Thank you for buying this asset from the Unity Store, with your contribution with the purchase it will help me to be able to pay for courses and continue learning more about the world of videogames and create great Universes that excite players

Pro menu system English

Alexander Cirac Antio 22/06/2022

INSTRUCTIONS FOR PRO MENU SYSTEM

Version 1.0 6/22/2022

– All pro content

Version 2.0 3/25/2023

-UI Manager Opt8: Lock/Unlock buttons

-UI Manager Opt9: Timers

INDEX

INTRODUCTION

Thank you very much for purchasing the package! I hope it will be very useful for you. You don't have to be a programmer to use it. The version of this system is designed to support PC and mobile The content of this asset is; The following sections will be for the controls of your interface for menus or HUD. You have a core script that explains everything you need, and the number of scripts needed

1) Exit the game (Actions 1.1 and 1.2 can be activated with button/key/Joystic)
    1.1) Option to ask if you want to go out
    1.2) Directly

2) Button to load a level of unity (You can add more than 1 button) (both actions work Button/Key/Joistic)
    2.1) Question option to ensure player action 2.2) Directly

3) Game pause menu (Can be activated with button/key/Joystic)
    3.1) Stop time when the game is paused or continues in the background

4) Open a URL by button

5) Play a video or cinematic (Unity and WebGL) (you can add more than 1 cinematic/video)
    5.1) You can project in Image, UI panel, 2D object or 3D Objects 5.2)
    Pause, play and remove button 5.3)
    Different ways to activate it; go through X 2D or 3D zone; by touching a button

6) UI panel controller (to enable/disable via button/key/custom key/joystick)

7) Button customization controllers, generic or each button its own customization
    7.1) You can change the image or add sound or add animation in the three states cOnClick, OnEnter and OnExit

8) Asynchronous loading
    screen 8.1) Customize the visualization by means of a loading bar; sound ; activate a UI element; internally
    8.2) Different ways to pass the screen when the button is loaded; key; joystick; automatically;

9) Put Mouse Icon (you can choose which scenes activate or deactivate them) (you can only open a mouse icon)
    9.1) Customization of the Mouse Icon images; animation; in the three actions OnClick, OnDown, OnUp

10) Lock / unlock buttons: Allows the buttons to be enabled for the gamer according to the need of the plot or the game, Eg: Enable hero abilities 10.1) You can have as many buttons as the
    user wants 10.2) Offers different ways to interaction to use
    this option 10.2.1) You can indicate each way if it works as a blocker or
        unlocker 10.2.2) You can have the same button multiple ways to block or unlock as
        the user wants 10.2.3) You can execute the blocking or unlocking by Tirigger 2D, 3D, with another button,
        at the end of the scene and finally by code (it is explained in the PDF how to do it, in a simple way and in only 1 step.

    10.3) Customize the appearance of the button to indicate its current state, if it is locked or unlocked
        12.3.1) It can be through Sprites, different colors or the basic one (white unlocked gray with opacity for locked)

11) Timers : This allows the user to add timers, but also allows you to read events when it activates or when it ends or deactivates E,: a timer to reinforce a base before the zombies come 11.1) You can have as many timers as the user wants 11.2) Offer different ways of interaction to activate this option

11.2.1) You can indicate each way if it works like starting the timer or stopping it prematurely
11.2.2) You can have the same timer multiple ways to start or stop 11.2.3) You can execute start or stop by Tirigger 2D, 3D, with another botton 11.3) You can add event when the timer starts and also events for when it ends 11.4) Customize the appearance of the timer

11.4.1) Through legacy text or mesh pro text, that changes images or with a dwarf animation

The following sections will be for the controls of your options. These options are subject to the choice of making the change of the options automatically or after pressing a button.

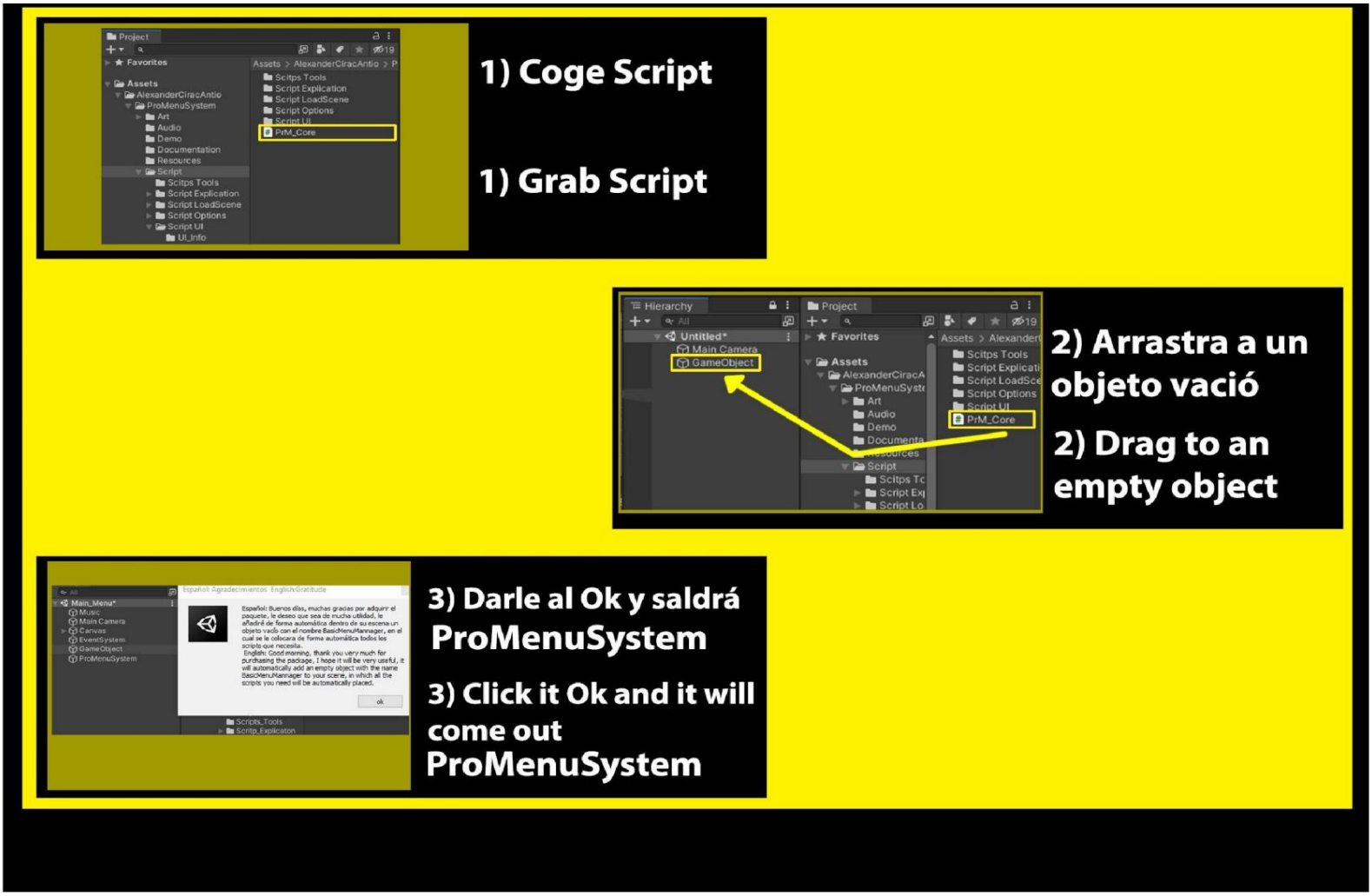
1) Place an option to control audios
2) Place a general brightness controller and another just for the scene
3) Place a Contrast Controller
4) Button for Vsync
5) Button to activate/deactivate shadows
6) Control limit FPS
7) Control the quality of the shadows
8) Anti-aliasing controller
9) Textures quality controller
10) Full Screen or Window Controller
11) Screen resolution
12) Reset button options
14) (Exclusive for mobile use,) Lefty Option

IMPORTANT!!!!!! It does not have a language option, because I am working separately with a new system that implements that section and more things to make it useful for the user

FIRST STEPS

In order to use this system, you just have to take the "PrM_Core" script and drag it to any object on the stage, it will automatically add an empty object with the name "ProMenuMannager" inside your scene, in which they will be placed automatically all the scripts you will need.PrM_Explication

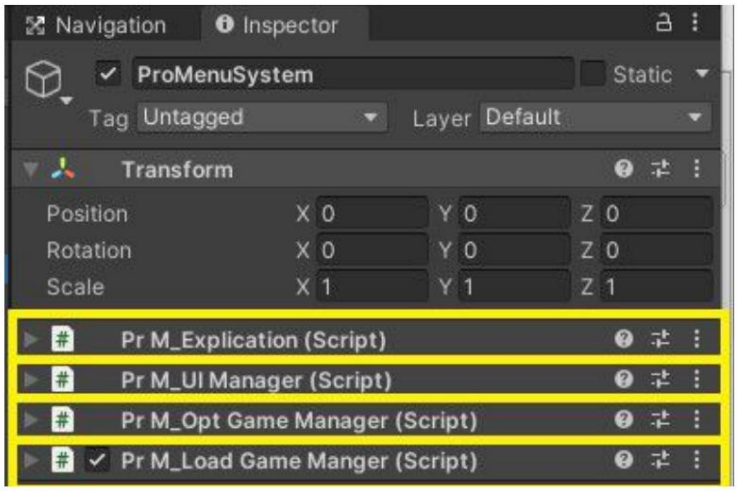
PrM_UIManager
PrM_OptManager
PrM_LoadGameManager



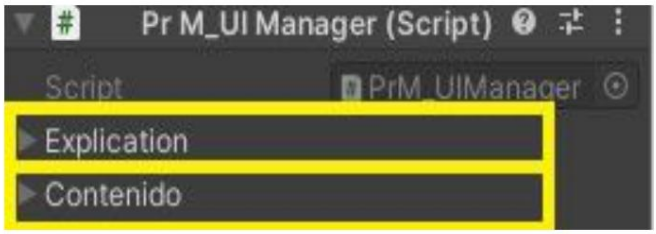
PRM_EXPLICATION

This script is only explanatory, since it does not contain any function or variables that are of an important nature, the only thing that it contains as relevant is a text, to explain what this entire menu system does and what each script that has been placed does. shape automatically
The first thing you will see are three important scripts

PrM_UIManager
PrM_OptGameManager
PrM_LoadGameManager



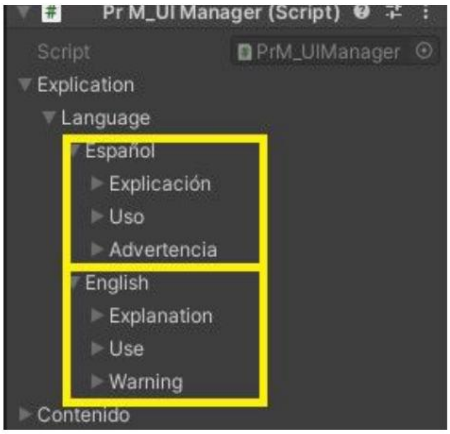
This Asset will allow you to customize the components of your menus, providing them with the functionalities that you designate. What ¡NO!, does is design, or rescale, or reorganize, or animate. That is already the task of the user himself. What it will allow you to do is assign animation elements and add new sprites. All the scripts contain an explanation and a content, in which you will have to add these elements to give them the functionalities that you want them to have. Each script, when you select it, will open two drop-downs “Explanation” and “Content”



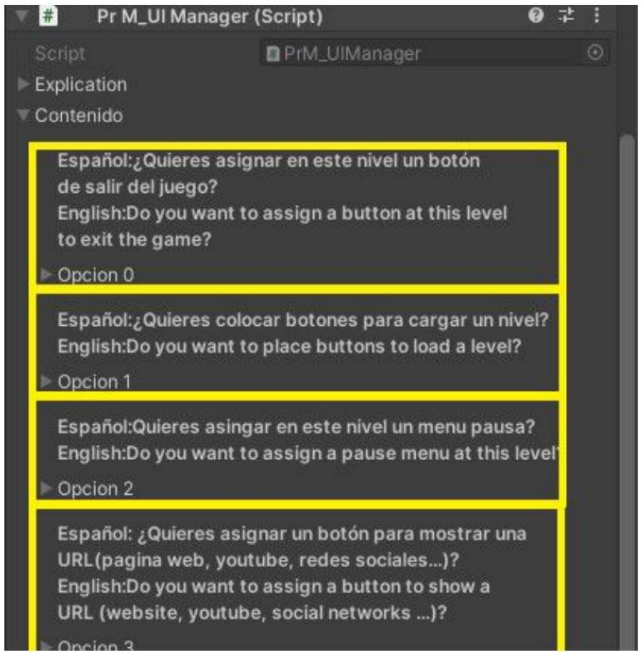
In the content of the "Explanation" section, the "Language" will be displayed and within each language, you will have the following points to read, if you are interested in finding out in a general way, what is the function of said script deployed.

1)Language: Here you can display the following sections corresponding to your language, for easy and quick understanding.

1.1) Explanation: Describes the function of said section 1.2) Use:
Indicates how you have to use it and what to do with this section 1.3) Warning:
Clarifies important points and warns the user of restrictions or facilitates how to access certain variables, in the in case you want to save information or retouch wby the user's own code outside this system



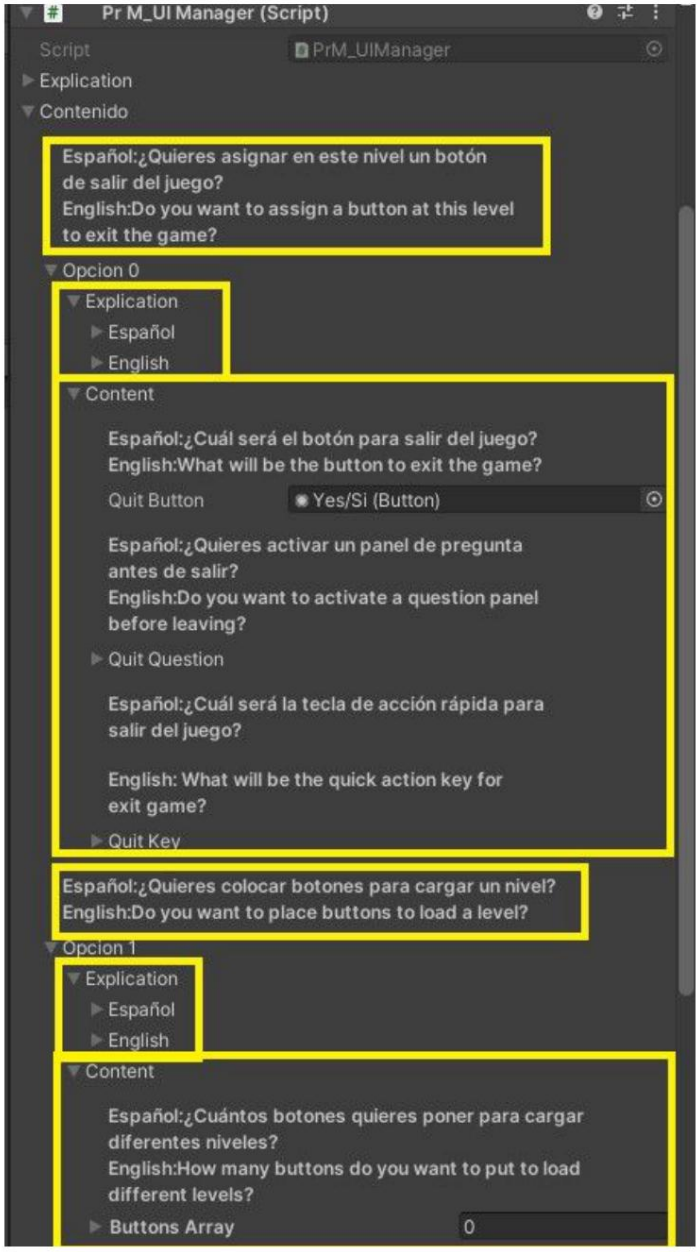
The "Content" is where all the options offered by the system will be found to provide functionality to the elements, based on asking the user in their respective languages, about what they want to do and functionality they want to give.



When you display the "Content" and select the section that interests you that said panel or button has that function, the following options will be displayed:

1) Explanation: Here it will be explained what is allowed to be done, where you will have your <Explanation> <Use> <Warning>

2) The sections to drag the components that you are interested in adopting or being affected by said function

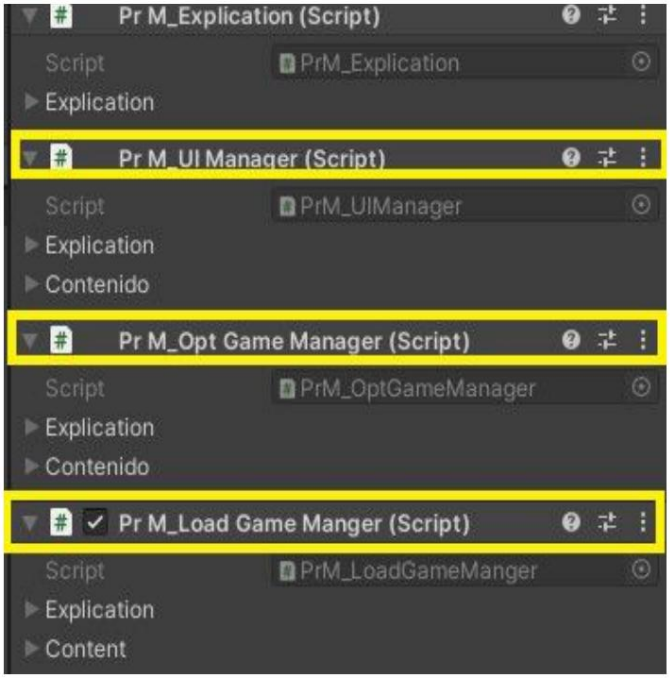


GENERAL EXPLANATION

Of a general nature, it has been divided into 3 scripts to differentiate which will be the buttons for menus, which will be specific for the part of the options

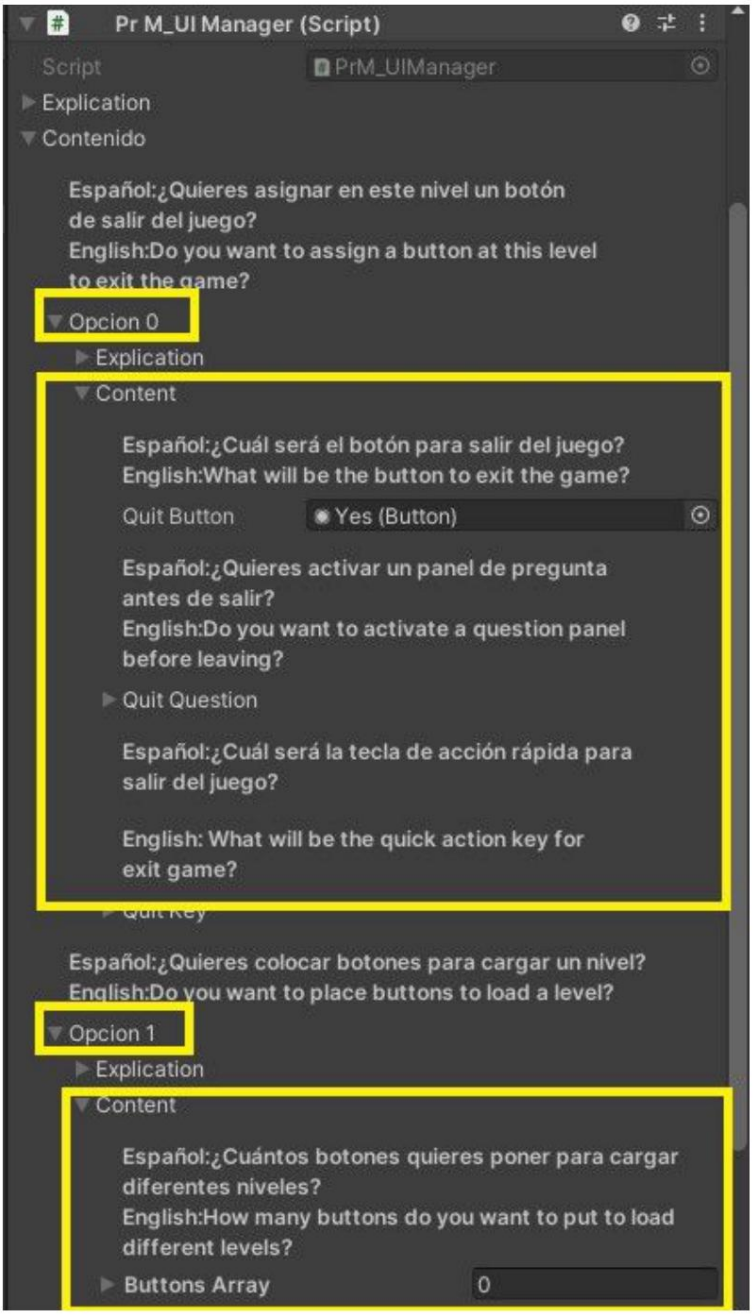
1)SC_MenuProPanelesUI: it will be used to place the main menu or pause buttons.
2) SC_MenuBasicoOptions: it will be used to assign the buttons that will function as Options Eg: graphically, brightness, shadows...
3) SC_ProMenuPantallCarga: it will be used to indicate if you want the game to have a loading screen, to load a new scene or an additive scene

In the event that you are interested in obtaining variables to save or reassign through your own scripts, you will be provided with how to access these variables in the <Warning> section, although these variables will not be visible in the inspector, because in this way it is achieved that the user's work area is as clean as possible



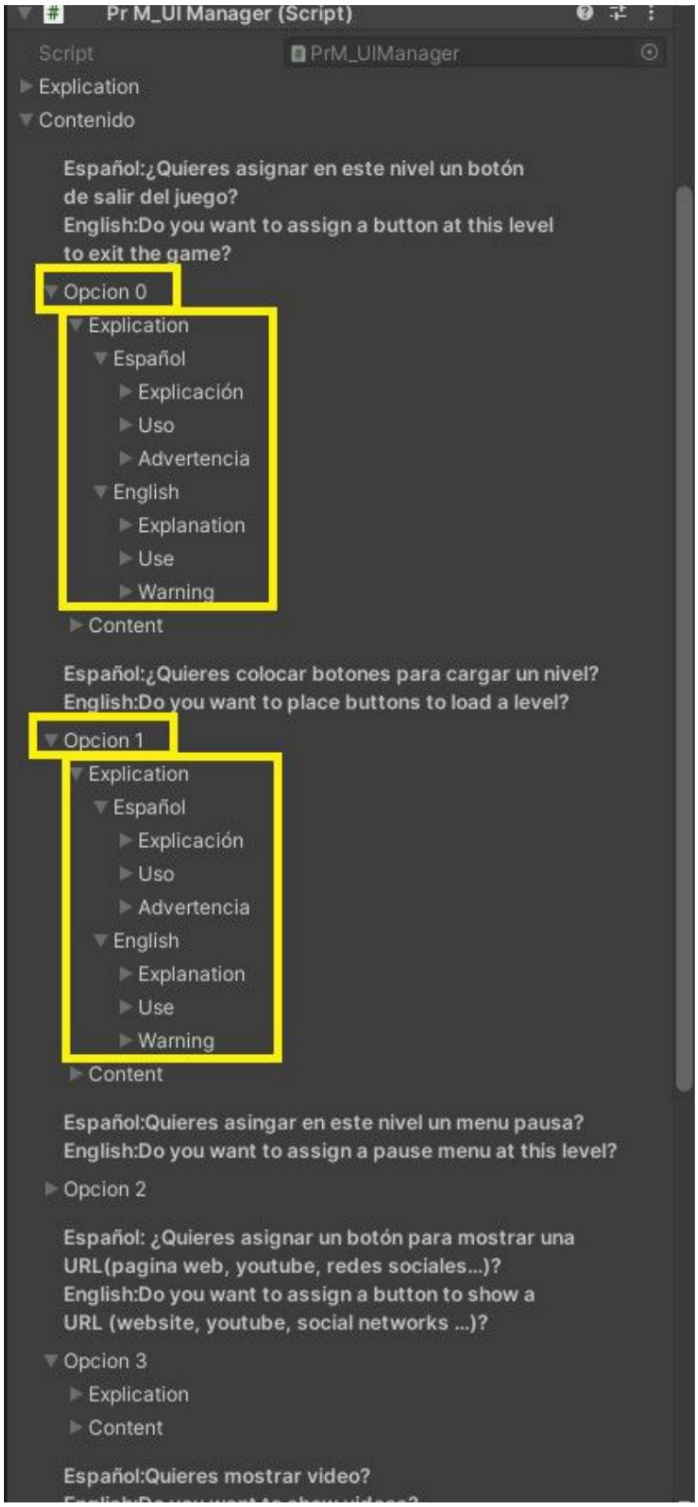
USE

Each section or section works as a selectable drop-down from which its content will appear. Remember that in <Content> you will see different questions in which you will provide the user with what utility you want to give to that specific button, you just have to select the question and fill in the gaps , some may contain more questions in case you want to do something extra.



WARNING

Scripts don't design, they don't retouch, they don't animate. They only give utilities to the buttons. In the event that each thing does not remember what it does, it has an explanation in Spanish as well as in English.



PRM_UIMANAGER

Like every important script, it has two essential parts, which are the following sections:

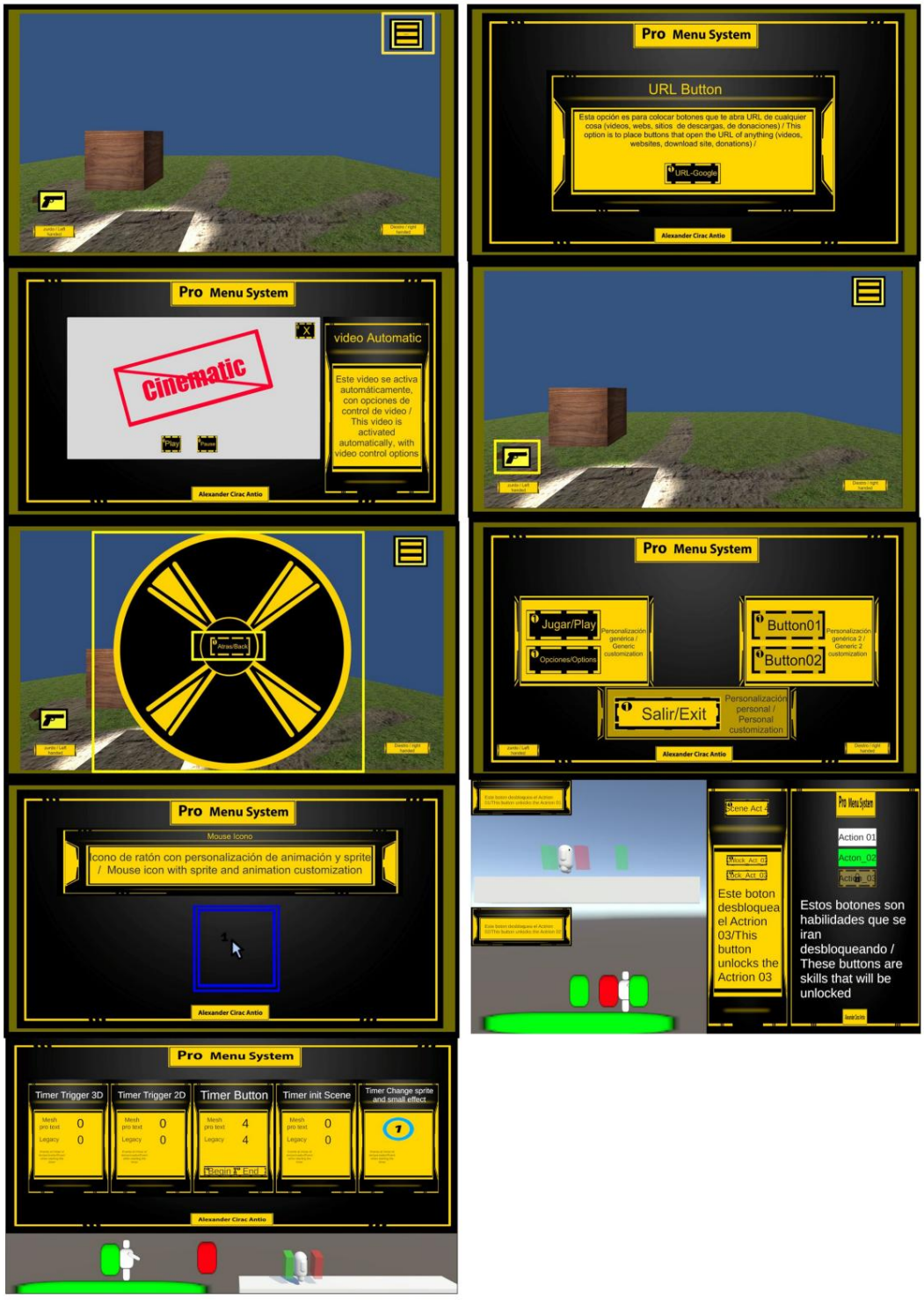
1) Explanation: It will explain in a general way everything you can do in this script, divided into <Explanation> <Use> <Warning>
2) Content: Here you can give functionality to the component that interests you that adopts said function, each question also has its <Explanation> <Use> <Warning> in case you need more information about it

GENERAL EXPLANATION

This script is designed to be able to create an interface to suit the user, providing utilities to the needs to make it more complete and useful. The options that this section allows are the following:

-Option_0: Add a Quit game option
-Option_1: Add an option to load a level
-Option_2: Assign a pause menu
-Option_3: Assign a button to load a URL link
-Option_4: Show a video in panel or in the scene
-Option_5: Assign a pause menu
-Option_6: Control which panels activate or deactivate according to the button
-Option_7: Customize your button with different images/animations depending on the action
-Option_8: Create an icon for the mouse
-Option_9: Create button that locks/unlocks
-Option_10: Create a Timer

Each section will be indicated with a question so that the user himself, if he wants this level, to contain said function and assign the component of his choice



USE

Each question has a dropdown that contains an explanation of what it does and also its content, which will be to place and drag the desired objects in the gaps to give it functionality.

You can fill in those that interest you and in the event that you are missing an important parameter, the system will notify you.
I repeat, there are a total of 5 options to customize the UI, in the following sections they will be explained one by one in more detail about these, both what you can do and what you cannot do

-Option_0: Add a Quit game option
-Option_1: Add an option to load a level
-Option_2: Assign a pause menu
-Option_3: Assign a button to load a URL link
-Option_4: Show a video in panel or in the scene
-Option_5: Assign a pause menu
-Option_6: Control which panels activate or deactivate according to the button
-Option_7: Customize your button with different images/animations depending on the action
-Option_8: Create an icon for the mouse
-Option_9: Create button that locks/unlocks
-Option_10: Create a Timer

WARNING

Remember that the text content is the graphic part; it is not created by the script. It just makes handling easier. Eg: the control of the panels when it comes to showing and which panels to hide, not referring to the graphic part or content, it does not create them, but rather you can place the graphic and the control of the global/container

OPTION 0

It is to give the button the functionality to exit the game with the option to open a panel first (where to place a question and its answers), which will give the user the option to accept said action and/or enable a quick action to exit the game. game

It has eight parts: -

Explanation: The documentation of this option is summarized and concise.

- Content: It will contain all the elements you need to be able to use it.

- Quit button: for the action to quit the game -

Quit question: What will come out before to ask the user if he really wants to quit the game

-Question Panel: The panel where it will contain the question and the answers

-Show Panel Button: The button that will open the panel function

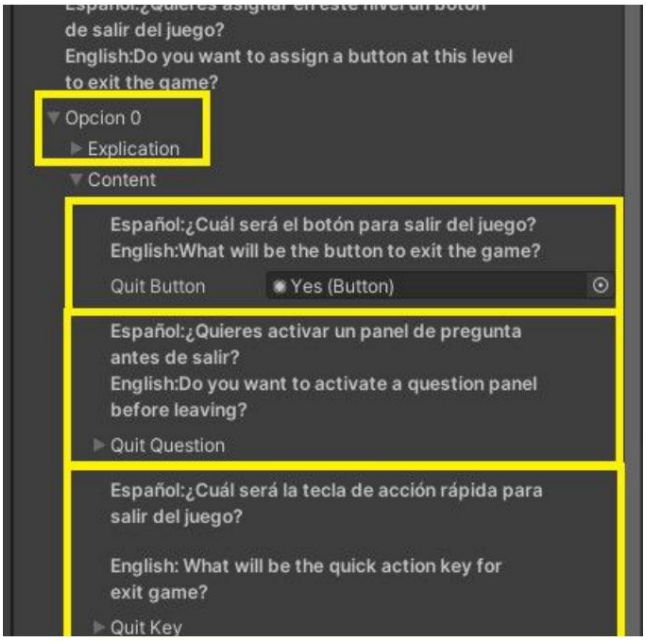
-Hidden Panel Button: The button that will disable the question panel

-Quit Key: you can enable a quick option for both the joystick and the keyboard or a custom button

-Joystick Button: You can indicate a quick access to a joystick button

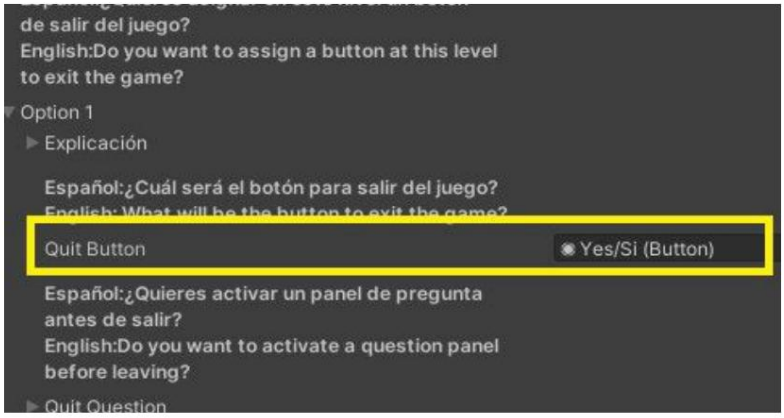
-key Button: You can indicate a quick keyboard access

-Own Button: You can indicate a quick access by placing the name of your personalized key



USE

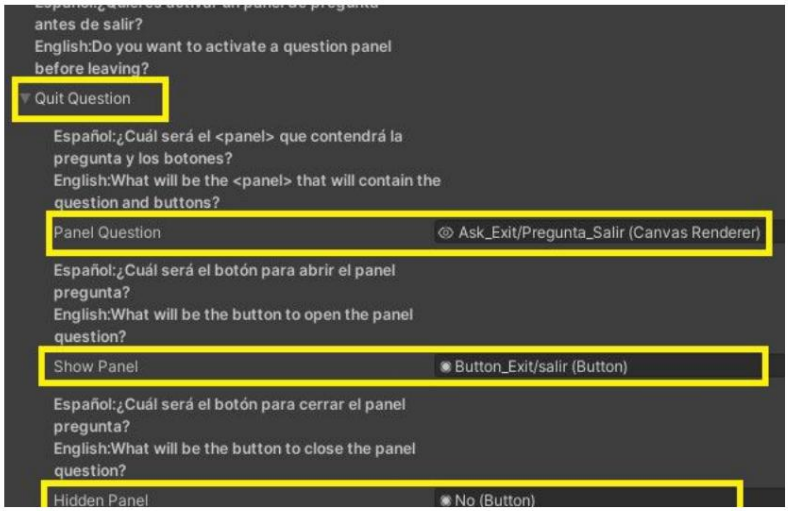
1) Basic Use: To use this option in the most basic way, is to drag the button(*) that you want to do this function to the slot < Quit Button>



2) Ask before exiting: If you are interested in asking a question before exiting the game, I will indicate the next steps. First drop down the <Quit Question> option and fill in the following slots.

-The Show Panel: Drag the button that will open the question panel, which will contain the answer and said question, normally it will be for the button that puts <Exit>
-The Hidden Panel: Drag the button to close the question panel, normally it will be for the answer “No”
-Quit Button: Drag the button that will close the game, normally it will be for the answer "Yes"

-Panel question: You will drag the panel of the Canvas that will contain the question and its answers

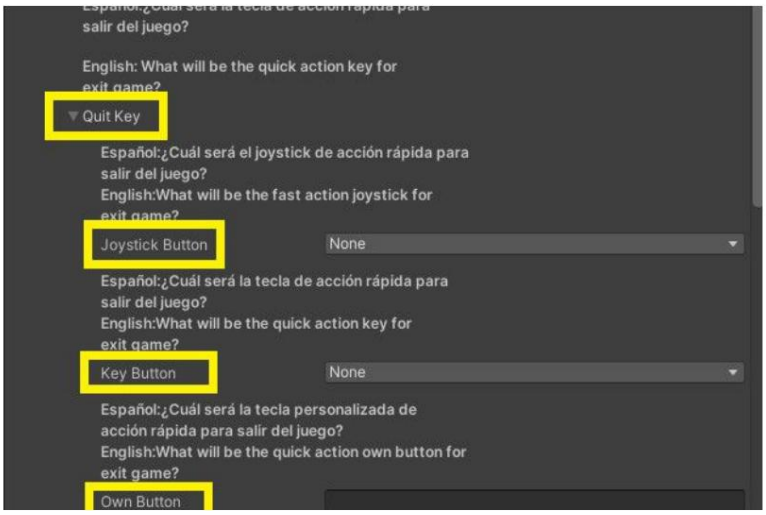


3) You want quick access to exit: If you want to place a quick access button, you have 3 modes at your disposal, with a PC keyboard key, with joystick control for i/o pc consoles or a custom button, the 3 options can be enabled if you want your game to be on both PC and consoles or if you want your PC game to be controller controlled

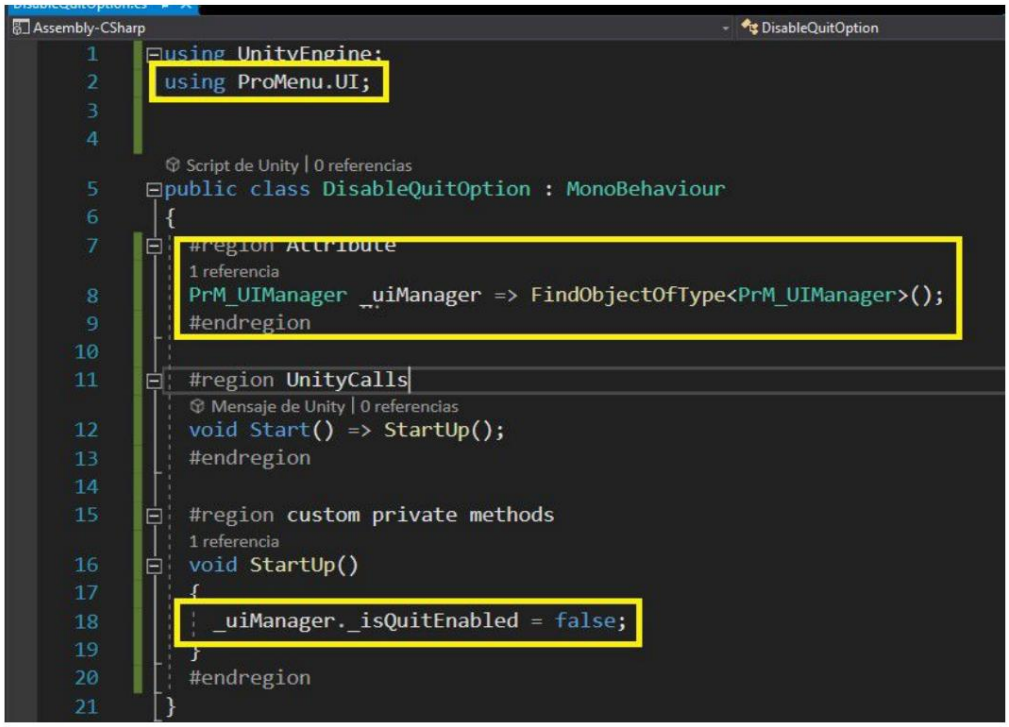
-Joystick Button: Fast action for the console controller (PS3 or Xbox)

-Key Button: Quick action for any key on the keyboard

-Own Button: Quick action for custom buttons, you will have to write the button that you have previously created in Edit > Project Settings > Input Manager > Axis > the name of your button



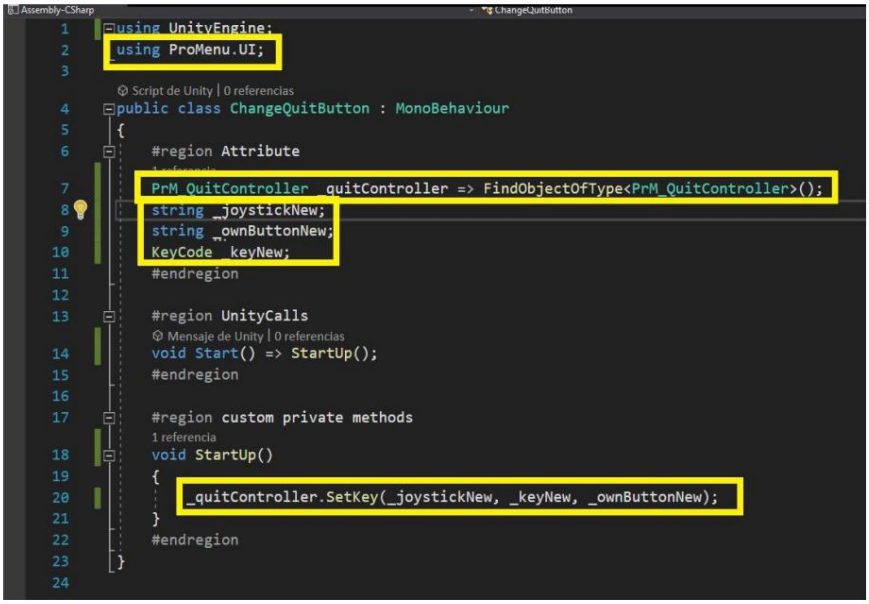
4Momentarily disable quick access: If you wanted to disable the option of being able to use quick access because there is a cinematic and you don't want it to be enabled, you just have to refer to the "PrM_UIManager" script and set the "_isQuitEnabled" bool to true



"

WARNING

If you want to control or change the previous configuration of the buttons, with your own scripts, you can do it, the only thing that if you want to save those changes and preserve it will have to be done by yourself
1) For that, you have to refer to the “QuitController” script when it has been created
2)Access your Setkeys(string _joystick, KeyCode _key,string _ownButton)



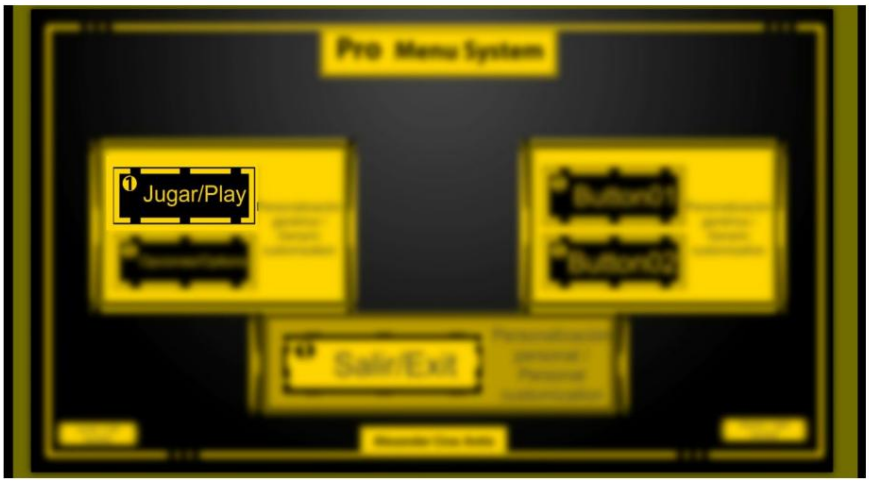
If you try to use the same button to perform the “Show Panel” function and it “Hidden Panel” and/or “Quit button” will be will notify you of the error and the system will not execute the “hidden panel” or “Show Panel” function but will prioritize the basic function, which is to exit the game. However, this system does not add the buttons or the questions, you will have to do that

LEGEND

(*) Button : is a component that is used to create an interactive element in a game scene or in a user interface. A button can contain text, an image, or other visual content and can be clicked to perform an action that affects the game or another interface

OPTION 01

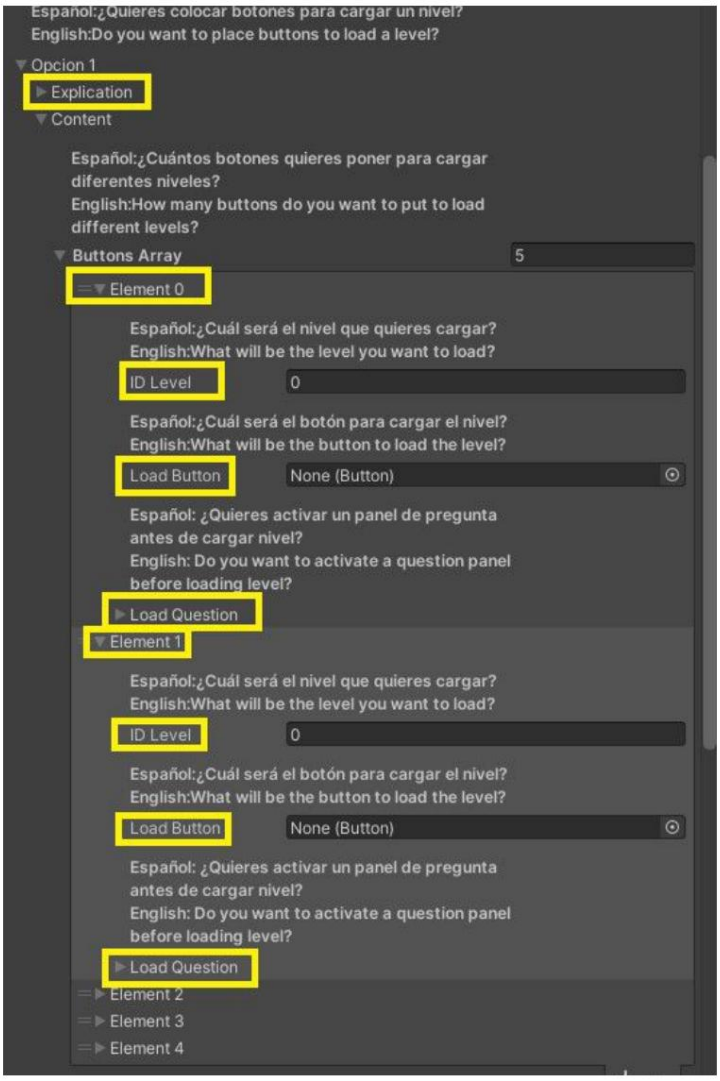
It is to give the button the functionality to change the scene and with the option to open a panel first (where to place a question and its answers), which will give the user the option to accept said action. But in this case, you can create as many buttons to load different scenes as you want.



It has eight parts: -

Explanation: The documentation of this option is summarized and concise.

- Content: It will contain all the elements you need to be able to use it.
  - -Button Array: You can create as many buttons to load different levels
    - -Name Button: Place a name to better identify this element
    - -ID Level To whom you pass the desired scene that you want to load
    - -Load Button: The button that will be in charge of loading the scene
    - -Load Question: create the option to ask the player if he wants to load X scene
      - -Panel Question: the panel where the answers and the question will be contained
      - -Show Button: the button that will open the panel
      - -Hidden Button: the button that will close the panel

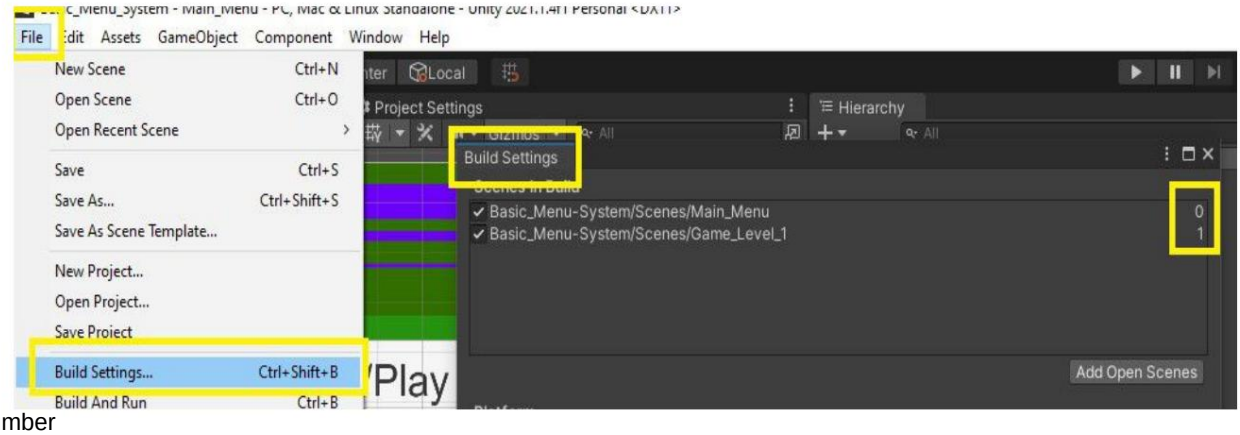


USE

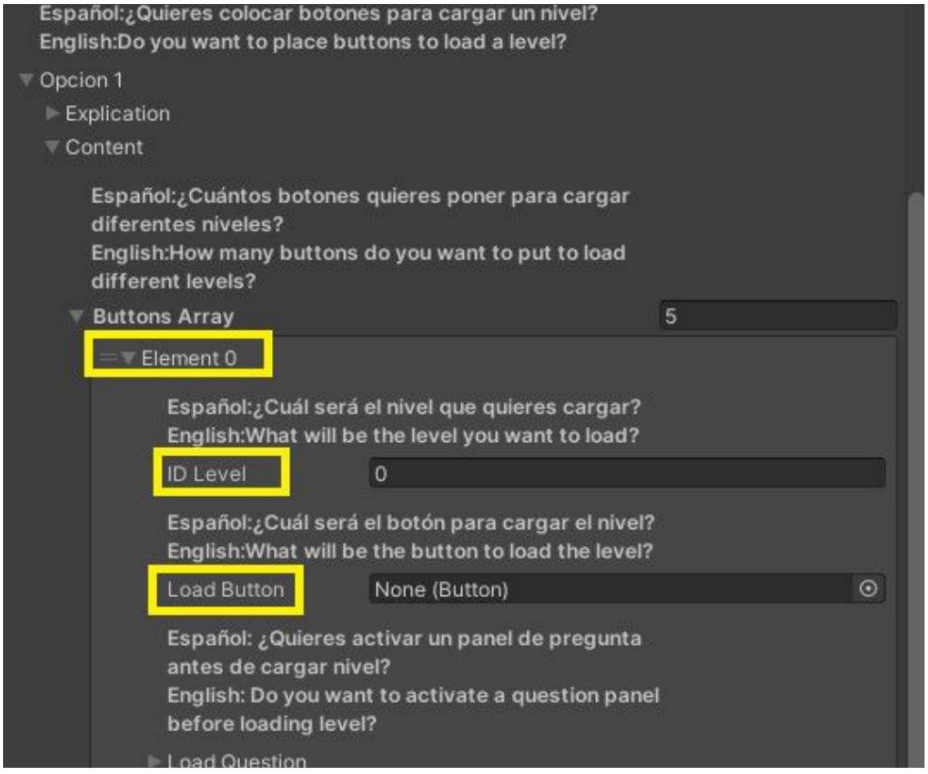
1) Basic Use: To use this option in the most basic way, to load the scene do the following steps

-Load Button: drag the Button you want to load the scene

-ID Level : You will assign with an enumeration the id of the scene that you want to load, to find the enumeration of your scene you will go to file> build Settings> of the scenes look at the



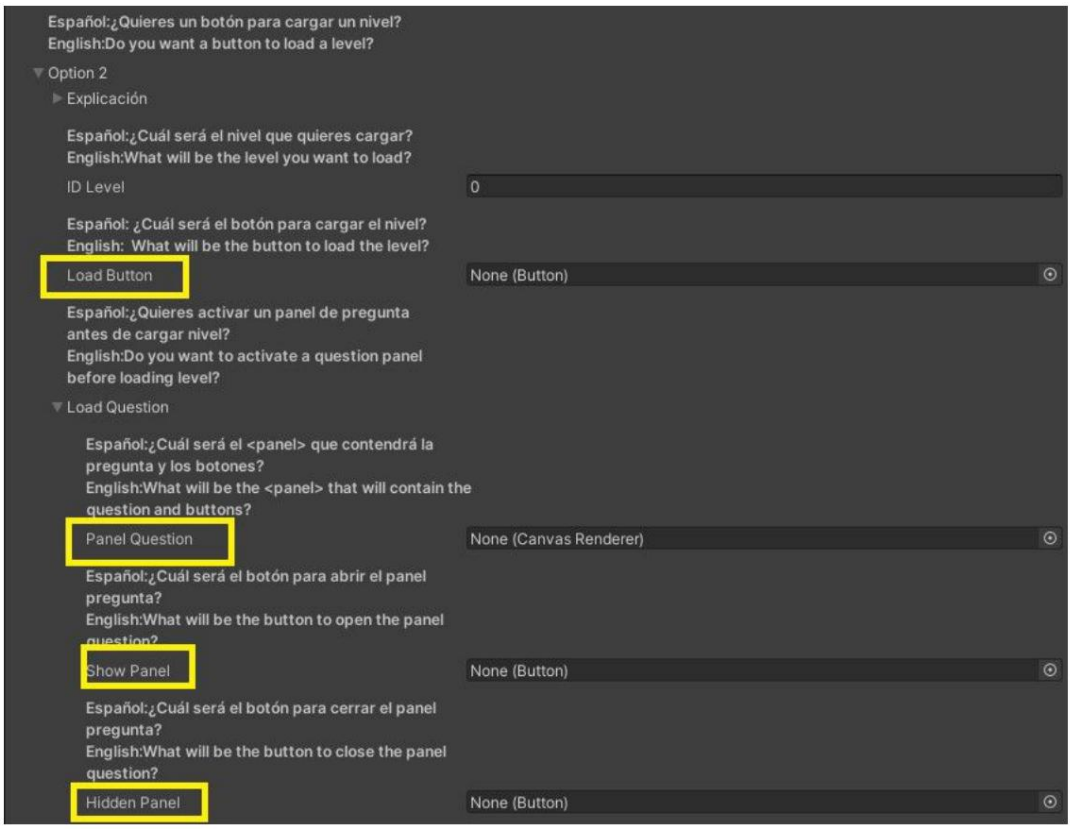
number



2) Ask before loading scene: If you are interested in asking a question before exiting the game, I will indicate which are the next steps. First display the <Quit Question> option and fill in the following slots.

-The Show Panel: Drag the button that will open the question panel, which will contain the answer and said question, normally it will be for the button that puts <Exit>
-The Hidden Panel: Drag the button to close the question panel, normally it will be for the answer “No”
-Quit Button: Drag the button that will close the game, normally it will be for the answer "Yes"

-Panel question: You will drag the panel of the Canvas that will contain the question and its answers



WARNING

If you try to use the same button to perform the “Show Panel” function, it will notify you of “ Hidden Panel” and/or “Quit button” will be the error and the system will not execute the “hidden panel” or “Show Panel” function but will prioritize the basic function, which is to exit. of the game. However, this system does not add the buttons or the questions, that will have to be done by you.

If in a situation you place more buttons in the list, but in one of them it does not have any content, the system will also notify you that in the list of buttons, there is an element that has not been assigned any component, this will not affect the operation of the system only give even notice

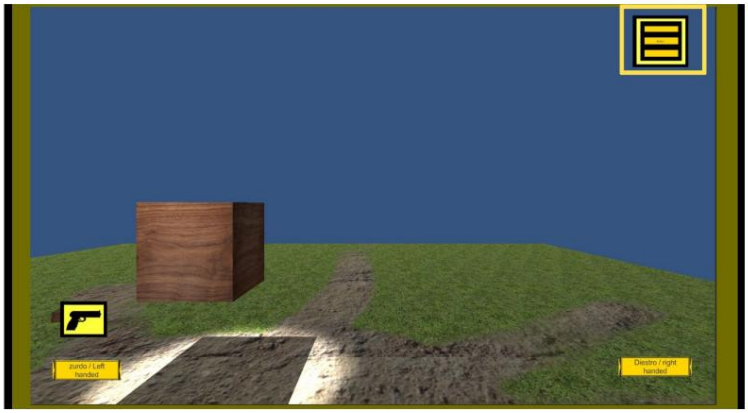
If in a situation you place more buttons in the list, but in one of them it does not have any content, the system will also notify you that in the list of buttons, there is an element that has not been assigned any component, this will not affect the operation of the system only give even notice

LEGEND

(*) Button : is a component that is used to create an interactive element in a game scene or in a user interface. A button can contain text, an image, or other visual content and can be clicked to perform an action that affects the game or another interface

OPTION 02

It is to give the button the functionality of pausing the game and for a drop-down menu to appear, which is the pause menu. In this mode you can choose whether your pause menu stops the game or not, and you can also indicate if you want to have quick access to open said pause menu

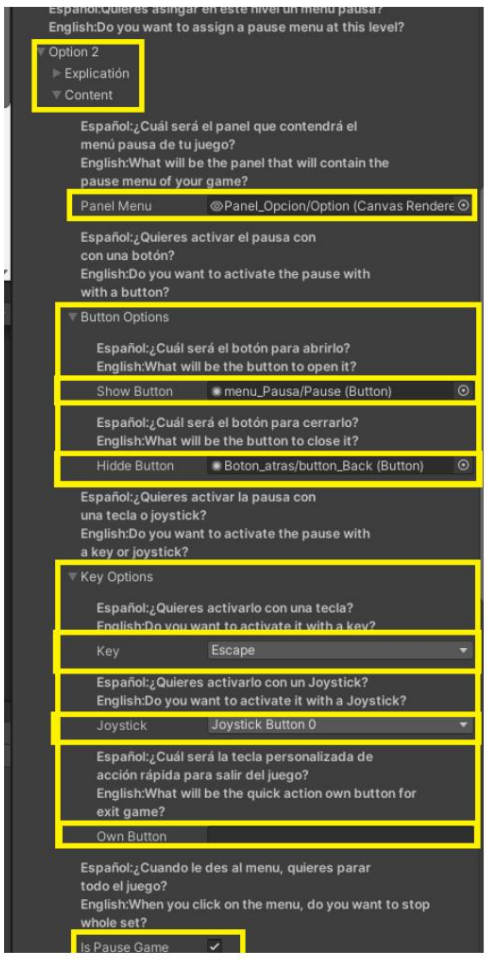


It has eight parts: -

Explanation: The documentation of this option is summarized and concise.

- Content: It will contain all the elements you need to be able to use it.

-Panel Menu: You will indicate which will be the Pause Panel(*) -Button Options: You will indicate which are the buttons that will open the pause panel -Show Button(**): the button that will activate the pause menu -Hidden Button: the button that will deactivate the pause menu -Key Options: The quick access buttons to open the pause panel -Key: What will be the keyboard key -Joystick: What will be the button of the controller (PS3 and Xbox) -Own Button: What will be the custom button



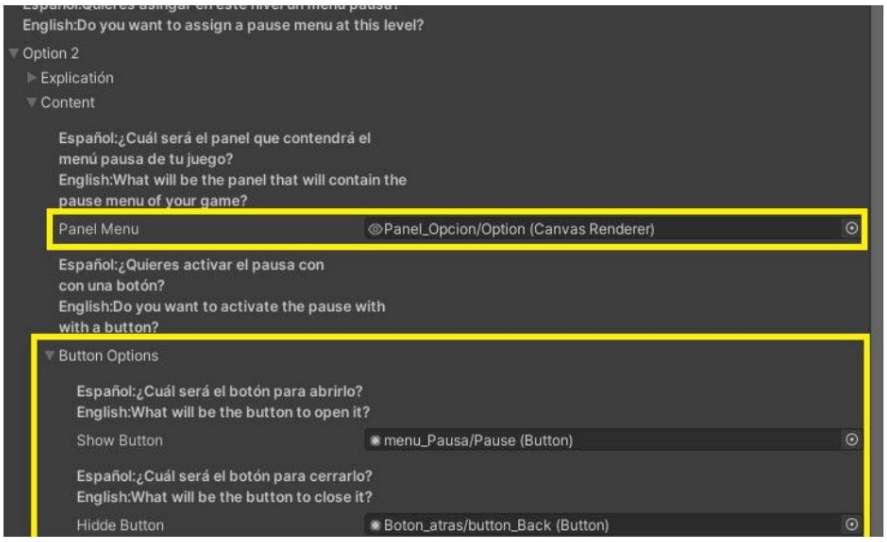
USE

1) Basic Use: To use this option in the most basic way, to control the pause menu you only have to indicate which buttons activate and deactivate

-Panel Menu: Drag the Panel that will contain the pause menu

-The Show Button: It will be to open the pause menu

-The Hidden Button: it will be to close the pause menu

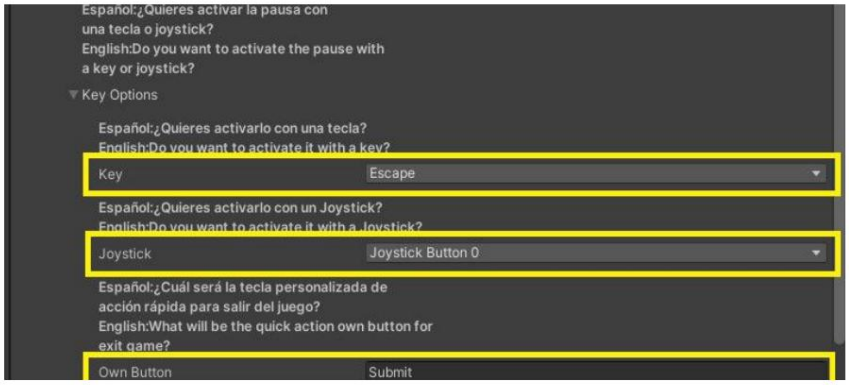


2) You want a quick access: If you want to place a quick access button, you have at your disposal 3 modes, with a keyboard key for PC, with joystick control for i/o pc consoles or a custom button, all 3 options can be enabled if you want your game to be on both PC and consoles or if you want your game to be of PC can be controlled by command

-Joystick : Fast action for console command (PS3 or Xbox)

-Key : Fast action for any key on the keyboard

-Own Button: Quick action for custom buttons, you will have to write the button that you have previously created in Edit > Project Settings > Input Manager > Axis > the name of your button

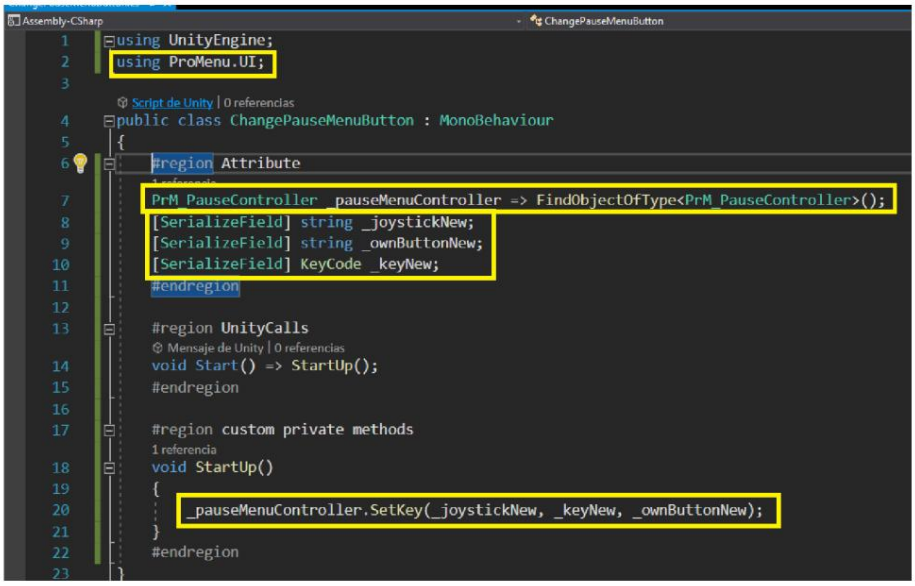


3) Pause the game: If you want the game to stop, when the user activates the pause menu, you can enable this option, selecting the <IsPauseGame>

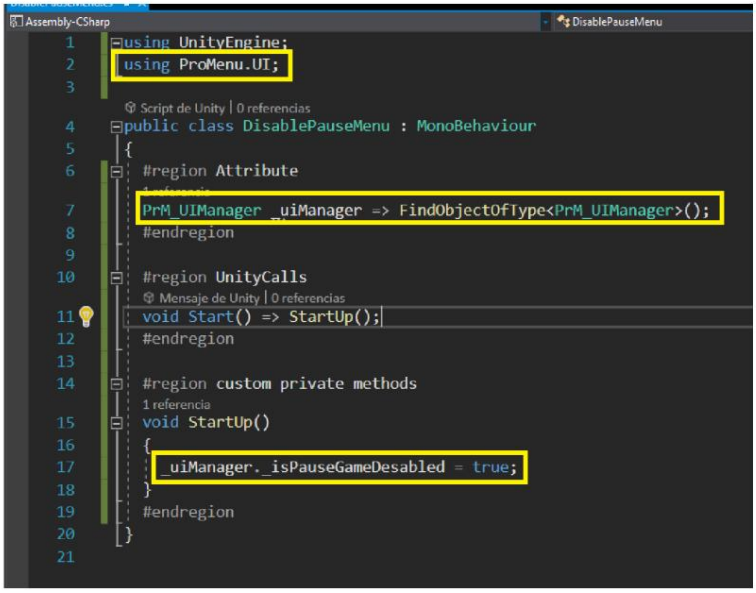
WARNING

If you want to control or change the previous configuration of the buttons, with your own scripts, you can do it, the only thing that if you want to save those changes and preserve it will have to be done by yourself

1) For that, you have to refer to the script “PauseGameController” when it has been created
2)Access your Setkeys(string _joystick, KeyCode _key,string _ownButton)



If you wanted to disable the option of being able to use the shortcuts because there is a kinematics and you don't want it to be enabled, you just have to reference the “PrM_UIManager” script and set the “_isPauseGameDesabled” bool to true.



If you try to use the same button to perform the “Show Panel” function and the “Hidden Panel” and/or “Quit button”, the error will be notified and the system will not execute the “hidden panel” or “Show Panel” function but if it prioritized the basic function, which is to pause the game. However, this system does not add the buttons or the content, you will have to do that

LEGEND

(*) Panel : a component used to group and organize other elements in a game scene or user interface. The panel can be used to create containers to hold other elements, such as text, images, buttons, and other panels, and panel properties can be adjusted to set the position, size, and appearance of the contained elements.

(**) Button : is a component that is used to create an interactive element in a game scene or in a user interface. A button can contain text, an image, or other visual content and can be clicked to perform an action that affects the game or another interface

OPTION 03

It is to give the functionality to the button, to open web page links, YouTube or social networks. The internet that you have set as default in your system will always be open. You can create these buttons as many as you want

It has eight parts: -

Explanation: The documentation of this option is summarized and concise.

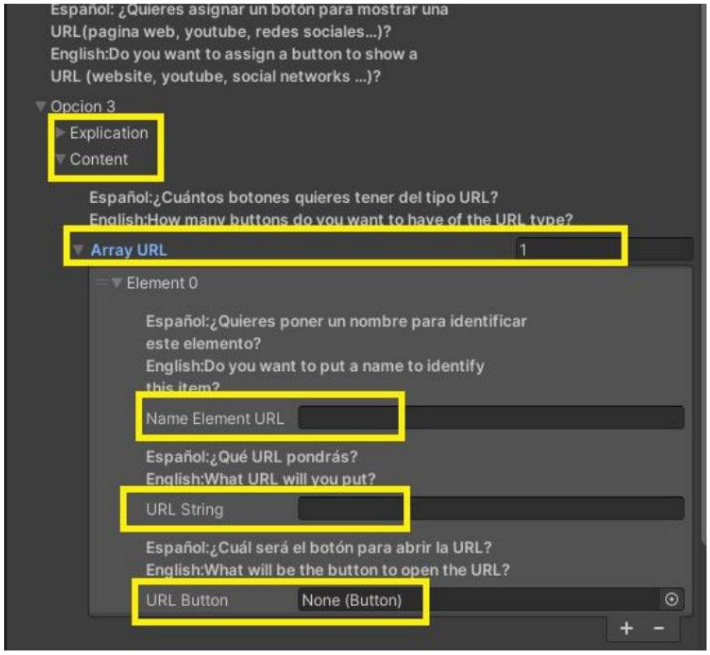
- Content: It will contain all the elements you need to be able to use it.

-Array URL: You can create as many buttons as you want to open different links

-Name Element URL: A name to identify the element

-URL String: the link of the link you want to open

-URL Button(*): the button that will open the link

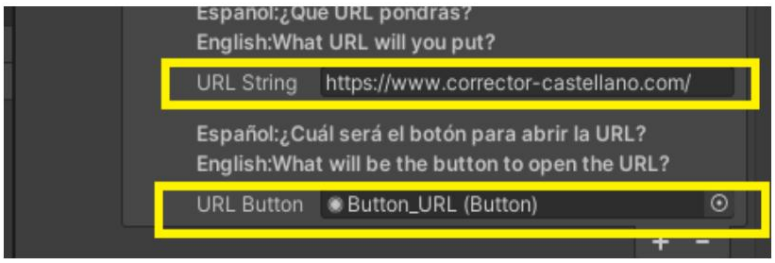


USE

1) Basic Use: To use this option in the most basic way, to redirect you to a web page or a video of YouTube or any link, just fill in the following parameters.

-URL String: Paste the desired link

-URL Button: Drag the button that you want this function to have and open said link



WARNING

The system does not design or scale or write the content, it only applies the designated function
If in a situation you place more buttons in the list, but in one of them it does not have any content, the system will also notify you that in the list of buttons, there is an element that has not been assigned any component, this will not affect the operation of the system only give even notice

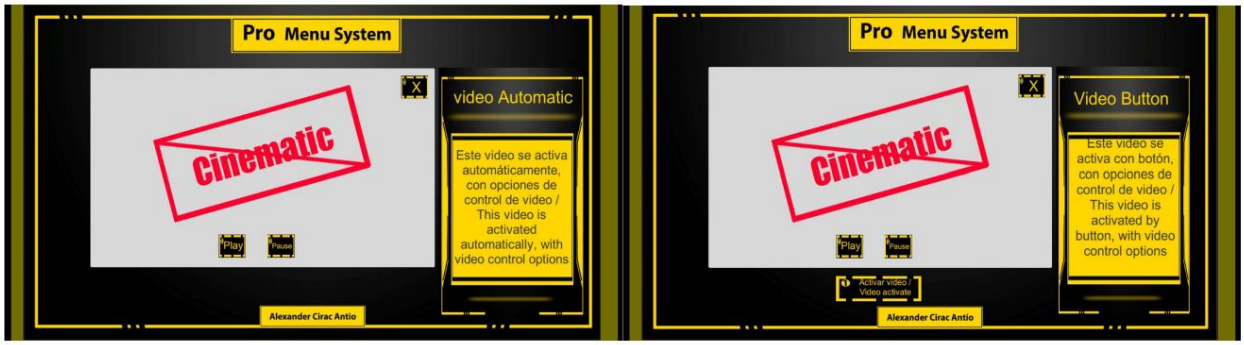
LEGEND

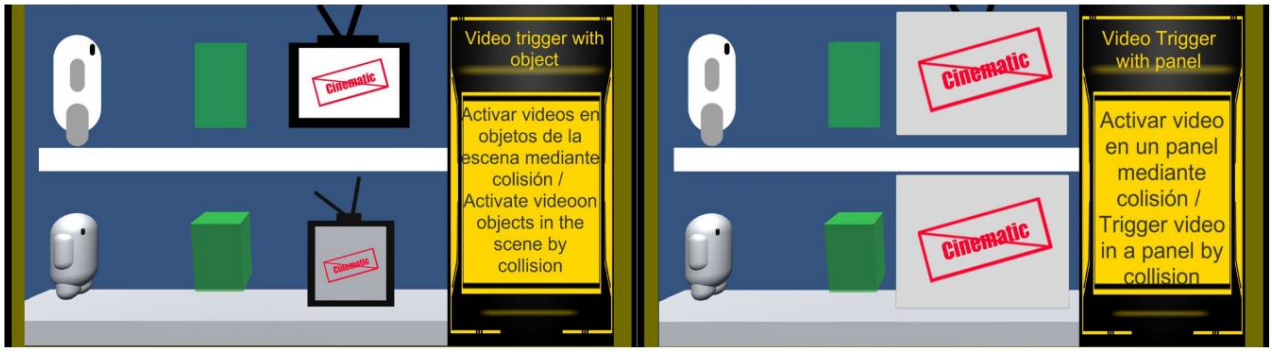
(*) Button : is a component that is used to create an interactive element in a game scene or in a user interface. A button can contain text, an image, or other visual content and can be clicked to perform an action that affects the game or another interface

OPTION 04

It is to be able to place video clips, inside the game or in the interface, in this way it gives you the option to show it in <canvas> panels, as if it were a cinematic, but you can also place those videos, in 2D and 3D objects , in this way simulating the content of a television channel or in what your imagination allows you. You will also have to select how you want these videos to be played, whether automatically or by means of a button or by means of a trigger in the 2D or 3D world of the game. And in turn, you will also indicate how you want it to behave once the video is finished, if you want it to be deactivated or frozen or repeated again.

You can also control the playback of the video: pausing, playing or removing





-

You have eighteen parts: -

Explanation: The documentation of this option is summarized and concise.

- Content: It will contain all the elements you need to be able to use it.

- Array Video: You can create as many video/cinematic elements in the scene as necessary -Element name: You can put the name to identify this video -Clip(*): You will put which video you want to place to be displayed -How see: Here it is to indicate where you will show this video

-panel: Show the video in a panel(**), of the <Canvas>(***)

-2D: Play the video inside a 2D object

-3D: play the video in a 3D model

-Activate mode : You will indicate how you want the video to be activated

-Automatic mode: It is to say that it plays automatically

-Button(****) mode: It is to indicate that it is played when the button is clicked

-Trigger 2D: It is to indicate that it is played when the player passes through a 2D object

-Trigger 3D (*****): It is to indicate that it is played when the player passes through a 3D object

-Name Tag(******): Here you will indicate what will be the tag that will have to be identified when it collides with the trigger

-Controller Video: ways to control video playback

-Pause button: Here you can place the button that will stop the video

-Play button: Here you can place the button that will play the video if it is stopped

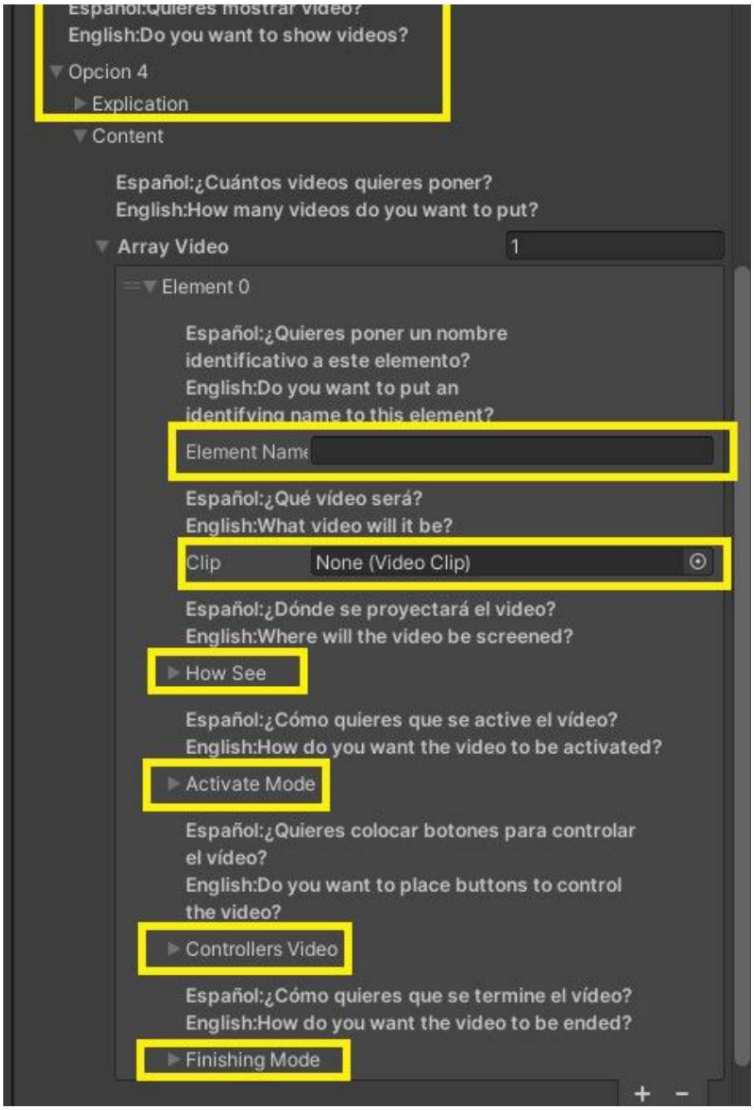
-Quit button: Here you can place a button to remove the video

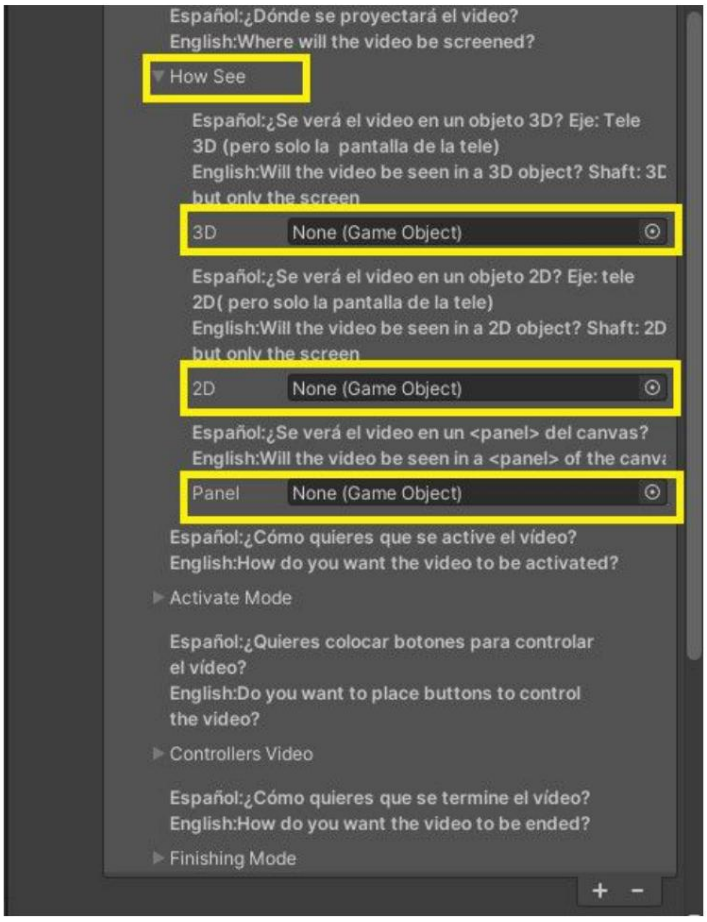
-Finishing Mode: Here you will indicate how the video will behave when it ends

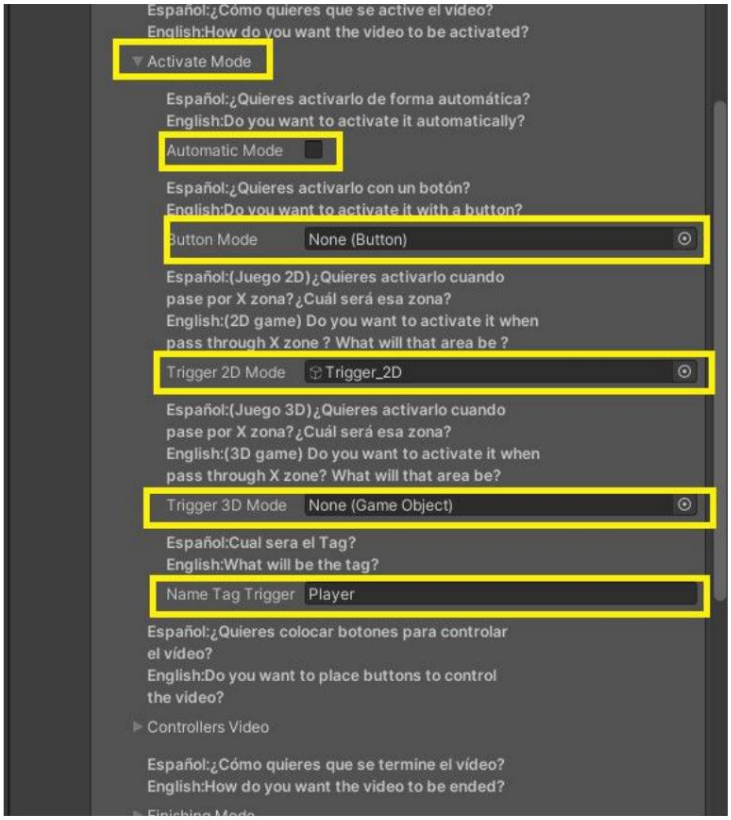
-Stop video: Deactivates the video, but without deactivating the object that emits the video

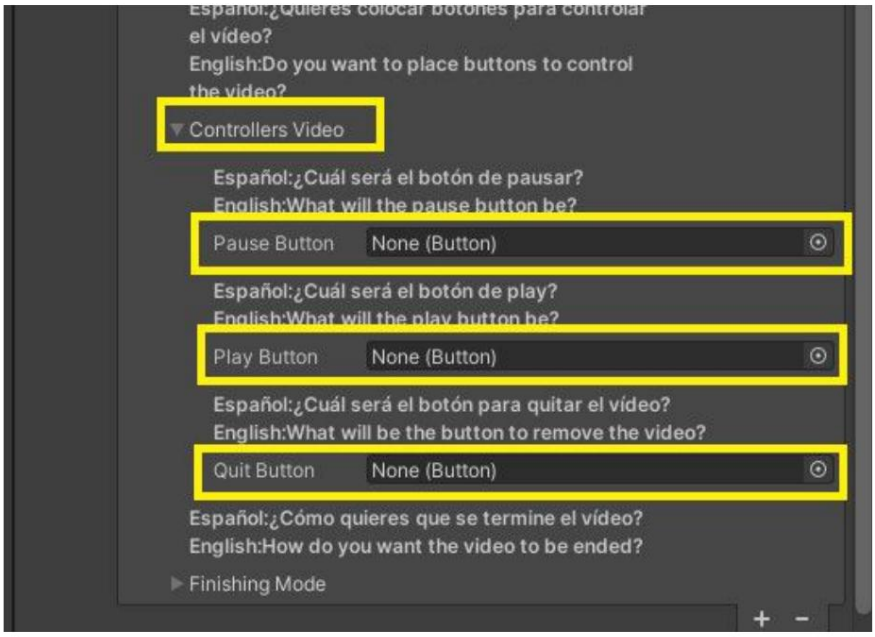
- Stop all: Deactivates the video and the object that emits it

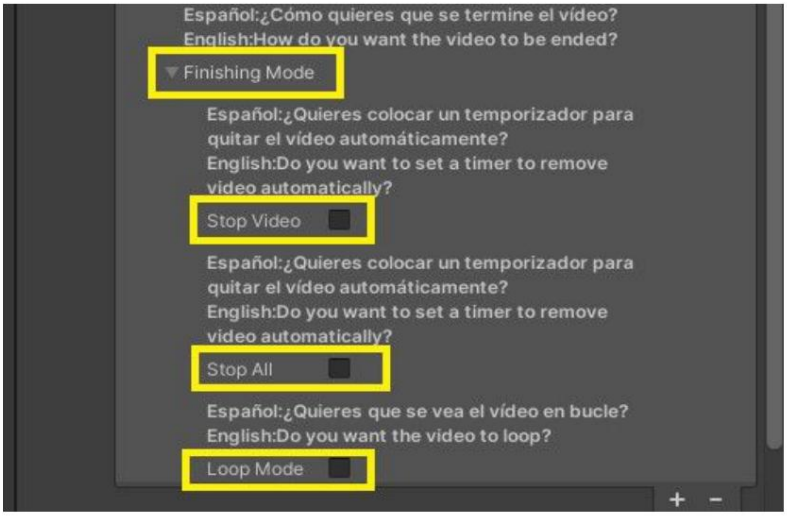
- Loop Mode: Makes the video repeat all the time











USE

1) Basic Use: To use this option in the most basic way, to have a video inside the game or a cinematic you only have to fill in the following parameters.

-Clip: drag the video you want in the slot, the video can be in the following formats (MP3, MP4, AVI, MOV)

-How See: in this dropdown you will choose 1 option where the video will be played, as if it were the screen where the video is viewed

-2D: if you want in a 2D object,

-3D: In the mesh of a 3D object

-Panel: In a Panel element of the interface

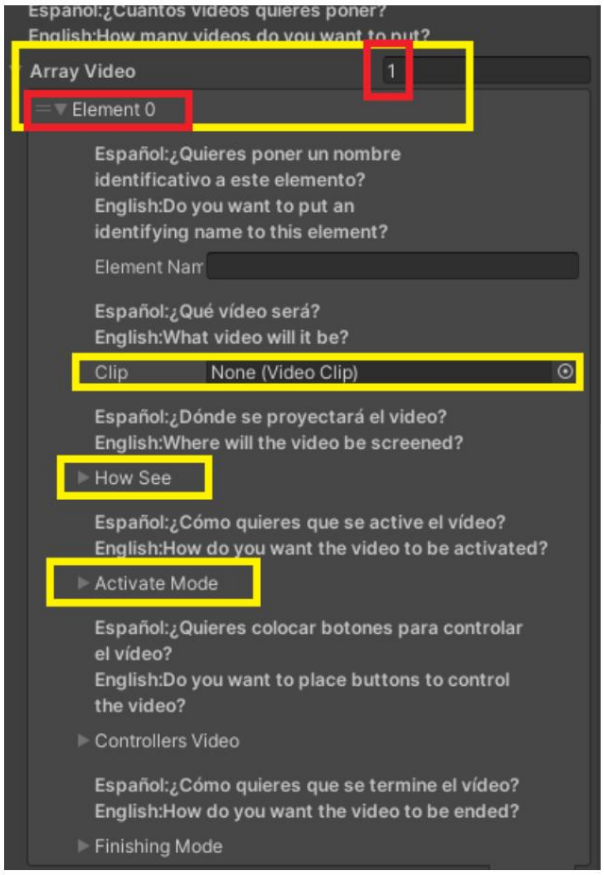
-Activate Mode: Indicate 1 of the options offered in this drop-down menu as to how to activate said video, the one that for the screen must be activated, if it is deactivated it will not be able to be seen even if you indicate different ways to activate it

-Automatic: will play automatically

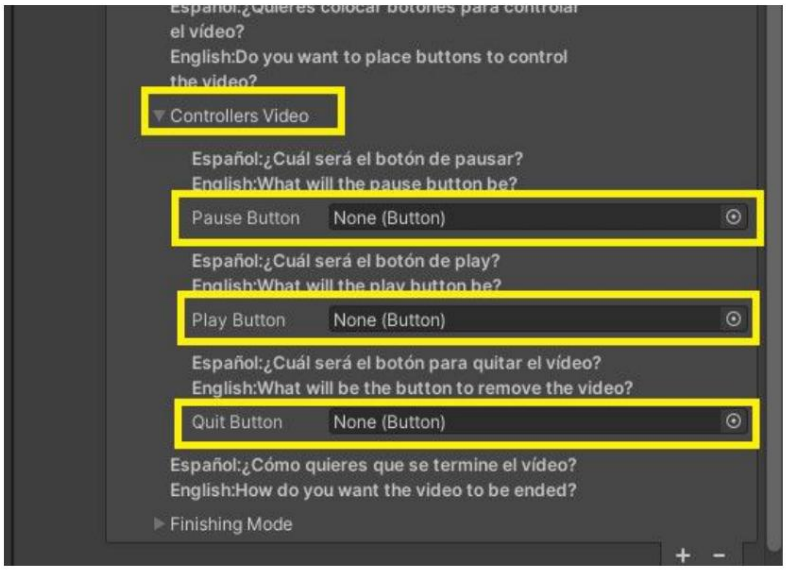
-3D Trigger: When an object passes through the 3D trigger

-Trigger 2D: When an object passes through the 2D trigger

-Name Tag: Indicate the name of the tag that will interact with the triggers

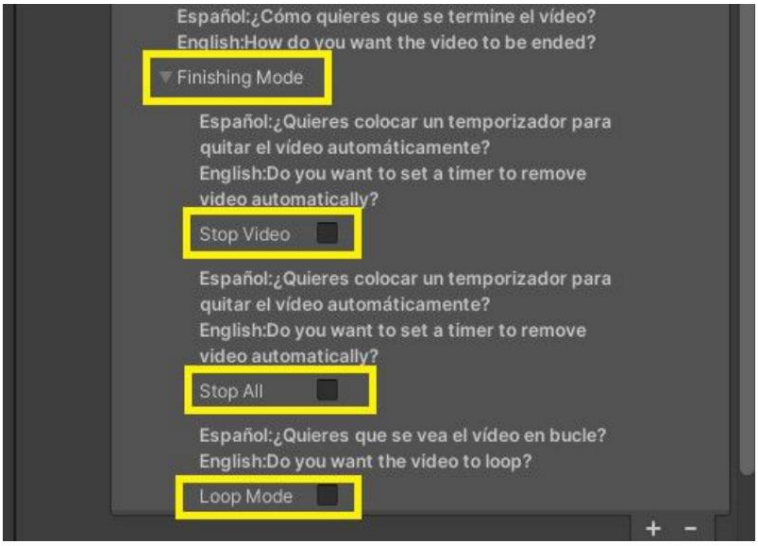


2) You want to place controls for the video: If you want the video to have some controls as if it were a video player, where you have a button to pause, to press Play or you are only interested in removing it in the traditional way with a button, you only have to pass those buttons that you want to do that functionality in "Controllers videos"



3) Behaviors at the end of the video: To be able to indicate what type of behavior you want the video to adopt when it ends, you just have to select the box you want or pass said object that contains the video, it can only be one option per video

-Stop video: Deactivates the video, but without deactivating the object that emits the video -Stop all: Deactivates the video and the object that emits it 3.Loop Mode: Makes the video repeat all the time -Loop Mode: Always is playing to infinity



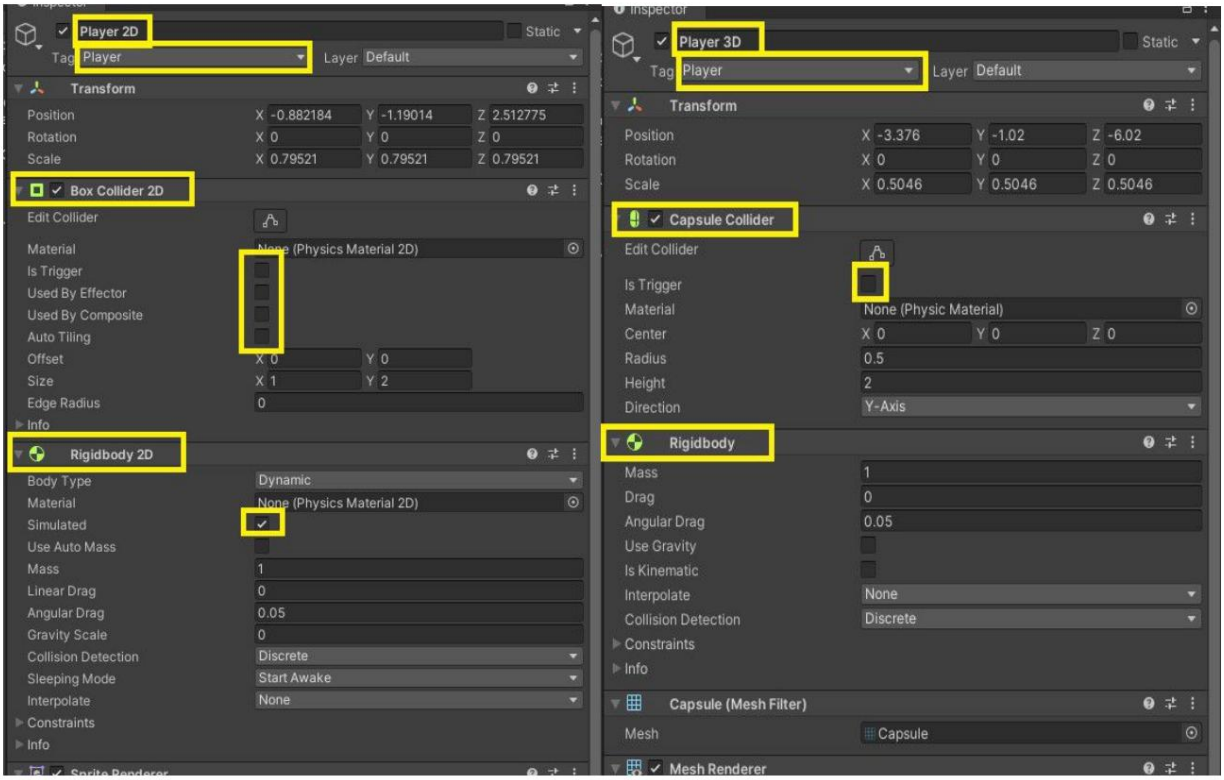
WARNING

In each dropdown, if you have more than one option selected, or if one of the video list does not have any element and the following, if it has or is missing to fill in some components that are important for the system, it will warn of all errors and will not be will perform the function of that option.

For ***WebGL*** you have to do new steps,

1) create a folder named StreamingAssets in the project and it has to be at the same level of hierarchy than the Assets folder
2) Place the clip you want to play in that folder (in that place you can't pass the clip file to the system, you must have 2, one to put it and another for the folder
3) the video in this case must be mp4, but if it is not for WebGL, any format is fine

It is important that the objects that will be the 3D or 2D triggers have these elements



LEGEND

(*) Clip : is a component that is used to create an interactive element in a game scene or in a user interface. A button can contain text, an image, or other visual content and can be clicked to perform an action that affects the game or another interface

(**) **Panel :** it is a component that is used to group and organize other elements in a scene of game or in a user interface. The panel can be used to create containers to hold other elements, such as text, images, buttons, and other panels, and panel properties can be adjusted to set the position, size, and appearance of the contained elements.

(***) **Canvas :** is a component used to create a user interface (UI) in a game scene or in an app. The Canvas acts as a container for user interface elements such as text, images, buttons, and other UI components, and provides a coordinate system that is used to position and scale the elements on the screen.

(****) **Button :** is a component used to create an interactive element in a game scene or user interface. A button can have text, an image or other type of visual content and can be clicked to perform an action that affects the game or another interface (*****) Trigger : is a component **that** can be added to an object in a game scene to detect when another object collides with it or is within a specified area. Triggers are used to create in-game events, such as triggering an animation, playing a sound, or starting a dialogue sequence, when a collision is detected or an object enters the trigger area.

(******) **Tag :** is an identifier used to classify and group objects in a game scene. Tags are used to quickly identify objects in the scene and to make them easier to select and manipulate in the Unity editor or in scripts.

OPTION 05

This option is to control, if you want to show or hide panels or menus, depending on what you are interested in through the interaction of buttons or keyboard shortcuts, to create a series of drop-down menus in the interface in which you show one interface and hide others according to your designs. For example, you can make an interface of a weapon wheel that is activated by a button or by a keyboard shortcut or joystick.

It has ten parts: -

Explanation: The documentation of this option is summarized and concise.

- Content: It will contain all the elements you need to be able to use it.

-Panel: You can create as many panel activator/deactivator controllers(*)

-Name Element: A way to identify the element -oldPanel and

newPanel : You will drag the panels that you want to be the panels of the oldPanel and those of the newPanel

-New Button(**): The button to give the functionality of hiding the panels of the oldPanel and show the newPanel -Back

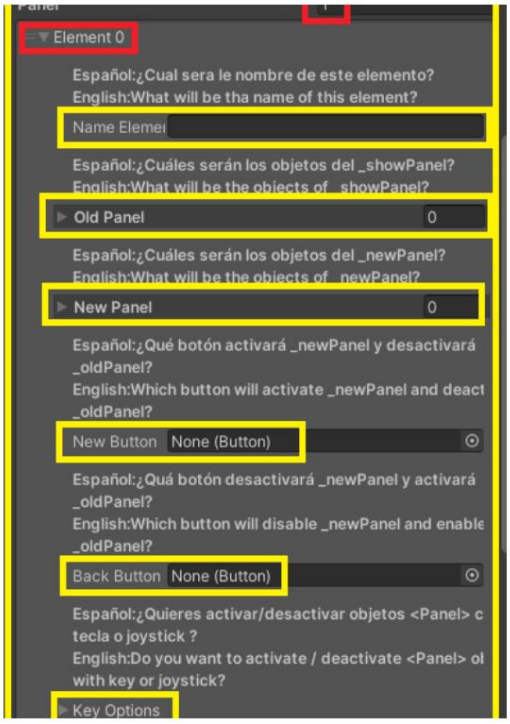
Button: The button to give the functionality to show the oldPanel panels and hide the newPanel Key Options: Here

you can indicate if you want to control the menu with any i/o joystick key.

-Key Button: To open/close it using a keyboard key

-Joystick: Through a button on the hand controller (PS3, Xbox)

-Own Button: Using a custom button



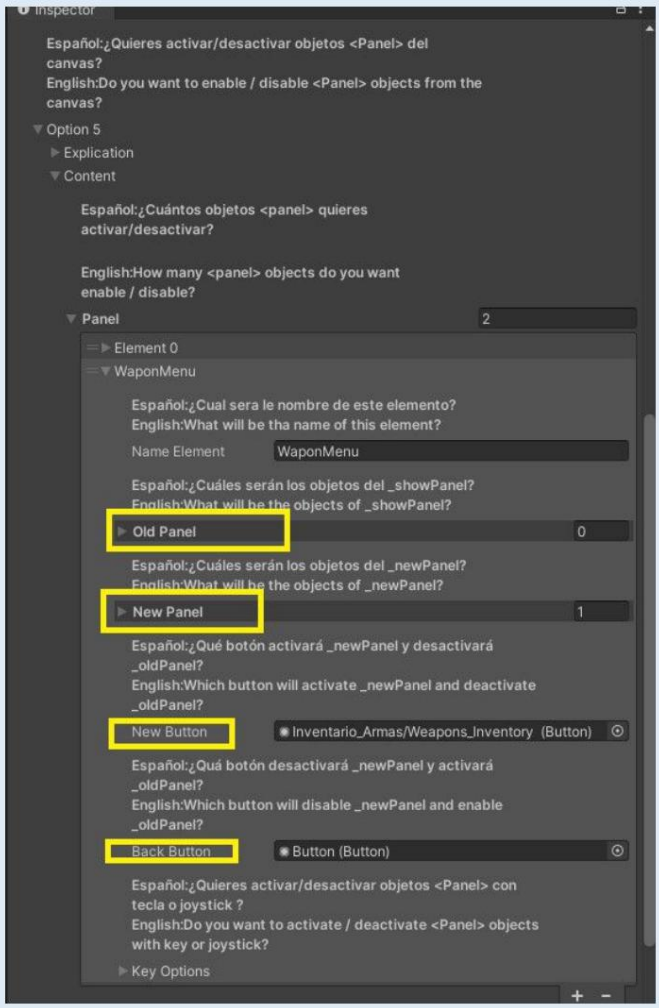
USE

1) Basic Use: To use this option in the most basic way, to create the controllers enable/disable menus; options, weapons or inventory

-Old Panel: You will pass the panels that you want when interacting with the button and deactivate them

-New Panel You will pass the panels that you want when interacting with the button and activate them -new Button: Drag the button that you want to deactivate the <Old Panel> and activate <New Dashboard>

-Back Button: Drag the button that you want to activate the <Old Panel> and deactivate the <New Panel>



2) Quick access button: you have at your disposal 3 modes, with a Key of the keyboard for PC, with the joystick control for consoles or a custom button, all 3 options can be enabled if you want your be game to be both for PC and consoles or if you want your PC game to be controlled by controller

-Joystick Button: Quick action for the console control knob

-Key Button: Quick action for any key on the keyboard

-Own Button: Quick action for custom buttons, you will have to write the button that you have previously created in Edit > Project Settings > Input Manager > Axis > the name of your button

WARNING

If you want to control or change the previous configuration of the buttons, with your own scripts, you can do it, the only thing that if you want to save those changes and preserve it will have to be done by yourself

1) For that, you have to refer to the script "PanelController_+iDArray" when it is created

2)Access your Setkeys(string _joystick, KeyCode _key,string _ownButton)

```
Assembly-CSharp                                   ChangePanelButton                    _menuController
1  using UnityEngine;
2  using AlexanderCA.ProMenu.UI;
3
   Script de Unity | 0 referencias
4  public class ChangePanelButton : MonoBehaviour
5  {
6      #region Attribute
       1 referencia
7      PrM_MenuController _menuController => GameObject.Find("PrM_MenuController_" + _arrayElement.ToString()).GetComponent<PrM_MenuController>();
8      [SerializeField] int _arrayElement;
9      [SerializeField] string _joystickNew;
10     [SerializeField] string _ownButtonNew;
11     [SerializeField] KeyCode _keyNew;
12     #endregion
13
14     #region UnityCalls
       Mensaje de Unity | 0 referencias
15     void Start() => StartUp();
16     #endregion
17
18     #region custom private methods
       1 referencia
19     void StartUp()
20     {
21         _menuController.SetPanelsKey(_joystickNew, _keyNew, _ownButtonNew);
22     }
23     #endregion
24
25 }
26
```

If you wanted to disable the option of being able to use the shortcuts because there is a cinematic and you don't want it to be enabled, you just have to refer to the script

"PrM_UIManager" and set the bool "_isPanelEnabled" to true

```
Assembly-CSharp                                   DisablePanelOption
1  using UnityEngine;
2  using AlexanderCA.ProMenu.UI;
3
   Script de Unity | 0 referencias
4  public class DisablePanelOption : MonoBehaviour
5  {
6      #region Attribute
       1 referencia
7      PrM_UIManager _uiManager => FindObjectOfType<PrM_UIManager>();
8      [SerializeField] bool _bool;
9      #endregion
10
11     #region UnityCalls
       Mensaje de Unity | 0 referencias
12     void Start() => StartUp();
13     #endregion
14
15     #region custom private methods
       1 referencia
16     void StartUp()
17     {
18         _uiManager._isPanelDesabled = _bool;
19     }
20     #endregion
21 }
22
```

If you try to use the same button to perform the “Show Panel” function, it will “ Hidden Panel” and/or “Quit button” will be notify you of the error and the system will not execute the “hidden panel” or “Show Panel” function but will prioritize the basic function, which is to exit. of the game. However, this system does not add the buttons or the questions, you will have to do that. If in a situation you place more buttons in the list, but in one of them it does not have any content, the system will also notify you that in the list of buttons, there is an element that has not been assigned any component, this will not affect the operation of the system, it will only give a warning

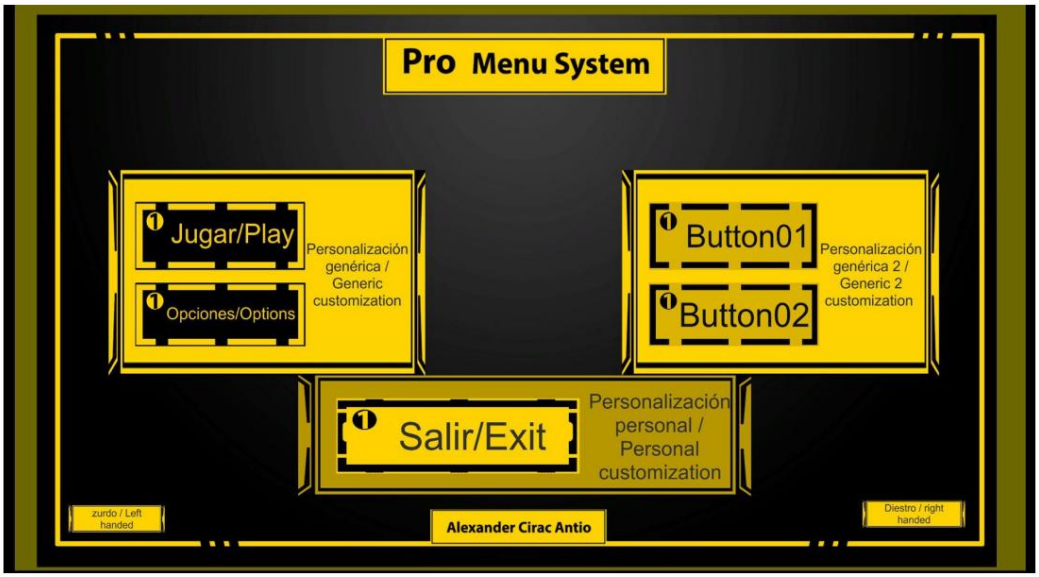
LEGEND

(*) **Panel :** is a component that is used to group and organize other elements in a game scene or in a user interface. The panel can be used to create containers to hold other elements, such as text, images, buttons, and other panels, and panel properties can be adjusted to set the position, size, and appearance of the contained elements.

(**) **Button :** is a component that is used to create an interactive element in a game scene or in a user interface. A button can contain text, an image, or other visual content and can be clicked to perform an action that affects the game or another interface

OPTION 06

In this section you can provide a series of different customizations to the button, so that it adapts an image and/or an animation when there are different interactions with said button, creating a more complex and complete interface, from which you can change different images, animations and sounds according to mouse over, click hold and normal state You can indicate if you want a generic preset (you can create more than one generic) or you can make said a generic customization or you can make said button have its own customization.



You have twenty parts: -

Explanation: The documentation of this option is summarized and concise.

- Content: It will contain all the elements you need to be able to use it.

-Custom Default: You can create as many customizations of a generic nature for the appearance of your buttons

- OnDefoult Imaget: It is the image that will be seen initiallyImage

-OnClick Image(*): It is the image that will be seen when the button is clicked

-OnEnter Image: It is the image that will be seen when the mouse passes over the button without clicking

-Animator Controller(**): It will indicate which will be the controller of the button animation

-OnDefoult Video: What will be the rest animation of the button

-OnEnter Video: What will be the animation when the mouse passes over the button

-OnClick Video: What will be the animation when the button is clicked

-Audio Source(***): You will have to place which will be the audio source where the sound will be applied for the button effect

-OnEnter Sound: The sound that will be emitted when the mouse passes over the button

-OnClick Sound: The sound that will be emitted when the button is clicked

-Loop: You can tell if you want the sound to be emitted in a loop when you do an action with the button

-Custom Buttons(****): Here you will have to indicate how many buttons you want to customize

-Name element: You can put a name to the element to better identify it

-Button: Drag the desired button that you want to customize

-Is Own Custom: This is an option that you will activate if you want the buttons to use those of your own customizations

-Int Generic: Here you will indicate, with an enumeration, which generic from the list you want to choose

– Own Custum: This is an own customization option in case you have not activated the option <Is generic>

- OnDefoult Imaget: It is the image that will be seen initiallyImage

-OnClick Image: It is the image that will be seen when the button is clicked

-OnEnter Image: It is the image that will be seen when the mouse passes over the button without clicking

-Animator Controller: It will indicate which will be the controller of the button animation

-OnDefoult Video: What will be the rest animation of the button

-OnEnter Video: What will be the animation when the mouse passes over the button

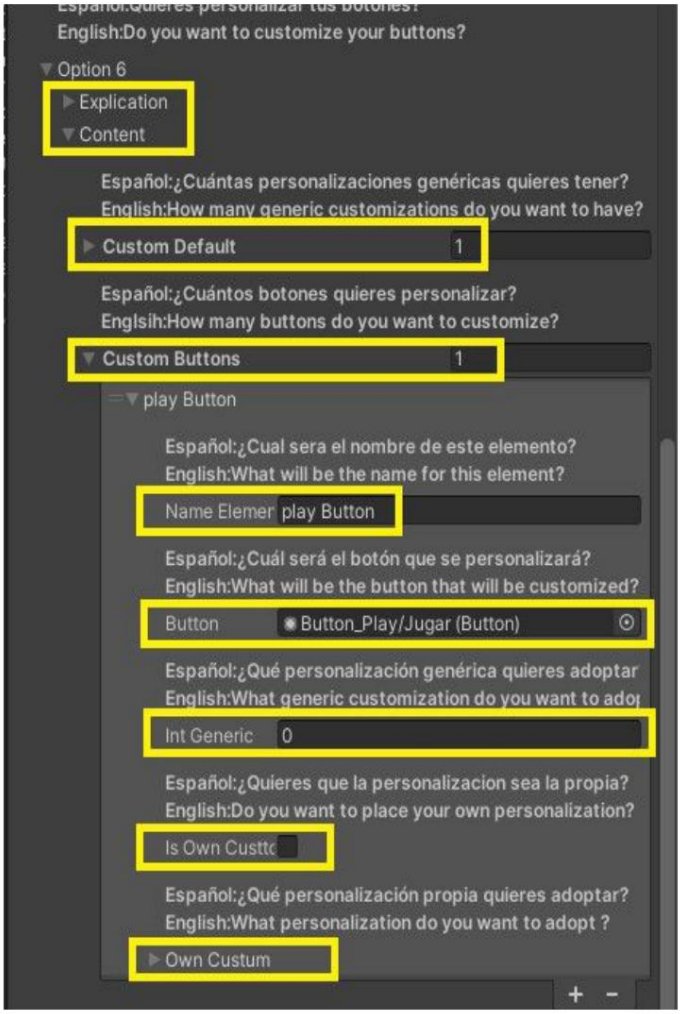
-OnClick Video: What will be the animation when the button is clicked

-Audio Source: You will have to place which will be the audio source where the sound will be applied for the button effect

-OnEnter Sound: The sound that will be emitted when the mouse passes over the button

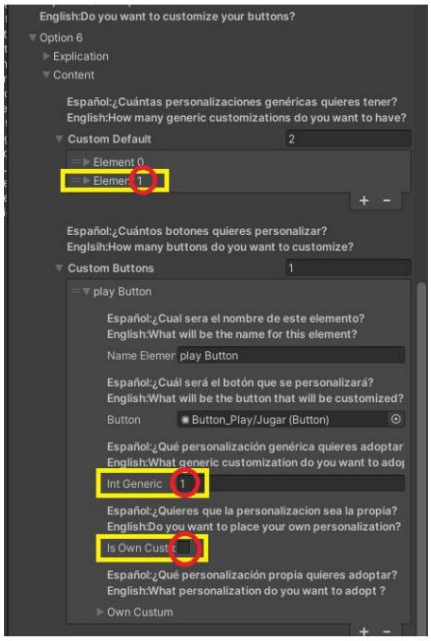
-OnClick Sound: The sound that will be emitted when the button is clicked

-Loop: You can tell if you want the sound to be emitted in a loop when you do something button action

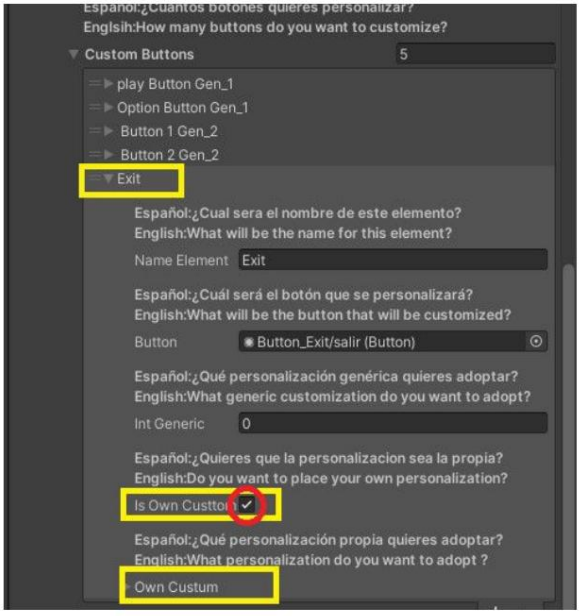


USE

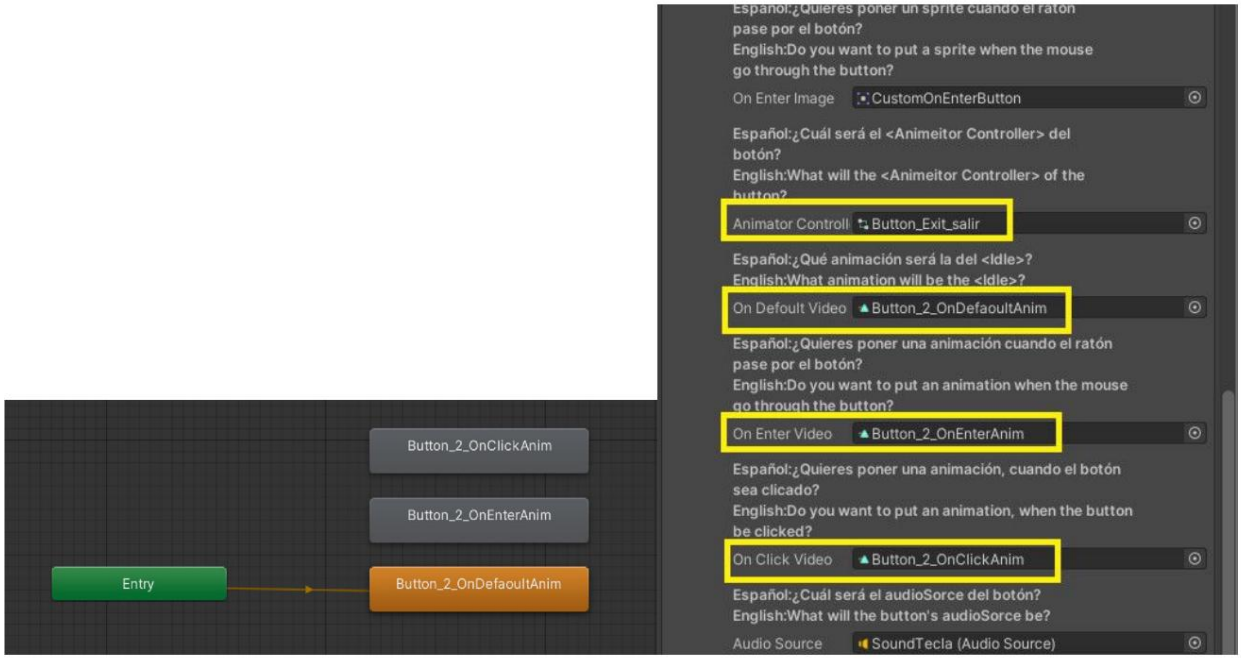
1) If you want a generic customization, in the drop-down <Custom Button> you must have deselected the option <Is Own Custom> and indicate in the part that says <Int Generic> an enumeration from 0 to the number representing the <Custom default> that you want it to adopt for that generic customization and in the <Custom Default> you only have to fill in the images that you want to show, videos or sound, depending on the type of interaction of the mouse with the button, those that you do not want to fill in, nothing happens, no However, you must always place <Image Default>



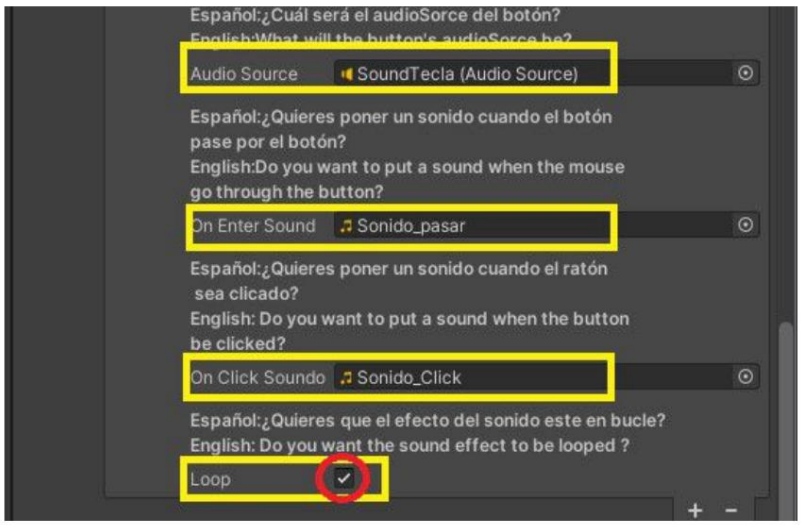
2) Own customization (Not Generic): If you only want the button to have its own customization, in the <Custom Button> dropdown, you do not have to click the <Is Generic> option and in the <Own Custom> you only have to fill in the images, videos or sound that you want to show according to the type of mouse interaction with the button, that gap that you do not want to fill, nothing happens, however you always have to place <Image Default>



3) Add animation: If you only want it to have an animation or add that this button has an animation, you have to create an animation control and create the idle animations, when it is clicked or when the mouse passes over the button and then assign this controller and its animations to the boxes where you are most interested, whether to the generic or to the customization itself



4) Adding sound: If you only want your button to emit sound or you want to add a sound, you have to go through and fill in the sections that interest you, you will have to give it an "Audio source" which will be where the sounds that you want it to emit will be placed , then you will have to have the sounds that interest you and add them in the holes of the generic customization or in the customization of the button, only those that interest you, you also have the option to indicate if you want said sound to be repeated in a loop, just You have to select the option called <Loop>.



WARNING

If in a situation you put more buttons in the list, but in one of them it doesn't have any content in <Custom Buttons>, the system will also notify you, that in the list of buttons, there is an element that has not been assigned any component, this will not affect the operation of the system, it will only give a warning
In the event that an important component is missing for both the generic customization and your own, the system is also designed to notify you of such missing elements.

LEGEND

(*) **Image :** is a user interface component that is used to display images on a screen in a game scene or in an application. Images can be used to create backgrounds, buttons, icons, status indicators, and other visual elements in a user interface.

(**) **Animator Controller :** it is a component that is used to control and coordinate the reproduction of animations in an object in a game scene. The Animator Controller is an important part of Unity's animation system and is used to define the flow of animation and the transitions between them.

(***) **Audio Source:** is a component that is used to play sounds in a game or application scene. The Audio Source is an important part of Unity's audio system and is used to configure and control the playback of sounds.

(****) **Button :** is a component used to create an interactive element in a game scene or user interface. A button can contain text, an image, or other visual content and can be clicked to perform an action that affects the game or another interface

OPTION 07

This part is to give the option of customizing and adding an icon to the mouse thus visualizing it within the game, it is also allowed to create new responses to the integration with the mouse, such as the click, changing the image or animating the mouse as needed. user, icon controls also works with controller joystick for consoles

You have twenty parts: -

Explanation: The documentation of this option is summarized and concise.

- Content: It will contain all the elements you need to be able to use it.

-OnFunction: If you want this function activated in X scene -onDefoultImage(*):

It is the image that will be seen on the mouse by default -onClickImage: It is the image that will be seen on the mouse when you click -Panel Icon Visualizer: The place where the mouse icon will be created

-Animator Controller(**): It will be to place the controller where the animations will be assigned

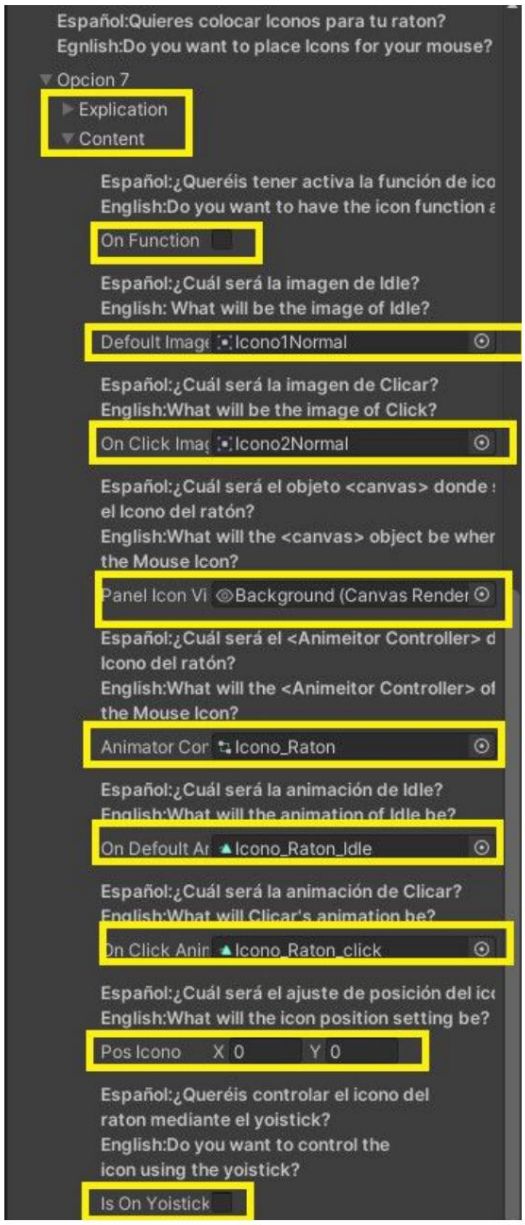
-OnDefoult Anim: What will be the animation of the mouse icon in its default state

-OnClick Anim: What will be the animation of the mouse icon in its click state

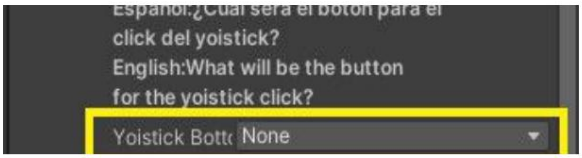
-Pos Icon: If you want the icon to be created in a different position with respect to the mouse pointer, the mouse being the central axis

-Is On Yoistick: If you want the mouse icon to be able to move if there is a console controller

-Yoistick Button: You will indicate which will be the click button of the console controller



-



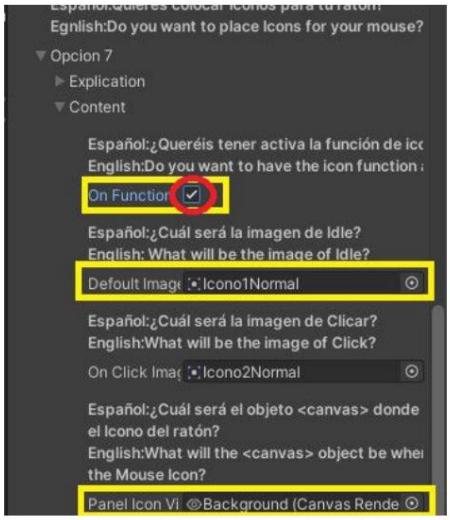
USE

1) Basic Use: To use this option in the most basic way, and you want the mouse to have an icon, you will have to activate which scenes you want the icon to have that function, regardless of whether you later want to activate or deactivate it through your own scripts.

-On Function: To indicate that the scene will have the mouse icon

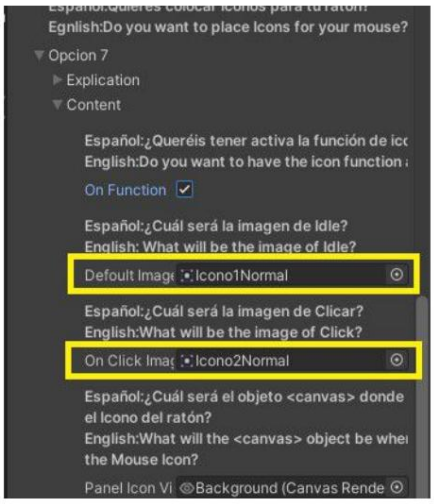
-PanelIconoVisualicer> It will be the general panel where the mouse icon UI will be created

-DefoultImage> The default image that the mouse icon will adopt must be a sprite



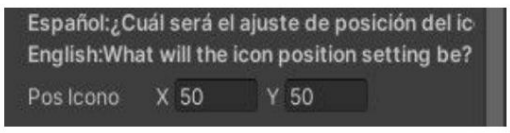
2) Changing image according to its state: If you want the mouse icon to change its image, when it clicks you have to fill in the following section using a Sprite

-OnClickImage Is the image that will be seen on the mouse when it clicks



3) Change origin position: The mouse icon will always appear centered on the mouse position, in its , but if you want center the icon to have another position when it starts, so that the image agree modify the following parameter

-PosIcono If you want the icon to be created in a custom position

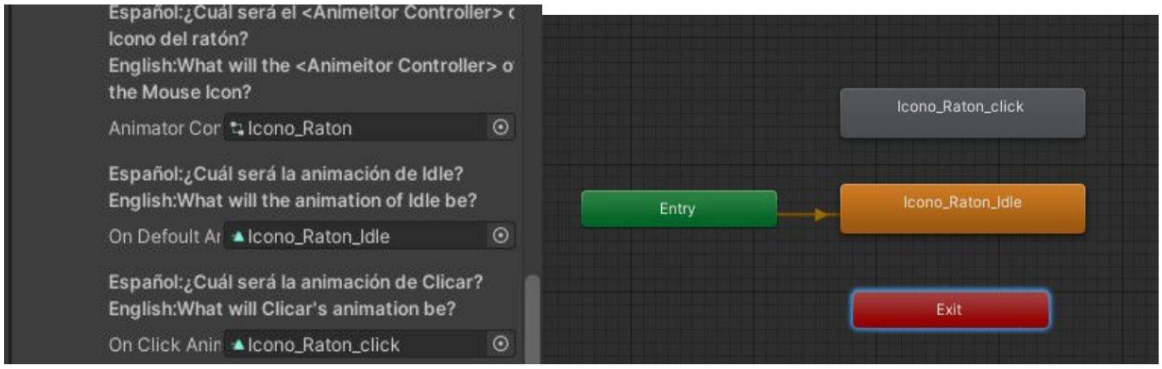


4) Adding animation according to state: If you want the mouse to have some animations you can place them by filling in the following sections, however I suggest that you first create an image and make the corresponding animations and then fill in the system parameters with them

-AnimatorController It will be to place the controller where the animations will be assigned

-OnDefoulAnim What will be the animation of the mouse icon in its default state

-OnClickAnim What will be the animation of the mouse icon in its click state



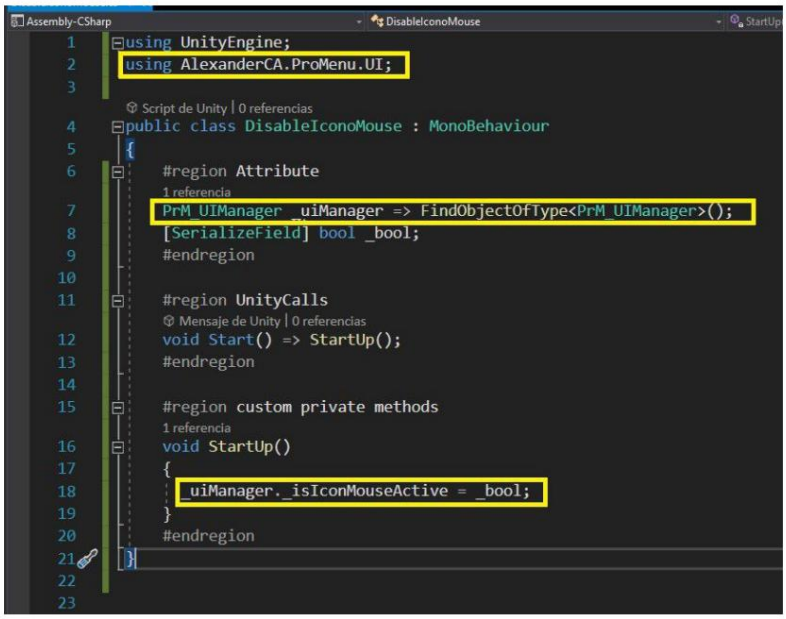
5) Control with command: If you want to control the icon with the command of the console or when it is active, just activate the option

-IsOnYoistick: If you want the mouse icon to be able to move if there is a console controller But if you also want it to also react when clicked, you have to fill in the following parameter

-YoistickBotton: You will indicate what will be the click button of the console controller

WARNING

If you want to control when to have this icon option activated, because there is a kinematics or a situation that needs to be hidden, you only have to refer to the “PrM_UIManager” script and set the “_isIconMouseActive” bool to true.



LEGEND

(*) **Image :** is a user interface component that is used to display images on a screen in a game scene or in an application. Images can be used to create backgrounds, buttons, icons, status indicators, and other visual elements in a user interface.

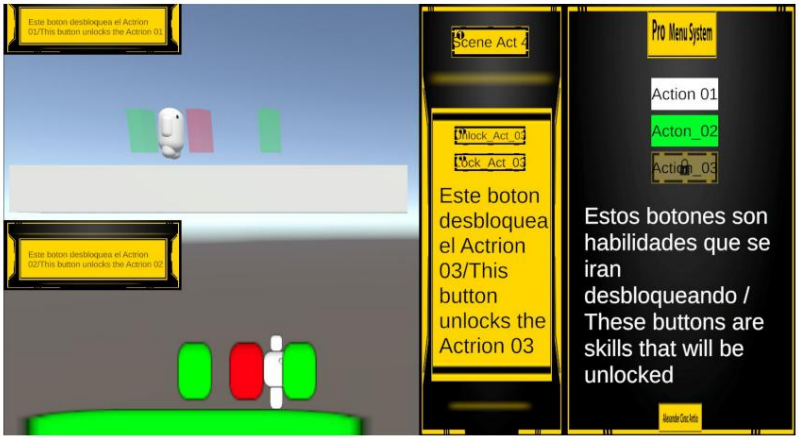
(**) **Animator Controller :** it is a component that is used to control and coordinate the reproduction of animations in an object in a game scene. The Animator Controller is an important part of Unity's animation system and is used to define the flow of animation and the transitions between them.

OPTION 08

This option allows users to create multiple lockable and unlockable buttons by creating a dynamic button interface. This allows game developers to place buttons that will have different states (locked/unlocked).

For example, in a role-playing game (RPG), ability buttons might be locked at the start of the game and unlocked as the player character progresses through the game or acquires certain abilities or resources.

This part publicly saves a list of IDs that are already unlocked, every time a new scene passes or an item is added to the list or removed, it will update and check against another private list of IDs in this way you will know that the new items are to unlock and removed items are to lock



-You have **thirteen**

parts: – Explanation: The documentation of this option is summarized and concise.

– Content: It will contain all the elements you need to be able to use it.

– UnlockUIElements: The button that you want to acquire these states

-Name element: Place a name to identify it.

–ID Element: A number that the user will indicate to identify this button –

Button: Which button will be the one that will be blocked

or unlocked –Image Status: Drop-down to place images that indicate the status of the button

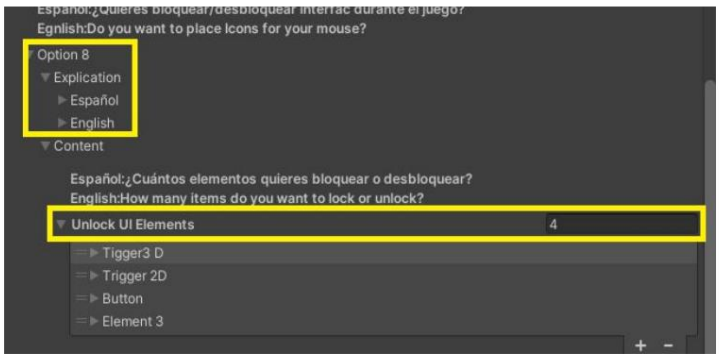
–Image Unlock: Image to indicate the unlocked status

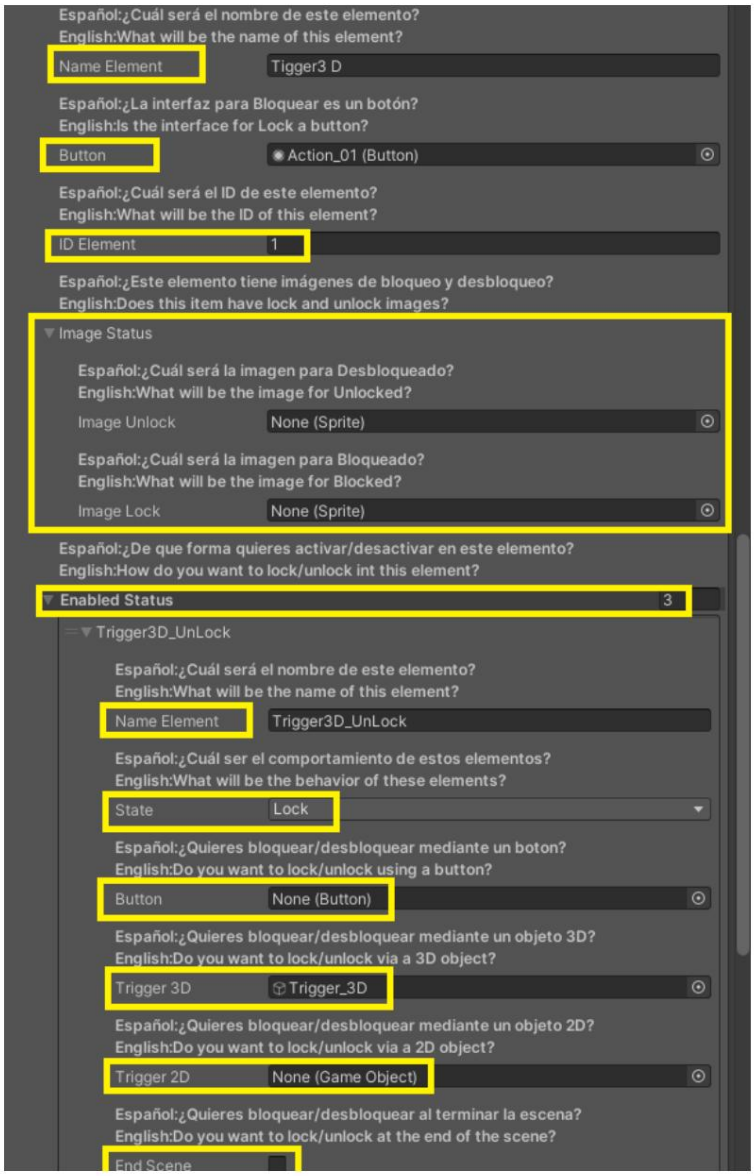
–Image Lock: Image to indicate Image status for locked

– Enabled Status: You can indicate how you want to enable or disable the button

-Name element: Place a name to identify it.

–State: The created element will behave as a blocker or unlocker –Button: When the player presses a button that the user indicates –Trigger 3D: When the player collides with a 3D object –Trigger 2D: When the player collides with a 2D object – End Scene: At the end of a scene



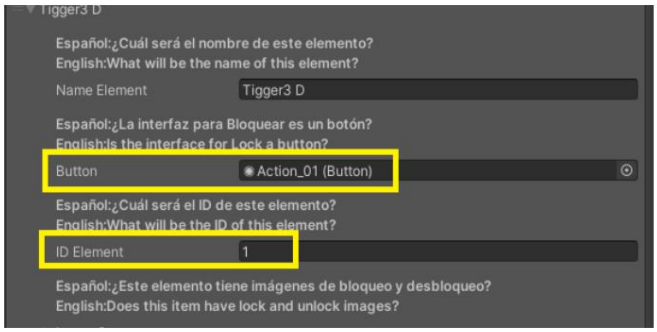


USE

1)Basic Usage: To use this option, you must first indicate the desired number of timers you want to add to the current scene you are working on. To do this, you can go to <Unlock UI Elements> and place the desired number using an enum. If you need more, you can add them additionally.

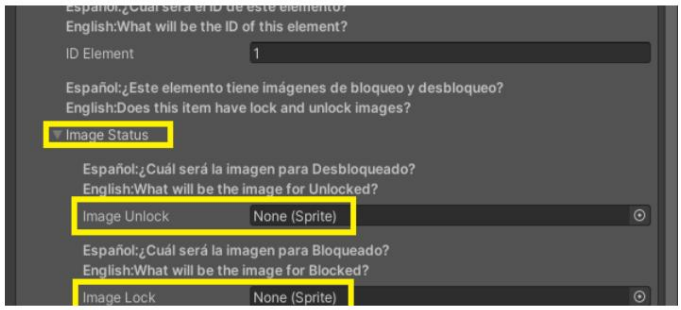
1.1) Place Identification: For the system to work correctly, a numbering not less than 0 is required in the <ID Element> field, so that the corresponding status of "Locked" or "Unlocked" can be assigned later. The identifier number must be unique and exclusive for the corresponding button, and must not be used to identify any other.

1.2) Assign Button to Lock/Unlock: It is not necessary to specify which button will be affected in the <Unlock Button> field, but this is only necessary if the button is present in the scene and the system is required to perform its function. For example, in scene A, the button will only be used to unlock it for use in scene B, but not while in scene A. Therefore, in scene A only the identifier <Element ID> should be included. , while in scene B it must be completed with the same identification number in the <ID Element> field and add the corresponding button in <Unlock Button>.

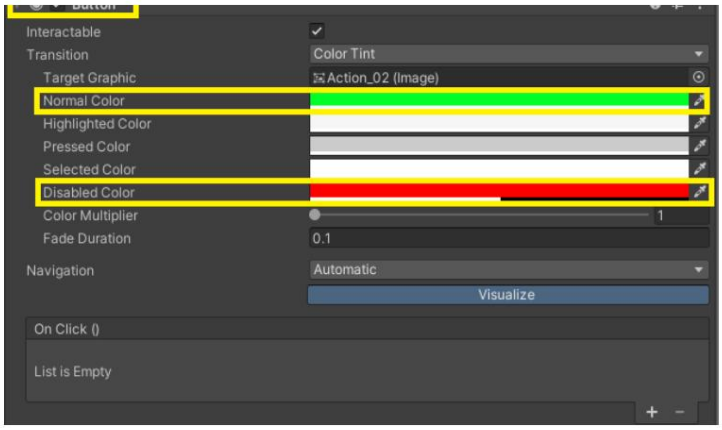


The other sections are optional, since they only provide image changes, different ways in which the system locks/unlocks, and the possibility of adding names to the elements so that the user can identify them more easily.

2) Visual aspect (Optional): If you want the button to adopt an image according to its status, you just have to display the <Image Status> section. Both sections do not need to be completed, only those that are relevant to your needs.
2.1) Image Unlock: Pass the sprite(*) that will visually indicate that it is unlocked 2.2) Image Lock: Pass the sprite that will visually indicate that it is locked

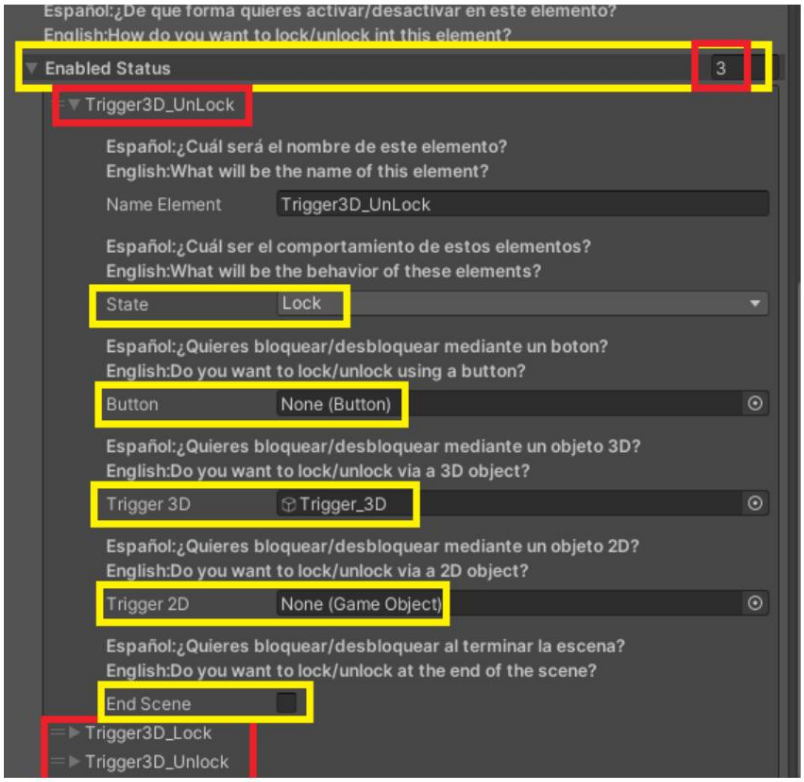


2.3) If you just want the buttons to have a different color without adding images, since by default when the button is locked or unlocked its color is gray or white respectively, you just need to access the "button" component in the color palette section and specify which color it should assume when it is locked at "Disabled Color" and when it is unlocked at "Normal Color".



3)Ways to Lock/Unlock(Optional): The system offers predefined ways to lock/unlock. In addition,
we will provide a way to do it through code for those users who want more control and customization.

3.1) Name Element: Place a name so that the user can better identify the element.
3.2) State: You will indicate what type of function will take the following parameters of this subsection. If it behaved like Unlock or Lock.
3.3) Items to lock/unlock: Here are some basic options for the
The user can choose how they want certain interactions to be blocked/unblocked based on what they have selected in the <State> field.
-Button element: It will be dragging a button from the interface to perform the action
-Trigger 3D element: You must drag a 3D object from the scene so that it collides with the object that has the "Player" tag. to the object Trigger 3D elements what is necessary will be added automatically to function correctly. It is important to make sure that the "Player" object has collision and that the "Trigger"(**) option is not activated, otherwise there will be no collision -Trigger 2D element: You will
have to drag a 2D object from the scene so that it collides with the object that has the tag "Player". to the object automatically to function Trigger 3D elements what is necessary will be added correctly. It is important to make sure that the "Player" object has a collision and that the "Trigger"(**) option is not activated, otherwise there will be no collision -End Scene: If you want to unlock/block as soon
as you load the scene

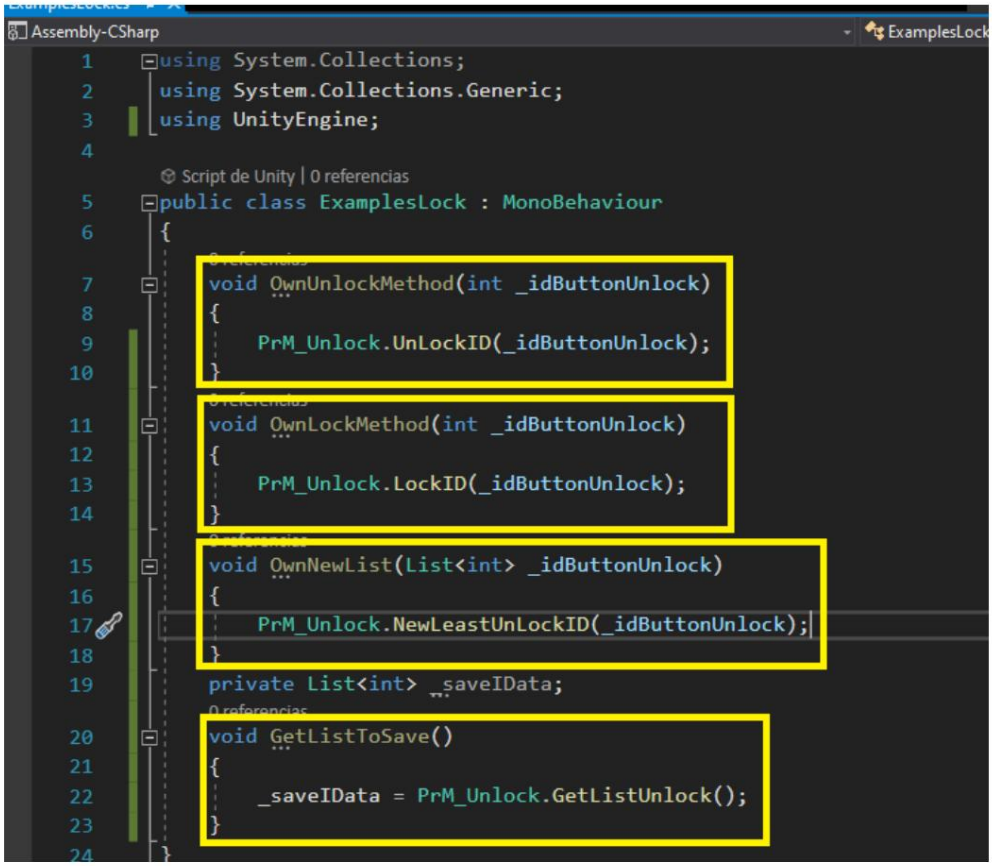


3.4) If you want to block or unlock programmatically, I leave you the following steps
-Unlock: To add the ID that will affect the button you want to unlock, you just have to add the following code snippet: "PrM_Unlock.UnLockID()" and use the "UnLockID()" method passing the ID number between the brackets "( )". The system will then identify this ID to unlock the corresponding button.
-Lock: To add the ID that will affect the button you want to unlock, you just have to add the following code snippet: "PrM_Unlock.LockID()" and use the "LockID()" method passing the ID number between the brackets "( )". The system will then identify this ID to unlock the corresponding button.

-Get the list of IDs to save it: To get the list of all IDs, either to save data or for any other purpose the user wants, add the following

snippet: "PrM_Unlock.GetListUnlock()" and use the "GetListUnlock()" method. In this way, you will get the list of IDs that are unlocked.

-Pass a new list when loading game or data: To stop a new list with the new ID, either for loading data or for any other purpose the user wishes, add the following code snippet: "PrM_Unlock._instanceUIManager. NewLeastUnLockID ( )" and use the method NewLeastUnLockID ()". In this way, you will add the list of IDs that will be to unlock.



WARNING

-Remember that this modality does not scale or modify the position of the images.
-It is essential to assign a unique ID to each new enabled section, since this is essential for its proper functioning.
On the other hand, other aspects, such as visual customization, the different ways to lock or unlock, or the possibility of adding names to the elements to facilitate their identification, are optional and can add value to improve the user experience -If you want the button to display its state using
colors instead of images, you can achieve this simply by changing the colors of the button component that Unity offers in its color palette. In this case, you do not need to fill the slots.

-The system will automatically transfer the unlocked button information between scenes. In this way, the user will only need to program the save and load system, and store the updated list each time the player loads a game - This system has an automatic
unlock method at start and during the scene, which checks all the buttons and their respective IDs with the unlock IDs list. Those IDs that are not on the unlock list will be blocked. Also, if any item is removed from the list of unlocked IDs by the options suggested by the option or by code or by passing a new list, the system will recheck if the deleted item is still not on the list and will be locked automatically.

-To add the ID that will affect the button you want to unlock, you just have to add the following code snippet: "PrM_Unlock.UnLockID()" and use the "UnLockID()" method passing the ID number between the brackets "()" .
The system will then identify this ID to unlock the corresponding button.
-To add the ID that will affect the button you want to unlock, you just have to add the following code snippet: "PrM_Unlock.LockID()" and use the "LockID()" method passing the ID number between the brackets "()" . The system will then identify this ID to unlock the corresponding button.
-To get the list of all IDs, either for saving data or for any other purpose the user wants, add the following code snippet: "PrM_Unlock.GetListUnlock()" and use the "GetListUnlock()" method. In this way, you will get the list of IDs that are unlocked.
-To make a new list with the new IDs, either for loading data or for any other purpose the user wishes, add the following code snippet: "PrM_Unlock.NewLeastUnLockID ()" and use the " method NewLeastUnLockID ()". In this way, you will add the list of IDs that will be to unlock.

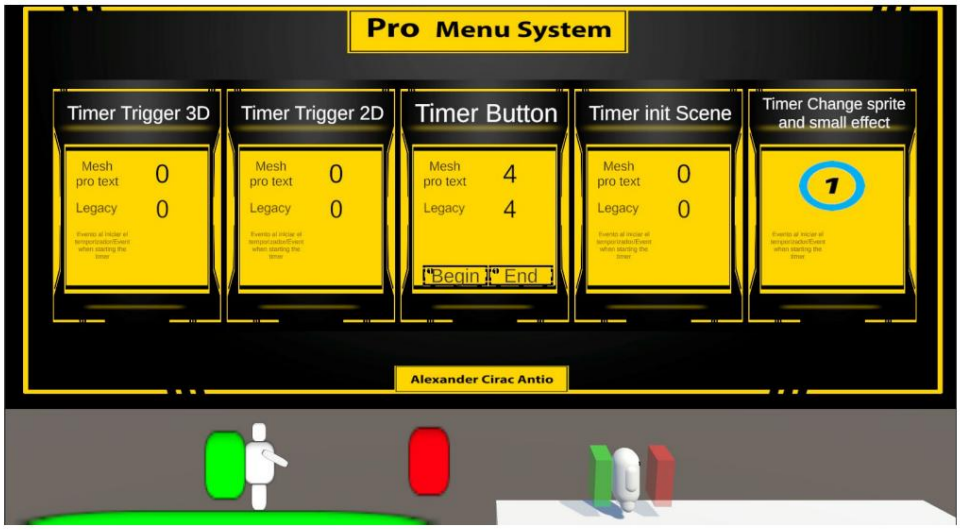
LEGEND

(*)Trigger (*is a component that can be added to a game object to detect when another object enters or leaves a specific area in the game space.)

(**)Sprites (*It is a two-dimensional image used to represent graphic elements in a game, such as characters, objects, backgrounds, and user interface elements. Sprites are a common way of representing graphic elements in 2D games and are used to create animations and visual effects in the game.

OPTION 09

This option provides a useful tool for game developers, allowing them to create timer panels or a countdown based on the user's needs. In addition, it offers the option of when to set this functionality to be activated, either through a button on the interface, a collision with the player in the scene, or simply starting the scene itself. Likewise, this system allows the addition of events that are activated both when the counter starts and when it ends, providing greater flexibility and customization in game design.



-You have **sixteen** parts:

– Explanation: The documentation of this option is summarized and concise.

– Content: It will contain all the elements you need to be able to use it.

-Timers elements: Here you will configure the parameters of each timer and fill each slot with what you need.

-Name element: Place a name to identify it.

-Timer max: It will indicate the total duration.

-Beginig event: If you want to call an event when it starts.

-Ending event: If you want to call an event when it ends.

-Activators or deactivators: As you want it to start or deactivate, you can create as many elements as you need in the scene.

-Name element: Place a name to identify it.

-Button element: You want it to start when the player presses a button chosen by the user.

-Trigger 3D element: You want it to start when the <Player> tag collides with a 3D object.

-Trigger 2D element: You want it to be triggered when the <Player> tag collides with a 2D object.

-State function: You will indicate what behaviors all the previously filled elements will have, if they will be activators or deactivators.

-Visual effects: How you want it to be displayed -Mesh pro text: If you want it to be displayed in new Unity texts.

-Legacy text: If you want it to be displayed in the old Unity texts.

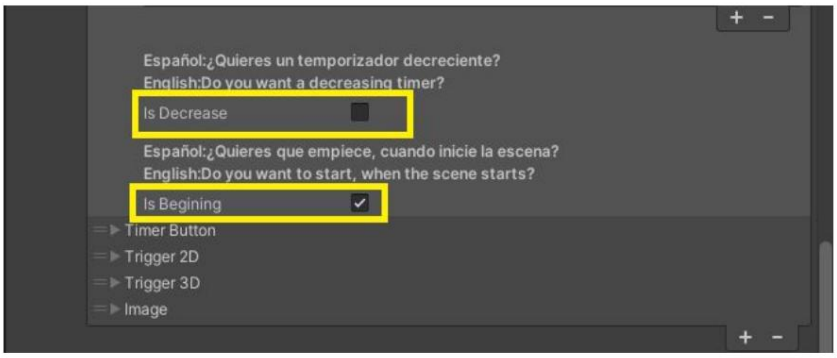
-Effect image: Indicates which image will have visual changes.

-Different sprites: You will pass the different images that change progressively over time.

-Efect small: Activate it if you want the image/object to have an animated effect of reducing its size as time goes by.

-Is Begining: You want them to activate when the scene begins.

-Is Decrease: Activating it will make the countdown count down.



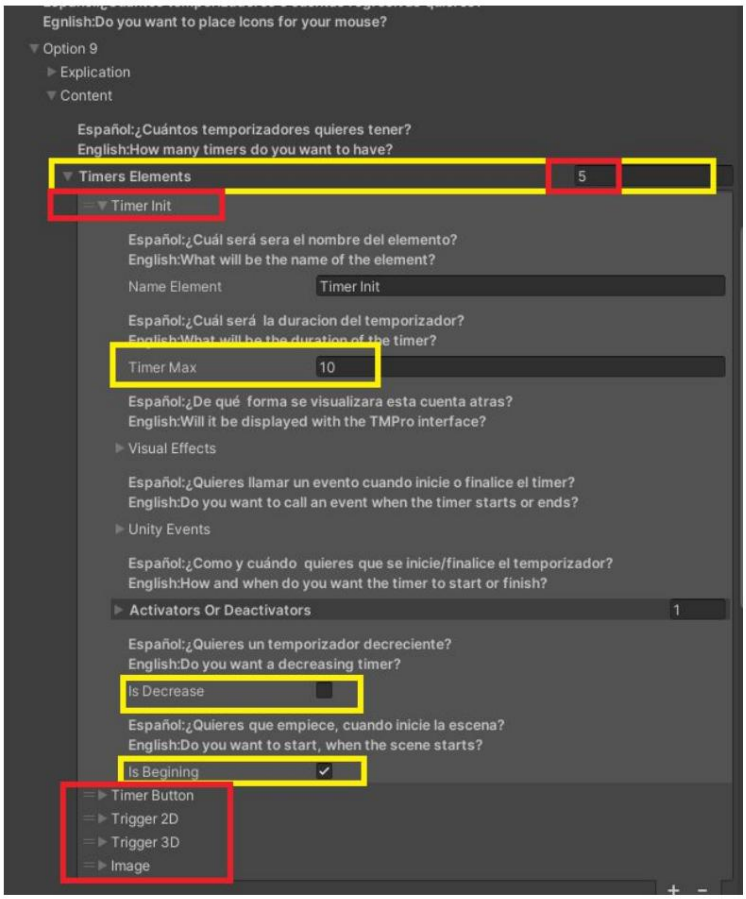
USE

1) Basic Usage: To use this option, you must first indicate the desired number of timers you want to add to the current scene you are working on. To do this, you can go to <Timers Elements> and place the desired number using an enum. If you need more timers, you can add them additionally.

1.1) Time: it is necessary to establish the maximum length of time that each timer will have by means of an enumeration in the <Timer Max> field. This value is important because it determines how long it will take for the timer to finish.
It is essential to ensure that the value of <Timer Max> is not less than zero.
On the other hand, the speed of the timer is set using the value "Timer.DeltaTimer"(*).

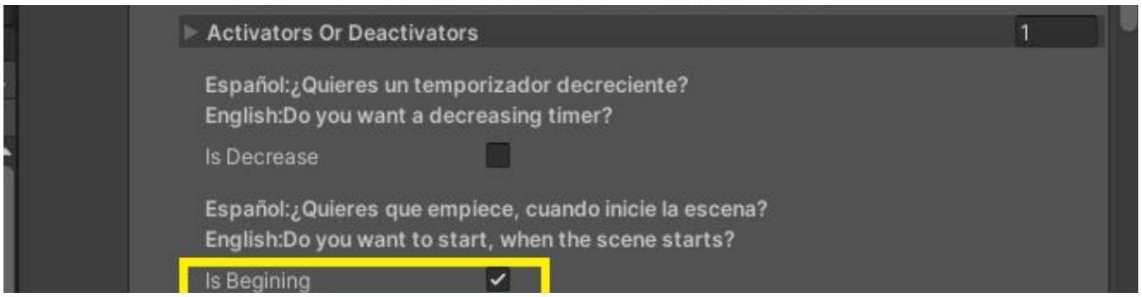
1.2) Countdown timer: If you want a countdown or a countdown, that is, it will go from the maximum time to 0, you only have to select the option called <Is Decrease>.

1.3) Start Timer: If you want to create a countdown or go back in time, that is, from the maximum time to 0, you just have to select the <Is Decrease> option in the timer. This option will reverse the direction of the timer so that the time decreases instead of increasing it, resulting in a countdown.

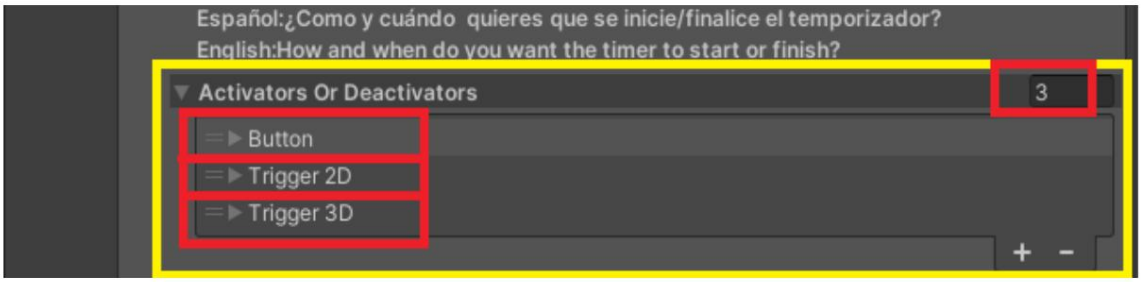


The other sections are already optional, since they only provide image changes, animations, assign events when activating/deactivating, extra ways to activate/deactivate them and add names to the elements so that the user can better identify them.

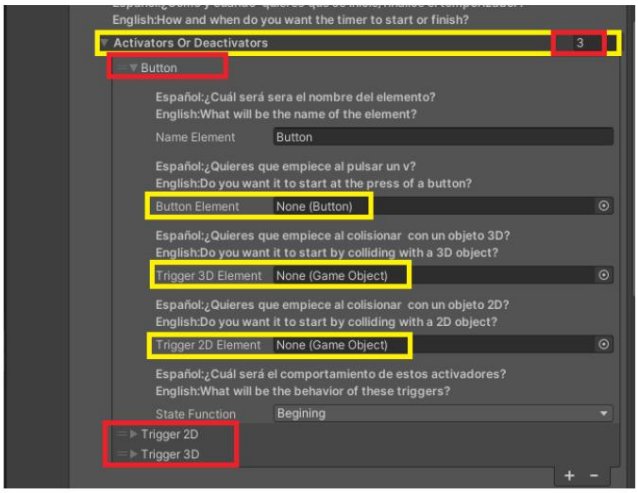
2) Different ways to activate/deactivate (Optional): It is possible to configure different ways to start or end the timer prematurely through a specific section. For correct operation, the system must know at least one way of activation, either at the beginning of the scene or through the parameters offered in this section. On the other hand, the system already has automatically implemented the completion of the timer, when reaching the maximum or minimum time, but the user can indicate if he wants to finish it in other ways. This section allows you to implement as many different ways of starting or ending according to the needs of the game 2.1) The first way to activate it is when the scene is loaded, the option is called <isBegining>



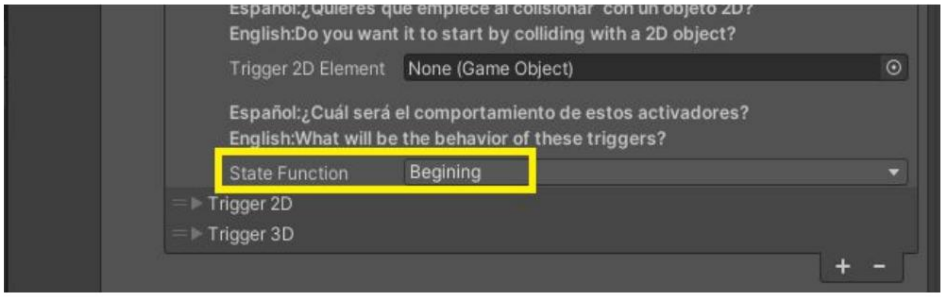
2.2) There is a second way to activate or deactivate, through the options that are displayed in the "activatorsOrDeactivators" section. It is divided as follows:



2.2.1) In what way will the interaction be activated or deactivated -Button element: It will be dragging a button on the interface to perform the action -Trigger 3D element: You must drag a 3D object from the scene to collide with the object tagged "Player". The object will automatically add what is necessary for it to function correctly. It is important to make sure that the "Player" object has collision and that the "Trigger"(**) option is not activated, otherwise there will be no collision -Trigger 2D element: You will have to drag a 2D object from the scene so that it collides with the object that has the tag "Player". The object will automatically add what is necessary for it to function correctly. It is important to make sure that the "Player" object has collision and that the "Trigger"(**) option is not activated, otherwise there will be no collision

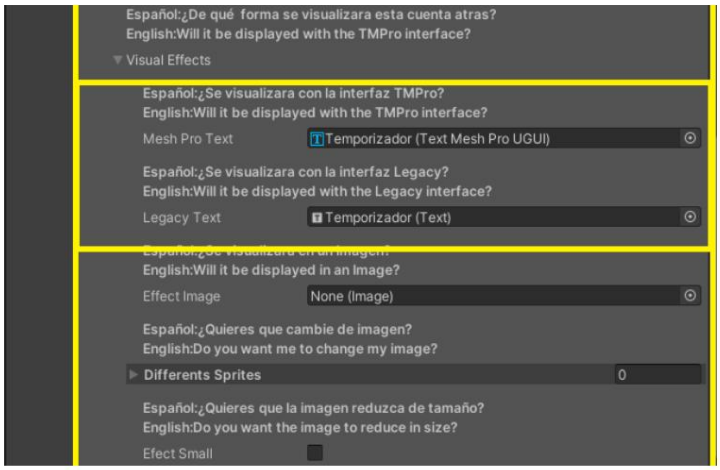


2.2.2) Finally, you must indicate an option that will affect all the previous elements. You must choose if they will be activators or deactivators through the option called <State function".



3) Customize the way it is displayed (Optional): If you want to add visual effects to the timer in your game, you can choose from several available options. For example, you can display the progression of the timer in a text, change the image as the time progresses, or apply an animated shrinking effect. To achieve this, simply drag the corresponding objects into the designated slots.

3.1) Legacy text and Mesh Pro UGUI: The first two are to indicate the timer by means of a text, it works in both modes, the Legacy(***) and the new one is the Mesh Pro UGUI (****)

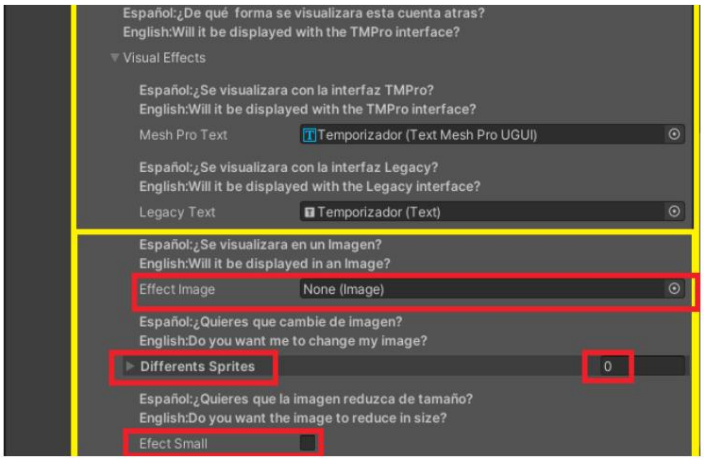


3.2) If you want the timer to be displayed with image changes or that it affects the size (by means of a system animation), you must specify which will be the image component of the interface in which the change will be applied 3.2.1)

Effect Image : The component that will be affected by the animations or the change of sprites(*****)

3.2.2) Different Sprites: Here you can add images that will change according to the elapsed time. The way in which the change will be made is by calculating the maximum time that you have indicated and the number of images that you want to show. The system calculates how much time must pass before the next image is displayed. For example, if you have a 45 second timer and there are a total of 10 images, the calculation would be 45/10 = 4.5, so the image will change every 4.5 seconds.

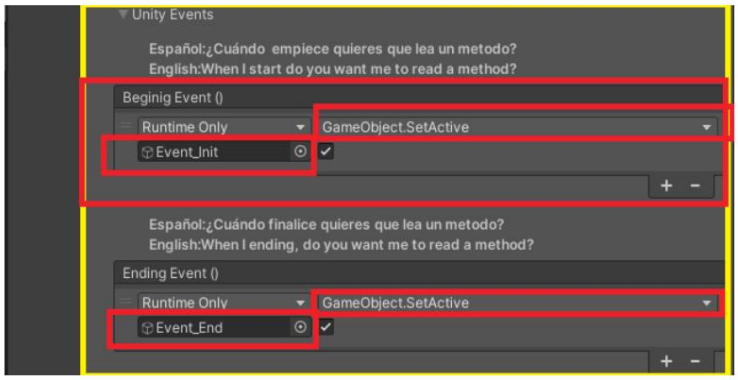
3.2.3) Effect Small: Add the animated effect of shrinking the image



4) Custom Events (Optional): As with all timers, something usually happens when a new one ends or starts. This option allows you to trigger custom events for the various states of the timer, should you wish to add any. To do this, you just need to make sure that the corresponding method in your code is public. Then drag it to where you want it to be called and specify which method should be called.

4.1) Beginning event: When the timer starts, it will call any event that is in the list. In order to add your own method to the list, it's important that you define it as public in your code. Once this is done, drag the script to the corresponding section and display the options in "no function". Find your method and select it so that it is associated with the corresponding event.

4.2) Ending event: When the timer ends, it will call any event that is in the list. In order to add your own method to the list, it's important that you define it as public in your code. Once this is done, drag the script to the corresponding section and display the options in "no function". Find your method and select it so that it is associated with the corresponding event.



WARNING

In the <Timer Max> section, it must never be less than
0. In this option it is important to indicate how each timer will be activated, either when the scene starts (selecting the <isBeginning> option or using any of the options available in the drop-down menu <activatorsOrDeactivators> In this way, you can define the way in which the timers will be activated and adapt their operation to your specific needs.
It is important to remember that for the system to detect any interaction with the triggers, it is necessary for the object that causes the crash to be assigned the <Player> tag while the trigger itself must not carry that same tag. In this way, the system will be able to correctly identify the colliding object and process the interaction appropriately.

Remember that the display options, the content of <activatorsOrDeactivators > and the events are optional

LEGEND

(*)Timer.DeltaTimer (*is a variable that indicates the time in seconds that has elapsed since the rendered the last frame.)

(**)Trigger (*is a component that can be added to a game object to detect when another object enters or leaves a specified area in the game space.)

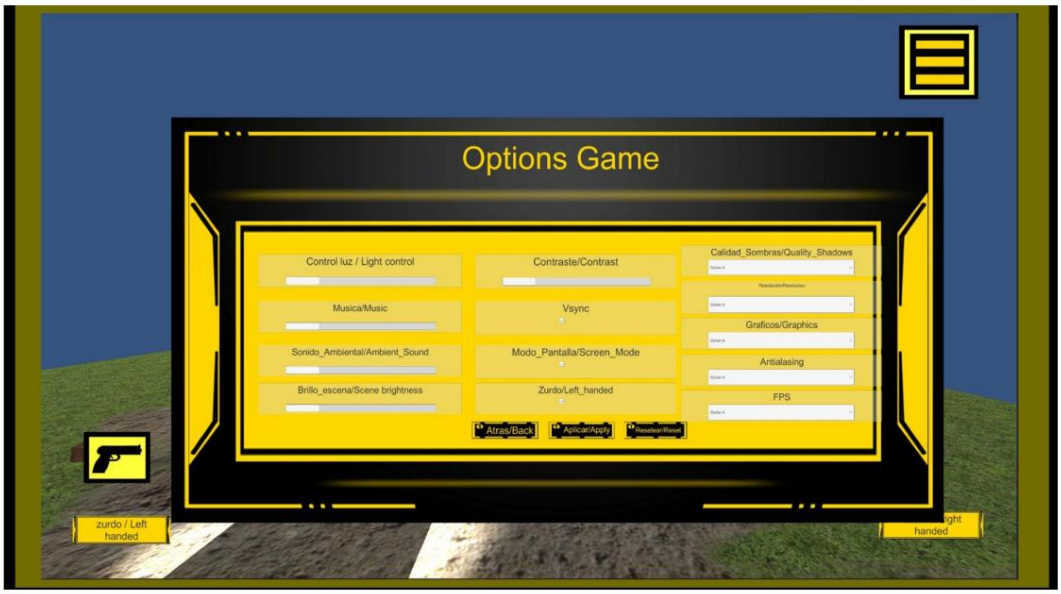
(***)Legacy (*The Legacy variable in Unity is a user interface (UI) component used to display text on a game screen. Unlike Mesh Pro UGUI, which is used to create UI elements detailed and custom, the Legacy component is used to display simple text on the screen.)

(****)Mesh Pro UGUI (*Mesh Pro UGUI is a feature in Unity that allows game developers to create more detailed, high-quality user interface (UI) elements by using custom meshes.
This means that UI elements can have a more realistic and detailed appearance, and can be animated and customized with greater flexibility than traditional UI elements. In addition, Mesh Pro UGUI also improves the performance of games, making them more suitable for mobile devices and other resource-constrained systems.

(*****)Sprites (*It is a two-dimensional image used to represent graphic elements in a game, such as characters, objects, backgrounds, and user interface elements. Sprites are a common way to represent graphic elements in games 2D and are used to create animations and visual effects in the game.

PRM_OPTGAMEMANAGER

This section of the script is to be able to control and add options so that the player can modulate and assign different options so that his computer can perform in the best way, or enjoy the highest quality that the user can offer to the player, assigning different options that allow and facilitate greater control. The options that the user can choose and add to it are of their own choosing. It can offer general brightness, sound control, scene brightness, shadow quality, light quality, antialasing



Like every important script, it has two essential parts, which are the following sections:

1) Explanation: It will explain in a general way everything you can do in this script, divided into <Explanation> <Use> <Warning>
2) Content: Here you can give functionality to the component that interests you that adopts said function, each question also has its <Explanation> <Use> <Warning> in case you need more information about it

USE

Expand the section that interests you. Each option has its explanation and the content to place or drag the objects in the indicated slots for the option you want to use. There are a total of 6 options to customize the Graphic Options

-Option_0: Add a general lighting control option that affects scene and UI
-Option_1: Add a multiple Sound control option and a general one
-Option_2: Add a Camera Contrast control option
-Option_3: Add an option to control scene lighting in the camera
-Option_4: Add a Graphic Options control option that will control
    -Apply changes: Different options to make these changes
    -Vsync Option: To control the Vsync
    -Shadow Option: To control the quality of the game's shadow
    -Resolution Screen: To control the resolution of the chosen screen
    -Windows Mode: To apply the Windows mode of the screen
    -Grafic Option: To control the different options of the Graphic quality
    -Antialasing Option: To control the different antialasing options
    -FPS Option: To control the different options of the FPS speeds
-Option_5: Add a simple option for mobile devices to be able to change the order of the UI by means of changes for people who are left or right handed

WARNING

You can manipulate the information of the variables of each option externally or save the variables to be able to put them in a saving system. I will comment each section separately in the Warnings section, to know how to access those variables.
The system itself takes care that at each level, the values of the options are remembered and updated in the event that it is different from the default options (this does not mean that each level has to be configured and placed all the things and their values), in this way the user does not have to worry that at each level that is loaded he must remember the variables of his chosen options.

OPTION 0

This section will be to control the general lighting of the game, which visually affects the interface and the scene, giving a sensation of more brightness and luminosity in both parts

-You have **four** parts:

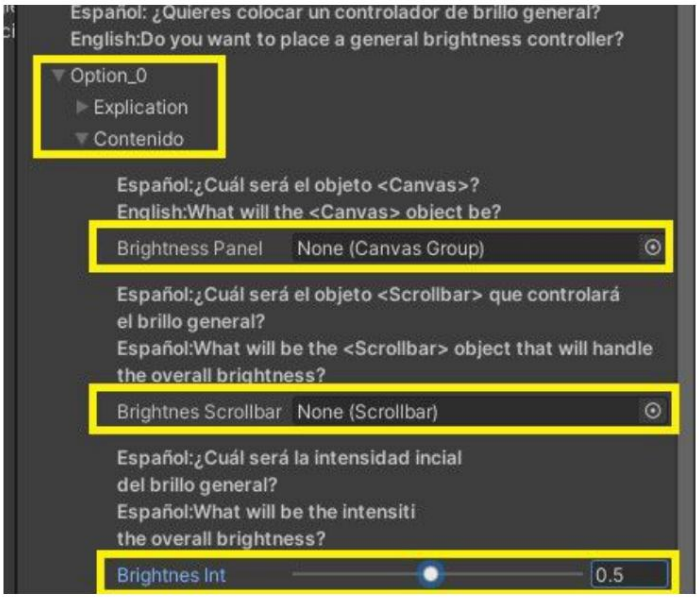
– Explanation: The documentation of this option is summarized and concise.

– Content: It will contain all the elements you need to be able to use it.

-Brightness Panel: Here is where the lighting will be integrated

-Brightness Scroball: Here is to control the intensity of lighting

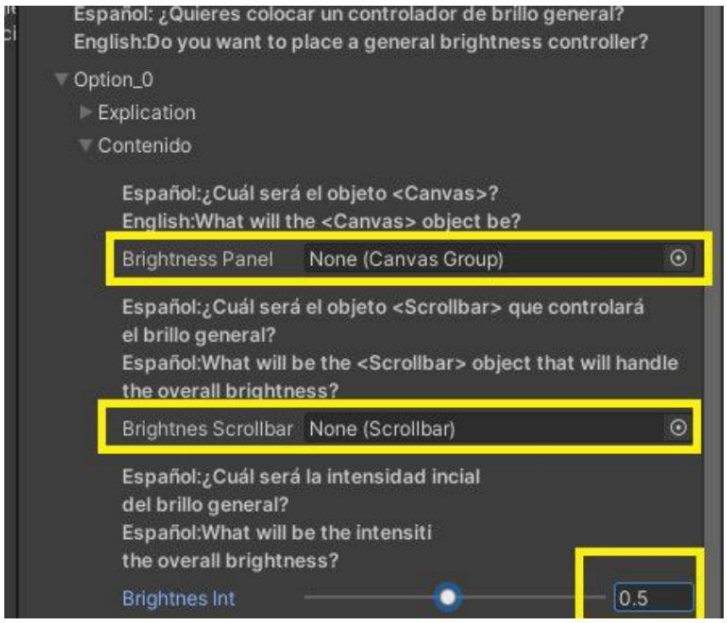
-BrightnesInt : Here you will indicate what the initial intensity of the general lighting will be



USE

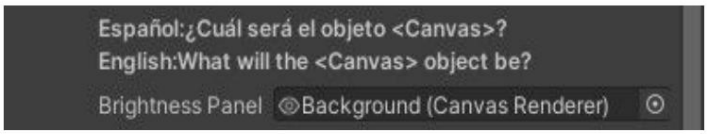
1) Basic use: To control the brightness you have to drag two things, the panel that will control the UI canvas and the scroll that will indicate the intensity. So that in this way you can place the object that shows the general lighting effect -Brightness Panel: Drag the panel that you want the lighting effect to be on -Brightness Scroball: Drag the scrollbar component so you can regulate and control The intensity

-Brightness Int: It is to give an initial intensity



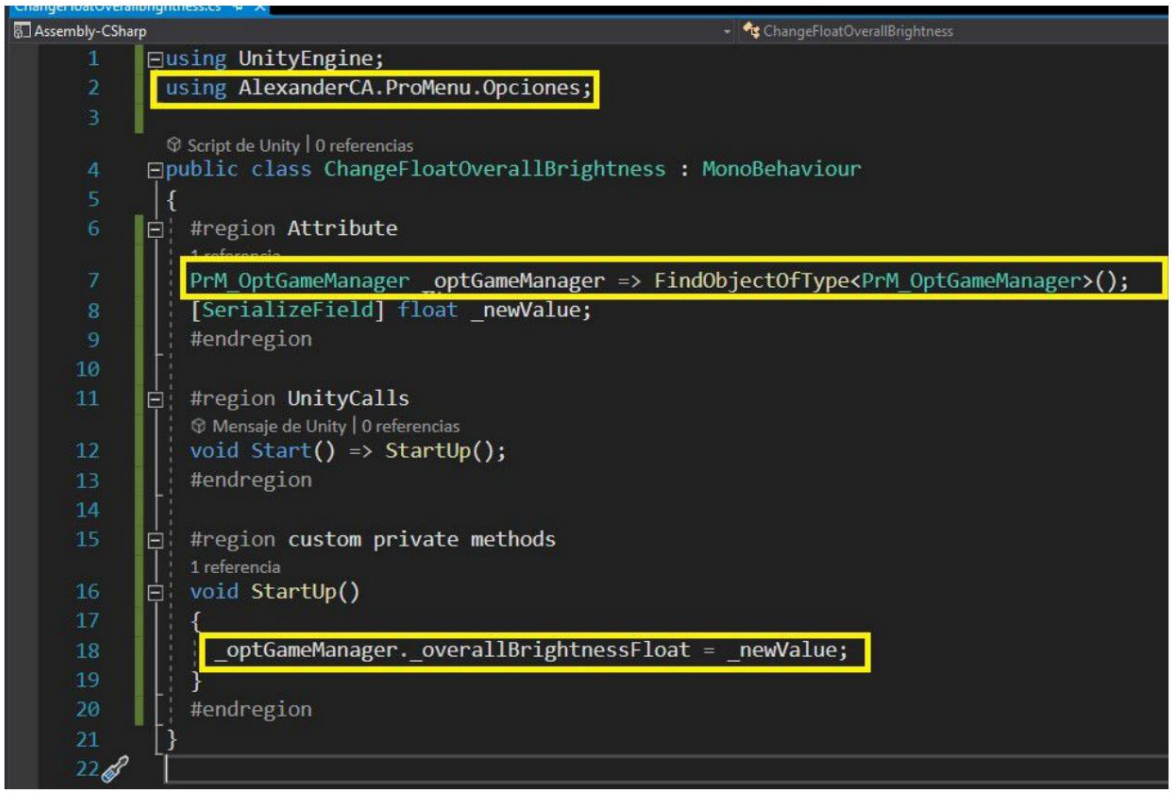
• 2) In the event that a scene does not have a scrollbar to control the intensity of the lighting, nothing happens, you just have to pass the brightness Panel and with that it will be more than enough to have the effect of the general brightness according to the information to be transferred from scene to scene by DataPersisten

-Brightness Panel: Drag the panel that you want to put the lighting effect on



WARNING

If you wanted to modify the value of the general lighting float option for whatever reason, you just have to reference the script “PrM_OptGameManager” and put the value you want in the float “_overallBrightnessFloat”



OPTION 01

This section will be to control the sound of each element, eg: ambient sound, music sound, vfx sound, everything you want to insert, which will affect each scene in a sound way

-You have **five** parts:

– Explanation: The documentation of this option is summarized and concise.

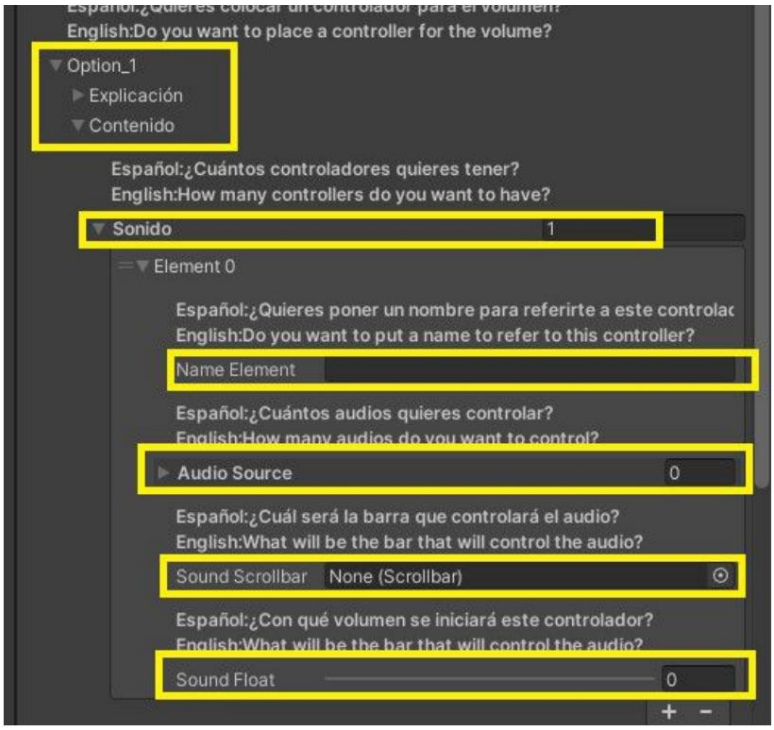
– Content: It will contain all the elements you need to be able to use it.

- ElementName: Here you will indicate what the name of this controller will be

-AudioSource: Here is to place all the source audios that you want to control

-SoundScrollbar: Here is to control the volume of the audio

-soundFloat: Here you will indicate what the initial volume will be



USE

1) Basic use: To control the sound you have to drag, two things, the AudioSource that will be controlled and the scroll that will indicate the volume of said audios.

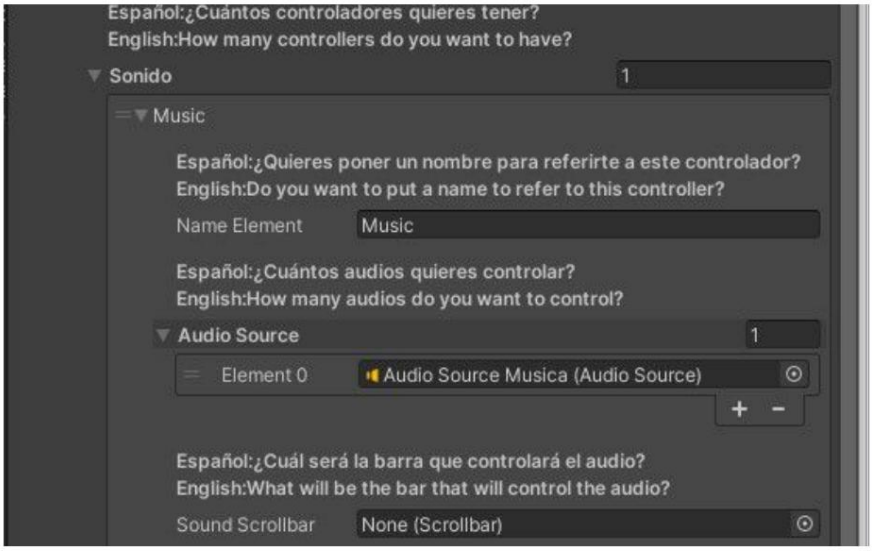
-AudioSource: Drag the source audios you want to control -Audio

Scroball: Drag the scrollbar component so you can regulate and control the volume -Brightness

Int: It is to give an initial volume

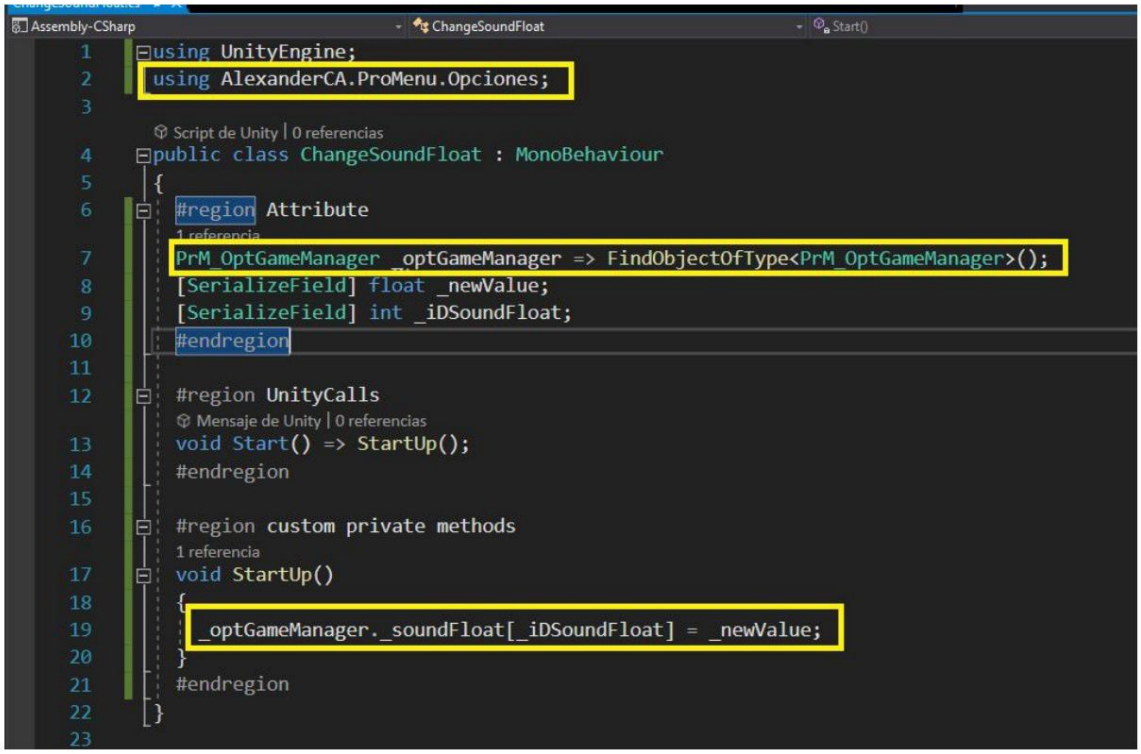
2) Sound effect only: In the event that a scene does not have a scrollbar to control the volume, nothing happens, you just have to pass the audioSource that you want and with that it will be more than enough to have the volume effect according to the information that is passed from scene to scene by DataPersist

-AudioSource: Drag the panel that you want the lighting effect to be on



WARNING

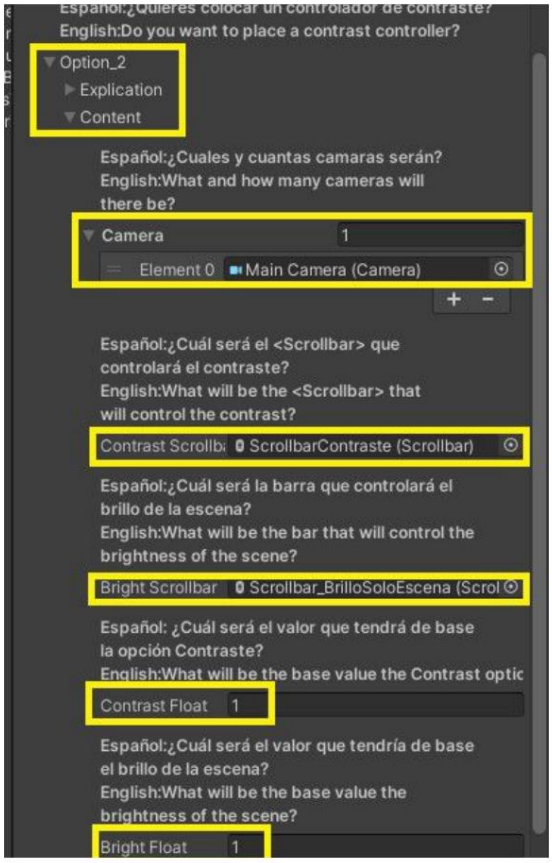
If you wanted to modify the value of the general lighting float option for whatever reason, you just have to refer to the script "PrM_OptGameManager" and put the value you want in the float "_soundFloat" but before you have to pass that value of the list of flaots you want to change with the “_iDSoundFloat”



Remember that each scene must have the same amount of audios and source audios even if it is not used and it is advisable that the first scene load the correct sound data

OPTION 02

This section will be to control the contrast and lighting of the cameras in the scene, which visually affects the contrast, giving a feeling of more definition between black and white tones and their lighting that is in the scene -You have **six** parts : – Explanation: The documentation
of this option is
summarized and concise.
– Content: It will contain all the elements you need to be able to use it.
-Camera: Here you will place all the cameras that you want the contrast to display
-Contrast Scroball: Here is to control the contrast intensity
- Bright Scroball: Here is to control the contrast intensity
-ContrastFloat : Here you will indicate what the initial intensity of the contrast will be
-BrightFloat : Here you will indicate what the initial intensity of the contrast will be



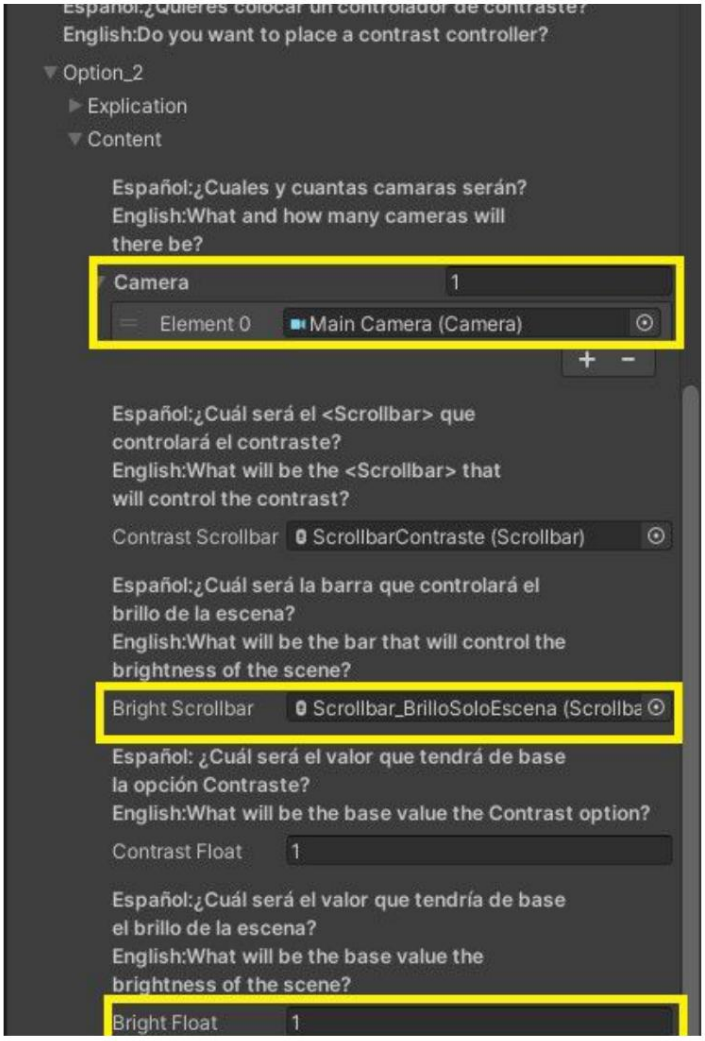
USE

1) Basic use: To control the brightness you have to drag two things, the camera and the scroll that will indicate the intensity. And finally if you want the initial value that it will have. So that in this way place the object that shows the effect of lighting.

-Camera: Drag the camera that you want to put on the lighting effect

-Brightness Scroball: You drag the scrollbar component so you can regulate and control The intensity

-Brightness float: It is to give an initial intensity

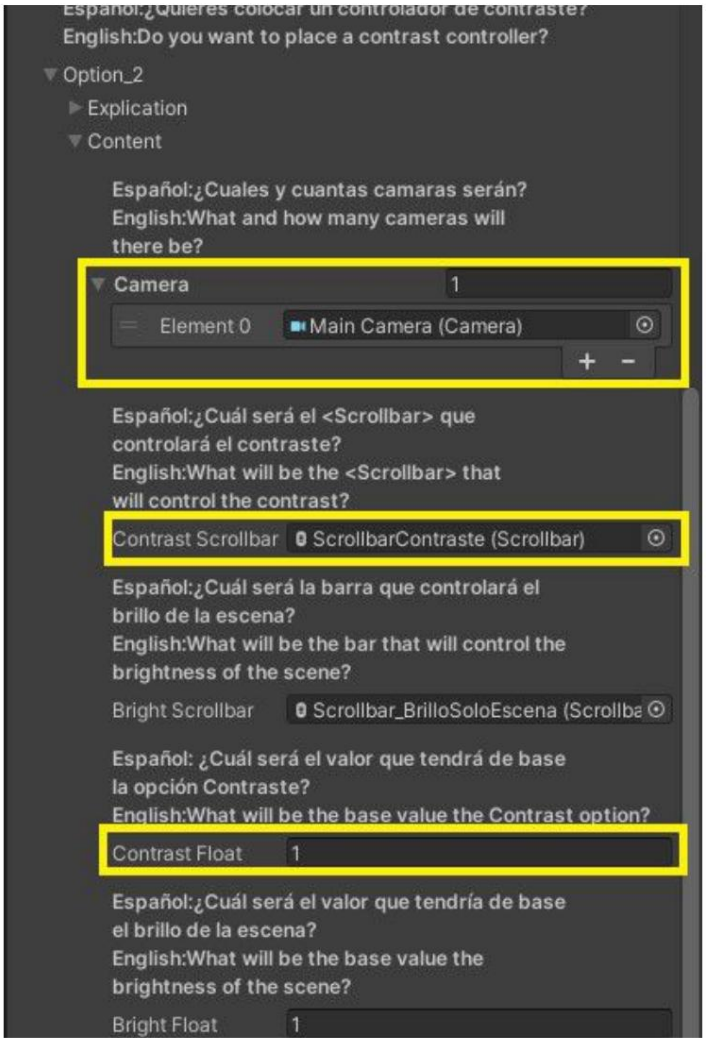


2) Control Contrast: To control the contrast you have to drag two things, the camera and the scroll that indicate the intensity. And finally if you want the initial value that it will have. So that in this way place the object that shows the effect of contrast

-Camera: Drag the camera that you want to put on the lighting effect

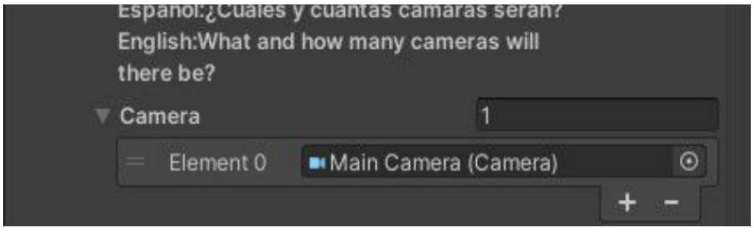
-Contrast Scroball: You drag the scrollbar component so you can regulate and control the Intensity

-Contrast float: It is to give an initial intensity



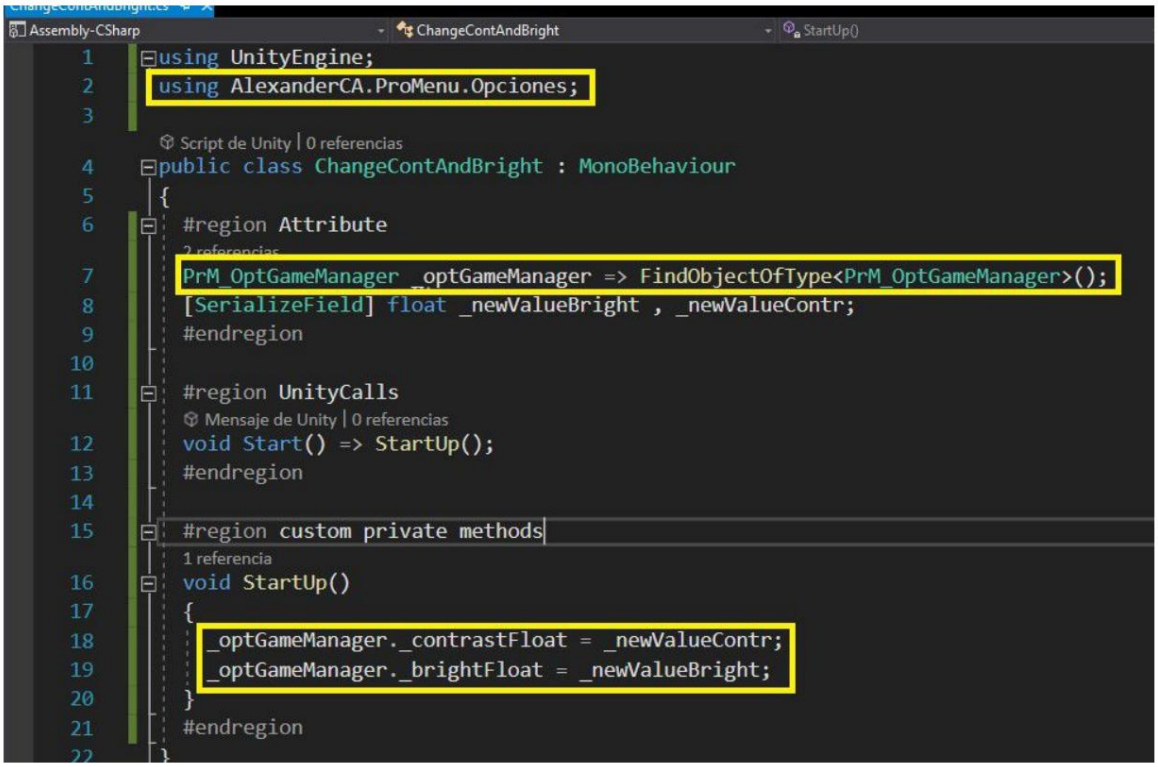
3) Brightness and contrast effect without controller: In the event that a scene does not have a scrollbar to control the intensity of contrast or brightness, nothing happens, you just have to pass the camera that interests you and with that it will be more than enough to have the effect of brightness and contrast, depending on the information that is transferred from scene to scene by DataPersisten

-Camera: Drag the panel that you want the lighting effect to be on



WARNING

If you wanted to modify the value of the general lighting float option for whatever reason, just reference the script “PrM_OptGameManager” and put the value you want in the float “_contrastFloat” and “_brightFloat”



OPTION 03

In this option you will indicate which parts of the graphic section you want to enable so that the player can configure them to their liking so that the PC they have performs better in the game or that they can enjoy the maximum experience that the game can provide.

-You have **six** parts:

– Explanation: The documentation of this option is summarized and concise.

– Content: It will contain all the elements you need to be able to use it.

-How Apply Changes: Here you will say how you want these changes to be made

-Change Button: To be able to apply the new changes when clicking this button

-Reset Button: Here you can reset all the options to the basic one

-Automatic: Here you will say if you want these changes to be made automatically

-Vsync: Here you can activate the vertical synchronization, it helps to give stability when synchronizing the the frame rate of the game or application image with the refresh rate of the monitor.

-Shadow Quality: Here you will indicate what type of quality you want the shadows to have in the game

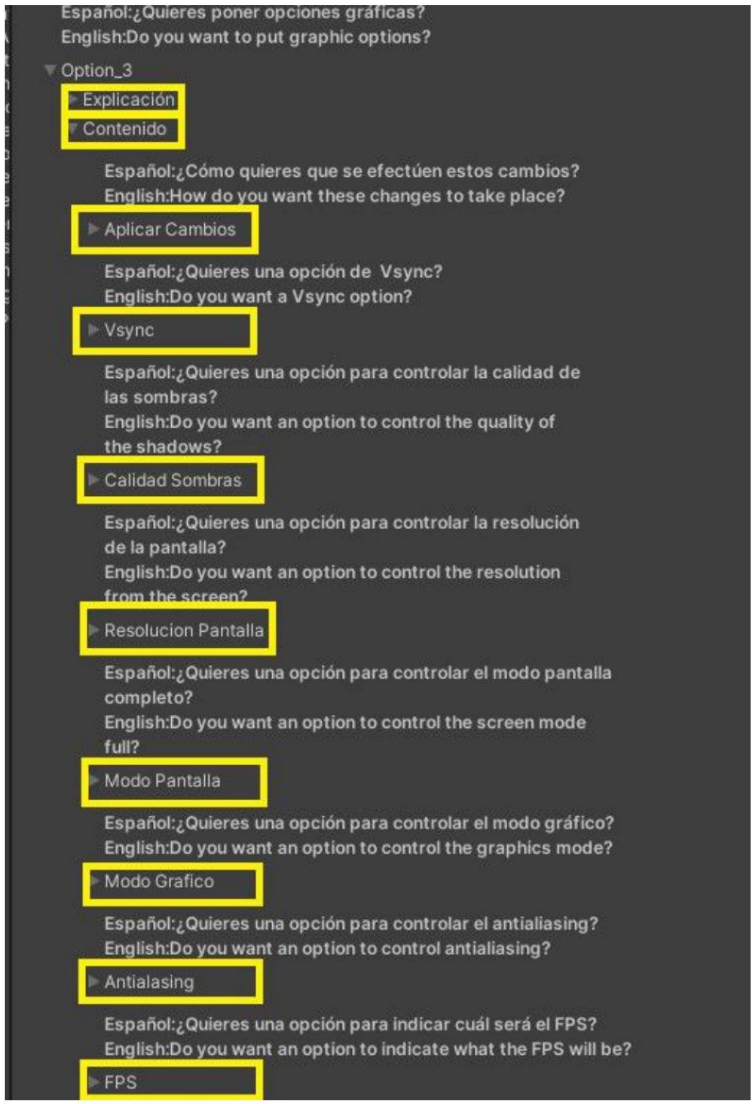
-Screen Resolution: Here you will indicate what type of resolution you want the screen to adapt to watch the game

-Windows Mode: Here you can activate if you want the player to be able to play the game in window

-Grafic: Here you can indicate which graphic you want to adopt in a general way

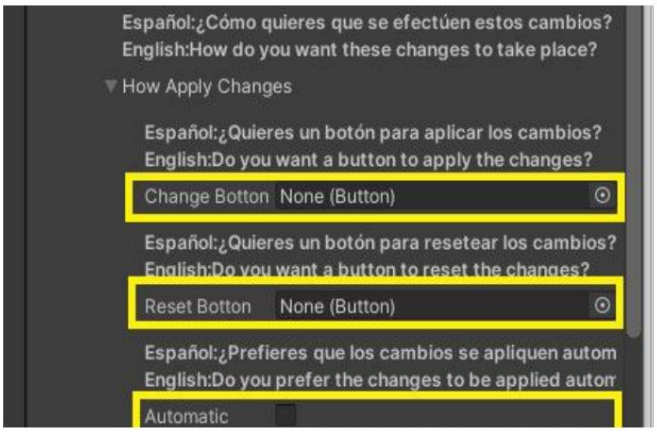
-Antialasing: Here you can indicate what type of antialsing you want the game to have to reduce distortions and graphic artifacts that appear in a high resolution image when it is presented at a lower resolution and vice versa

-FPS: Here you can indicate how much the refresh rate of the number of frames will be, you want the game to have or that the player can indicate that it has



USE

1) You can indicate how you want these changes to be made, if you want it to be automatic or by means of a button. You can only have one active option, and you can combine it with the reset

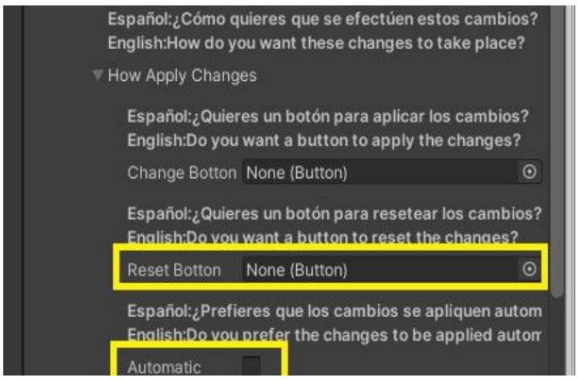


If you want it to be automatic and with a button to reset the options, you just have to select "HowApplyChange" and pass which button will perform the reset

-HowApplyChanges: Here you can say if you want the changes to be made automatic

-Automatic: You will select this option so that the change is automatic

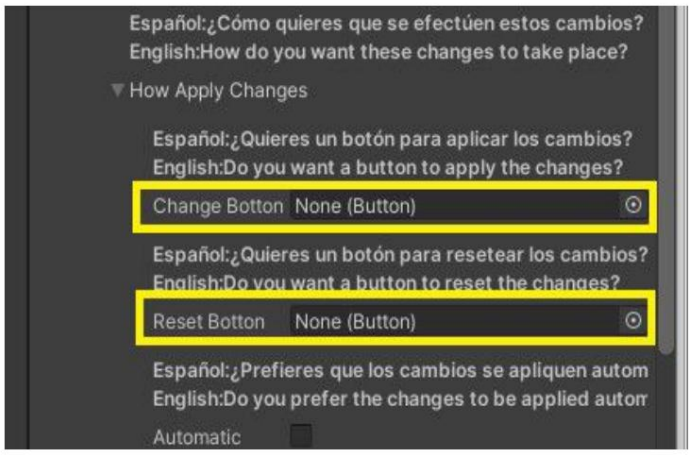
-ResetButton: Drag which button you want to have to reset the options



Or if you want the changes to be applied by means of one more button with the reset

-Change Button: Drag the button that you want to do the function of applying the new changes by button when clicked

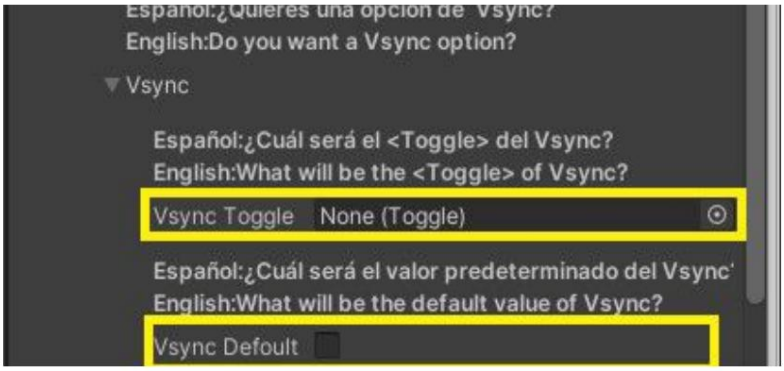
-ResetButton: Drag which button you want to have to reset the options



2) Vsync, to enable this option you only have to pass which will be the toggle that will control this option and also indicate which will be the default option of the system

-Vsync Toggle: Here you drag which toggle will control the activation and deactivation of Vsync

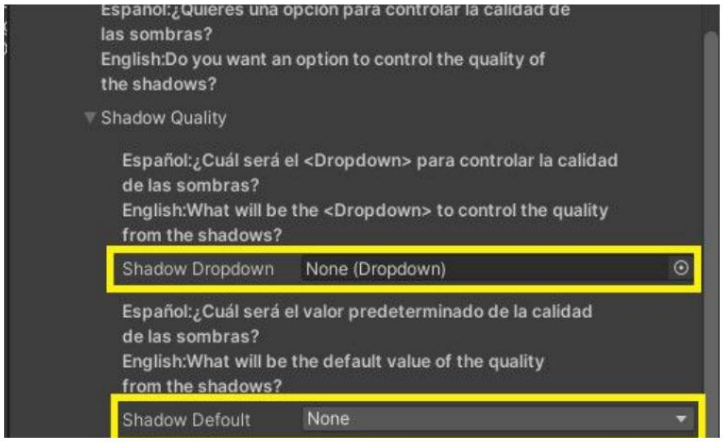
-Vsync Defoult: You can say what will be the default vsync option



3) Shadow Quality, you can control what kind of detail you want to show the shadows on the scenarios, you will have to pass two data, what will be the Drowpdown to show the different qualities and the type of what will be the default quality

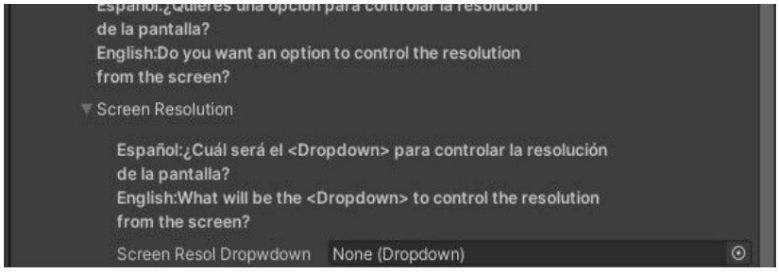
-Shadow Drowpdow: Here you drag the drowpdow that will control the quality of the shadows

-Shadow Default: You can say what the default shadow option will be.



4)Screen Resolution: You can modify the screen resolution to one that is suitable for the monitor, in order to have a wide experience

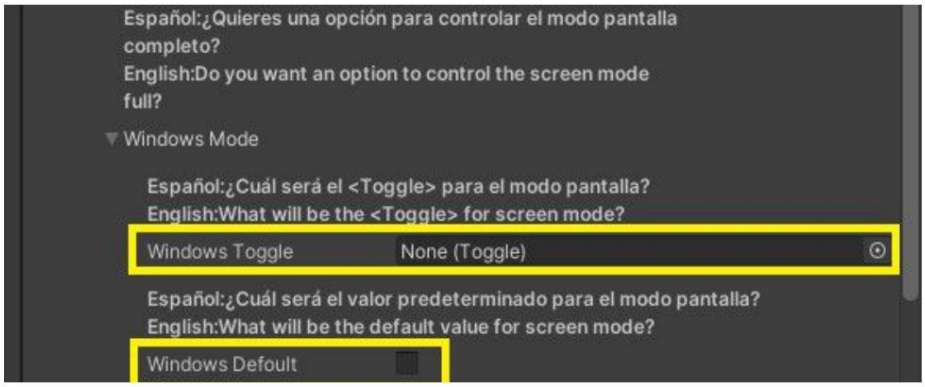
-ScreenResolDropdown: Here you drag that drowpdow that will control the resolution of the screen



5)Windows Mode : You can enable the option to activate the screen mode of the game, increase performance, by reducing the resolution, which means that the hardware does not have to represent as many details or simply a larger image, you will have to spend two data, what will be the Toggle to activate the option and indicate what will be your default

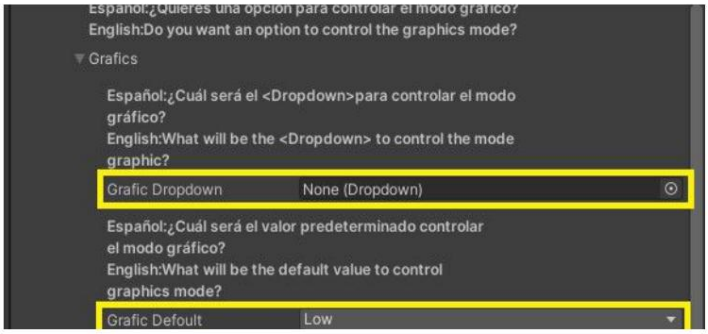
-WindowsToggle: Here you drag the toggle that will control or activate this mode

-Windows Default: You can say what will be the default option

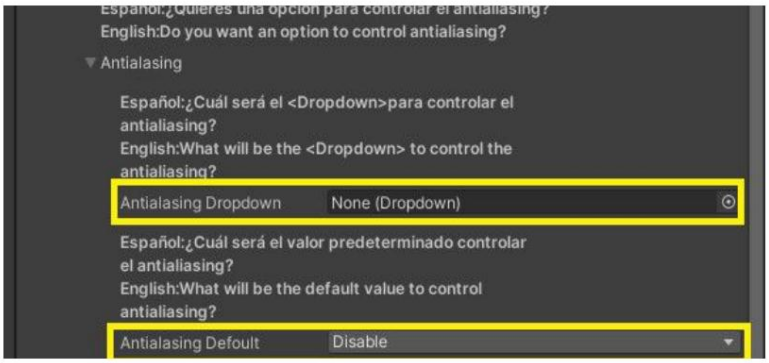


6) Graphic, you can control what type of detail you want to show the graphics in the game, you will have to pass two data, what will be the Drowpdown to show the different qualities and the type of which will be the default quality

-Grafic Drowpdow: Here you drag the drowpdow that will control the quality of the graphic
-Grafic Default: You will be able to say what the default shadow option will be.

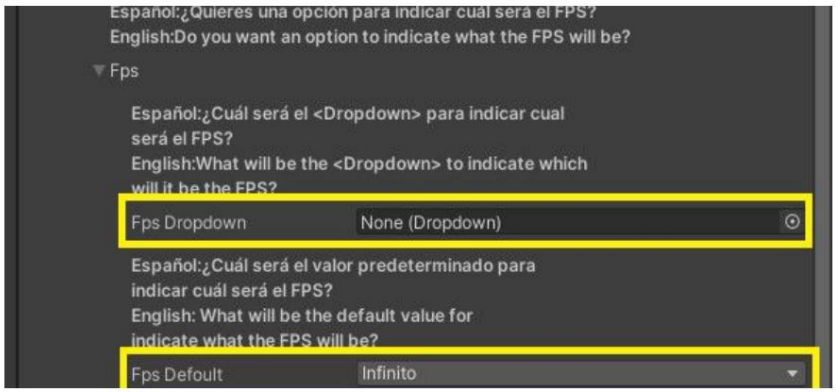


7) Anti-alasing, you can control what type of anti-alasing quality you want to be displayed regarding the shadows on the scenes, you will have to pass two data, what will be the Drowpdown to show the different qualities and the type of what will be the default quality - Antialasing -Drowpdow: Here you drag the drowpdow that will control the quality of the shadow -Antialasing Default: You will be able to say what the default shadow option will be.



8) FPS, you can control the amount or speed of the refresh rate of the images during the game, you will have to pass two data, what will be the Drowpdown to show the different qualities and the type of which will be the default quality

-Fps Drowpdow: Here you drag the drowpdow that will control the quality of the shadows
-Fps Default: You can say what the default shadow option will be.



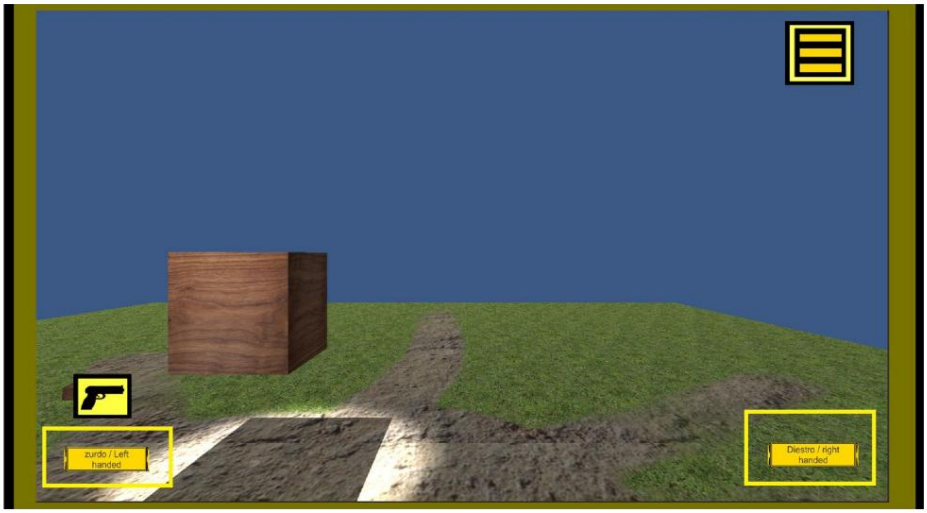
WARNING

Keep in mind that like all options, the information will pass between different scenes, but the system will not save those values once the game is closed, if you are interested in saving, recovering and placing them, the system does not do that, you will have to implement that part you, but what the system does is pass said information between levels and you can also overwrite said data so that it is updated with the correct ones. The steps to do are the following

Reference the script optionsManager of the system and then indicate what will be the variable to modify and then implement that value

OPTION 04

This section is intended more for platforms in the mobile gaming sector and to create the opportunity to change controllers for people who are not right-handed and are more left-handed, making it easier to change the controller panel, depending on the option chosen by the player. , it does not move the buttons but turns on and off the UI panels that are associated with the left or right controllers

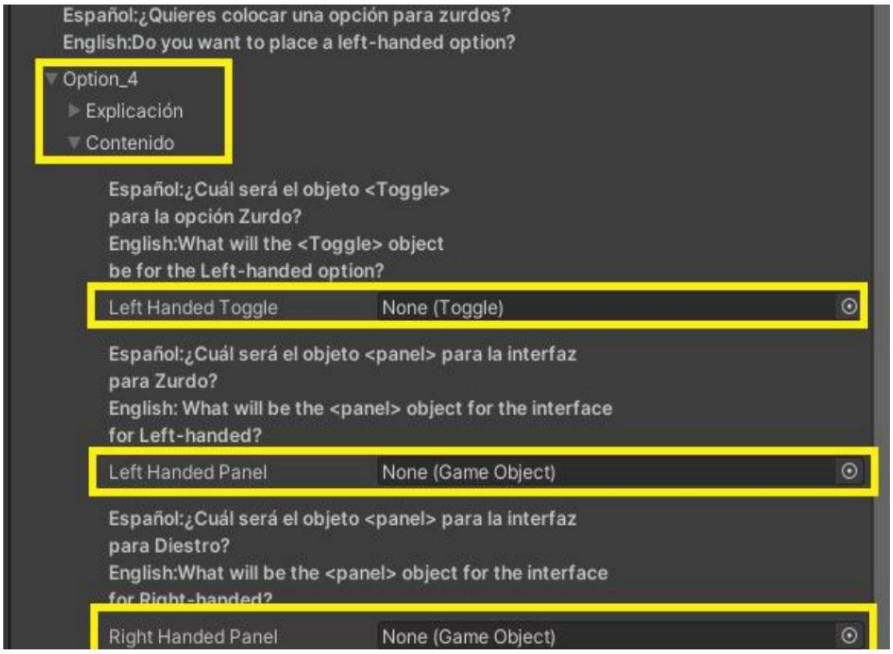


-You have **four** parts:

– Explanation: The documentation of this option is summarized and concise.

– Content: It will contain all the elements you need to be able to use it.

-LeftHandedToggle: Here you will place the togge to activate and deactivate -LeftHandedPanel: Here is to pass the panel that will contain the left-handed controllers -RightHandedPanel: Here is to pass the panel that will contain the right-handed controllers



USE

1) Basic use: To control the left-handed option you have to drag, three things, the panel of each type of control for left-handed and right-handed and the toggle that will indicate which panel must be active.

-LeftHandedToggle: Drag the toggle to indicate which panel to have activated

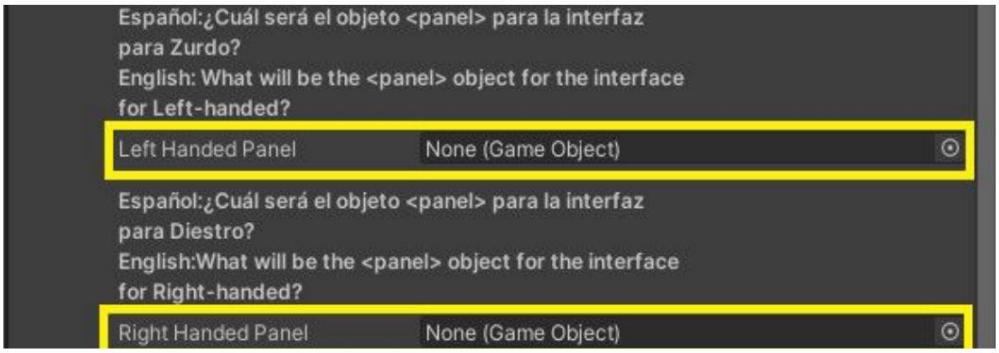
-LeftHandedPanel: You drag the panel that will have the controls for left-handed people

-RightHandedPanel: You drag the panel that will have the controls for right-handed people

2) Left-handed effect without controller: In the event that in a scene you do not have the toggle to give the option of change the panels, but you want this one, nothing happens, you just have to pass the panels and with that it will be more than enough to have the effect, depending on the information that is transferred from scene to scene by DataPersisten

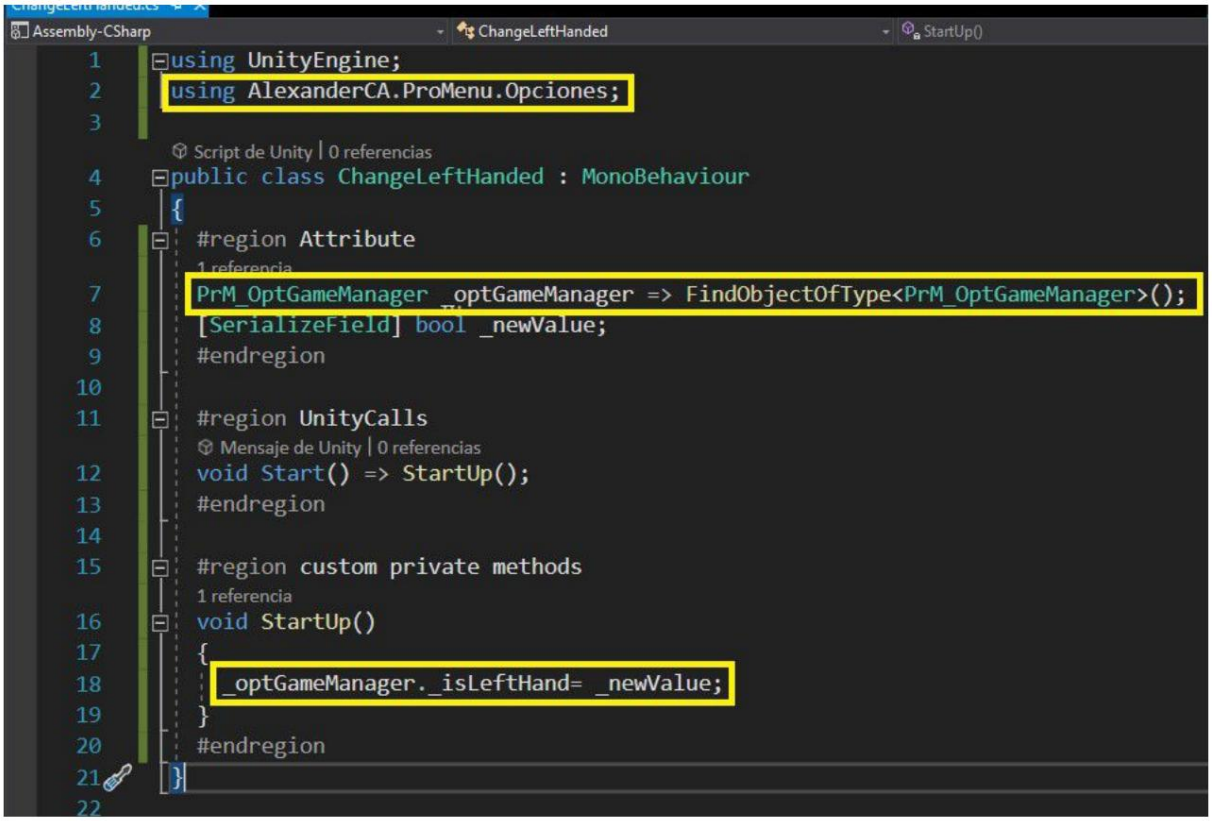
-LeftHandedPanel: You drag the panel that will have the controls for left-handed people

-RightHandedPanel: You drag the panel that will have the controls for right-handed people



WARNING

If you wanted to modify the value of the panels toggle for whatever reason, you just have to reference the script “PrM_OptGameManager” and put the value you want in the bool “_isLeftHanded”



PRM_DATAPERSISTEN

This script is designed to persist certain data between different scenes, so that the user does not have to deal or worry about passing all the information from the system, it will be done automatically and each manager will update the data of this persistence ignoring the information that is found. previously placed in said scene, so that it has the corresponding and correct data

USE

You do not have to do anything at all, or drag, or have to fill in any data, however this script modifies the manager values, but that does not remove or should not even interfere with the user's personal script that also modifies the data of the managers In general, what this script does is so simple, the first time it is executed, it reads the data of the managers and updates its information or every time the data of the managers is different from that of DataPersist, in this way we make it always up to date with the correct values. When going to a scene, DataPersist passes its information to the managers regardless of what they have on so that it is updated with the correct data and the update of the DataPersist would be executed again, every time the values of the manager are different from the ones. own DataPersist

WARNING

The function of this script is designed to save data while playing and passing between scenes, what it does not do is store that information when the game is closed, save data and load it once the game ends or when it starts , that is already the user's task, the only thing that this script is in charge of is passing data between scenes.

PRM_LOADGAMEMANAGER

Like every important script, it has two essential parts, which are the following sections:

3) Explanation: It will explain in a general way everything you can do in this script, divided into <Explanation> <Use> <Warning>

4) Content: Here you can give functionality to the component that interests you that adopts said function, each question also has its <Explanation> <Use> <Warning> in case you need more information about it

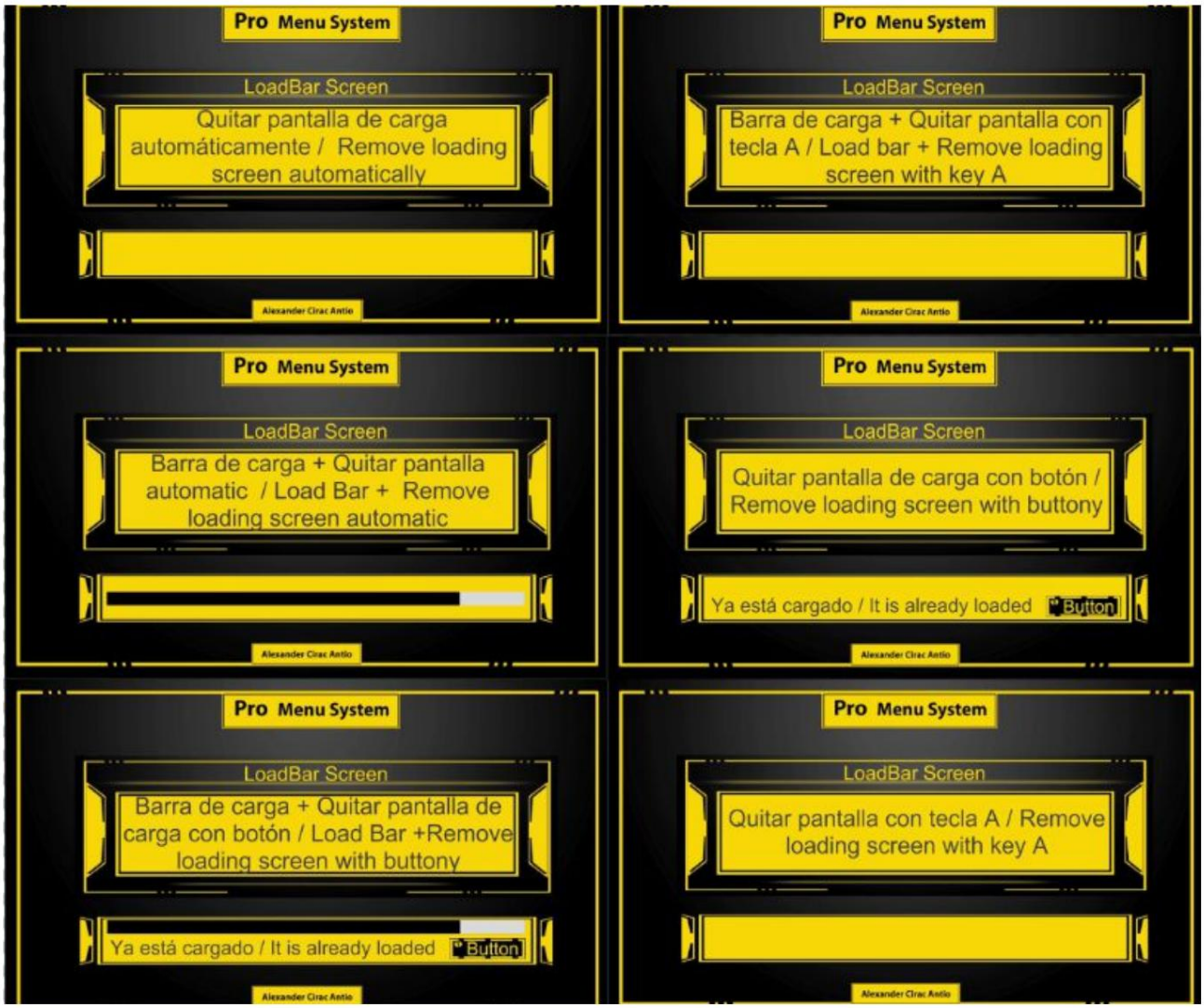
GENERAL EXPLANATION

This script is designed to be able to create a loading screen interface to give more dynamism and make level loading more enjoyable, providing utilities to the needs to make it more complete and useful.

The option that this section allows is the following:

- Option 0: Here you will indicate how you want said scene to behave with respect to the screen of burden

Each section will be indicated with a question so that the user himself, if he wants this level, to contain said function and assign the component of his choice



USE

Each question has a dropdown that contains an explanation of what it does and also its content, which will be to place and drag the desired objects in the gaps to give it functionality.
You can fill in those that interest you and in the event that you are missing an important parameter, the system will notify you.

I repeat, there is only one option to customize the loading screen, below it will be explained in more detail about this, both what you can do and what you cannot do

- Option 0: Here you will indicate how you want said scene to behave with respect to the loading screen

WARNING

Remember that the text content is the graphic part; it is not created by the script. It just makes handling easier. Example: the control of the panels when it comes to showing and which panels to hide, not referring to the graphic part or content, does not create them. If an important element is missing without placing it and/or there are more elements, position when only 1 is allowed, the system will also notify

OPTION 0

This option is to be able to place in a scene the one that will be the loading screen or the one that will receive the loading screen while it is being rendered, there are multiple options and combinations

-You have **twelve** parts:

– Explanation: The documentation of this option is summarized and concise.

– Content: It will contain all the elements you need to be able to use it.

-Use: It is to indicate what type of use you want to give to the scene, if it will be the scene that loads the loading screen or that scene where you will customize the loading screen -LoadBarScene Here you will indicate with an enumeration where the scene will be in which you customized previously the loading screen -SceneType: Here you will indicate if the scene you are currently in, you want it to be the loading screen itself, where you will customize it to your liking, or if this current scene is the one that will have to receive the loading screen so that can render the entire scene while the loading screen lasts so that later the player can play quietly -Costumization: Here you will have to say what elements you want the loading screen to have according to its different phases

-Avancing: Here you will indicate how you want this screen advance to be displayed (you can only choose one option)

-Is Loading Automatic: Here it will indicate that the advance will be automatically without a visualization by means of a bar burden

-SliderOption: Here it will indicate that the progress will be through the display of a <Slider> that will be the loading bar

-Warning: Here you will indicate what types of warning you want to give the player when he has finished loading the scene (here you can mix both options)

-WarningPanel: Here you will put all the elements that you want to It will indicate that the loading screen has finished, such as putting texts, buttons, images, animations

-WarningSound: Here you can drag an audio Source with the sound of alarm or whatever you want to be able to notify the player that the screen has loaded correctly

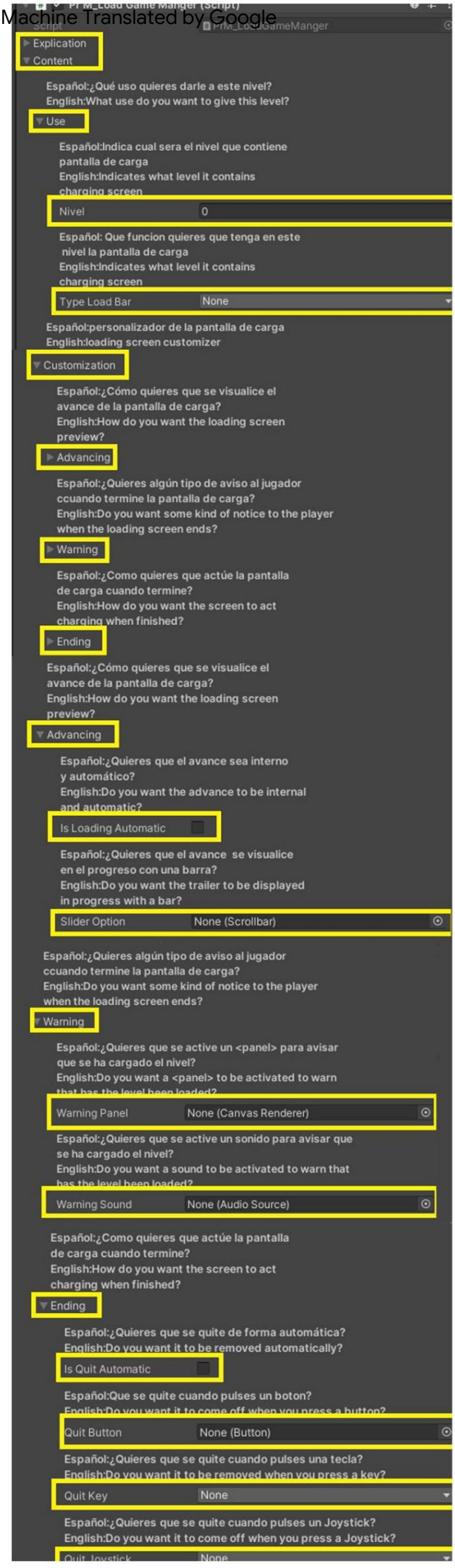
-Ending: Here you will indicate how you want to remove this loading screen when the other scene has finished rendering all its elements (you can only select automatic or activate the last 3 and/or combine them between the last 3, but automatic cannot be combined with any option)

-IsQuitAutomatic: Here it will indicate that when the rendering of the scene is finished, the screen load will be automatically removed

-QuitButton: Here it will indicate that when the rendering of the scene is finished, it will be removed when a button is pressed

-QuitKey: Here it will indicate that when the rendering of the scene is finished, it will be removed when a key is pressed

-QuitJoystick: Here you will indicate that when the rendering of the scene is finished, it will be removed when you press a button on the joystick



USE

1)} Loading screen automatically without a slash: , but playing a sound (is
optional) that notifies the user that it is already loaded and that this scene closes automatically

-IsLoadingAutomatic: clicking only this option so that everything is loaded automatically
-WarningSound: Drag the audio source that has the sound on to warn the user that the scene has
finished loading
-IsQuitAutomatic: Clicking only this option will automatically remove the loading screen when you finish
rendering the image.
scene

2) Loading screen automatically with a bar and sound: (it is optional) to notify the
user that is already loaded and that this scene closes automatically -sliderOption: Drag the slider
that you want the progress to be displayed
-WarningSound: Drag the audio source that has the sound on to warn the user that the scene has
finished loading
-IsQuitAutomatic: clicking only this option to remove the loading screen from
automatically when finished rendering the scene

3)Loading screen that closes when you press a button without a bar and a sound (it is optional) :to warn the user that it is already loaded (the content of the warningPanel is where you will put the text and the button)

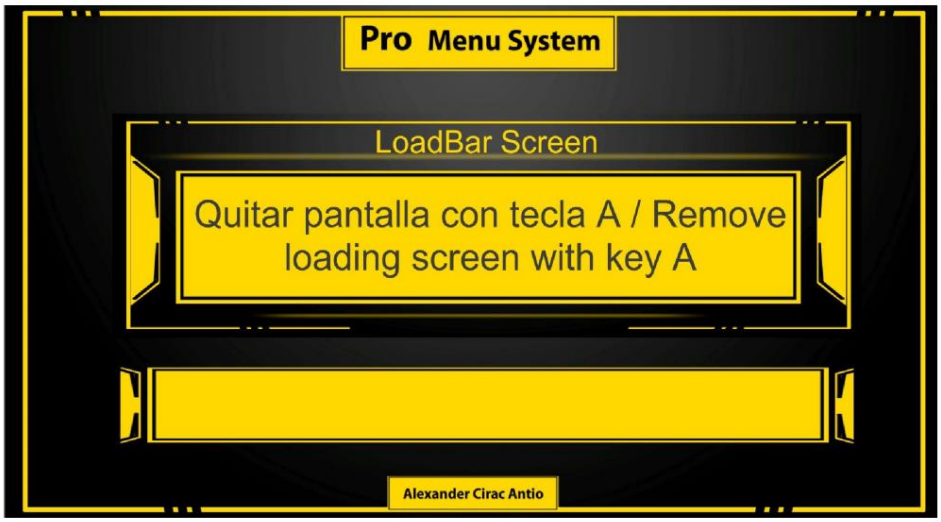
-IsLoadingAutomatic: clicking only this option so that everything is loaded automatically
-WarningSound: Drag the audio source that has the sound on to warn the user that the scene has finished loading
-WarningPanel: Drag the panel that contains the button and/or if you want with the text that will warn the user that the load has already finished
-IsQuitButton: Dragging the button that you want to have the function of removing the screen from burden

4) Loading screen that closes when you press a button with a bar and sound (optional): to notify the user already loaded

-sliderOption: drag the slider you want the trailer to be displayed
-WarningSound: Drag the audio source that has the sound on to warn the user that the scene has finished loading
-WarningPanel: Drag the panel that contains the button and/or if you want with the text that will warn the user that the load has already finished
-IsQuitButton: Dragging the button that you want to have the function of removing the screen from burden

5) Loading screen that closes when you click a key without a bar and a sound (optional): to notify to the user who is already loaded

-IsLoadingAutomatic: clicking only this option so that everything is loaded automatically
-WarningSound: Drag the audio source that has the sound on to warn the user that the scene has finished loading
-WarningPanel: Drag the panel that contains the text that will warn the user that the load has already finished
-IsQuitKey: Select which key you want to have the function of removing the loading screen
-IsQuitJoystick: Select which Joystick you want to have the function of removing the screen from burden

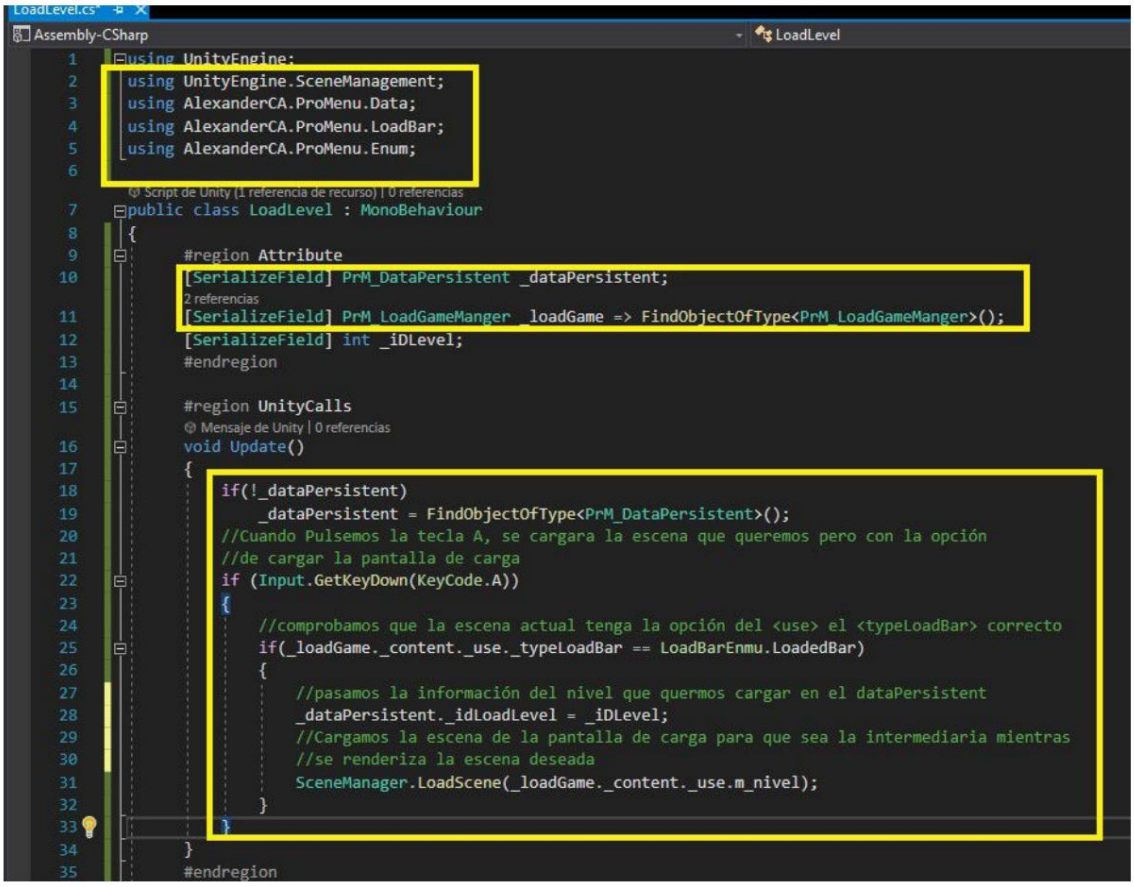


5) Loading screen that closes when you click a key with a bar and a sound (optional): to notify the user that it is already loaded

-SliderOption: Drag the slider that you want the preview to be displayed
-WarningSound: Drag the audio source that has the sound on to warn the user that the scene has finished loading
-WarningPanel: Drag the panel that contains the text that will warn the user that the load has already finished
-IsQuitKey: Select which key you want to have the function of removing the loading screen
-IsQuitJoystick: Select which Joystick you want to have the function of removing the screen from burden

WARNING

Note that if a scene is to receive a loading screen while it is rendering, set
<Use> in the <Type LoadBar> option of Loaded Bar", because if not, what will happen is that it crashesUnity
If you have to load a scene X with some of your scripts and you want it to also load the loading screen, 1) you
have to refer to the PrM_DataPersistent, pass it the scene number you want
2) And in your script, you will load the scene where the loading screen is located and it will be in charge of
loading the scene you want, having put it in PrM_DataPersisten, since the loading screen is only an intermediary
for that desired scene



If you are missing any component or, if you put more than one option as in <Use>, the system will notify you and it will not load scenes