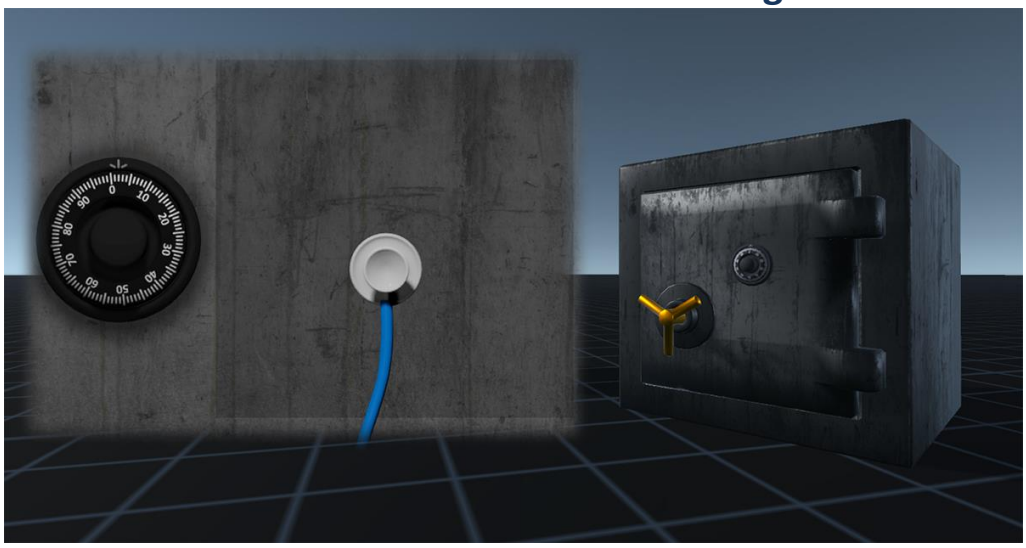
Locks Plus: Safe Cracking

Game documentation and HowTo guide.



This document contains:

Package Description and features

LOCKS+ Safe Cracking 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

PACKAGE INCLUDES

- A safe you need to crack by detecting the hotspot and listening for the dial as you turn it in the correct direction.
- A standalone demo showing off several examples of the Safe Cracking lock system, which also works on mobile.
- Video guide to show you the basics of the system.

Current version 1.05

Update history

1.05 (28.02.2018)
- Updated integration and documentation for Realistic FPS (RFPS) package and Ultimate FPS framework (UFPS).

1.01 (20.12.2017)
- Added support for Ultimate FPS (UFPS) package with quick integration guide.

1.0 (20.11.2017)
- Initial version

Please rate my file, I'd appreciate it 🙂

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:

- **Safe:** Holds all the assets related to the Dial Pad lock 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 objects for the door and bomb.
    - **Prefabs:** Holds all the lock examples we use in the demo, including 4 door examples and 3 bomb 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.
- **SafeCrackingDemo:** 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!)

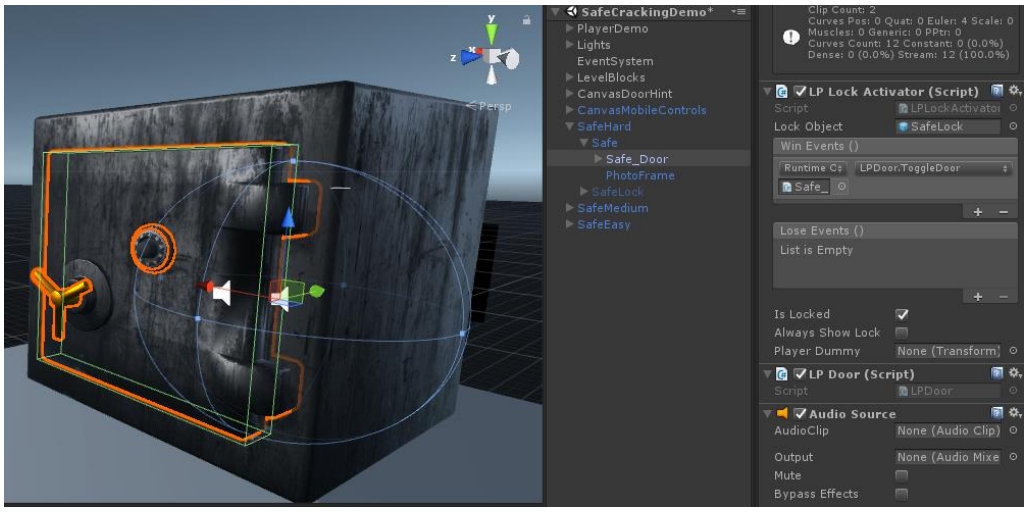
To start testing the locks, simply import the package into an empty project and open the demo scene SafeCrackingDemo. Here you can see several examples of the dial pad lock you can interact with.

There are two scripts that allow you to use the Dial Pad lock system, LPLockActivator, and LPSafe. 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.

Select one of the bomb locks and take a look at the LPActivator component assigned to it.



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

Win Events – A list of actions that will be triggered when we successfully unlock the lock. Here you can choose any function that is attached to this object.

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

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.

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, like the Big Red Button example.

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, like the Big Red Button.

LPSafe

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

Dial Right Button – The button that rotates the dial right.

Dial Left Button – The button that rotates the dial left.

Dial Rotate Speed – How fast the dial rotates left or right.

Dial Turns – How many correct dial rotation (until we hear the click) are needed in order to unlock this lock.

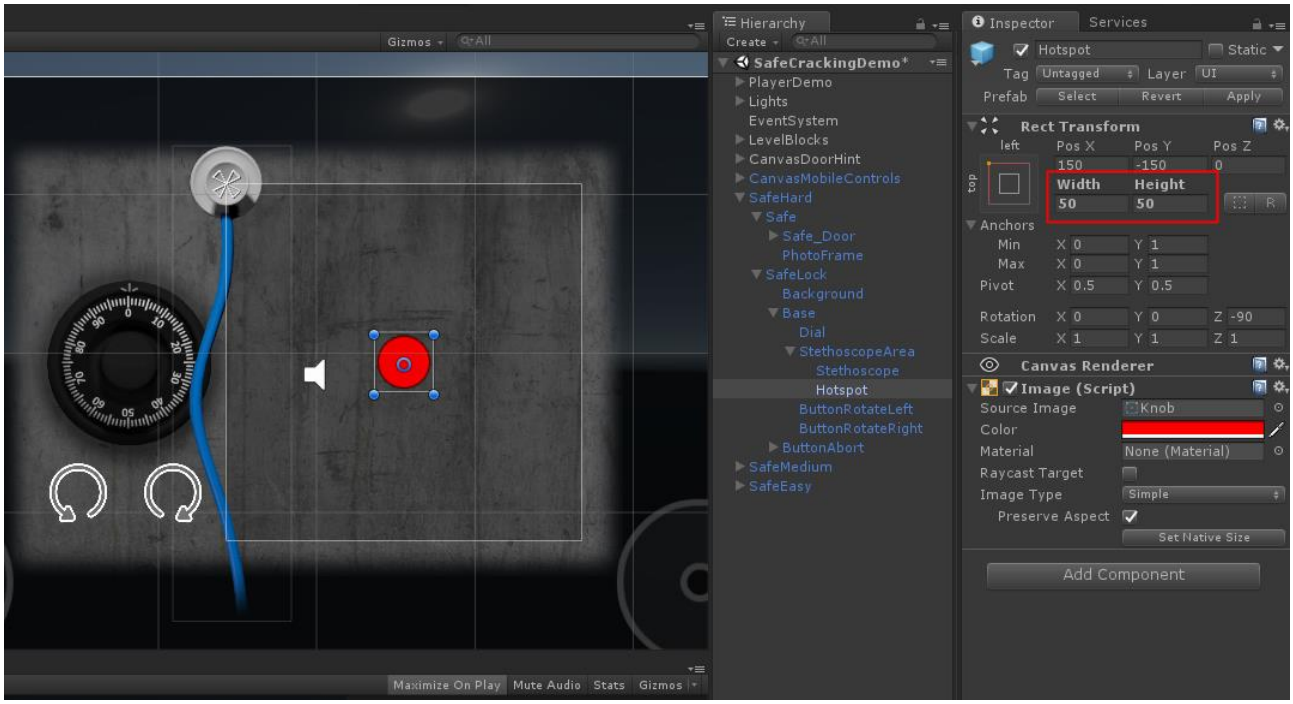
Min/Max Rotation – The rotation range needed to reach the correct point. For example is we may need to rotate to the right 100 degrees, or to the left 30 degrees until we hear the click.

Dial Reset – How far in the wrong direction must a dial rotate before the whole sequence is reset.

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

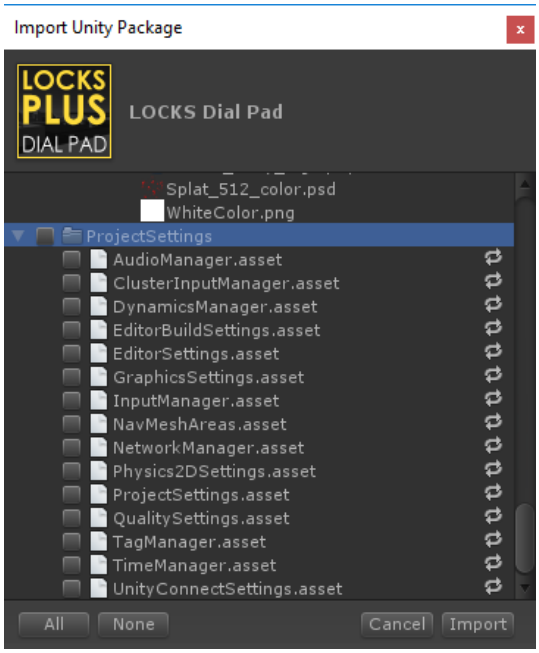
Sounds – Sounds for dialing, winning and losing.

Another thing you can set in the lock is the Hotspot. The hotspot decides how loud the sound of the rotating dial is. When we get closer to the hotspot with the stethoscope, the sound volume increases.



Integration with Realistic FPS Prefab

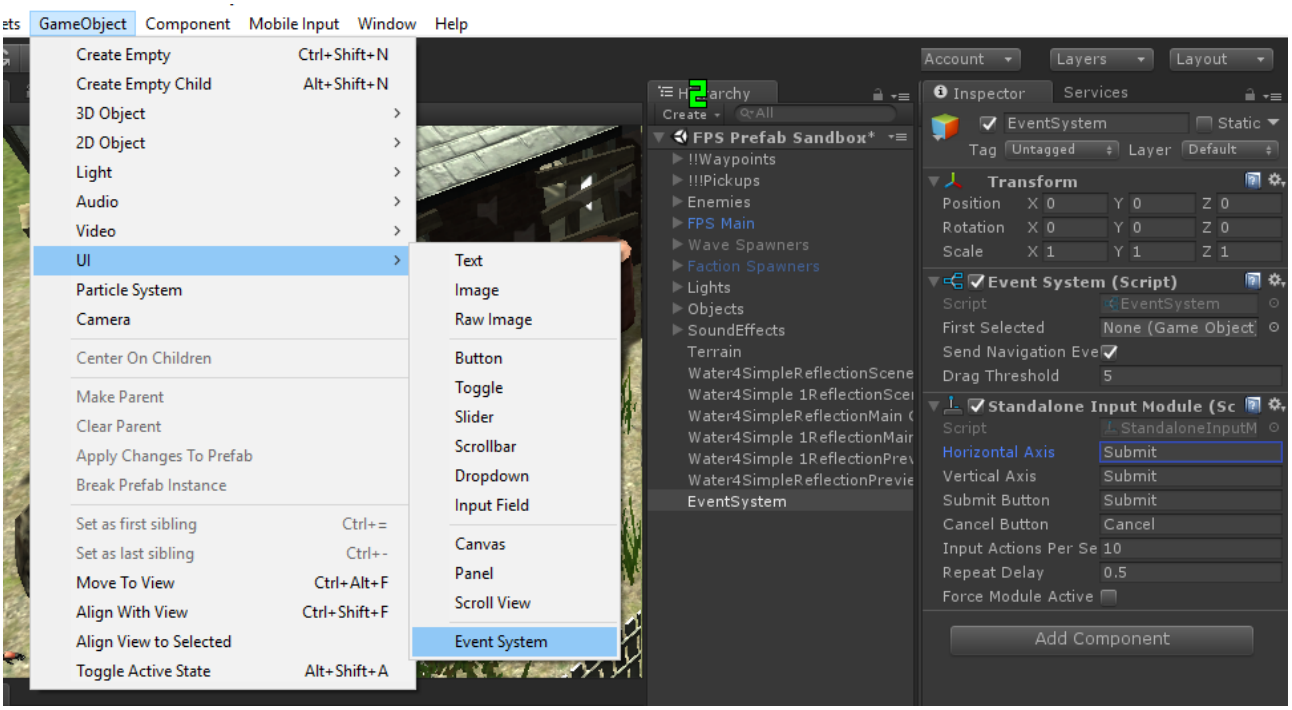
(When importing into an existing UFPS, RFPS, or other complete packages, don't import Project Settings!)



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.

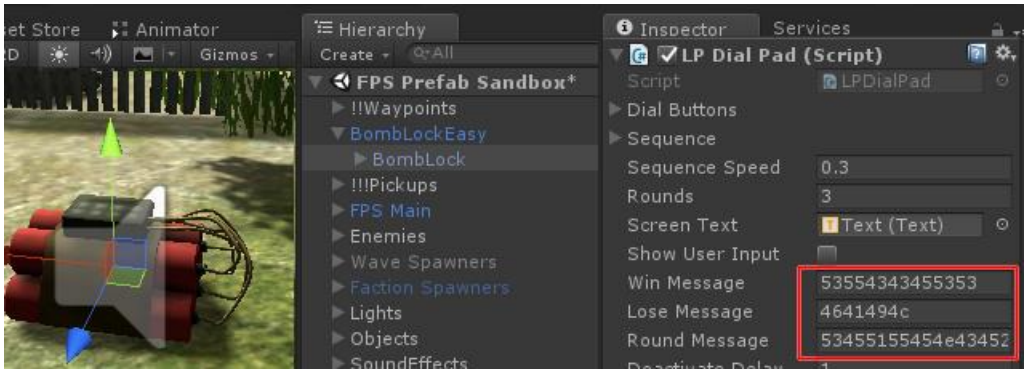
(This guide refers to the LOCKS+ Dial Pad package, but it works the same way in all LOCKS+ packages)

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.

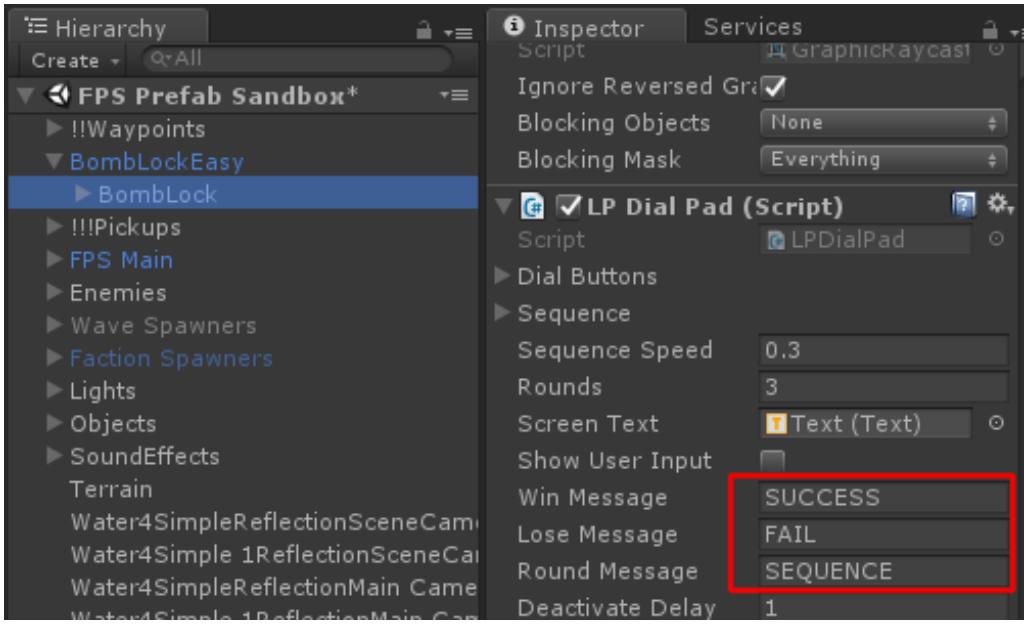


Next thing is to set the tag of your lock Activator to "Useable" which allows RFPS to interact with objects.

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:



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



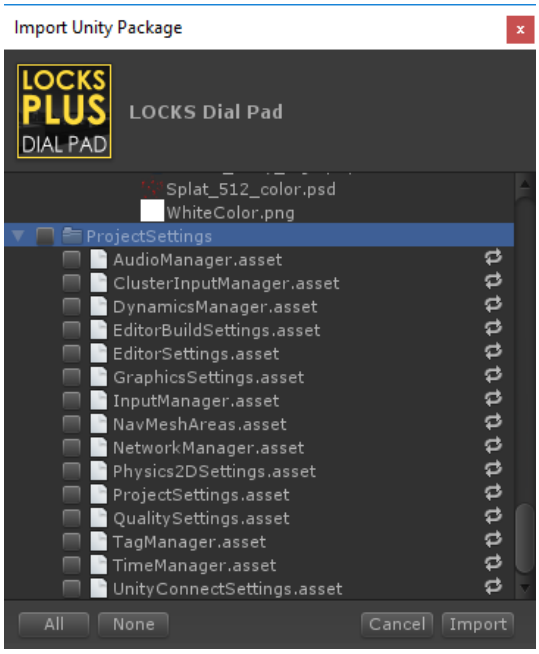
If you test the game now you'll be able to interact with the door, play the dial pad lock, unlock it and then open and close the door freely.

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

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

Integration with Ultimate FPS framework

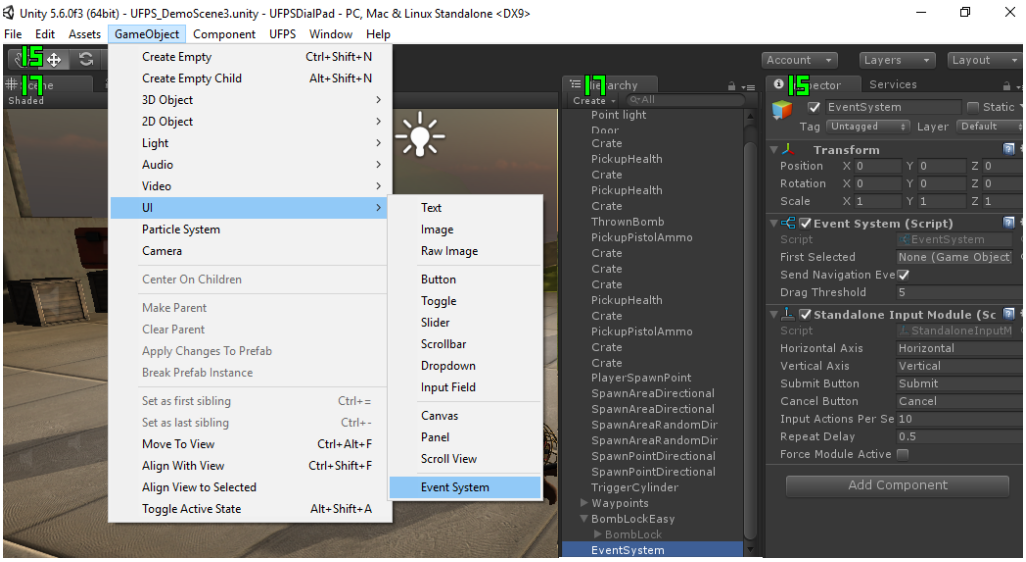
(When importing into an existing UFPS, RFPS, or other complete packages, don't import Project Settings!)



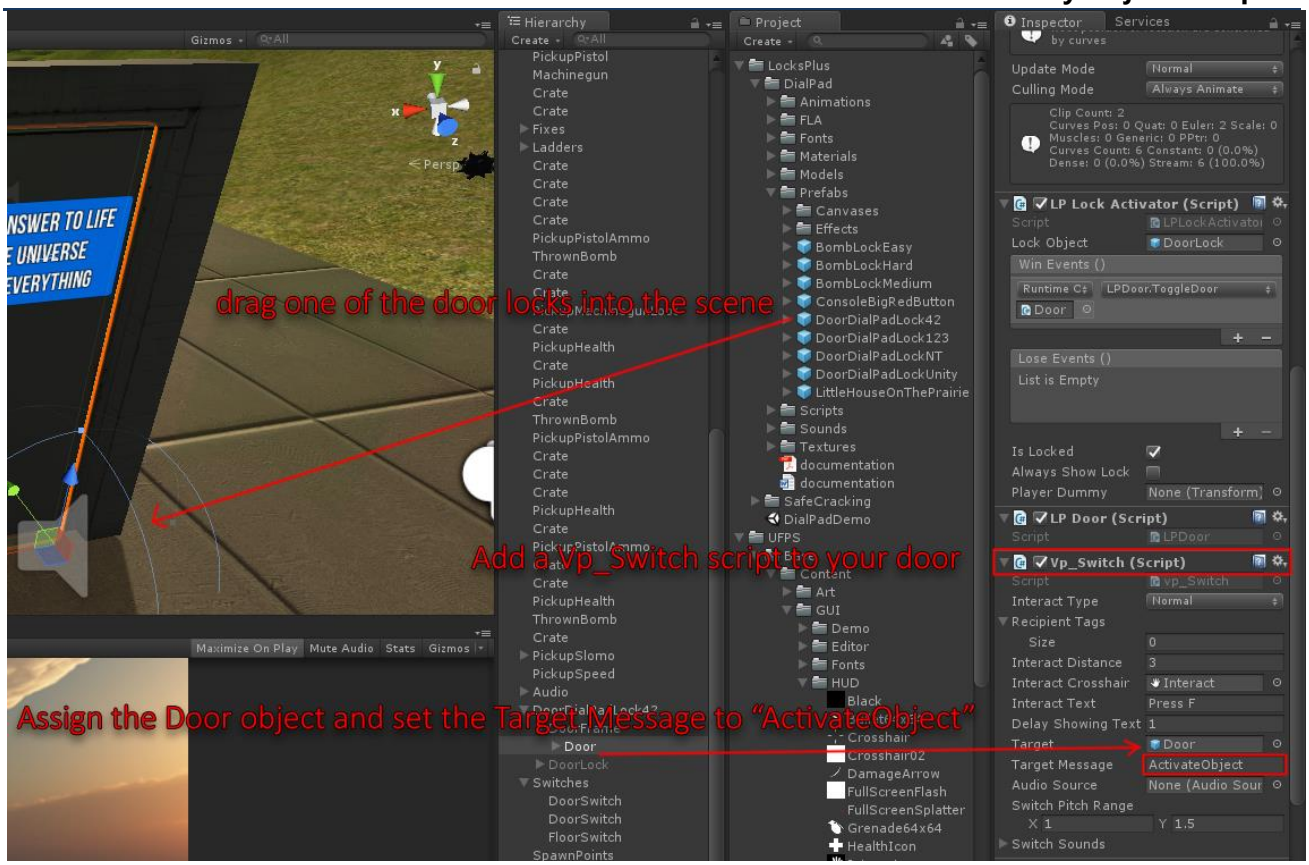
Import into your project the Ultimate FPS package (tested with 1.7.2), open the UFPS_DemoScene3 scene from the UFPS project, and drag one of the lock examples to the scene. In order to make it work with UFPS you need to set a few basic things.

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

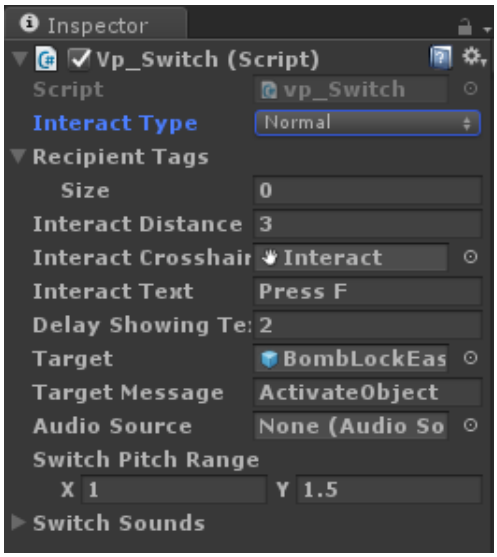
Add an EventSystem object to the scene. LOCKS+ is based on Unity UI and event system so we need that to make the buttons clickable.



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



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**".

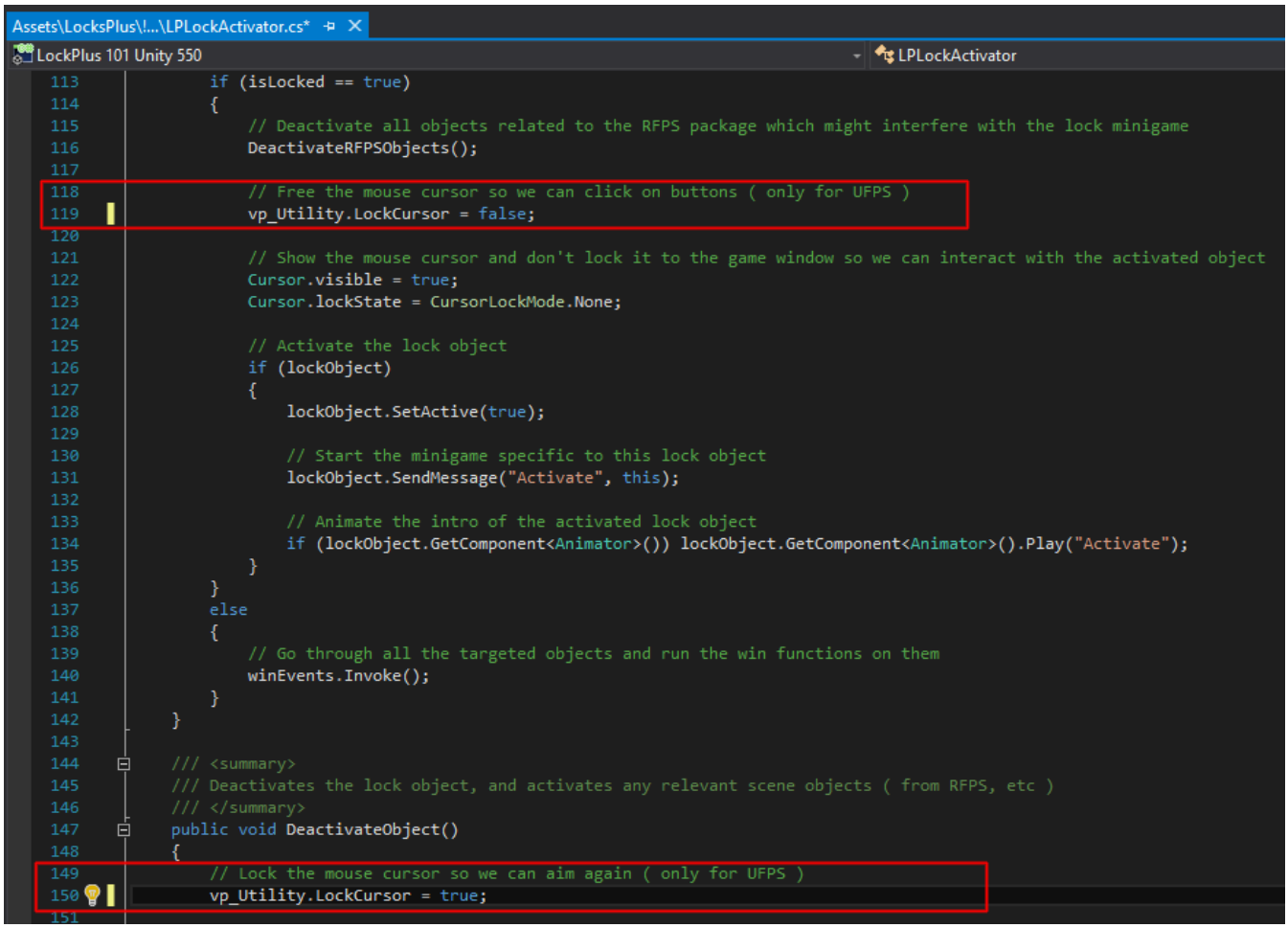


Here's a closer look at the component setup. Make sure interact type is **Normal**, and interact distance is larger than 0 so you can activate it from afar.

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:



That should be all, test it out!