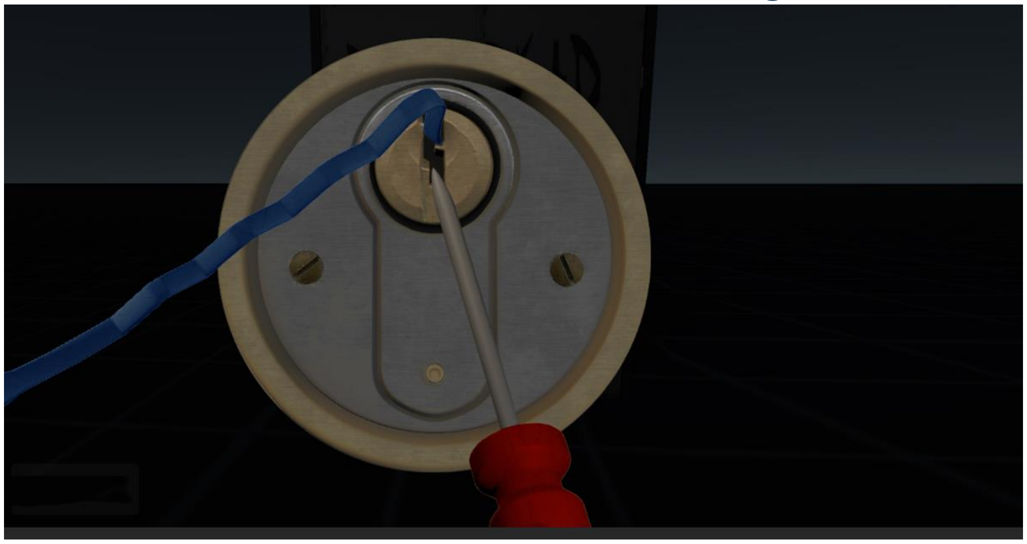
By Majd Abdulqadir

Locks Plus: Kingdom Lock

Game documentation and HowTo guide.



This document contains:

Package Description and features 2
PACKAGE INCLUDES........ **Error! Bookmark not defined.**
Update history 2
Overview of the project's library contents 3
Customization Guide........ 4
Getting started........ 4
LPActivator 4
LPKingdomLock........ 5
Integration with Realistic FPS Prefab 7
Integration with Ultimate FPS framework 10

Package Description and features

LOCKS+ Lockpicking gives you a powerful locking system you can easily integrate into your FPS game. It can work as a standalone project, and can also be integrated into your current FPS project.

Works with Ultimate FPS Framework

Works with Realistic FPS Prefab too

Current version 1.0

Update history

1.0 (27.06.2018)
- Initial version

Overview of the project's library contents

Let's take a look inside the game files. Open the main LocksPlus folder using Unity3D 5.5 or newer. Take a look at the project library, usually placed on the right or bottom side of the screen. Here are the various folders inside:

- **Lockpicking:** Holds all the assets related to the Lockpicking system.
    - **Animations:** Holds all the animations we use in the locks.
    - **Fonts:** Holds the font used in the project.
    - **Materials:** Holds some of the materials used in the demo.
    - **Models:** Holds the 3D model for the chest.
    - **Prefabs:** Holds all the lock examples we use in the demo, including 3 chest examples.
    - **Scripts:** Holds all the scripts used in the project.
    - **Sounds:** Holds all the sounds used in the demo.
    - **Textures:** Holds all the textures used in the project.
- **Lockpicking Demo:** This is the main demo scene we can test.

Getting started

Locks Plus is meant to be used as part of a game project such as Realistic FPS Prefab, or any other game that allows you to interact with objects. You can import it on its own and test the demo scene, or import it into another project and integrate it into the demo scenes of that project.

(When importing into an existing UFPS, RFPS, or other complete packages, don't import Project Settings!)

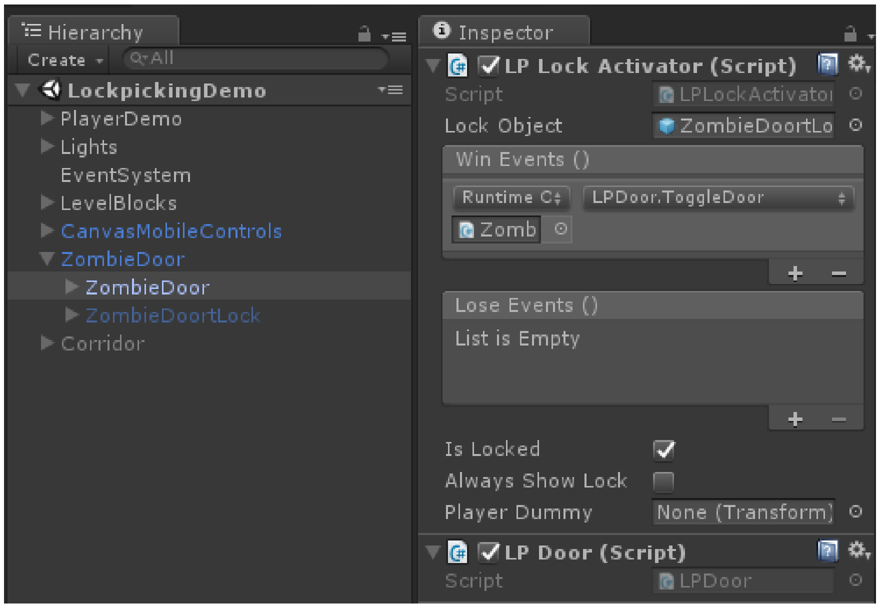
To start testing the locks, simply import the package into an empty project and open the demo scene LockpickingDemo. Here you can see several examples of the kingdom lock you can interact with.

There are two scripts that allow you to use the Dial Pad lock system, LPLockActivator, and LPLockpicking. Read more about them below.

LPActivator

The Activator script allows the player to interact with objects in the scene and activate the relevant lock, then apply either a Win or Lose command which can trigger any custom function you want from the scene.

Select one of the chest objects and take a look at the LPActivator component assigned to it.



LP Lock Activator (Script)
Script: LPLockActivator
Lock Object: ZombieDoortLo
Win Events ()
Runtime C: LPDoor.ToggleDoor
Zomb
Lose Events ()
List is Empty
Is Locked: ✓
Always Show Lock:
Player Dummy: None (Transform)
LP Door (Script)
Script: LPDoor

Lock Object – The lock object that will be activated when we interact with this activator.

Win Actions – A list of actions that will be triggered when we successfully unlock the lock.

Lose Actions – A list of actions that will be triggered when we fail unlocking the lock. It uses the same elements as Win Actions.

Winning in the chest lock above will trigger a function called ToggleDoor, which is in the attached script LPDoor and simply toggles between two animations between opened and closed for the current chest. On the Lose Actions we didn't put anything, so nothing happens when we fail.

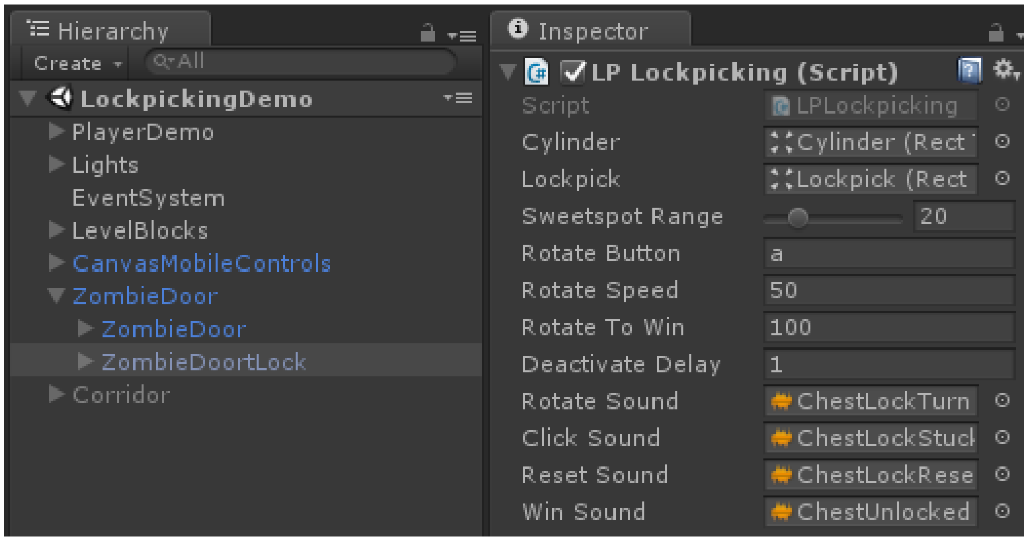
Is Locked – If the lock is locked, we interact with it and try to unlock it. If we succeed it becomes unlocked. If it is unlocked the Win Actions are triggered when we interact with it. This way you can set an object to be unlocked from the start, and even lock it during gameplay.

Always Show Lock – Always show the lock object (Don't hide it when at the start of the game or when we close it). This is good if you want to put the lock in world space, such as lock that can be played right in the game scene rather than showing a new window.

Player Dummy – Reposition the player and rotate it to face the lock object. This is useful if you want the player to look in a certain direction when interacting with a lock in the world space, similar to how you change your position when interacting with a console (facing a lock, or an alchemy table for example).

LPLockpicking

This is the component that holds the actual lock game with all its custom gameplay. Click on the object ZombieDoor inside and unhide it.



Script	LPLockpicking
Cylinder	Cylinder (Rect
Lockpick	Lockpick (Rect
Sweetspot Range	20
Rotate Button	a
Rotate Speed	50
Rotate To Win	100
Deactivate Delay	1
Rotate Sound	ChestLockTurn
Click Sound	ChestLockStuck
Reset Sound	ChestLockRese
Win Sound	ChestUnlocked

Sweetspot Range – How accurately close we need to be to the sweetspot. If set to 0, we need to be exactly at the sweetspot position, but if set to a higher number we can be farther from the angle of the sweetspot.

Rotate Button – The button that rotates the cylinder. The cylinder will only rotate if we are in the sweetspot.

Rotate Speed – How fast the cylinder rotates.

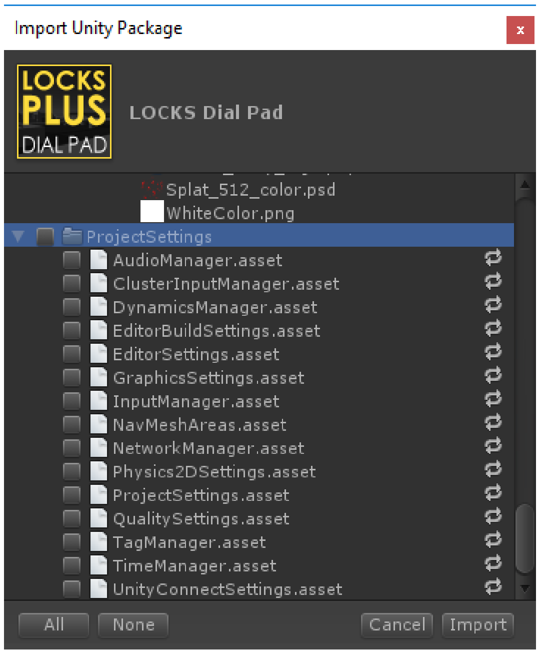
Rotate To Win – How much we need to rotate the cylinder in order to win.

Deactivate Delay – How long to wait before deactivating the lock object when we exit the game, either after winning/losing.

Sounds – Various sounds for rotating, clicking, resetting, and winning the lock.

Integration with Realistic FPS Prefab

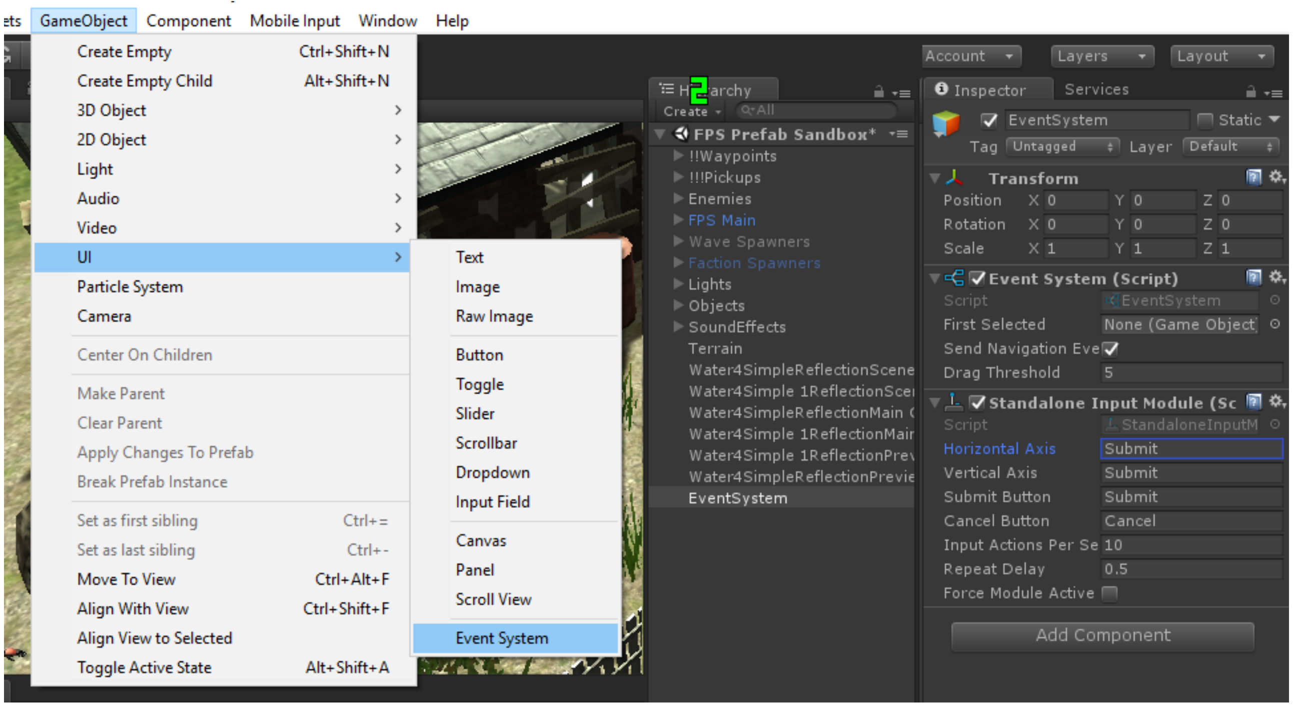
(When importing into an existing UFPS, RFPS, or other complete packages, don't import Project Settings!)



Import into your project the Realistic FPS Prefab project (tested with 1.44), open the Sandbox scene from the RFPS project, and drag one of the lock examples to the scene. In order to make it work with RFPS you need to set a few basic things.

(This guide refers to the LOCKS+ Dial Pad package, but it works the same way in all LOCKS+ packages)

First add an **EventSystem** object, and change the "Horizontal" and "Vertical" to local inputs that RFPS can understand like for example "Submit". The lock in RFPS doesn't use that input anyway, so we just want to avoid an error.



EventSystem — Tag: Untagged, Layer: Default

Transform: Position X 0 Y 0 Z 0; Rotation X 0 Y 0 Z 0; Scale X 1 Y 1 Z 1

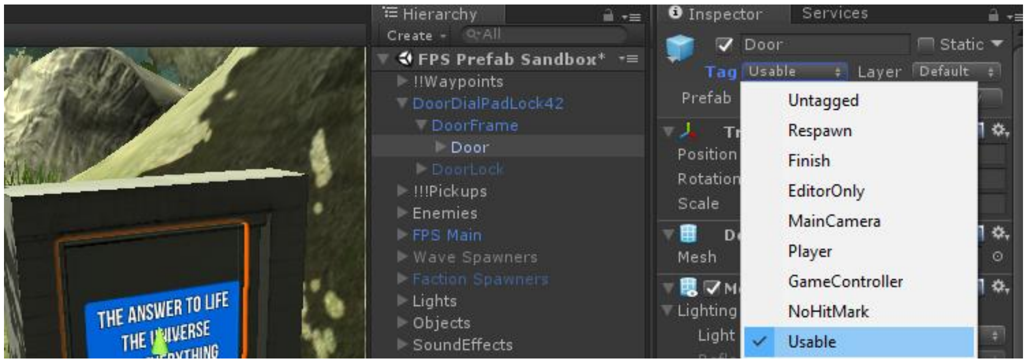
Event System (Script)

Script	EventSystem
First Selected	None (Game Object)
Send Navigation Eve	✓
Drag Threshold	5

Standalone Input Module (Sc

Script	StandaloneInputM
Horizontal Axis	Submit
Vertical Axis	Submit
Submit Button	Submit
Cancel Button	Cancel
Input Actions Per Se	10
Repeat Delay	0.5
Force Module Active	

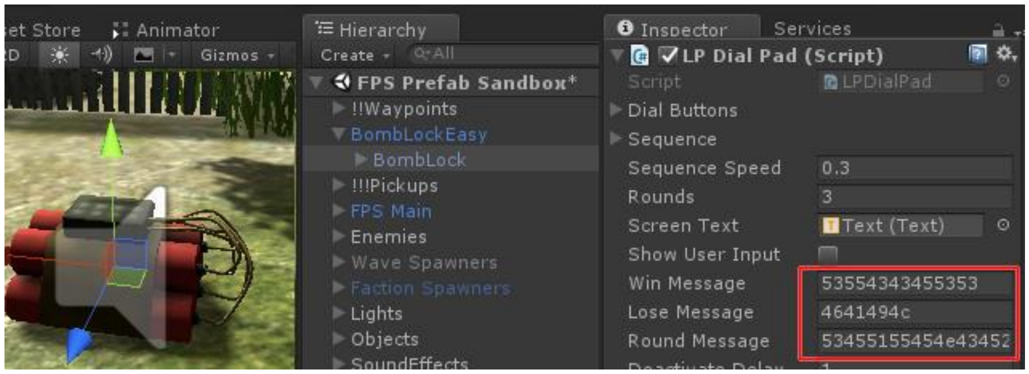
Next thing is to set the tag of your lock Activator to "Useable" which allows RFPS to interact with objects.



Door

Tag Usable Layer Default

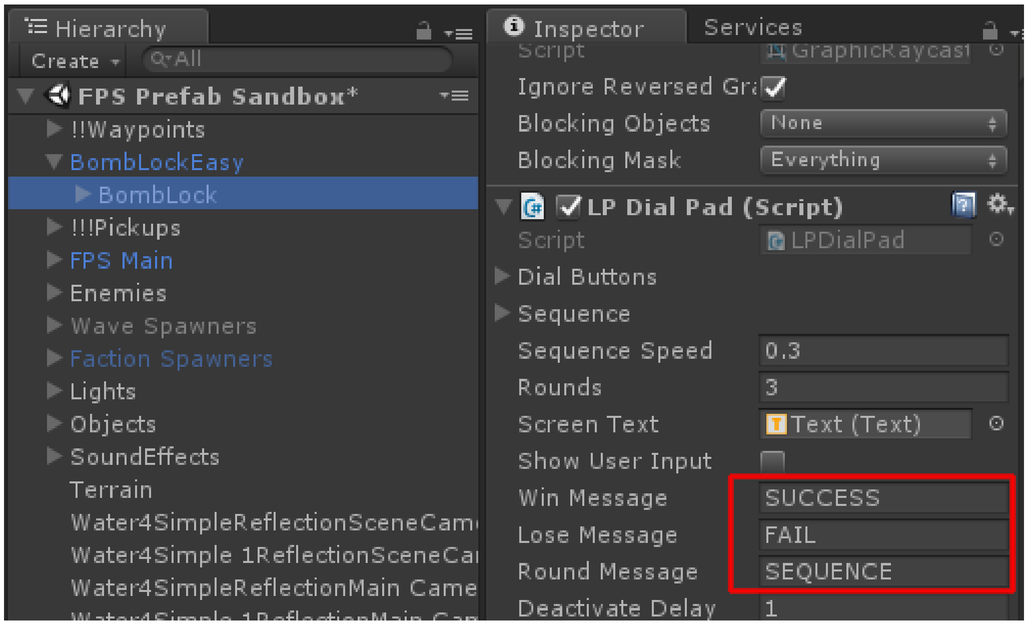
One last thing to check is that sometimes when importing a LOCKS+ package into RFPS, some of the text fields in the components are jumbled into random number/letter combinations, like this:



LP Dial Pad (Script)

Script	LPDialPad
Sequence Speed	0.3
Rounds	3
Screen Text	Text (Text)
Show User Input	
Win Message	53554343455353
Lose Message	4641494c
Round Message	53455155454e43452

To solve this simply import the LOCKS+ package again, and it will return to normal, like this:



Script	GraphicRaycast
Ignore Reversed Gr	✓
Blocking Objects	None
Blocking Mask	Everything

LP Dial Pad (Script)

Script	LPDialPad
Sequence Speed	0.3
Rounds	3
Screen Text	Text (Text)
Show User Input	
Win Message	SUCCESS
Lose Message	FAIL
Round Message	SEQUENCE
Deactivate Delay	1

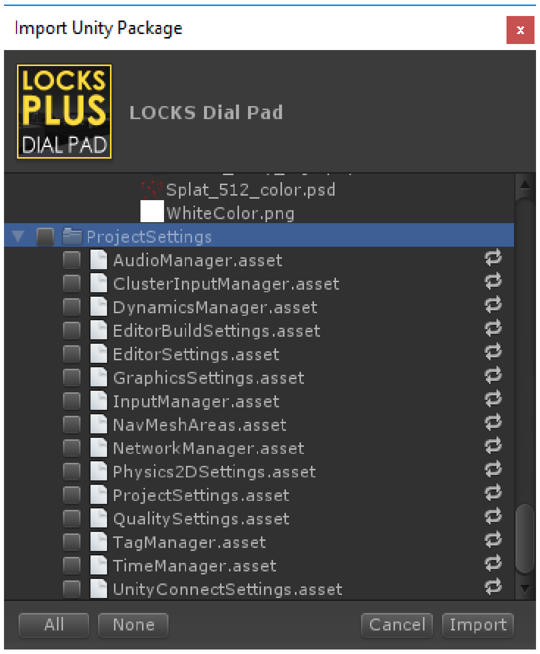
If you test the game now you'll be able to interact with the door, play the dial pad lock, unlock it and then open and close the door freely.

Check out this video to see how it's done:

https://www.youtube.com/watch?v=Qn5V6uDUrWE

Integration with Ultimate FPS framework

(When importing into an existing UFPS, RFPS, or other complete packages, don't import Project Settings!)

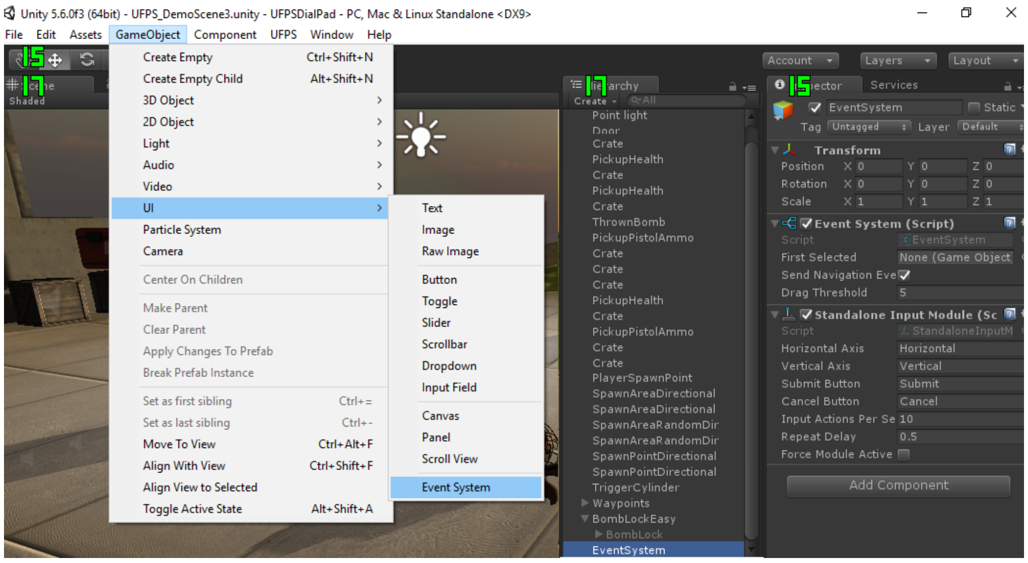


```
Splat_512_color.psd
WhiteColor.png
ProjectSettings
    AudioManager.asset
    ClusterInputManager.asset
    DynamicsManager.asset
    EditorBuildSettings.asset
    EditorSettings.asset
    GraphicsSettings.asset
    InputManager.asset
    NavMeshAreas.asset
    NetworkManager.asset
    Physics2DSettings.asset
    ProjectSettings.asset
    QualitySettings.asset
    TagManager.asset
    TimeManager.asset
    UnityConnectSettings.asset
```

Import into your project the Ultimate FPS package (tested with 1.7.2), open the UFPS_DemoScene3 scene from the UFPS project, and drag one of the lock examples to the scene. In order to make it work with UFPS you need to set a few basic things.

(This guide refers to the LOCKS+ Dial Pad package, but it works the same way in all LOCKS+ packages)

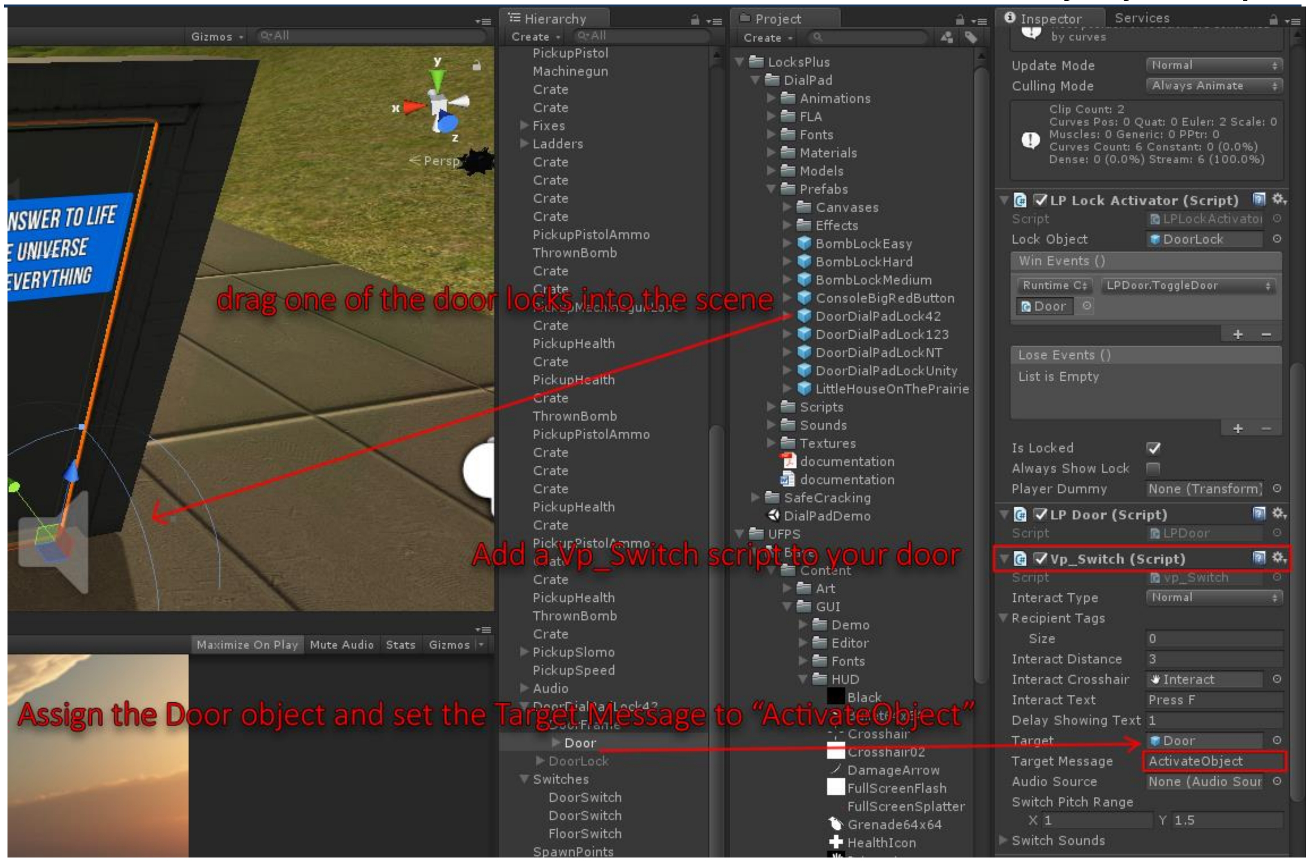
Add an EventSystem object to the scene. LOCKS+ is based on Unity UI and event system so we need that to make the buttons clickable.



```
Point light
Door
Crate
PickupHealth
Crate
PickupHealth
Crate
ThrownBomb
PickupPistolAmmo
Crate
Crate
Crate
PickupHealth
Crate
PickupPistolAmmo
Crate
Crate
PlayerSpawnPoint
SpawnAreaDirectional
SpawnAreaDirectional
SpawnAreaRandomDir
SpawnAreaRandomDir
SpawnPointDirectional
SpawnPointDirectional
TriggerCylinder
▶ Waypoints
▼ BombLockEasy
    ▶ BombLock
EventSystem
```

```
✓ EventSystem                 Static
Tag Untagged    Layer Default
Transform
Position   X 0   Y 0   Z 0
Rotation   X 0   Y 0   Z 0
Scale      X 1   Y 1   Z 1
✓ Event System (Script)
Script                 EventSystem
First Selected         None (Game Object
Send Navigation Eve    ✓
Drag Threshold         5
✓ Standalone Input Module (Sc
Script                 StandaloneInputM
Horizontal Axis        Horizontal
Vertical Axis          Vertical
Submit Button          Submit
Cancel Button          Cancel
Input Actions Per Se   10
Repeat Delay           0.5
Force Module Active
```

Now drag one of the LOCKS+ objects into the scene and let's define it.


drag one of the door locks into the scene

Add a Vp_Switch script to your door

Assign the Door object and set the Target Message to "ActivateObject"

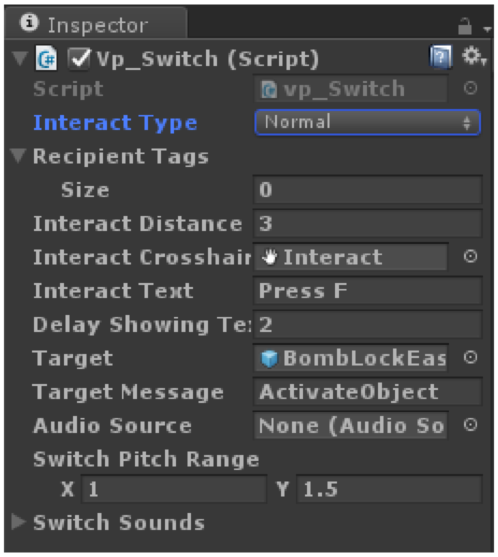
LP Lock Activator (Script)

Script	LPLockActivator
Lock Object	DoorLock
Win Events ()	Runtime C# LPDoor.ToggleDoor, Door
Lose Events ()	List is Empty
Is Locked	✓
Always Show Lock	
Player Dummy	None (Transform)

Vp_Switch (Script)

Script	vp_Switch
Interact Type	Normal
Size	0
Interact Distance	3
Interact Crosshair	Interact
Interact Text	Press F
Delay Showing Text	1
Target	Door
Target Message	ActivateObject
Audio Source	None (Audio Sour
Switch Pitch Range	X 1 Y 1.5

We need to add a Vp_Switch script component to the object, and then define it by assigning the object that has a **LockActivator** component on it, and set the Target Message to "**ActivateObject**".


Vp_Switch (Script)

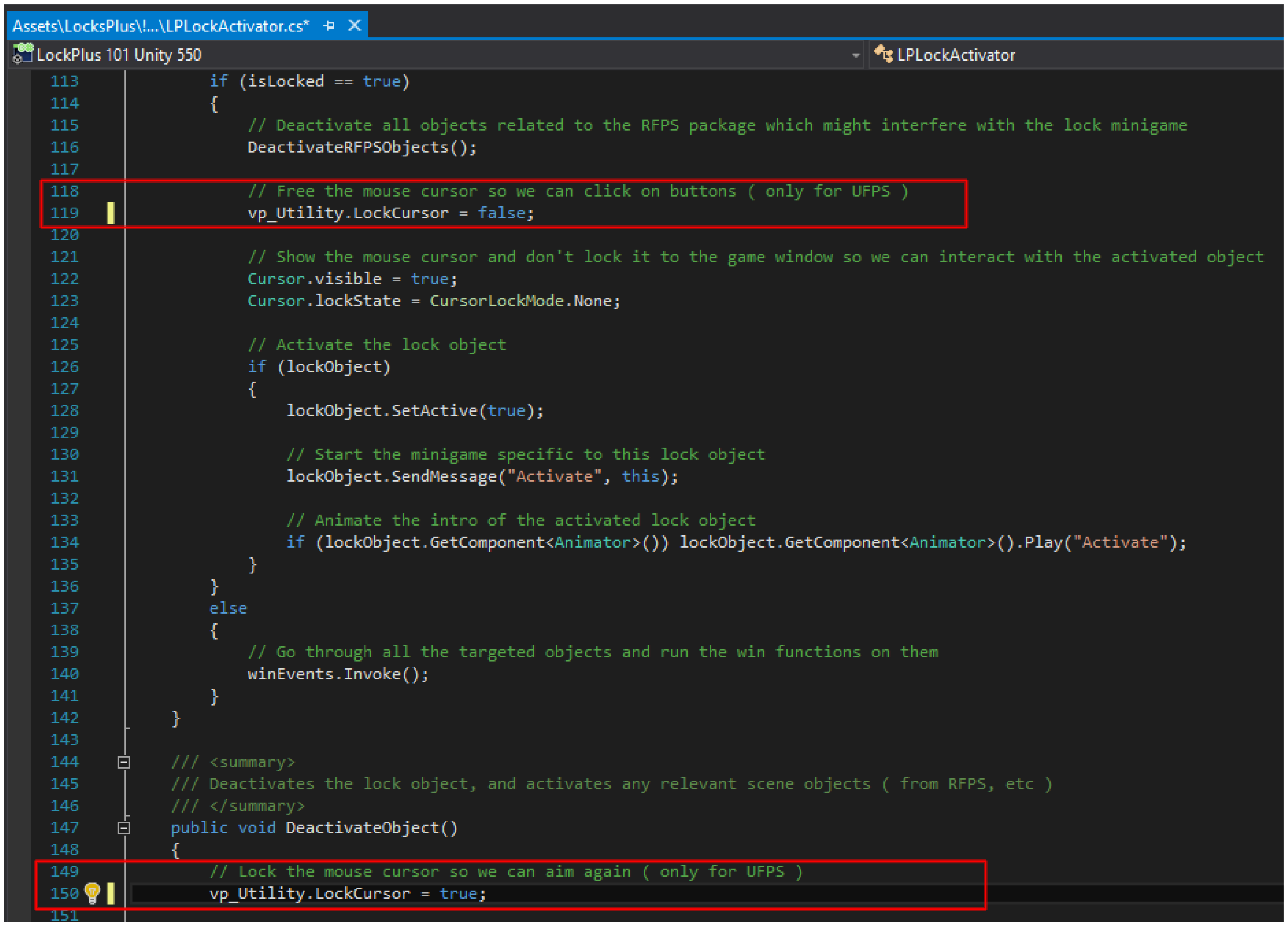
Script	vp_Switch
Interact Type	Normal
Recipient Tags	
Size	0
Interact Distance	3
Interact Crosshair	Interact
Interact Text	Press F
Delay Showing Te:	2
Target	BombLockEas
Target Message	ActivateObject
Audio Source	None (Audio So
Switch Pitch Range	X 1 Y 1.5
Switch Sounds	

Here's a closer look at the component setup. Make sure interact type is **Normal**, and interact distance is larger than 0 so you can activate it from afar.

You can also set a **Crosshair** and **Text**.

Don't forget to target the lock object itself so we can activate it.

Finally, we need to allow the player to interact with locks by enabling mouse controls in UFPS. To do that, open the script **LPLockActivator** and uncomment these lines:



```
        if (isLocked == true)
        {
            // Deactivate all objects related to the RFPS package which might interfere with the lock minigame
            DeactivateRFPSObjects();

            // Free the mouse cursor so we can click on buttons ( only for UFPS )
            vp_Utility.LockCursor = false;

            // Show the mouse cursor and don't lock it to the game window so we can interact with the activated object
            Cursor.visible = true;
            Cursor.lockState = CursorLockMode.None;

            // Activate the lock object
            if (lockObject)
            {
                lockObject.SetActive(true);

                // Start the minigame specific to this lock object
                lockObject.SendMessage("Activate", this);

                // Animate the intro of the activated lock object
                if (lockObject.GetComponent<Animator>()) lockObject.GetComponent<Animator>().Play("Activate");
            }
        }
        else
        {
            // Go through all the targeted objects and run the win functions on them
            winEvents.Invoke();
        }
    }

    /// <summary>
    /// Deactivates the lock object, and activates any relevant scene objects ( from RFPS, etc )
    /// </summary>
    public void DeactivateObject()
    {
        // Lock the mouse cursor so we can aim again ( only for UFPS )
        vp_Utility.LockCursor = true;
```

That should be all, test it out!