Doc Part 3: Integrate puzzle in your project

Table of contents:

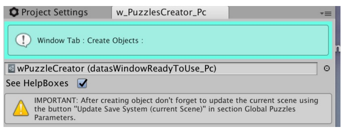
Part 3: Read First

This section will help you find the information you are looking for in Doc Part 3.

Window tab: w_PuzzlesCreator

The window w_PuzzleCreator allows:
-To generate puzzles in your project.
-To manage global puzzles settings (Inputs choose, Save format)
-To create basic elements when you integrate the puzzle system into an existing project.

To access that window:
-Go to Tools → Puzzles → Puzzles Creator (w_PuzzlesCreator)



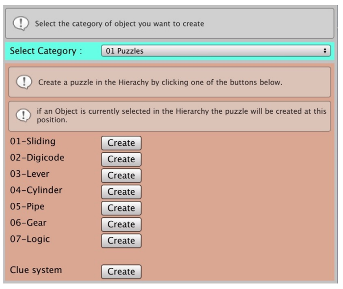
Window Tab : Create Objects :

wPuzzleCreator (datasWindowReadyToUse_Pc)

IMPORTANT: After creating object don't forget to update the current scene using the button "Update Save System (current Scene)" in section Global Puzzles Parameters.

This section allows to create puzzles:

(More info about puzzle creation in Doc Part 2)



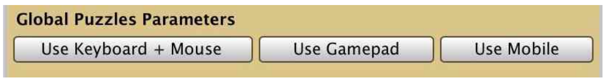
Select the category of object you want to create

Create a puzzle in the Hierachy by clicking one of the buttons below.

if an Object is currently selected in the Hierarchy the puzzle will be created at this position.

Global Puzzles Parameters:

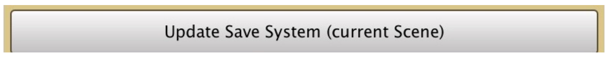
This section allows to switch between the different Puzzle input types.
You can choose input type when the game starts.



It is possible to choose between:
-Keyboard + Mouse
-Gamepad
-Mobile

This button allows to choose if the save system use .Dat format or PlayerPrefs format.

The button Update Save System (current Scene) allows to update the save system after adding a new puzzle in the a scene.

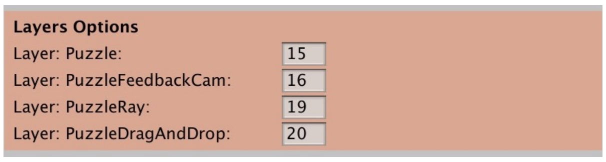


Show .Dat In Explorer open the folder that contains the save files in your computer.

Delete all save datas (all Scenes) delete all the save datas in the project.

Layer Options:
This section is useful if you want to change the default layers used by the puzzle system.

(more info about how to change a layer in Doc Part 1 Tuto 2 Step 3: Setup Layers)

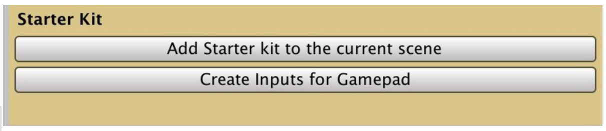

Layers Options

Layer: Puzzle:	15
Layer: PuzzleFeedbackCam:	16
Layer: PuzzleRay:	19
Layer: PuzzleDragAndDrop:	20

Starter Kit:

Add Starter kit to the current scene:

This button is used when you want to integrate the puzzle system inside your project. *(more info about how to use the starter Kit in Doc Part 1 Tuto 2 Step 4: Setup the starter kit)*


Starter Kit
Add Starter kit to the current scene
Create Inputs for Gamepad

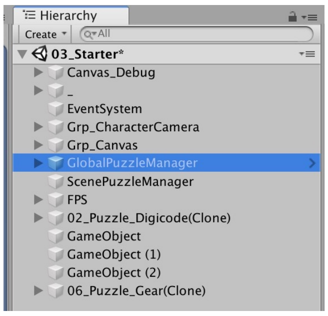
Create Inputs for Gamepad:

If you want to use gamepad in your game:
This button create two new inputs in the Inputs Manager.
The new inputs manage the gamepad mouse cursor (Vertical and Horizontal axis) in a puzzle using Focus Mode.

Global Puzzle Manager:

Overview:

The GlobalPuzzleManager:
-Manage the player inputs when the player interacts with a puzzle.
-Manage what happens when a player enters or leaves a puzzle.

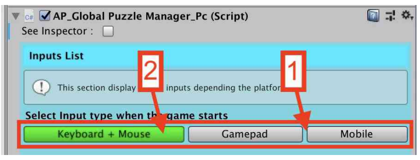


For example:
-Block the movement of the character when he enters a puzzle.
-Allows the character's movement when he comes out of a puzzle.
-Activate an icon, a sound when the character enters a puzzle
-Disable an icon, a sound when the character comes out of a puzzle.
-If the player holds a weapon in his hand, disable that weapon when the character enters a puzzle.
You can add your own conditions to fits your game.

Important:
You must have only one globalPuzzleManager in your scene. This object is not destroy when a new scene is loaded.
Do not rename this object in the Hierarchy.

Manage puzzle inputs:

In section Input List you can choose input type when the game starts (spot 1).


AP_Global Puzzle Manager_Pc (Script)
See Inspector :
Inputs List
This section display
inputs depending the platfor
2
1
Select Input type when th game starts
Keyboard + Mouse
Gamepad
Mobile

It is possible to choose between:
-Keyboard + Mouse
-Gamepad
-Mobile

The selected input type is green (spot 2).
*(**Info**: In the example Keyboard + Mouse is the selected input when the game starts)*

Scripting It is possible to change the Input type at runtime.

For Keyboard + Mouse call:

AP_GlobalPuzzleManager.instance.
AP_SwitchPuzzleInputsToKeyboardAndMouse(true);

For Gamepad call:

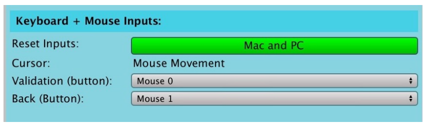
AP_GlobalPuzzleManager.instance.
AP_SwitchPuzzleInputsToGamepad(true);

For Mobile call:

AP_GlobalPuzzleManager.
instance.AP_SwitchPuzzleInputsToMobile();

Manage Keyboard + Mouse inputs:

In section Keyboard + Mouse Inputs you can choose the input when the player press:
-The button to validate an action in a puzzle.
-The button to leave a puzzle using Focus mode.


Keyboard + Mouse Inputs:

Reset Inputs:	Mac and PC
Cursor:	Mouse Movement
Validation (button):	Mouse 0
Back (Button):	Mouse 1

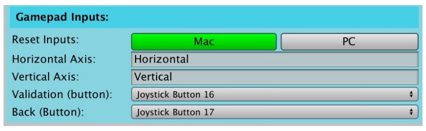
Scripting It is possible to change the Input type at runtime.

-Validation (button): (**KeyCode**). Call:
AP_GlobalPuzzleManager.instance.validationButtonKeyboard

-Back (button): (**KeyCode**). Call:
AP_GlobalPuzzleManager.instance.backButtonKeyboard

Manage Gamepad inputs:

In section Gamepad you can choose the inputs when the player:
-Move the cursor Horizontally
-Move the cursor Vertically
-Press the button to validate an action in a puzzle.
-Press the button to exit a puzzle on Focus mode.


Gamepad Inputs:

Reset Inputs:	Mac	PC
Horizontal Axis:	Horizontal	
Vertical Axis:	Vertical	
Validation (button):	Joystick Button 16	
Back (Button):	Joystick Button 17	

Scripting It is possible to change the Input at runtime.

-Horizontal Axis: (**string**). Call:
AP_GlobalPuzzleManager.instance.HorizontalAxisJoystickLeft

-Vertical Axis: (**string**). Call:
AP_GlobalPuzzleManager.instance.VerticalAxisJoystickLeft

-Validation (button): (**KeyCode**). Call:
AP_GlobalPuzzleManager.instance.validationButtonJoystick

-Back (button): (**KeyCode**). Call:
AP_GlobalPuzzleManager.instance.backButtonJoystick

Call methods when puzzle start or stop:

It is possible to call custom methods when a puzzle starts or stops.

For example: *When a puzzle starts using focus mode the character must stop moving in the scene. Then when the player exit the puzzle the character needs to move again.*

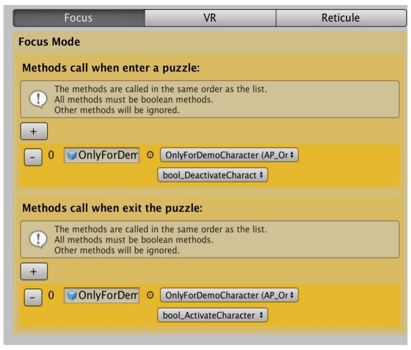
How to do that?
-In the section Methods call when enter a puzzle:
Add a custom method that stop the player movement.

-In the section Methods call when exit a puzzle:
Add a custom method that allows the player to move

Important:
-The methods called need to be boolean methods.
Other methods will be ignored.

-You can add more than one boolean method.

Focus Tab:
Methods used when the player enter or exit a puzzle in Focus Mode



Focus Mode

Methods call when enter a puzzle:

The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.

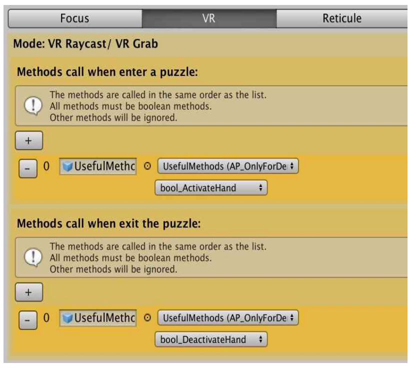
0 OnlyForDem OnlyForDemoCharacter (AP_Or
bool_DeactivateCharact

Methods call when exit the puzzle:

The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.

0 OnlyForDem OnlyForDemoCharacter (AP_Or
bool_ActivateCharacter

VR Tab:
Methods used when the player enter or exit a puzzle in Mode: VR Raycast or VR Grab



Mode: VR Raycast/ VR Grab

Methods call when enter a puzzle:

The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.

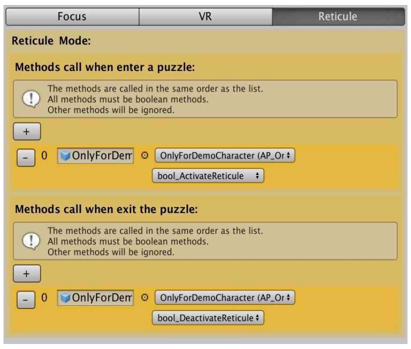
0 UsefulMethc UsefulMethods (AP_OnlyForDe
bool_ActivateHand

Methods call when exit the puzzle:

The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.

0 UsefulMethc UsefulMethods (AP_OnlyForDe
bool_DeactivateHand

Reticule Tab:
Methods used when the player enter or exit a puzzle in Reticule Mode



Reticule Mode:

Methods call when enter a puzzle:

The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.

0 OnlyForDem OnlyForDemoCharacter (AP_Or
bool_ActivateReticule

Methods call when exit the puzzle:

The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.

0 OnlyForDem OnlyForDemoCharacter (AP_Or
bool_DeactivateReticule

Add new method in section Methods call when enter a puzzle or Methods call when exit a puzzle (Tuto)
Even though the sections are different, the process for adding methods remains the same.

In several sections of the puzzle system it is possible to add custom methods.

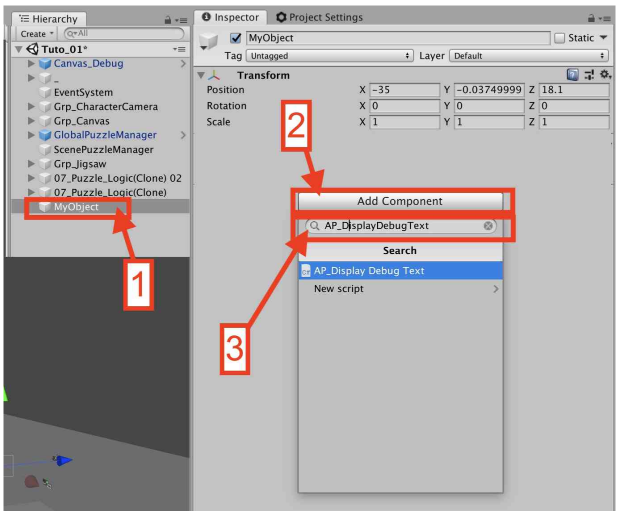
(*More Info: If you add a method for the first time and want more information have a look to the section How to add a method (overview) link*)

Caution:
Methods call when enter a puzzle:
-Only boolean methods are allowed.
Other methods are ignored.

Methods call when exit a puzzle:
-Only boolean methods are allowed.
Other methods are ignored.

In this example we are going to add a method in section Methods call when enter a puzzle in Focus Mode.
You can follow the same process for Methods call when exit a puzzle in Focus, VR, and Reticule Mode

Example: Add the Method bool_Display_Txt_01 that:
-return true.
-Display text "Activated" in the Console Tab.



-Create an empty object in the Hierarchy and rename it MyObject for example (spot 1).

-In the Inspector press button Add Component (spot 2)

-In search field write AP_DisplayDebugText (spot 3)

-Press keyboard key Enter to add the component.

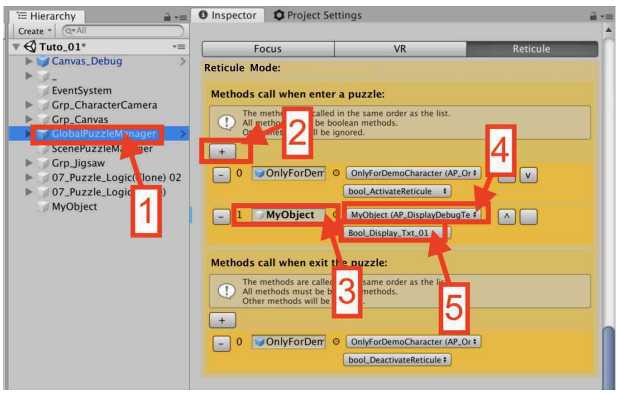
-In the Hierarchy select object GlobalPuzzleManager (spot 1).

-In the Inspector press button + in the section Method call when enter a puzzle to create a new empty slot (spot 2)

-Drag and drop MyObject in the empty slot (spot 3)

-Select the script AP_DisplayDebugText (spot 4)

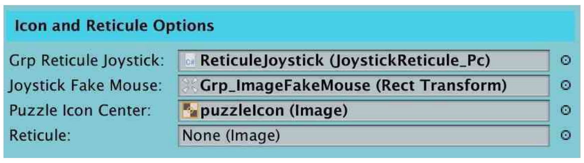
-Select the method AP_Display_Txt (spot 5)



Focus
VR
Reticule
Reticule Mode:
Methods call when enter a puzzle:
OnlyForDemoCharacter (AP_Or
bool_ActivateReticule
MyObject
MyObject (AP_DisplayDebugTe
Bool_Display_Txt_01
Methods call when exit the puzzle:
bool_DeactivateReticule
1
2
3
4
5

Icon and reticule Options

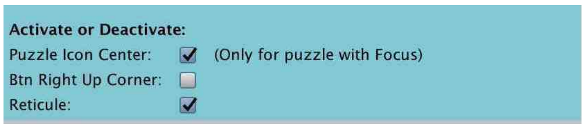
They are already setup. There is no reason to modify them.


Icon and Reticule Options

Grp Reticule Joystick:	ReticuleJoystick (JoystickReticule_Pc)
Joystick Fake Mouse:	Grp_ImageFakeMouse (Rect Transform)
Puzzle Icon Center:	puzzleIcon (Image)
Reticule:	None (Image)

Activate or deactivate:

This section allows to activate or deactivate UI elements when the player enter or leave a puzzle.


Activate or Deactivate:

Puzzle Icon Center:	✓	(Only for puzzle with Focus)
Btn Right Up Corner:	☐	
Reticule:	✓	

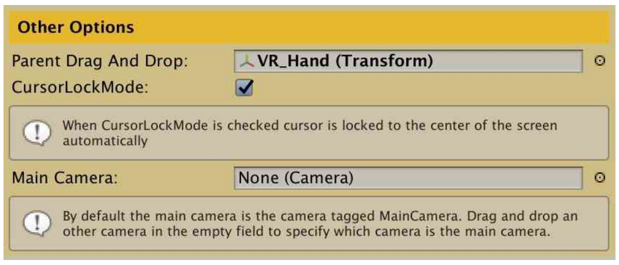
Puzzle Icon center: Allows to display a puzzle icon in the center of the screen when a puzzle is detected. This option works only on Focus Mode.

Btn Right Up Corner: Allows to display a puzzle icon in the right up corner of the screen when a puzzle is detected.

Reticule: Allows to activate the reticule when no puzzle is detected.

Other options:

Parent Drag And Drop is already setup. There is no reason to modify them.


Other Options

Parent Drag And Drop: VR_Hand (Transform)

CursorLockMode: ✓

When CursorLockMode is checked cursor is locked to the center of the screen automatically

Main Camera: None (Camera)

By default the main camera is the camera tagged MainCamera. Drag and drop an other camera in the empty field to specify which camera is the main camera.

CursorLockMode:
If the button is checked:
When the player leaves a puzzle the cursor is automatically locked to the center of the screen.

If the button is unchecked:
You can manage manually the cursor locked state when the player leaves a puzzle.
***Note**: Even if you manage manually the cursor lock state: The cursor is automatically set to confined when the player enter in a puzzle set to Focus Mode.*

Main Camera:
If your player camera is not tagged MainCamera:
Drag and drop your Player camera here

Scripting:Useful methods or variables that can be called at runtime:

-b_Pause (bool):
***Example**: If we don't want the player be able to interact with the puzzle when the game is paused.*
b_Pause allows to deactivated player interactions.

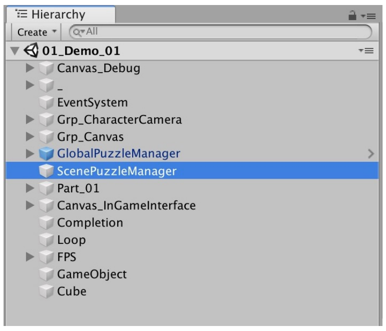
if b_Pause = true paused all the puzzle
if b_Pause = false interaction with puzzle is available

Scripting It is possible to change the b_Pause at runtime.
b_Pause: (call) AP_GlobalPuzzleManager.instance.b_Pause

ScenePuzzleManager:

Overview:

The scenePuzzleManager allows:
-To initialize the puzzles at the beginning of the scene.
-To save the puzzles present in the scene.



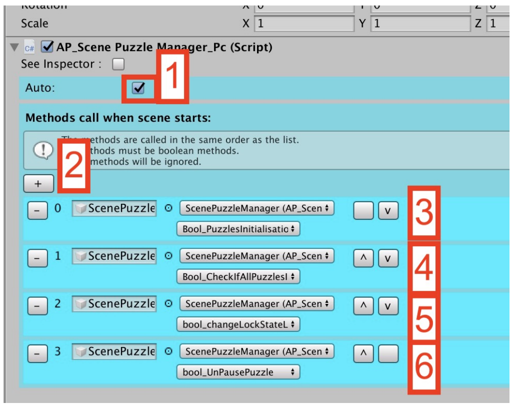
Hierarchy
Create
01_Demo_01
Canvas_Debug
_
EventSystem
Grp_CharacterCamera
Grp_Canvas
GlobalPuzzleManager
ScenePuzzleManager
Part_01
Canvas_InGameInterface
Completion
Loop
FPS
GameObject
Cube

For example:
-Load the puzzles settings present in the scene
-You can also put your own conditions.

Important:
You must have only one scenePuzzleManager in your scene.
Do not rename this object in the Hierarchy.

Methods call when scene starts:

This section is used to call the methods contained in methods when the scene starts.



Scale X 1 Y 1 Z 1
AP_Scene Puzzle Manager_Pc (Script)
See Inspector :
Auto: ✓
Methods call when scene starts:
The methods are called in the same order as the list.
...thods must be boolean methods.
...methods will be ignored.

0	ScenePuzzle	ScenePuzzleManager (AP_Scen	Bool_PuzzlesInitialisatio
1	ScenePuzzle	ScenePuzzleManager (AP_Scen	Bool_CheckIfAllPuzzlesI
2	ScenePuzzle	ScenePuzzleManager (AP_Scen	bool_changeLockStateL
3	ScenePuzzle	ScenePuzzleManager (AP_Scen	bool_UnPausePuzzle

By default:
The parameter Auto (spot 1) is checked (true):
(Methods contained in section Methods call when scene starts (spot 2) are called when the scene starts)

if Auto is unchecked methods contained in section Methods call when scene starts are not called when the scene starts.
It is possible to call Methods call when scene starts manually at runtime. Call the method:
StartCoroutine(CallAllTheMethodsOneByOne());
contained in the script AP_ScenePuzzleManager_Pc
attached to ScenePuzzleManager in the Hierarchy.

By default:
Methods call in section
methods when the scene starts are:

Method: Bool_PuzzleInitialisation()
-The puzzles are initialized (spot 3).

Method: Bool_CheckIfAllPuzzlesInitialized()
-Wait until all the puzzles are initialized (spot 4).

Method: Bool_ChangeLockStateLock()
-The Mouse cursor is locked to the center of the screen (spot 5).

Method: Bool_UnPausePuzzle()
-Unpause all the puzzles (spot 6).

***Info**: the following four methods are available in script AP_ScenePuzzleManager_Pc*
attached to ScenePuzzleManager in the Hierarchy

Add new method in section Methods call when scene starts (Tuto):

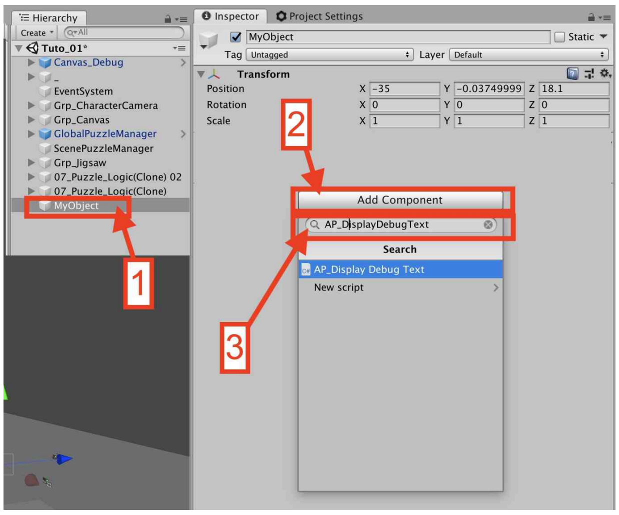
Even though the sections are different, the process for adding methods remains the same.

In several sections of the puzzle system it is possible to add custom methods.

(***More Info***: *If you add a method for the first time and want more information have a look to the section How to add a method (overview) link*)

Caution:
Methods call when scene starts:
-Only boolean methods are allowed.
Other methods are ignored.

Example: Add the Method bool_Display_Txt_01 that:
-return true.
-Display text "Activated" in the Console Tab when the scene starts.

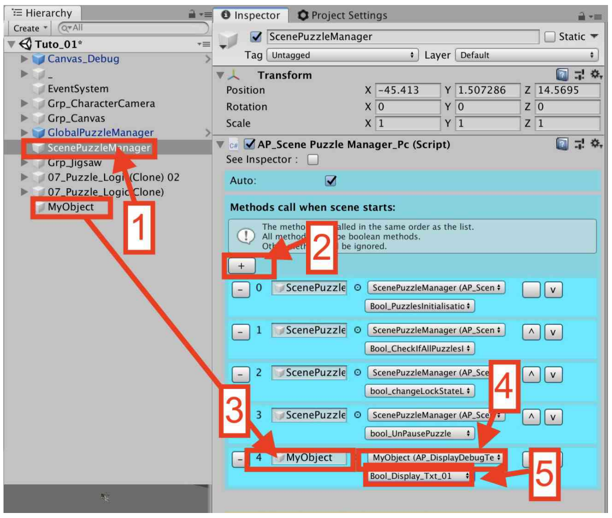


-Create an empty object in the Hierarchy and rename it MyObject for example (spot 1).

-In the Inspector press button Add Component (spot 2)

-In search field write AP_DisplayDebugText (spot 3)

-Press keyboard key Enter to add the component.


Hierarchy
Tuto_01*
Canvas_Debug
EventSystem
Grp_CharacterCamera
Grp_Canvas
GlobalPuzzleManager
ScenePuzzleManager
Grp_Jigsaw
07_Puzzle_Logic(Clone) 02
07_Puzzle_Logic(Clone)
MyObject
Inspector
Project Settings
ScenePuzzleManager
Static
Tag Untagged
Layer Default
Transform
Position X -45.413 Y 1.507286 Z 14.5695
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
AP_Scene Puzzle Manager_Pc (Script)
See Inspector :
Auto:
Methods call when scene starts:
+
0 ScenePuzzle ScenePuzzleManager (AP_Scen Bool_PuzzlesInitialisatio
1 ScenePuzzle ScenePuzzleManager (AP_Scen Bool_CheckIfAllPuzzlesI
2 ScenePuzzle ScenePuzzleManager (AP_Sce bool_changeLockStateL
3 ScenePuzzle ScenePuzzleManager (AP_Sce bool_UnPausePuzzle
4 MyObject MyObject (AP_DisplayDebugTe Bool_Display_Txt_01
1
2
3
4
5

-In the Hierarchy select object ScenePuzzleManager (spot 1).

-In the Inspector press button + in the section Method call when scene starts to create a new empty slot (spot 2)

-Drag and drop MyObject in the empty slot (spot 3)

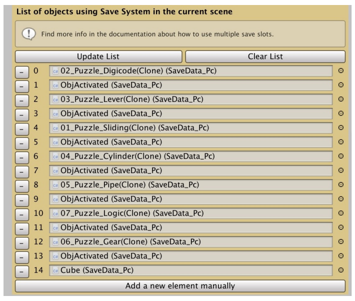
-Select the script AP_DisplayDebugText (spot 4)

-Select the method AP_Display_Txt (spot 5)

List of Objects using Save System in the current scene.

This section show the list of the object integrated to the save system.

There is nothing to do in this section



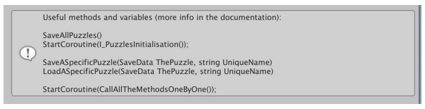
List of objects using Save System in the current scene

Find more info in the documentation about how to use multiple save slots.

0	02_Puzzle_Digicode(Clone) (SaveData_Pc)
1	ObjActivated (SaveData_Pc)
2	03_Puzzle_Lever(Clone) (SaveData_Pc)
3	ObjActivated (SaveData_Pc)
4	01_Puzzle_Sliding(Clone) (SaveData_Pc)
5	ObjActivated (SaveData_Pc)
6	04_Puzzle_Cylinder(Clone) (SaveData_Pc)
7	ObjActivated (SaveData_Pc)
8	05_Puzzle_Pipe(Clone) (SaveData_Pc)
9	ObjActivated (SaveData_Pc)
10	07_Puzzle_Logic(Clone) (SaveData_Pc)
11	ObjActivated (SaveData_Pc)
12	06_Puzzle_Gear(Clone) (SaveData_Pc)
13	ObjActivated (SaveData_Pc)
14	Cube (SaveData_Pc)

Scripting:Useful methods or variables that can be called at runtime:

It is possible to:



Useful methods and variables (more info in the documentation):

```
SaveAllPuzzles()
StartCoroutine(I_PuzzlesInitialisation());

SaveASpecificPuzzle(SaveData ThePuzzle, string UniqueName)
LoadASpecificPuzzle(SaveData ThePuzzle, string UniqueName)

StartCoroutine(CallAllTheMethodsOneByOne());
```

Save all the puzzle manually at runtime.

Call:
SaveAllPuzzles();
contained in the script AP_ScenePuzzleManager_Pc
attached to ScenePuzzleManager in the Hierarchy.

Initialized all the puzzles manually using the save datas at runtime.

Call:
StartCoroutine(I_PuzzlesInitialisation());
contained in the script AP_ScenePuzzleManager_Pc
attached to ScenePuzzleManager in the Hierarchy.

Save System overview

We will take an example:
-The player approaches a puzzle.
-He solves the puzzle.
-A door opens.
The player save his game progression.

How it works:

During save process, the state of the puzzle will be automatically saved.

On the other hand, the door is an element not included in the puzzle.
During the next save, the state of the door will not be saved.
To save the state of the door, it is necessary to use the save extension system.

In summary:

During save process, the state of a puzzle is automatically saved. This includes the position of the puzzle pieces and whether the puzzle is solved or not.

On the other hand, all elements not included in the puzzle must be saved with the save expansion system
(more info about how to save an object different from a puzzle Link).

Other info:

Each scene has its own save file.
If the door and the puzzle are in the same scene:
They will be saved in the same file.

If you want to manually load or save go to the section ScenePuzzleManager (script needed)
Link

ObjIsActivated

Overview:

In the documentation we use the name ObjIsActivated for:
-An object with the 2 specific scripts attached to it:
IsObjectActivated_Pc.cs and SaveData_Pc.cs

(More info to setup an Object as ObjIsActivated)

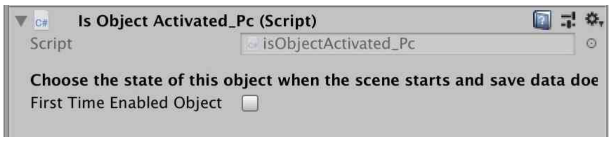
Why use ObjIsActivated:

For example:
Save that a door is open after solving a puzzle.
Save the position of an object.
Save if an object is enabled or disabled in the scene

ObjIsActivated allows you to save any type of parameters.

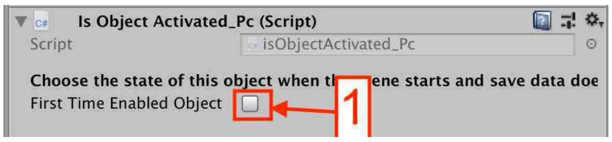
ObjIsActivated is separated in 3 parts:

First part:
We will take an example to explain this part:


Is Object Activated_Pc (Script)
Script: isObjectActivated_Pc
Choose the state of this object when the scene starts and save data doe
First Time Enabled Object ☐

There is a key on a table.
When the player grab this key, the key disappears (disable in the Hierarchy).
During the saving process, it saved that the object is disabled in the Hierarchy.
If the player reloads the scene, the save datas indicate that the object must be disabled in the Hierarchy.

Important:
When the player loads the scene for the first time:
It must be indicated if the key must be enabled or disabled in the scene.


Is Object Activated_Pc (Script)
Script: isObjectActivated_Pc
Choose the state of this object when th ene starts and save data doe
First Time Enabled Object ☐

If we want the key to be enabled:
Check FirstTimeEnabledObject (spot 1)

If we want the key to be disabled:
Uncheck FirstTimeEnabledObject (spot 1)

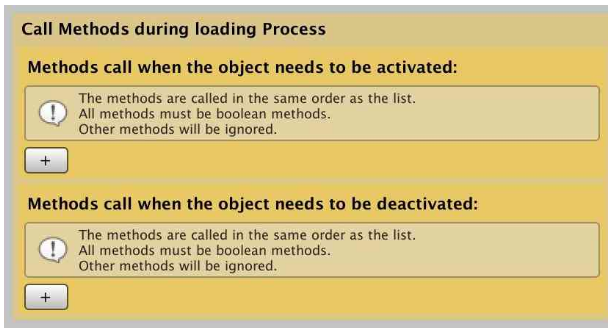
Note*: During the loading process*
If a meshRenderer is attached to your ObjIsActivated:
This mesh component is enabled/disabled in the Hierarchy.

*If **No** meshRenderer is attached to your ObjIsActivated:*
This gameObject is enabled/ disabled in the Hierarchy.

Second part:
During the loading process it is possible to call custom methods.

If the object has been saved as enabled during the save process:
The methods in section
Methods call when the object needs to be activated
are called.

If the object has been saved as disabled during the save process:
The methods in section
Methods call when the object needs to be deactivated
are called.
(More about adding your custom method here)


Call Methods during loading Process

Methods call when the object needs to be activated:

The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.

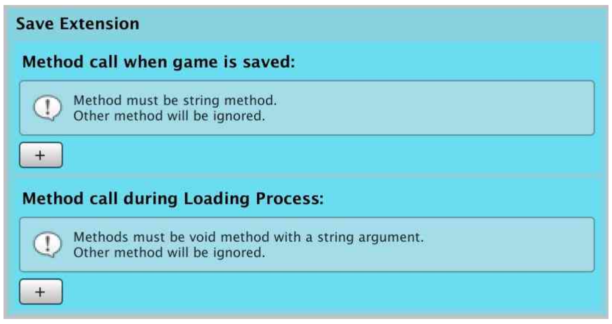
Methods call when the object needs to be deactivated:

The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.

Third section:
This section allows to save / load your custom settings.

For example:
-Save if a door is open / closed.
-Save if a padlock is open or closed.
-Save the position of an object.

(More about adding custom methods on section Save Extension)


Save Extension

Method call when game is saved:

Method must be string method.
Other method will be ignored.

Method call during Loading Process:

Methods must be void method with a string argument.
Other method will be ignored.

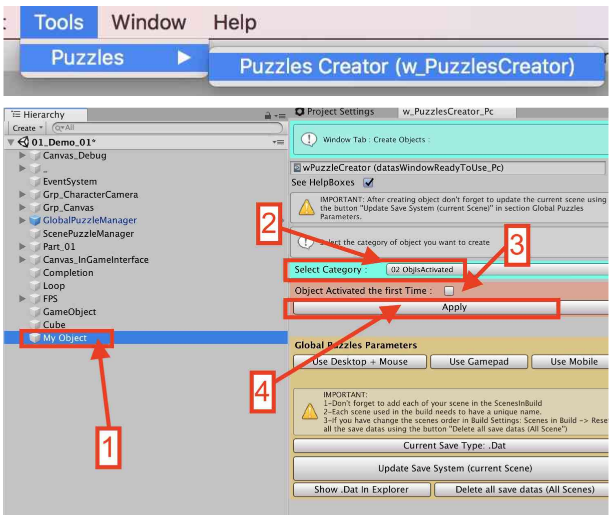
How to setup an object as ObjIsActivated (Tuto).

-Go to Tools → Puzzles → Puzzles Creator (w_PuzzlesCreator)

-In the Hierarchy select the object to setup (spot 1).

-In window tab w_PuzzlesCreator_Pc select category 02 ObjIsActivated (spot 2).

-Choose if the object is enabled or disabled the first time the object is loaded in a scene (spot 3).



Window Tab : Create Objects :

wPuzzleCreator (datasWindowReadyToUse_Pc)

See HelpBoxes ✓

IMPORTANT: After creating object don't forget to update the current scene using the button "Update Save System (current Scene)" in section Global Puzzles Parameters.

Select the category of object you want to create

IMPORTANT:
1-Don't forget to add each of your scene in the ScenesInBuild
2-Each scene used in the build needs to have a unique name.
3-If you have change the scenes order in Build Settings: Scenes in Build -> Rese
all the save datas using the button "Delete all save datas (All Scene")

***Caution**:*
If a meshRenderer is attached to your object:
This mesh component is enabled/disabled in the Hierarchy.

*If **No** meshRenderer is attached to your object:*
This gameObject is enabled/disabled in the Hierarchy.

-Click Apply (spot 4)

***Info**: In the Inspector IsObjectActivated_Pc.cs and SaveData_Pc.cs scripts have been added.*

Add new method in section Call Methods during loading process (Tuto):

Important: If you do not have an object setup as ObjIsActivated first read the previous section.
(How to setup an object as ObjIsActivated)

Even though the sections are different, the process for adding methods remains the same.

In several sections of the puzzle system it is possible to add custom methods.

*(**More Info**: If you add a method for the first time and want more information have a look to the section How to add a method (overview))*

Caution:
In section **Call Methods during loading process:**
-Only boolean methods are allowed.
Other methods are ignored.

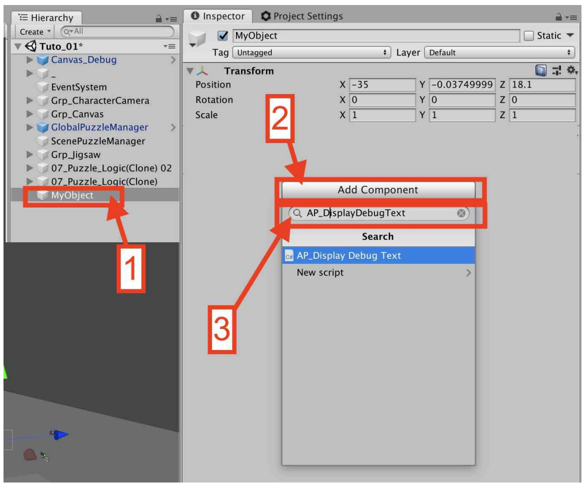
Example:
Add the Method bool_Display_Txt_01 that:
-Return true.
-Display text “Activated” in the Console Tab during the loading process.

-Create an empty object in the Hierarchy and rename it MyObject for example (spot 1).

-In the Inspector press button Add Component (spot 2)

-In search field write AP_DisplayDebugText (spot 3)

-Press keyboard key Enter to add the component.



-In the Hierarchy select your ObjIsActivated (spot 1).

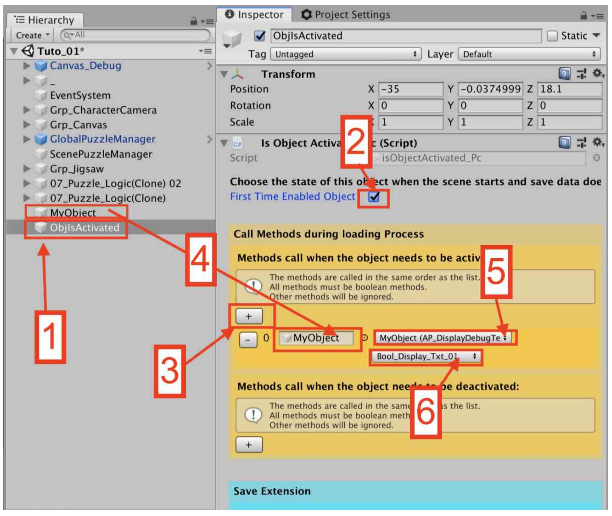
-Check the boxFirstTimeEnabledObject (spot 2)
(More info about boxFirstTimeEnabledObject)

-In the Inspector press button + in the section Method call when the object need to be activated to create a new empty slot (spot 3)

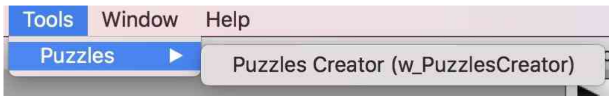
-Drag and drop MyObject in the empty slot (spot 4)

-Select the script AP_DisplayDebugText (spot 5)

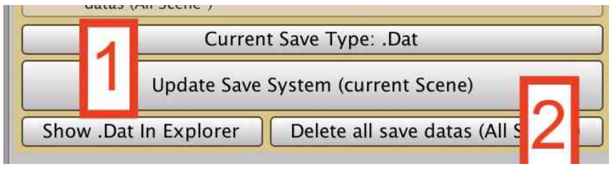
-Select the method AP_Display_Txt (spot 6)


Hierarchy
Create
Tuto_01*
Canvas_Debug
EventSystem
Grp_CharacterCamera
Grp_Canvas
GlobalPuzzleManager
ScenePuzzleManager
Grp_Jigsaw
07_Puzzle_Logic(Clone) 02
07_Puzzle_Logic(Clone)
MyObject
ObjIsActivated
Inspector
Project Settings
ObjIsActivated
Static
Tag Untagged
Layer Default
Transform
Position X -35 Y -0.0374999 Z 18.1
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Is Object Activa c (Script)
Script isObjectActivated_Pc
Choose the state of this object when the scene starts and save data doe
First Time Enabled Object
Call Methods during loading Process
Methods call when the object needs to be activ
The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.
MyObject
MyObject (AP_DisplayDebugTe
Bool_Display_Txt_01
Methods call when the object nee e deactivated:
Save Extension
1
2
3
4
5
6

-Open the window tab w_PuzzlesCreator
Tools → Puzzles → Puzzles Creator (w_PuzzlesCreator)



Press Update Save System (current scene) (spot 1) and Press Delete all save datas (spot 2) to reset the save system.


Current Save Type: .Dat
Update Save System (current Scene)
Show .Dat In Explorer
Delete all save datas (All S
1
2

Press Play.

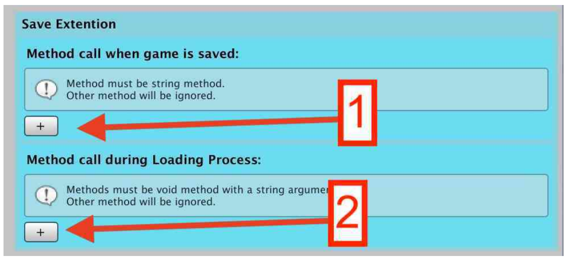
Info: *The text “Activated” is displayed in the console tab when the scene starts.*

Save Extension (script needed)**:**
This section allows to save and load your own parameters.

For example:
-Save if a door is open / closed.
-Save if a padlock is open or closed.
-Save the position of an object.

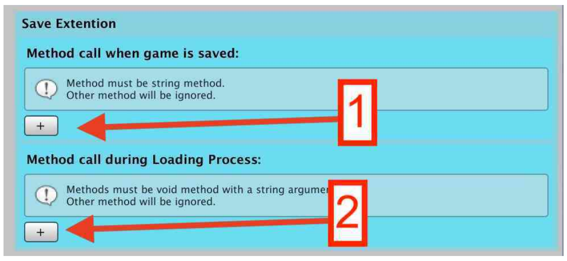
Save extension is divided in 2 parts:
Part 1: Methods call when game is saved (spot 1):
This section allows to save your own parameters.

Part 2: Methods call during loading Process (spot 2)
This section allows to load and use your parameters during the loading process.


Save Extention
Method call when game is saved:
Method must be string method.
Other method will be ignored.
Method call during Loading Process:
Methods must be void method with a string argume
Other method will be ignored.

How it works:
The section Methods call when game is saved is used to save your datas in a string method when the scene is saved (spot 1).

***Important**: Only a string method is allowed. Other methods types are ignored.*


Save Extention
Method call when game is saved:
Method must be string method.
Other method will be ignored.
Method call during Loading Process:
Methods must be void method with a string argume
Other method will be ignored.

The section Methods call during loading Process:
-Is used when puzzles are initialized.
-Is used to load your datas in a void method with one string argument (spot 2).

***Important**: Only a void method with one string argument is allowed. Other methods types are ignored.*

Save Extension Example (Tuto)**:**

Example:
On a scene we have a door.
-We want to save if the door is opened or closed when the player saves the game.
-We want to load and initialized the door state during the scene loading process.

In the example we are going to save and load the state of 2 variables:
b_IsDoorUnlocked
b_IsDoorOpened

In this example, we are going to:

In your favorite script editor:
Step 1-Create a method to save the parameters
Step 2-Create a method to load and initialized the parameters.

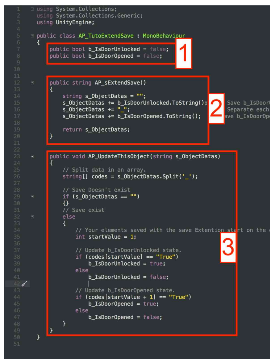
In Unity:
Step 3-Setup ObjIsActivated script

Step 1:Create the string method to save the variables states:

In the example we are going to save and load the state of 2 variables (spot 1):
b_IsDoorUnlocked
b_IsDoorOpened

Info:
To save the state of our variables a string method is needed (spot 2).

To load the state of our variables a void method with a string argument is needed (spot 3).



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class AP_TutoExtendSave : MonoBehaviour
{
    public bool b_IsDoorUnlocked = false;
    public bool b_IsDoorOpened = false;


    public string AP_sExtendSave()
    {
        string s_ObjectDatas = "";
        s_ObjectDatas += b_IsDoorUnlocked.ToString();       Save b_IsDoor
        s_ObjectDatas += "_";                               Separate each
        s_ObjectDatas += b_IsDoorOpened.ToString();         ave b_IsDoorOp

        return s_ObjectDatas;
    }


    public void AP_UpdateThisObject(string s_ObjectDatas)
    {
        // Split data in an array.
        string[] codes = s_ObjectDatas.Split('_');

        // Save Doesn't exist
        if (s_ObjectDatas == "")
        {}
        // Save exist
        else
        {
            // Your elements saved with the save Extention start on the
            int startValue = 1;

            // Update b_IsDoorUnlocked state.
            if (codes[startValue] == "True")
                b_IsDoorUnlocked = true;
            else
                b_IsDoorUnlocked = false;

            // Update b_IsDoorOpened state.
            if (codes[startValue + 1] == "True")
                b_IsDoorOpened = true;
            else
                b_IsDoorOpened = false;
        }
    }
}
```

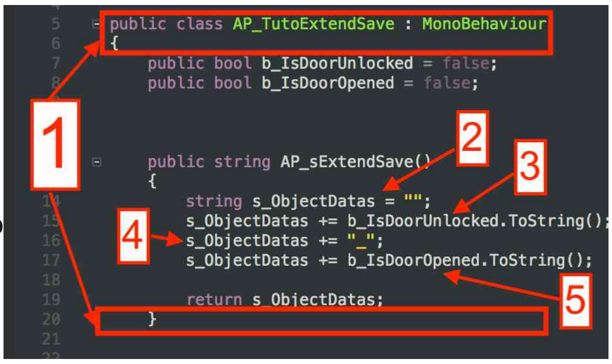
To do that:
-Create a method that return a string (spot 1)

-Create a string named s_ObjectDatas (spot 2).

-Add to the string s_ObjectDatas the state of b_IsDoorUnlocked and convert this boolean value to a string (spot 3).

-Add _ (underscore) to separate each element that need to be saved (spot 4).

-Add to the string s_ObjectDatas the state of IsDoorOpened and convert this boolean value to a string (spot 5).



```
public class AP_TutoExtendSave : MonoBehaviour
{
    public bool b_IsDoorUnlocked = false;
    public bool b_IsDoorOpened = false;


    public string AP_sExtendSave()
    {
        string s_ObjectDatas = "";
        s_ObjectDatas += b_IsDoorUnlocked.ToString();
        s_ObjectDatas += "_";
        s_ObjectDatas += b_IsDoorOpened.ToString();

        return s_ObjectDatas;
    }
```

Step 2: Create the void method with 1 string argument to load the variables states:

-Create a void method with one string argument (spot 1)

-Create a string array named codes to split datas in an array (spot 2).

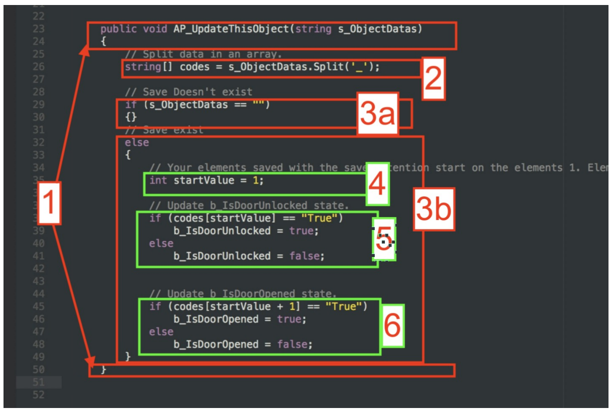
-Create a if condition for the case where there is no existing save (spot 3a).

-Create the else condition to initialized the variables (spot 3b).

__important__: Element 0 in the string array codes is used to saved if the object is activated or deactivated in the scene. The Elements saved in the Save Extension must start at element 1. It is the reason why the variable startValue = 1 is created (spot 4)

Then update the b_IsDoorUnlocked state using the first value saved in the previous section (spot 5).

Then update the b_ IsDoorOpened state using the second value saved in the previous section (spot 6).



```
    public void AP_UpdateThisObject(string s_ObjectDatas)
    {
        // Split data in an array.
        string[] codes = s_ObjectDatas.Split('_');

        // Save Doesn't exist
        if (s_ObjectDatas == "")
        {}
        // Save exist
        else
        {
            // Your elements saved with the save[illegible]ention start on the elements 1. Ele
            int startValue = 1;

            // Update b_IsDoorUnlocked state.
            if (codes[startValue] == "True")
                b_IsDoorUnlocked = true;
            else
                b_IsDoorUnlocked = false;


            // Update b_IsDoorOpened state.
            if (codes[startValue + 1] == "True")
                b_IsDoorOpened = true;
            else
                b_IsDoorOpened = false;
        }
    }
```

Step 3: Unity Setup

-Create an empty object in your scene and rename it MyObject (spot 1).

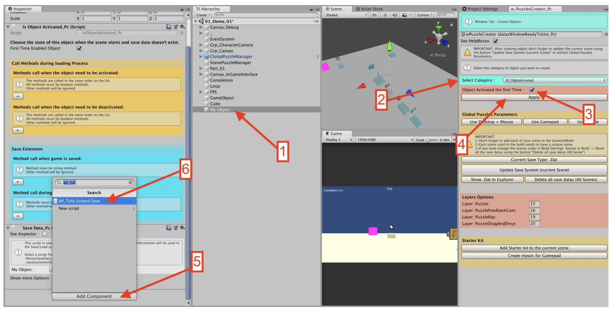
-In the w_PuzzleCreator_Pc window tab
(Tools → Puzzles → Puzzles Creator)
select 02 ObjIsActivated (spot 2).

-Check the box Object Activated the first time (spot 3)

-Press Apply (spot 4)

-In the Inspector press Add Component (spot 5)

-Choose AP_TutoExtendSave.cs (spot 6).
(This script contains the variables and the methods used for this example.)



-Select MyObject in the Hierarchy (spot 1)

-In the Inspector press the button + (spot 2)

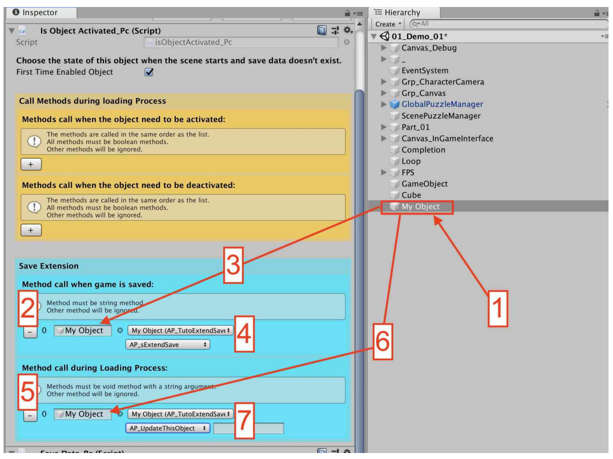
-Drag and drop MyObject in the empty field (spot 3)

-Choose the script AP_TutoExtendSave and the string method AP_sExtendSave (spot 4).

-Press the button + (spot 5)

-Drag and drop MyObject in the empty field (spot 6)

-Choose the script AP_TutoExtendSave and the string method AP_UpdateThisObject (spot 7).



Is Object Activated_Pc (Script)
Script isObjectActivated_Pc
Choose the state of this object when the scene starts and save data doesn't exist.
First Time Enabled Object ✓

Call Methods during loading Process
Methods call when the object need to be activated:
The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.
Methods call when the object need to be deactivated:
The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.

Save Extension
Method call when game is saved:
Method must be string method
Other method will be ignored.
0 My Object My Object (AP_TutoExtendSave
AP_sExtendSave
Method call during Loading Process:
Methods must be void method with a string argument.
Other method will be ignored.
0 My Object My Object (AP_TutoExtendSave
AP_UpdateThisObject

01_Demo_01*
Canvas_Debug
_
EventSystem
Grp_CharacterCamera
Grp_Canvas
GlobalPuzzleManager
ScenePuzzleManager
Part_01
Canvas_InGameInterface
Completion
Loop
FPS
GameObject
Cube
My Object

Info: *Now MyObject is able to save and load the b_IsDoorUnlocked and b_IsDoorOpened parameters*

Summary:
1-The elements are saved as a string.
2-Use a string method to save your elements.
3-Between each element to save add an underscore _
4-Initialize your elements in a void method with one string argument.

The Character.

The asset is designed to use your own character.

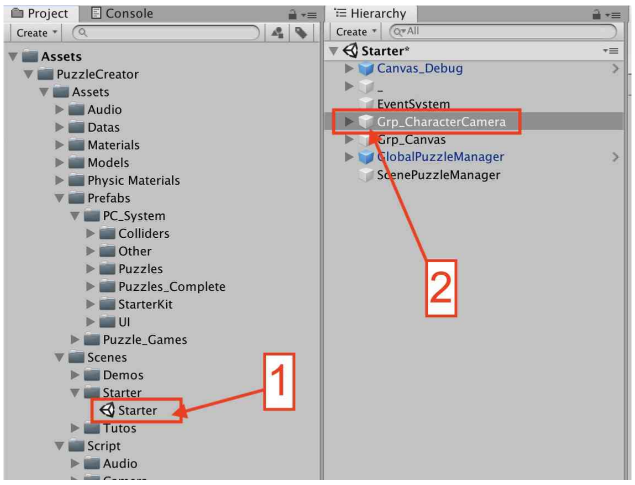
However you can use the character provided with the asset.

This section explains how to setup the character provided in the asset depending different scenario.

Case 1: Ready to use

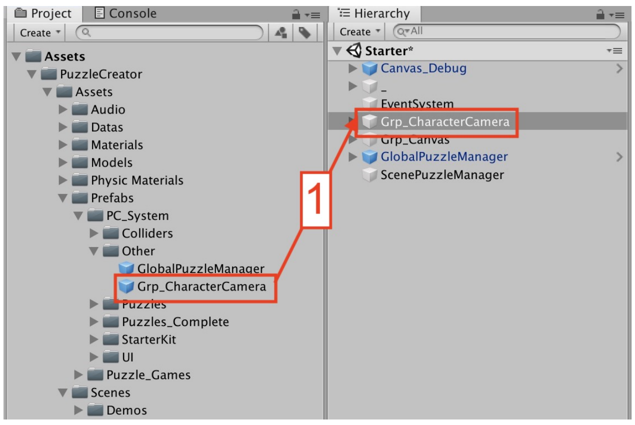
You use the starting scene named Starter (spot 1): *(PuzzleCreator → Assets → Scenes → Starter → Starter)*

The character is already integrated by default (spot 2). *(Hierarchy: Grp_CharacterCamera)*



Case 2: Desktop

If you want to integrate the character to an existing desktop project:
Drag and drop the prefab Grp_CharacterCamera in your scene (spot 1)
(PuzzleCreator → Assets → Prefabs → Pc_System → Other → Grp_CharacterCamera)

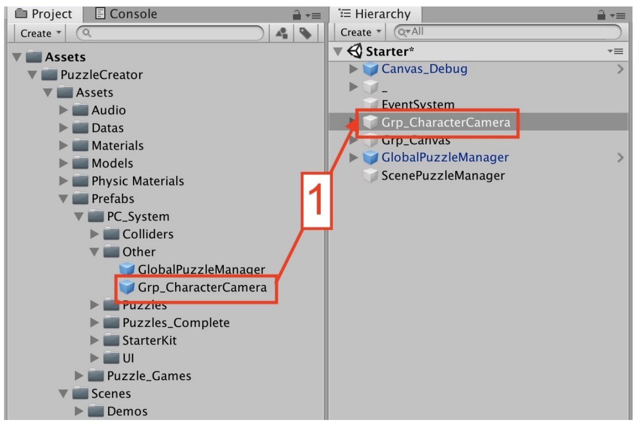


Case 3: Mobile

If you want to integrate the character to an existing mobile project:

Drag and drop the prefab Grp_CharacterCamera in your scene (spot 1)

(PuzzleCreator → Assets → Prefabs → Pc_System → Other → Grp_CharacterCamera)

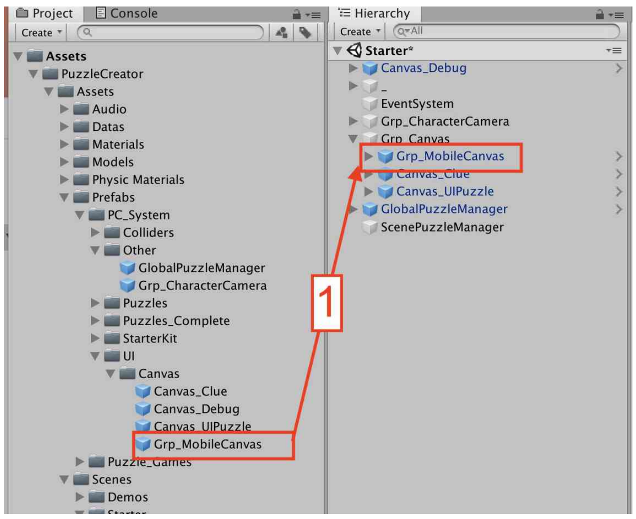


Drag and drop the prefab Grp_MobileCanvas in your scene (spot 1)

(PuzzleCreator → Assets → Prefabs → Pc_System → UI → Canvas → Grp_MobileCanvas)

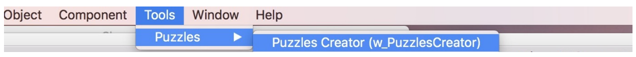
***Info**: The Grp_MobileCanvas prefab only works with the character provided in the asset.*

If you want to use the UI with your character you will have to code the connection between UI and character by yourself.



Open the window w_PuzzlesCreator:

(Tools→ Puzzles → Puzzles Creator (w_PuzzlesCreator)

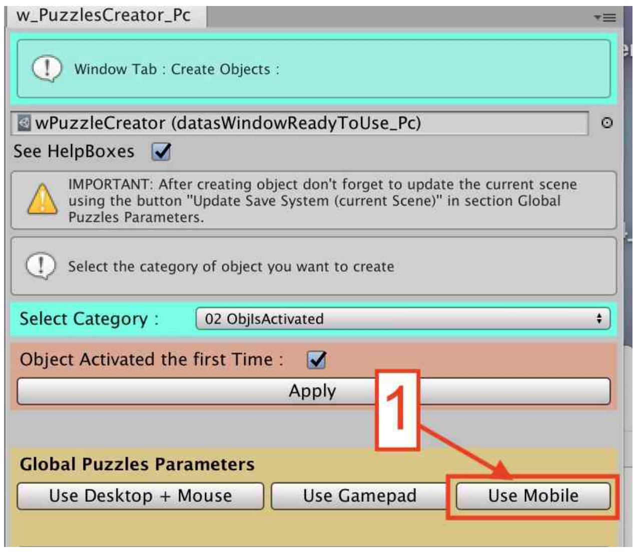


In window w_PuzzlesCreator press button Use Mobile (spot 1)

Info:

This button allows to:

-Activated the Mobile UI inputs for the character if the prefab Grp_MobileCanvas is in the Hierarchy.



Window Tab : Create Objects :

wPuzzleCreator (datasWindowReadyToUse_Pc)

See HelpBoxes ✓

IMPORTANT: After creating object don't forget to update the current scene using the button "Update Save System (current Scene)" in section Global Puzzles Parameters.

Select the category of object you want to create

Select Category : 02 ObjIsActivated

Object Activated the first Time : ✓

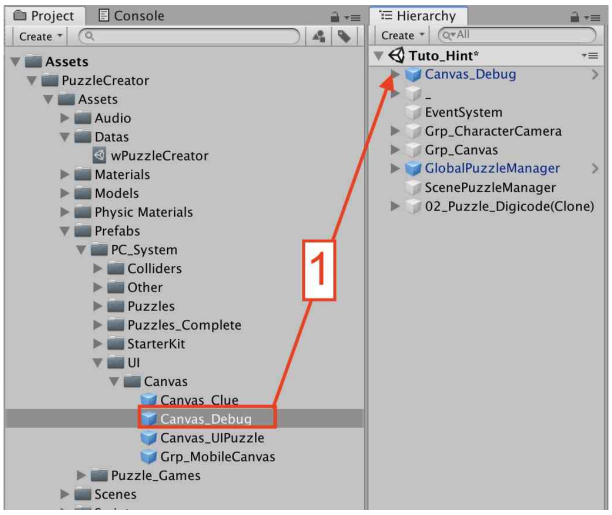
Debugger

This object help you debug puzzles more easily:

How to use the debugger:

-Drag and drop into your scene the prefab Canvas_Debug (spot 1)

Project tab: Assets → PuzzleCreator → Assets → Prefabs → PC_System → UI → Canvas → Canvas_Debug



By default when scene is in Play Mode:
-Press F to finish a puzzle without solving it.
-Press F a second time to cancel the debugger action.

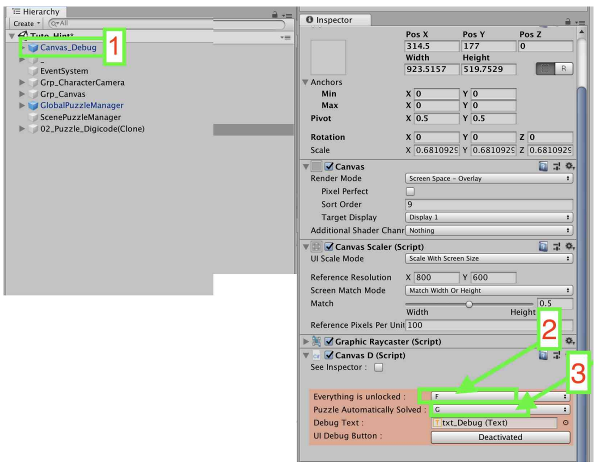
-Press G to bypass conditions to open a puzzle.
-Press G a second time to cancel the debugger action.

Customize debugger parameters:

-Select Canvas_Debug in the Hierarchy (spot 1).

-Select the input in the dropdown list to bypass conditions to open a puzzle (spot 2).

-Select the input in the dropdown list to finish a puzzle without solving it (spot 3).



Pos X 314.5 Pos Y 177 Pos Z 0
Width 923.5157 Height 519.7529
Anchors
Min X 0 Y 0
Max X 0 Y 0
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 0.6810929 Y 0.6810929 Z 0.6810929

Canvas
Render Mode: Screen Space – Overlay
Pixel Perfect: ☐
Sort Order: 9
Target Display: Display 1
Additional Shader Chanr: Nothing

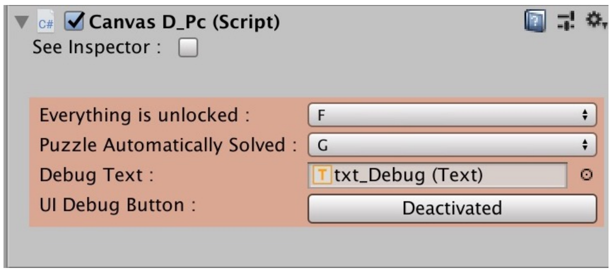
Canvas Scaler (Script)
UI Scale Mode: Scale With Screen Size
Reference Resolution X 800 Y 600
Screen Match Mode: Match Width Or Height
Match: 0.5 (Width – Height)
Reference Pixels Per Unit: 100

Graphic Raycaster (Script)

Canvas D (Script)
See Inspector : ☐
Everything is unlocked : F
Puzzle Automatically Solved : G
Debug Text : txt_Debug (Text)
UI Debug Button : Deactivated

Customize debugger for Mobile:

-Click on the button next to UI Debug Button to activate or deactivate the UI Debug button.

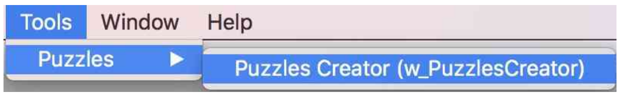


Canvas D_Pc (Script)
See Inspector : ☐
Everything is unlocked : F
Puzzle Automatically Solved : G
Debug Text : txt_Debug (Text)
UI Debug Button : Deactivated

Troubleshooting

1-The scene doesn't work / There is an error in the console tab

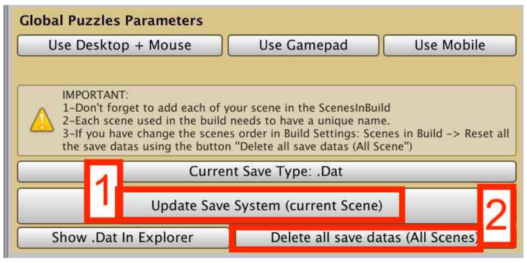
If you have an error in the console tab the first thing to try is:



-Go to: Tools → Puzzles → Puzzles creator (w_PuzzlesCreator)

-Press button Update Save System (current scene) (spot 1)

-Press button Delete all the save datas (All scenes) (spot 2)



Global Puzzles Parameters
Use Desktop + Mouse
Use Gamepad
Use Mobile
Current Save Type: .Dat
1
Update Save System (current Scene)
2
Show .Dat In Explorer
Delete all save datas (All Scenes)

IMPORTANT:
1-Don't forget to add each of your scene in the ScenesInBuild
2-Each scene used in the build needs to have a unique name.
3-If you have change the scenes order in Build Settings: Scenes in Build -> Reset all the save datas using the button "Delete all save datas (All Scene")

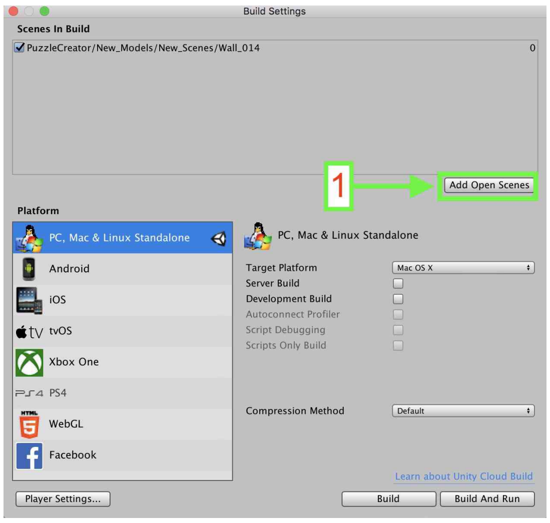
2-Current scene save is not erased after pressing update Save System (current scene)

When the button Update Save System (current scene) is pressed save datas for this scene are deleted only if the scene is part of build scenes.

-Go to File → Build Settings
-Press add Open Scene

Now when the button Update Save System (current scene) is pressed the puzzle datas are deleted for this scene.

Important: If you change the scene order in the SceneInBuild window you need to press Delete all save datas (All Scenes) to update the save system.



☑ PuzzleCreator/New_Models/New_Scenes/Wall_014 0

PC, Mac & Linux Standalone

Target Platform	Mac OS X
Server Build	☐
Development Build	☐
Autoconnect Profiler	☐
Script Debugging	☐
Scripts Only Build	☐
Compression Method	Default

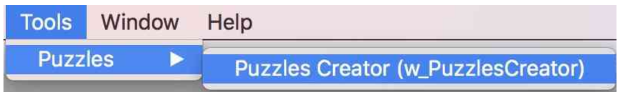
3-Puzzle Detection issues

If 2 puzzles are two closed to each other it can cause issues.
(More info in the sub-section Detection Restrictions section Puzzle Detection Type in Doc Part 2)

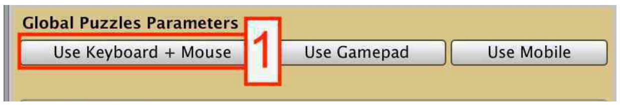
4-The character controls are reversed

If the character controls are inverted when you use the keyboard:

-Go to: Tools → Puzzles → Puzzles creator (w_PuzzlesCreator)



-In section Global Puzzles Parameters press button Use Keyboard + Mouse (spot 1)

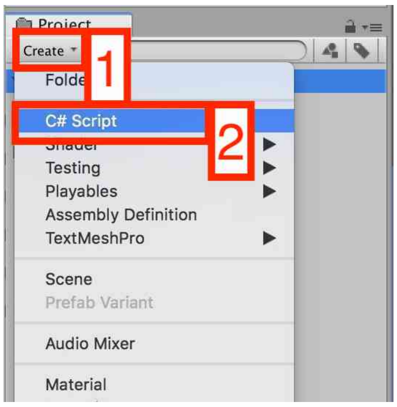


Scripting:

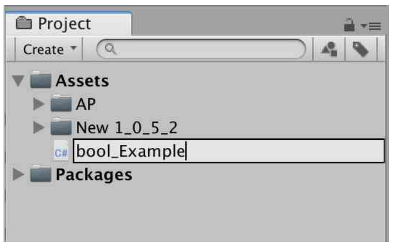
Scripting: How to create a boolean method

-On project tab click on Create button (spot 1)

-On the dropdown menu choose C# script (spot 2)

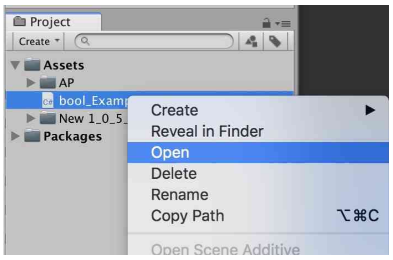


Rename the script bool_Example

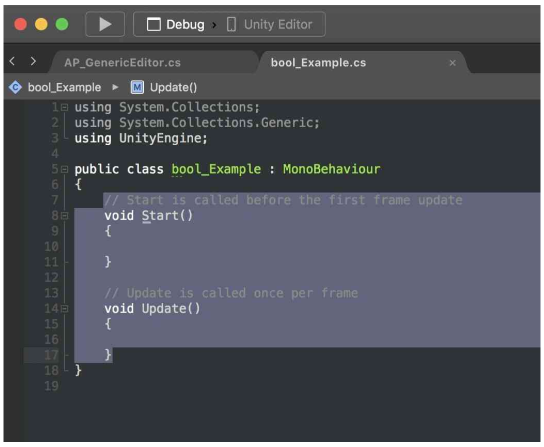


-Right click on script bool_Example.

-On the dropdown menu choose Open.



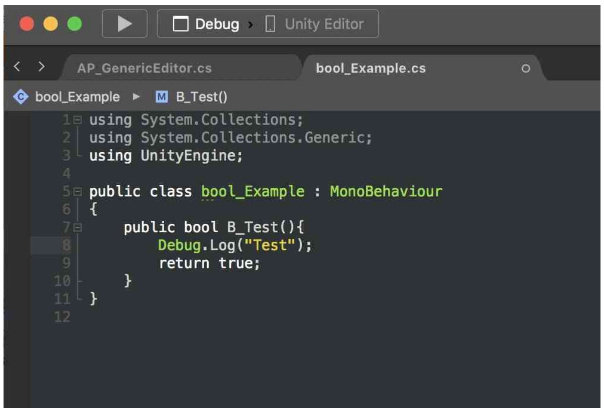
-Select and delete line 7 to 17 (method start and Update).



```
 1  using System.Collections;
 2  using System.Collections.Generic;
 3  using UnityEngine;
 4
 5  public class bool_Example : MonoBehaviour
 6  {
 7      // Start is called before the first frame update
 8      void Start()
 9      {
10
11      }
12
13      // Update is called once per frame
14      void Update()
15      {
16
17      }
18  }
19
```

Write this:

```
public bool B_Test(){
 Debug.Log("Test");
 return true;
}
```



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class bool_Example : MonoBehaviour
{
    public bool B_Test(){
        Debug.Log("Test");
        return true;
    }
}
```

How it works:

public (spot 1):
Allows to access this method outside this script.

bool (spot 2):
Define that it is a boolean method.

B_Test (spot 3):
It is the name of the method.

Debug.Log (spot 4):
In this example we write Test in the console when the method is call. Write your code here.

Return true; (spot 5):
A boolean method return True or False

In our case the method must return True at the end of the method.

This allows the modules contained in the asset to know that the method is complete.

In the asset it is possible to call a list of boolean methods. These methods are called one after the other. When the first method is finished its returns True. With that information it is possible to call the following method and so on.

Scripting: Puzzle Inputs:

Switch between inputs types at runtime:

From any script you can switch inputs to:

Desktop input with:

```
AP_GlobalPuzzleManager.instance.AP_SwitchPuzzleInputsToKeyboardAndMouse(true);
```

Gamepad input with:

```
AP_GlobalPuzzleManager.instance.AP_SwitchPuzzleInputsToGamepad(true);
```

Mobile input with:

```
AP_GlobalPuzzleManager.instance.AP_SwitchPuzzleInputsToMobile();
```

Access inputs at runtimes:

Keyboard Inputs:

Validation button (KeyCode Type):

```
AP_GlobalPuzzleManager.instance.validationButtonKeyboard
```

Exit Puzzle (Mode Focus) (KeyCode Type):

```
AP_GlobalPuzzleManager.instance.backButtonKeyboard
```

Gamepad Inputs:

Cursor Horizontal (Mode Focus) (string Type):

```
AP_GlobalPuzzleManager.instance.HorizontalAxisJoystickLeft
```

Cursor Vertical (Mode Focus) (string Type):

```
AP_GlobalPuzzleManager.instance.VerticalAxisJoystickLeft
```

Validation button (KeyCode Type):

```
AP_GlobalPuzzleManager.instance.validationButtonJoystick
```

Exit Puzzle (Mode Focus) (KeyCode Type):

```
AP_GlobalPuzzleManager.instance.backButtonJoystick
```

Scripting: Useful Methods or variables for AR:

By default a puzzle is activated and available:
If the player click on the puzzle
or enter the trigger PuzzleDetector.

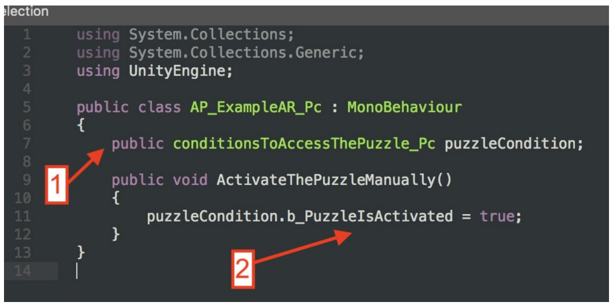
In AR it is not possible. So it needed to activate manually a puzzle.

How to do that:

Change the state of b_PuzzleIsActivated in the script ConditionsToAccessThePuzzle_Pc.cs that can be find in the puzzle Inspector.

-Declare a variable to access the script ConditionsToAccessThePuzzle_Pc (spot 1).

-Call a method to change to true the state of the variable b_PuzzleIsActivated;



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class AP_ExampleAR_Pc : MonoBehaviour
{
    public conditionsToAccessThePuzzle_Pc puzzleCondition;

    public void ActivateThePuzzleManually()
    {
        puzzleCondition.b_PuzzleIsActivated = true;
    }
}
```

How to add a method (overview):

Even though the sections are different, the process for adding methods remains the same:

In several sections of the puzzle system it is possible to add custom methods.

Step 1
-Add your script to an object of the Hierarchy. (Spot 1)

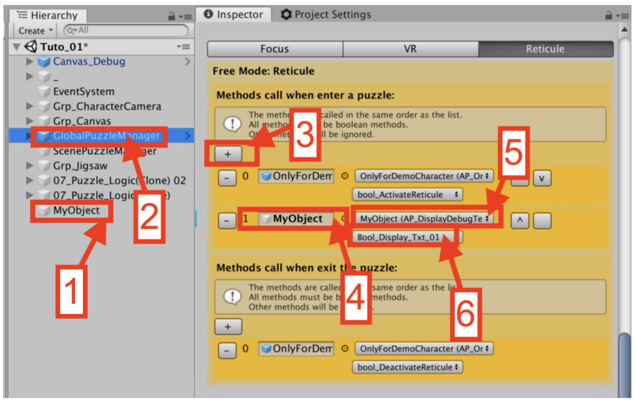
Step 2
-In Inspector go to the section where you want to add your method (spot 2)

Step 3
-Create a new slot by pressing + (spot 3)

Step 4
-In the Inspector, drag and drop the object of the Hierarchy including your script into the new empty slot (spot 4).

Step 5
-Choose in the dropdown menu the script that contains the method you want to use (spot 5).

Step 6
-Choose in the second dropdown menu your method (spot 6).



The methods are called in the same order as the list.
All methods must be boolean methods.
Other methods will be ignored.

Scripting: Check if the access to a puzzle is allowed or denied

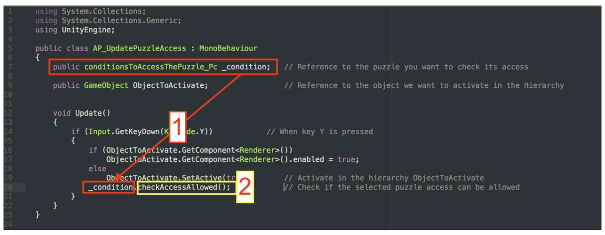
We will take an example:
There is a key and a puzzle in the scene.
Access to the puzzle is only possible if the key is present in the player's inventory.

For optimization reasons:
We don’t want that the puzzle check each frame that the key is in the player's inventory.

What to do?
Once the key is in the player's inventory:
You have to ask the puzzle to check if the conditions of access to the puzzles are done (In our case: The key must be in the player's inventory).

How to do it?
In each puzzle you have a conditionToAccessPuzzle.cs script.
In this script is the checkAccessAllowed() method checks if the conditions are met to access this puzzle.

To call this method you must:
Access the conditionToAccessPuzzle.cs script of your puzzle (spot 1).
Call the checkAccessAllowed () method (spot 2).



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class AP_UpdatePuzzleAccess : MonoBehaviour
{
    public conditionsToAccessThePuzzle_Pc _condition;   // Reference to the puzzle you want to check its access

    public GameObject ObjectToActivate;                  // Reference to the object we want to activate in the Hierarchy


    void Update()
    {
        if (Input.GetKeyDown(K    de.Y))          // When key Y is pressed
        {
            if (ObjectToActivate.GetComponent<Renderer>())
                ObjectToActivate.GetComponent<Renderer>().enabled = true;
            else
                ObjectToActivate.SetActive(tr             // Activate in the hierarchy ObjectToActivate
            _condition.checkAccessAllowed();               // Check if the selected puzzle access can be allowed
        }
    }
}
```

To have another example read:
Doc 1 section Tuto 1
sub-section Add a condition to start a puzzle.