A Simple Virtual World in Unity

# Introduction

This report documents the development of a simple virtual world using the Unity game engine. Creating a virtual world in a game engine like Unity allows us to visualize how a user can interact with a 3D environment and how the simulation of physics and textures enhance the user experience of immersion and reality.

A two storey house has been constructed using Unity standard assets and imports from the Unity asset store and internet. The virtual reality world is focused on the building and its interior textures and objects. The house resides on a plane terrain with a spotlight to represent some form of a sun so that we can see the projection of believable shades and textures around the interior and exterior of the house.

# Implementation

**Figure.1**



Figure.1 displays the house that has been created. There are some features to the exterior structure of the house, a front and back door, multiple windows and an unfinished attempt at a roof. The bottom of floor of the house consists of a living room which the user will enter through the front door, a dining room to the left and a kitchen to the back of the house. As the bottom floor of the house demonstrated the relevant tools and techniques of this assignments scope and time a dwindling factor the top floor. With a short time scale on this assignment the relevant tools and techniques needed to award merit have been demonstrated on the bottom floor of the house. A Top floor consisting of rooms and a roof had begun development but not finished.

## Character Navigation

The user sees the world through a first person view and controls the character with the following keys and mouse and operations.

↑ = forwards = jump = action

Space

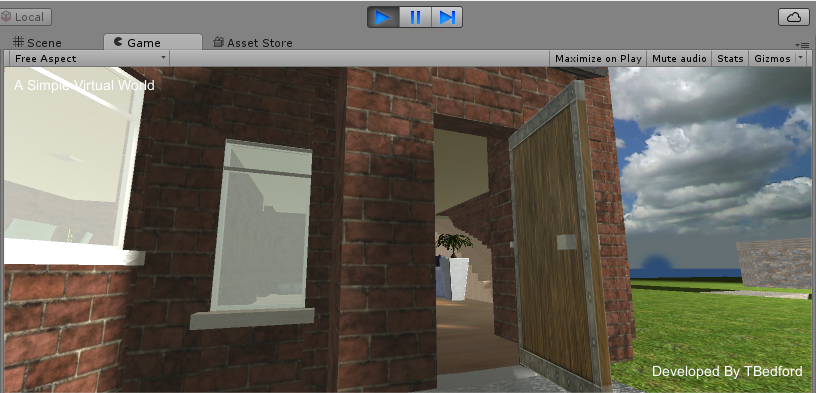
↓ = backwards

← = side step left

→ = side step right = mouse movement to look

Clicking the left mouse button anywhere on objects that can be interacted with such as the front and back door to the house and light switches that are found on the bottom floor of the house.

The C# scripts that has been written to animate the lights and doors are referenced in the Appendix. Figure.2 displays the front door of the house after the user’s mouse click has triggered the open door script. After entering the house there is an interactive light switch on the right wall. There is a window just round the corner to the right which is allowing some of the sun objects light in casting a realistic shadow on the floor which can be seen in Figure.3



**Figure.2**

. **Figure.3**

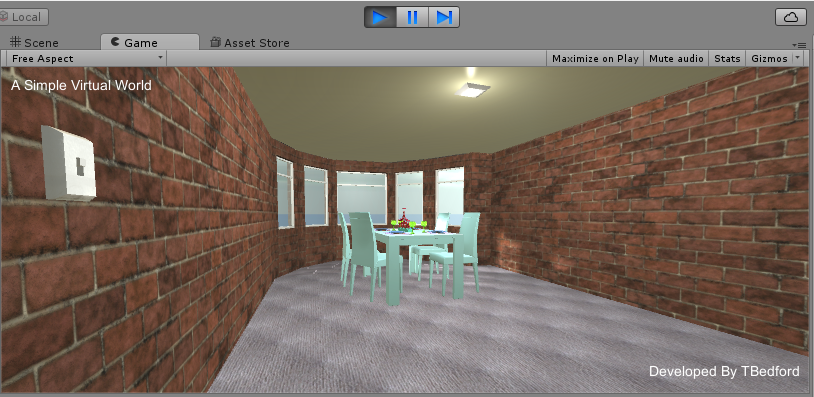


Figure.4 illustrates the lighting with the switch on. This is a simple script where the switch position is moved up and the spotlight centred with the light plate is enabled.

Figure.5 shows the dining room to the left of the house and 6 the kitchen to the back.

**Figure.4**

**Figure.5**



**Figure.6**

The next couple of figures 7, 8, 9 and 10 display the 3D models that have been imported into the scene and placed around the house. These items are not interactive, they are to relate to typical objects that one might come across in the average home. The use of these models enhances the users feeling of being inside a home. The objects have had box colliders added to simulate there physical properties, well at least there physical presence to the user.

**Figure.7** **Figure.8**

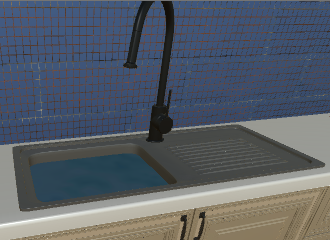


**Figure.9** **Figure.10**



Observed over the images in this document you can see the use of materials such as the brick walls, wooden flooring, glass windows and carpeted dining room that create the reality of real objects. Some of these materials like the brick texture have a height map improving the visual 3D effect of the material.

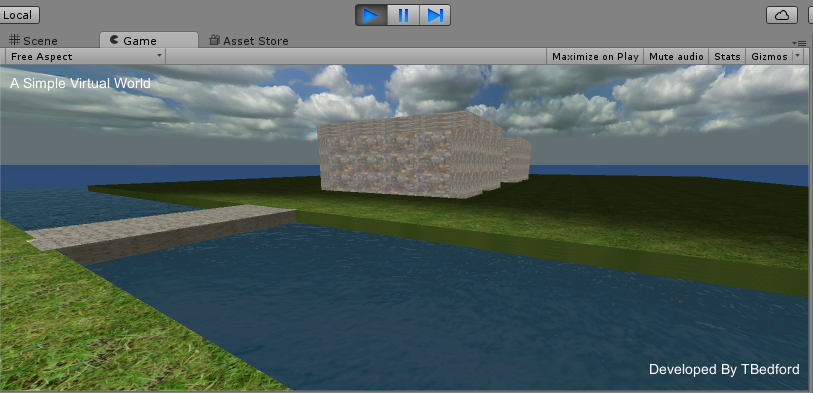
**Figure.11**



The kitchen sink seen in Figure.11 looks as though it is filled with water. Imported material scripts from the unity asset store where applied to add a material to the sink that simulated the effect of water. The script used and its source has been referenced below at the end of this document.

The name of application ‘A simple virtual world’ is anchor and displayed at the top left of the screen. The developers name is in the bottom right corner.

A quick attempt at a first civilization remains in the scene over a bridge and river from the main house as you will see when running the game. Here was the testing ground for some basic Unity tutorials.



**Figure.12**

# References

Unity asset store:

[1] Youtube, Creating glass with Hard Surface Shader, available at: <https://www.youtube.com/watch?v=rXNv9EPHyHw>

[2] Youtube, Jimmy Vegas, a simple world tutorials, available at: <https://www.youtube.com/watch?v=oFOtgvO-5R8&index=3&list=PLZ1b66Z1KFKhez_bV_DuR3EKENzXJTgAe>

[3] Youtube, Light Switch, available at: <https://www.youtube.com/watch?v=JD_dqoYGFfc>

[4] Youtube, Clickable Door, available at: <https://www.youtube.com/watch?v=scm7r0uBepU>

[5] Unity Community, images marked as legacy debug, available at: <http://answers.unity3d.com/questions/577801/animation-must-be-marked-as-legacy.html>

# Appendix

1. **Door.cs**

using UnityEngine;

using System.Collections;

public class Door : MonoBehaviour {

private int \_mLastIndex;

public void PlayDoor()

{

if (!GetComponent<Animation>().isPlaying)

{

if (\_mLastIndex == 0)

{

GetComponent<Animation>().Play("DoorOpen");

\_mLastIndex = 1;

}

else

{

GetComponent<Animation>().Play("DoorClose");

\_mLastIndex = 0;

}

}

}

}

1. **InputHandler.cs**

using UnityEngine;

using System.Collections;

public class InputHandler : MonoBehaviour {

public Camera camera;

// Update is called once per frame

void Update () {

if (Input.GetMouseButton(0))

{

Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);

RaycastHit rayCastHit;

if (Physics.Raycast(ray.origin, ray.direction, out rayCastHit, Mathf.Infinity))

{

Debug.Log("Mouse Click!");

Door door = rayCastHit.transform.GetComponent<Door>();

if(door)

{

door.PlayDoor();

}

}

}

}

}

1. **LightSwitch.cs**

using UnityEngine;

using System.Collections;

public class LightSwitch : MonoBehaviour {

public Light light;

public GameObject lightswitch;

public AudioClip switchOn;

public AudioClip switchOff;

private bool \_switchOn;

private GameObject \_upPosition;

private GameObject \_downPosition;

//private AudioSource \_audioSource;

void Awake()

{

\_upPosition = GameObject.Find("Switch\_Up");

\_downPosition = GameObject.Find("Switch\_Down");

}

void OnMouseDown()

{

if(\_switchOn)

{

\_switchOn = false;

lightswitch.transform.position = new Vector3(lightswitch.transform.position.x, \_downPosition.transform.position.y, lightswitch.transform.position.z);

light.enabled = false;

}

else if(!\_switchOn)

{

\_switchOn = true;

lightswitch.transform.position = new Vector3(lightswitch.transform.position.x, \_upPosition.transform.position.y, lightswitch.transform.position.z);

light.enabled = true;

}

}

}

**SimpleWater.cs**

using System;

using UnityEngine;

namespace UnityStandardAssets.Water

{

[ExecuteInEditMode]

public class WaterBasic : MonoBehaviour

{

void Update()

{

Renderer r = GetComponent<Renderer>();

if (!r)

{

return;

}

Material mat = r.sharedMaterial;

if (!mat)

{

return;

}

Vector4 waveSpeed = mat.GetVector("WaveSpeed");

float waveScale = mat.GetFloat("\_WaveScale");

float t = Time.time / 20.0f;

Vector4 offset4 = waveSpeed \* (t \* waveScale);

Vector4 offsetClamped = new Vector4(Mathf.Repeat(offset4.x, 1.0f), Mathf.Repeat(offset4.y, 1.0f),

Mathf.Repeat(offset4.z, 1.0f), Mathf.Repeat(offset4.w, 1.0f));

mat.SetVector("\_WaveOffset", offsetClamped);

}

}

}