THANKS FOR BUYING HORROR HIDE & SEEK KIT!

Horror Hide And Seek is a template that will help you create your own horror game for mobile devices!

The template includes the basic functions of the player, mobile controll, enemy AI, hide places, ready demo scenes, xml save system and much more!

Summary

1) About kit.....

3) Features.....

4) Project Setup.....

5) Build Scene.....

6) Saviable objects.....

7) Add new item.....

9) Door script.....

10) Saviable objects.....

11) Setup new Enemy.....

Features

System functions

- XML save system (save all dynamic objects in the scene)
- Enemy calling system (sound of an object falling attracts the attention of the enemy)
- Dynamic objects (door, drawer)
- Sliding puzzle system
- Pause menu, simple options configuration (volume, sensitivity)

Player Functions

- Player controller (Walk, crouch)
- Foot steps
- Player breaks legs when falling
- Hiding from the enemy in cabinets, under the bed, in a chest

Enemy AI

- Patroll way points
- Hears the sounds of falling objects
- Chases and loses a player
- Interacts with Hide places (pulls the player out from under the bed, chest)
- Player strike animation
- Have FOV and see distance

Hide places

- Hide from enemy under bed, in chests and in the lockers

Traps and Sekrets

- False floor (player break legs when fall)
- Puzzle picture with event (Do what you need after solving puzzle)

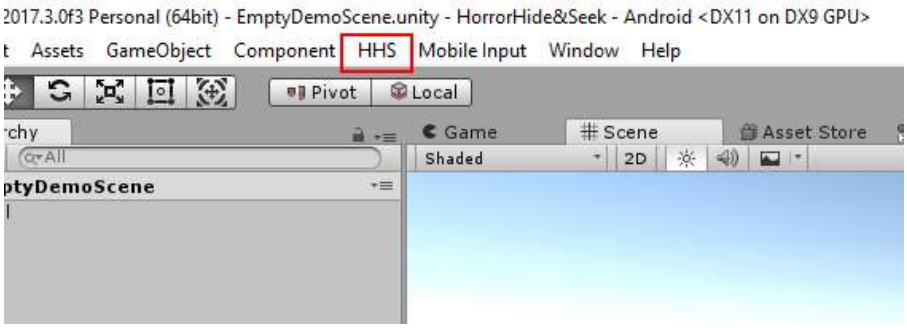
Other

- Simple house modular set(walls,floor,ceiling,stairs room), furniture and other

Project Setup

1) Import asset in to a empty project.

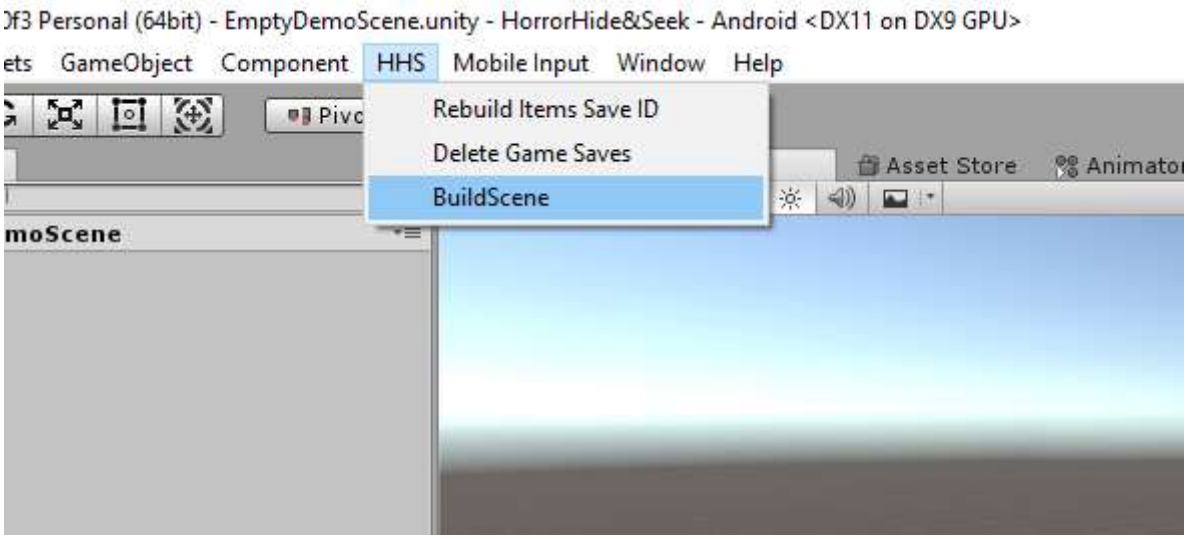
2) After importing asset you can setup you own scene with a new menu button



3)Create a empty scene or open ready empty scene (HHs project/Scenes/EmptyDemoScene)

4)If you create new scene you need create floor and bake navigation mesh.

5) When everything is ready go to HHS menu (HHS/Build Scene) and press press Build scene.

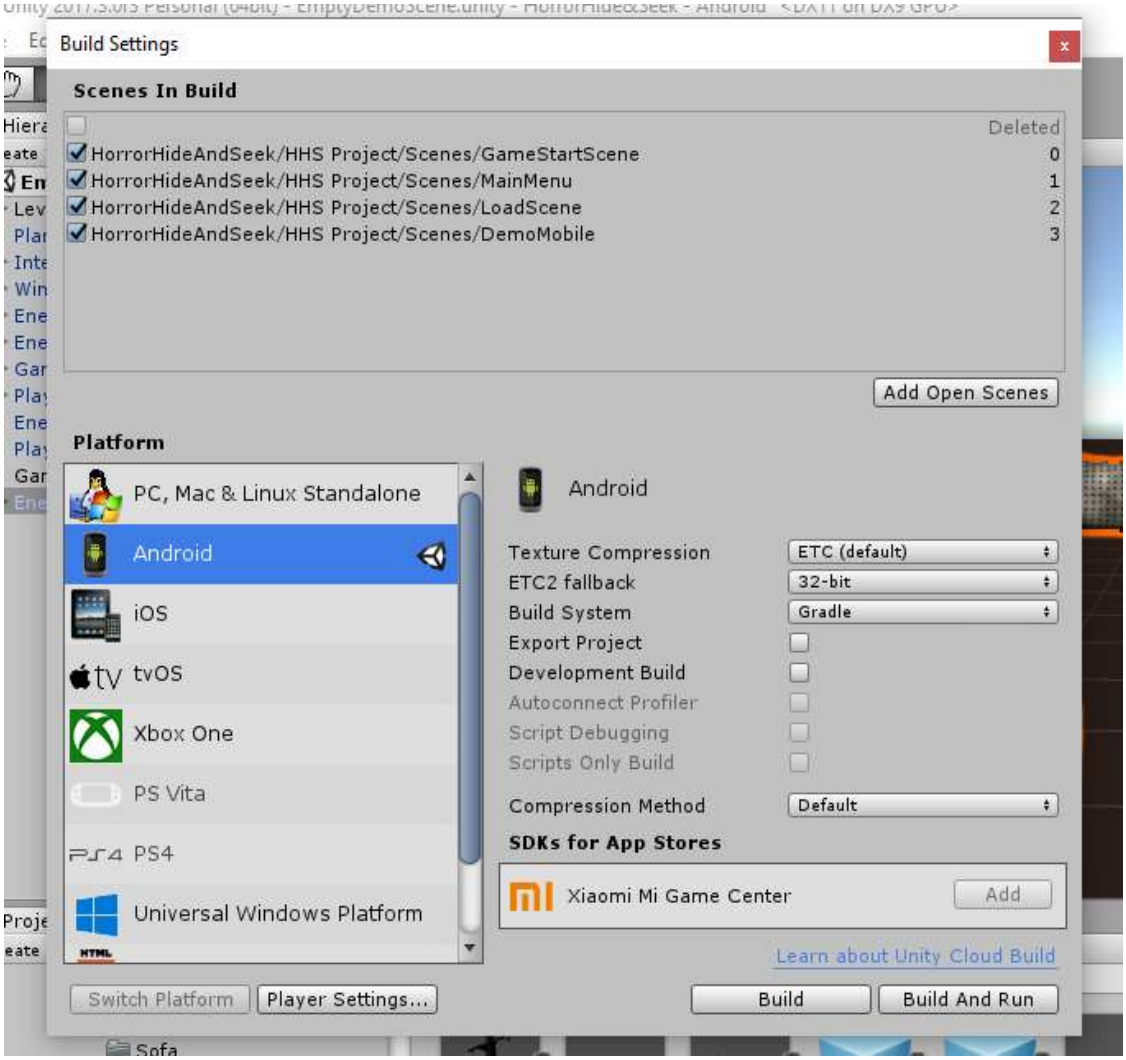


6) Your scene is ready!

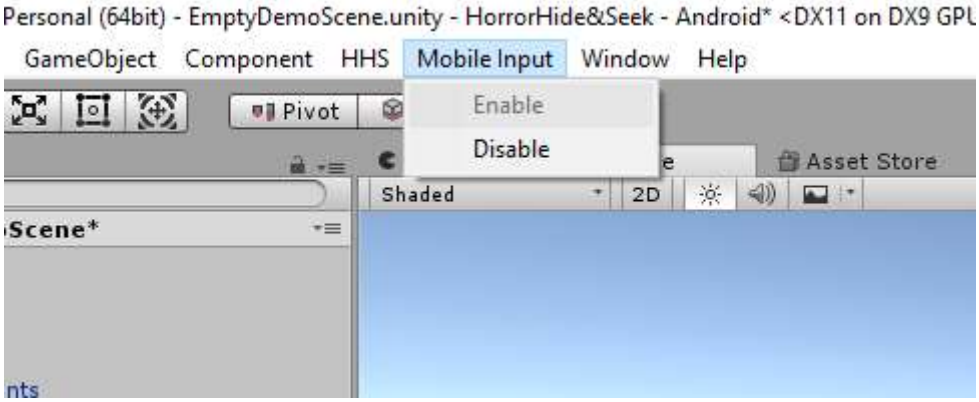
7) Now you can place objects in the right places and add prefabs of furniture and other items.

Build Scene

1) To build your project for the Android platform, you need to download all the necessary libraries (Android SDK and others)
Also, after importing the project, you will need to switch to the platform you need (android etc).

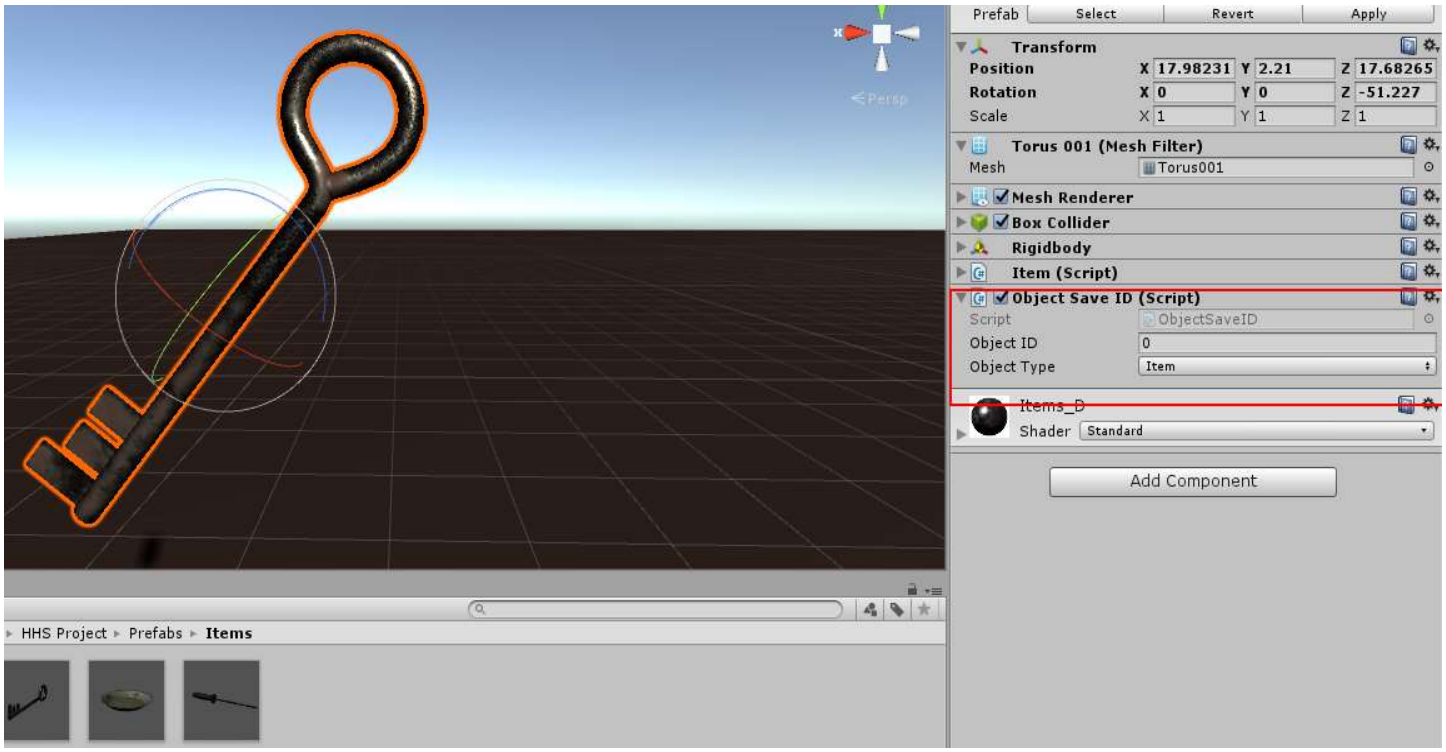


Also, before building the project, be sure to **enable Mobile Input** in the top menu.



Saviable objects

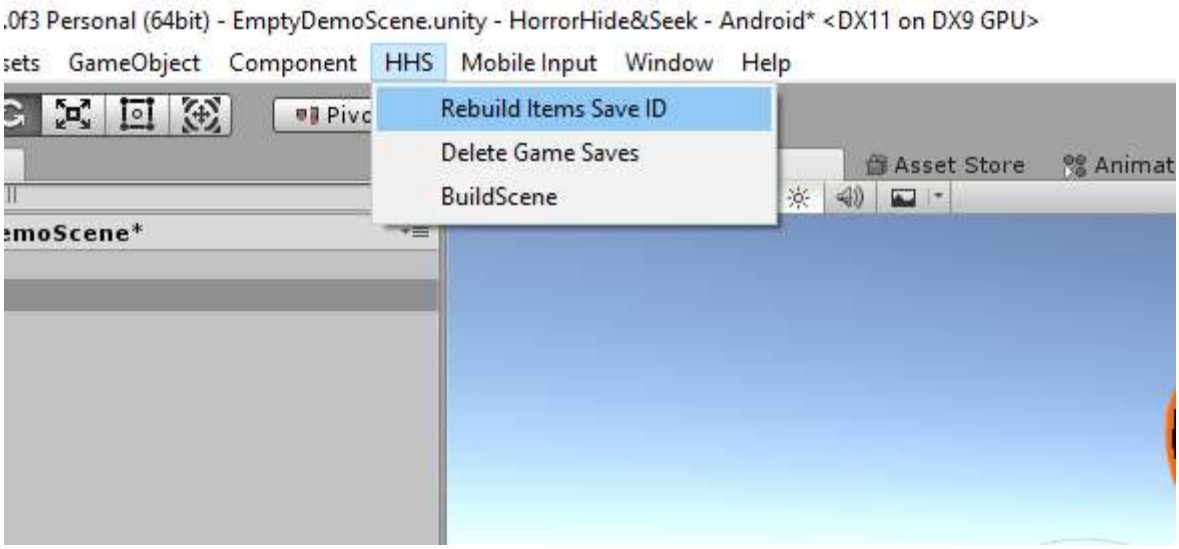
1) All saviable objects have special script **ObjectSaveID.cs**



2)How you can see, script have **Object Type** parameter. (Item, Door etc)

3) You must assign each type of object its type (if it is a door, then the **Object Type** must be **Door,** if item - **item**)

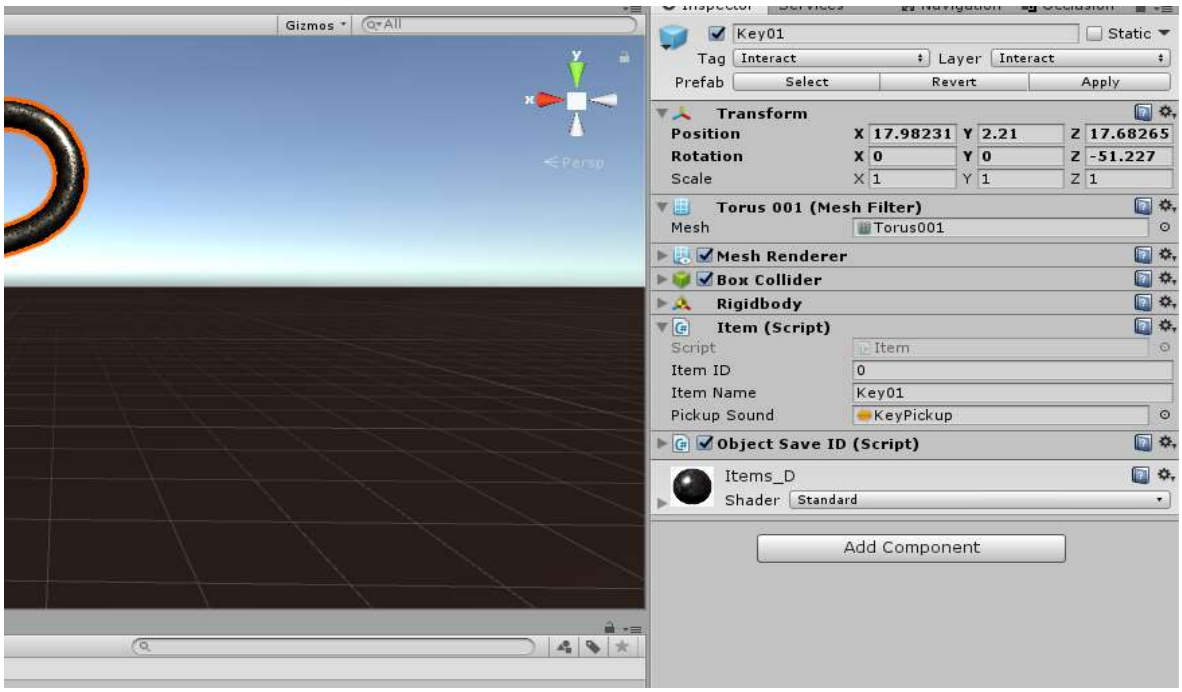
4) You don't need assing Object ID for every new object. After you add all objects to the scene go to HHS/Rebuild Item Save ID.



5)This **must** be done after each saved item is added to the scene!

Add new item

1) After you have added the model of your object to the scene, you need to do the following: Change object tag to **Interact** tag, Layer to **Interact** layer, add rigidbody and collider to object, add **Item.cs** script and **ObjectSaveID.cs**. Change **Object Type** value on **ObjecSaveID** script to **Item.**



2) Script **Item.cs** have next parameters:

Item ID - *must be individual for every item*

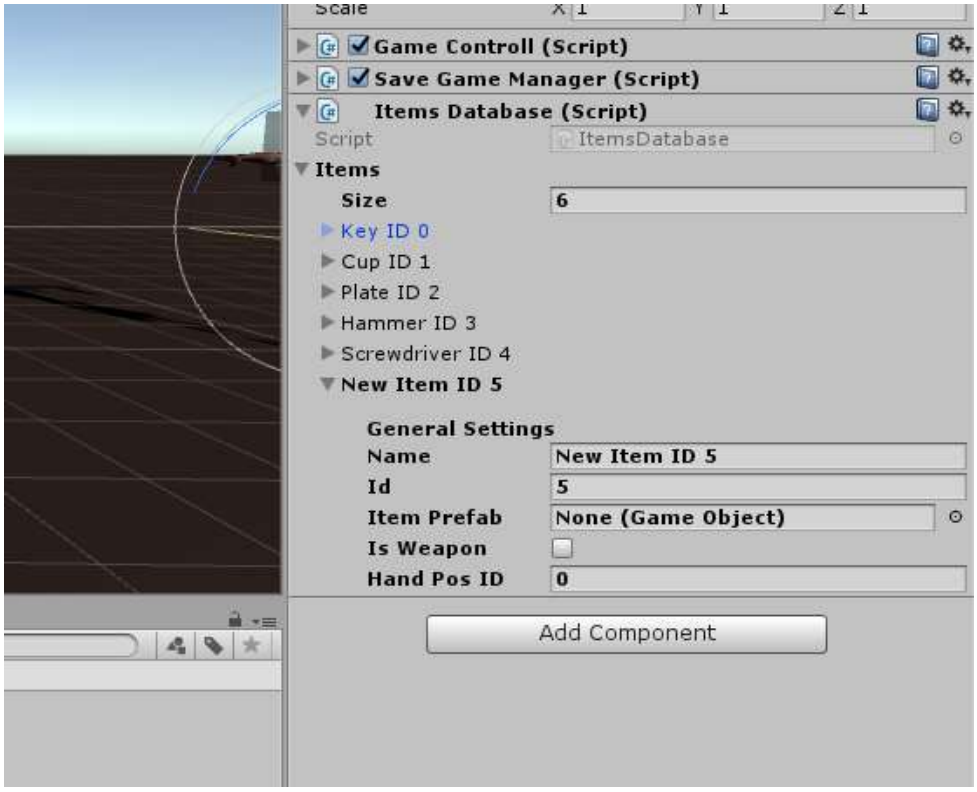
Item name - *name of the item*

Pickup sound - *the sound that will be played when the player picks up an item*

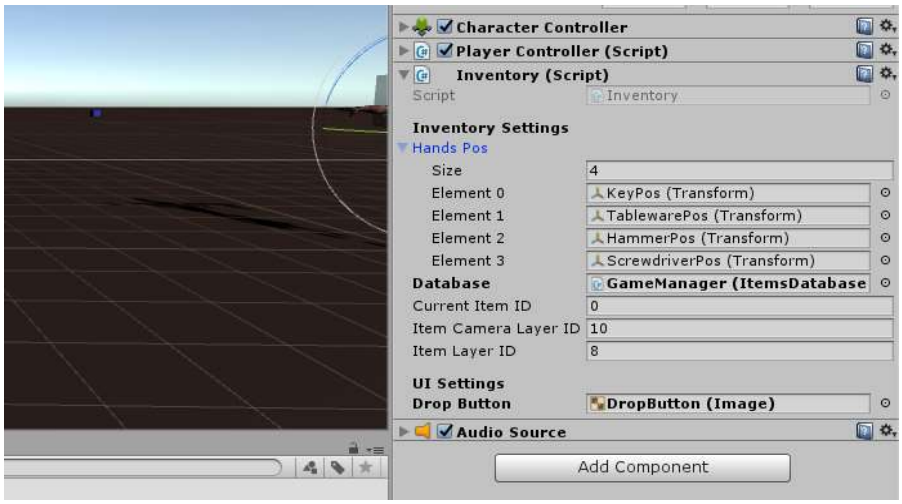
3) To know the free ID for a new item, find the **GameManager** object on the scene and you find script **ItemDatabase.cs**

4)The ID of your new item must be greater than the ID of the last item.

5) Change the Item size to 1 in the script and write the required data. You also need to create and add a prefab to your new item.



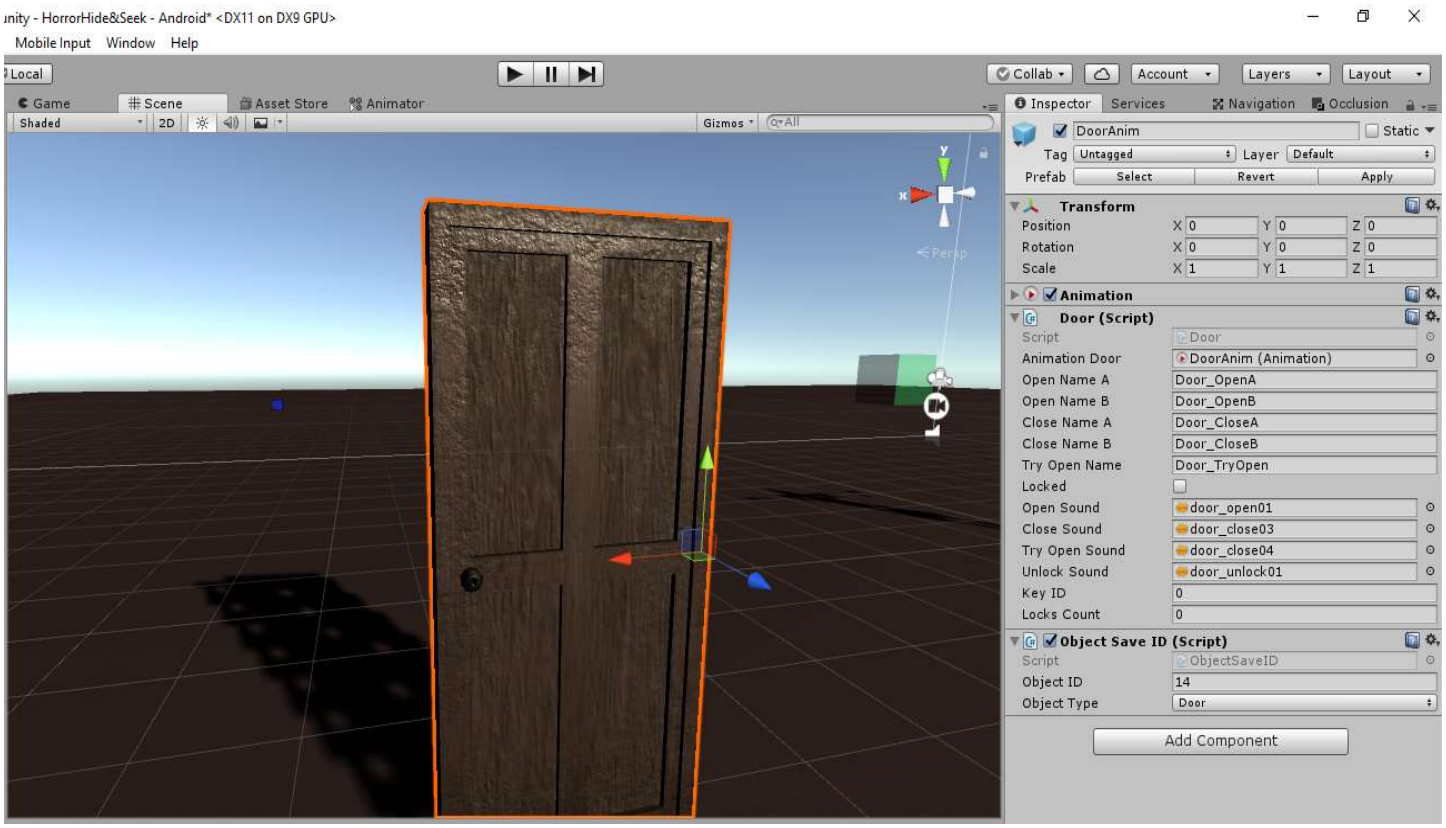
6) How you can see script also have value **Hand Pos ID**. This value controls the transfrom, in which the item will be placed after the player picks up the item. To add a new transform, find the Player object on the scene. There will be an **Inventory.cs** script on this object.



7) **Hands Pos** have list of transforms. For example, Hand pos ID 0 this is keyPos transform. You can add new tranfrom in sctipt and write id value in **Item Database** script.

Door script

1) All doors have next scripts: **Door.cs, ObjectSaveID.cs**

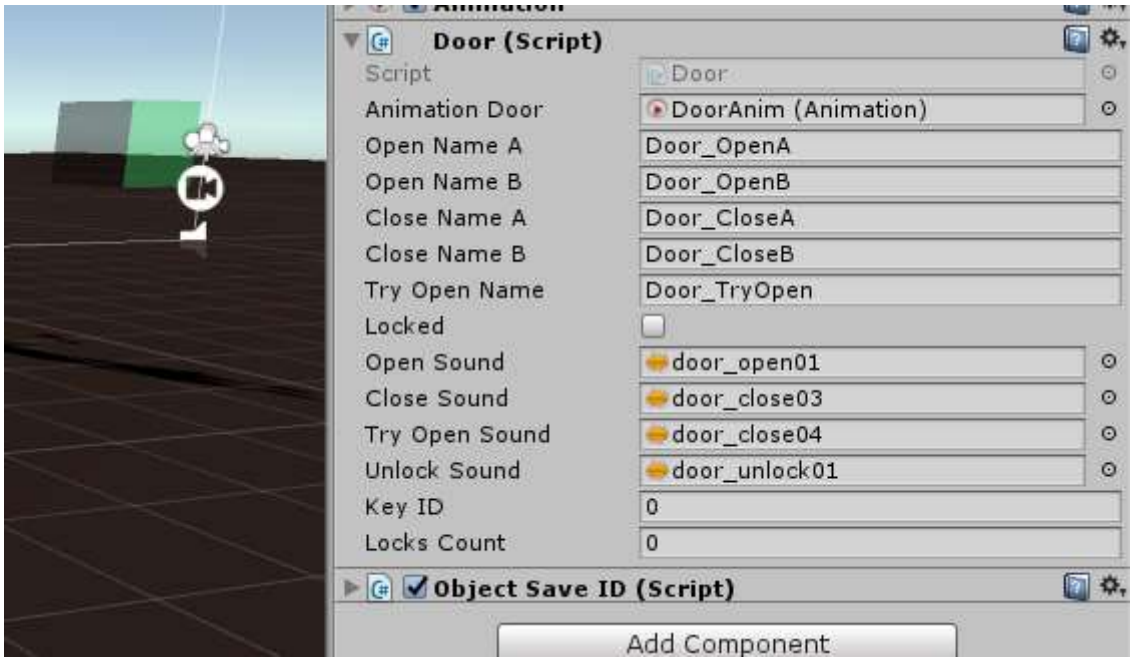


2) Also door have 2 child gameobjects with **DoorSiders.cs** scripts.

3) Why do we need these 2 objects? *The fact is that in many projects when AI opens the door, it opens only in one direction, which leads to the fact that the enemy passes through the door*. To avoid this, HHS project was implemented opening doors on both sides. One object have **Door SIde ID 0**, other **ID 1**. When AI open door from first object, door open with 0 value, when AI open door from other side, door opening with 1 value with other animation. This way the door will always open from AI.

4) Also, door have **ObjectSaveID.cs** script with object type value **Door.**

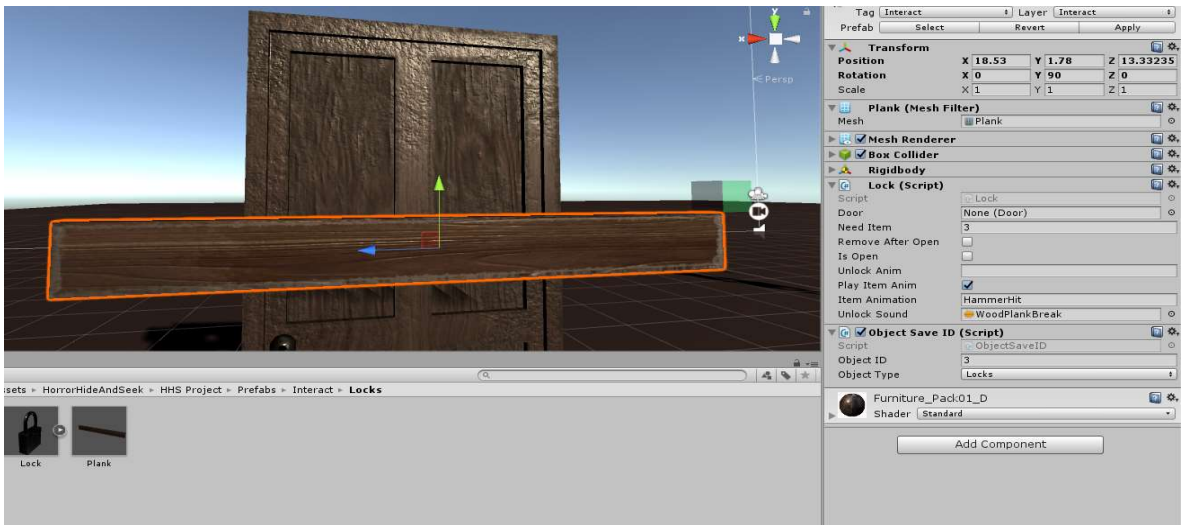
5) The door can be locked with a key, as well as locked with locks or boards.



6) If door locked by key, you must set value **Locked** true, and set **Key ID Key id** must be equal to the id of the key object from **Item.cs**

7) If the door is locked with locks or boards, then the value of **Locks Count** should be equal to the number of locks or boards.

8) To add a lock or a board to the door, place the prefab on the scene and configure it:



9) In the **Lock.cs** script, add your door to the **Door** parameter. Need item is the id of the item that will unlock the lock or board. If the value **Play item Anim** = **true**, then when interacting with the lock, the item will perform an animation (for example, the hammer will perform a hit animation) If value **Remove After Open** = true, then the item will be deleted after interaction (for example, key)

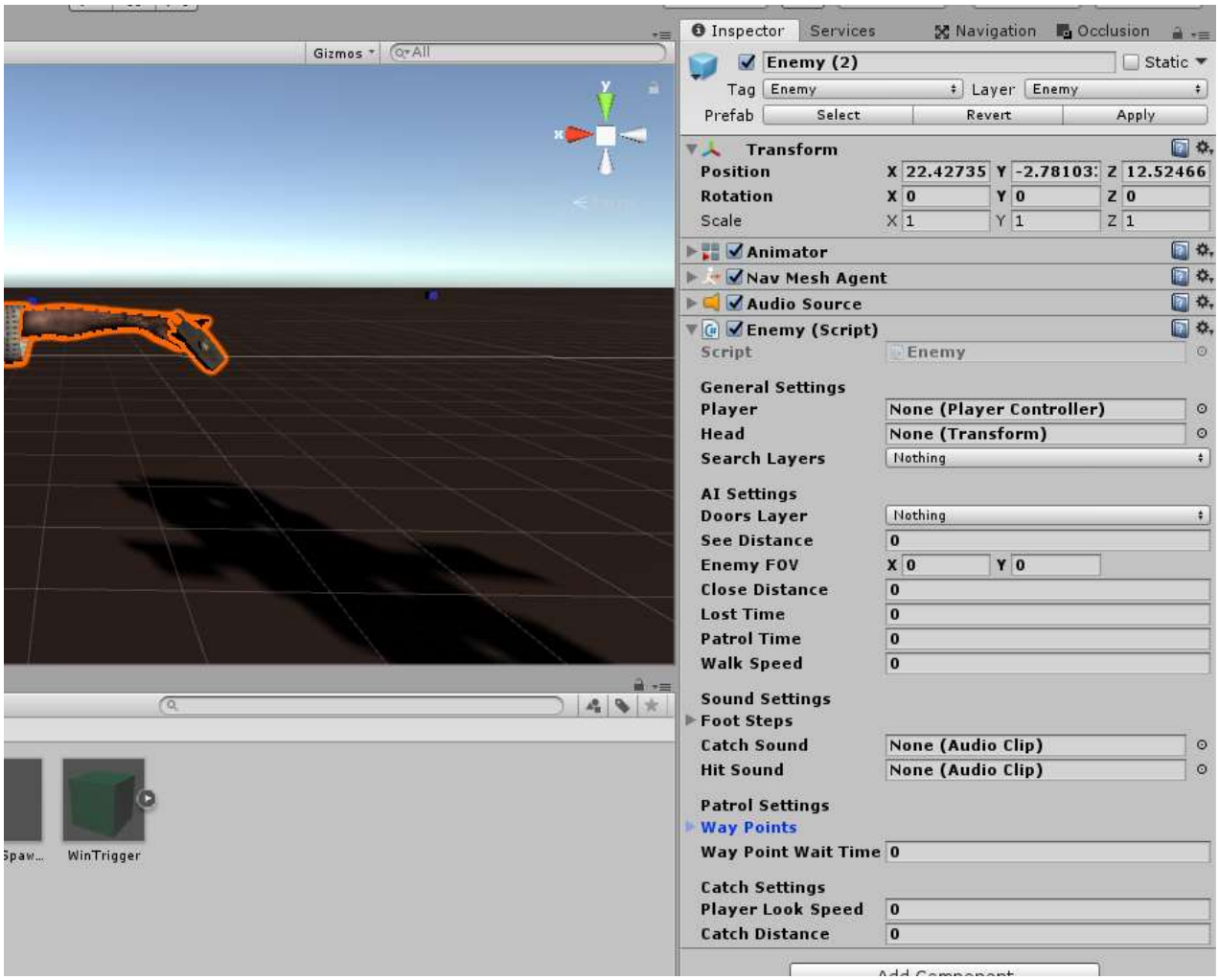
Setup new enemy

1) To replace the enemy model, you need a model with Humanoid rig. Place the model on the scene and set up tags and layers: Tag - **Enemy**, layer - **Enemy.**

2) Add **EnemyController** to Animator.

2) Add **Nav Mesh Agent** and **Audio Source** components to object.

3) Now you can add **Enemy.cs** script to object and configure script:



- Put **Player** gameobject in to **Player** value;
- Find head bone of your new model and put it to **Head** value;
- Setup search layers (Default, Player, Interact);
- Setup door layer (Interact);
- Create some empty objects and place them at ground level. Then put these objects in **Way Points** list;
- Configure Enemy FOV (default is X -80, Y 80)
- Configure other values (see distance, walk speed, sounds and other);

Credits

This asset developed by ***AlexeyCrayne.*** All Rights Reserved.

Almost all assets are created by ***AlexeyCrayne.*** Some assets downloaded with royalty free license.

Links

https://assetstore.unity.com/publishers/41127 Assetstore

https://voronalexeydakurlz.wixsite.com/alexcrayne Website