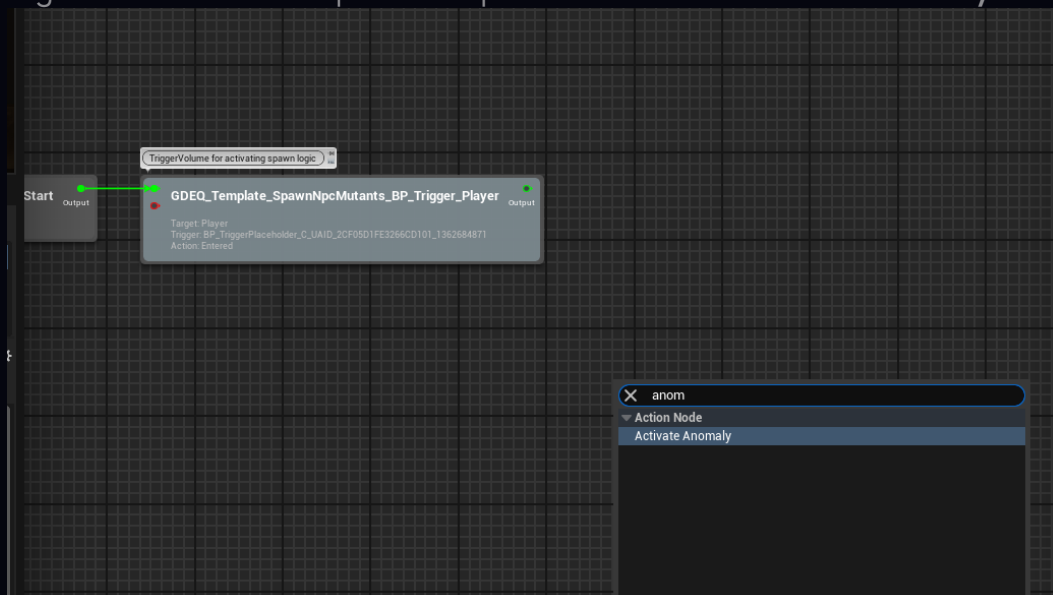
Please be careful when monitoring where you are dragging from and where you are inserting the spline.



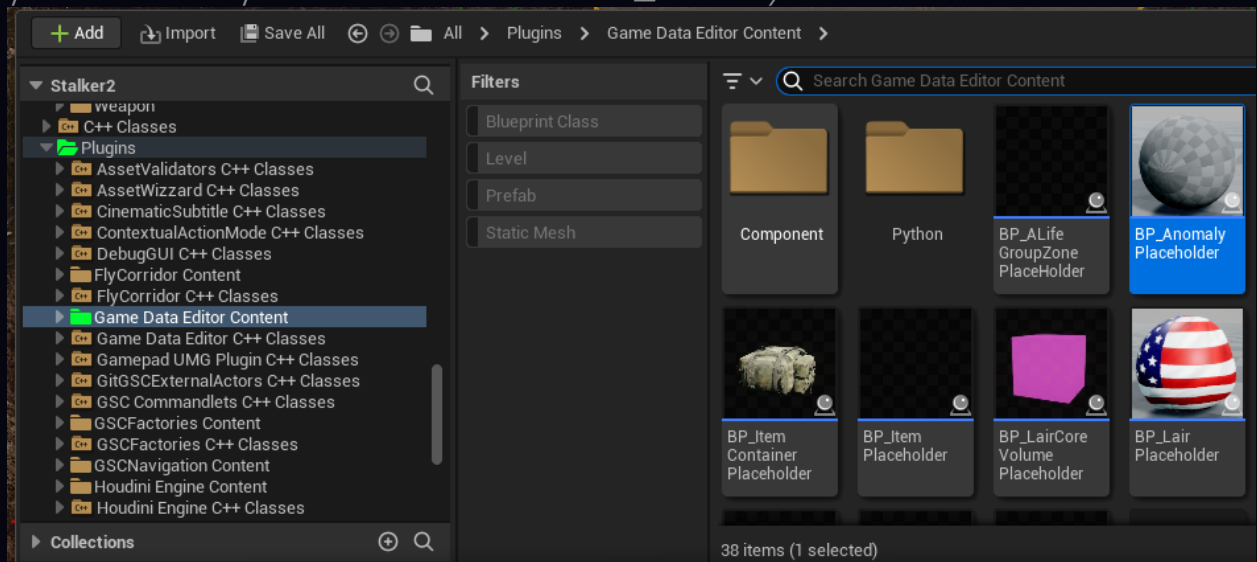
The final result should look like this:



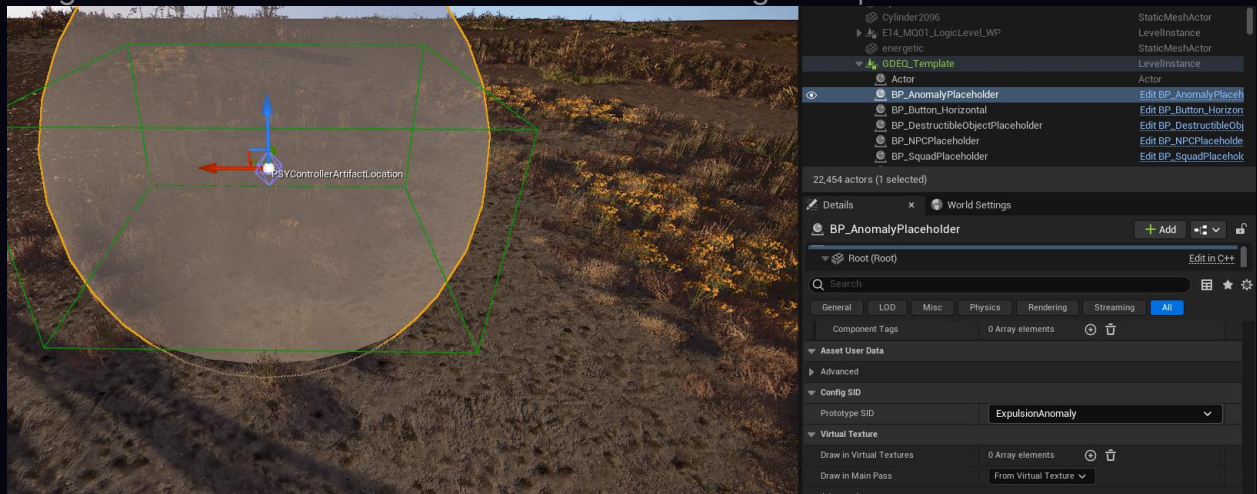
11. Right-click inside the quest blueprint and select **Activate Anomaly**.



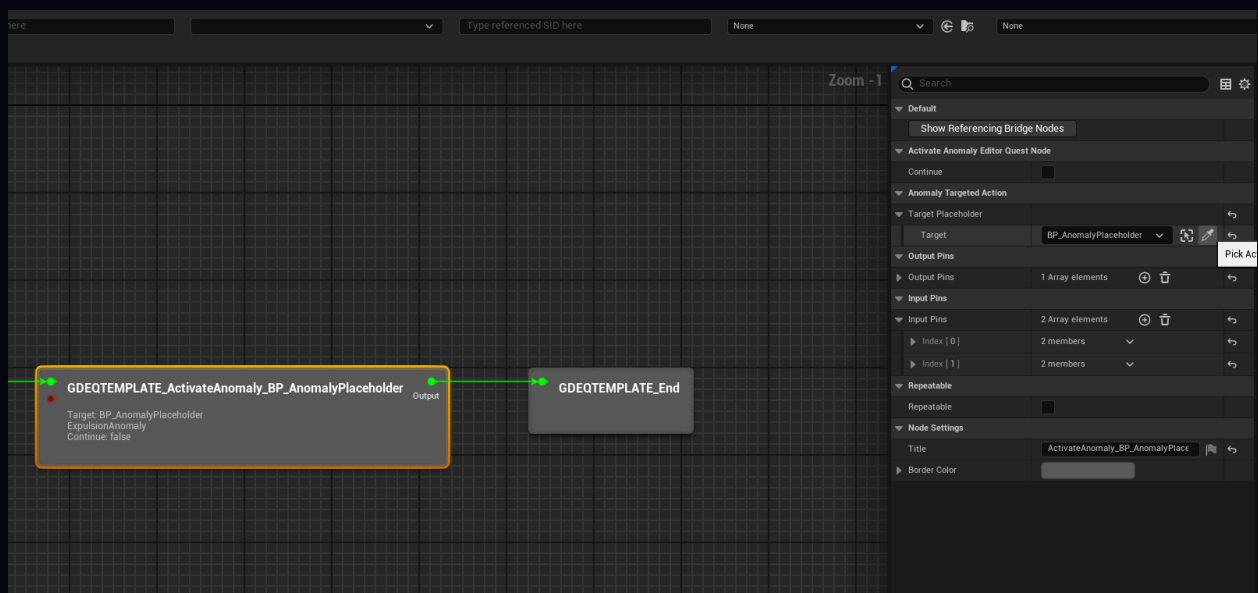
12. Find the folder where the trigger placeholder is is: *Plugins/GameDataEditorContent*. Also, you can find it by the name of the actor: *BP_AnomalyPlaceholder*.



13. Drag the actor from the context browser to the stage and place it in the desired area.



14. Use the pipette tool to select the anomaly actor that was previously placed on the map. This way, the node will refer to a specific actor in the world.
15. Connect the Activate Anomaly node to the End node. This will ensure that the script completes successfully when the Activate Anomaly node finishes working.



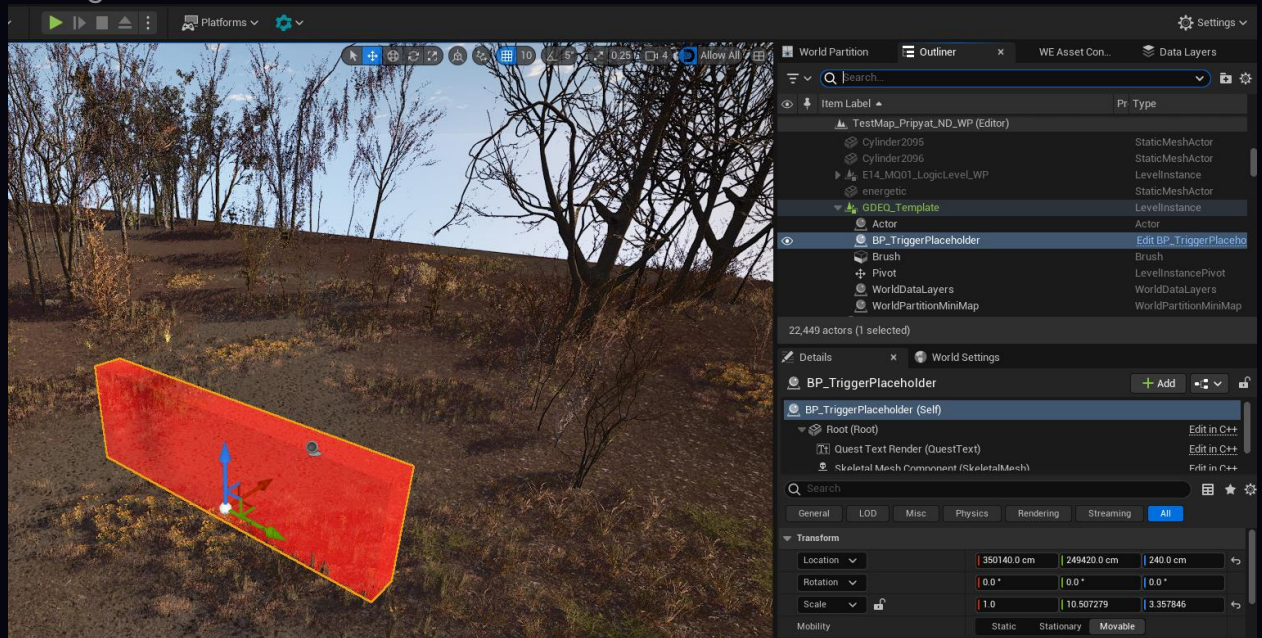
Example of GDEQ implementation on WorldMap: *GDEQ_Template_AnomalyAct_BP*



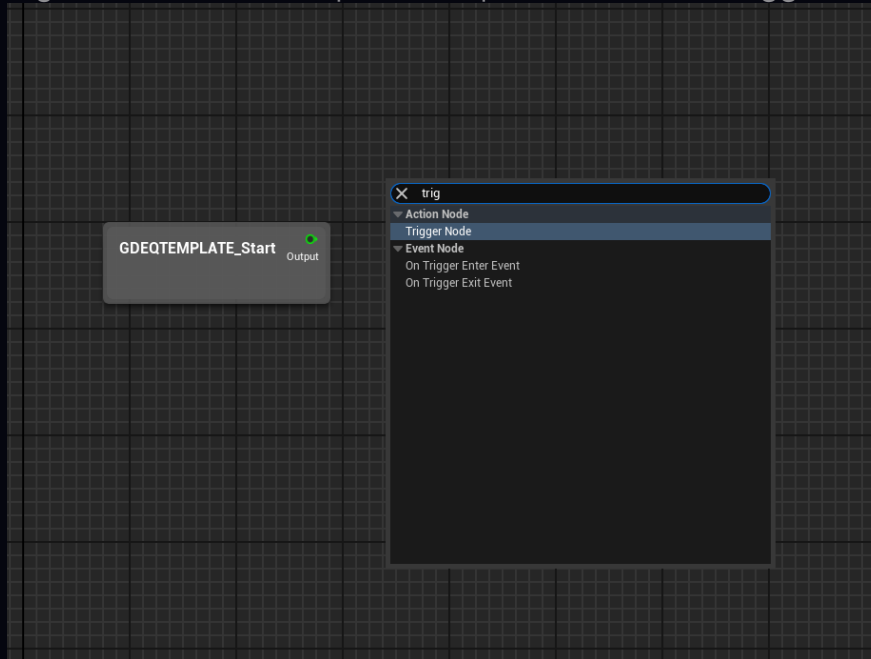
Lair state change

To activate the lair using a script, follow these steps:

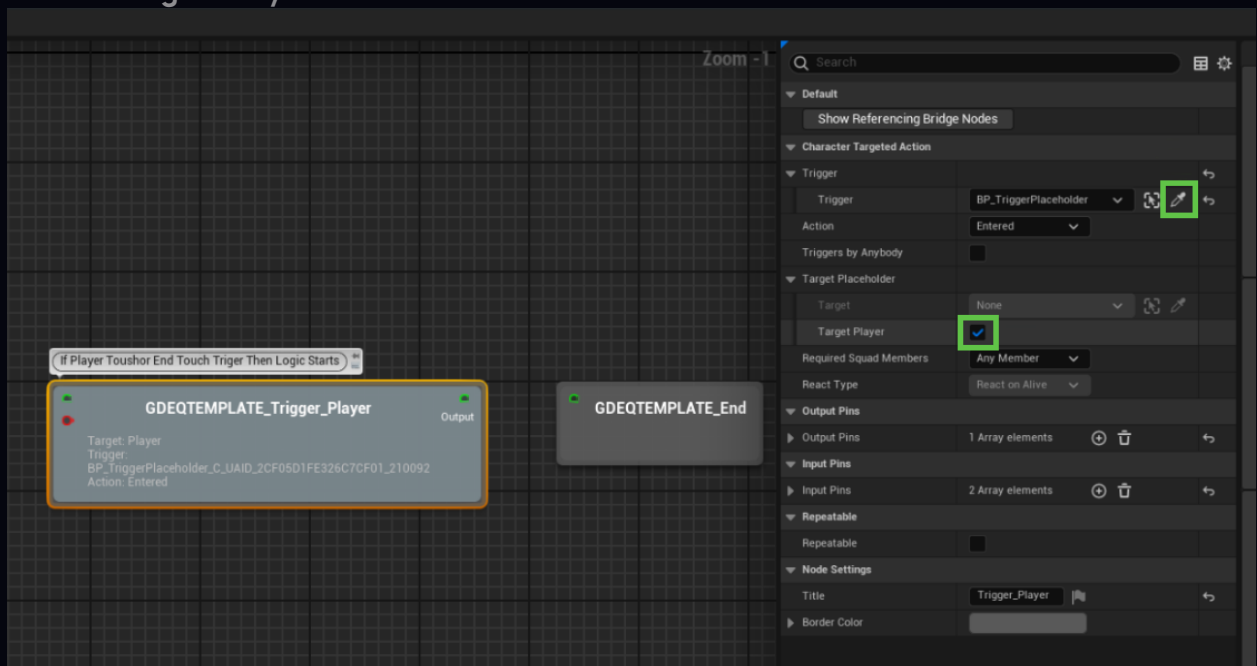
- Find the folder where the trigger placeholder is located:
Plugins/GameDataEditorContent. Also, you can find it by the name of the actor:
BP_TriggerPlaceholder.
- Drag the trigger from the context browser to the stage and stretch it over the desired area.
 - The volume thickness must be no less than 100 units (1 meter) and no more than 200 units (2 meters).
 - The width and height depend on the environment.
 - The volume should not extend beyond the boundaries of the room or opening in which it is installed so that it cannot be activated through walls, floors, and ceilings.



3. Right-click inside the quest blueprint and select **Trigger Node**.



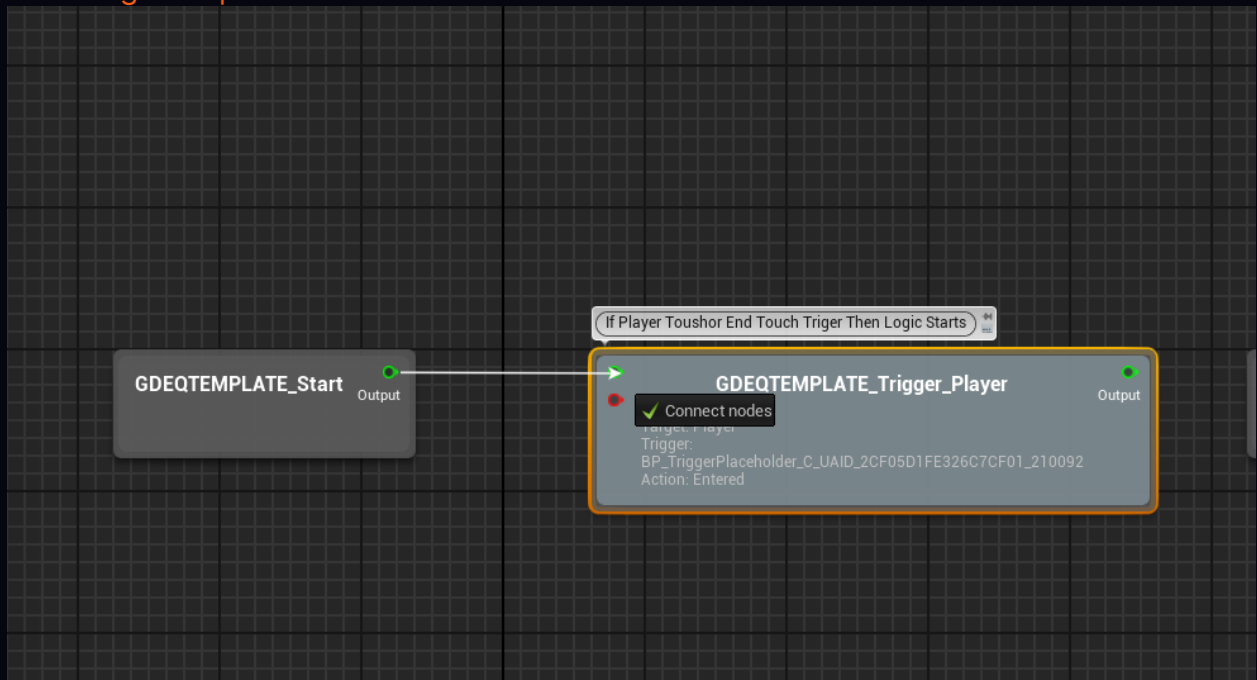
4. Use the pipette tool to select the previously set trigger volume. This way, the node will refer to a specific actor in the world.
5. Select **Target Player**.



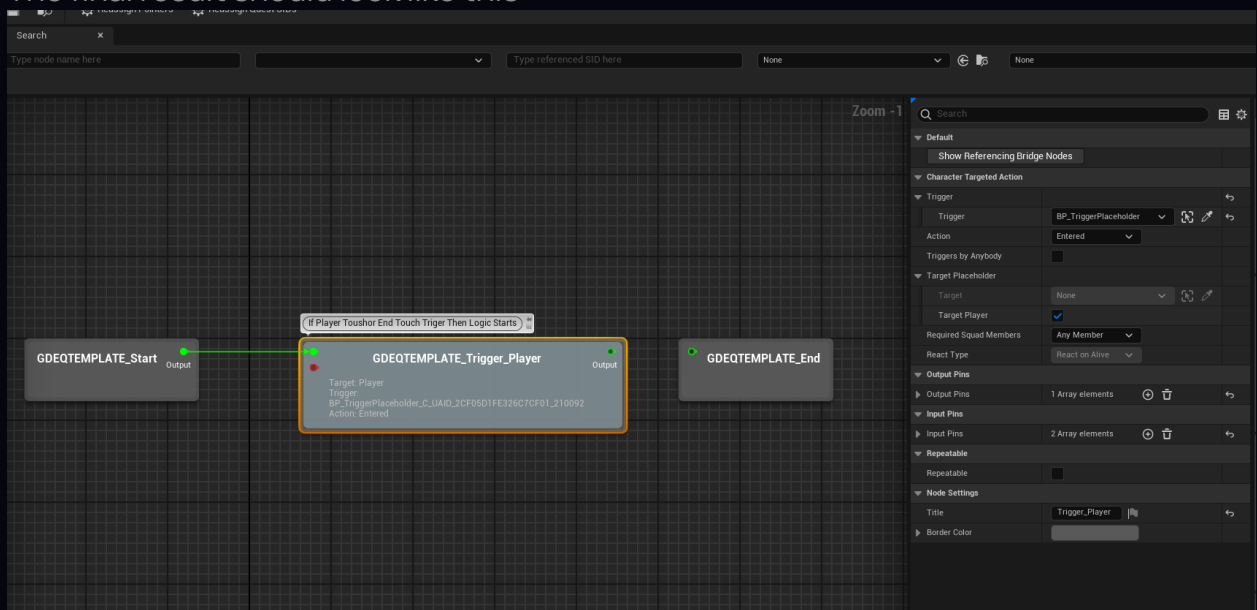
6. Drag a spline from the green start pin of the quest to the node with the trigger, as shown in the screenshot below. Do this for each node.

IMPORTANT

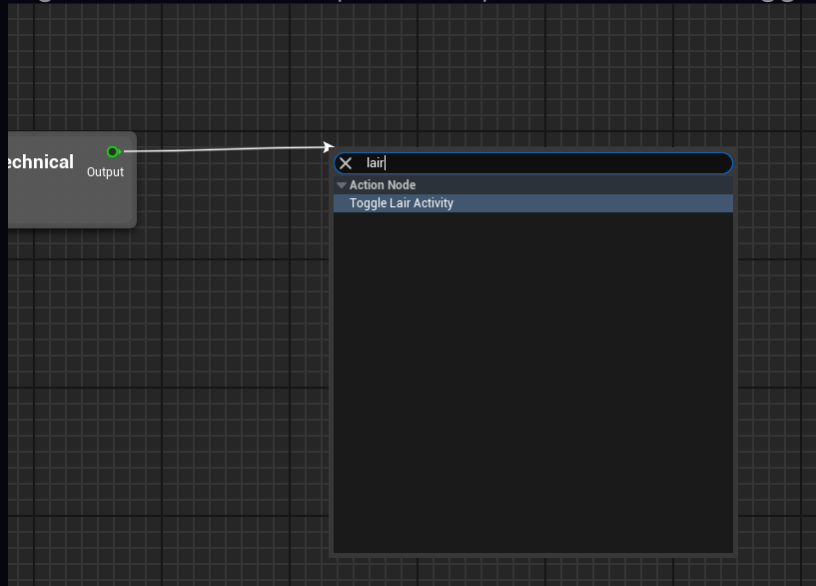
Please be careful when monitoring where you are dragging from and where you are inserting the spline.



The final result should look like this:

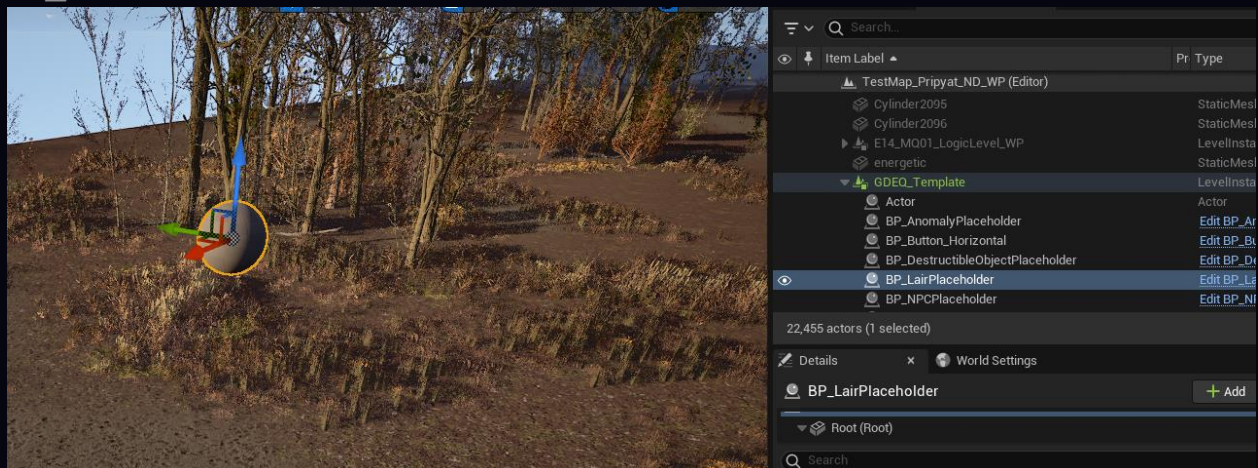


7. Right-click inside the quest blueprint and select **Toggle Lair Activity**.



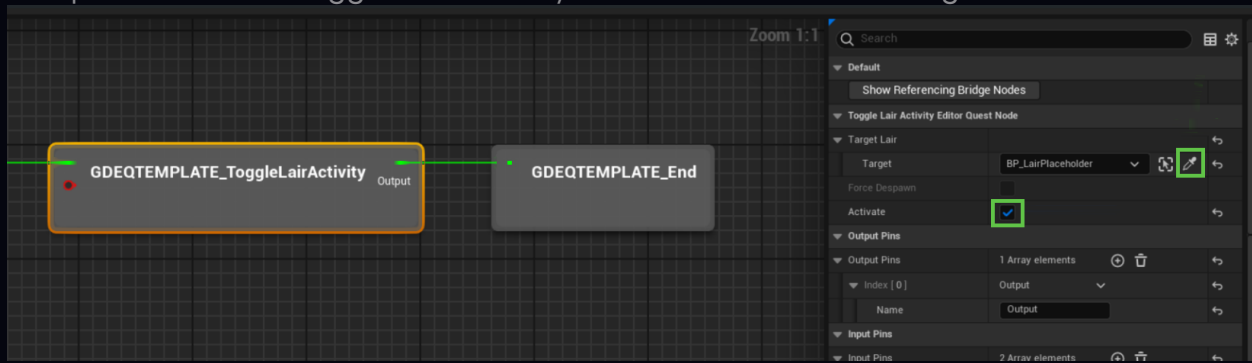
To set up and configure the lair, follow these steps:

1. Find the folder where the trigger placeholder is located:
Plugins/GameDataEditorContent. Also, you can find it by the name of the actor:
BP_LairPlaceholder.



2. Use the pipette tool to select the previously set lair actor. This way, the node will refer to a specific actor in the world.
3. Select **Activate**.

4. Connect the Toggle Lair Activity node to the End node for the script to successfully complete after the Toggle Lair Activity node has finished working.



Example of GDEQ implementation on WorldMap *GDEQ_Template_ActivateLair_BP*



Helpful nodes

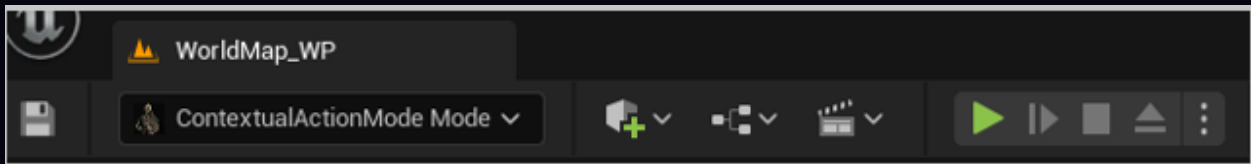
To direct a mutant/NPC to contextual action, connect the AI behavior node with a spline to the node that receives the signal during the spawn of the character. Follow these steps:

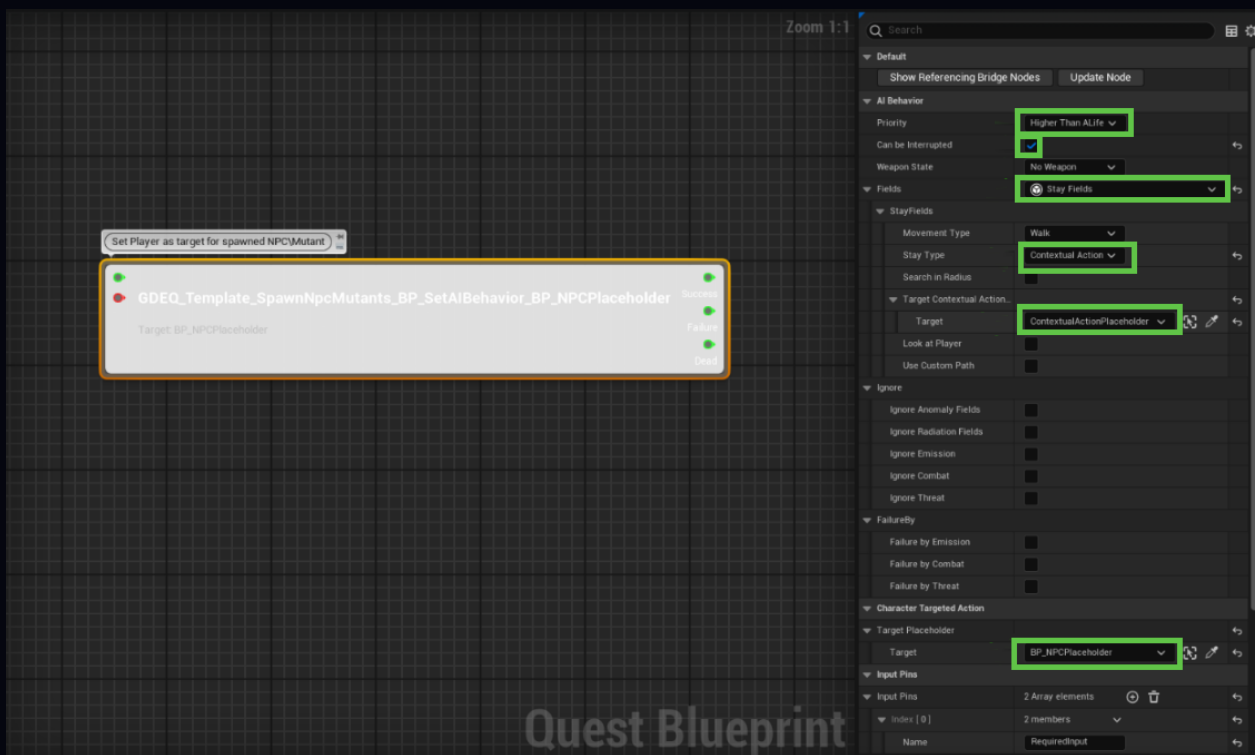
1. Check that all the marked fields correspond to the values shown in the screenshot.
2. In **Character Target Action** → **Target Placeholder** → **Target**, select which placeholder this node belongs to: *BP_SquadPlaceholder* or *BP_NPCPlaceholder*.

IMPORTANT

Place contextual actions at the GDEQ level:

1. Go to the GDEQ level editing mode.
2. Select **ContextualActionMode**.
3. Place the contextual action.





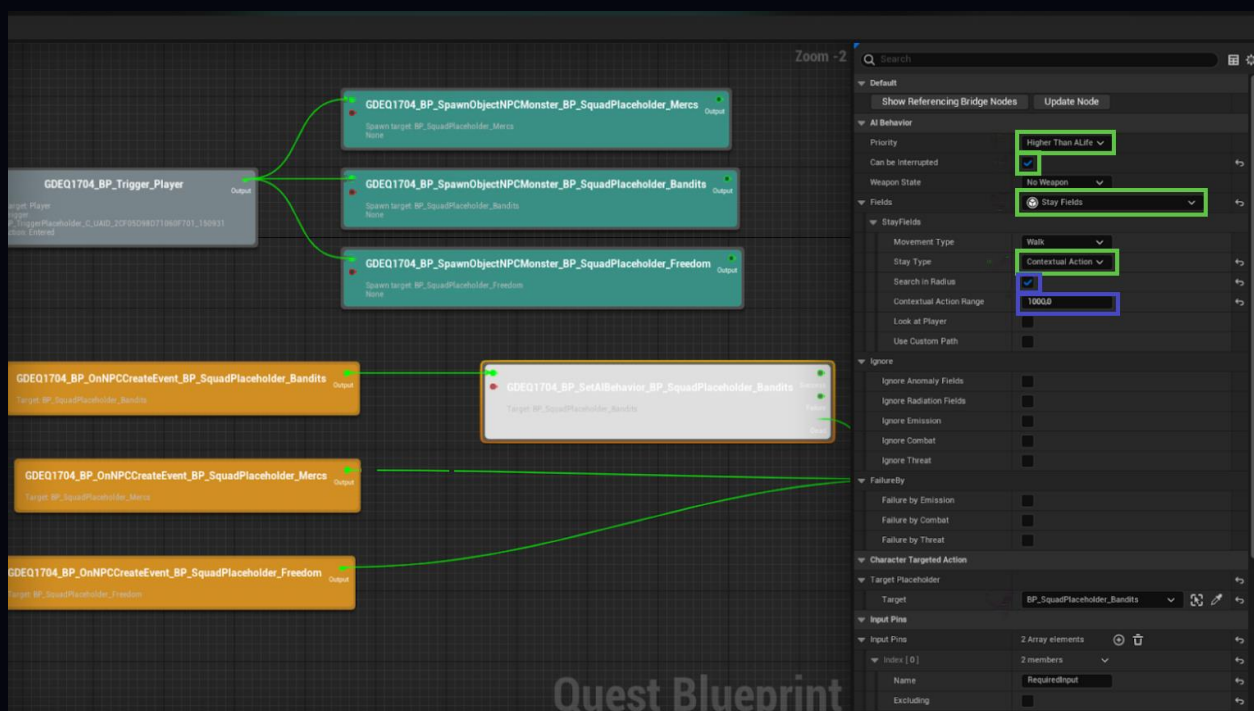
To set a contextual action within a radius or for a group of NPCs/mutants (BP_SquadPlaceholder), follow these steps:

1. Check that all the marked fields correspond to the values shown in the screenshot.
2. In **Character Target Action** → **Target Placeholder** → **Target**, select which placeholder this node belongs to: *BP_SquadPlaceholder* or *BP_NPCPlaceholder*.

IMPORTANT

Place contextual actions at the GDEQ level:

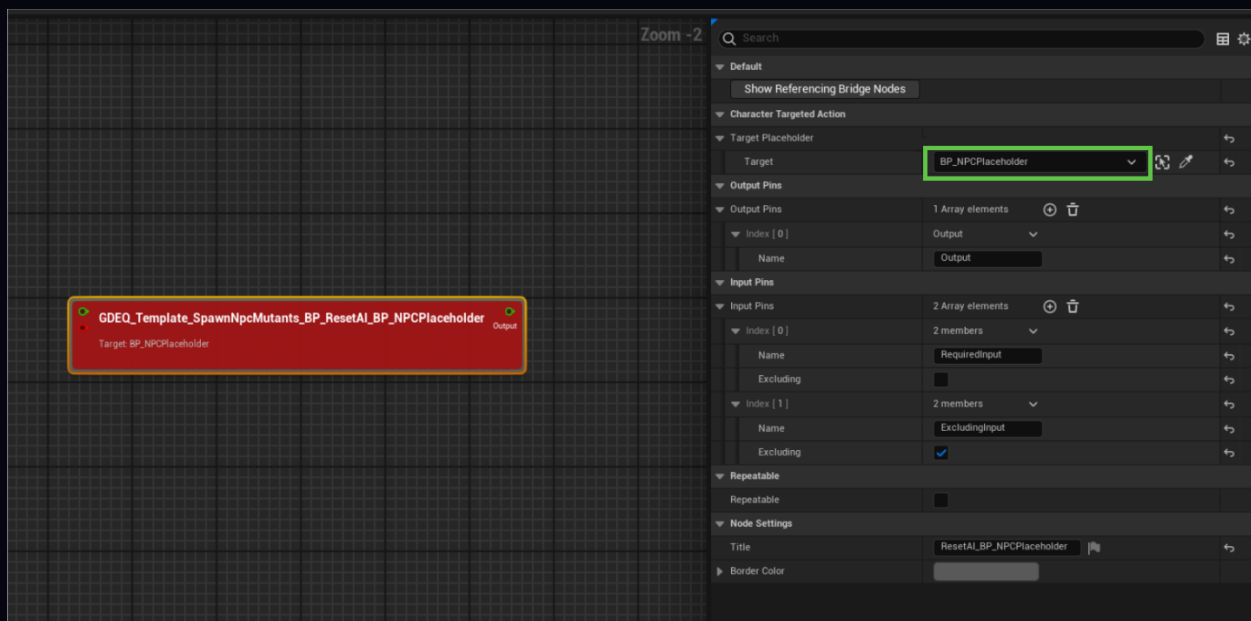
1. Go to the GDEQ level editing mode.
2. Select **ContextualActionMode**
3. Place the contextual action.



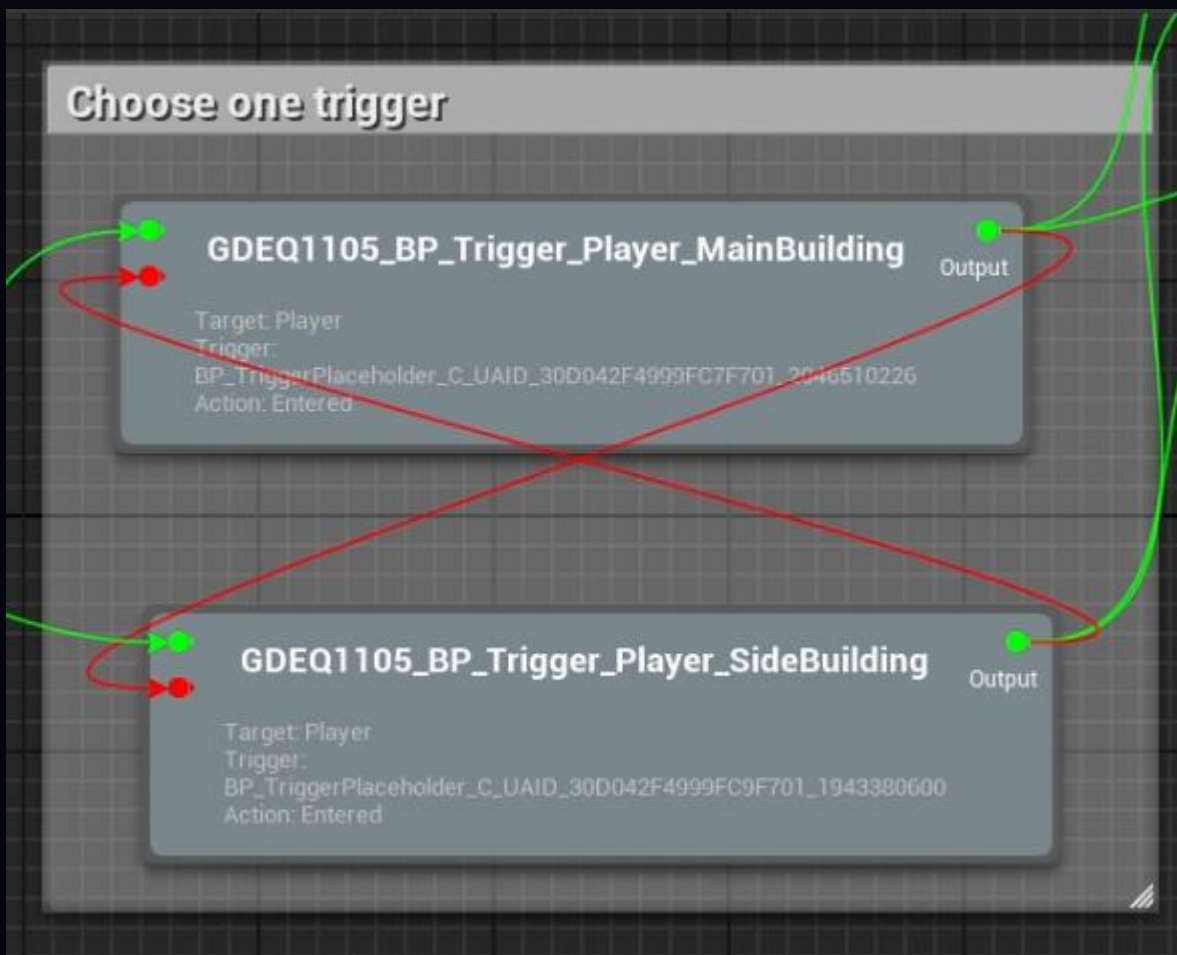
Reset AI node

To send a mutant/NPC into the world, use the Reset AI node. Follow these instructions:

- Connect the node at the moment in the script when you want the spawned character to follow the A-Life rules.
- **IMPORTANT**
All characters belonging to A-Life despawn when they enter offline mode.
- If the character remains quest-bound, i.e., spawned via *BP_SquadPlaceholder* or *BP_NPCPlaceholder*, they will teleport to the border of the visualizer and follow the player.
- Choose the option that is required for each specific GDEQ.
- This node allows despawned characters to perform contextual actions that are not set in the GDEQ level.



To use multiple triggers to activate GDEQ, configure the triggers as shown in the screenshot. This combination of triggers disables the others when one is used. As shown in the screenshot, logical OR works. Please pay attention to the red pins/lines/arrows.

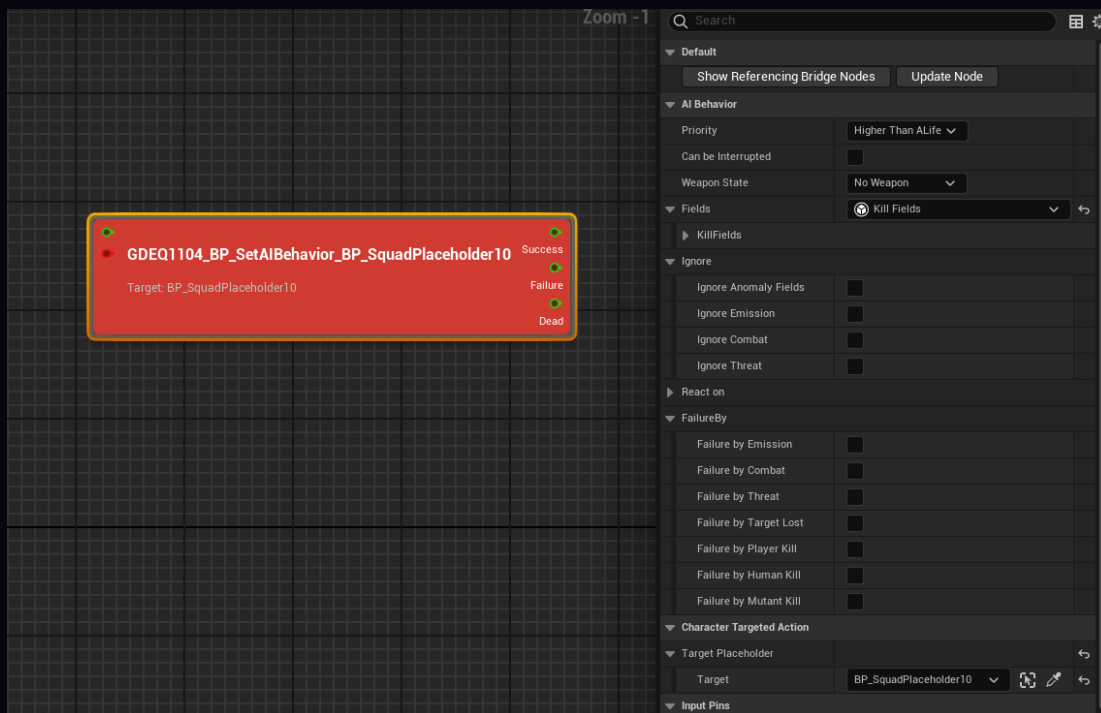


Kill node

The Kill node works even outside the visualizer. This means that if this node is activated outside the visualizer, the characters specified in it will fight offline, i.e., they will calculate the outcome, and the player will only see the consequences of the fight when they appear in the player's visualizer.

This means that you can use the node for two purposes:

1. To show the consequences of the battle, but make them random.
2. To use for target selection. In this case, the node must be activated no further than 7000 units from the player.



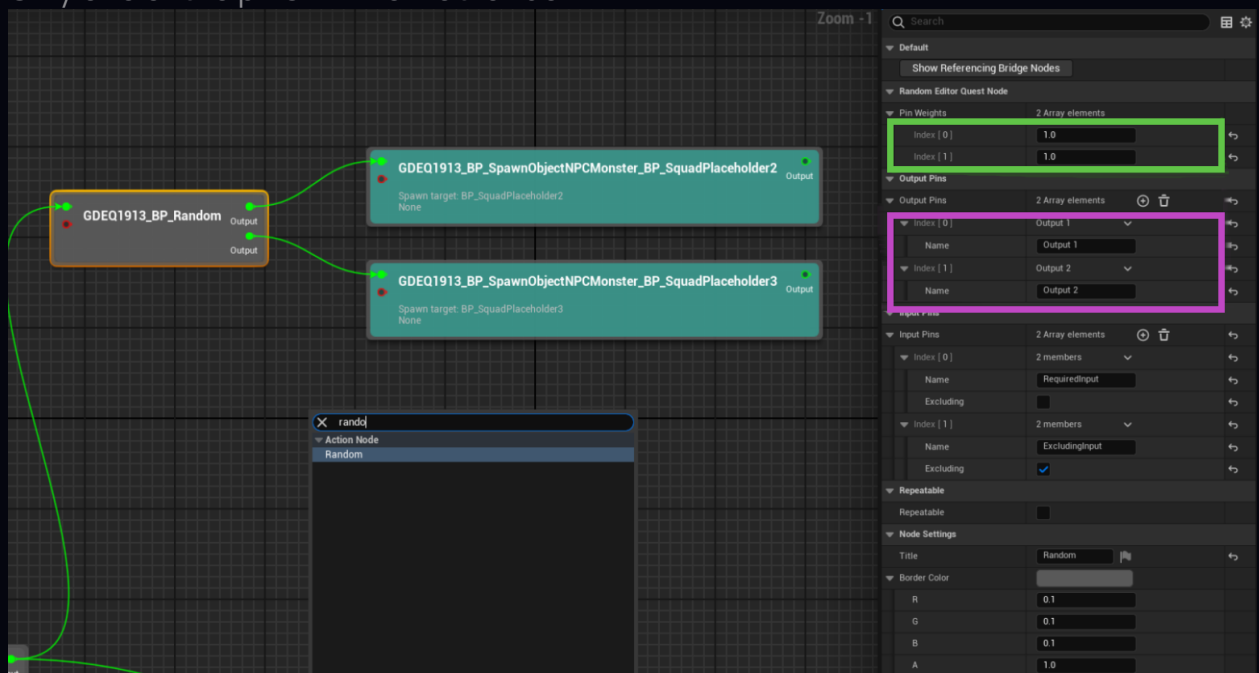
Random node

If you need a character to spawn in different locations, or if you need the same character to spawn in different locations, use the Random node. Considering that GDEQ uses generic NPCs, it will be a random spawn point for the player, not different spawns.

The Random node works as follows:

1. Add more than one pin in **Output Pins**. Name each uniquely without spaces.
2. Specify the weights for each pin in **Pin Weights**. A weight of 0 is not allowed.
3. Connect the pins to the spawners.

1. Only one of the pins will work at random.



Dialog tool

Description

A visual graph editor for dialog blueprints. Narrative designers use it to create the logic behind dialogs for quests and encounters.

A dialog blueprint is created in the engine based on previously created dialog topics. Only one dialog blueprint can be created for each dialog topic.

Working in a dialog blueprint is similar to working in a quest blueprint. A narrative designer has access to a set of nodes that are connected to each other by links. A node starts working when a signal enters it and stops after it performs its function.

Unlike quest nodes, dialog nodes cannot be excluded via an exclusion pin because there are no such pins in Dialog Tool.

There are the following types of dialog nodes:

- Start nodes – the essential part of the dialog blueprint with which every dialog starts. There is no input pin, only one output pin.
- Label nodes – the type that represents the main part of the dialog. They contain information about the text and the voiceover. They have one input pin and one output pin.
- Reference-label nodes – link-nodes to a label-node in another dialog box. They have one input pin and one output pin.
- Condition and IF nodes – they execute precisely the same functions as in the quest blueprint. However, there are more options on how they can be connected in the dialog blueprint. Each of these nodes has one input pin. The Condition node has one output pin, while the IF node has two.
- Wait For Reply (WFR) and TimedWFR nodes – these nodes are required to implement the line selection functionality in the dialog. If comparing WFR and TimedWFR, the latter has a timer responsible for how much time a player has to choose a line before the node finishes by sending a signal to the timeout pin. They have one input pin and an unlimited number of output pins.
- End notes – end a dialog with several possible scenarios. They have only one input pin.

The DBP settings include the following options:

- Dialog on the Go – the selected dialog in PIE will be played on the fly and without a dialog interface. When this checkbox is selected, previously created and new End nodes will have the Close UI setting, and WFR nodes will only be able to include two responses.
- Continue after interrupt – the dialog can be interrupted by an event – the start of combat, emission, another dialog, etc. If this setting is enabled, the dialog will pause when interrupted and continue when the interrupting event ends. If disabled, the dialog will simply end when interrupted.
- Interrupt by combat – interrupts the dialog if the speaker enters combat.

- Interrupt by emission – interrupts the dialog if an emission begins.
- Main Reply in Topic Selection – if enabled, the topic with this dialog will be highlighted in color in the NPC's topic selection menu.
- DLCName – indicates which DLC the dialog belongs to. Selecting **None** means the original game.

There are also editor tools:

- Comment block – a section used to create comments. To create one, hover your mouse over the desired spot in the column and press C.

Technical limitations

Please consider these upper limits for text volume and do not exceed them. You do not have to use all the characters available – the more concisely you can express your idea, the better:

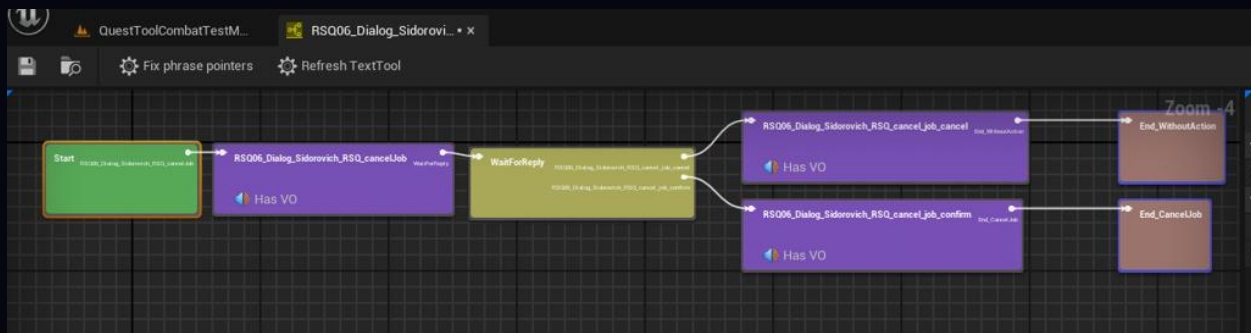
- Number of characters in a dialog line – *preferably* up to 140 characters, including spaces. Technical maximum – 200 characters.
- Number of characters in WFR – *preferably* up to 70 characters, including spaces, but not less than 6 characters. Technical maximum – 140 characters.
- Number of characters in the speaker's name – *preferably* up to 18 characters, including spaces. Technical maximum – 36 characters.
- Number of choices in one WFR (including Global WFR) – up to 4 including.
- Number of characters in quest description – from 300 to 400 characters, including spaces.
- Number of characters in stage description – up to 50 characters, including spaces.
- Number of characters in the inventory item description – up to 500 characters, including spaces.
- Number of characters in notes – up to 1200 characters, including spaces.
- Number of characters in audiologs – up to 750 characters, including spaces.

Pay attention to these approximate limits – you can exceed them, but only reasonably:

- Main Quest – 3000 words
- Side Quest – 1500 words
- Encounters – 750 words

References

Example of a dialog blueprint



DBP settings window

Search		
▼ Dialog Blueprint		
▼ Dialog Members	1 Array elements	⊕ ⊖ ↺
▶ Index [0]	sidorovic_0	↺
Topic SID	RSQ06_Dialog_Sidorovich_DeclineJob	↺
Global Wfr SID	sid_topic_RSQ06_Dialog_Sidorovich_DeclineJob	↺
Global Wfr	I can't do the job for you.	↺
Dialog on the Go	☐	
Continue After Interrupt	☒	
Main Reply in Topic Selection	☐	
Interrupt by Combat	☐	↺
Interrupt by Emission	☐	↺
▼ DLC		
DLCName	None	

Use case

Dialog Tool functionality allows scripting dialogs and comments.

Call Player feature

You need to configure Set Dialog in the node, which would instruct an NPC to which the dialog is attached (Dialog Target) to call the player when they approach a certain radius.

Add the following settings to the node:

- Call Player – enables the setting
- Call radius – specifies the radius in which an NPC starts calling the player. By default, it equals the player's hearing radius.

As soon as the player enters the radius, the NPC begins to play its Call Player comment (set as Hello Topic in the NPCPrototypes config) once every 60 seconds. Call Player comment cannot be played more than once every 60 seconds, even if the player enters and exits the radius.

Call Player disables the following options with a value of **False**:

- Repeatable
- Start Forced Dialog

- Is Comment
- Target Player
- Override

Call Player enables the option with a value of **True**:

- Ignore Global WFR

Later, also configure the player call animation or embed it in the Call Player comment.

Dialog Node: Label

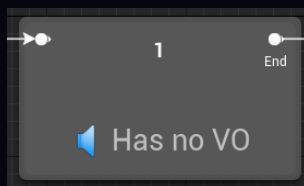
Description

This dialog node contains information about the text and voiceover of dialog lines.

The node starts when it receives a signal and ends after playing all the dialog phrases of the label. After the node is completed, it sends the signal further.

You can configure Dialog Actions in the node. Also, you can configure dialog animations, LookAt, and a voice modulator for voiceover in each phrase.

References



Dialog Node		
Set Sequence as Record Target		
Dialog Actions	0 Array elements	⊕ 🗑
Label SID	RSQ06_Dialog_Sidorovich_C10_Finish_aboutArtefact	↶
WFRIn Game Text SID	sid_label_RSQ06_Dialog_Sidorovich_C10_Finish_aboutArtefact	↶
WFRIn Game Text	About that artifact.	↶
Localized Sequence Holder		
Is Cutscene	☐	
Wait for Sequence to Finish	☐	
▶ Sequence Holders	2 Map elements	⊕ 🗑
Loop Sequence	☐	
Should Preblend	☐	
▼ Phrases	2 Array elements	

▼ Phrases	2 Array elements	
▼ Index [0]	RSQ06_Dialog_Sidorovich_C10_Finish_aboutArtefact_64138	↩
Phrase SID	RSQ06_Dialog_Sidorovich_C10_Finish_aboutArtefact_64138	↩
Phrase Text	About the artifact you requested...	↩
Dialog Member Name	Skif	↩
Dialog Member SID	PC	↩
Unskippable	☐	↩
▶ Sequence Audio Section Holders	3 Map elements ⊕ 🗑	↩
▶ Localized Sound Texts	3 Map elements	
▼ Dialog Members Animations		↩
▼ Dialog Members Animations	2 Array elements	
▼ Index [0]	sidorovic_0	↩
Dialog Member Name	sidorovic_0	↩
Emotional State	Angry ▼	↩
Look At	PC ▼	↩
▼ Dialog Animations	1 Array elements ⊕ 🗑	↩
▼ Index [0]	3 members ▼	↩
Dialog Animation Categ...	All ▼	↩
Dialog Animation Type	NPCApprovalTalkingUsual ▼	↩
▼ Gesture Timing	2 Map elements ⊕ 🗑	↩
Ukrainian ▼	0.0 ▼	↩
English ▼	0.0 ▼	↩
▼ Index [1]	PC	↩
Dialog Member Name	PC	↩
Look At	Don't change ▼	↩
Dialog Animations	0 Array elements ⊕ 🗑	↩
Voice Modulator	Gasmask ▼	↩
▼ Index [1]	RSQ06_Dialog_Sidorovich_C10_Finish_aboutArtefact_64139	↩

How to use

The node is used in all dialogs.

Parameter	Description
-----------	-------------

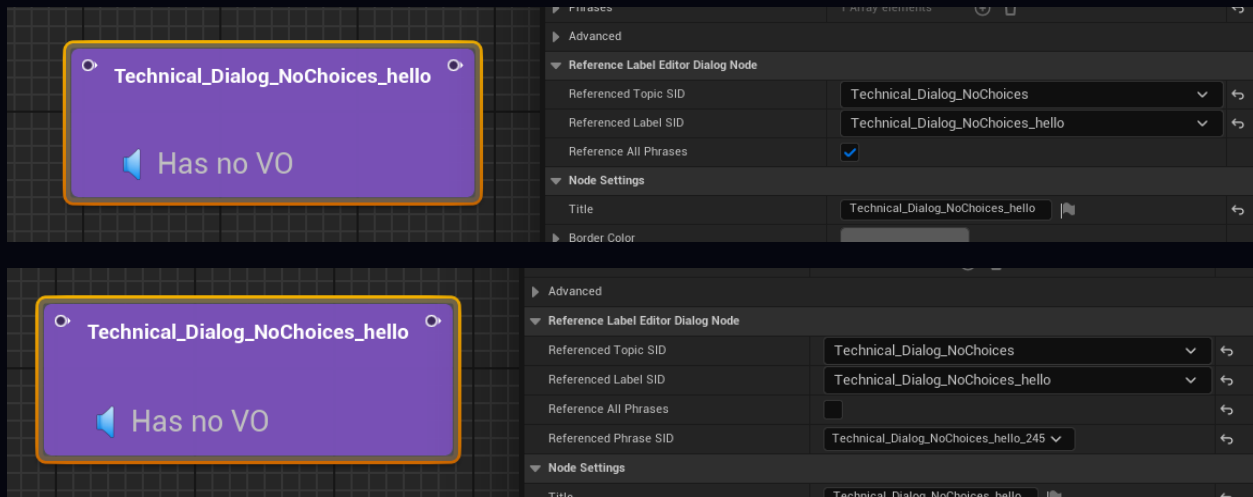
Dialog Actions	<p>Dialog actions, which are used if the label can be selected in the Wait For Reply window.</p> <p>A dialog action allows the use of a certain feature in the dialog.</p> <p>Please do not use more than one dialog action per label.</p>
----------------	--

Dialog Tool Node: Reference Label

Description

This dialog node is functionally similar to Label, but it allows inserting labels or phrases from another dialog topic into the dialog blueprint.

Node view



IMPORTANT

Please do not use a node in the same container from which you need to reference a label.

How to use

Parameter	Description
-----------	-------------

Referenced Topic SID	The name of the dialog topic from which the label will be referenced
Referenced Label SID	SID of the label that will be reproduced by the node
Reference All phrases	A checkbox that activates the reference of all phrases from the selected label
Referenced Phrase SID	If Reference All phrases is cleared, it allows referencing a specific phrase from the label

Dialog Node: Wait for Reply

This node allows the player to select a response option in a dialog.

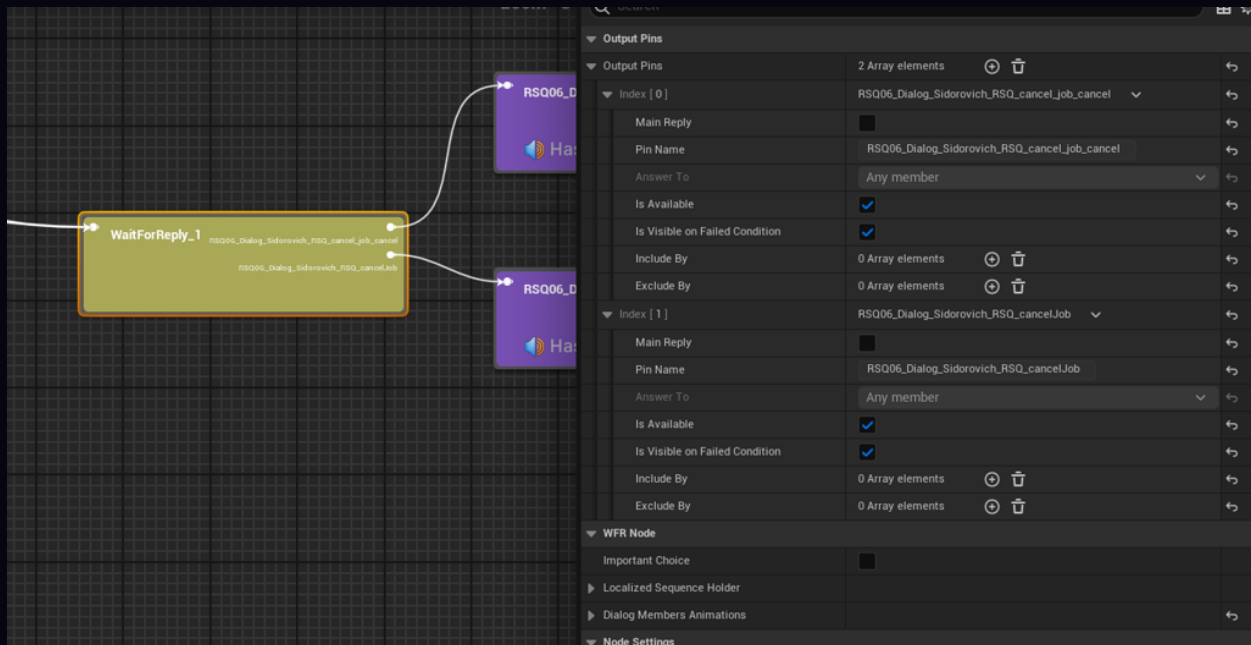
Node settings:

- Important choice – enables special display of the choice in the dialog UI. The **Important Choice Branch** option will be added to the output pin settings, and the response options will be marked as the corresponding plot branch selection in the UI.
- Localized Sequence Holder – allows customizing the dialog in a cutscene.
- Dialog Members Animations – allows customizing animations and looking at speakers while waiting for a reply.

Pin settings:

- Main Reply – highlights the answer with color
- Answer To – allows grouping answers by who they are intended for. Cannot be configured in a dialog with one member.
- Is Available, Include By, Exclude By – settings for the answer display system
 - Is Available – determines whether the reply is in the list of replies from the beginning of the dialog
 - Include By – determines which replies in this WFR must be selected (all of them, not just any) for this reply to appear in the list (if it was not there)
 - Exclude By – determines which answers in this WFR must be selected (all of them, not just any) for this answer to disappear from the list (if it was there)
- Is Visible on Failed Condition – if enabled, answers that are unavailable due to a Condition node will be displayed as unavailable for selection. If disabled, such answers will not be displayed at all. Responses blocked by Exclude By or disabled Is Available are never displayed, regardless of this setting.

- Important Choice Branch – available only if the Important Choice node setting is enabled. Allows choosing which plot branch will be taken when this response is selected.



Dialog Tool Node: Timed Wait for Reply

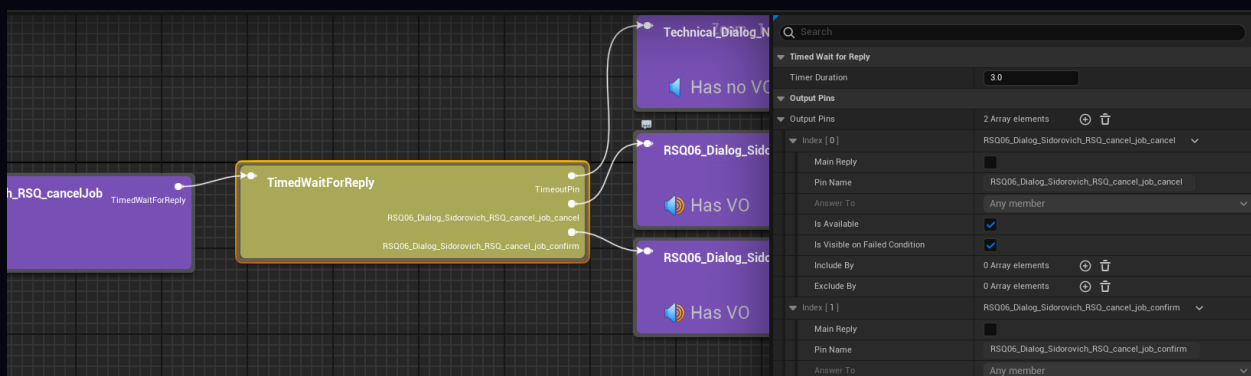
Description

This dialog node is functionally similar to WFR, but in the Timed version, it has a timer, after which the node will send a signal to the timeout output. The timer determines how much time the player has to select a response before the node ends.

IMPORTANT

Please do not set the timer to 0. Also, do not add conditions to the timeout pin (however, you can attach an If node).

References



Use case

The node is used in situations where you need to put the player in a more stressful situation, for example, where the life of a quest character depends on their choice.

How to use

1. Create a dialog blueprint.

2. Open it.
3. Create the Timed WFR node.
4. Set Timer Duration.
5. Create the required number of outputs in the node settings
6. Configure each output:
 - a. Connect the label.
 - b. Configure the main reply.
 - c. Configure the availability of the choice.
7. Save the changes.

Dialog Node: Condition / If

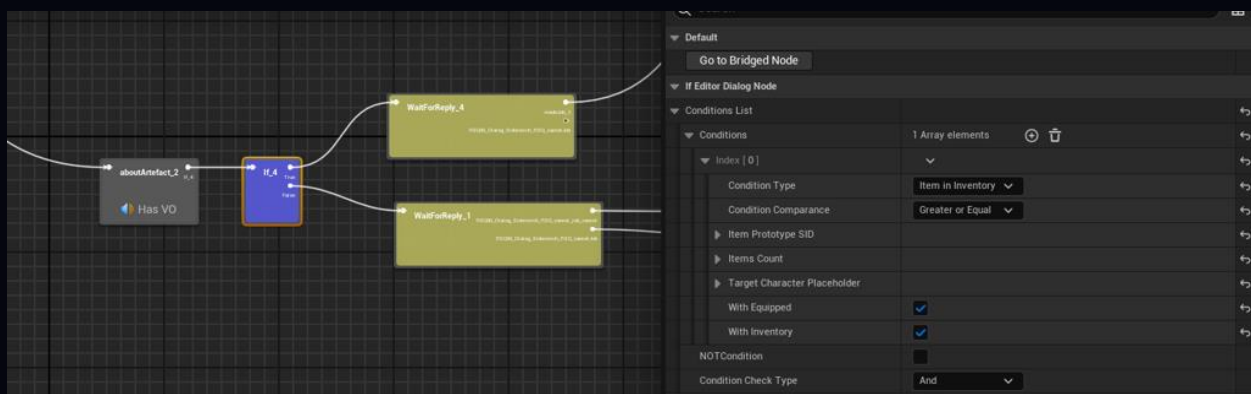
These dialog nodes allow checking conditions and play different parts of the dialog depending on them.

Settings:

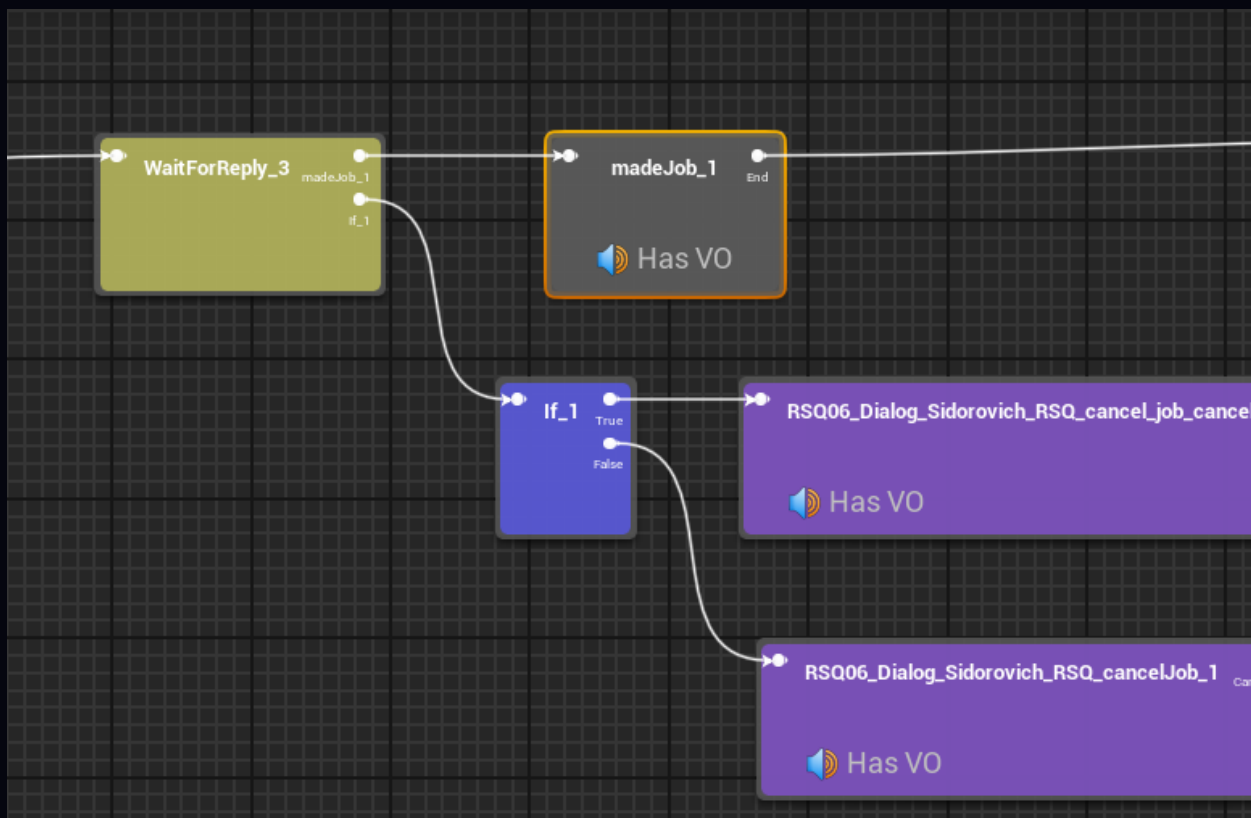
- Condition Type Check – a way to check an array of conditions:
 - And = activation when all conditions are met
 - Or = activation when one of the conditions is met
- Conditions – the check used in a specific node
- NOT Condition – a setting that causes the node to trigger when the check is not performed, and vice versa, not to trigger when all conditions are met

Both Condition and If nodes perform their checks at the moment of triggering. That is, if during the dialog the state of the world changes so that the check changes the value, the player will be able to listen to different dialog options.

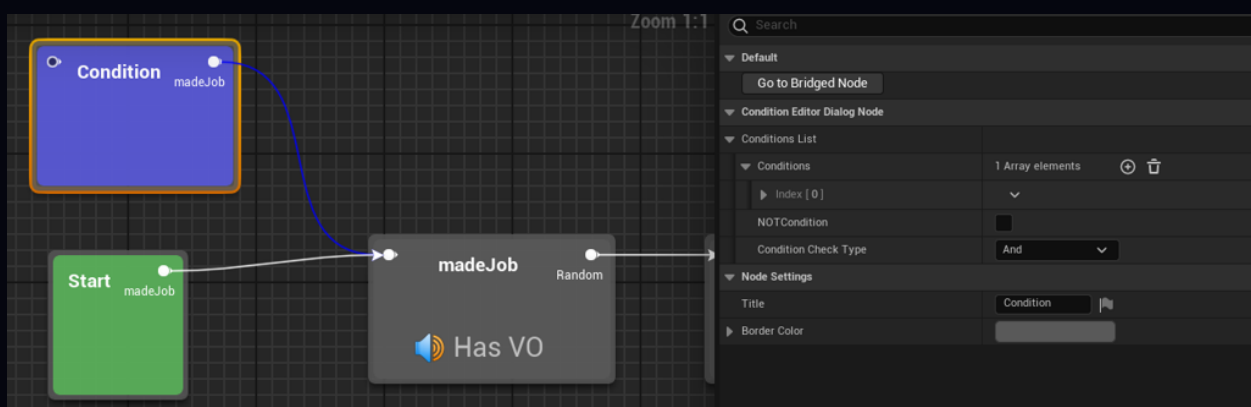
The If node has one input and two output pins. It launches the **True** pin branch if the check passes, or **False** if the check fails. Thus, it is used to run the dialog along one of two branches.



The If node can also be connected after the WFR node. Then, one of the two responses connected to the If node will be added to the WFR node, depending on the condition check.



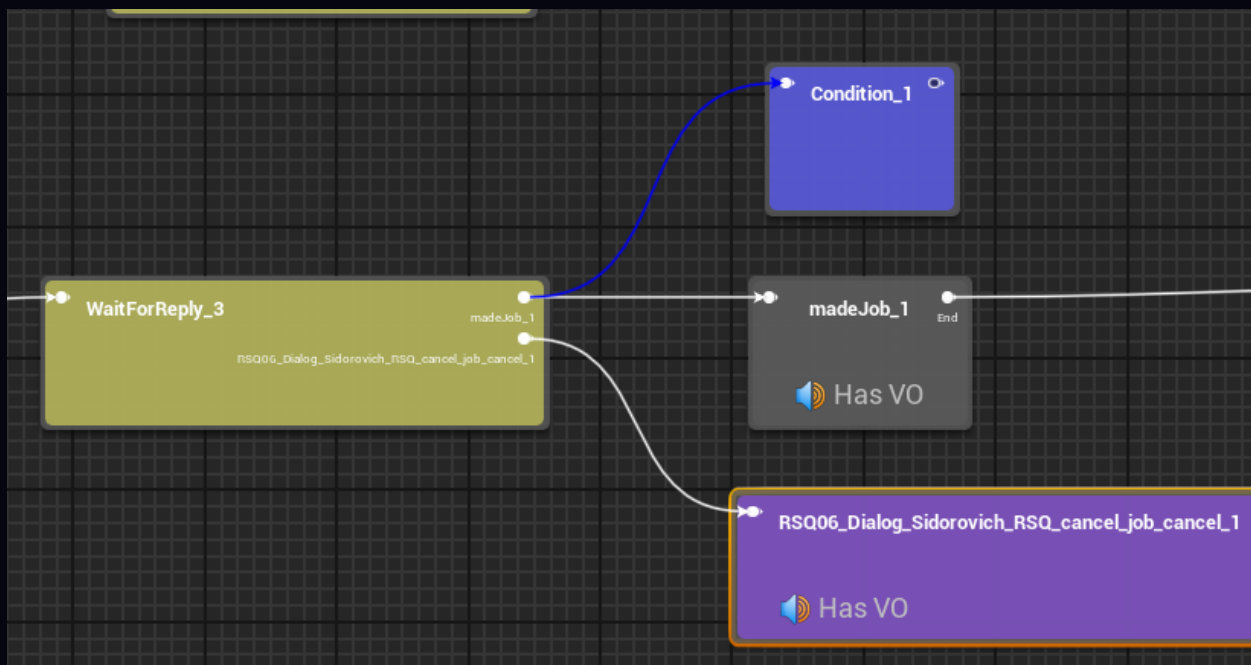
You can connect Condition with the outgoing pin to the Label (or Reference Label) node, leaving the incoming pin unconnected. If the condition is met, this label will be played as usual, but if the condition is not met, all phrases in this label will be skipped, and the next node will start immediately.



You can also connect the Condition node with its input pin to one of the WFR node's responses, leaving the output pin unconnected. Then this response will be available for selection only if the condition check passes. If not, the response will either be hidden or displayed as blocked, depending on the **Is Visible on Failed Condition** setting in the WFR node pin.

IMPORTANT

Please do not connect condition nodes to the Timeout pin of the Timed WFR node.



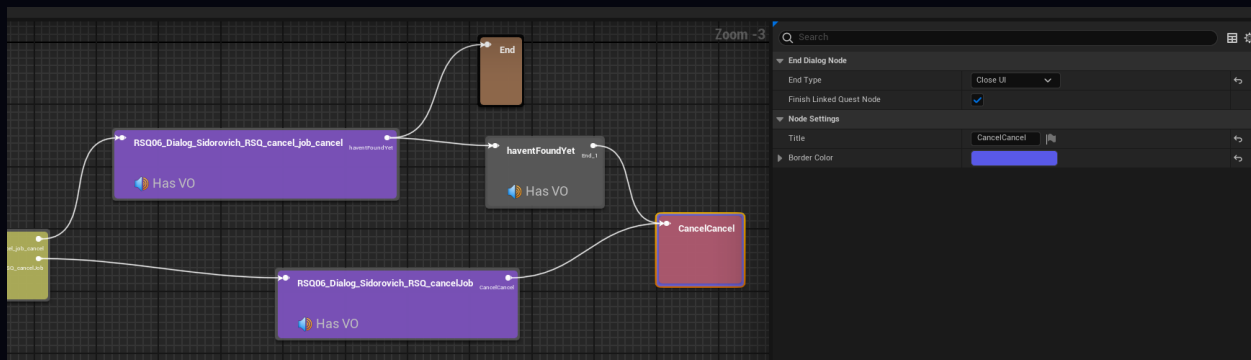
Dialog Node: End

The End node in the dialog indicates the end of the topic and/or the moment when the signal from the dialog is sent to the quest system, activating the output pin in the corresponding SetDialog node. There can be several of them in the dialog, each with its own settings.

Since dialog End nodes are linked to the Set Dialog quest node, when you modify them in the dialog, select **Update Node** in the settings panel and then resave the quest blueprint for all Set Dialog nodes that use this dialog.

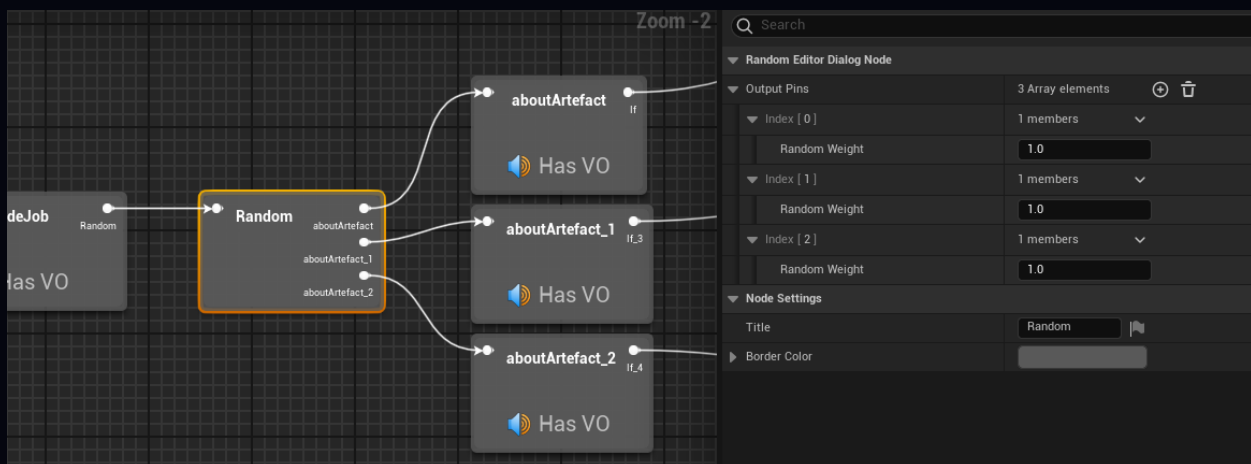
Settings:

- End type
 - Do nothing – used for “gateways, used to create a pin in the linked Set Dialog quest node that is activated when this point is reached in the dialog. Only this setting allows connecting the end node in parallel with the following nodes.
 - Close UI – when this end node is reached, the dialog ends, and the dialog UI or subtitles disappear.
 - Return to topic selection – when this end node is reached, the dialog exits to the topic selection menu of the character with whom the interaction happened (or who is specified as **NPC To Start Dialog** in the corresponding Set Dialog quest node).
- Finish Linked Quest Node – used to determine whether to finish the corresponding Set Dialog quest node when the end is reached. If this setting is not enabled, the Set Dialog node will send a signal to the corresponding pin but will remain active. Changing this setting requires resaving the quest blueprint with the Set Dialog node where this dialog is selected.



Dialog Node: Random

This node allows selecting one of the random replies. You can set the probability of selection using **Random Weight** in the pin settings of this node. This is simply the ratio of probabilities of selecting a particular label. Their sum does not necessarily have to be equal to one, but it is better to keep them within reasonable limits (you are unlikely to need values such as 0.01 or 999999). The reply choice may change when the dialog is replayed, so the player has a chance to hear all the options.



Dialog Tool Node: Dialog Actions

Description

Dialog Actions is a set of actions performed when playing labels. They can be either graphical (icon in the UI) or logical (Action).

They can look like this:

- Changing the color of replies (main reply checkbox)
- Item icons from the inventory
- Special icons

Dialog Actions work both from Global WFR and through Ignore global WFR. Also, Dialog Actions are displayed in WFR and GWFR even after the If node (or several nodes of this kind) before the label where these actions were set.

Only one dialog action can be in a label, except in some cases provided below.

List of Dialog Actions and their features

Label Dialog Actions:

- Open Trade Menu – open the trade menu, UI + Action.
- Open Upgrade Menu – open the repair and upgrade menu, UI + Action.
- Heal Player – restore the player's health and reset all negative effects, UI + Action.
 - If the player has no negative effects and their HP = 100%, the dialog phrase with this action is not available.
 - You can set a price for this service in the action settings.
- Give Money – take the selected amount of money from the player, UI + Action.
 - If the player does not have this amount, the response will be blocked. This money is added to the character with whom the dialog was started.
- Get Money – give the player the selected amount of money, UI + Action.
- Show Money – show a certain amount of money, UI.
- Attack – the icon showing the start of combat, UI.
- Hit – the icon showing a hit, UI.
- Give Medkit – give the character a first aid kit and take them out of the curled-up pose on the ground, UI + Action.
 - The cheapest first aid kit from the inventory is given, which should be displayed in the icon.
 - If there is no first aid kit in the player's inventory, the line is unavailable.
- Show Item – show the character an item, the line is unavailable if the selected item is missing, UI.
 - You can find an **In Equip** checkbox in the **Show item** settings, which allows showing equipped items.
- Rank – check the player's rank, you can set one of four ranks.
- Faction Relationships – checks the player's relationship with the selected faction.
- Give Item – Take the selected item from the player, does not work if the selected item is missing, UI + Action.
 - Up to three items can be included in the action. If there is more than one item, then instead of their names, **Give item** is in brackets, and the items are indicated as icons.
 - You can find an **Equip** checkbox in the **Give item** settings, which allows transferring equipped items.
 - The taken item is added to the inventory of the character with whom the dialog was started.

- Get Item – give the player the selected item, UI + Action.
 - Up to three items can be included in the action. If there is more than one item, their names are not displayed, only icons.
- Threat action – the icon showing a potentially dangerous action, UI.
- Eat – feed the player selected food not from the player's inventory, UI + Action.
 - You can set a price for this service in the action settings.
- Drink – give the player selected alcohol not from the player's inventory, UI + Action.
 - You can set a price for this service in the action settings.
- Travel – there are three different types:
 - Travel (UI) – move to another location (UI). Add it to a dialog label, if the action icon should be in the topic selection menu, or the first label after WFR, if the travel icon should be in this WFR.
 - The action dialog has a built-in check for the emission. During the emission, the topic is unavailable.
 - Travel to point (Action) – move to another location (action). It should be added to the label, after which fast travel will occur. This action automatically closes the dialog interface.
 - Action parameter – fast travel location prototype SID, where the teleport will be performed.
 - Travel point selection (Action + UI) – an action that informs that after the end of this topic, a selection of locations for fast travel by the guider will open.
 - The topic to which this action is added must end with an End node of type end = do nothing. You can only use it in the conductor's travel topic. It does not need to be supplemented with the Travel (UI) action, as it is already built into it. Like Travel (UI), add it to the first label.
 - Action parameter – PlayerOverload. None = the conductor will take you even with overload, Full = with full overload (>60kg), you will not be taken. Partial = you will not be taken with a full or partial (>50kg) overload. In addition to checking for overload, this topic is unavailable during the emission.
- Dialog Exit Reply – the icon indicating that the line leads to an exit from the dialog, UI.
- Reward – the icon indicating that, after this line, the player will receive a reward. It is needed for cases where rewards are issued through a config with different sets of rewards.
- Get Side Quest – the icon indicating that the line leads to the start of a Side Quest. It is used on a label with the Main Reply setting, leading to a dialog branch, where the player can get a side quest.

- Faction Relationship Check – the icon indicating that a certain reputation with the faction is required to select this line. This action adds a Faction Relationship condition with the corresponding parameters to the phrase.
- Laugh – Action, not displayed in UI, makes the specified dialog members laugh. It is needed for implementing jokes.

Other Dialog Actions:

- Main Reply – a quest-significant line that advances the plot and changes the color of the line in UI.
- Already seen Reply – a line that has already been heard, set automatically, changes the color of the line in the UI.

It is also possible to configure Dialog Action for Global WFR. To do this, select the first dialog node in the asset and create the necessary Dialog Actions in it.

Use case

The nodes are used everywhere.

How to use

Label Actions

1. Open the dialog blueprint.
2. Select **Label**, connected to WFR with an input pin.
3. Add **Dialog Action**.
4. Select the required action and configure it.
5. Save the blueprint.

Main reply

1. Open the dialog blueprint.
2. Select **Label**, connected by an input pin to WFR.
3. In the **Label** settings, select the **is Main Reply** checkbox.
4. Save the blueprint.

Global WFR, hello topic, bye topic, info topic, defeat state topic, and comment

Global WFR

Global WFR is a menu for selecting topics available for conversation with NPCs. It appears when the player approaches an NPC and wants to talk to them by pressing a key.

- Global WFR contains all topics that are available for conversation at a specific moment in time with a specific NPC. These can be topics added by a quest or info topics for conversation, such as 'What's new in the Zone?'
- Topics in Global WFR are added as they are connected to a specific NPC.
- The Set Dialog node should have the **Skip Global WFR** setting, which allows skipping Global WFR and immediately launching the dialog attached to it.

Hello topic

The Hello topic is a topic that launches automatically when a player interacts with an NPC before seeing Global WFR. Usually, these are phrases like “Hello, nice to see you!”

- If the **Skip Global WFR** setting is enabled in the Set Dialog node, the Hello topic will also be skipped.
- Hello topic is set by default in the config, where the NPC prototype is initially created as a link to the dialog asset in the engine.
- The Set Dialog node has the **Override Hello topic** option, which replaces the default Hello topic of a specific NPC with the topic attached to it. This will continue until this node is unloaded via Exclude (it does not end automatically) or another Set Dialog node with the **Override Hello topic** parameter is activated. The last activated node is the one that will always be working.

Bye topic

Bye topic is a topic that starts automatically when the player selects the **Exit** button in the dialog. This interrupts any ongoing dialog (assuming that it is possible to exit it at any time).

- The Bye topic is set by default in the config, where the NPC prototype is initially created as a link to the dialog asset in the engine.
- The Set Dialog node has the **Override Bye topic** option, which replaces the default Bye topic of a specific NPC with the topic attached to it. This will continue until this node is unloaded via Exclude (it does not end automatically) or another Set Dialog node with the **Override Bye topic** parameter is activated (the last activated node is the one that will always be working).

Busy topic (WIP)

The busy topic is a topic that is launched instead of a regular dialog in an interaction if the dialog blocks an event in A-Life (emission, attack, etc.).

- The busy topic is set by default in the config, where the NPC prototype is initially created as a link to the dialog asset in the engine.
- The Set Dialog node has the **Override Busy topic** option, which replaces the default Busy topic of a specific NPC with the topic attached to it. This will continue until this node ends automatically (at the end of the A-Life event blocking the dialog) or another Set Dialog node with the **Override Busy topic** parameter is activated (the last activated node is the one that will always be working).

Override Hello topic, Override Bye topic, Override Info topic, and Override Defeat State topic cannot be active simultaneously in one Set Dialog node.

Info topic

The Info topic is a random information topic, such as 'What's new in the Zone?', which some NPCs have. In NPCPrototype, there is a checkbox that allows the character to have an info topic, as well as a field where you can select the initial info topic. If the initial topic is not specified, but can be according to the settings, you can select it from the list of suitable topics when the NPC is spawned.

- The Info topic is selected randomly from the entire array of suitable comments in the Info topic prototypes. After selecting it, the player will hear one of these comments.
- The Set Dialog node has the **Override Info topic** option, which will replace the random Info topic of a specific NPC with the topic attached to it. This will continue until this node is unloaded via Exclude (it does not end automatically) or another Set Dialog node with the **Override Info topic** parameter is activated (the last activated node is the one that will always be working).

Defeat topic

The Defeat topic is a topic that is connected to an NPC and played when interacting with an NPC that is in the Defeat state (curled up on the ground).

- The Defeat topic is set by default in the config, where the NPC prototype is initially created as a link to the dialog assets in the engine.
- The Set Dialog node has the **Override Defeat topic** option that will replace the random Defeat topic of a specific NPC with the topic attached to it. This will continue until this node is unloaded via Exclude (it does not end automatically) or another Set Dialog node with the **Override Defeat State topic** parameter is activated (the last activated node is the one that will always be working).

Dialog sound+anim content implementation pipeline

AudioKinetic files

The recorded voiceover for event reaction lines is packaged so that each line corresponds to one AKEvent, which will contain a voiceover for all voices and all localizations.

Each event contains a mandatory Event Name with a suffix that relates it to one of the voiceover categories.

Voiceover category	Event name
--------------------	------------

Contextual comments from the main character	General_CharacterSID_Player_***_VariantName
NPC comments about the player and interaction with them	General_CharacterSID_NPC_Interact_***
Comments reacting to the emission	General_CharacterSID_NPC_Emission_***
NPC combat comments	General_CharacterSID_NPC_Combat_***
NPC comments about damage, statuses, and effects	General_CharacterSID_NPC_HitReaction_***_
NPC comments in quest contextual actions	General_CharacterSID_NPC_QContextual_***
NPC comments in contextual actions	General_CharacterSID_NPC_Contextual_***
Comments on peaceful actions	General_CharacterSID_NPC_Peacefull_***
Comments reacting to events in the world	General_CharacterSID_NPC_Reaction_***
Radio and stationary radios	General_CharacterSID_NPC_Radio_***
Loudspeakers	General_CharacterSID_NPC_Speakers_***

In the engine, AKEvents must be located in this folder:
_STALKER2/Audio/WwiseAudio/Events/Voiceover/General/{EventType}.

The event must be named VO_{Phrase SID}_{Phrase_ID}.

- PhraseID – a unique number (as in quests), the phrase identifier
- PhraseSID – (according to the narrative convention) will look like this:
General_CharacterSID_EventName
Example: General_bandit_0_ThrowGrenade

Phrase SID and Phrase_ID will be fully specified in the data exported from TextTool.
Example: VO_General_bandit_0_ThrowGrenade_18345

Sound file names are formed using the same formula with the voice identifier prefix after the voiceover asset identifier: [AssetTypePrefix]_[PhraseSID]_[PhraseID].

Example: VO_VoiceB_ThrowGrenade_Bandit_18345.wav,
VO_VoiceD_WeatherChange_Duty_37162.wav

Parameters (switches) for this event:

- The voice is located in this folder:
_STALKER2/Audio/WwiseAudio/Switches/Default_Work_Unit/Voice/. Assets are called Voice-A, Voice-B, etc.

Quest dialog events are placed here:

- _STALKER2/Audio/WwiseAudio/Events/Voiceover/MainQuests/QuestTitle/CharacterSID/ – main quests
- _STALKER2/Audio/WwiseAudio/Events/Voiceover/SideQuests/QuestTitle/CharacterSID/ – side quests
- _STALKER2/Audio/WwiseAudio/Events/Voiceover/Encounters/QuestTitle/CharacterSID/ – encounters
- _STALKER2/Audio/WwiseAudio/Events/Voiceover/Hubs/HubHame/CharacterSID/
- _STALKER2/Audio/WwiseAudio/Events/Voiceover/ANC/ANCTitle/CharacterSID/ – AdditionalNarrativeContent voiceovers
- _STALKER2/Audio/WwiseAudio/Events/Voiceover/Bossfights/BossfightTitle/CharacterSID/ – bossfight voicovers

Example:

_STALKER2/Audio/WwiseAudio/Events/Voiceover/SideQuests/SQ01/valenok/SQ01_valenok_LineSID/

There are no parameters for quest AKEvents.

Face animation

Recorded facial animation files in the engine should be formed as follows:

[AssetTypePrefix]_[HeadPrefix]_[LanguageID]_[VoiceID/QuestID]_[PhraseSID]_[FactionID/none]_[PhraseID]

Generic phrases	Quest (main, side, encounter) phrases
Prefix type: AS_FAC_	
Head identifier: will mainly be in the [age]_[variant number]_ format, for example, 30_04_ (4th head of a 30-year-old)	
Localization prefix: EN_ or RU_	
Voice identifier, for example, VoiceB_	Character identifier, for example, Warlok_
	Quest ID, for example, SQ01_
Event ID, for example, ThrowGrenade_ or WeatherChange_	Dialog ID, also known as the container name in the dialog tree, for example, QuestStartWarlokDebt
Faction ID, for example, Army_ или Bandits_	
Phrase ID in numbers, for example, 1234	
Full example	
AS_FAC_30_04_RU_VoiceB_ThrowGrenade_Army_1234	AS_FAC_EN_Warlok_SQ01_QuestStartWarlokDebt_1234

Animations should be located in _STALKER2\Animations\CharacterHead\.

Internal folder structure:

- General
 - ThrowGrenade (a folder for each game event) – here you can find animations for the “I’m throwing a grenade” line
- MainQuests – here you can find Richter’s animations, which are used in the E01_MQ01 quest
 - E01_MQ01 (quest)
 - Richter (character)
- SideQuests
- RepeatableSideQuests
- EncounterQuests
- Hubs
- ANC
- Bossfights

Comments in-game

It is necessary to implement a system that will trigger appropriate comments from NPCs near the player when certain events occur. SpeechManager would be suitable for this.

Intro to configs

This brief guide is created to provide an understanding of what configs are, how they work, and how to use them.

All configs are located here: *s2gamebuild\Content\GameLite\GameData*

Each config contains several prototypes, which are selected in the engine in various nodes, fields, etc.

Configs are sensitive to spaces and alignment, so *be sure* to run the play mode before committing your work. The engine will show an error if there is incorrect formatting and suggest that you fix it.

Types of configs

There are different types of configs, each of which is stored in a separate file. Narrative designers mainly need the following:

- QuestItemPrototypes
- NPCPrototypes
- QuestObjPrototypes
- GlobalVariablePrototypes
- NotePrototypes

Also, there are other configs that are the result of the work of systems within editors.
Example:

- DialogPrototypes, DialogChainPrototypes – created by the dialog editor when saving a dialog blueprint
- QuestNodePrototypes – created by the quest editor when saving a quest blueprint
- JournalPrototypes – created by the journal editor when saving a journal blueprint
- SpawActorPrototypes – created by any placeholder when saving, storing the necessary information for creating NPCs/anomalies/squads/etc.

Detailed information about configs

QuestItemPrototypes

It is used to create item configs required for narrative designers. This config extends game design configs, such as ArmorPrototypes, ArtifactPrototypes, ConsumablePrototypes, DetectorPrototypes, and WeaponPrototypes, in which items are created.

All fields, except SID, can be absent. If a field is missing, it will take the default value from the first prototype in the file.

Path to QuestItemPrototypes:

s2gamebuild\Content\GameLite\GameData\ItemPrototypes\QuestItemPrototypes.cfg

Template example:

```
1 QuestItemTemplate : struct.begin {refurl=../ItemPrototypes.cfg;refkey=[0]}
2     SID = QuestItemTemplate
3     IsQuestItemPrototype = true
4     ShouldTriggerAnomalies = false
5     EffectPrototypeSIDs : struct.begin
6     | [0] = empty{bskipref}
7     struct.end
8 struct.end
```

Below you can see the example descriptions of prototype lines:

Line	Description
------	-------------

QuestItemTemplate	The name of the prototype structure. It must be unique. Please leave it the same as the SID.
: struct.begin	The beginning of a structure, each struct.begin must be accompanied by struct.end, otherwise, the program will not know where the beginning and end are.
refurl=../ItemPrototypes.cfg	Configs use an inheritance system. This config inherits the item from the prototype in ItemPrototypes.
refkey=[0]	It inherits from the structure named [0]. If you open ItemPrototypes, this will be the very first prototype, which starts with [0]: struct.begin.
SID = QuestItemTemplate	The item's SID must be unique. This is what we select in the engine node or something else.
IsQuestItemPrototype	<p>Serves only to search for localization. It should not be changed or added anywhere. If IsQuestItemPrototype = true, the engine will search in the QuestItemPrototypes category in TextTool, otherwise, in Items.</p>  <p>The screenshot shows the GSC TextTool interface. At the top, there are tabs: Dialogues, Journals, Characters, UI Objects (selected), and Misc. Below the tabs, there is a search bar and a row of buttons: MQ, SQ, EQ, RSQ, and E_SQ. A dropdown menu is open from the search bar, showing a list of categories: Items (highlighted with a green box), Locations, Notes, QuestItemPrototypes (highlighted with a green box), and Audiologs.</p>
Icon	Responsible for storing the icon that will be displayed on the item in the inventory
IconBig	Responsible for storing the icon that will be displayed in the item description

MeshPrototypeSID	Responsible for which mesh will be displayed for the item. All available meshes are located in the <i>MeshPrototypes.cfg</i> file. The SID of the desired mesh prototype is inserted into the field.
ItemGridWidth	How much space the item will take up in width in the inventory cells
ItemGridHeight	How much space the item will take up in height in the inventory cells
IsQuestItem = true/false	Responsible for whether the item can or cannot be thrown away, eaten, sold, etc.
Weight	Weight of the item in kg. Please note that weight affects the balance in the game.
Cost	The price of the item. Please note that cost affects the balance in the game.

Quest artifacts, armor, weapons, consumables, or detectors must be inherited from the corresponding template within QuestItemPrototypes or directly from a similar item in the recommended config – e.g., *EQ08_FreshBread : struct.begin {refurl=ConsumablePrototypes.cfg;refkey=Bread}*). In the case of documents, they can be inherited from the zero prototype (*refkey=[0]*).

Examples:

- A PDA that cannot be thrown away takes data from the base PDA, where the icons are already configured. To add meshes to *a//PDAs*, specify the configured MeshPrototypeSID only for BasePDA, and then the meshes will change for all PDAs in the game. This is done specifically to simplify and speed up working with configs.
- Base PDA, where icons for all PDAs in the game are defined.

```
BasePDA : struct.begin {refkey=[0]}
    SID = BasePDA
    Icon = PaperSprite'/Game/GameLite/FPS_Game/UI/ItemIcons/PDA_icon.PDA_icon'
    IconBig = PaperSprite'/Game/GameLite/FPS_Game/UI/ItemIcons/PDA_icon.PDA_icon'
struct.end
```

- Quest AK, inherited from the existing regular AK from WeaponPrototypes in the game, with a changed SID and IsQuestItemPrototypes to find it in the desired QuestItemPrototypes tab in TextTool.

```
SQ86_GenijAK74 : struct.begin {refurl=WeaponPrototypes.cfg;refkey=GunAK74_ST}
SID = SQ86_GenijAK74
IsQuestItemPrototype = true
IsQuestItem = true
// Wwise Events and parameters
FireEventOneShot = AkAudioEvent'/Game/_STALKER2/Audio/WwiseAudio/Events/Weapons/AR/AK74/AK74_Fire/SFX_AK74_OneShot.SFX_AK74_OneShot'
FireEventLoop = AkAudioEvent'/Game/_STALKER2/Audio/WwiseAudio/Events/Weapons/AR/AK74/AK74_Fire/SFX_AK74_Loop.SFX_AK74_Loop'
FireEventBreak = AkAudioEvent'/Game/_STALKER2/Audio/WwiseAudio/Events/Weapons/AR/AK74/AK74_Fire/SFX_AK74_Break.SFX_AK74_Break'
struct.end
```

NPCPrototypes

It contains information about available NPCs!

Path to NPCPrototypes: *s2gamebuild\Content\GameLite\GameData\NPCPrototypes.cfg*

Line	Description
SID	The same unique field, inserted into QuestObjPrototypes
NameTextKey	Field with a key in TextTool, which consists of a constant sid_character_ a and the SID of the NPC from TextTool

QuestObjPrototypes

It contains information about quest objects.

Path to QuestObjPrototypes:

Is2gamebuild\Is2gamebuild\Content\GameLite\DLCLGameData\DLCL1\ObjPrototypes.cfg

Line	Description
SID	The unique name
Faction	NPC faction
NPCPrototype SID	The field that pulls localized names and phrases from TextTool. Insert the SID from NPCPrototypes into it.

GlobalVariablePrototypes

Contains information about global variables.

Path to GlobalVariablePrototypes:

s2gamebuild\Content\GameLite\GameData\GlobalVariablePrototypes

Line	Description
SID	The unique name
Type	The type of a global variable. For now, two types exist: <ul style="list-style-type: none">• EGlobalVariableType::Int – counter• EGlobalVariableType::Bool – True/False
DefaultValue	The default value for the variable, which it will have by default

NotePrototypes

Contains only the SID from TextTool and note selection in the engine. It will pull the rest of the data itself.

Path to NPCPrototypes: *s2gamebuild\Content\GameLite\GameData\NotePrototypes.cfg*

DialogPoolPrototypes

It is for phrases that NPCs can say at runtime. These phrases cannot be changed dynamically through nodes. It is used for dynamic comments such as “Reloading,” “Stalker, back off, I’m not up for you right now,” etc. So far, two systems use this prototype: busy comments, which are played immediately when interacting with NPCs (if there is a reason for it), and phrases that NPCs can say during combat.

Path to DialogPoolPrototypes:

s2gamebuild\Content\GameLite\GameData\DialogPoolPrototypes.cfg

IMPORTANT

Busy comments do not yet use Priority, SoundAppearanceTimeoutSec, so you can press the NPC interaction button many times per second and get the same number of comments.

Line	Description
------	-------------

SID	The unique name
DialogEventType	The event type, when the phrase should be spoken. See all event types via ConfigEditor.
DialogMemberRestrictions	Structure that determines who should say this phrase
FactionRestrictions	Restriction by factions
Faction	Faction SID
ObjPrototypeRestrictions	Restrictions on specific NPCs
NPCPrototypeSID	The NPC prototype SID. You can find it by copying the SID and searching through <i>NPCPrototypes.cfg</i> .
AvailableDialogs	A structure that determines which dialogs can be played
DialogChain	The DialogChain ID – it is usually the same as the name of the dialog blueprint
SoundAppearanceTimeoutSec	Delay between phrases of the same type so that NPCs do not spam the phrase “Reloading” every 3 seconds
Priority	The priority that can be set so that the phrase about emission is given higher priority than the phrase “Stalker, get lost, I’m thirsty”

Config examples

Config for the phrase, when an NPC takes shelter in combat.

```
[12] : struct.begin {refkey=Default}
      SID = Combat_Action_Cover
      DialogEventType = EDialogEventType::Combat_Action_Cover
      AvailableDialogs : struct.begin
        [0] = QA_Voiceover_bandits_combat_test_takecover
      struct.end
    struct.end
```

Config for a busy comment, when the player wants to talk to an NPC during emission, and one of them is not in a shelter.

```
Busy_OutOfShelter : struct.begin {refkey=Default}
  SID = Busy_OutOfShelter
  DialogEventType = EDialogEventType::Busy_OutOfShelter
  AvailableDialogs : struct.begin
    [0] = General_Comment_Busy_OutOfShelter
  struct.end
struct.end
```

Creation

Item creation

IMPORTANT

Please read the detailed information about [QuestItemPrototypes](#).

If you have a request to create an item with a SID in TextTool, e.g., SQ06_DocumentsFromKPP, open the QuestItemPrototypes file and insert a new structure at the end. You may copy the one below and change its fields:

```
(item sid) : struct.begin {refkey=[0]}
  SID = (item sid)
  Icon = (path to asset)
  IconBig = (path to asset)
  MeshPrototypeSID = (sid from MeshPrototypes)
struct.end
```

The result is the following:

```
SQ06_DocumentsFromKPP : struct.begin {refkey=[0]}
  SID = SQ06_DocumentsFromKPP
  Icon =
  PaperSprite'/Game/GameLite/FPS_Game/UI/ItemIcons/DocumentsItem2.DocumentsItem2'
  IconBig =
  PaperSprite'/Game/GameLite/FPS_Game/UI/ItemIcons/DocumentsItem2.DocumentsItem2'
  MeshPrototypeSID = QuestDocuments
struct.end
```

In this particular case, there was a request to create an item in the form of documents, so two additional fields with icons and mesh were filled in.

To create a PDA, a template was created, specifying all the general parameters for all PDAs.

```

BasePDA : struct.begin {refkey=[0]}
SID = BasePDA
Icon = PaperSprite'/Game/GameLite/FPS_Game/UI/ItemIcons/PDA_icon.PDA_icon'
IconBig = PaperSprite'/Game/GameLite/FPS_Game/UI/ItemIcons/PDA_icon.PDA_icon'
struct.end

```

All other PDAs are inherited from it – it cannot be thrown away or sold, and it has the ChobitkoPDA sid. Example:

```

ChobitkoPDA : struct.begin {refkey=BasePDA}
SID = ChobitkoPDA
Type = EltemType::QuestItem
struct.end

```

Above, there was an example with the minimum necessary set of data, which is sufficient in most cases. If you have a special request, please take a look at more parameters in the very first prototype in *ItemPrototypes.cfg*.

NPC creation

NPCs can have unique lines and dialogs – thus, you need to create NPCs in TextTool. If you have a request to create an NPC with a SID in TextTool, e.g., stalker_1, follow these steps:

1. Open the *NPCPrototypes.cfg* file and insert a new structure at the end to create an NPC with unique lines. You may copy the one below and change its fields:

```

(sid): struct.begin {refkey=[0]}
SID = (sid)
NameTextKey = sid_character_(sid)
struct.end

```

The result is the following:

```

stalker_1 : struct.begin {refkey=[0]}
SID = stalker_1
NameTextKey = sid_character_stalker_1
struct.end

```

2. Open the file with QuestObjPrototypes and insert a new structure at the end to create QuestObjPrototype (the one that is inserted in NPCPlaceholder).

You can find available factions in existing NPCs or in *RelationPrototypes.cfg*.

Let's leave an NPC SID for selection in the placeholder, the same as the selected NPCPrototype, but without the index at the end.

```

(NPC SID for selection in the placeholder) : struct.begin {refkey=QuestNPCBase}
SID = (NPC SID for selection in the placeholder)
Faction = (faction)
NPCPrototypeSID = (SID of the newly created NPC prototype from step 1)
struct.end

```

The result is the following:

```

    stalker_1 : struct.begin {refkey=QuestNPCBase}
SID = stalker_1
Faction = Neutrals
NPCPrototypeSID = stalker_1
struct.end

```

Above, there was an example with the minimum necessary set of data, which is sufficient in most cases. If you have a special request, please take a look at more parameters in the very first prototype in *QuestObjPrototypes.cfg* and *ObjPrototypes.cfg*.

Note creation

If you have a request to create a note with a SID in TextTool, e.g., EQ38_Note, follow these steps:

1. Open the NotePrototypes file and insert a new structure at the end. You may copy the one below and change its fields:

```

(note sid) : struct.begin {refkey=[0]}
    SID = (note sid)
struct.end

```

The result is the following:

```

EQ38_Note: struct.begin {refkey=[0]}
SID = EQ38_Note
struct.end

```

Global variable creation

If you have a request to create an item with a sid, e.g., SQ03_Stash_MastifReward, open the file with GlobalVariableProtptypes and insert a new structure at the end. You may copy the one below and change its fields:

```

(variable sid) : struct.begin {refkey=[0]}
SID = (variable sid)
Type = (variable type)
DefaultValue = (default value)
struct.end

```

The result is the following:

```

SQ03_Stash_MastifReward : struct.begin {refkey=[0]}
SID = SQ03_Stash_MastifReward
Type = EGlobalVariableType::Bool
DefaultValue = False
struct.end

```

Busy comment creation

Busy comments are played when a player tries to interact with an NPC in a certain state:

- EDialogEventType::Busy_OutOfShelter – during emission, if outside of shelter
- EDialogEventType::Busy_Combat – during combat

- `EDialogEventType::Busy_Threat` – in a state of searching for a threat
- `EDialogEventType::Busy_ContextualAction` – in a contextual action
- `EDialogEventType::Busy_Bleeding` – player has 4th degree bleeding
- `EDialogEventType::Busy_Radiation` – player has extremely high radiation
- `EDialogEventType::Busy_NoTopics` – an NPC has no topics to be played
- `EDialogEventType::Busy_NoTopicsSquadMember` – an NPC is a squad member, is not the leader, does not have their own dialog

There are dialogs to be played at a specific event, e.g., “the player is bleeding”, called `Richter_Comment_Bleeding1`, `Richter_Comment_Bleeding2`.

```
Busy_Bleeding_Richter : struct.begin {refkey=Default}
SID = Busy_Bleeding_Richter
DialogEventType = EDialogEventType::Busy_Bleeding
AvailableDialogs : struct.begin
  [0] = Richter_Comment_Bleeding1

  [1] = Richter_Comment_Bleeding2
struct.end

  DialogMemberRestrictions : struct.begin
  [0] : struct.begin {bskipref}
    ObjPrototypeRestrictions : struct.begin
      NPCPrototypeSID = Rihter
    struct.end
  struct.end
struct.end
```

Separated configs

To prevent conflicts and save time when switching to the narrative engine, some configs are separated by quests. A quest config is a child prototype.

For example, quest rewards are not added to `ItemGeneratorPrototypes` or `QuestRewardsPrototypes`, but to config files for each quest, which are located at `..|GameData|ItemGeneratorPrototypes|QuestRewardsPrototypes` and named according to the `Quest_Reward.cfg` mask.

The main thing is to correctly specify the inheritance in the refurl of the first structure. Subsequent structures can inherit from the zero value of the current config, or also inherit from the original prototype.

Rewards

Quest reward configs are located here:

`Z:\git\ls2gamebuild\Content\GameLite\GameData\ItemGeneratorPrototypes\QuestRewardsPrototypes` and inherit from `ItemGeneratorPrototypes.cfg`.

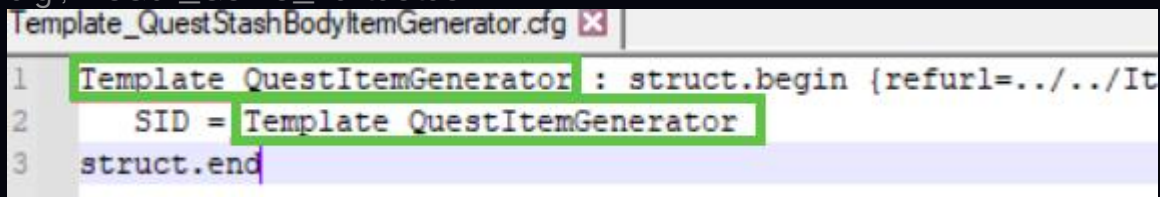
Item generator for stashes/corpses

Item generator configs for quest stashes and corpses are located here:

Z:\git\ls2gamebuild\Content\GameLite\GameData\ItemGeneratorPrototypes\QuestStashBody\ItemGeneratorPrototypes and are inherited from *ItemGeneratorPrototypes.cfg*.

To create a config for your quest, follow these steps:

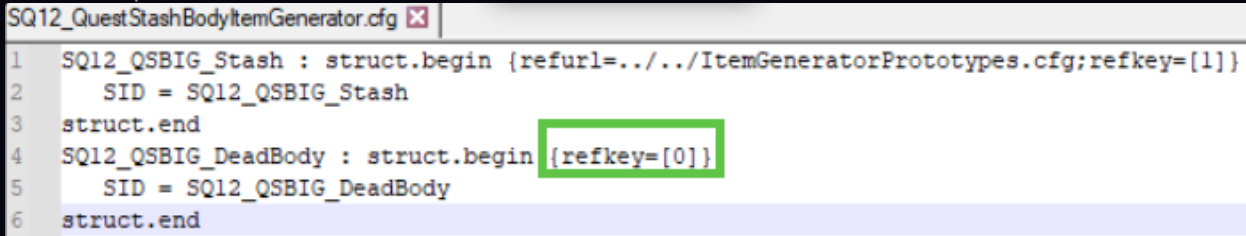
1. Go to the path above.
2. Find the *Template_QuestStashBodyItemGenerator.cfg* there.
3. Create a copy of it.
4. Rename it using the *Quest_QuestStashBodyItemGenerator.cfg* mask, e.g., *SQ01_QuestStashBodyItemGenerator.cfg*.
5. In this file, rename the structure and SID using the *Quest_QSBIG_OptionalInfo* mask, e.g., 1: *SQ01_QSBIG_VartaStash*.



```
Template_QuestStashBodyItemGenerator.cfg
1 Template_QuestItemGenerator : struct.begin {refurl=../../It
2   SID = Template_QuestItemGenerator
3 struct.end
```

6. If you need to add another structure, duplicate the first structure, set the correct name and SID for it, and specify inheritance from the first one. If you need more structures in this config, they will all inherit from the first one.

Example:



```
SQ12_QuestStashBodyItemGenerator.cfg
1 SQ12_QSBIG_Stash : struct.begin {refurl=../../ItemGeneratorPrototypes.cfg;refkey=[1]}
2   SID = SQ12_QSBIG_Stash
3 struct.end
4 SQ12_QSBIG_DeadBody : struct.begin {refkey=[0]}
5   SID = SQ12_QSBIG_DeadBody
6 struct.end
```

No further adjustments to the config are needed. This generator should be sent to the game designers for balancing. Until the balance arrives, this generator will add random food.

Item generator for quest NPCs

Item generator configs for quest NPCs are located at

Z:\git\ls2gamebuild\Content\GameLite\GameData\ItemGeneratorPrototypes\QuestNPCGeneratorPrototypes and inherit from *ItemGeneratorPrototypes.cfg*.

To create a config for your quest, follow these steps:

1. Go to the path above.
2. Find the *Template_QuestNPCGenerator.cfg* there.
3. Create a copy of it.
4. Rename it using the *Quest_QuestNPCGenerator.cfg* mask, e.g., *SQ01_QuestNPCGenerator.cfg*.

5. In this file, rename the structure and SID using the Quest_QuestNPCGenerator_OptionalInfo mask, e.g., E03_MQ03_QuestNPCGenerator_Varan.
6. If you need to add another structure, duplicate the first structure, set the correct name and SID for it, and specify inheritance from the first one. If you need more structures in this config, they will all inherit from the first one.

Example:

```

1  SQ18_QuestNPCGenerator_Barik : struct.begin {refurl=../../ItemGeneratorPrototypes.cfg;refkey=[8]}
2  SID = SQ18_QuestNPCGenerator_Barik
3  struct.end
4  SQ18_QuestNPCGenerator_Lukic : struct.begin {refkey=[0]}
5  SID = SQ18_QuestNPCGenerator_Lukic
6  struct.end
7  SQ18_QuestNPCGenerator_Polak : struct.begin {refkey=[0]}
8  SID = SQ18_QuestNPCGenerator_Polak
9  struct.end

```

No further changes to the config are required. This generator should be sent to the game designers for balancing. Until the balance arrives, this generator will add random basic equipment.

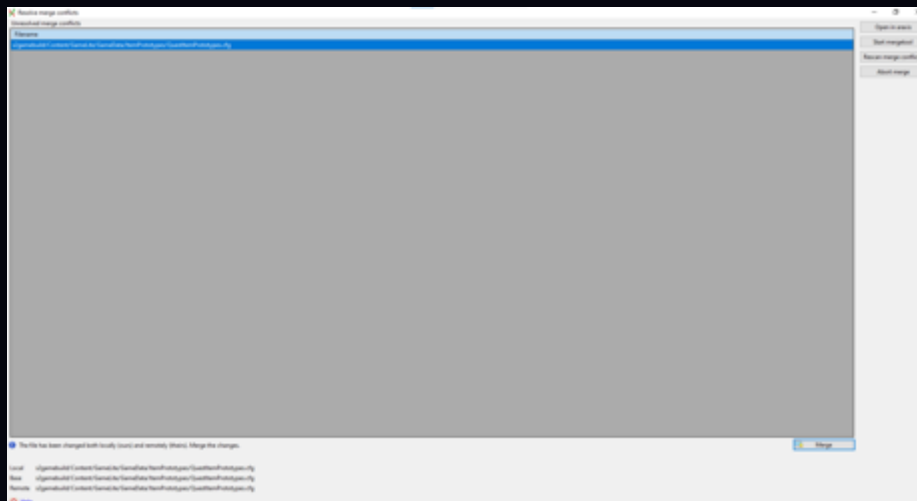
Conflicts merge

This is how you can resolve merge conflicts:

1. Open GitExtensions.

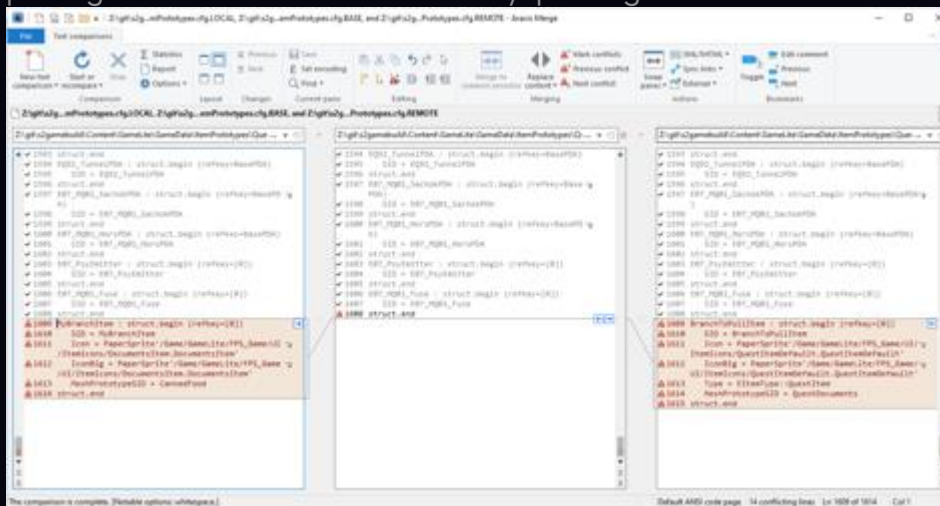


2. Select **Resolve...**

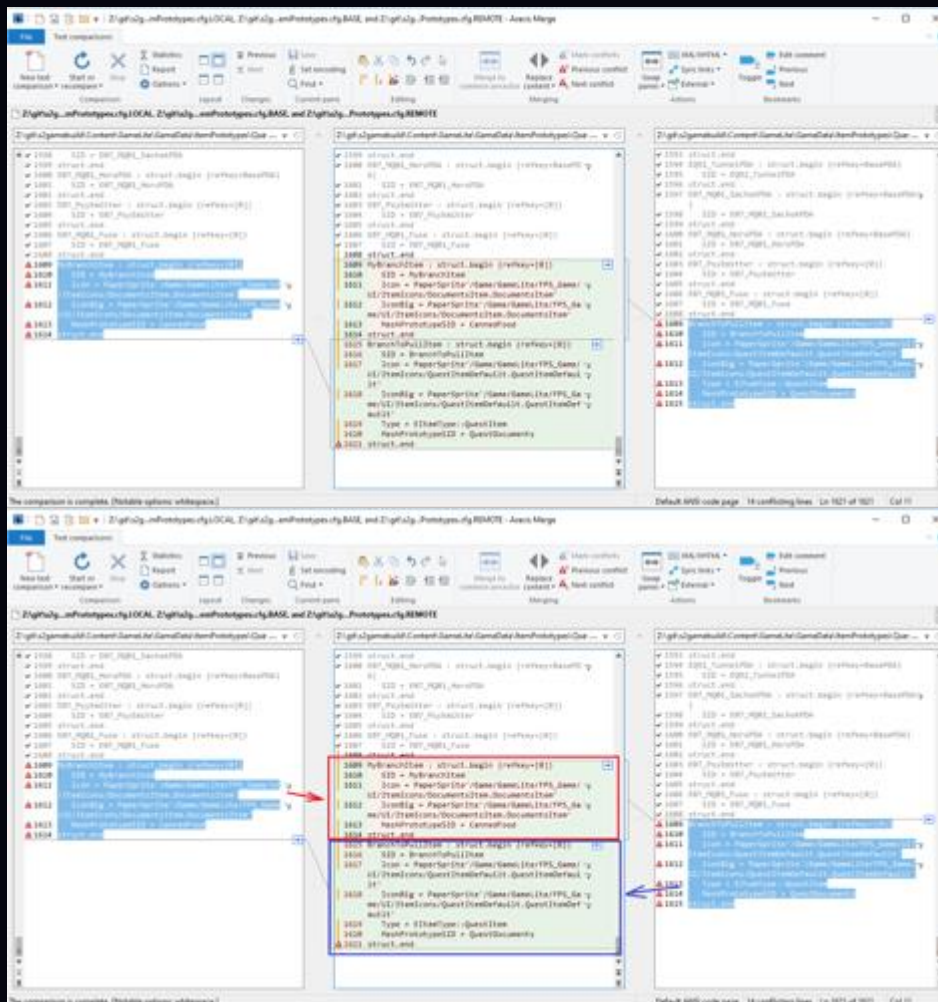


3. Open the conflicting file – in this case, it is *Quest/ItemPrototypes.cfg*.
 - On the left – changes in our branch.
 - In the center – the common version of both files.

- On the right – changes in the remote branch. This could be the branch we are pulling or the commit we are cherry-picking.



4. In most cases, it is enough to simply merge what is on our branch and on the remote branch.



IF the config contains information about the number of a specific prototype:

```
[34]: struct.begin {refkey=[0]}
```

```
ID = 34
```

```
SID = SQ01_Stash
```

```
Type = EGlobalVariableType::Bool
```

```
DefaultValue = False
```

```
struct.end
```

THEN the order of ID fields and the unique name of the prototype [34] : *struct.begin {refkey=[0]}*, where [34] must not be repeated

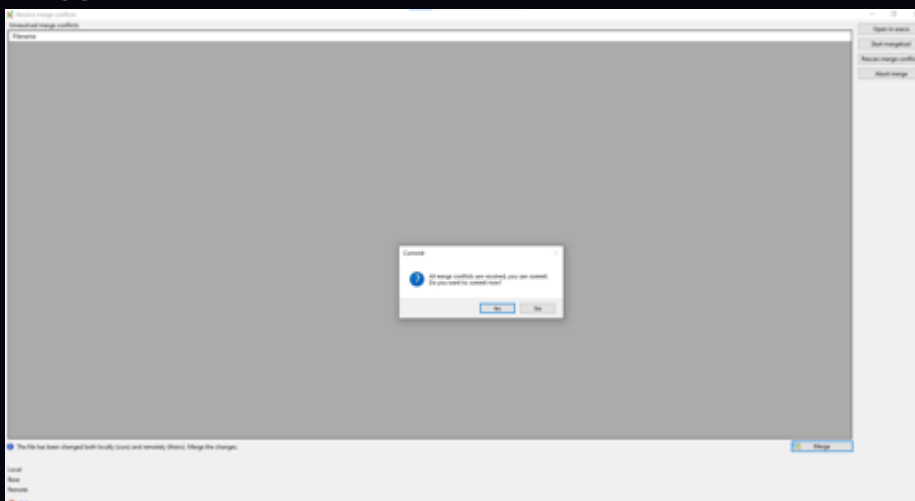
Example of the correct order:

```
[32] : struct.begin {refkey=[0]}
ID = 32
SID = E70_MarkerOnItemContainer
Type = EGlobalVariableType::Bool
DefaultValue = False
Struct.end
[33] : struct.begin {refkey=[0]}
ID = 33
SID = E02_SQ01_ZorikStash
Type = EGlobalVariableType::Bool
DefaultValue = False
Struct.end
[34] : struct.begin {refkey=[0]}
ID = 34
SID = SQ01_Stash
Type = EGlobalVariableType::Bool
DefaultValue = False
struct.end
```

To simplify the process of resolving merge conflicts, it is better if the config does not contain the information above, but is of this type:

```
SQ03_Stash_MastifReward : struct.begin {refkey=[0]}
SID = SQ03_Stash_MastifReward
Type = EGlobalVariableType::Bool
DefaultValue = False
struct.end
```

5. Press **Ctrl + S** and return to the window with conflicting files. If the files are still there, continue resolving conflicts in specific files. If they are not there, you will see such a window:



You can commit your work right away or double-check changes and commit after that.

6. If something was resolved incorrectly, return to the previous commit (the commit before the merge) and repeat.

