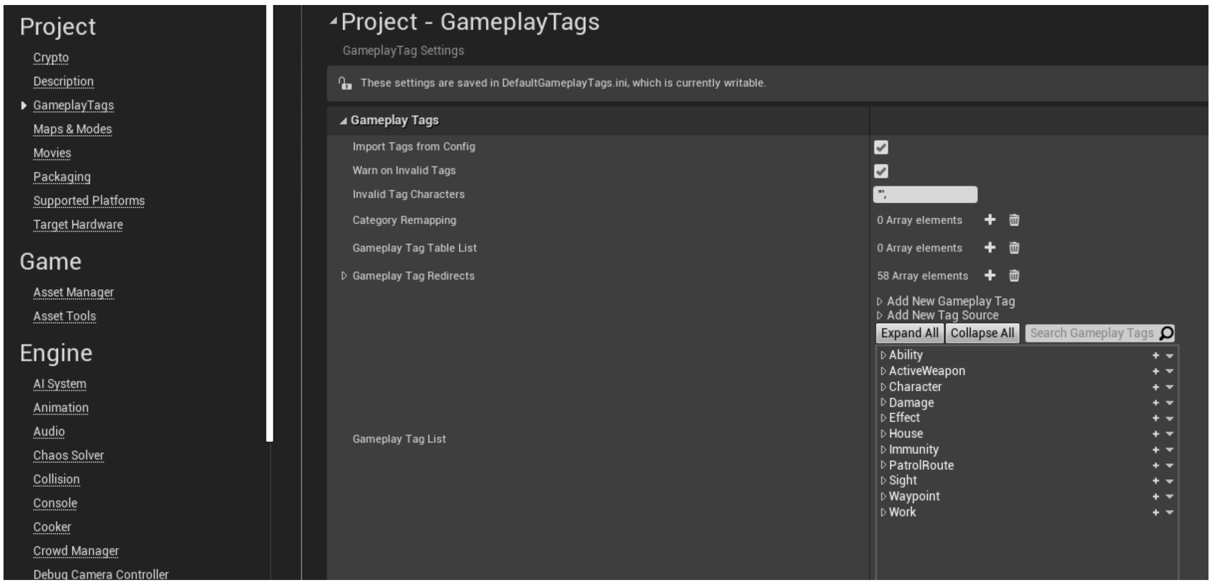
Each system has its own separate doc

How to Migrate

Gameplay Tags

If you want to migrate RPG Tools, before you do this, you need to copy gameplay tags.
If you **create new project** from downloaded RPG Tools content and open it, in project settings you will find Gameplay Tag List. These tags are used in gameplay logic.



Project - GameplayTags

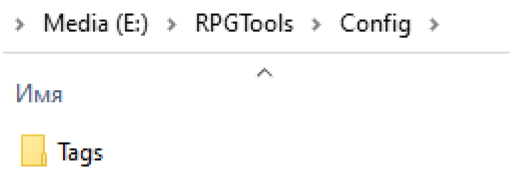
GameplayTag Settings

These settings are saved in DefaultGameplayTags.ini, which is currently writable.

Gameplay Tags

Import Tags from Config	✓
Warn on Invalid Tags	✓
Invalid Tag Characters	",
Category Remapping	0 Array elements
Gameplay Tag Table List	0 Array elements
Gameplay Tag Redirects	58 Array elements
Gameplay Tag List	Add New Gameplay Tag Add New Tag Source Expand All · Collapse All · Search Gameplay Tags Ability ActiveWeapon Character Damage Effect House Immunity PatrolRoute Sight Waypoint Work

In order to copy them to your own project you need to open config folder and copy Tags folder to your project



Media (E:) › RPGTools › Config ›

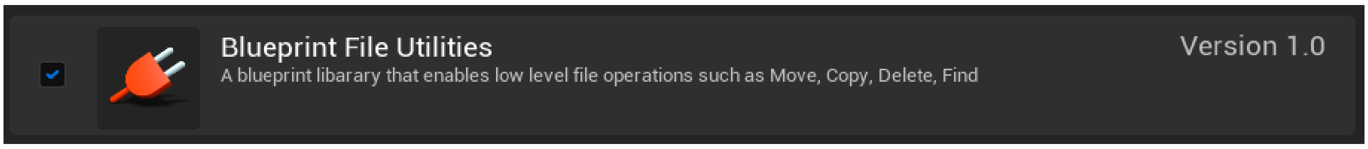
Имя

Tags

You can open your project and make sure the tags appear in project settings.
Only after this you can migrate all RPG Tools content to your project.

Plugins

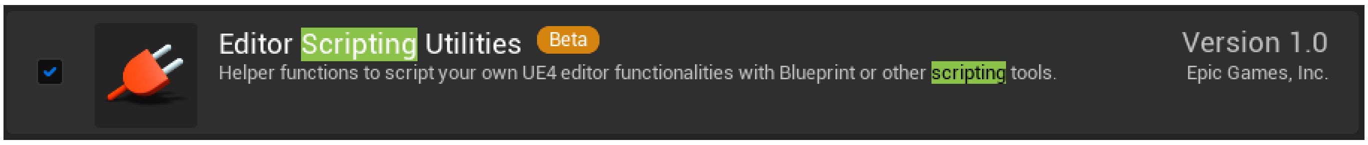
These plugins should be enabled in your project.



Blueprint File Utilities

A blueprint libarary that enables low level file operations such as Move, Copy, Delete, Find

Version 1.0



Editor Scripting Utilities Beta

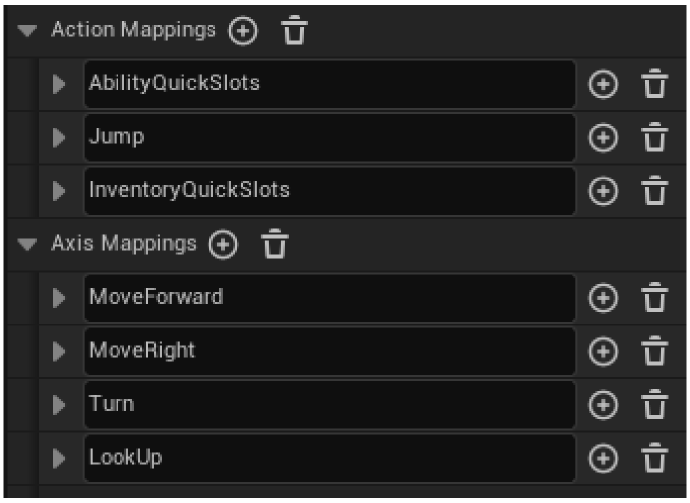
Helper functions to script your own UE4 editor functionalities with Blueprint or other scripting tools.

Version 1.0

Epic Games, Inc.

Inputs

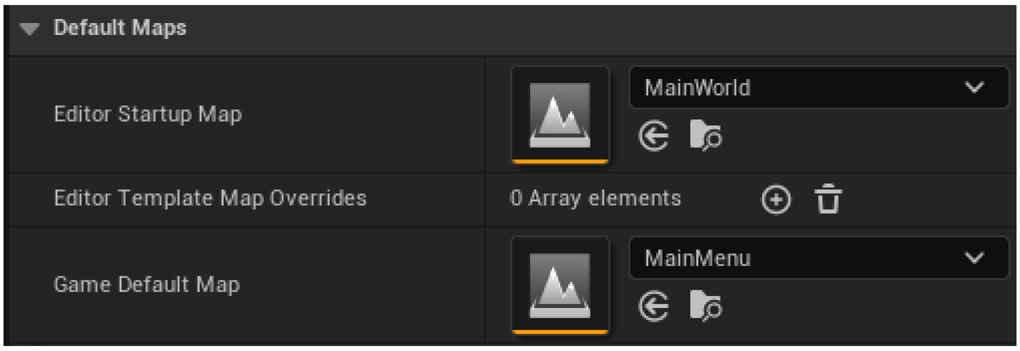
Project Settings -> Inputs -> Create the next inputs



AbilityQuickSlots
Jump
InventoryQuickSlots
MoveForward
MoveRight
Turn
LookUp

Maps and Modes

Project Settings -> Maps and Modes -> Set Maps and Game Instance

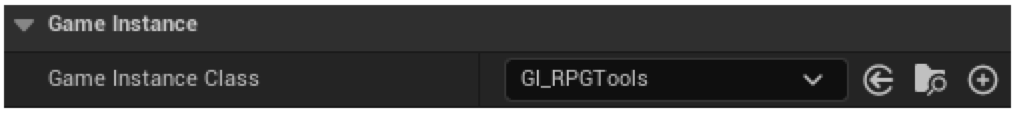


Default Maps

Editor Startup Map: MainWorld

Editor Template Map Overrides: 0 Array elements

Game Default Map: MainMenu



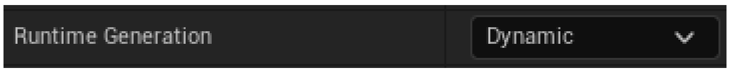
Game Instance

Game Instance Class: GI_RPGTools

Navigation

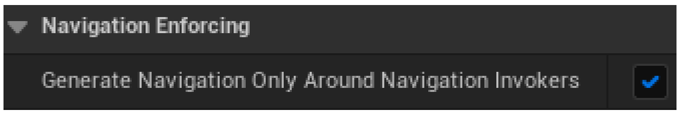
My AI generate navmesh around itself as Invoker. To enable Navigation Invoker feature you need to change the following settings.

Project Settings -> Navigation Mesh



Runtime Generation: Dynamic

Project Settings -> Navigation System

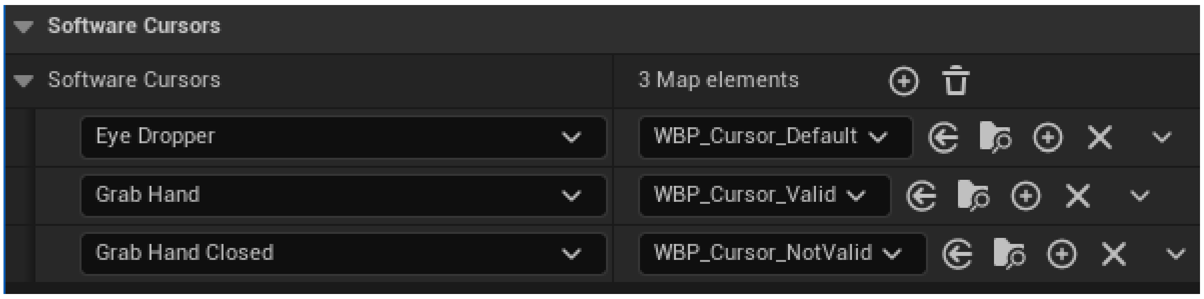


Navigation Enforcing

Generate Navigation Only Around Navigation Invokers ✓

Software Cursors

This is the only visual feature for cursors in Top Down mode.
Project Settings -> User Interface



Software Cursors

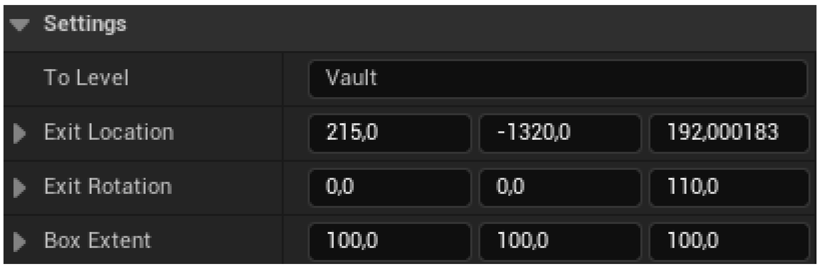
Software Cursors	3 Map elements
Eye Dropper	WBP_Cursor_Default
Grab Hand	WBP_Cursor_Valid
Grab Hand Closed	WBP_Cursor_NotValid

Teleport between levels

Let’s create a teleport between two levels: **MainWorld** and **Vault**.

Open MainWorld and Place **BP_Teleport** on the scene.

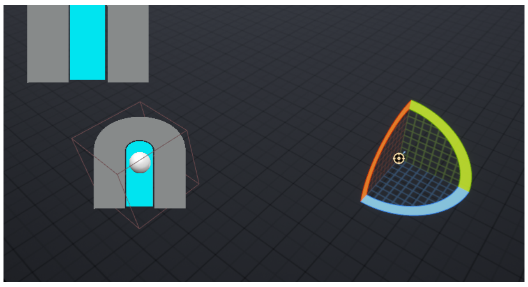
Select it and set settings.

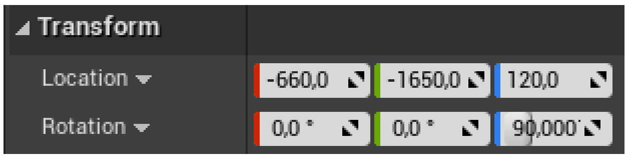


Settings

To Level	Vault		
Exit Location	215,0	-1320,0	192,000183
Exit Rotation	0,0	0,0	110,0
Box Extent	100,0	100,0	100,0

In order to get **ExitLocation** and **ExitRotation**, open Vault, place any actor you want (for example,Target Point) in place where you want the Player to spawn, and copy Location and Rotation(only Yaw).

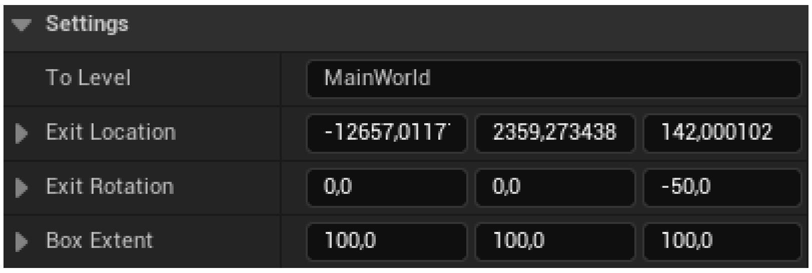




Transform

Location	-660,0	-1650,0	120,0
Rotation	0,0 °	0,0 °	90,000°

Do the same for BP_Teleport in Vault.



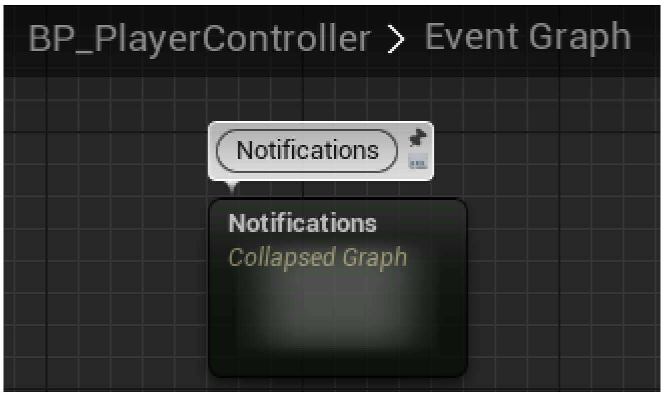
Settings

To Level	MainWorld		
Exit Location	-12657,0117	2359,273438	142,000102
Exit Rotation	0,0	0,0	-50,0
Box Extent	100,0	100,0	100,0

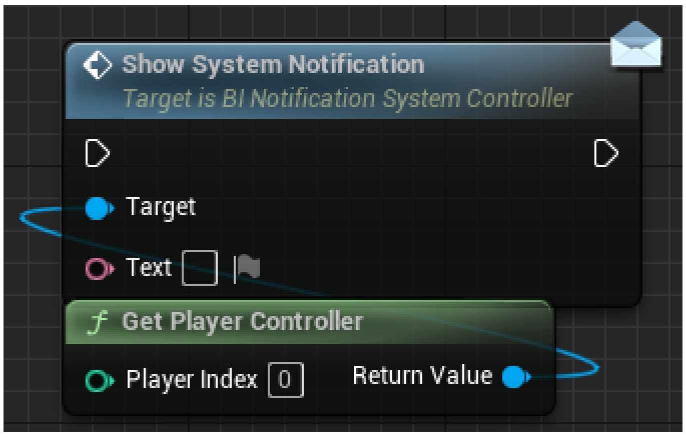
BI_Game_GI interface is implemented in **GameInstance**.
When the player teleports, **TeleportToLevel** interface function is called.
All game progress is saved, ExitLocation and ExitRotation are also saved.
Then a new level is opened, and in Player Controller (Begin Play), **LoadPlayerLocation** interface function is called.

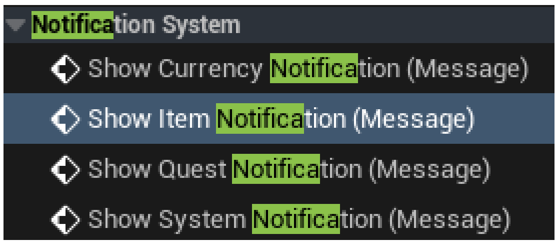
Notification System

BI_NotificationSystem_Controller interface is implemented in Player Controller.

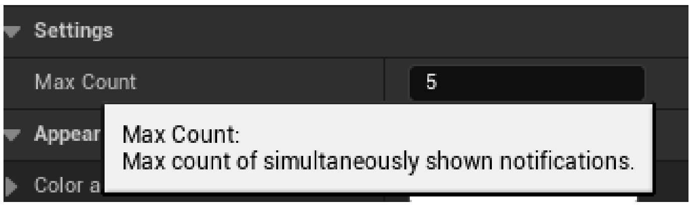


Just call the function from any place you want.




Notification System
Show Currency Notification (Message)
Show Item Notification (Message)
Show Quest Notification (Message)
Show System Notification (Message)

Widgets are not placed on HUD. I add them to Viewport in order to display notifications above all windows.
You can open **WBP_ItemNotificationContainer** and **WBP_CurrencyNotificationContainer** and set **MaxCount** default value.



Max Count 5

Max Count:
Max count of simultaneously shown notifications.

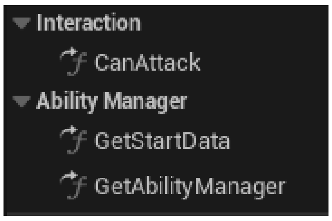
BP_Object

All static world actors with which player can interact or destroy should be created from **BP_Object**. BP_Object has all core functionality:

- core interfaces are added
- core parameters are created
- core logic for class is created

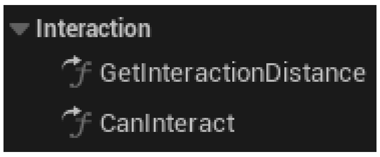
Note that the logic for the following functions are not created in BP_Object: **CanInteract**, **CanAttack**, **GetInteractionDistance**. That means every time they will return False. When you create a child class, you need to create logic for them.

See my **BP_Tower**.
I override and create logic for CanAttack in order for the player can attack tower.
Ability System interface functions are also overridden and implemented.



On Event Graph I create logic especially for tower.

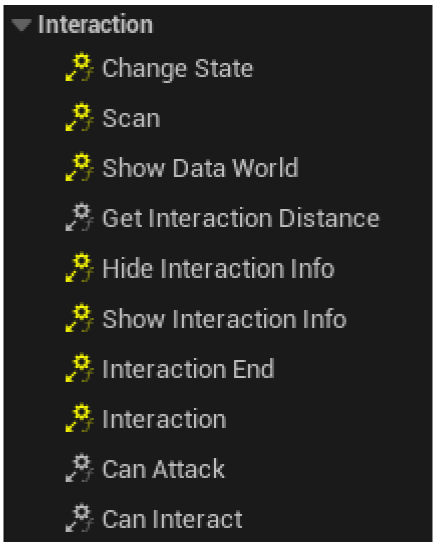
See my **BP_LootActor**.
I override and create logic for the next functions in order for the player can interact with actor.



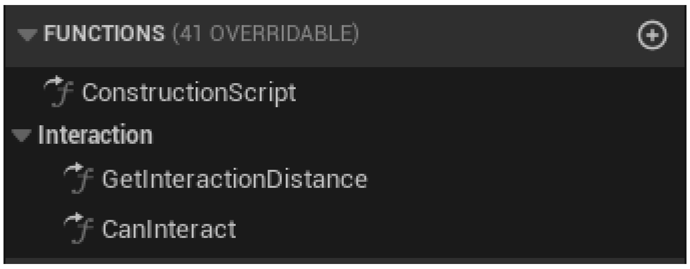
On Event Graph I create logic especially for tower.

Create new actor for Activate/Destroy objective

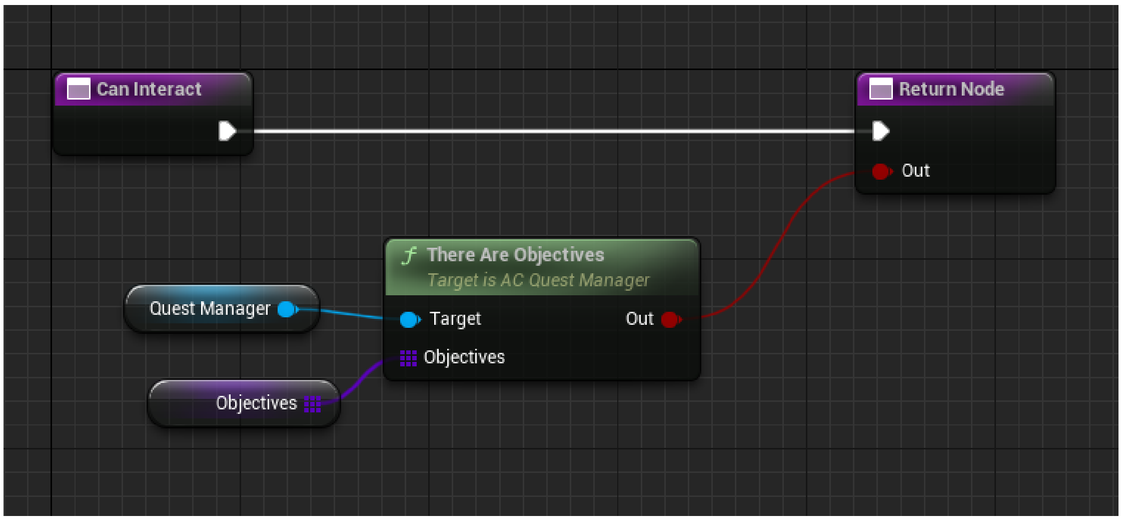
1. Create new objective with Activate/Destroy type. Set Index and Amount parameters (must have)
2. Add this objective to quest settings, ofc
3. Create new actor as child of **BP_Object.** Set mesh and create actor logic for this actor
4. Be sure to override and implement CanInteract, GetInteractionDistance, CanAttack (if actor can be attackable). Just double click on gray functions



5. Add logic for overridden functions



For example, if you want the player to interact with actor only if objective is activated

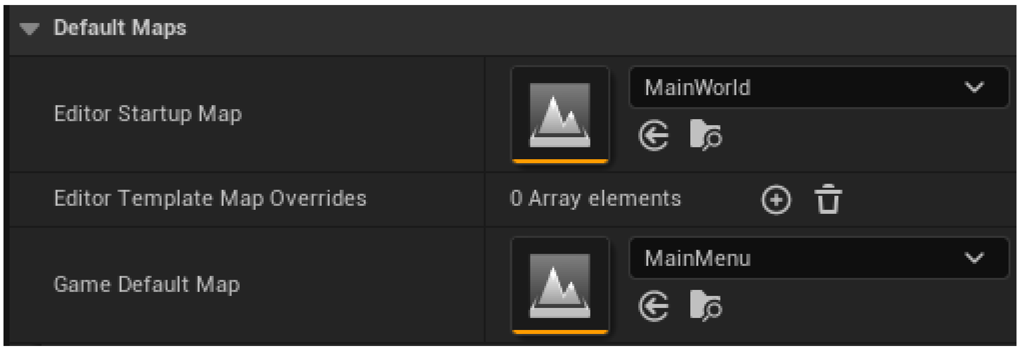


6. Place actor on the scene and add new objective to Objectives parameter in Details

Menu

Main Menu

MainMenu map is opened at start game.



Default Maps

Editor Startup Map: MainWorld

Editor Template Map Overrides: 0 Array elements

Game Default Map: MainMenu

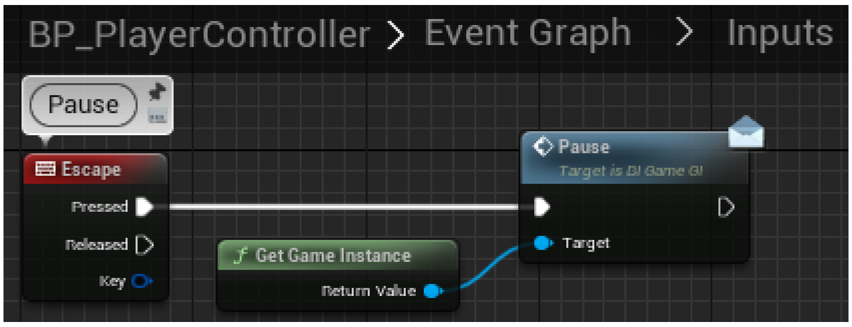
Project Settings -> Maps and Modes

In **GM_MainMenu** I create a **WBP_MainMenu** widget.

Pause Menu

BI_Game_GI interface is implemented to Game Instance.

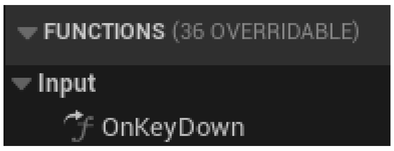
Pause interface function is called when using **Esc**.



WBP_PauseMenu is created and added to Viewport.

On Construct I set **Input Mode UI Only** and set **focus** to this widget.

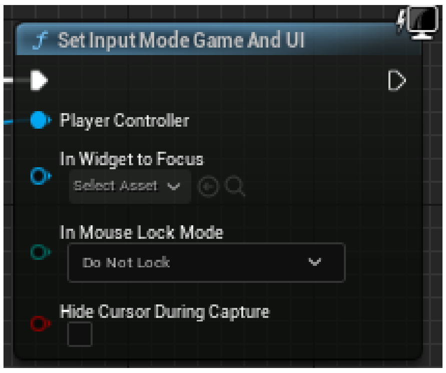
Focus is used to implement OnKeyDown event.



When Esc is pressed (this is handled in OnKeyDown), I call **Pause** function again.

When Pause is false, widget is removed from Viewport, and OnDestruct event is called.

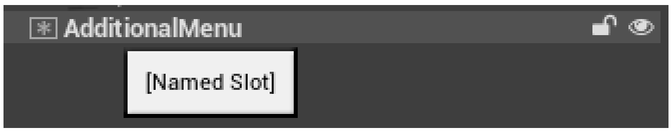
On Destruct I set **Input Mode Game and UI**.



NamedSlot is added to Pause Menu.

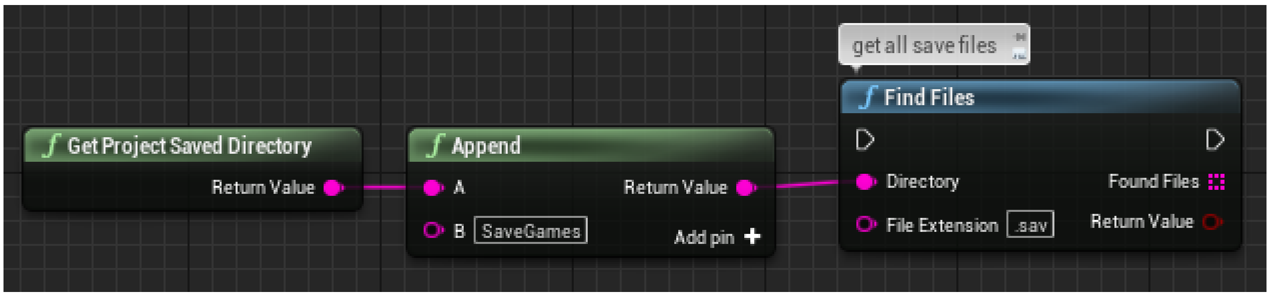
When we open the next menu, for example, Save Menu, it will be added to this slot.

This is necessary so that when we close Save Menu, keyboard focus will automatically be set to Pause Menu again.

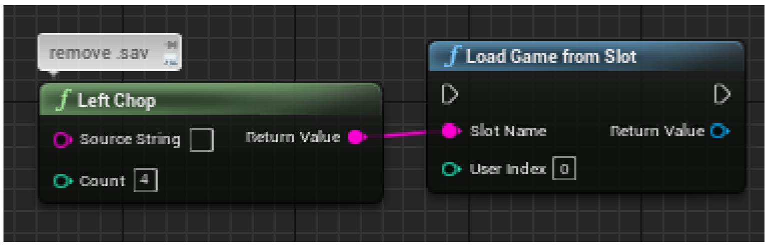


Continue

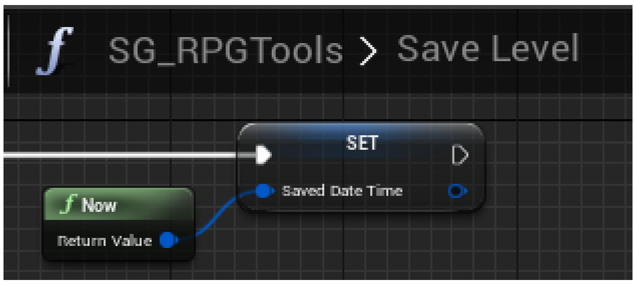
GetAllSaveSlots interface function is implemented in **GI_RPGTools**.
I use these nodes to get all save files.



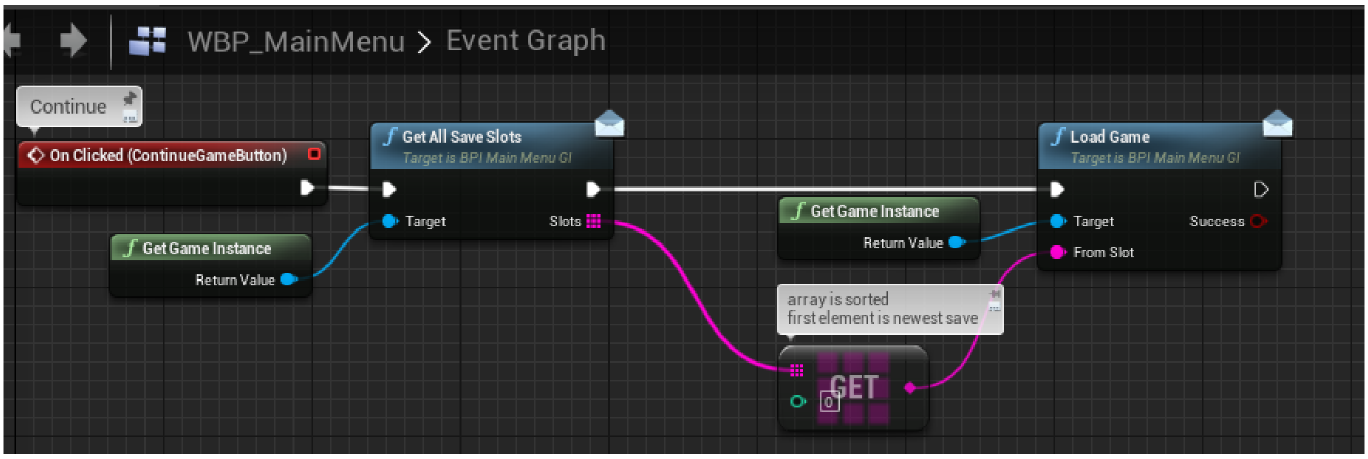
Then I remove file extension from file name and get Save Game class.



When I save the game I also save Date Time

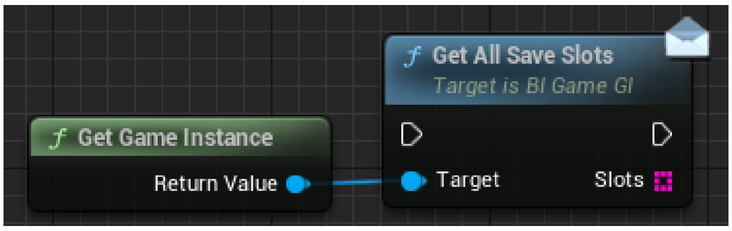


Comparing Date Time I can sort an array.

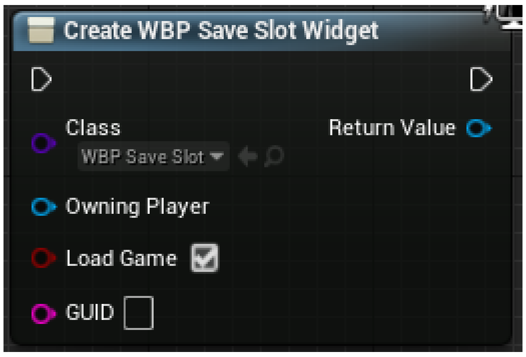


Load Menu

GetAllSaveSlots interface function is implemented in Game Instance.

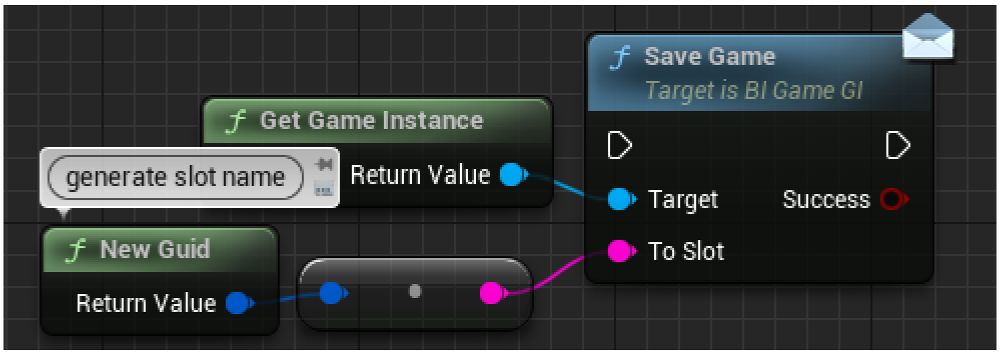


This function returns all files with **.sav** extention except **Settings.sav**, since this slot is used only for Game Settings. Array is sorted: first save is newest.
Since I use the same widget for both Load Menu and Save Menu, I set LoadGame = True.
GUID is file name.

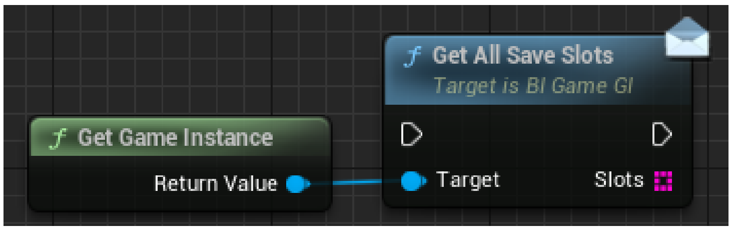


Save Menu

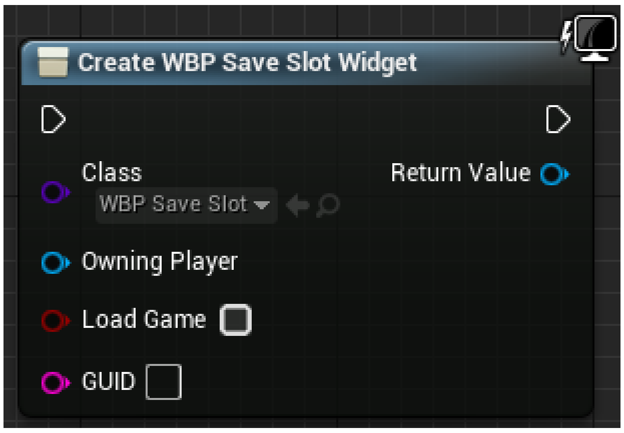
There are no save slots at the start of the game.
To create a new save I use a special button.
SaveGame interface function is implemented in Game Instance.
I use **NewGuid** function to generate a slot name (file name).



Also I generate all existing save slots, so the player can save the game to already existing slot.
GetAllSaveSlots interface function is implemented in Game Instance.



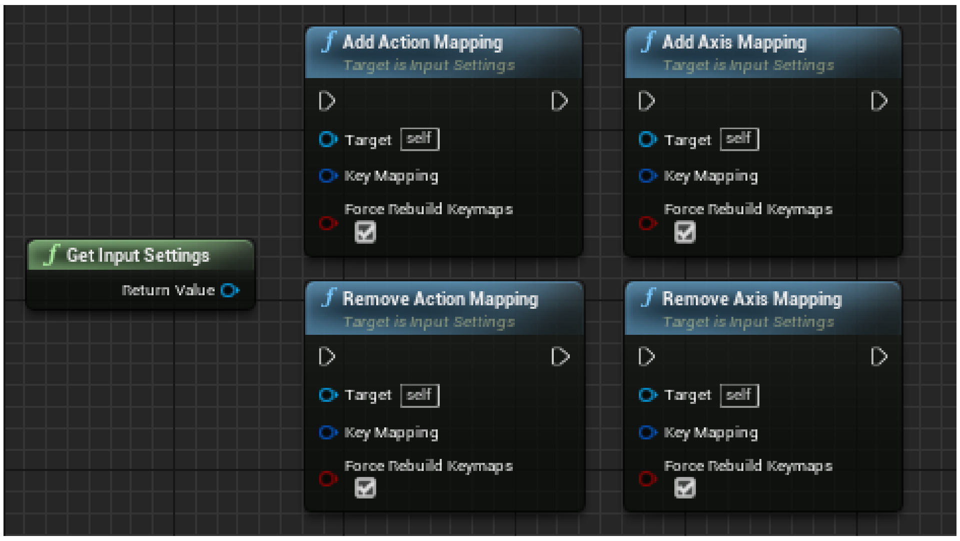
This function returns all files with **.sav** extention except **Settings.sav**, since this slot is used only for Game Settings. Array is sorted: first save is newest.
Since I use the same widget for both Load Menu and Save Menu, I set LoadGame = False.
GUID is file name.



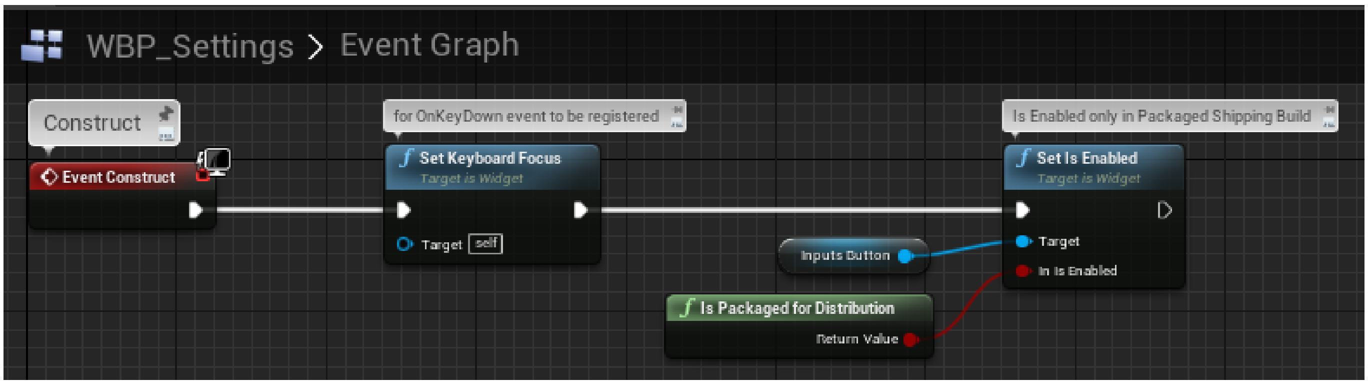
Input Settings

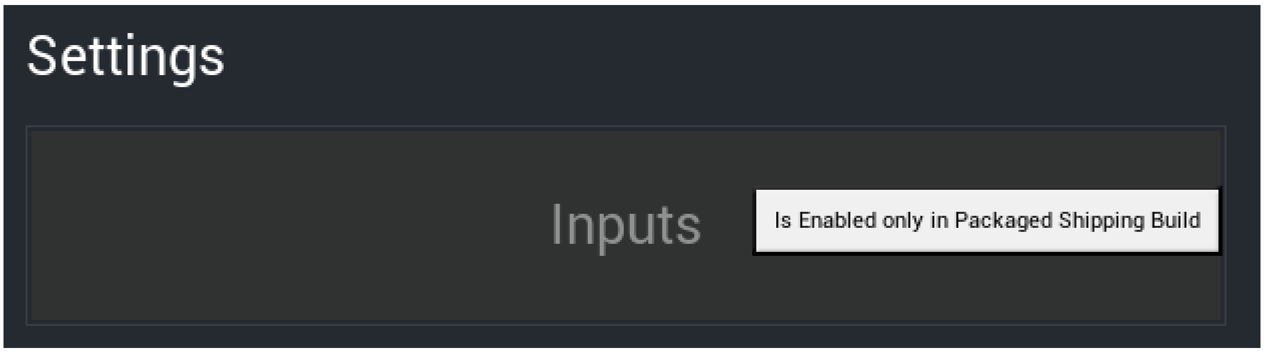
PIE Mode

If you use the following nodes in Play-In-Editor mode, you will directly change the config file of your project.



Using such logic, I make Input Settings available only for Shipping Build.



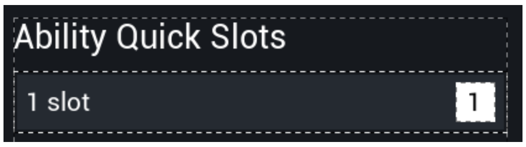


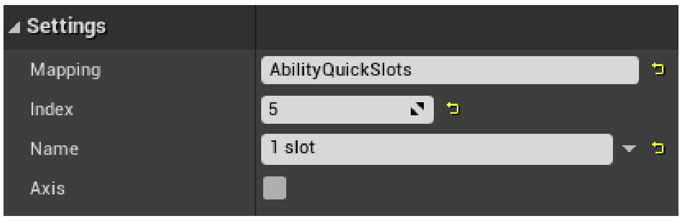
You can disable this node, run the game in PIE mode, open Project Settings -> Inputs, and you will see how inputs will be changed.
In Standalone or Shipping Build config file is not changed.

Add New

Open **WBP_Inputs** and place **WBP_InputButton** where you want.

Now you need to bind this button to mapping from Project Settings -> Inputs.
Let's look at the next button.

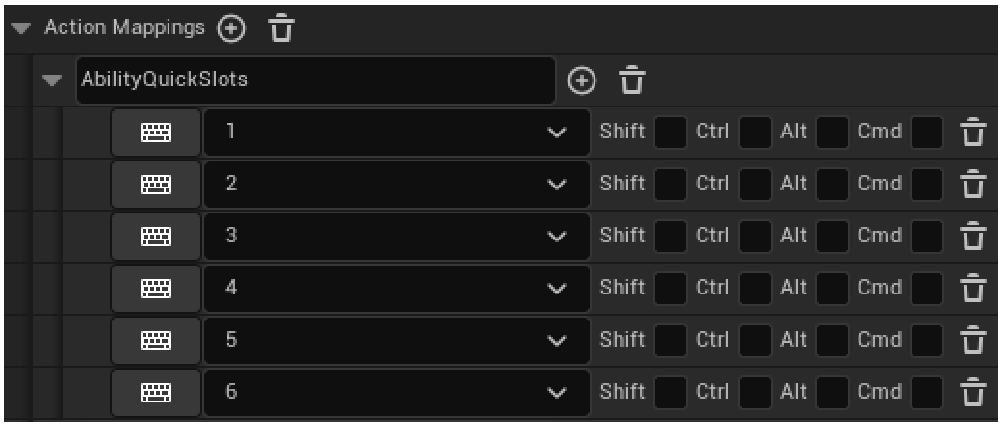




Settings

Mapping	AbilityQuickSlots
Index	5
Name	1 slot
Axis	☐

At first look it seems we need to set Index = 0, since '1' is the first element in the array.

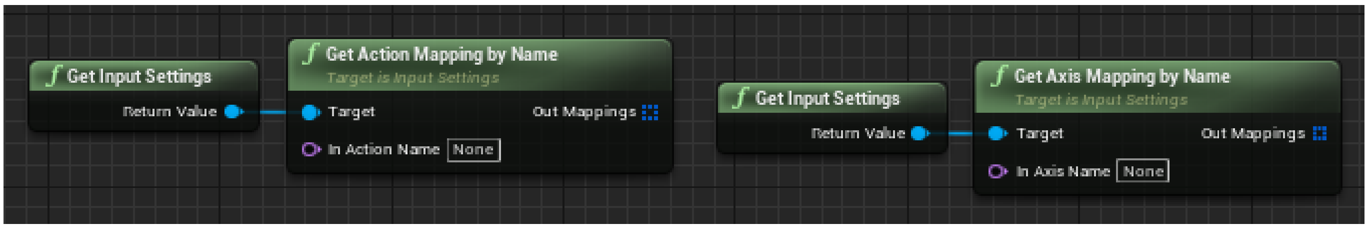


Action Mappings

AbilityQuickSlots

1	Shift ☐	Ctrl ☐	Alt ☐	Cmd ☐
2	Shift ☐	Ctrl ☐	Alt ☐	Cmd ☐
3	Shift ☐	Ctrl ☐	Alt ☐	Cmd ☐
4	Shift ☐	Ctrl ☐	Alt ☐	Cmd ☐
5	Shift ☐	Ctrl ☐	Alt ☐	Cmd ☐
6	Shift ☐	Ctrl ☐	Alt ☐	Cmd ☐

But it is not like that. When you use these nodes you will get an array (6,5,4,3,2,1), where '1' will be the last element. Just consider it.



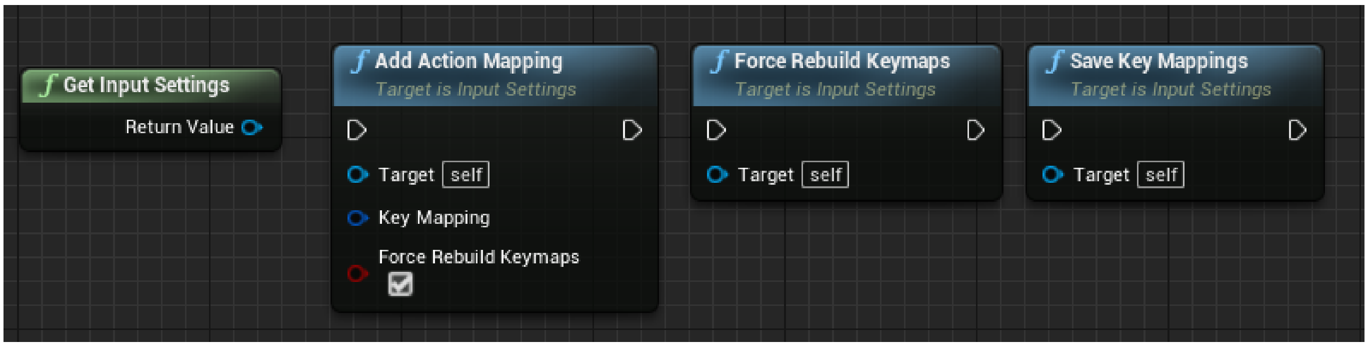
Modifying

First I want to say that all info from Project Settings -> Inputs is stored in config file

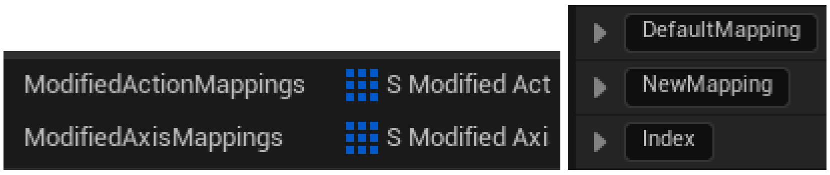
If you run the game in Play-In-Editor mode, it is enough to use only **AddActionMapping**, and config file will be changed.

SaveKeyMappings is used to change config file in Standalone or Packaged Build.

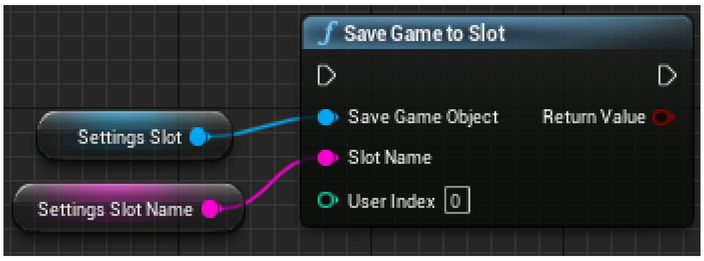
I don't use it so I don't modify config file.



I have save game class **SG_RPGTools_Settings** where I store all modified Action and Axis Mappings.



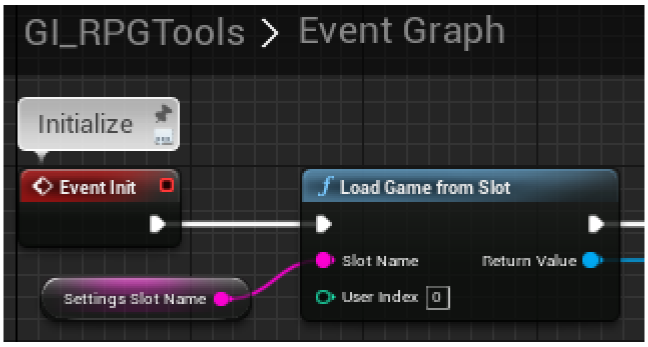
When I modify settings I save all changes in a separate slot.



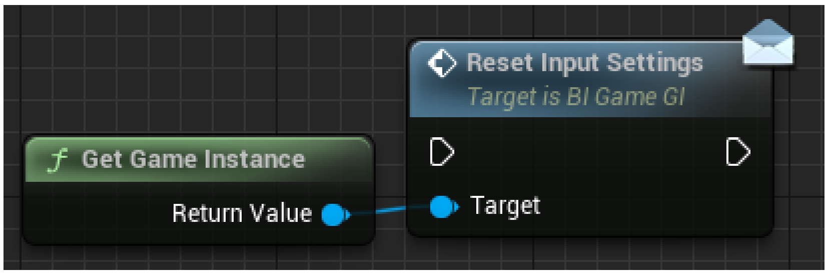
Config file always contains default inputs.

When game starts, Initialize event is triggered in Game Instance class.

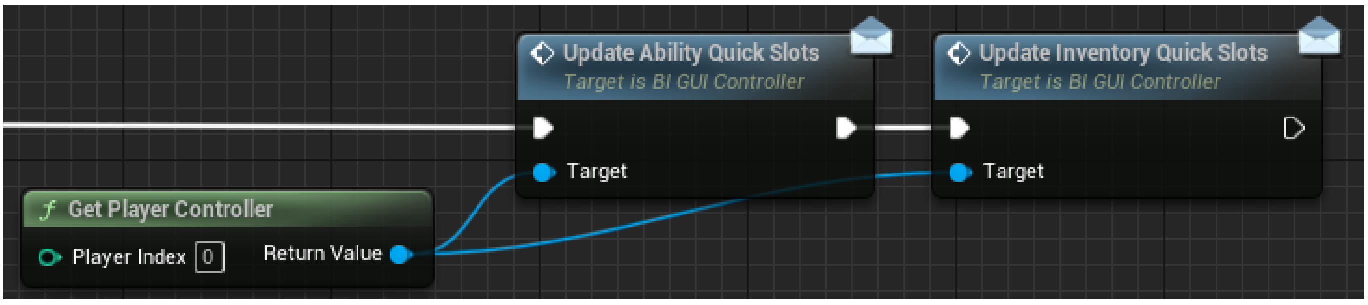
I load save slot with settings and update Mappings.



Since structures store **DefaultMapping** variable I can easily Reset Input Settings.

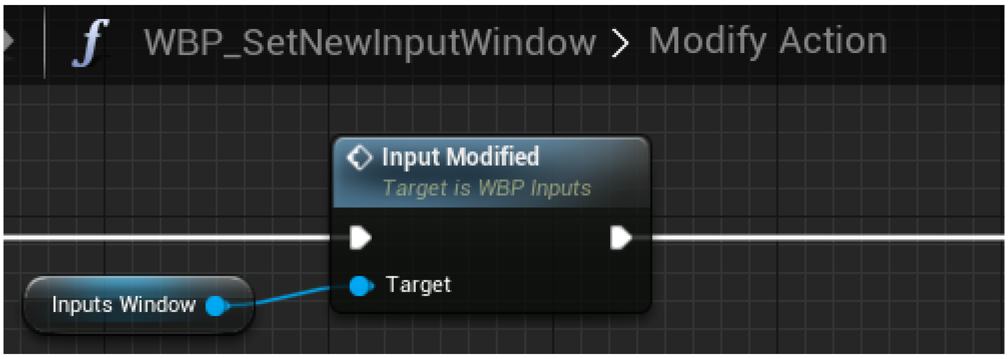


After Action is changed or All Settings are reset I update QuickSlots

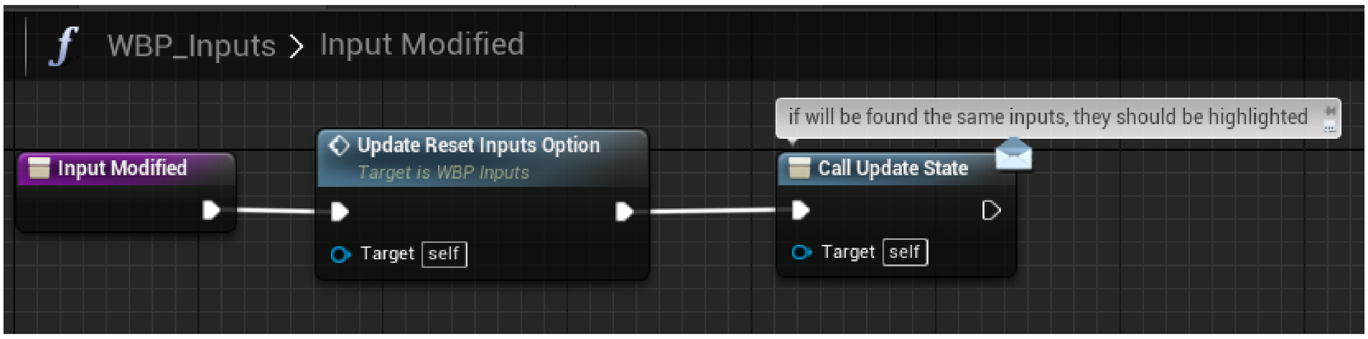

Update Ability Quick Slots
Target is BI GUI Controller
Target
Update Inventory Quick Slots
Target is BI GUI Controller
Target
Get Player Controller
Player Index 0
Return Value

Highlight the same

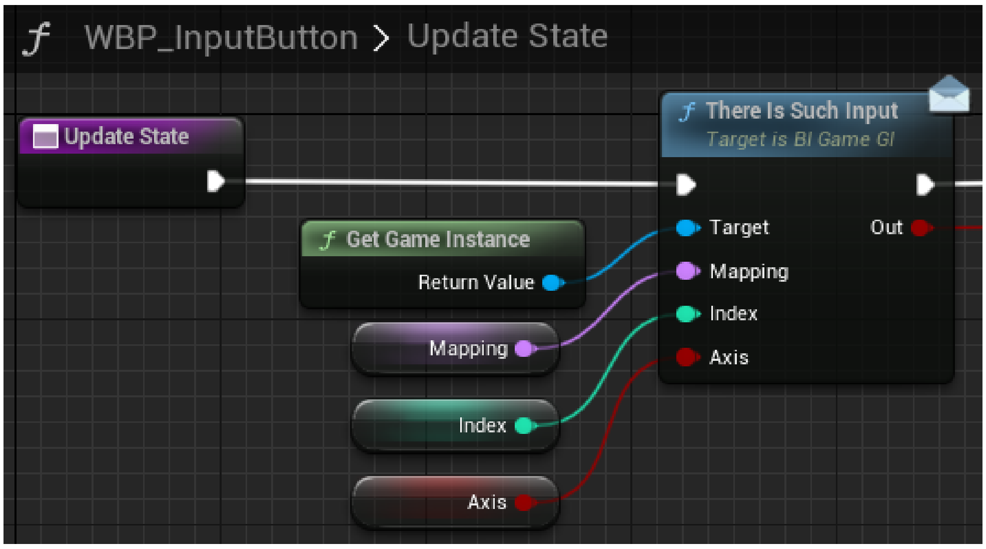
Here we press a button and modify the input.



Here we update states after modifying.


WBP_Inputs > Input Modified
if will be found the same inputs, they should be highlighted
Input Modified
Update Reset Inputs Option
Target is WBP Inputs
Target self
Call Update State
Target self

All WBP_InputButton widgets are bound to UpdateState event.



Change Log

2.0.1

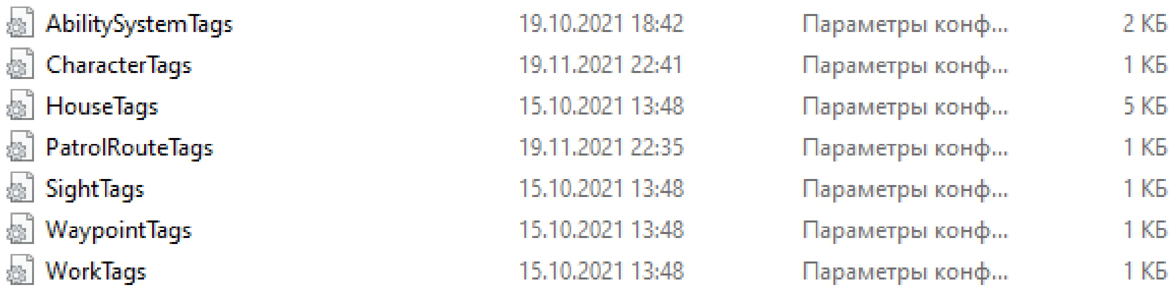
20.11.2021

Quest System

- See 2.1

Overall

- Hotfix: Tags from **DefaultGameplayTags.ini** are copyed to **PatrolRouteTags.ini** and **CharacterTags.ini**



AbilitySystemTags	19.10.2021 18:42	Параметры конф...	2 КБ
CharacterTags	19.11.2021 22:41	Параметры конф...	1 КБ
HouseTags	15.10.2021 13:48	Параметры конф...	5 КБ
PatrolRouteTags	19.11.2021 22:35	Параметры конф...	1 КБ
SightTags	15.10.2021 13:48	Параметры конф...	1 КБ
WaypointTags	15.10.2021 13:48	Параметры конф...	1 КБ
WorkTags	15.10.2021 13:48	Параметры конф...	1 КБ

When you wanted to migrate RPGTools to your own project you needed to copy the Tags folder. Some tags were placed in DefaultGameplayTags.ini and were not copied. Now, all tags will be copied.

2.1

01 March 2022

Ability System

- See 1.1

Quest System

- See 2.2

Inventory

- See 3.1

RPG Tools

***Blueprint File Utilities** plugin is enabled. It is used for working with save slots.*

- **Teleport** between Worlds is improved

  Now you can set Location and Rotation where the player will spawn after teleporting.

  Now the interface is used and teleport knows nothing about Game Instance class.

  *More in **Teleport between Worlds** section.*
- **Notification System** is improved

  Now there are Exp, Currency, Item, Quest and System notifications.

  *More in **Notification System** section.*
- **Menu** is added
  - Continue (*load last save*)
  - New game
  - Save Game to Slot
  - Load Game from Slot
  - Input Settings
  - Back to Main Menu
  - Quit

  *More in **Menu** and **Input Settings** sections.*
- Fixed issue. Now **Day Night Cycle** data is saved correctly

  Earlier, if the game is started in the world without the Manager, when teleporting to world with the manager, incorrect data is loaded and game is started at night.

  Appropriate check is added to manager (Begin Play event).
- Fixed issue. Now the player cannot Interact with AI in ‘Combat’ state

  ***Character.Combat** tag is added to **InteractionBlockedTags** (AbilityManager).*
- Fixed issue. Now the player cannot jump when he is dead or has control effect

  ***JumpBlockedTags** variable and **CanJump** function are created in AbilityManager.*

  The appropriate check is added to player controller.

2.2

07 July 2022

Inventory System

- See 3.2

Quest System

- See 2.3

RPG Tools

- New Inventory and Quest systems are added
- Demo scene is reworked
- Issue with BP_IceArea and BP_FireArea is fixed
  Areas attacked the player even if he was not in them.
- Issue with Merchant Window is fixed.
  Sometimes player could put items on Warehouse. It’s because the system incorrectly determined when the warehouse window was opened
- *Now RPG Tools is compatible only with UE5 and newer versions*