HOLOLENS

HoloToolkit

- The HoloLens comprises of three processing units, the CPU + GPU + HPU.
- 2 Speakers - Present above each ear.
- Inertial Measurement Unit (IMU) - Consists of : accelerometer, gyroscope, magnetometer. The IMU is used to track movements.
- A laser camera and laser project are located at the front of the HoloLens, they help in mapping out the local environment.

The Unity platform will be used for developing applications for the HoloLens.

https://github.com/Microsoft/HoloToolkit-Unity

https://unity3d.com/learn/tutorials

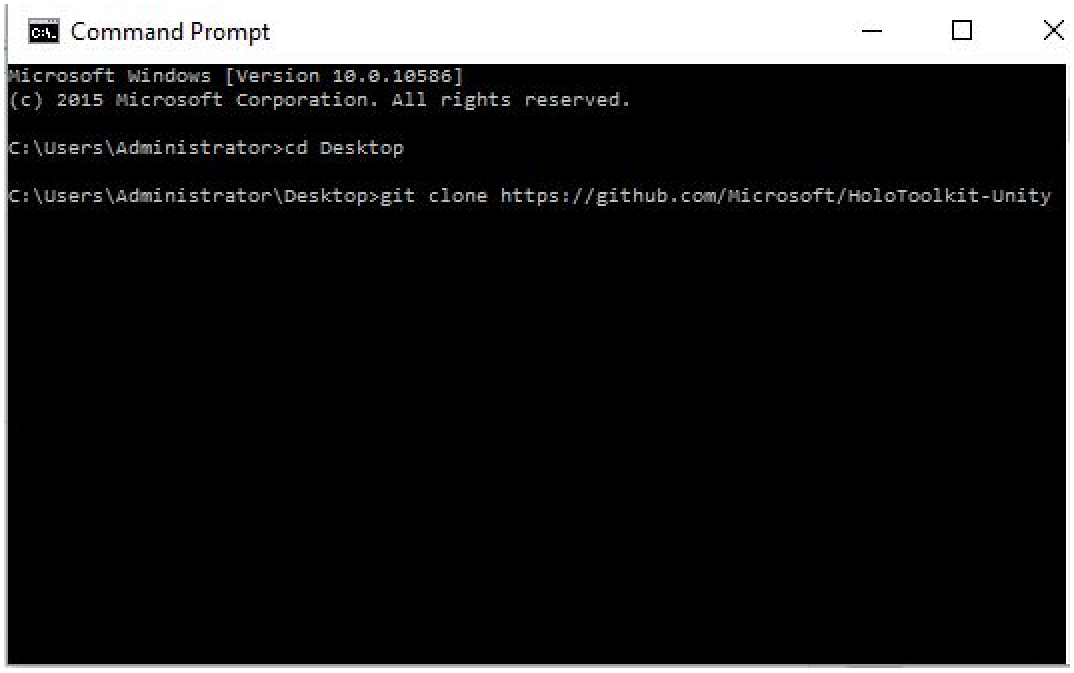
The Unity environment needs to be configured before beginning the development of the application. The configuration of this environment differs to the Hololens Academy.

The HoloToolkit differs from the HoloLens Academy in it's working.

INSTALLING THE HOLOTOOLKIT

The install requires Git to be installed on the system/PC.

Open the command window and navigate to the directory you wish to install the HoloToolkit, in the figure below, we have chosen the Desktop as the install directory. Then perform a git clone of the HoloToolkit github repository.



```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>cd Desktop

C:\Users\Administrator\Desktop>git clone https://github.com/Microsoft/HoloToolkit-Unity
```

Figure 1 : Install the HoloToolkit

SETTING UP THE HOLOTOOLKIT *(The Scene in Unity)*

Before importing the HoloToolkit into the Unity environment, the Main Camera and Directional Light prefabs need to be deleted.

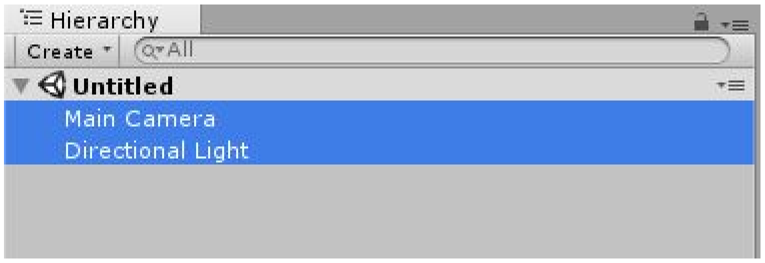


Figure 2 : Delete the prefabs

Importing the HoloToolkit

Assets -> Import Package -> Custom Package -> Select -> Desktop (Directory) -> HoloToolkit-Unit -> External -> Unity Packages -> Select the Unity Package with the right version number

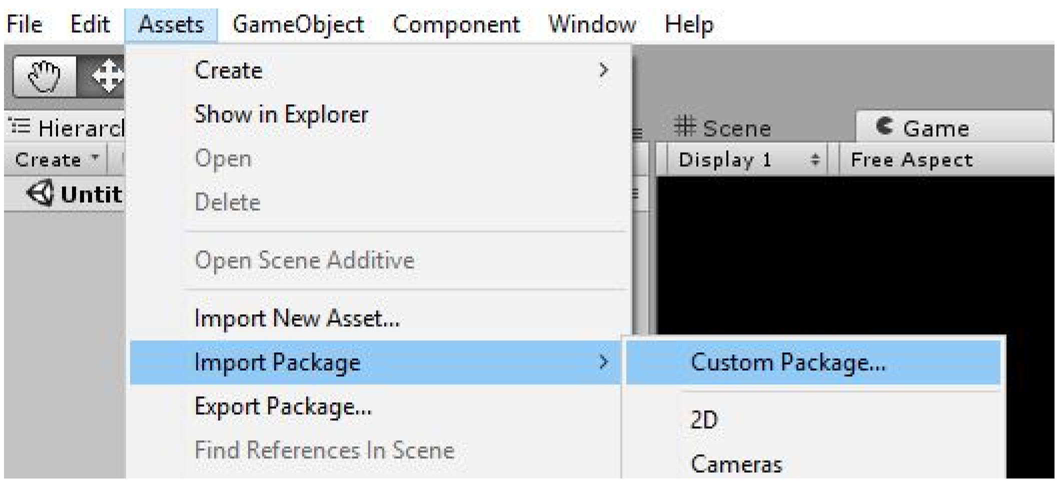
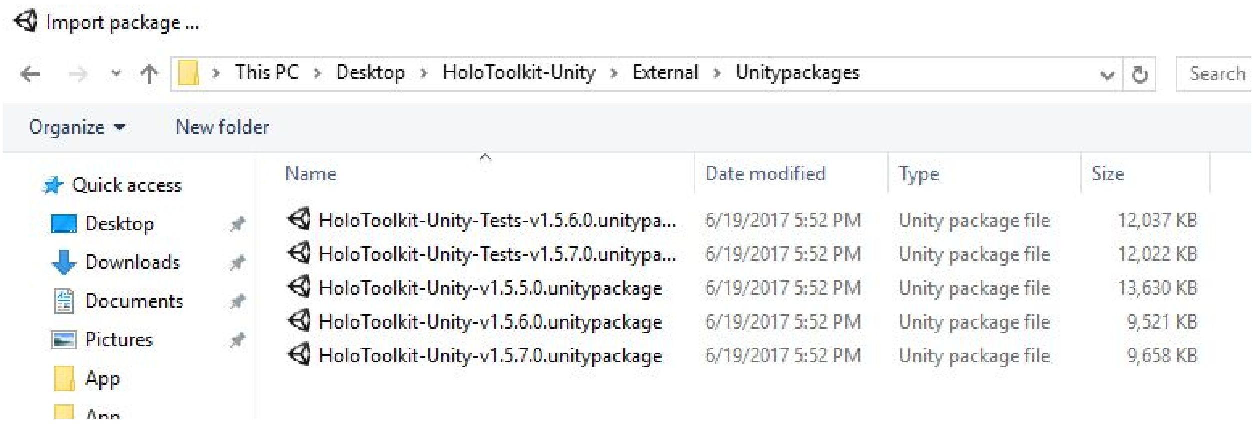


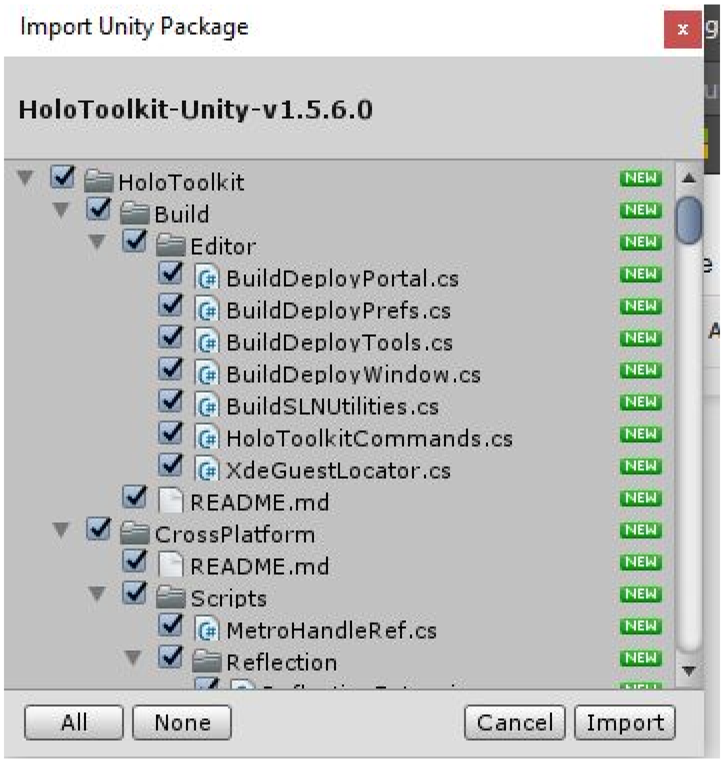
Figure 3 : Import the HoloToolkit



Name	Date modified	Type	Size
HoloToolkit-Unity-Tests-v1.5.6.0.unitypa...	6/19/2017 5:52 PM	Unity package file	12,037 KB
HoloToolkit-Unity-Tests-v1.5.7.0.unitypa...	6/19/2017 5:52 PM	Unity package file	12,022 KB
HoloToolkit-Unity-v1.5.5.0.unitypackage	6/19/2017 5:52 PM	Unity package file	13,630 KB
HoloToolkit-Unity-v1.5.6.0.unitypackage	6/19/2017 5:52 PM	Unity package file	9,521 KB
HoloToolkit-Unity-v1.5.7.0.unitypackage	6/19/2017 5:52 PM	Unity package file	9,658 KB

Figure 4 : Select the HoloToolkit Unity package

Next a Import Unity Package pop up window will appear, select Import, and the import will begin.


Import Unity Package

HoloToolkit-Unity-v1.5.6.0

- HoloToolkit NEW
  - Build NEW
    - Editor NEW
      - BuildDeployPortal.cs NEW
      - BuildDeployPrefs.cs NEW
      - BuildDeployTools.cs NEW
      - BuildDeployWindow.cs NEW
      - BuildSLNUtilities.cs NEW
      - HoloToolkitCommands.cs NEW
      - XdeGuestLocator.cs NEW
    - README.md NEW
  - CrossPlatform NEW
    - README.md NEW
    - Scripts NEW
      - MetroHandleRef.cs NEW
      - Reflection NEW

Figure 5 : Import the package

Once the import is complete a HoloToolkit option in the toolbar will appear.

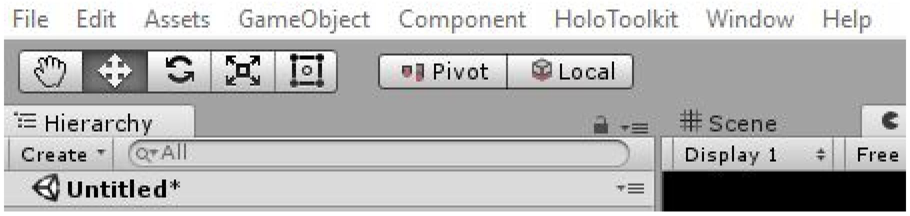


Figure 6 : HoloToolkit added to Unity

HoloToolkit folder in the Assets folder as shown in the project panel on importing.

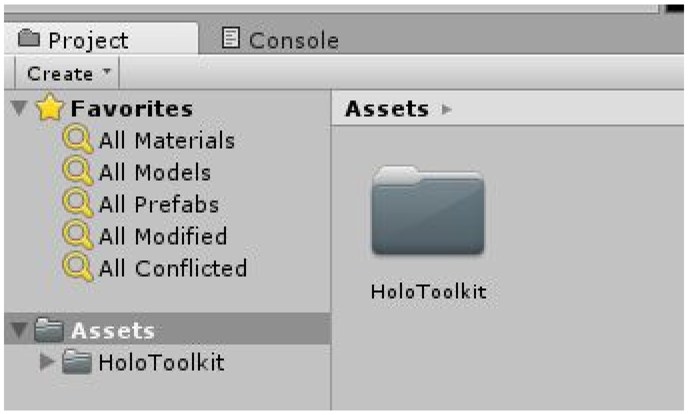


Figure 7 : HoloToolkit folder

The Assets folder in the project panel has the HoloToolkit folder, this folder contains scripts and prefabs that can be used in the application.

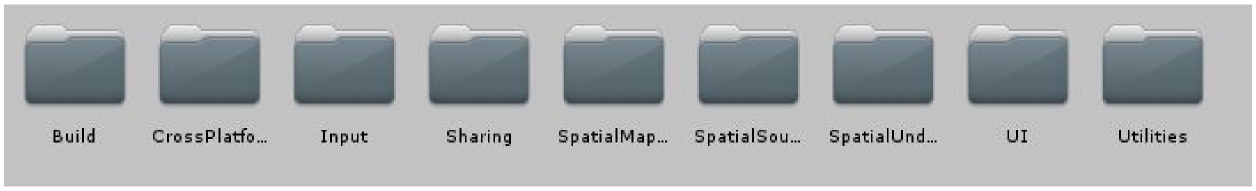


Figure 8 : Contents of the HoloToolkit folder

Setting the scene

The following steps will set the scene.

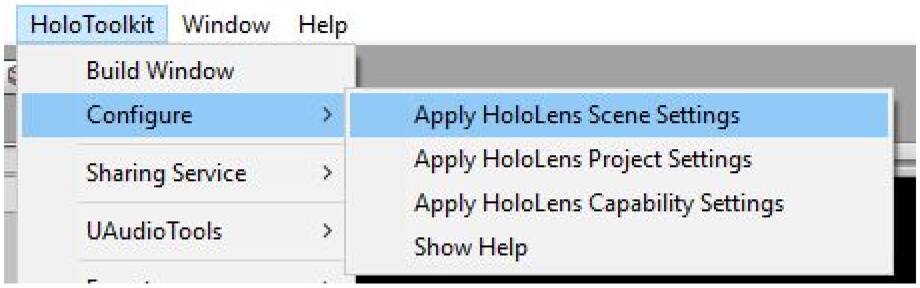
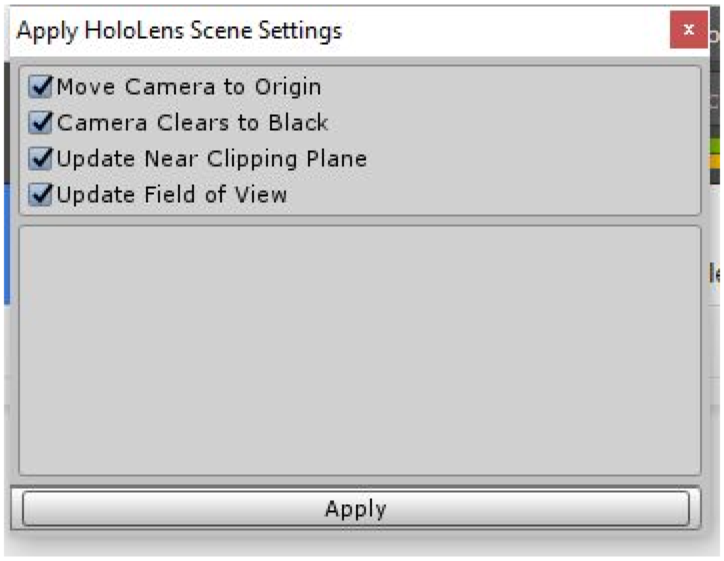

HoloToolkit
Window
Help
Build Window
Configure
Sharing Service
UAudioTools
Apply HoloLens Scene Settings
Apply HoloLens Project Settings
Apply HoloLens Capability Settings
Show Help

Figure 9 : Hololens Scene Settings



Apply HoloLens Scene Settings

- [x] Move Camera to Origin
- [x] Camera Clears to Black
- [x] Update Near Clipping Plane
- [x] Update Field of View

Figure 10 : Hololens Scene Settings applied

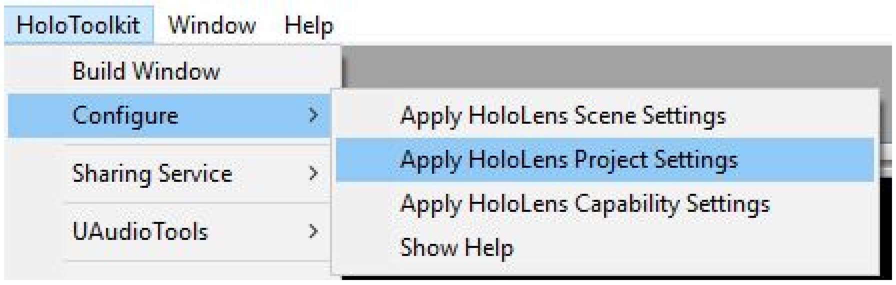
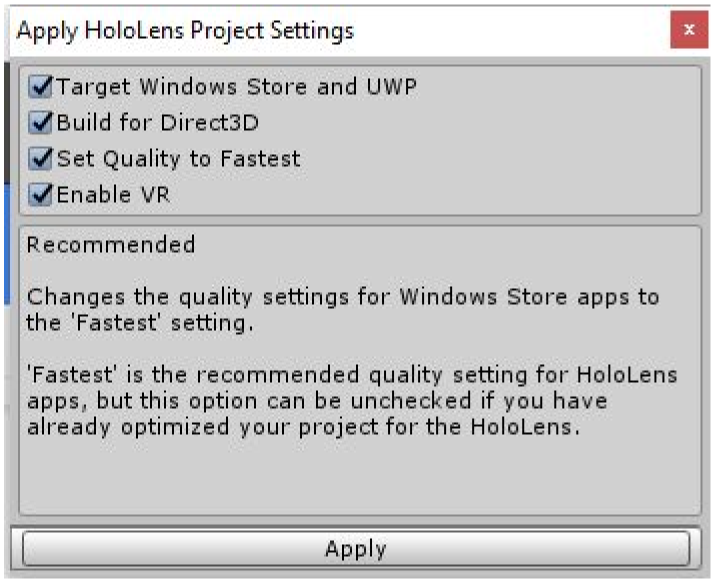

HoloToolkit
Window
Help
Build Window
Configure
Sharing Service
UAudioTools
Apply HoloLens Scene Settings
Apply HoloLens Project Settings
Apply HoloLens Capability Settings
Show Help

Figure 11 : Hololens Project Settings


Apply HoloLens Project Settings

- [x] Target Windows Store and UWP
- [x] Build for Direct3D
- [x] Set Quality to Fastest
- [x] Enable VR

Recommended

Changes the quality settings for Windows Store apps to the 'Fastest' setting.

'Fastest' is the recommended quality setting for HoloLens apps, but this option can be unchecked if you have already optimized your project for the HoloLens.

Figure 12 : Hololens Project Settings applied

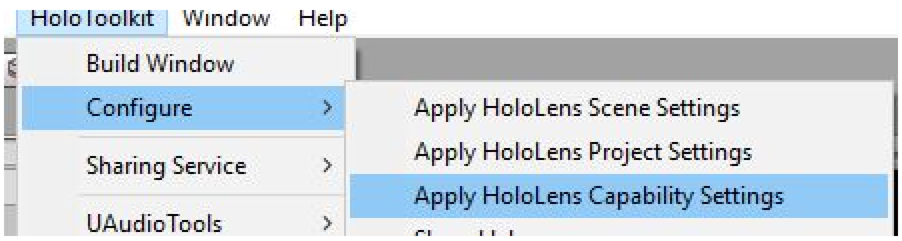
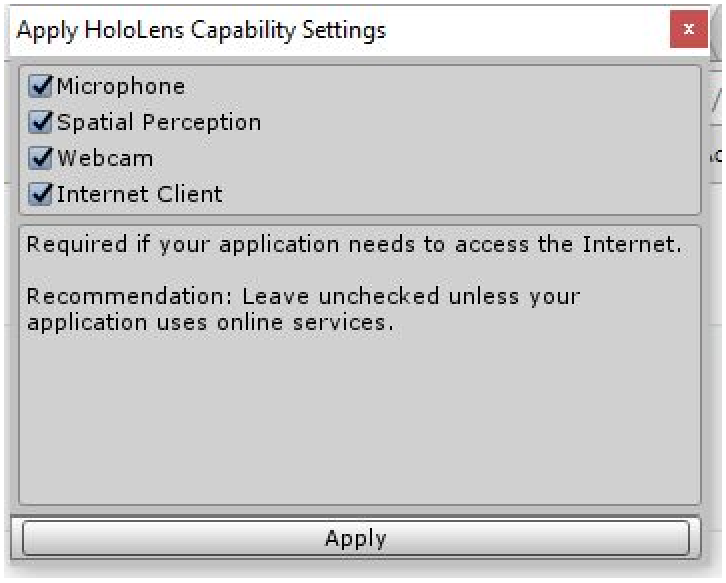

HoloToolkit
Window
Help
Build Window
Configure
Sharing Service
UAudioTools
Apply HoloLens Scene Settings
Apply HoloLens Project Settings
Apply HoloLens Capability Settings

Figure 13 : Hololens Capability Settings.


Apply HoloLens Capability Settings

- [x] Microphone
- [x] Spatial Perception
- [x] Webcam
- [x] Internet Client

Required if your application needs to access the Internet.

Recommendation: Leave unchecked unless your application uses online services.

Figure 14 : Hololens Capability Settings applied.

Now the scene is set and the developer can start building the application. The developer still needs to manually set the Spatializer plugin for Spatial Sound.

- Edit -> Audio -> Spatializer Plugin -> *Enable* MS HRTF Spatializer (Spatial Sound)

Camera

There is a built-in prefab called the HoloLensCamera, drag and drop it onto the scene, to set the camera.

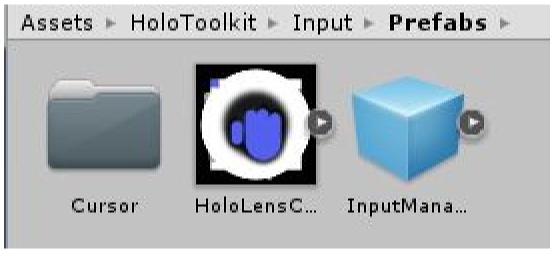
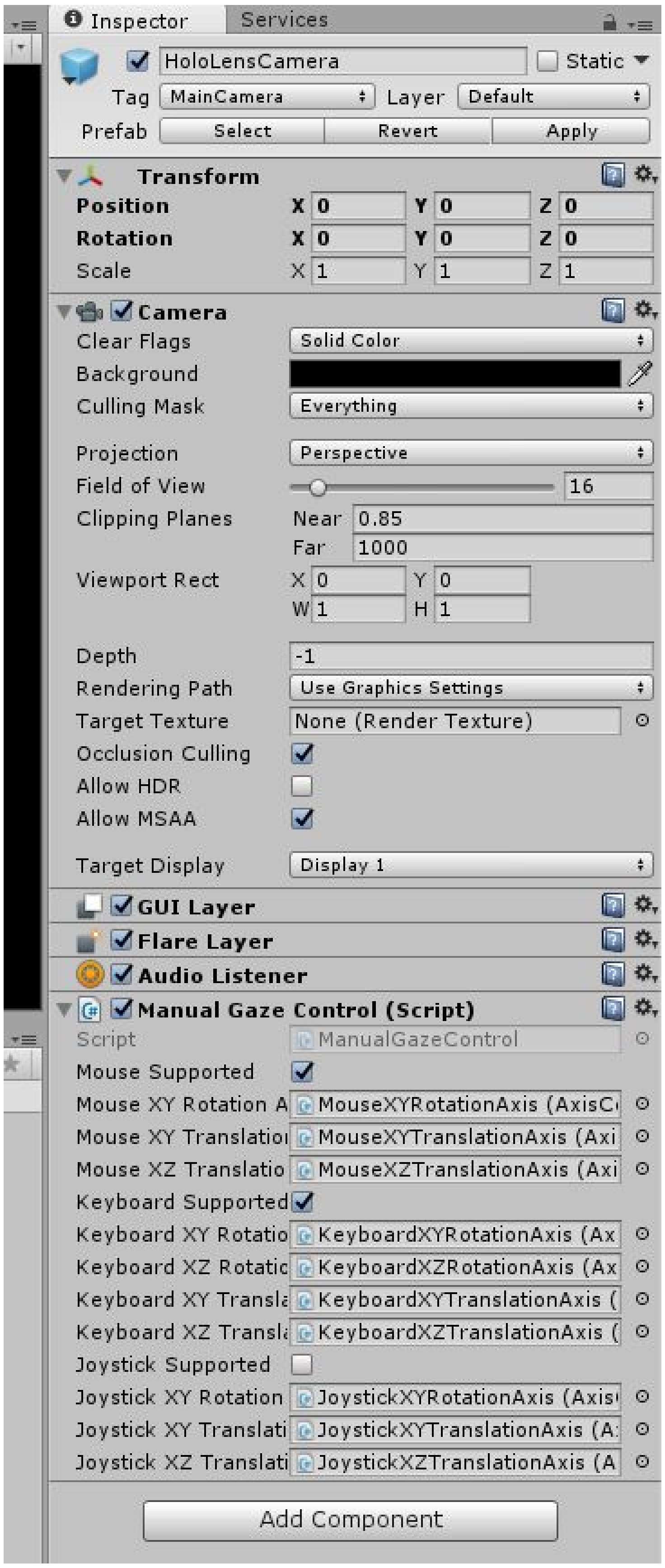


Figure 15 : HoloLensCamera prefab



Inspector | Services

☑ HoloLensCamera ☐ Static

Tag: MainCamera | Layer: Default

Prefab: Select | Revert | Apply

Transform

Position	X 0	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

☑ **Camera**

Clear Flags	Solid Color
Background	■
Culling Mask	Everything
Projection	Perspective
Field of View	16
Clipping Planes	Near 0.85
	Far 1000
Viewport Rect	X 0 Y 0
	W 1 H 1
Depth	-1
Rendering Path	Use Graphics Settings
Target Texture	None (Render Texture)
Occlusion Culling	✓
Allow HDR	☐
Allow MSAA	✓
Target Display	Display 1

☑ **GUI Layer**

☑ **Flare Layer**

☑ **Audio Listener**

☑ **Manual Gaze Control (Script)**

Script	ManualGazeControl
Mouse Supported	✓
Mouse XY Rotation A	MouseXYRotationAxis (AxisC
Mouse XY Translatio	MouseXYTranslationAxis (Axi
Mouse XZ Translatio	MouseXZTranslationAxis (Axi
Keyboard Supported	✓
Keyboard XY Rotatio	KeyboardXYRotationAxis (Ax
Keyboard XZ Rotatic	KeyboardXZRotationAxis (Ax
Keyboard XY Transla	KeyboardXYTranslationAxis (
Keyboard XZ Transl	KeyboardXZTranslationAxis (
Joystick Supported	☐
Joystick XY Rotation	JoystickXYRotationAxis (Axis
Joystick XY Translati	JoystickXYTranslationAxis (A
Joystick XZ Translati	JoystickXZTranslationAxis (A

Add Component

Figure 16 : HoloLensCamera prefab settings

The prefab already has the required values set and the developer doesn't need to change any values to set the camera gameobject.

Cursor

We can drag and drop any of the built-in cursor prefab objects depending on the application.

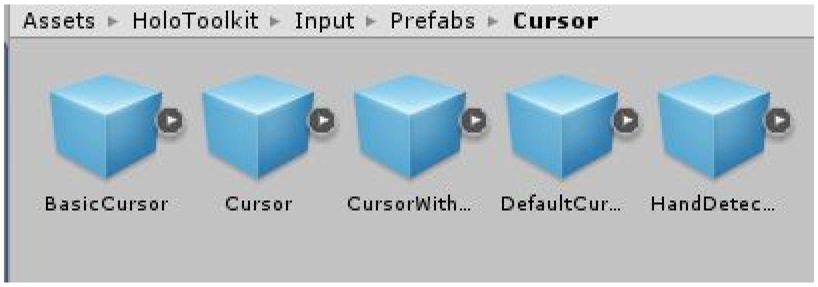


Figure 17 : Cursor prefabs

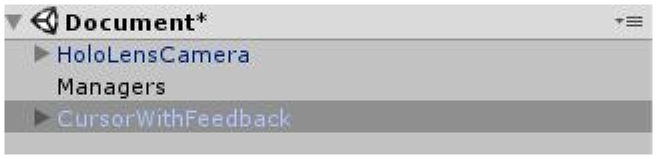


Figure 18 : CursorWithFeedback prefab in the scene

Managers GameObject

Create a gameobject, rename it to Managers, any name can be chosen, but generally the name used is Managers. The Managers gameobject will contain prefabs and scripts which will be used to manage input and other aspects in the scene. To handle the input we drag and drop the input manager prefab into the Managers gameobject.

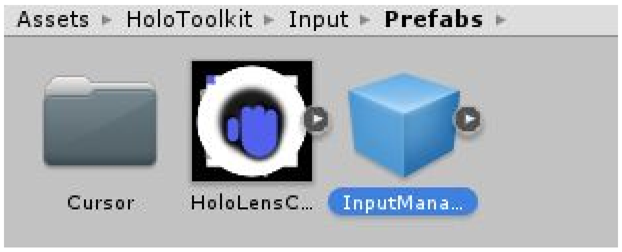
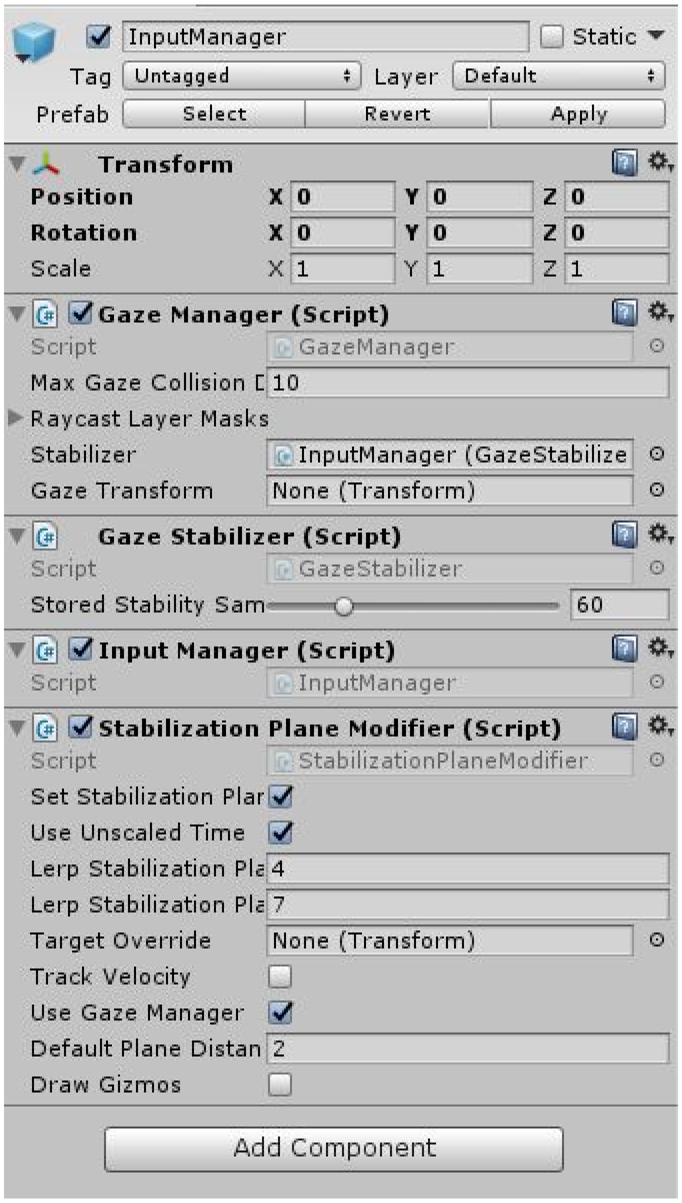


Figure 19 : InputManager prefab


InputManager ☑ Static
Tag: Untagged | Layer: Default
Prefab: Select | Revert | Apply

Transform

Position	X 0	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

Gaze Manager (Script) ☑

Script	GazeManager
Max Gaze Collision D	10
Raycast Layer Masks	
Stabilizer	InputManager (GazeStabilize
Gaze Transform	None (Transform)

Gaze Stabilizer (Script)

Script	GazeStabilizer
Stored Stability Sam	60

Input Manager (Script) ☑

Script	InputManager

Stabilization Plane Modifier (Script) ☑

Script	StabilizationPlaneModifier
Set Stabilization Plar	✓
Use Unscaled Time	✓
Lerp Stabilization Pla	4
Lerp Stabilization Pla	7
Target Override	None (Transform)
Track Velocity	☐
Use Gaze Manager	✓
Default Plane Distan	2
Draw Gizmos	☐

Figure 20 : InputManager prefab scripts and values

The input manager contains all the necessary scripts and values associated with the input capabilities of the HoloLens.

The input module is designed to be extensible: it could support various input mechanisms and various types of gazers.

Each input source (hands, gestures, others) implement a IInputSource interface. The interface defines various events that the input sources can trigger. The input sources register themselves with the InputManager, whose role it is to forward input to the appropriate game objects. Input sources can be dynamically enabled / disabled as necessary, and new input sources can be created to support different input devices.

Game objects that want to consume input events can implement one or many input interfaces, such as:

- IFocusable for focus enter and exit. The focus can be triggered by the user's gaze or any other gaze source.
- IHoldHandle for the Windows hold gesture.
- IInputHandler for source up and down. The source can be a hand that tapped, a clicker that was pressed, etc.
- IInputClickHandler for source clicked. The source can be a hand that tapped, a clicker that was pressed, etc.
- IManipulationHandler for the Windows manipulation gesture.
- INavigationnHandler for the Windows navigation gesture.
- ISourceStateHandler for the source detected and source lost events.
- ISpeechHandler for voice commands.
- IDictationHandler for speech to text dictation.

The input manager listens to the various events coming from the input sources, and also takes into account the gaze. Currently, that gaze is always coming from the GazeManager class, but this could be extended to support multiple gaze sources if the need arises.

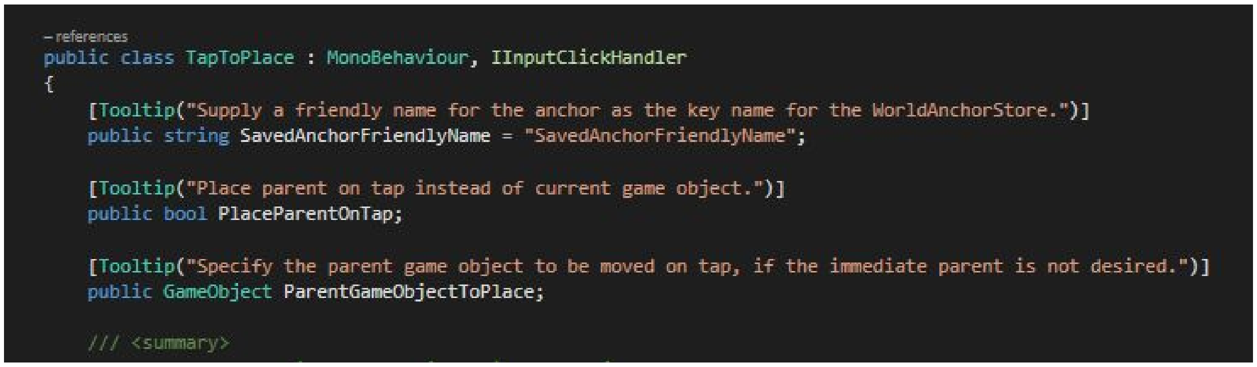
By default, input events are sent to the currently focused gameobject, if that object implements the appropriate interface. Modals input handlers can also be added to the input manager: these modal handlers will take priority over the currently focused object Fallback handlers can also be defined, so that the application can react to global inputs that aren't targeting a specific element. Any event sent by the input manager always bubbles up from the object to its ancestors.

In recap, the input manager forwards the various input sources events to the appropriate game object, using the following order:

1. The registered modal input handlers, in LIFO (Last-In First-Out) order of registration

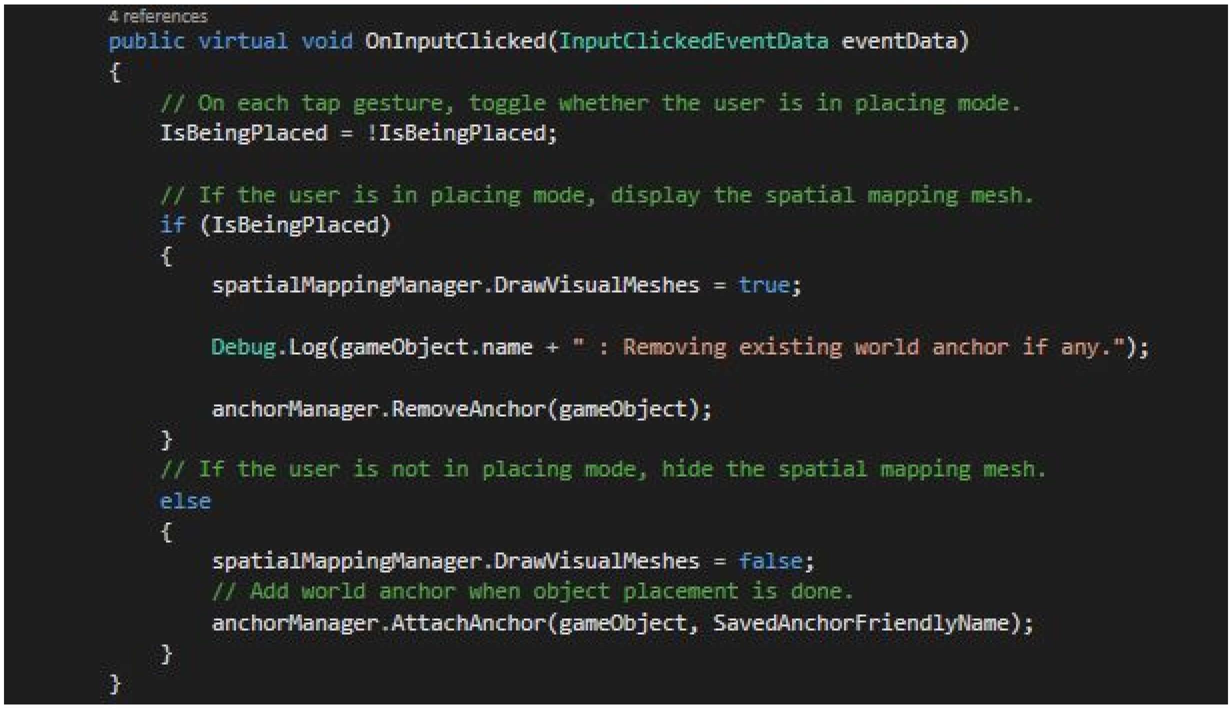
2. The currently focused object
3. The fallback input handlers, in LIFO order of registration

To uses these interface we implement something like this.



```
– references
public class TapToPlace : MonoBehaviour, IInputClickHandler
{
    [Tooltip("Supply a friendly name for the anchor as the key name for the WorldAnchorStore.")]
    public string SavedAnchorFriendlyName = "SavedAnchorFriendlyName";

    [Tooltip("Place parent on tap instead of current game object.")]
    public bool PlaceParentOnTap;

    [Tooltip("Specify the parent game object to be moved on tap, if the immediate parent is not desired.")]
    public GameObject ParentGameObjectToPlace;

    /// <summary>
```

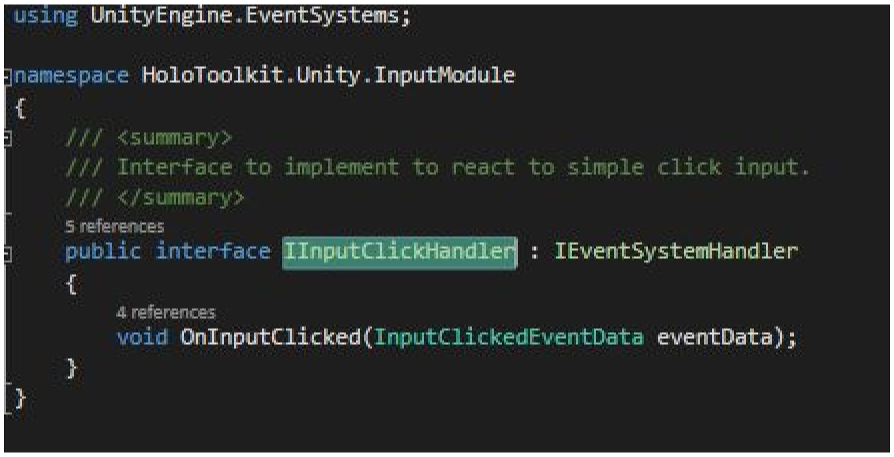
Figure 21 : Implement Interfaces (IInputClickHandler : Air-Tap)



```
4 references
public virtual void OnInputClicked(InputClickedEventData eventData)
{
    // On each tap gesture, toggle whether the user is in placing mode.
    IsBeingPlaced = !IsBeingPlaced;

    // If the user is in placing mode, display the spatial mapping mesh.
    if (IsBeingPlaced)
    {
        spatialMappingManager.DrawVisualMeshes = true;

        Debug.Log(gameObject.name + " : Removing existing world anchor if any.");

        anchorManager.RemoveAnchor(gameObject);
    }
    // If the user is not in placing mode, hide the spatial mapping mesh.
    else
    {
        spatialMappingManager.DrawVisualMeshes = false;
        // Add world anchor when object placement is done.
        anchorManager.AttachAnchor(gameObject, SavedAnchorFriendlyName);
    }
}
```

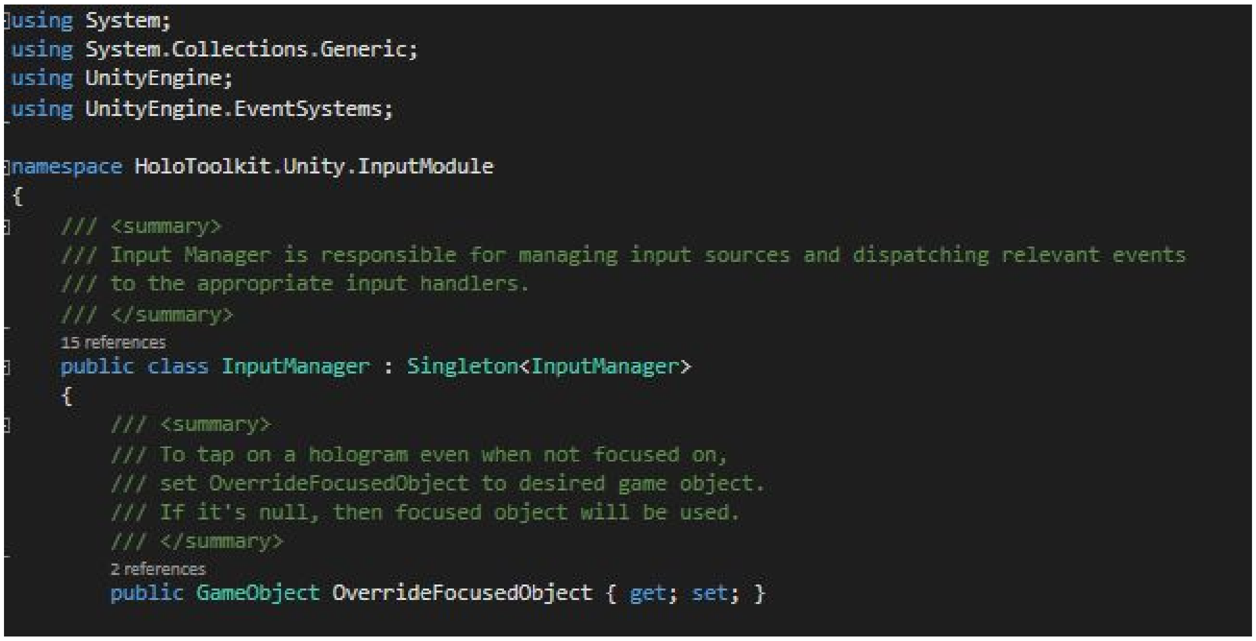
Figure 22 : OnInputClicked() is called when the Air-Tap gesture is performed

The working of these functions is based on the interfaces and the InputManager script, as shown below.



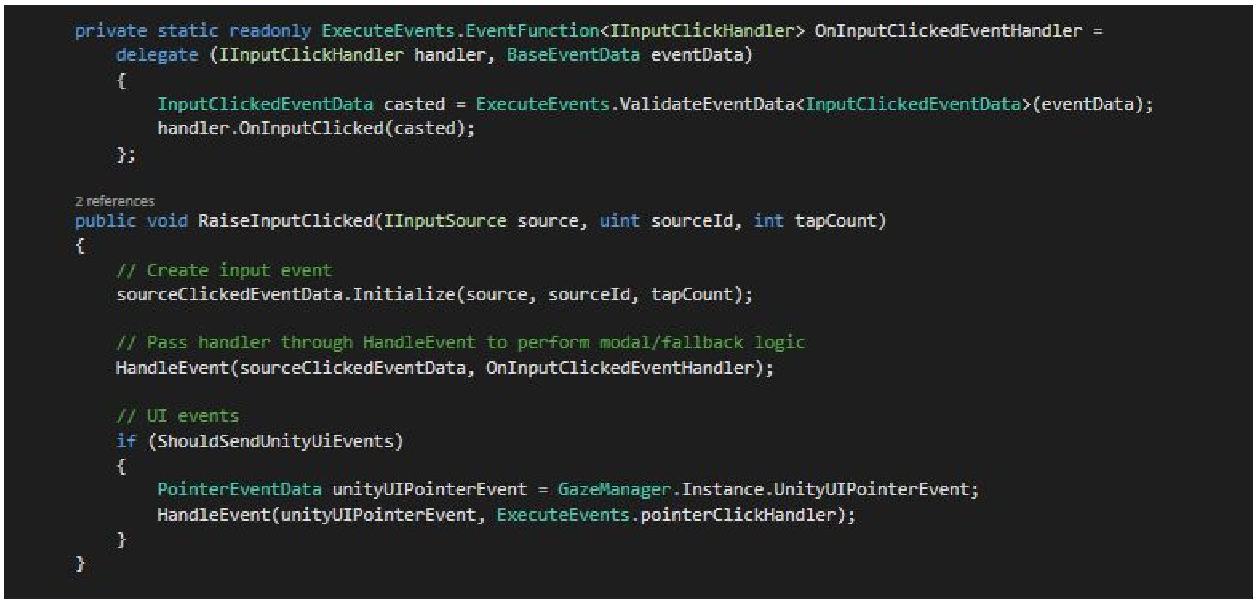
```
using UnityEngine.EventSystems;

namespace HoloToolkit.Unity.InputModule
{
    /// <summary>
    /// Interface to implement to react to simple click input.
    /// </summary>
    5 references
    public interface IInputClickHandler : IEventSystemHandler
    {
        4 references
        void OnInputClicked(InputClickedEventData eventData);
    }
}
```

Figure 23 : IInputClickHandler interface



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

namespace HoloToolkit.Unity.InputModule
{
    /// <summary>
    /// Input Manager is responsible for managing input sources and dispatching relevant events
    /// to the appropriate input handlers.
    /// </summary>
    15 references
    public class InputManager : Singleton<InputManager>
    {
        /// <summary>
        /// To tap on a hologram even when not focused on,
        /// set OverrideFocusedObject to desired game object.
        /// If it's null, then focused object will be used.
        /// </summary>
        2 references
        public GameObject OverrideFocusedObject { get; set; }
```

Figure 24 : Input Manager script



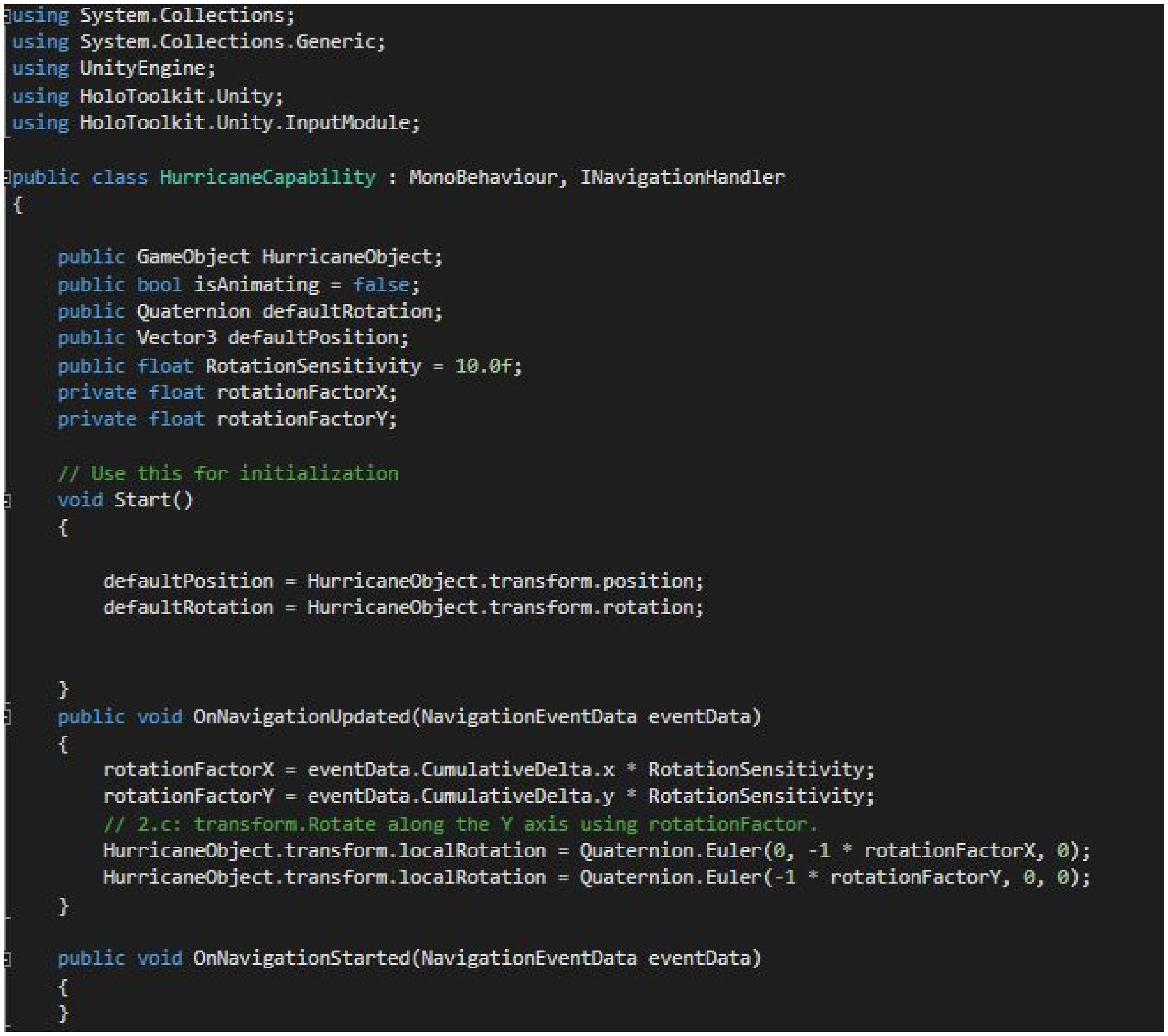
```
private static readonly ExecuteEvents.EventFunction<IInputClickHandler> OnInputClickedEventHandler =
    delegate (IInputClickHandler handler, BaseEventData eventData)
    {
        InputClickedEventData casted = ExecuteEvents.ValidateEventData<InputClickedEventData>(eventData);
        handler.OnInputClicked(casted);
    };

2 references
public void RaiseInputClicked(IInputSource source, uint sourceId, int tapCount)
{
    // Create input event
    sourceClickedEventData.Initialize(source, sourceId, tapCount);

    // Pass handler through HandleEvent to perform modal/fallback logic
    HandleEvent(sourceClickedEventData, OnInputClickedEventHandler);

    // UI events
    if (ShouldSendUnityUiEvents)
    {
        PointerEventData unityUIPointerEvent = GazeManager.Instance.UnityUIPointerEvent;
        HandleEvent(unityUIPointerEvent, ExecuteEvents.pointerClickHandler);
    }
}
```

Figure 25 : Input Manager script

These scripts show how the input events are handled.

We follow similar steps for manipulation or navigation gestures.

Remember to specify “using HoloToolkit.Unity.InputModule” before using any of the input module interfaces.



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using HoloToolkit.Unity;
using HoloToolkit.Unity.InputModule;

public class HurricaneCapability : MonoBehaviour, INavigationHandler
{

    public GameObject HurricaneObject;
    public bool isAnimating = false;
    public Quaternion defaultRotation;
    public Vector3 defaultPosition;
    public float RotationSensitivity = 10.0f;
    private float rotationFactorX;
    private float rotationFactorY;

    // Use this for initialization
    void Start()
    {

        defaultPosition = HurricaneObject.transform.position;
        defaultRotation = HurricaneObject.transform.rotation;


    }
    public void OnNavigationUpdated(NavigationEventData eventData)
    {
        rotationFactorX = eventData.CumulativeDelta.x * RotationSensitivity;
        rotationFactorY = eventData.CumulativeDelta.y * RotationSensitivity;
        // 2.c: transform.Rotate along the Y axis using rotationFactor.
        HurricaneObject.transform.localRotation = Quaternion.Euler(0, -1 * rotationFactorX, 0);
        HurricaneObject.transform.localRotation = Quaternion.Euler(-1 * rotationFactorY, 0, 0);
    }

    public void OnNavigationStarted(NavigationEventData eventData)
    {
    }
```

Figure 26 : INavigationHandler interface

The OnNavigation functions are called when the hand is in the pressed or released state, as discussed in the HoloLens Academy.

For the input module to work, we need to enable an event system for any input events and other events that occur in the application.

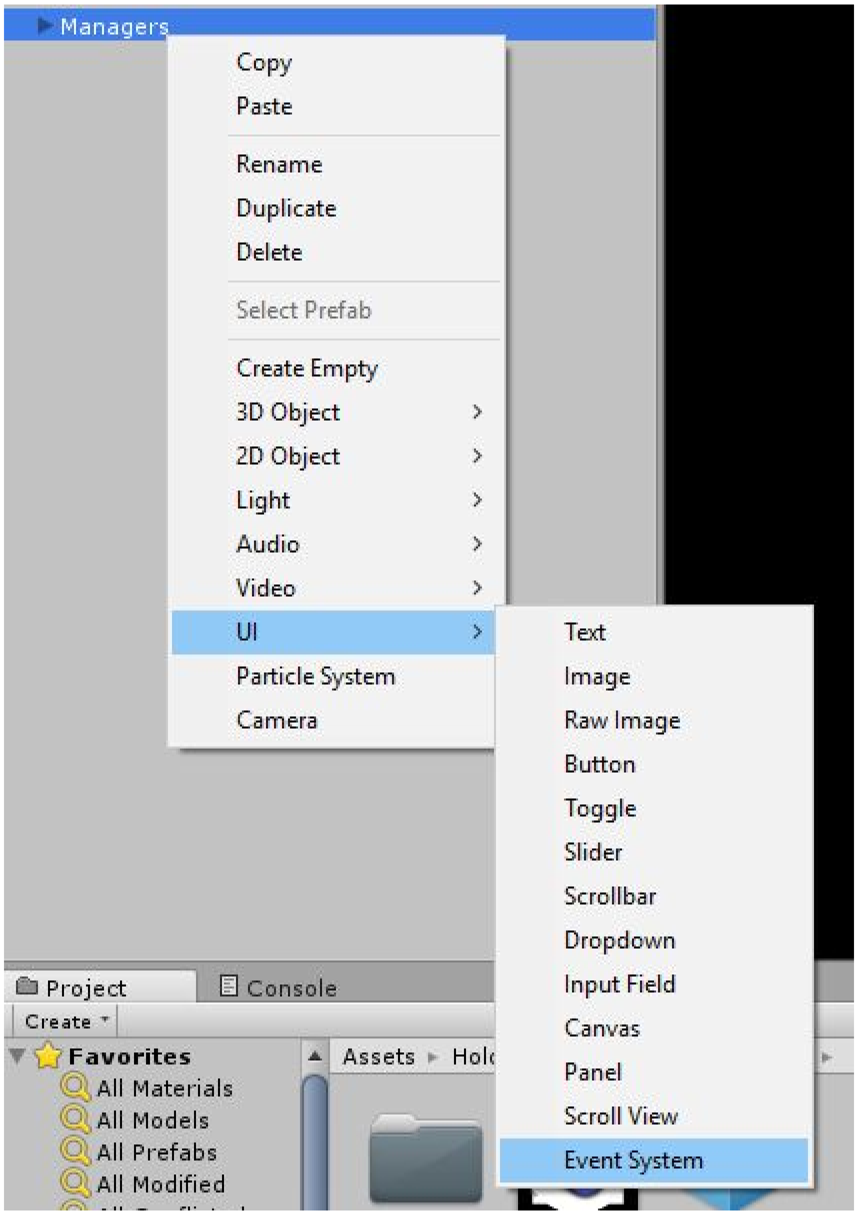


Figure 27 : Event System

To add Spatial mapping to the application we drag and drop the spatial mapping prefab into the Managers gameobject.

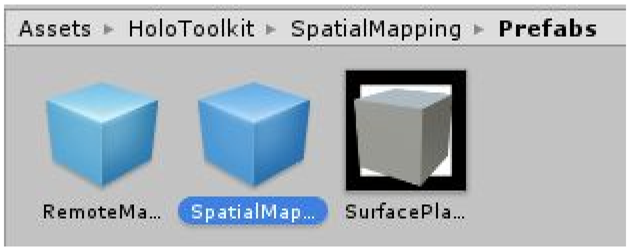
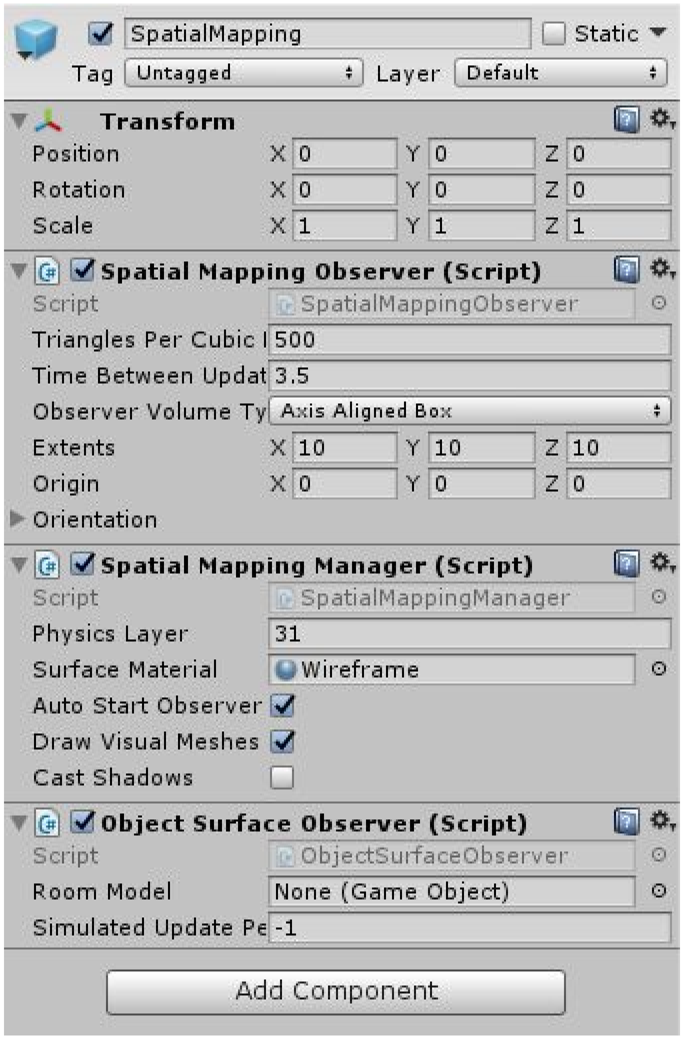


Figure 28 : Spatial Mapping prefab



SpatialMapping ✓ — Static ☐
Tag: Untagged — Layer: Default

Transform

	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Scale	1	1	1

Spatial Mapping Observer (Script) ✓

Script	SpatialMappingObserver
Triangles Per Cubic I	500
Time Between Updat	3.5
Observer Volume Ty	Axis Aligned Box
Extents	X 10 Y 10 Z 10
Origin	X 0 Y 0 Z 0
▸ Orientation	

Spatial Mapping Manager (Script) ✓

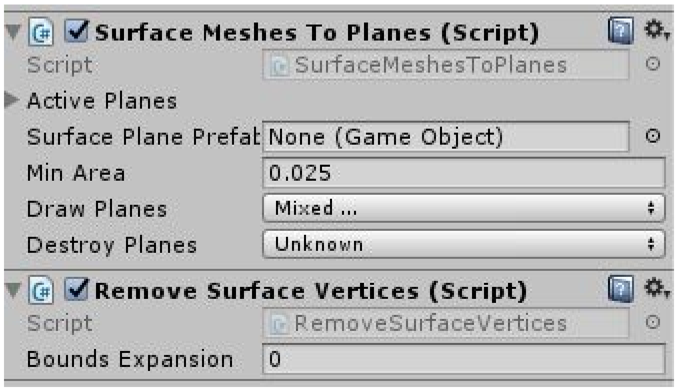
Script	SpatialMappingManager
Physics Layer	31
Surface Material	Wireframe
Auto Start Observer	✓
Draw Visual Meshes	✓
Cast Shadows	☐

Object Surface Observer (Script) ✓

Script	ObjectSurfaceObserver
Room Model	None (Game Object)
Simulated Update Pe	-1

Figure 29 : SpatialMapping prefab, the associated scripts and values

The spatial mapping has the scripts related to spatial mapping that were discussed in the HoloLens Academy, example : you can drag your room model and drop it in the Object Surface Observer script.



Surface Meshes To Planes (Script)

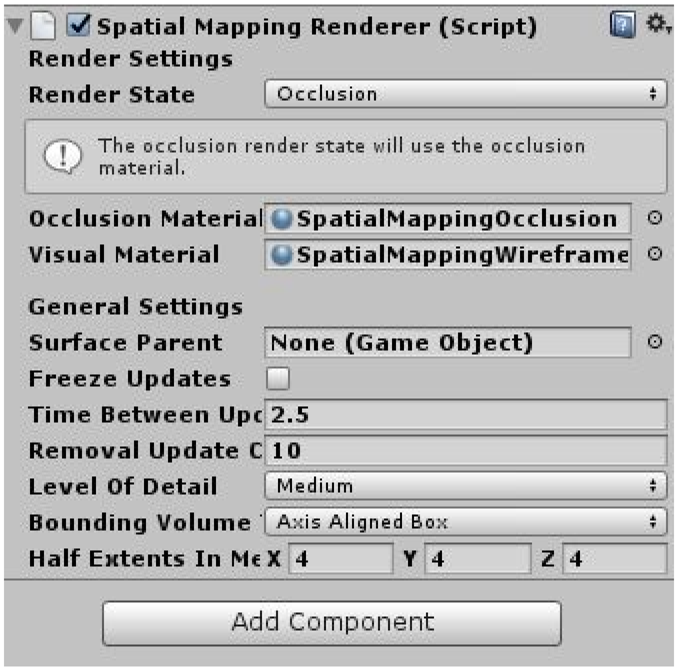
Script	SurfaceMeshesToPlanes
Active Planes	
Surface Plane Prefab	None (Game Object)
Min Area	0.025
Draw Planes	Mixed ...
Destroy Planes	Unknown

Remove Surface Vertices (Script)

Script	RemoveSurfaceVertices
Bounds Expansion	0

Figure 30 : Plane finding and vertex removal

These scripts are built-in and you can add them, to the spatial mapping object by dragging and dropping. Just like in the HoloLens Academy you can add the spatial mapping collider for physics collisions between the spatial mapping mesh and holograms. For enabling occlusion we can add the spatial mapping renderer component and set the occlusion material.



Spatial Mapping Renderer (Script)

Render Settings

Render State	Occlusion

The occlusion render state will use the occlusion material.

Occlusion Material	SpatialMappingOcclusion
Visual Material	SpatialMappingWireframe

General Settings

Surface Parent	None (Game Object)
Freeze Updates	☐
Time Between Upc	2.5
Removal Update C	10
Level Of Detail	Medium
Bounding Volume	Axis Aligned Box
Half Extents In Me	X 4 Y 4 Z 4

Figure 31 : Spatial Mapping Renderer component

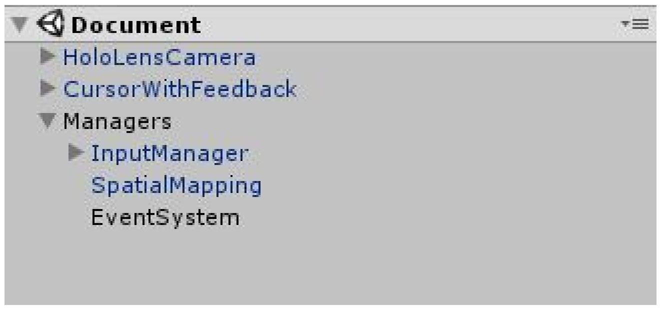


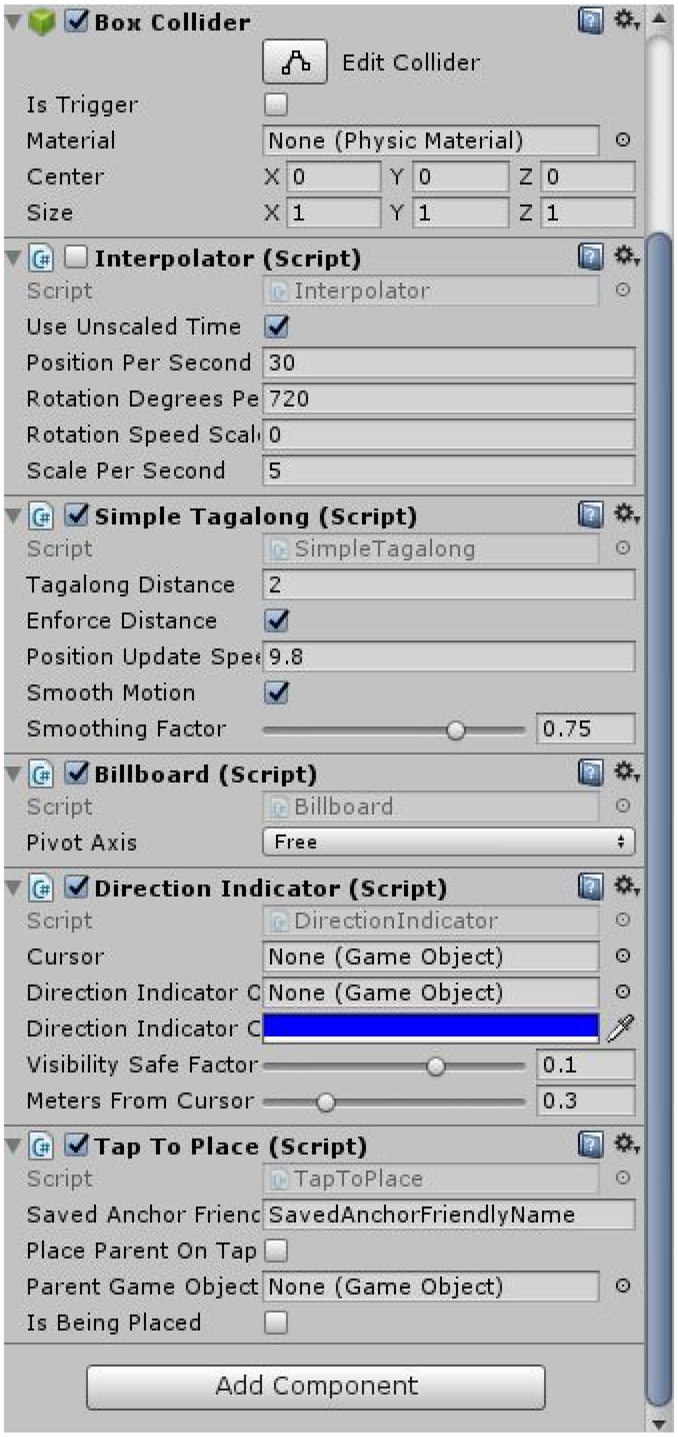
Figure 32 : Managers gameobject with all the prefabs

USING THE HOLOTOOLKIT

Many of the concepts discussed before have built-in scripts and prefabs that are present in the HoloToolkit. We need to drag and drop it onto the right gameobjects.

Example :

We can use concepts such a Billborading, TagAlong and having a directional indicator assigned to a gameobject by using the scripts related to these which are built into the HoloToolkit. This means we can directly add or drag and drop these scripts onto the desired gameobjects.



Box Collider

Edit Collider

Is Trigger	☐
Material	None (Physic Material)
Center	X 0 Y 0 Z 0
Size	X 1 Y 1 Z 1

Interpolator (Script)

Script	Interpolator
Use Unscaled Time	✓
Position Per Second	30
Rotation Degrees Pe	720
Rotation Speed Scal	0
Scale Per Second	5

Simple Tagalong (Script)

Script	SimpleTagalong
Tagalong Distance	2
Enforce Distance	✓
Position Update Spee	9.8
Smooth Motion	✓
Smoothing Factor	0.75

Billboard (Script)

Script	Billboard
Pivot Axis	Free

Direction Indicator (Script)

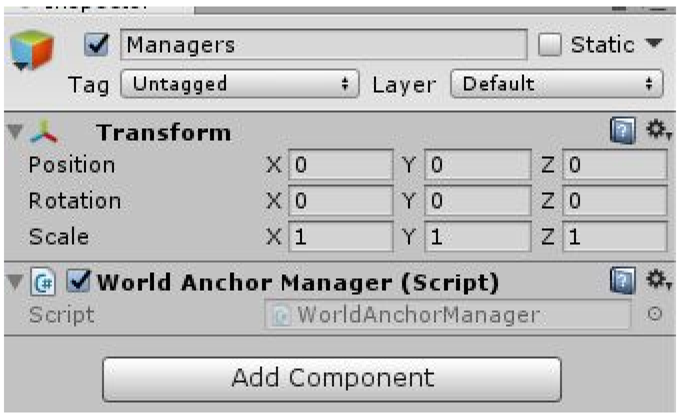
Script	DirectionIndicator
Cursor	None (Game Object)
Direction Indicator C	None (Game Object)
Direction Indicator C	🟦
Visibility Safe Factor	0.1
Meters From Cursor	0.3

Tap To Place (Script)

Script	TapToPlace
Saved Anchor Frienc	SavedAnchorFriendlyName
Place Parent On Tap	☐
Parent Game Object	None (Game Object)
Is Being Placed	☐

Figure 33 : HoloToolkit scripts

The TapToPlace script places an object using the air tap gesture and the user's gaze. The user's gaze is used to find a location to place the gameobject and the air tap gesture is used to pick up and place the object. To use this script we need to add the World Anchor script to the Managers gameobject. The World Anchor script creates spatial anchors, therefore the placed gameobject will retain its position even after the HoloLens is switched off and switched back on.



Managers
Static
Tag: Untagged
Layer: Default

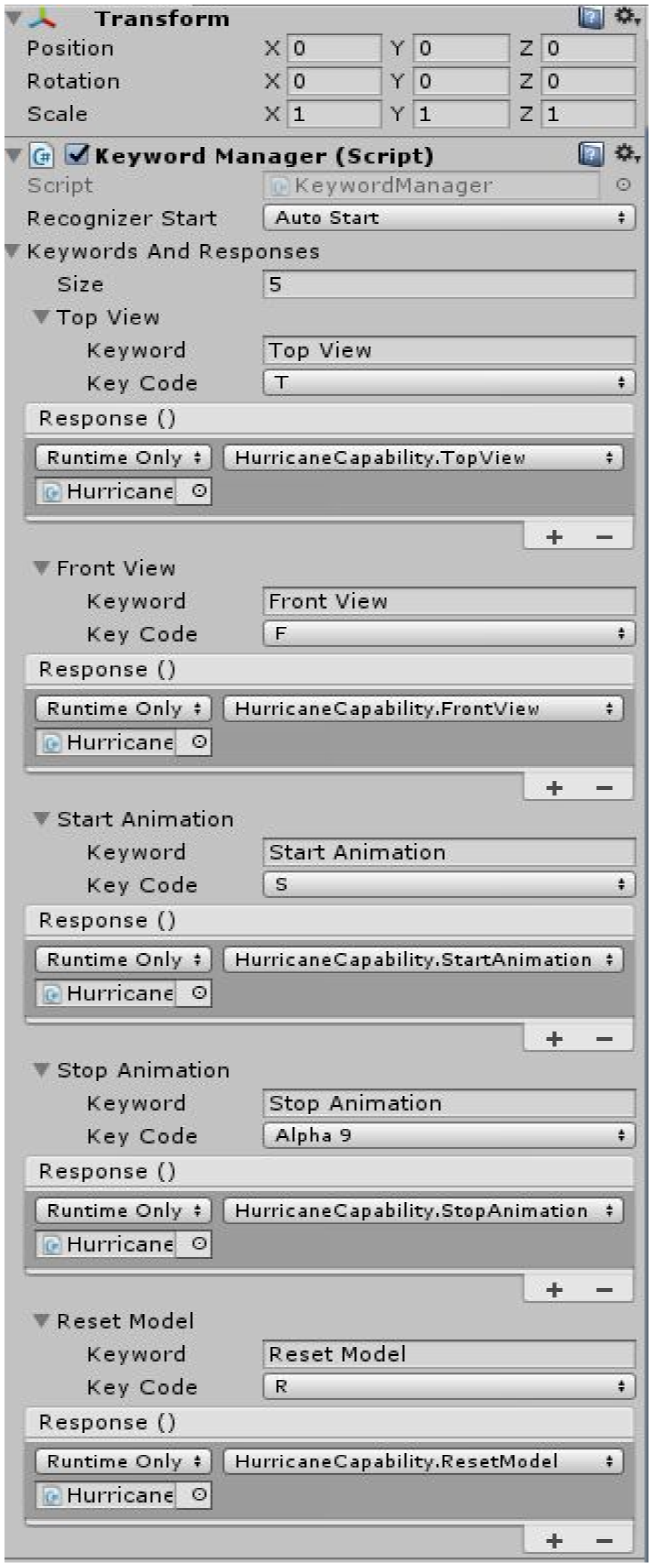
Transform

	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Scale	1	1	1

World Anchor Manager (Script)
Script: WorldAnchorManager

Figure 34 : World Anchor script

We can also add a keyword recognizer, we can do this by adding the Keyword recognizer script to the Managers gameobject, it will recognize the specified keywords and take the specified action.



Transform

Position	X	0	Y	0	Z	0
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

Keyword Manager (Script)

Script: KeywordManager
Recognizer Start: Auto Start
Keywords And Responses
Size: 5

Top View
Keyword: Top View
Key Code: T
Response ()
Runtime Only | HurricaneCapability.TopView
Hurricane

Front View
Keyword: Front View
Key Code: F
Response ()
Runtime Only | HurricaneCapability.FrontView
Hurricane

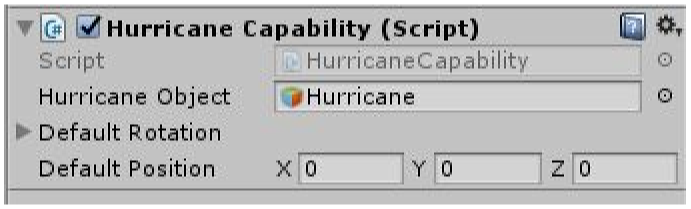
Start Animation
Keyword: Start Animation
Key Code: S
Response ()
Runtime Only | HurricaneCapability.StartAnimation
Hurricane

Stop Animation
Keyword: Stop Animation
Key Code: Alpha 9
Response ()
Runtime Only | HurricaneCapability.StopAnimation
Hurricane

Reset Model
Keyword: Reset Model
Key Code: R
Response ()
Runtime Only | HurricaneCapability.ResetModel
Hurricane

Figure 35 : Keyword Recognizer

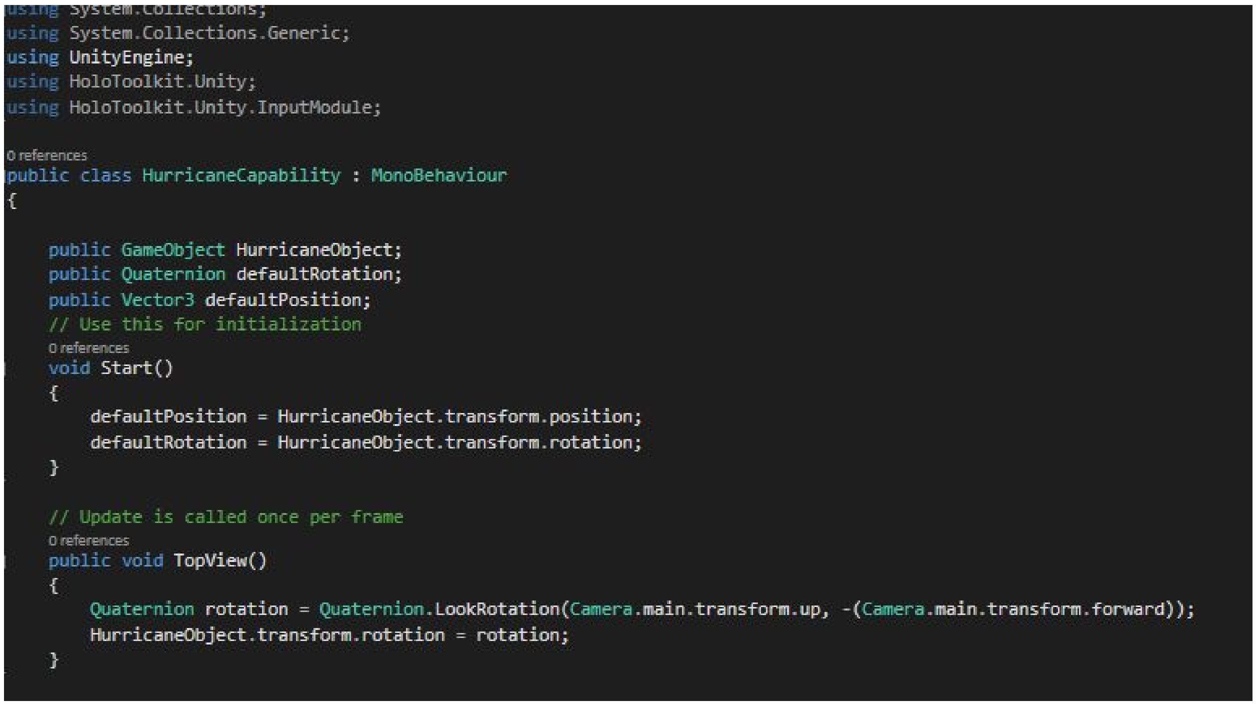
Top View is the keyword that invokes the TopView() function specified in the Hurricane Capability script. The hurricane capability script is attached to a HurricaneManager game object and this gameobject is drag and dropped into the space under the Runtime only drop down in the Managers gameobject, which contains the keyword recognizer.



Hurricane Capability (Script)

Script	HurricaneCapability
Hurricane Object	Hurricane
Default Rotation	
Default Position	X 0 Y 0 Z 0

Figure 36 : Hurricane Capability script, attached to HurricaneManager gameobject



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using HoloToolkit.Unity;
using HoloToolkit.Unity.InputModule;

0 references
public class HurricaneCapability : MonoBehaviour
{

    public GameObject HurricaneObject;
    public Quaternion defaultRotation;
    public Vector3 defaultPosition;
    // Use this for initialization
    0 references
    void Start()
    {
        defaultPosition = HurricaneObject.transform.position;
        defaultRotation = HurricaneObject.transform.rotation;
    }

    // Update is called once per frame
    0 references
    public void TopView()
    {
        Quaternion rotation = Quaternion.LookRotation(Camera.main.transform.up, -(Camera.main.transform.forward));
        HurricaneObject.transform.rotation = rotation;
    }
```

Figure 37 : Invoking actions based on the keyword

The function that is called when a keyword is recognized needs to be a public function.

HURRICANE SIMULATION

This application involved constructing a hurricane simulation, and adding interactivity so that the user can interact with the simulation.

There are 184 .fbx objects. Each fbx object is a 3D model of a hurricane, when all these models are stitched together it simulates a hurricane.

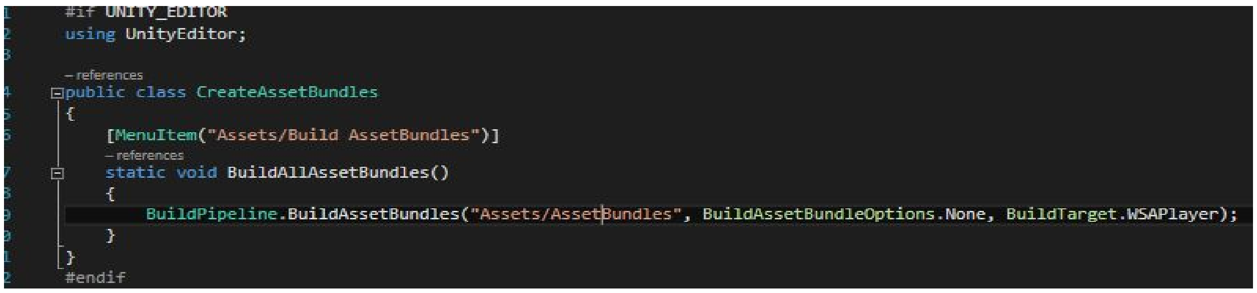
The maximum size of a HoloLens app can be 900MB. Building the hurricane simulation app with all 184 models would make the application size about 2GB. We needed to find a way to limit the memory usage. This was done by using asset bundles.

AssetBundles are files created in the Unity editor during edit-time, which can be used later by a build of a project at run-time. AssetBundles can contain asset files such as models, materials, textures and scenes. AssetBundles cannot contain scripts.

Specifically, an AssetBundle is a collection of assets and/or scenes from a project saved in a compact file with the purpose of being loaded separately to the built executable application. AssetBundles can be loaded on demand by a game or application built in Unity. This allows streaming and asynchronous loading of content such as models, textures, audio clips, or even entire scenes. AssetBundles can be "pre-cached" and stored locally for immediate loading when first running an application. The primary purpose of AssetBundles, however, is to stream content on demand from a remote location, to be loaded into the application as necessary. AssetBundles can contain any kind of asset type recognized by Unity, including custom binary data. The only exception is that script assets are not allowed.

Creating Asset Bundles

First we load this script (CreateAssetBundles) into our Unity environment.

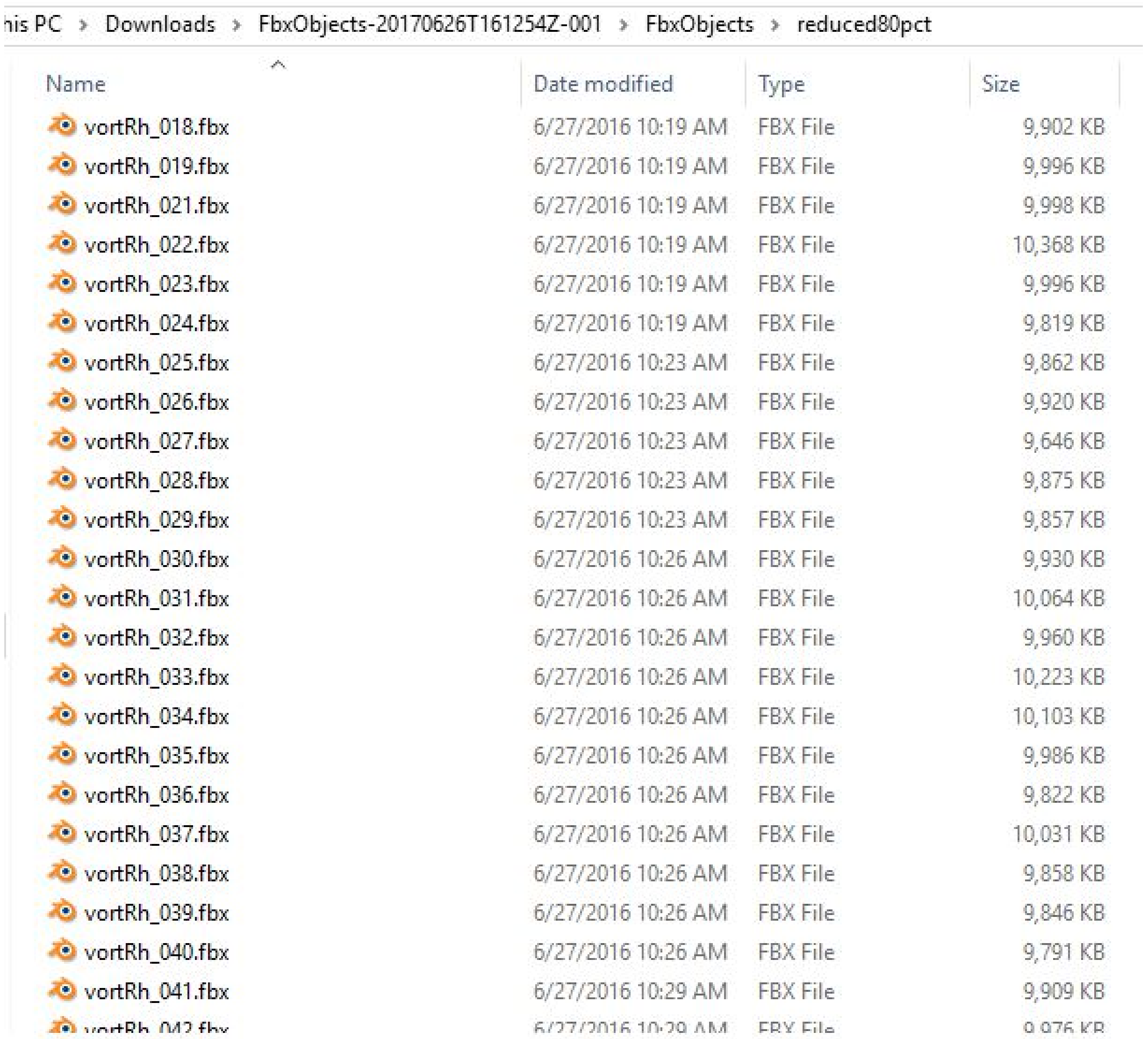


```
#if UNITY_EDITOR
using UnityEditor;

public class CreateAssetBundles
{
    [MenuItem("Assets/Build AssetBundles")]
    static void BuildAllAssetBundles()
    {
        BuildPipeline.BuildAssetBundles("Assets/AssetBundles", BuildAssetBundleOptions.None, BuildTarget.WSAPlayer);
    }
}
#endif
```

Figure 38 : CreateAssetBundles script

Next we create a folder AssetBundles in the Assets folder as specified in the script.

Then we drag and drop our .fbx files into the Unity project panel.



his PC > Downloads > FbxObjects-20170626T161254Z-001 > FbxObjects > reduced80pct

Name	Date modified	Type	Size
vortRh_018.fbx	6/27/2016 10:19 AM	FBX File	9,902 KB
vortRh_019.fbx	6/27/2016 10:19 AM	FBX File	9,996 KB
vortRh_021.fbx	6/27/2016 10:19 AM	FBX File	9,998 KB
vortRh_022.fbx	6/27/2016 10:19 AM	FBX File	10,368 KB
vortRh_023.fbx	6/27/2016 10:19 AM	FBX File	9,996 KB
vortRh_024.fbx	6/27/2016 10:19 AM	FBX File	9,819 KB
vortRh_025.fbx	6/27/2016 10:23 AM	FBX File	9,862 KB
vortRh_026.fbx	6/27/2016 10:23 AM	FBX File	9,920 KB
vortRh_027.fbx	6/27/2016 10:23 AM	FBX File	9,646 KB
vortRh_028.fbx	6/27/2016 10:23 AM	FBX File	9,875 KB
vortRh_029.fbx	6/27/2016 10:23 AM	FBX File	9,857 KB
vortRh_030.fbx	6/27/2016 10:26 AM	FBX File	9,930 KB
vortRh_031.fbx	6/27/2016 10:26 AM	FBX File	10,064 KB
vortRh_032.fbx	6/27/2016 10:26 AM	FBX File	9,960 KB
vortRh_033.fbx	6/27/2016 10:26 AM	FBX File	10,223 KB
vortRh_034.fbx	6/27/2016 10:26 AM	FBX File	10,103 KB
vortRh_035.fbx	6/27/2016 10:26 AM	FBX File	9,986 KB
vortRh_036.fbx	6/27/2016 10:26 AM	FBX File	9,822 KB
vortRh_037.fbx	6/27/2016 10:26 AM	FBX File	10,031 KB
vortRh_038.fbx	6/27/2016 10:26 AM	FBX File	9,858 KB
vortRh_039.fbx	6/27/2016 10:26 AM	FBX File	9,846 KB
vortRh_040.fbx	6/27/2016 10:26 AM	FBX File	9,791 KB
vortRh_041.fbx	6/27/2016 10:29 AM	FBX File	9,909 KB
vortRh_042.fbx	6/27/2016 10:29 AM	FBX File	9,976 KB

Figure 39 : .fbx files of hurricane models

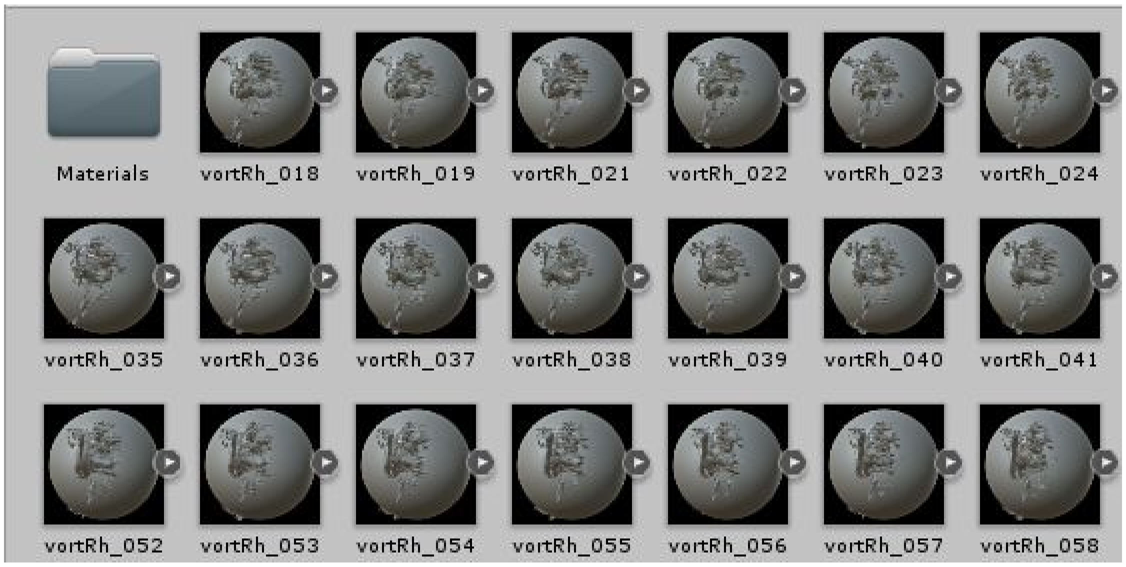


Figure 40 : Hurricane model/assets in Unity

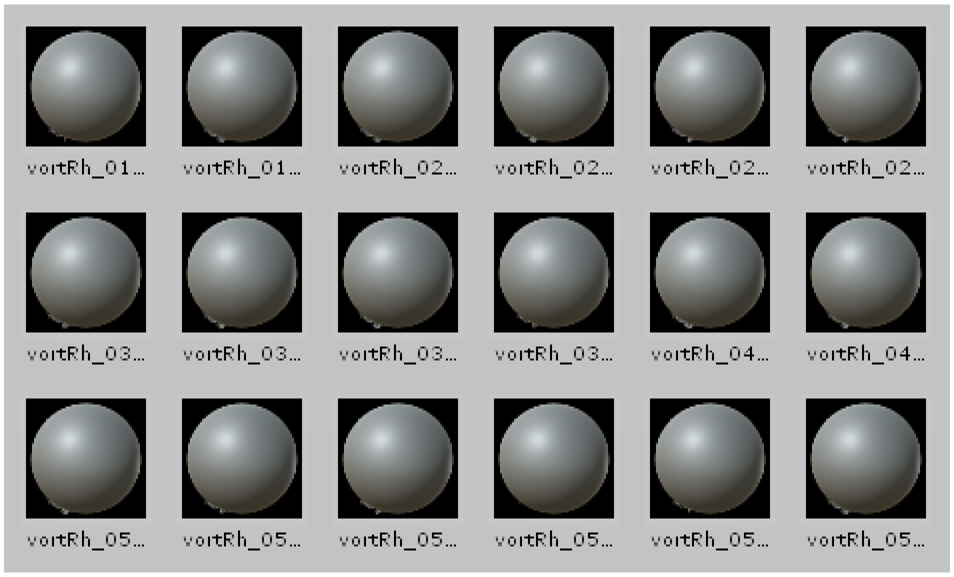


Figure 41 : Hurricane model materials/assets in Unity

Next step is to assign asset bundle names to our models.

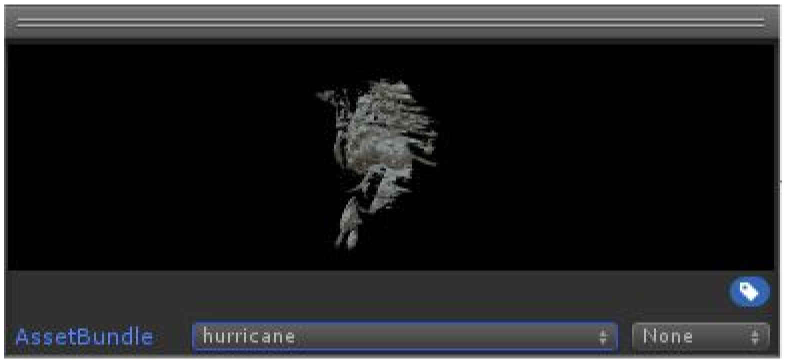

AssetBundle hurricane None

Figure 42 : Assigning asset bundle name hurricane to the model/asset

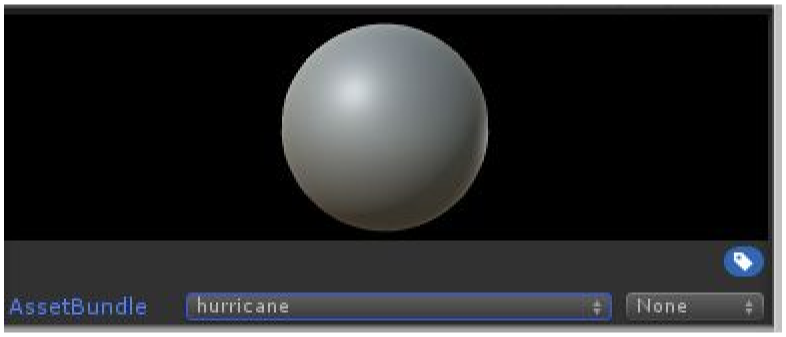

AssetBundle hurricane None

Figure 43 : Assigning asset bundle name hurricane to the material/asset

The models and their respective material need to have the same asset bundle name.

We assigned the asset bundle names as follows:

- For the first 7 models and their respective materials (VortRh_018 - VortRh_025) we assigned the name hurricane. The reason why we have 7 assets in the first bundle will be explained later, when we walk through the code.
- The models from VortRh_26 - VorthRh_202 have the names hurr1, hurr2, hurr3, hurr4, hurr5, hurricane1, hurricane2, …, hurricane172.

So totally we have 178 asset bundles and 184 assets (The assets contain both the model and its respective material).

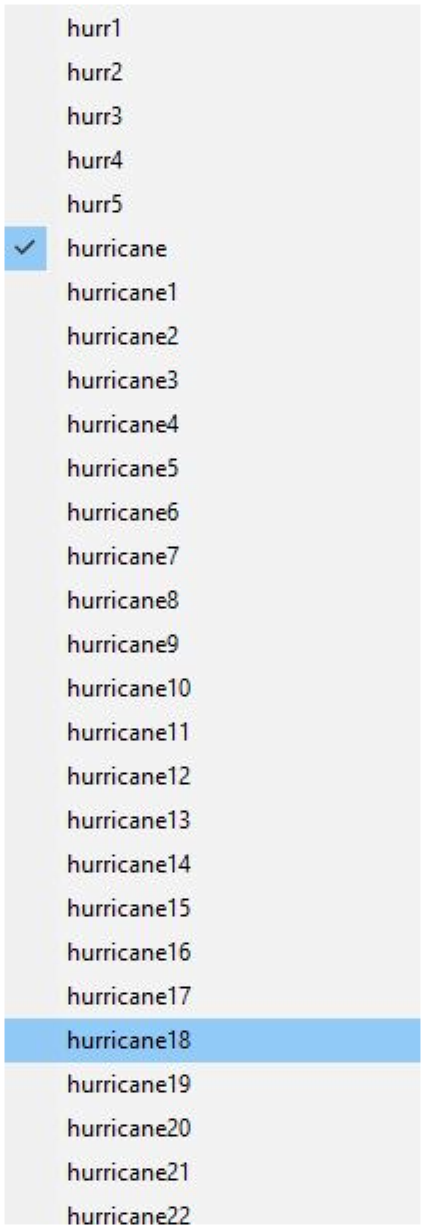


Figure 44 : Asset Bundle names

Once the names are assigned, we use the CreateAssetBuncdles script, when this script is loaded into Unity, the Build AssetBundles menu option appears under the assets menu, we select this and our asset bundles begin building.

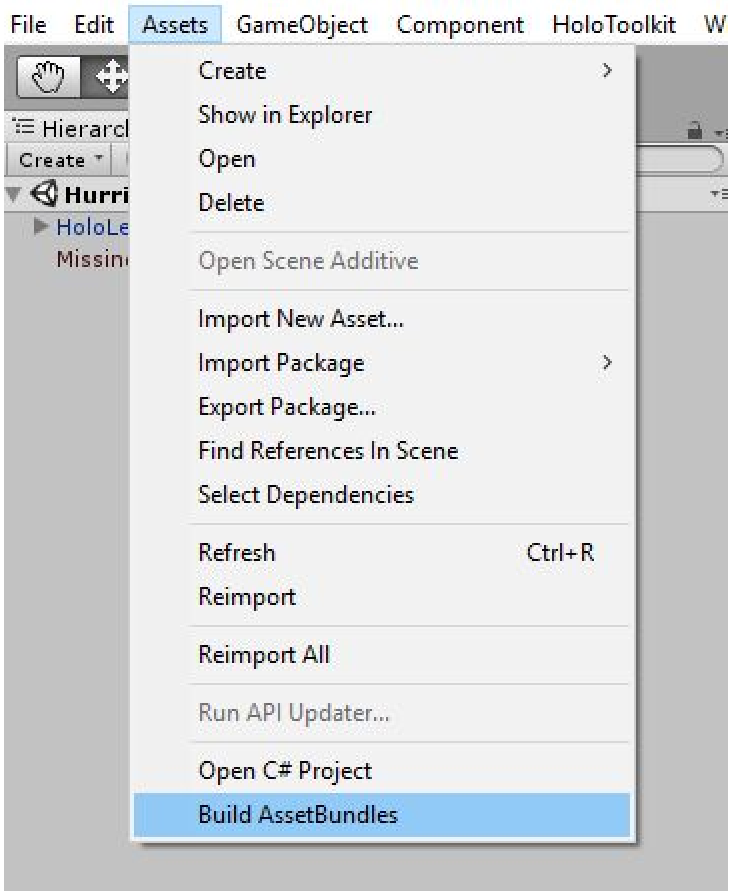
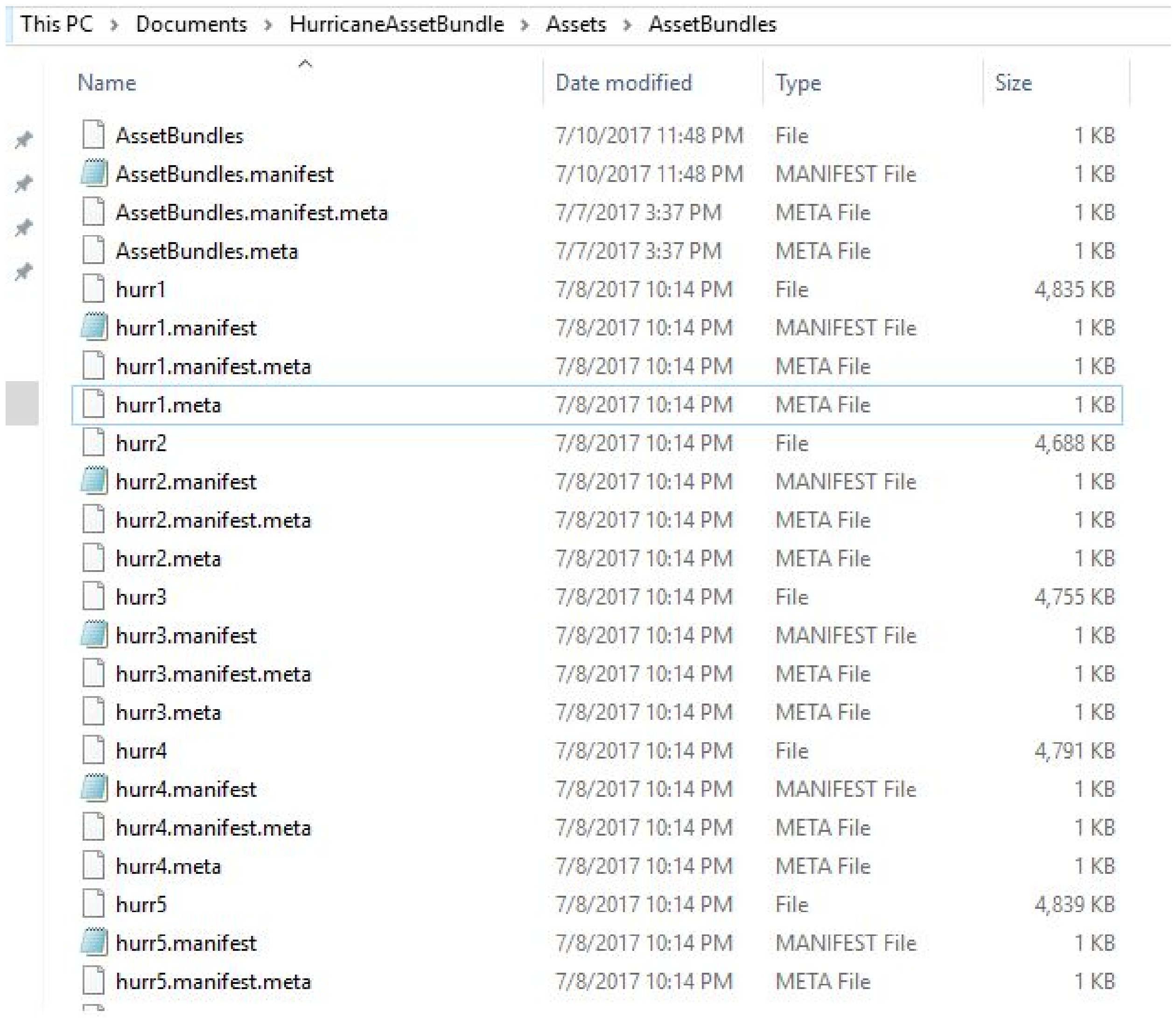


Figure 45 : Building the Asset Bundles

Once the asset bundles have been built, the AssetBundles folder looks like this.



This PC > Documents > HurricaneAssetBundle > Assets > AssetBundles

Name	Date modified	Type	Size
AssetBundles	7/10/2017 11:48 PM	File	1 KB
AssetBundles.manifest	7/10/2017 11:48 PM	MANIFEST File	1 KB
AssetBundles.manifest.meta	7/7/2017 3:37 PM	META File	1 KB
AssetBundles.meta	7/7/2017 3:37 PM	META File	1 KB
hurr1	7/8/2017 10:14 PM	File	4,835 KB
hurr1.manifest	7/8/2017 10:14 PM	MANIFEST File	1 KB
hurr1.manifest.meta	7/8/2017 10:14 PM	META File	1 KB
hurr1.meta	7/8/2017 10:14 PM	META File	1 KB
hurr2	7/8/2017 10:14 PM	File	4,688 KB
hurr2.manifest	7/8/2017 10:14 PM	MANIFEST File	1 KB
hurr2.manifest.meta	7/8/2017 10:14 PM	META File	1 KB
hurr2.meta	7/8/2017 10:14 PM	META File	1 KB
hurr3	7/8/2017 10:14 PM	File	4,755 KB
hurr3.manifest	7/8/2017 10:14 PM	MANIFEST File	1 KB
hurr3.manifest.meta	7/8/2017 10:14 PM	META File	1 KB
hurr3.meta	7/8/2017 10:14 PM	META File	1 KB
hurr4	7/8/2017 10:14 PM	File	4,791 KB
hurr4.manifest	7/8/2017 10:14 PM	MANIFEST File	1 KB
hurr4.manifest.meta	7/8/2017 10:14 PM	META File	1 KB
hurr4.meta	7/8/2017 10:14 PM	META File	1 KB
hurr5	7/8/2017 10:14 PM	File	4,839 KB
hurr5.manifest	7/8/2017 10:14 PM	MANIFEST File	1 KB
hurr5.manifest.meta	7/8/2017 10:14 PM	META File	1 KB

Figure 46 : AssetBundles folder

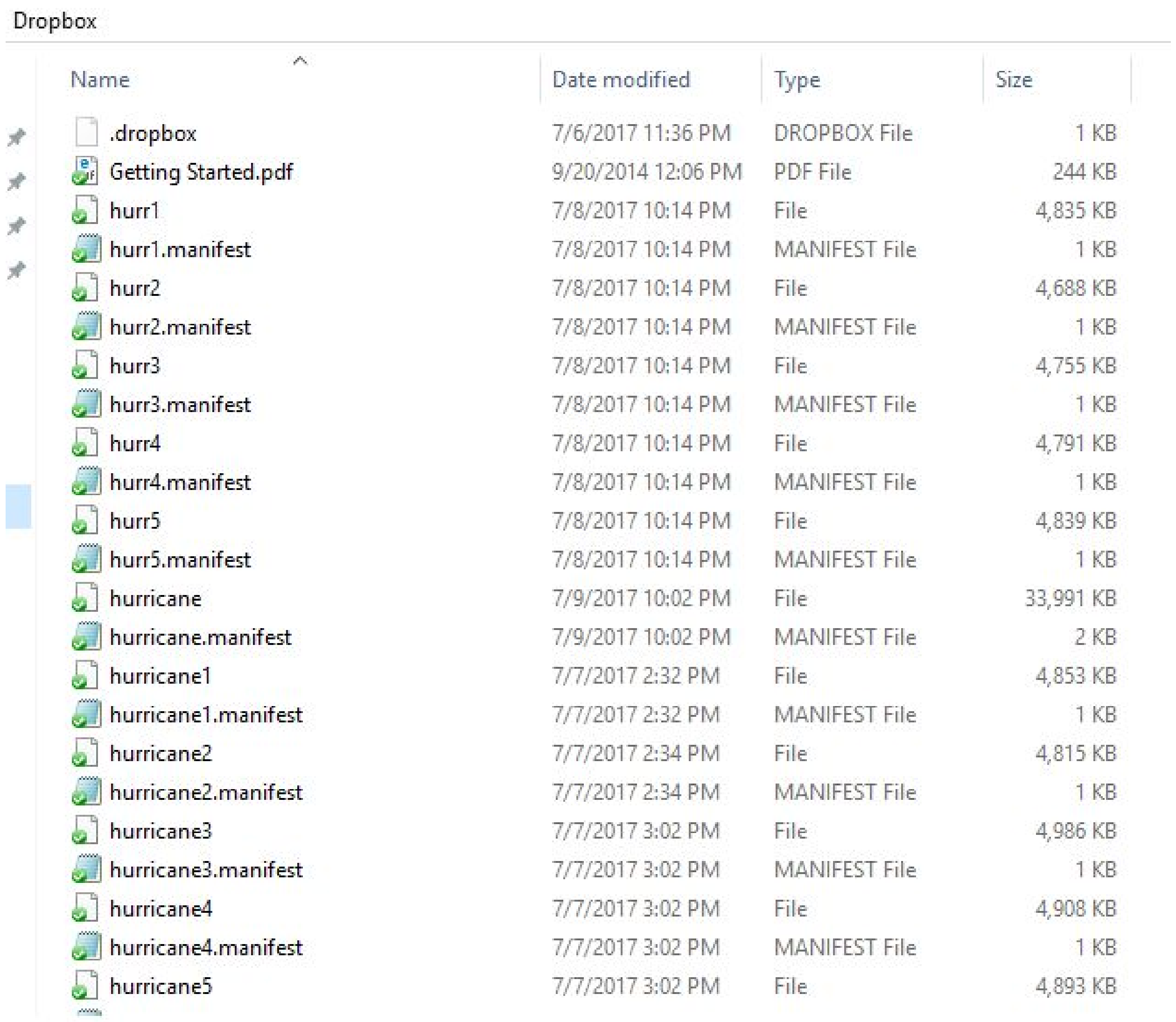
The files without an extension are the asset bundle files. The asset bundles and their assets can be loaded into the application during runtime.

Now we have our asset bundles and their respective assets.

To download these asset bundles we can set up a local server on a system or we can use a remote server (we can use Dropbox etc). We are hosting the asset bundles on a server or dropbox so that we can download these asset bundles from the server during the runtime of the application.

Dropbox

We load our asset bundles into Dropbox by copying and pasting the asset bundles into our Dropbox folder.



Dropbox

Name	Date modified	Type	Size
.dropbox	7/6/2017 11:36 PM	DROPBOX File	1 KB
Getting Started.pdf	9/20/2014 12:06 PM	PDF File	244 KB
hurr1	7/8/2017 10:14 PM	File	4,835 KB
hurr1.manifest	7/8/2017 10:14 PM	MANIFEST File	1 KB
hurr2	7/8/2017 10:14 PM	File	4,688 KB
hurr2.manifest	7/8/2017 10:14 PM	MANIFEST File	1 KB
hurr3	7/8/2017 10:14 PM	File	4,755 KB
hurr3.manifest	7/8/2017 10:14 PM	MANIFEST File	1 KB
hurr4	7/8/2017 10:14 PM	File	4,791 KB
hurr4.manifest	7/8/2017 10:14 PM	MANIFEST File	1 KB
hurr5	7/8/2017 10:14 PM	File	4,839 KB
hurr5.manifest	7/8/2017 10:14 PM	MANIFEST File	1 KB
hurricane	7/9/2017 10:02 PM	File	33,991 KB
hurricane.manifest	7/9/2017 10:02 PM	MANIFEST File	2 KB
hurricane1	7/7/2017 2:32 PM	File	4,853 KB
hurricane1.manifest	7/7/2017 2:32 PM	MANIFEST File	1 KB
hurricane2	7/7/2017 2:34 PM	File	4,815 KB
hurricane2.manifest	7/7/2017 2:34 PM	MANIFEST File	1 KB
hurricane3	7/7/2017 3:02 PM	File	4,986 KB
hurricane3.manifest	7/7/2017 3:02 PM	MANIFEST File	1 KB
hurricane4	7/7/2017 3:02 PM	File	4,908 KB
hurricane4.manifest	7/7/2017 3:02 PM	MANIFEST File	1 KB
hurricane5	7/7/2017 3:02 PM	File	4,893 KB

Figure 47 : Dropbox folder

We need to obtain the download links for these asset bundles, to download the asset bundles into the application, we get the dropbox download links and download the asset bundles using these links.

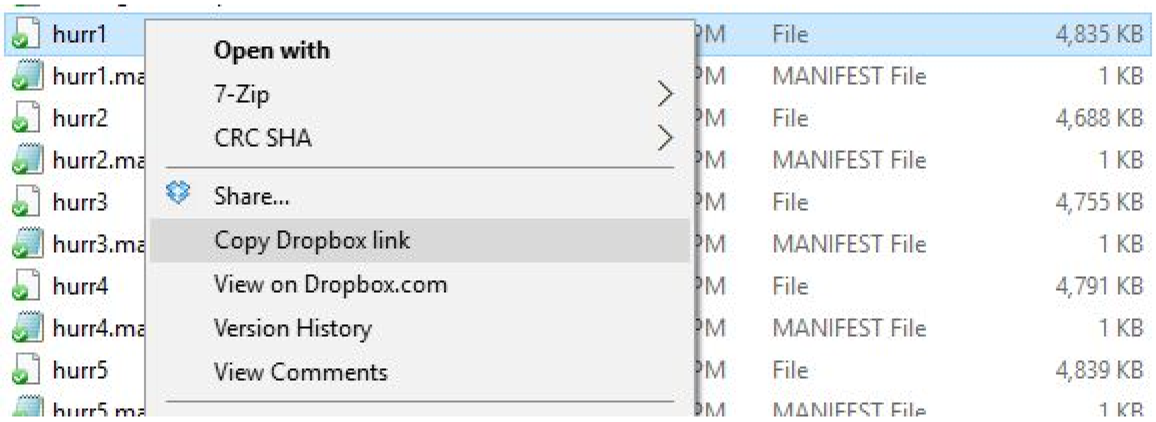


Figure 48 : Dropbox download links

The link would look something like this :

- https://www.dropbox.com/s/vuiwp1d8ispyjp1/hurr1?dl=0
- To download the asset bundles from this link we change the 0 to a 1.
- https://www.dropbox.com/s/vuiwp1d8ispyjp1/hurr1?dl=1

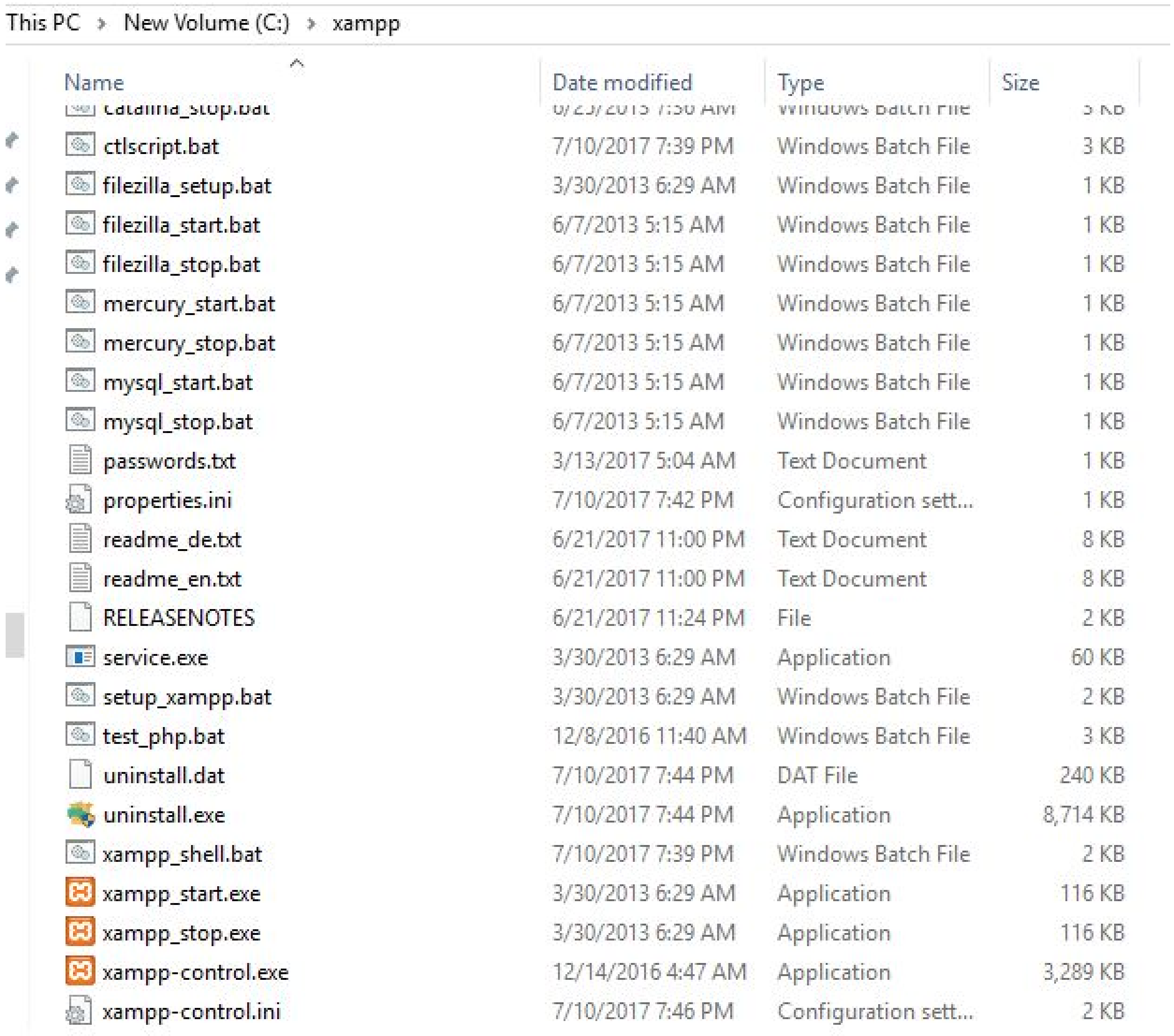
Each asset bundle has a unique link, therefore we will have 178 Dropbox download links, a link for each asset bundle.

Local server

We are going to be using XAMPP and Apache for setting up our local server. To install XAMPP follow the steps on this website :

- https://www.maketecheasier.com/setup-local-web-server-all-platforms/

Once it is installed you can set up the local server by selecting xampp-control.exe, which is present in the xampp folder.

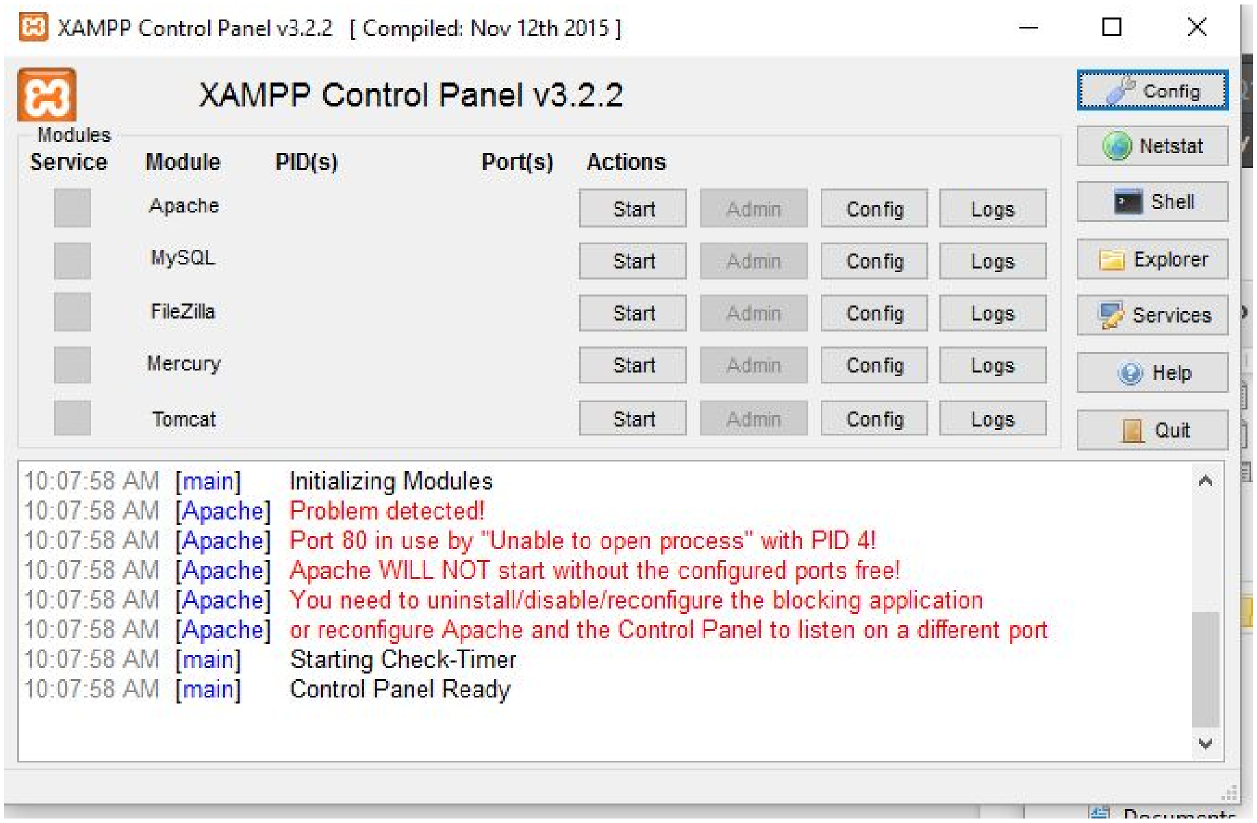


This PC › New Volume (C:) › xampp

Name	Date modified	Type	Size
catalina_stop.bat	6/23/2013 7:56 AM	Windows Batch File	3 KB
ctlscript.bat	7/10/2017 7:39 PM	Windows Batch File	3 KB
filezilla_setup.bat	3/30/2013 6:29 AM	Windows Batch File	1 KB
filezilla_start.bat	6/7/2013 5:15 AM	Windows Batch File	1 KB
filezilla_stop.bat	6/7/2013 5:15 AM	Windows Batch File	1 KB
mercury_start.bat	6/7/2013 5:15 AM	Windows Batch File	1 KB
mercury_stop.bat	6/7/2013 5:15 AM	Windows Batch File	1 KB
mysql_start.bat	6/7/2013 5:15 AM	Windows Batch File	1 KB
mysql_stop.bat	6/7/2013 5:15 AM	Windows Batch File	1 KB
passwords.txt	3/13/2017 5:04 AM	Text Document	1 KB
properties.ini	7/10/2017 7:42 PM	Configuration sett...	1 KB
readme_de.txt	6/21/2017 11:00 PM	Text Document	8 KB
readme_en.txt	6/21/2017 11:00 PM	Text Document	8 KB
RELEASENOTES	6/21/2017 11:24 PM	File	2 KB
service.exe	3/30/2013 6:29 AM	Application	60 KB
setup_xampp.bat	3/30/2013 6:29 AM	Windows Batch File	2 KB
test_php.bat	12/8/2016 11:40 AM	Windows Batch File	3 KB
uninstall.dat	7/10/2017 7:44 PM	DAT File	240 KB
uninstall.exe	7/10/2017 7:44 PM	Application	8,714 KB
xampp_shell.bat	7/10/2017 7:39 PM	Windows Batch File	2 KB
xampp_start.exe	3/30/2013 6:29 AM	Application	116 KB
xampp_stop.exe	3/30/2013 6:29 AM	Application	116 KB
xampp-control.exe	12/14/2016 4:47 AM	Application	3,289 KB
xampp-control.ini	7/10/2017 7:46 PM	Configuration sett...	2 KB

Figure 49 : xampp folder

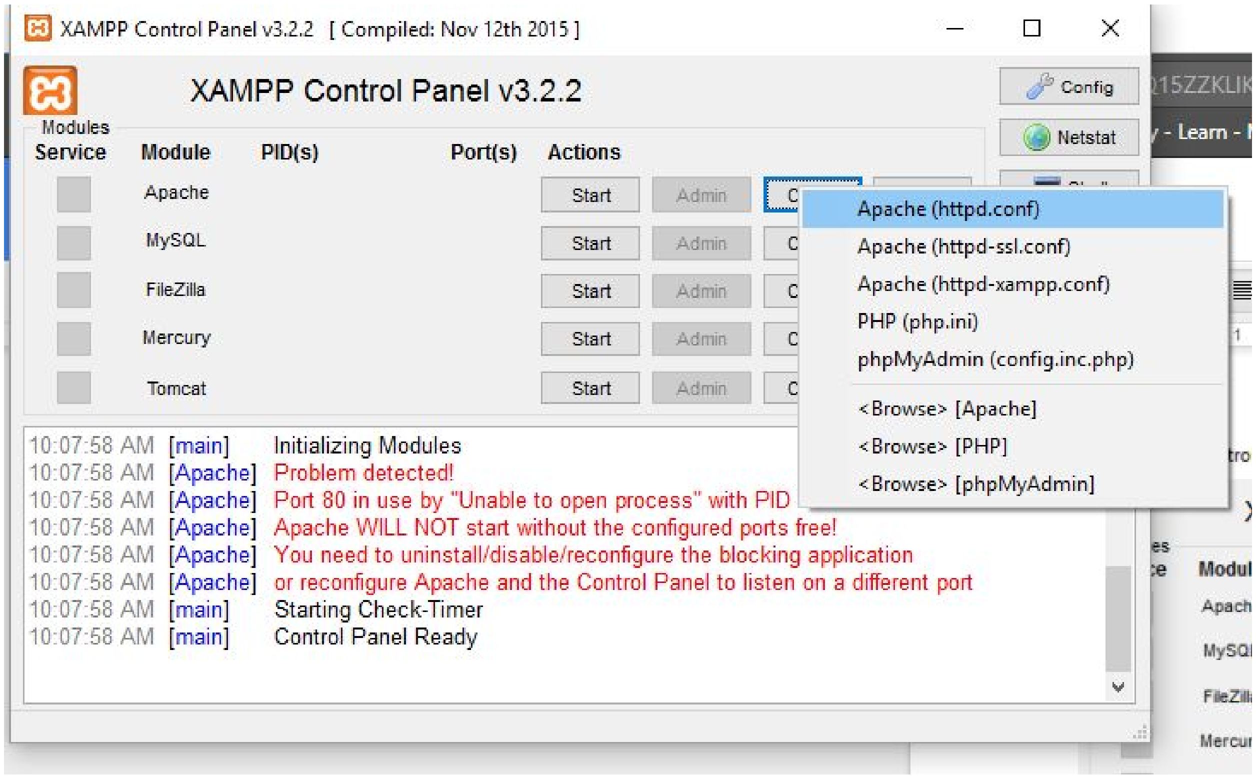
Once you run xampp-control.exe, the following window will pop up.



```
10:07:58 AM  [main]     Initializing Modules
10:07:58 AM  [Apache]   Problem detected!
10:07:58 AM  [Apache]   Port 80 in use by "Unable to open process" with PID 4!
10:07:58 AM  [Apache]   Apache WILL NOT start without the configured ports free!
10:07:58 AM  [Apache]   You need to uninstall/disable/reconfigure the blocking application
10:07:58 AM  [Apache]   or reconfigure Apache and the Control Panel to listen on a different port
10:07:58 AM  [main]     Starting Check-Timer
10:07:58 AM  [main]     Control Panel Ready
```

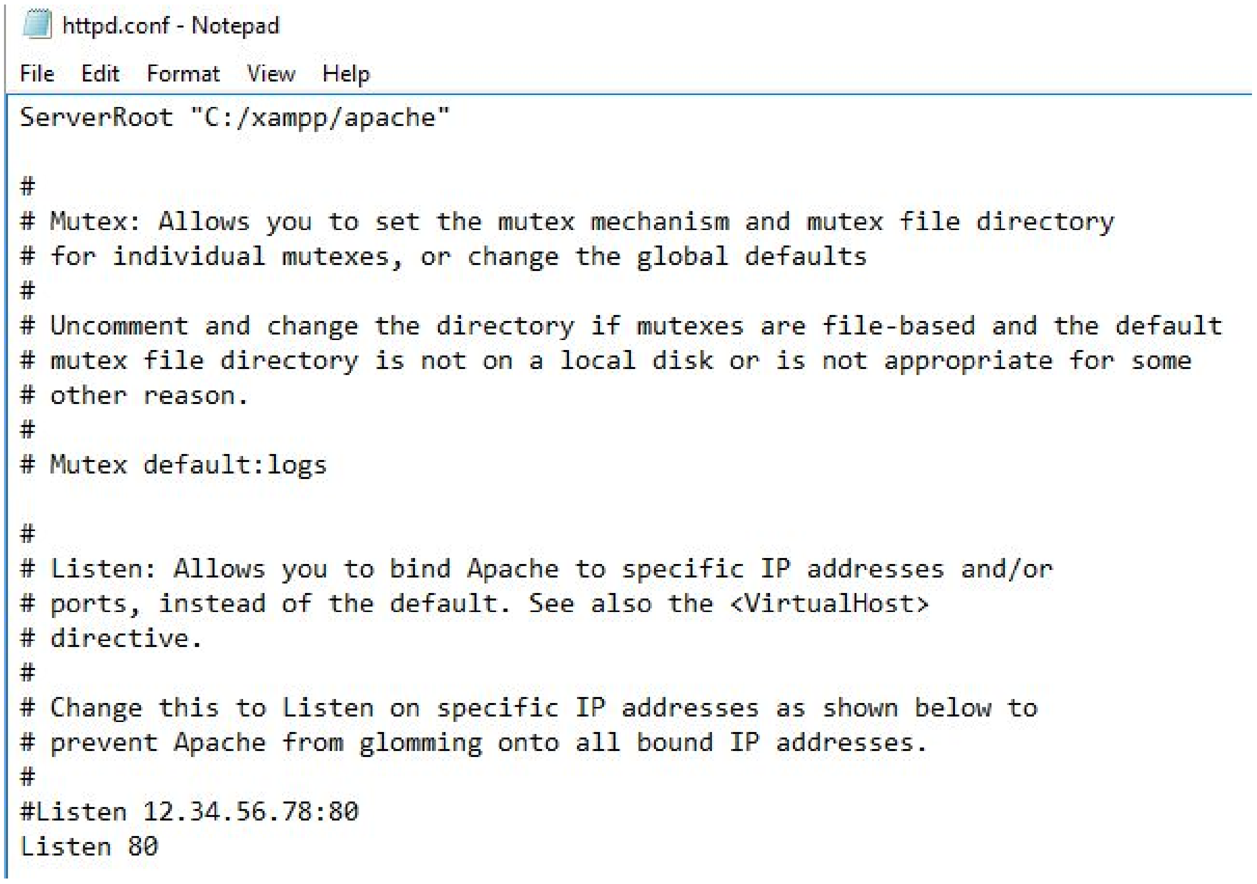
Figure 50 : XAMPP Control Panel

As you can see there is a port error with Apache. To correct this we need to open the httpd.conf file and change the port number. We are going to change it from 80 to 8080 (can be changed to any unused port).



```
10:07:58 AM  [main]     Initializing Modules
10:07:58 AM  [Apache]   Problem detected!
10:07:58 AM  [Apache]   Port 80 in use by "Unable to open process" with PID
10:07:58 AM  [Apache]   Apache WILL NOT start without the configured ports free!
10:07:58 AM  [Apache]   You need to uninstall/disable/reconfigure the blocking application
10:07:58 AM  [Apache]   or reconfigure Apache and the Control Panel to listen on a different port
10:07:58 AM  [main]     Starting Check-Timer
10:07:58 AM  [main]     Control Panel Ready
```

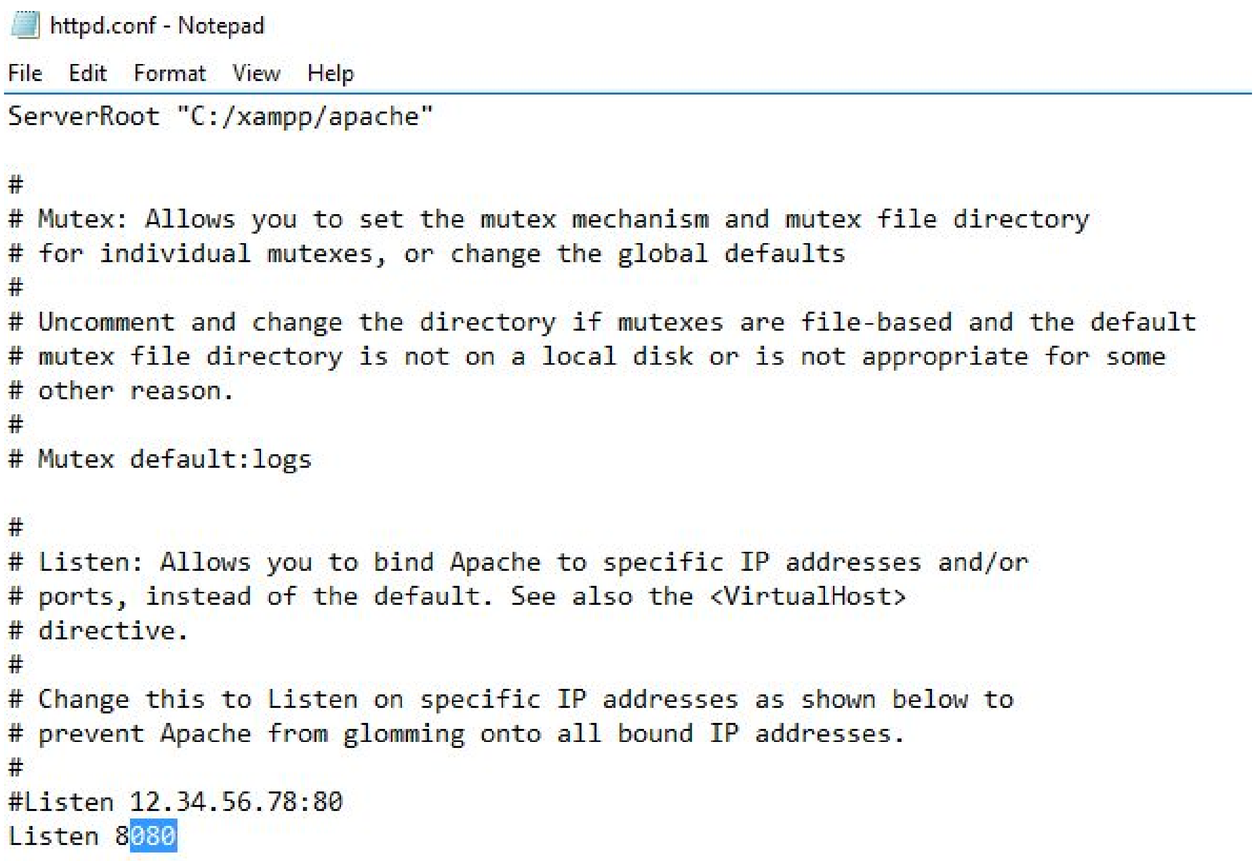
Figure 51 : Opening the httpd.conf file



```
ServerRoot "C:/xampp/apache"

#
# Mutex: Allows you to set the mutex mechanism and mutex file directory
# for individual mutexes, or change the global defaults
#
# Uncomment and change the directory if mutexes are file-based and the default
# mutex file directory is not on a local disk or is not appropriate for some
# other reason.
#
# Mutex default:logs

#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.
#
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.
#
#Listen 12.34.56.78:80
Listen 80
```

Figure 52 : httpd.conf file with port number 80

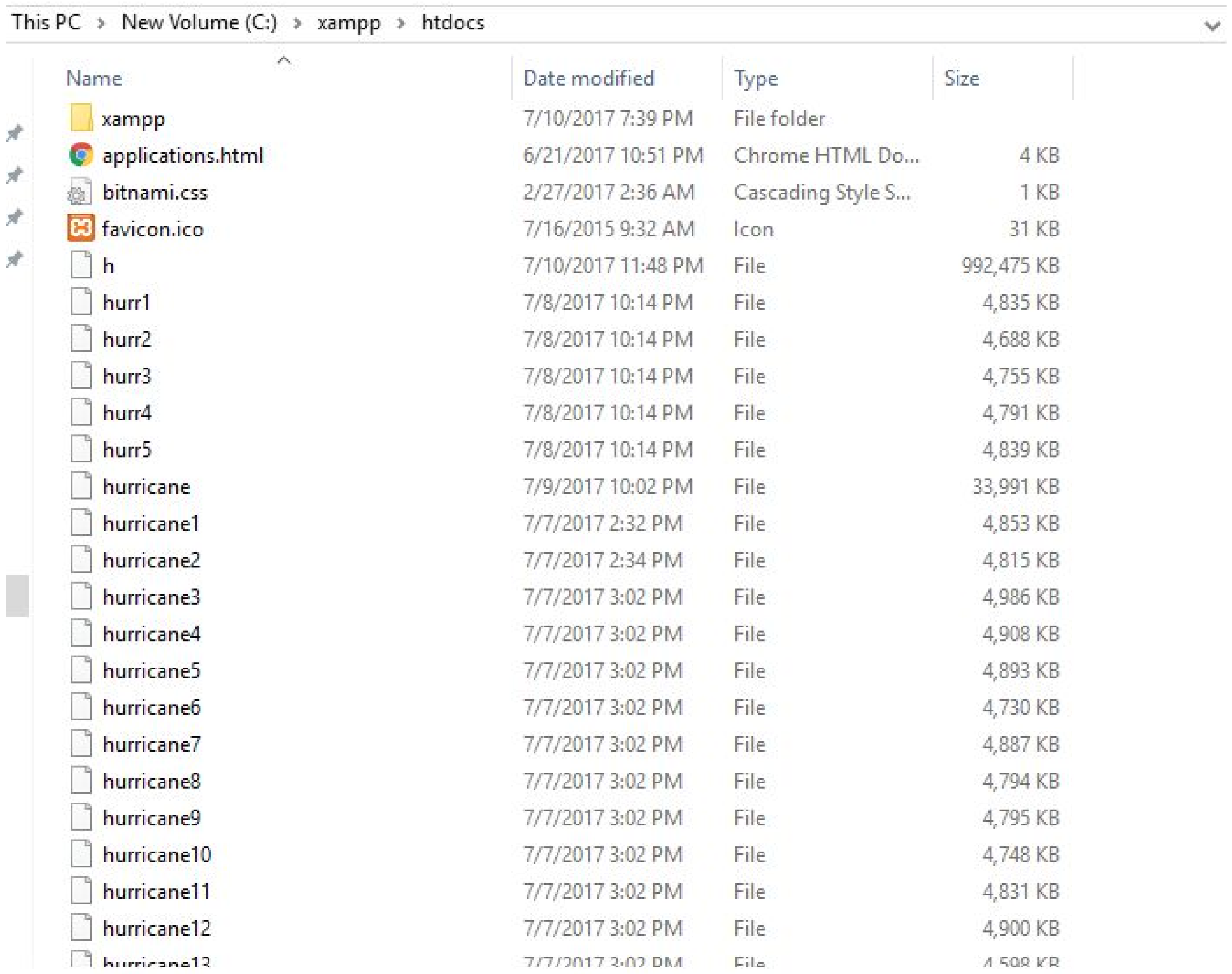
Change from 80 to 8080(or any other unused port).



```
ServerRoot "C:/xampp/apache"

#
# Mutex: Allows you to set the mutex mechanism and mutex file directory
# for individual mutexes, or change the global defaults
#
# Uncomment and change the directory if mutexes are file-based and the default
# mutex file directory is not on a local disk or is not appropriate for some
# other reason.
#
# Mutex default:logs

#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.
#
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.
#
#Listen 12.34.56.78:80
Listen 8080
```

Figure 53 : httpd.conf file with port number 8080

Now to host the files on the local server we copy and paste the asset bundles from the Assets/AssetBundles folder into the htdocs folder which is present in the xampp folder.

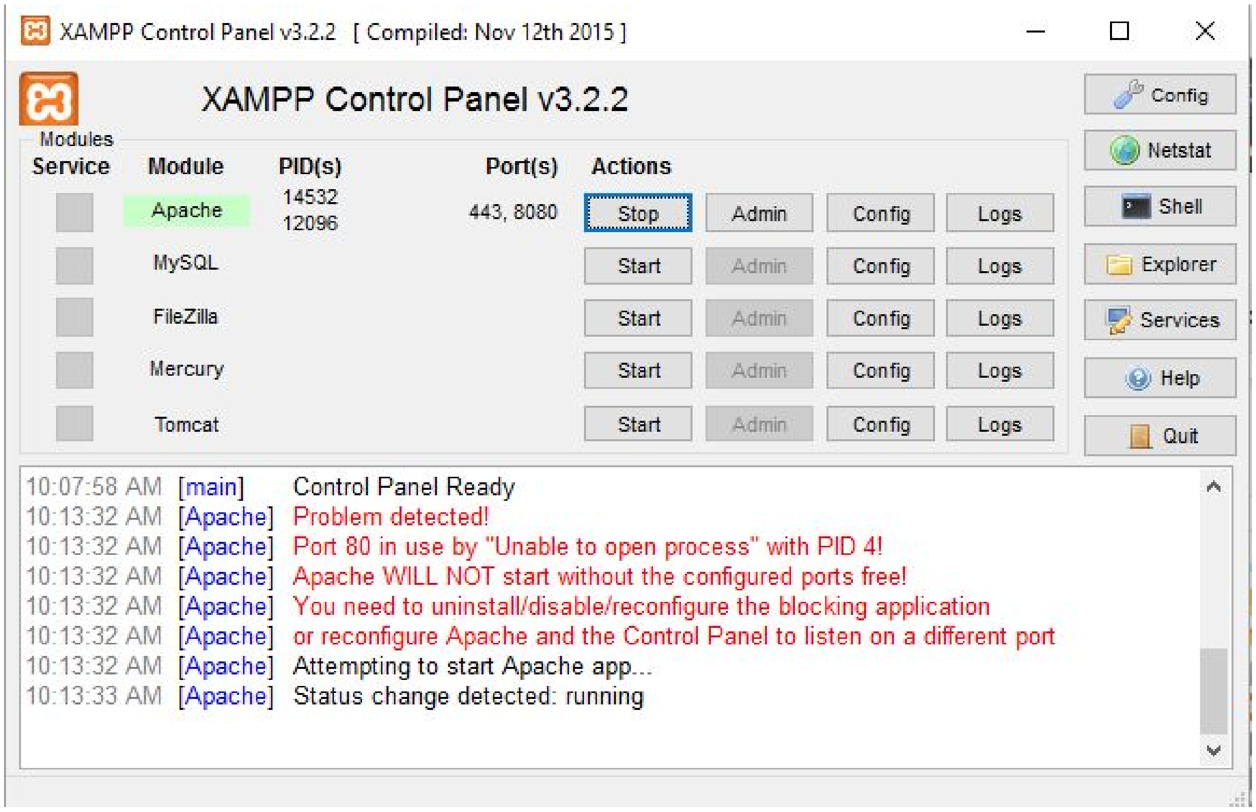


This PC › New Volume (C:) › xampp › htdocs

Name	Date modified	Type	Size
xampp	7/10/2017 7:39 PM	File folder	
applications.html	6/21/2017 10:51 PM	Chrome HTML Do...	4 KB
bitnami.css	2/27/2017 2:36 AM	Cascading Style S...	1 KB
favicon.ico	7/16/2015 9:32 AM	Icon	31 KB
h	7/10/2017 11:48 PM	File	992,475 KB
hurr1	7/8/2017 10:14 PM	File	4,835 KB
hurr2	7/8/2017 10:14 PM	File	4,688 KB
hurr3	7/8/2017 10:14 PM	File	4,755 KB
hurr4	7/8/2017 10:14 PM	File	4,791 KB
hurr5	7/8/2017 10:14 PM	File	4,839 KB
hurricane	7/9/2017 10:02 PM	File	33,991 KB
hurricane1	7/7/2017 2:32 PM	File	4,853 KB
hurricane2	7/7/2017 2:34 PM	File	4,815 KB
hurricane3	7/7/2017 3:02 PM	File	4,986 KB
hurricane4	7/7/2017 3:02 PM	File	4,908 KB
hurricane5	7/7/2017 3:02 PM	File	4,893 KB
hurricane6	7/7/2017 3:02 PM	File	4,730 KB
hurricane7	7/7/2017 3:02 PM	File	4,887 KB
hurricane8	7/7/2017 3:02 PM	File	4,794 KB
hurricane9	7/7/2017 3:02 PM	File	4,795 KB
hurricane10	7/7/2017 3:02 PM	File	4,748 KB
hurricane11	7/7/2017 3:02 PM	File	4,831 KB
hurricane12	7/7/2017 3:02 PM	File	4,900 KB
hurricane13	7/7/2017 3:02 PM	File	4,508 KB

Figure 54 : htdocs folder

To turn on the server we click on the start button in the apache column.



```
10:07:58 AM  [main]     Control Panel Ready
10:13:32 AM  [Apache]   Problem detected!
10:13:32 AM  [Apache]   Port 80 in use by "Unable to open process" with PID 4!
10:13:32 AM  [Apache]   Apache WILL NOT start without the configured ports free!
10:13:32 AM  [Apache]   You need to uninstall/disable/reconfigure the blocking application
10:13:32 AM  [Apache]   or reconfigure Apache and the Control Panel to listen on a different port
10:13:32 AM  [Apache]   Attempting to start Apache app...
10:13:33 AM  [Apache]   Status change detected: running
```

Figure 55 : Apache server running

Now the server is running. To download the asset bundles from the local server, we need a download link. That link will be of the format :

- http://ipaddress:8080/hurricane
- Replace the end of the link with the names of other asset bundles to download them.

"ipaddress" the IP Address of the host machine the server is running on.

The idea behind using asset bundles is to reduce memory usage. What we do is, we have a maximum of 14 asset bundles at any given point in the scene, this is done by using a buffer, the buffer will store a certain number of asset bundles (7) and the assets from these buffered asset bundles will be loaded later into the scene.

We have 7 asset bundles as a buffer in the scene, this means that the asset bundles are loaded into the memory but the asset from these asset bundles have not been loaded onto the scene. At the same time we will have 7 asset bundles that are loaded into the memory and whose assets are being loaded onto the scene.

A walk through the code will help in understanding this better. The following code is going to be using a local server.

First we set the HoloLensCamera, Cursor, and add some directional light.

A Hurricane gameobject which will act as a parent to our loaded hurricane models, so that we can edit the parent and the changes would reflect on all its children. We also have an animation associated with the Hurricane gameobject.

A HurricaneManager gameobject which handles the loading and unloading of assets (GetAssetBundle script) and also contains the HurricaneCapability script which controls the Top View, Front View, Start and Stop animation and reset model capabilities.

A canvas object with buttons, which lets the user input the IP address of the local server.

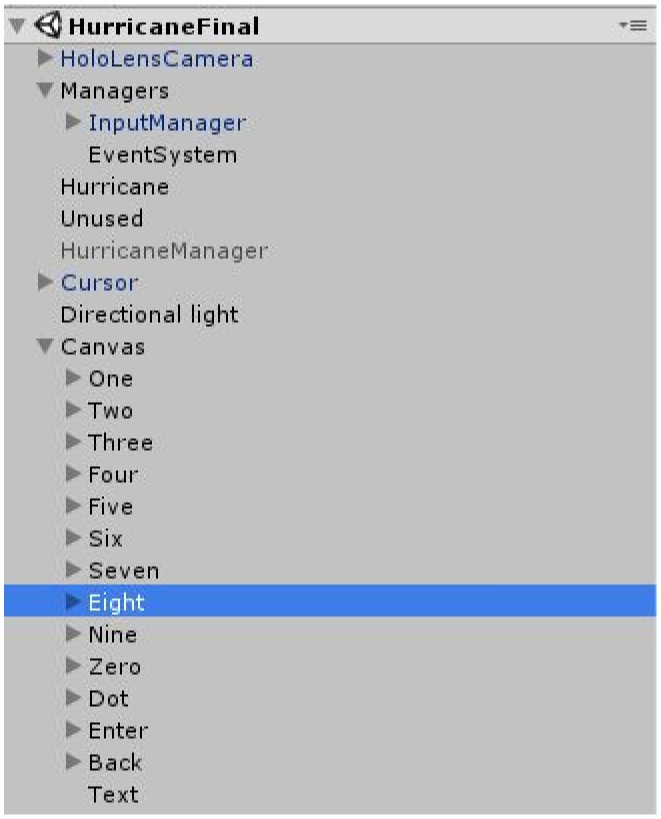
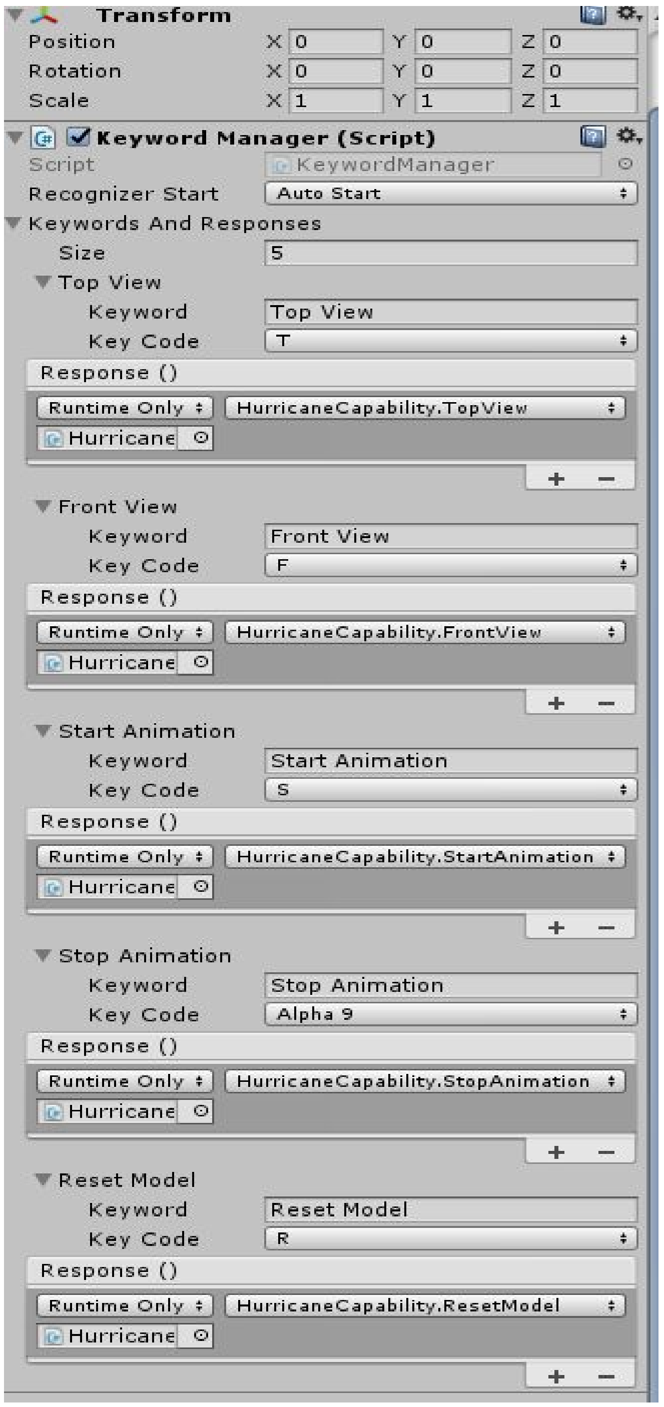

```
HurricaneFinal
  HoloLensCamera
  Managers
    InputManager
    EventSystem
  Hurricane
  Unused
  HurricaneManager
  Cursor
  Directional light
  Canvas
    One
    Two
    Three
    Four
    Five
    Six
    Seven
    Eight
    Nine
    Zero
    Dot
    Enter
    Back
    Text
```

Figure 56 : Scene setup

The Unused gameobject will be explained later.

Managers

The Managers gameobject contains the Input Manager and the Keyword Manager.



Transform

Position	X 0	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

Keyword Manager (Script)

Script	KeywordManager
Recognizer Start	Auto Start
Keywords And Responses	
Size	5

Element	Keyword	Key Code	Response ()
Top View	Top View	T	Runtime Only, HurricaneCapability.TopView, Hurricane
Front View	Front View	F	Runtime Only, HurricaneCapability.FrontView, Hurricane
Start Animation	Start Animation	S	Runtime Only, HurricaneCapability.StartAnimation, Hurricane
Stop Animation	Stop Animation	Alpha 9	Runtime Only, HurricaneCapability.StopAnimation, Hurricane
Reset Model	Reset Model	R	Runtime Only, HurricaneCapability.ResetModel, Hurricane

Figure 57 : Managers gameobject

The Keyword Recognizer is for handling the Top View, Front View, Start and Stop animation and reset model capabilities. It will recognise the keyword and call the functions associated with that keyword, the functions are present in the HurricaneCapability script.

Canvas

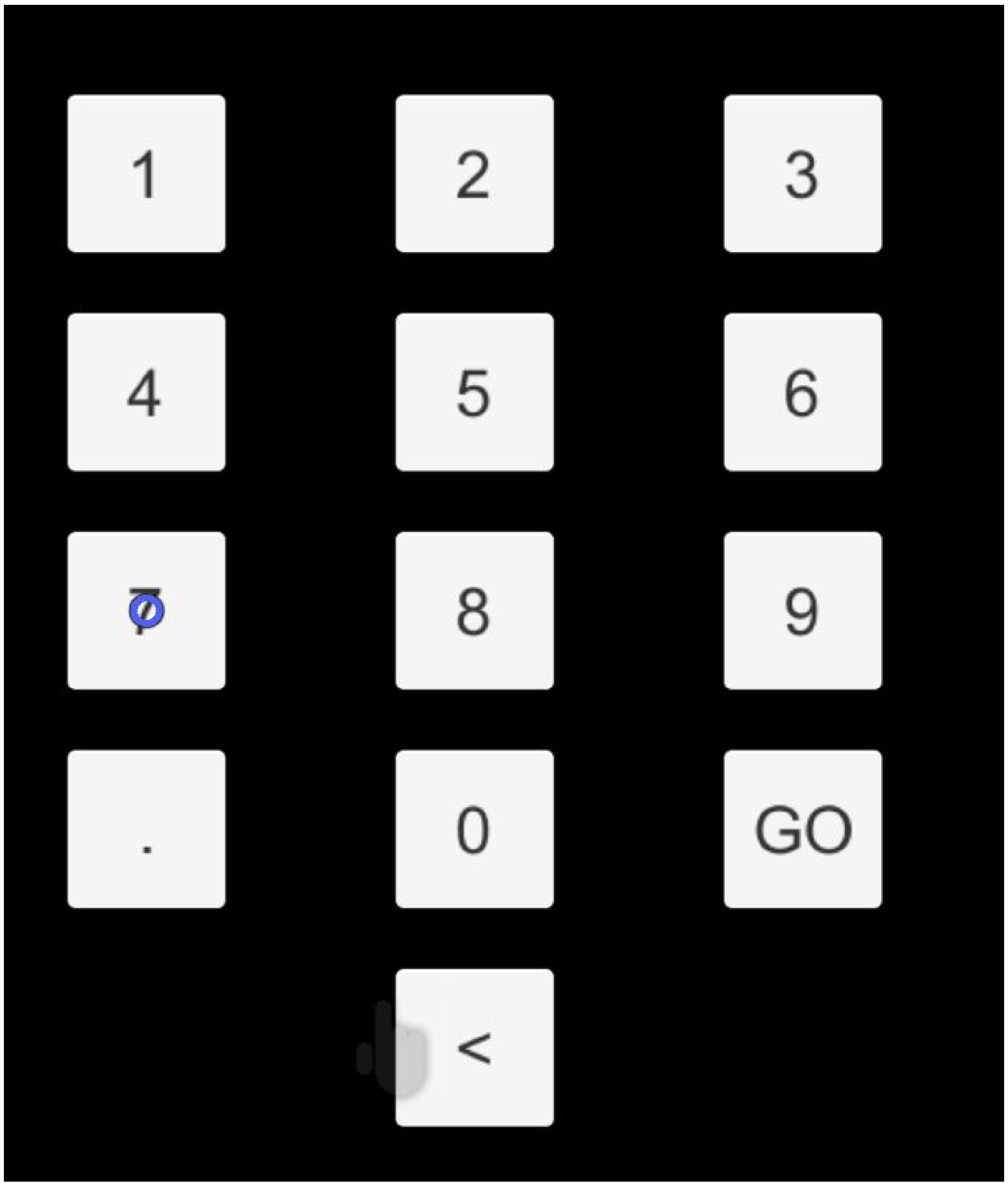
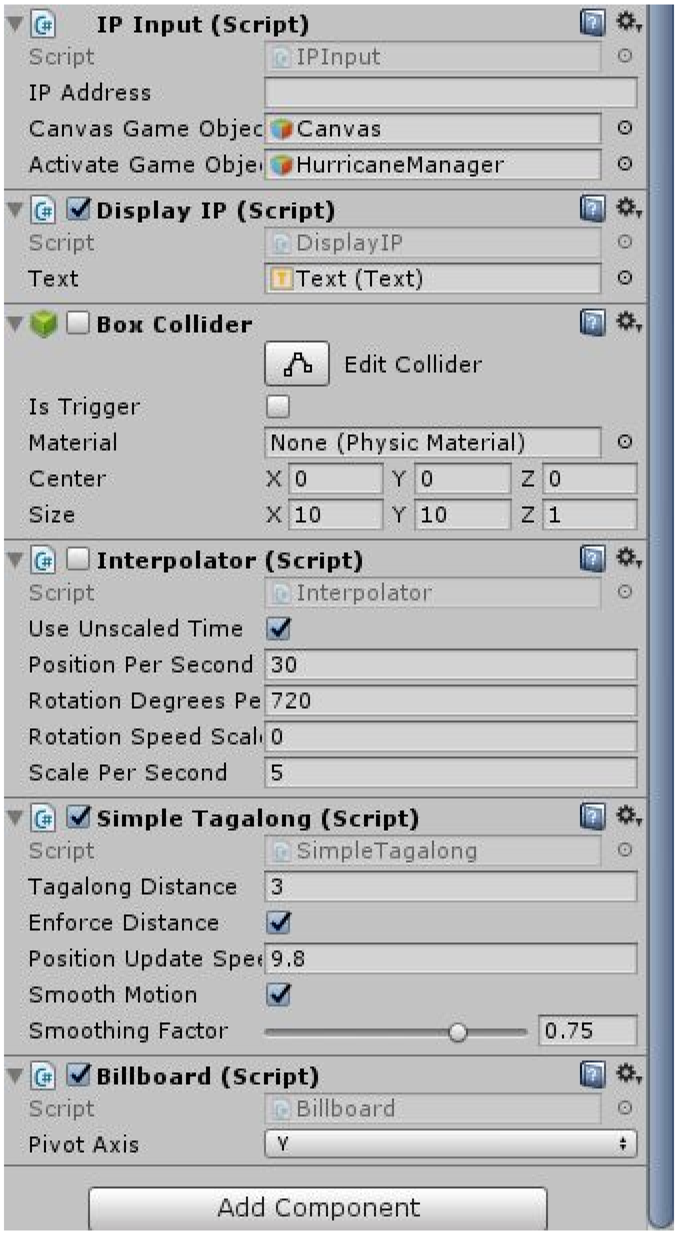


Figure 58 : Canvas with button (IP Address entry)

You enter the IP address by using the buttons and once you are done entering the IP address, you click on the GO button, to begin downloading the asset bundles and run the simulation.

The canvas object has the following components :



IP Input (Script)

Script	IPInput
IP Address	
Canvas Game Objec	Canvas
Activate Game Obje	HurricaneManager

Display IP (Script) ✓

Script	DisplayIP
Text	Text (Text)

Box Collider ☐

Edit Collider

Is Trigger	☐
Material	None (Physic Material)
Center	X 0 Y 0 Z 0
Size	X 10 Y 10 Z 1

Interpolator (Script) ☐

Script	Interpolator
Use Unscaled Time	✓
Position Per Second	30
Rotation Degrees Pe	720
Rotation Speed Scal	0
Scale Per Second	5

Simple Tagalong (Script) ✓

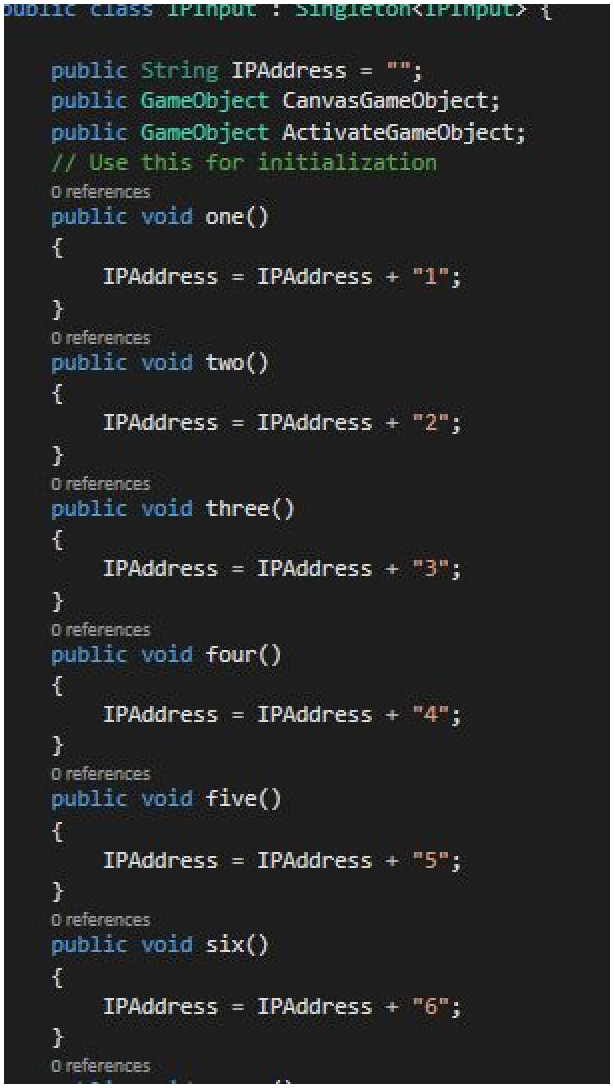
Script	SimpleTagalong
Tagalong Distance	3
Enforce Distance	✓
Position Update Spe	9.8
Smooth Motion	✓
Smoothing Factor	0.75

Billboard (Script) ✓

Script	Billboard
Pivot Axis	Y

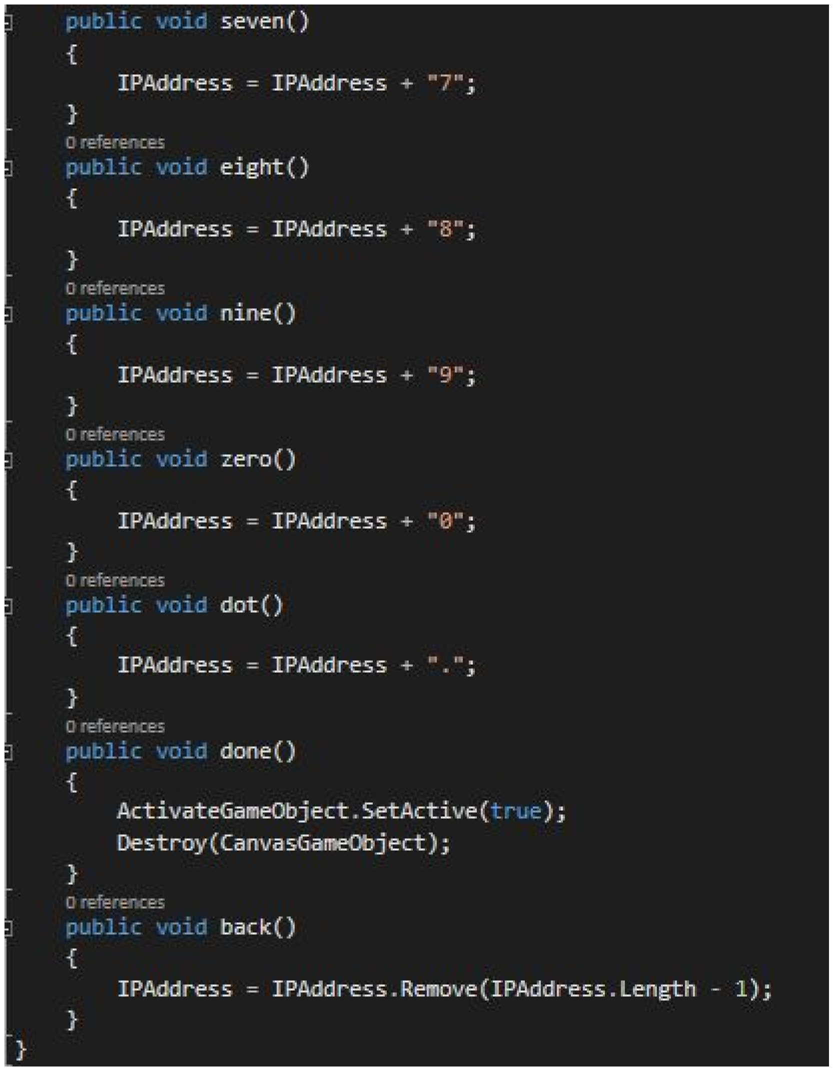
Figure 59 : Canvas gameobject components

The IPInput script is to take the user input for the IP address.



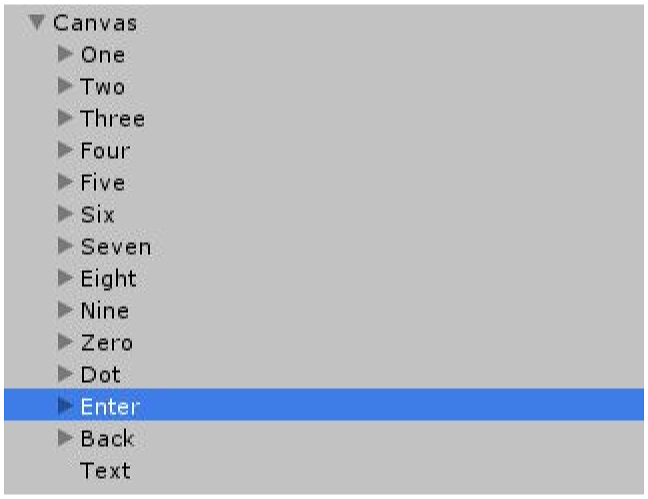
```
public class IPInput : Singleton<IPInput> {

    public String IPAddress = "";
    public GameObject CanvasGameObject;
    public GameObject ActivateGameObject;
    // Use this for initialization
    0 references
    public void one()
    {
        IPAddress = IPAddress + "1";
    }
    0 references
    public void two()
    {
        IPAddress = IPAddress + "2";
    }
    0 references
    public void three()
    {
        IPAddress = IPAddress + "3";
    }
    0 references
    public void four()
    {
        IPAddress = IPAddress + "4";
    }
    0 references
    public void five()
    {
        IPAddress = IPAddress + "5";
    }
    0 references
    public void six()
    {
        IPAddress = IPAddress + "6";
    }
    0 references
```

Figure 60 : IP address input



```
    public void seven()
    {
        IPAddress = IPAddress + "7";
    }
    0 references
    public void eight()
    {
        IPAddress = IPAddress + "8";
    }
    0 references
    public void nine()
    {
        IPAddress = IPAddress + "9";
    }
    0 references
    public void zero()
    {
        IPAddress = IPAddress + "0";
    }
    0 references
    public void dot()
    {
        IPAddress = IPAddress + ".";
    }
    0 references
    public void done()
    {
        ActivateGameObject.SetActive(true);
        Destroy(CanvasGameObject);
    }
    0 references
    public void back()
    {
        IPAddress = IPAddress.Remove(IPAddress.Length - 1);
    }
}
```

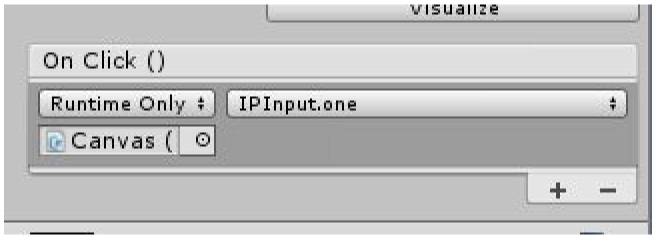
Figure 61 : IP address input



```
▼ Canvas
    ▶ One
    ▶ Two
    ▶ Three
    ▶ Four
    ▶ Five
    ▶ Six
    ▶ Seven
    ▶ Eight
    ▶ Nine
    ▶ Zero
    ▶ Dot
    ▶ Enter
    ▶ Back
      Text
```

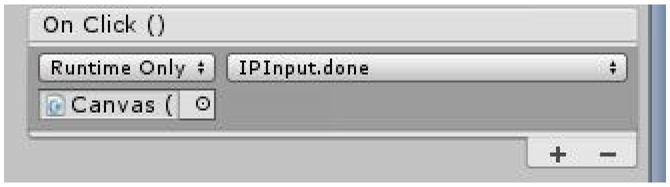
Figure 62 : Canvas gameobject with the buttons

Each of the buttons associated the canvas has an On Click() method, the On Click method() calls the function specified, so we specify the function present in the IPInput script. For the button 1 (Text = 1) we call the One() function and so on, for the button Back (Text = <) we call the back() function which removes the last entered digit or dot and finally for the Enter button (Text = GO) we call the done() function which enables the HurricaneManger object so that we can start downloading the asset bundles from the server that is hosted on the machine with IP address that is entered and then we destroy the Canvas object as we no longer need any input.



On Click ()
Runtime Only | IPInput.one
Canvas (

Figure 63 : Button one (1)

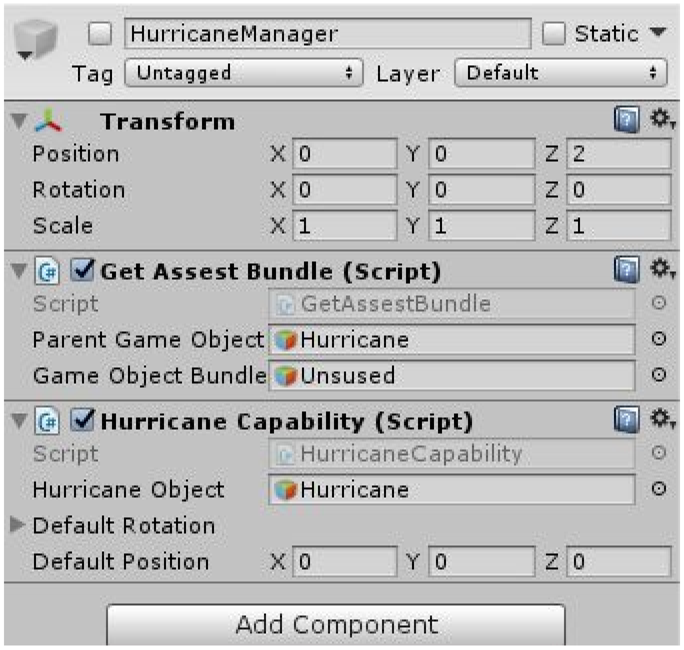


On Click ()
Runtime Only | IPInput.done
Canvas (

Figure 64 : Button Enter (GO)

HurricaneManager

This gameobject is activated once the IP address has been entered. It contains the GetAssetBundles script and the HurricaneCapability script.



HurricaneManager [ ] Static
Tag: Untagged · Layer: Default

Transform

	X	Y	Z
Position	0	0	2
Rotation	0	0	0
Scale	1	1	1

Get Assest Bundle (Script) [✓]

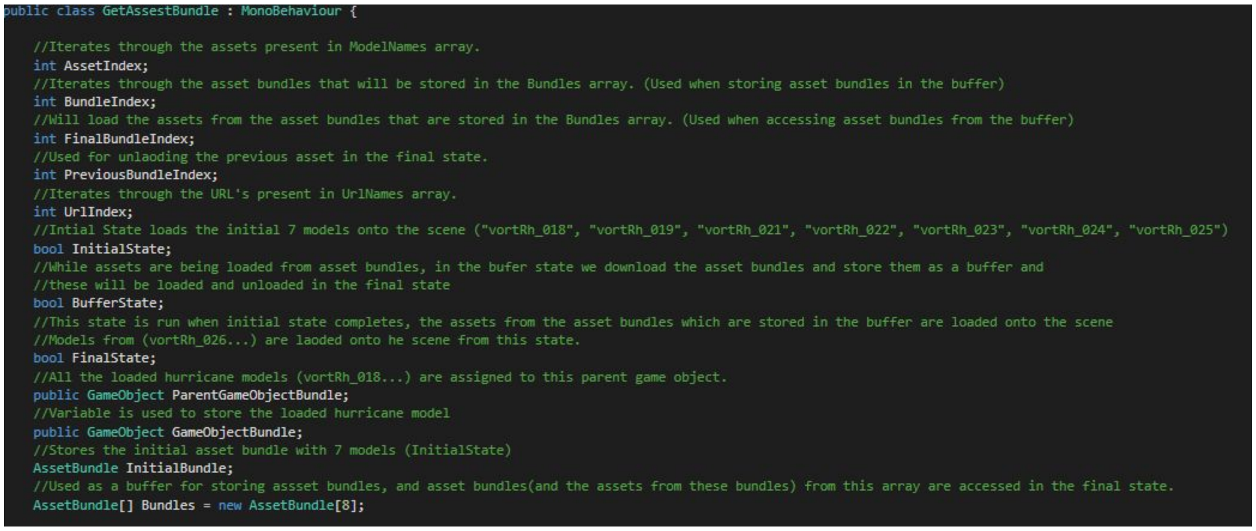
Script: GetAssestBundle
Parent Game Object: Hurricane
Game Object Bundle: Unsused

Hurricane Capability (Script) [✓]

Script: HurricaneCapability
Hurricane Object: Hurricane
Default Rotation
Default Position: X 0 Y 0 Z 0

Figure 65 : HurricaneManager gameobject

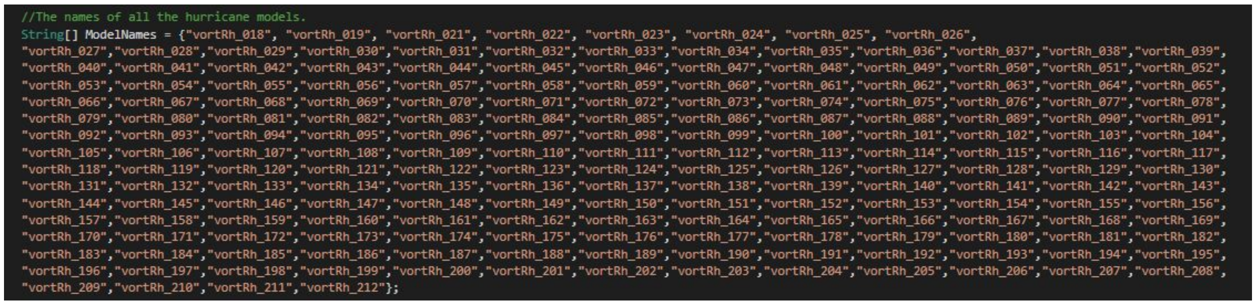
The GetAssetBundle script :



```
public class GetAssestBundle : MonoBehaviour {

    //Iterates through the assets present in ModelNames array.
    int AssetIndex;
    //Iterates through the asset bundles that will be stored in the Bundles array. (Used when storing asset bundles in the buffer)
    int BundleIndex;
    //Will load the assets from the asset bundles that are stored in the Bundles array. (Used when accessing asset bundles from the buffer)
    int FinalBundleIndex;
    //Used for unlaoding the previous asset in the final state.
    int PreviousBundleIndex;
    //Iterates through the URL's present in UrlNames array.
    int UrlIndex;
    //Intial State loads the initial 7 models onto the scene ("vortRh_018", "vortRh_019", "vortRh_021", "vortRh_022", "vortRh_023", "vortRh_024", "vortRh_025")
    bool InitialState;
    //While assets are being loaded from asset bundles, in the bufer state we download the asset bundles and store them as a buffer and
    //these will be loaded and unloaded in the final state
    bool BufferState;
    //This state is run when initial state completes, the assets from the asset bundles which are stored in the buffer are loaded onto the scene
    //Models from (vortRh_026...) are laoded onto he scene from this state.
    bool FinalState;
    //All the loaded hurricane models (vortRh_018...) are assigned to this parent game object.
    public GameObject ParentGameObjectBundle;
    //Variable is used to store the loaded hurricane model
    public GameObject GameObjectBundle;
    //Stores the initial asset bundle with 7 models (InitialState)
    AssetBundle InitialBundle;
    //Used as a buffer for storing assset bundles, and asset bundles(and the assets from these bundles) from this array are accessed in the final state.
    AssetBundle[] Bundles = new AssetBundle[8];
```

Figure 66 : Define the variables

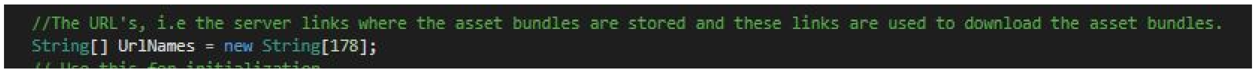
The Bundles array is our buffer. To load an asset from the asset bundle we need the asset names.



```
//The names of all the hurricane models.
String[] ModelNames = {"vortRh_018", "vortRh_019", "vortRh_021", "vortRh_022", "vortRh_023", "vortRh_024", "vortRh_025", "vortRh_026",
"vortRh_027","vortRh_028","vortRh_029","vortRh_030","vortRh_031","vortRh_032","vortRh_033","vortRh_034","vortRh_035","vortRh_036","vortRh_037","vortRh_038","vortRh_039",
"vortRh_040","vortRh_041","vortRh_042","vortRh_043","vortRh_044","vortRh_045","vortRh_046","vortRh_047","vortRh_048","vortRh_049","vortRh_050","vortRh_051","vortRh_052",
"vortRh_053","vortRh_054","vortRh_055","vortRh_056","vortRh_057","vortRh_058","vortRh_059","vortRh_060","vortRh_061","vortRh_062","vortRh_063","vortRh_064","vortRh_065",
"vortRh_066","vortRh_067","vortRh_068","vortRh_069","vortRh_070","vortRh_071","vortRh_072","vortRh_073","vortRh_074","vortRh_075","vortRh_076","vortRh_077","vortRh_078",
"vortRh_079","vortRh_080","vortRh_081","vortRh_082","vortRh_083","vortRh_084","vortRh_085","vortRh_086","vortRh_087","vortRh_088","vortRh_089","vortRh_090","vortRh_091",
"vortRh_092","vortRh_093","vortRh_094","vortRh_095","vortRh_096","vortRh_097","vortRh_098","vortRh_099","vortRh_100","vortRh_101","vortRh_102","vortRh_103","vortRh_104",
"vortRh_105","vortRh_106","vortRh_107","vortRh_108","vortRh_109","vortRh_110","vortRh_111","vortRh_112","vortRh_113","vortRh_114","vortRh_115","vortRh_116","vortRh_117",
"vortRh_118","vortRh_119","vortRh_120","vortRh_121","vortRh_122","vortRh_123","vortRh_124","vortRh_125","vortRh_126","vortRh_127","vortRh_128","vortRh_129","vortRh_130",
"vortRh_131","vortRh_132","vortRh_133","vortRh_134","vortRh_135","vortRh_136","vortRh_137","vortRh_138","vortRh_139","vortRh_140","vortRh_141","vortRh_142","vortRh_143",
"vortRh_144","vortRh_145","vortRh_146","vortRh_147","vortRh_148","vortRh_149","vortRh_150","vortRh_151","vortRh_152","vortRh_153","vortRh_154","vortRh_155","vortRh_156",
"vortRh_157","vortRh_158","vortRh_159","vortRh_160","vortRh_161","vortRh_162","vortRh_163","vortRh_164","vortRh_165","vortRh_166","vortRh_167","vortRh_168","vortRh_169",
"vortRh_170","vortRh_171","vortRh_172","vortRh_173","vortRh_174","vortRh_175","vortRh_176","vortRh_177","vortRh_178","vortRh_179","vortRh_180","vortRh_181","vortRh_182",
"vortRh_183","vortRh_184","vortRh_185","vortRh_186","vortRh_187","vortRh_188","vortRh_189","vortRh_190","vortRh_191","vortRh_192","vortRh_193","vortRh_194","vortRh_195",
"vortRh_196","vortRh_197","vortRh_198","vortRh_199","vortRh_200","vortRh_201","vortRh_202","vortRh_203","vortRh_204","vortRh_205","vortRh_206","vortRh_207","vortRh_208",
"vortRh_209","vortRh_210","vortRh_211","vortRh_212"};
```

Figure 67 : Asset names

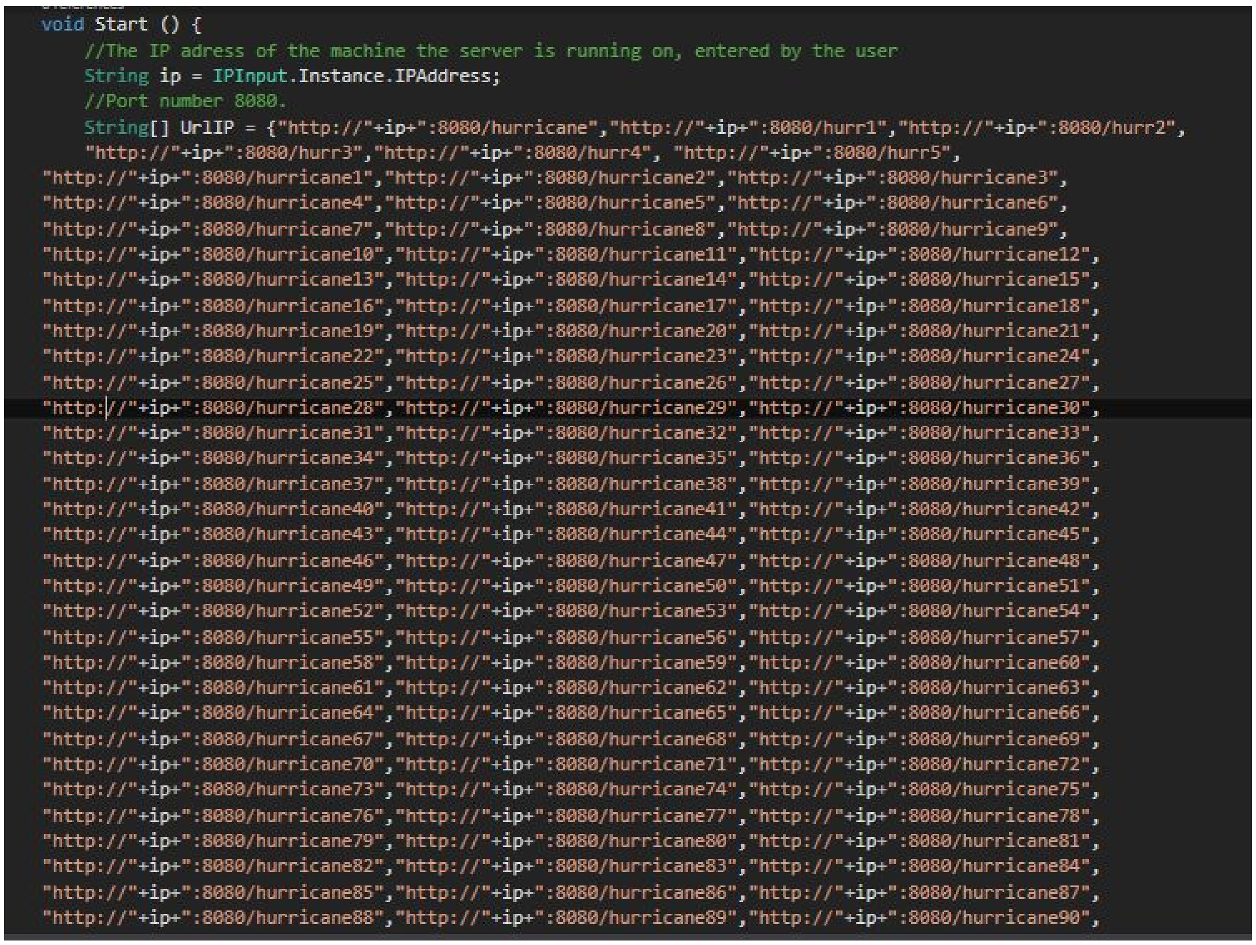
Then we need an array to store the links from which we need to download the asset bundles, the links to the files which are hosted on the local server.



```
//The URL's, i.e the server links where the asset bundles are stored and these links are used to download the asset bundles.
String[] UrlNames = new String[178];
```

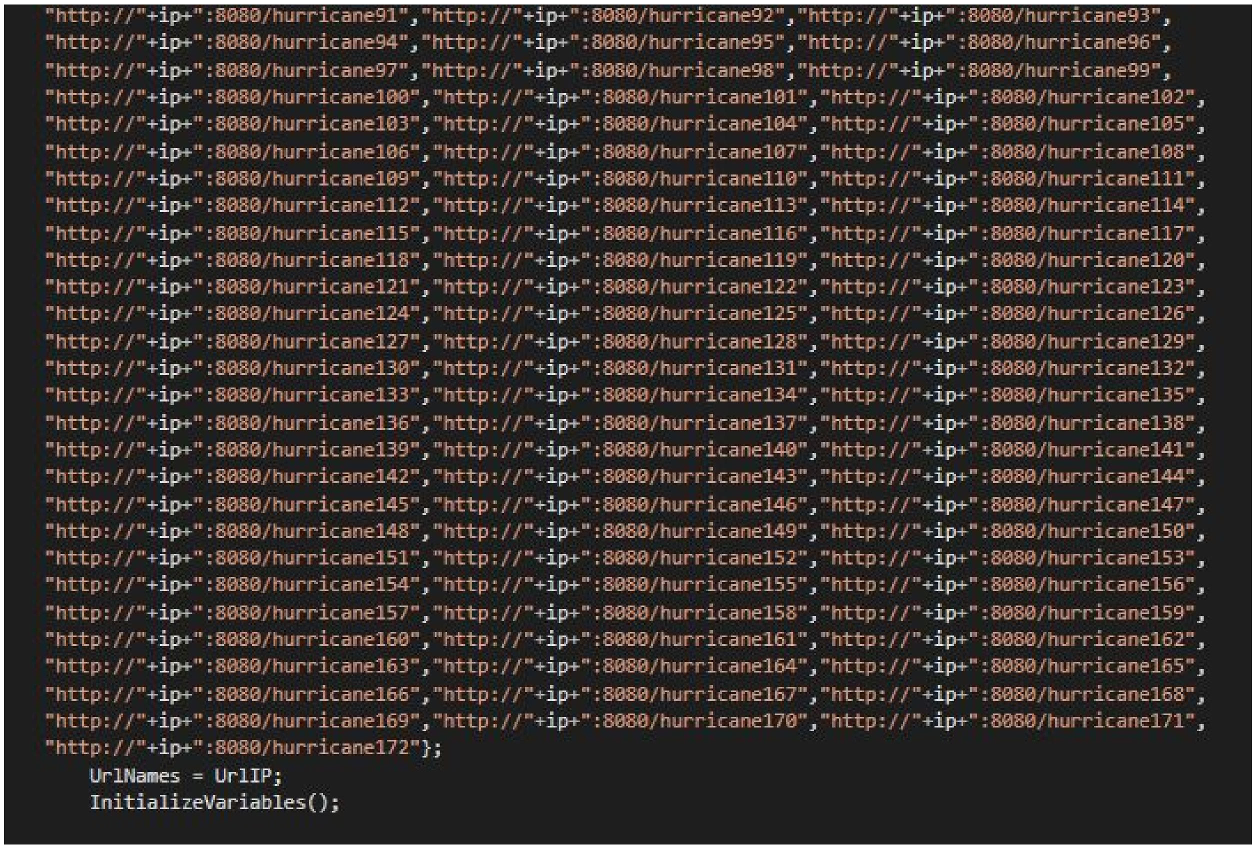
Figure 68 : Array to store asset bundle download links

Once the IP address has been entered, we get the download links to the local server.



```
void Start () {
    //The IP adress of the machine the server is running on, entered by the user
    String ip = IPInput.Instance.IPAddress;
    //Port number 8080.
    String[] UrlIP = {"http://"+ip+":8080/hurricane","http://"+ip+":8080/hurr1","http://"+ip+":8080/hurr2",
    "http://"+ip+":8080/hurr3","http://"+ip+":8080/hurr4", "http://"+ip+":8080/hurr5",
"http://"+ip+":8080/hurricane1","http://"+ip+":8080/hurricane2","http://"+ip+":8080/hurricane3",
"http://"+ip+":8080/hurricane4","http://"+ip+":8080/hurricane5","http://"+ip+":8080/hurricane6",
"http://"+ip+":8080/hurricane7","http://"+ip+":8080/hurricane8","http://"+ip+":8080/hurricane9",
"http://"+ip+":8080/hurricane10","http://"+ip+":8080/hurricane11","http://"+ip+":8080/hurricane12",
"http://"+ip+":8080/hurricane13","http://"+ip+":8080/hurricane14","http://"+ip+":8080/hurricane15",
"http://"+ip+":8080/hurricane16","http://"+ip+":8080/hurricane17","http://"+ip+":8080/hurricane18",
"http://"+ip+":8080/hurricane19","http://"+ip+":8080/hurricane20","http://"+ip+":8080/hurricane21",
"http://"+ip+":8080/hurricane22","http://"+ip+":8080/hurricane23","http://"+ip+":8080/hurricane24",
"http://"+ip+":8080/hurricane25","http://"+ip+":8080/hurricane26","http://"+ip+":8080/hurricane27",
"http://"+ip+":8080/hurricane28","http://"+ip+":8080/hurricane29","http://"+ip+":8080/hurricane30",
"http://"+ip+":8080/hurricane31","http://"+ip+":8080/hurricane32","http://"+ip+":8080/hurricane33",
"http://"+ip+":8080/hurricane34","http://"+ip+":8080/hurricane35","http://"+ip+":8080/hurricane36",
"http://"+ip+":8080/hurricane37","http://"+ip+":8080/hurricane38","http://"+ip+":8080/hurricane39",
"http://"+ip+":8080/hurricane40","http://"+ip+":8080/hurricane41","http://"+ip+":8080/hurricane42",
"http://"+ip+":8080/hurricane43","http://"+ip+":8080/hurricane44","http://"+ip+":8080/hurricane45",
"http://"+ip+":8080/hurricane46","http://"+ip+":8080/hurricane47","http://"+ip+":8080/hurricane48",
"http://"+ip+":8080/hurricane49","http://"+ip+":8080/hurricane50","http://"+ip+":8080/hurricane51",
"http://"+ip+":8080/hurricane52","http://"+ip+":8080/hurricane53","http://"+ip+":8080/hurricane54",
"http://"+ip+":8080/hurricane55","http://"+ip+":8080/hurricane56","http://"+ip+":8080/hurricane57",
"http://"+ip+":8080/hurricane58","http://"+ip+":8080/hurricane59","http://"+ip+":8080/hurricane60",
"http://"+ip+":8080/hurricane61","http://"+ip+":8080/hurricane62","http://"+ip+":8080/hurricane63",
"http://"+ip+":8080/hurricane64","http://"+ip+":8080/hurricane65","http://"+ip+":8080/hurricane66",
"http://"+ip+":8080/hurricane67","http://"+ip+":8080/hurricane68","http://"+ip+":8080/hurricane69",
"http://"+ip+":8080/hurricane70","http://"+ip+":8080/hurricane71","http://"+ip+":8080/hurricane72",
"http://"+ip+":8080/hurricane73","http://"+ip+":8080/hurricane74","http://"+ip+":8080/hurricane75",
"http://"+ip+":8080/hurricane76","http://"+ip+":8080/hurricane77","http://"+ip+":8080/hurricane78",
"http://"+ip+":8080/hurricane79","http://"+ip+":8080/hurricane80","http://"+ip+":8080/hurricane81",
"http://"+ip+":8080/hurricane82","http://"+ip+":8080/hurricane83","http://"+ip+":8080/hurricane84",
"http://"+ip+":8080/hurricane85","http://"+ip+":8080/hurricane86","http://"+ip+":8080/hurricane87",
"http://"+ip+":8080/hurricane88","http://"+ip+":8080/hurricane89","http://"+ip+":8080/hurricane90",
```

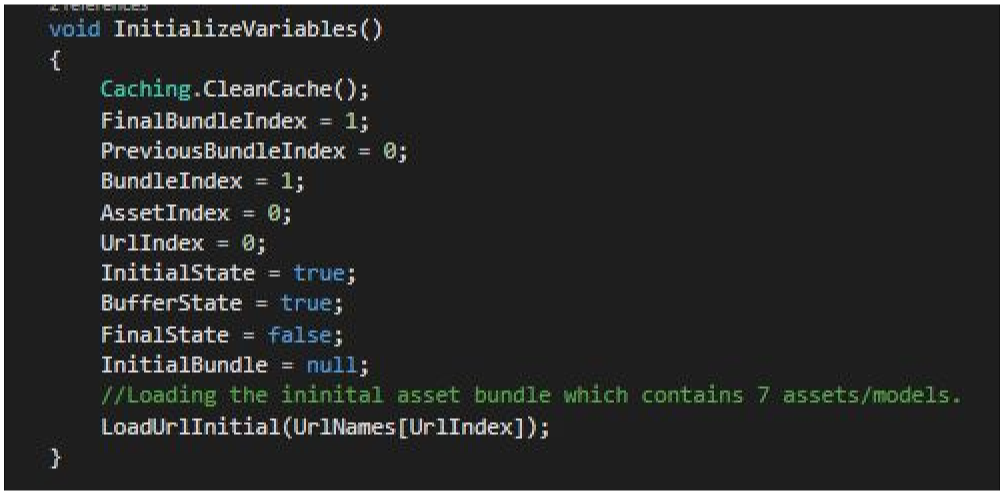
Figure 69 : Local server download links



```
"http://"+ip+":8080/hurricane91","http://"+ip+":8080/hurricane92","http://"+ip+":8080/hurricane93",
"http://"+ip+":8080/hurricane94","http://"+ip+":8080/hurricane95","http://"+ip+":8080/hurricane96",
"http://"+ip+":8080/hurricane97","http://"+ip+":8080/hurricane98","http://"+ip+":8080/hurricane99",
"http://"+ip+":8080/hurricane100","http://"+ip+":8080/hurricane101","http://"+ip+":8080/hurricane102",
"http://"+ip+":8080/hurricane103","http://"+ip+":8080/hurricane104","http://"+ip+":8080/hurricane105",
"http://"+ip+":8080/hurricane106","http://"+ip+":8080/hurricane107","http://"+ip+":8080/hurricane108",
"http://"+ip+":8080/hurricane109","http://"+ip+":8080/hurricane110","http://"+ip+":8080/hurricane111",
"http://"+ip+":8080/hurricane112","http://"+ip+":8080/hurricane113","http://"+ip+":8080/hurricane114",
"http://"+ip+":8080/hurricane115","http://"+ip+":8080/hurricane116","http://"+ip+":8080/hurricane117",
"http://"+ip+":8080/hurricane118","http://"+ip+":8080/hurricane119","http://"+ip+":8080/hurricane120",
"http://"+ip+":8080/hurricane121","http://"+ip+":8080/hurricane122","http://"+ip+":8080/hurricane123",
"http://"+ip+":8080/hurricane124","http://"+ip+":8080/hurricane125","http://"+ip+":8080/hurricane126",
"http://"+ip+":8080/hurricane127","http://"+ip+":8080/hurricane128","http://"+ip+":8080/hurricane129",
"http://"+ip+":8080/hurricane130","http://"+ip+":8080/hurricane131","http://"+ip+":8080/hurricane132",
"http://"+ip+":8080/hurricane133","http://"+ip+":8080/hurricane134","http://"+ip+":8080/hurricane135",
"http://"+ip+":8080/hurricane136","http://"+ip+":8080/hurricane137","http://"+ip+":8080/hurricane138",
"http://"+ip+":8080/hurricane139","http://"+ip+":8080/hurricane140","http://"+ip+":8080/hurricane141",
"http://"+ip+":8080/hurricane142","http://"+ip+":8080/hurricane143","http://"+ip+":8080/hurricane144",
"http://"+ip+":8080/hurricane145","http://"+ip+":8080/hurricane146","http://"+ip+":8080/hurricane147",
"http://"+ip+":8080/hurricane148","http://"+ip+":8080/hurricane149","http://"+ip+":8080/hurricane150",
"http://"+ip+":8080/hurricane151","http://"+ip+":8080/hurricane152","http://"+ip+":8080/hurricane153",
"http://"+ip+":8080/hurricane154","http://"+ip+":8080/hurricane155","http://"+ip+":8080/hurricane156",
"http://"+ip+":8080/hurricane157","http://"+ip+":8080/hurricane158","http://"+ip+":8080/hurricane159",
"http://"+ip+":8080/hurricane160","http://"+ip+":8080/hurricane161","http://"+ip+":8080/hurricane162",
"http://"+ip+":8080/hurricane163","http://"+ip+":8080/hurricane164","http://"+ip+":8080/hurricane165",
"http://"+ip+":8080/hurricane166","http://"+ip+":8080/hurricane167","http://"+ip+":8080/hurricane168",
"http://"+ip+":8080/hurricane169","http://"+ip+":8080/hurricane170","http://"+ip+":8080/hurricane171",
"http://"+ip+":8080/hurricane172"};
    UrlNames = UrlIP;
    InitializeVariables();
```

Figure 70 : Local server download links

We now have our 178 links which contain the asset bundles and the assets. Next we initialize our variables and download the first asset bundle, the first asset bundle contains 7 assets.

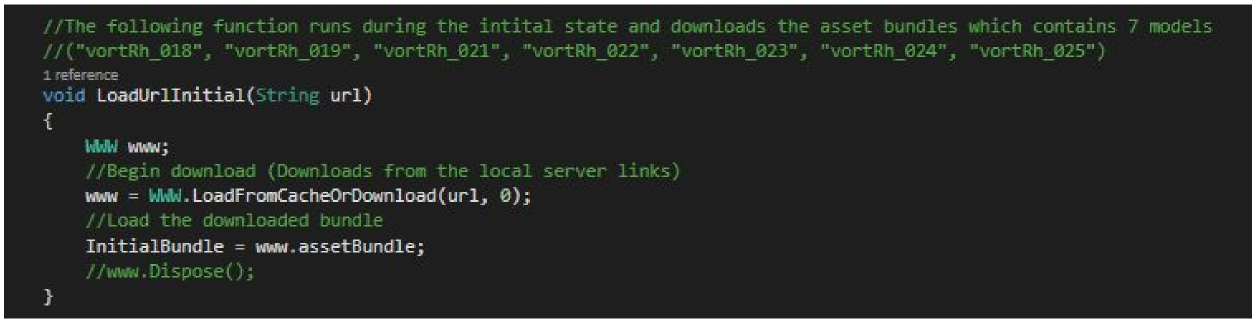


```
void InitializeVariables()
{
    Caching.CleanCache();
    FinalBundleIndex = 1;
    PreviousBundleIndex = 0;
    BundleIndex = 1;
    AssetIndex = 0;
    UrlIndex = 0;
    InitialState = true;
    BufferState = true;
    FinalState = false;
    InitialBundle = null;
    //Loading the ininital asset bundle which contains 7 assets/models.
    LoadUrlInitial(UrlNames[UrlIndex]);
}
```

Figure 71 : Initialise variables and download the first asset bundle

Initial State

LoadUrlInitial loads the first asset bundles into memory and the asset bundle is stored in InitialBundle variable.

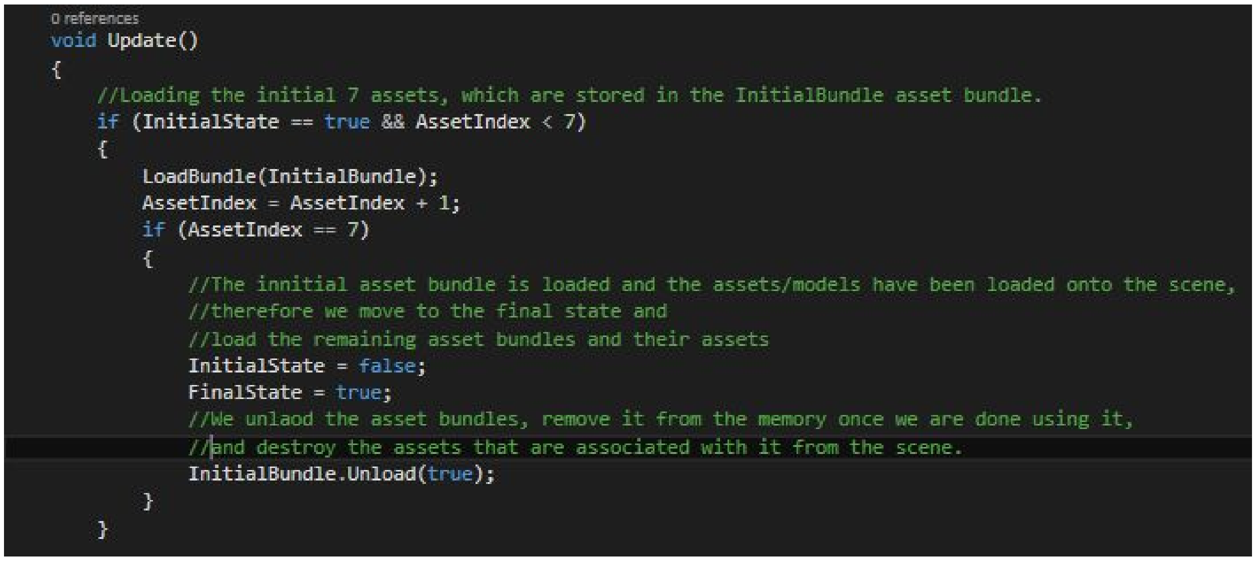


```
//The following function runs during the intital state and downloads the asset bundles which contains 7 models
//("vortRh_018", "vortRh_019", "vortRh_021", "vortRh_022", "vortRh_023", "vortRh_024", "vortRh_025")
1 reference
void LoadUrlInitial(String url)
{
    WWW www;
    //Begin download (Downloads from the local server links)
    www = WWW.LoadFromCacheOrDownload(url, 0);
    //Load the downloaded bundle
    InitialBundle = www.assetBundle;
    //www.Dispose();
}
```

Figure 72 : LoadUrlInitial

The variable InitialBundle contains our initial asset bundle which contains 7 assets/models.

Next we begin loading our assets, first we load these 7 assets and while these assets are being loaded onto the scene we start loading asset bundles into the buffer.

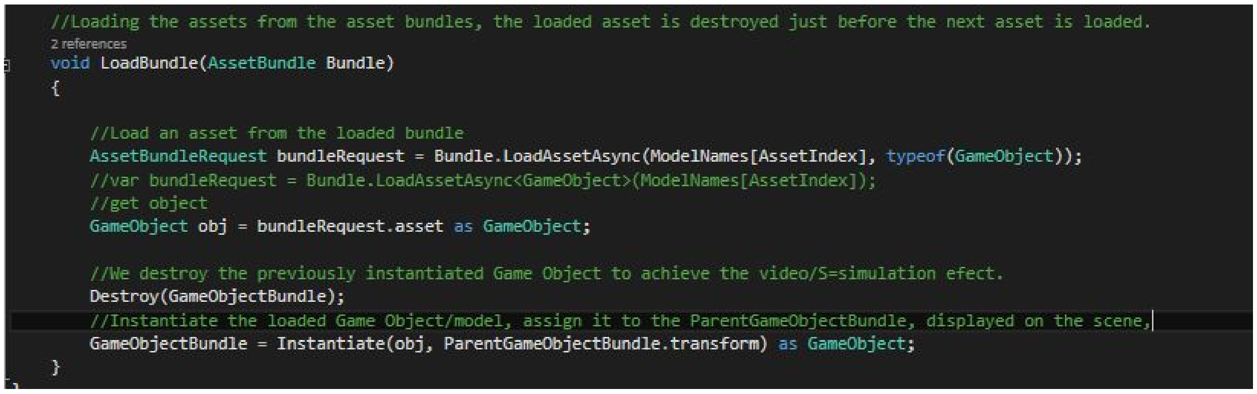


```
0 references
void Update()
{
    //Loading the initial 7 assets, which are stored in the InitialBundle asset bundle.
    if (InitialState == true && AssetIndex < 7)
    {
        LoadBundle(InitialBundle);
        AssetIndex = AssetIndex + 1;
        if (AssetIndex == 7)
        {
            //The innitial asset bundle is loaded and the assets/models have been loaded onto the scene,
            //therefore we move to the final state and
            //load the remaining asset bundles and their assets
            InitialState = false;
            FinalState = true;
            //We unlaod the asset bundles, remove it from the memory once we are done using it,
            //and destroy the assets that are associated with it from the scene.
            InitialBundle.Unload(true);
        }
    }
```

Figure 72 : Initial state - loading the initial 7 assets from the bundle

Once all the assets in the initial bundle have been loaded and destroyed we move to the final state and unload the initial bundle which removes the asset bundle from memory.

The function that loads the asset is the LoadBundle(AssetBundle Bundle) function, this function loads the asset and destroys the previous asset, thereby removing that asset from the scene.



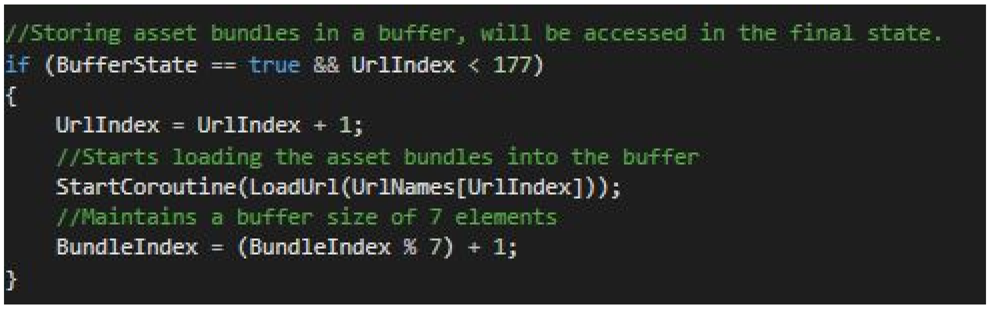
```
//Loading the assets from the asset bundles, the loaded asset is destroyed just before the next asset is loaded.
2 references
void LoadBundle(AssetBundle Bundle)
{

    //Load an asset from the loaded bundle
    AssetBundleRequest bundleRequest = Bundle.LoadAssetAsync(ModelNames[AssetIndex], typeof(GameObject));
    //var bundleRequest = Bundle.LoadAssetAsync<GameObject>(ModelNames[AssetIndex]);
    //get object
    GameObject obj = bundleRequest.asset as GameObject;

    //We destroy the previously instantiated Game Object to achieve the video/S=simulation efect.
    Destroy(GameObjectBundle);
    //Instantiate the loaded Game Object/model, assign it to the ParentGameObjectBundle, displayed on the scene,
    GameObjectBundle = Instantiate(obj, ParentGameObjectBundle.transform) as GameObject;
}
```

Figure 73 : LoadBundle(AssetBundle Bundle) function

We load the asset by passing the asset name, then we destroy the previously loaded asset before loading the next asset to achieve the video/simulation effect. Initially the GameObjectBundle won't have an asset from the asset bundle, so if we use try to use the Destroy() function on a null object we get an error, to overcome this we assign the Unused gameobject to GameObjectBundle initially and that Unused gameobject is destroyed just before the first asset is loaded.

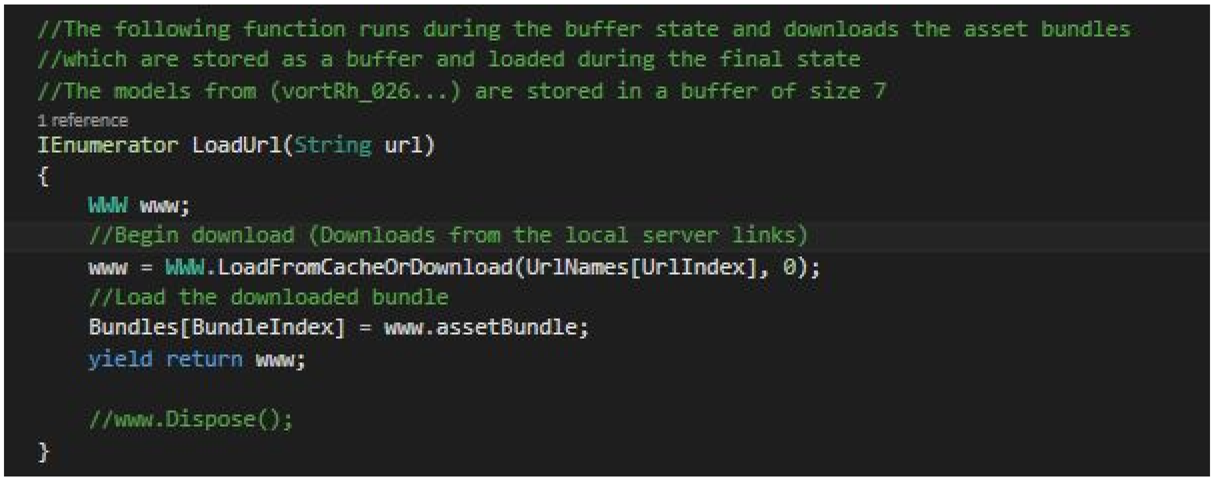
Buffer State

Storing asset bundles in the Bundles array, the size of the buffer/array is 7.



```
//Storing asset bundles in a buffer, will be accessed in the final state.
if (BufferState == true && UrlIndex < 177)
{
    UrlIndex = UrlIndex + 1;
    //Starts loading the asset bundles into the buffer
    StartCoroutine(LoadUrl(UrlNames[UrlIndex]));
    //Maintains a buffer size of 7 elements
    BundleIndex = (BundleIndex % 7) + 1;
}
```

Figure 74 : Buffer State



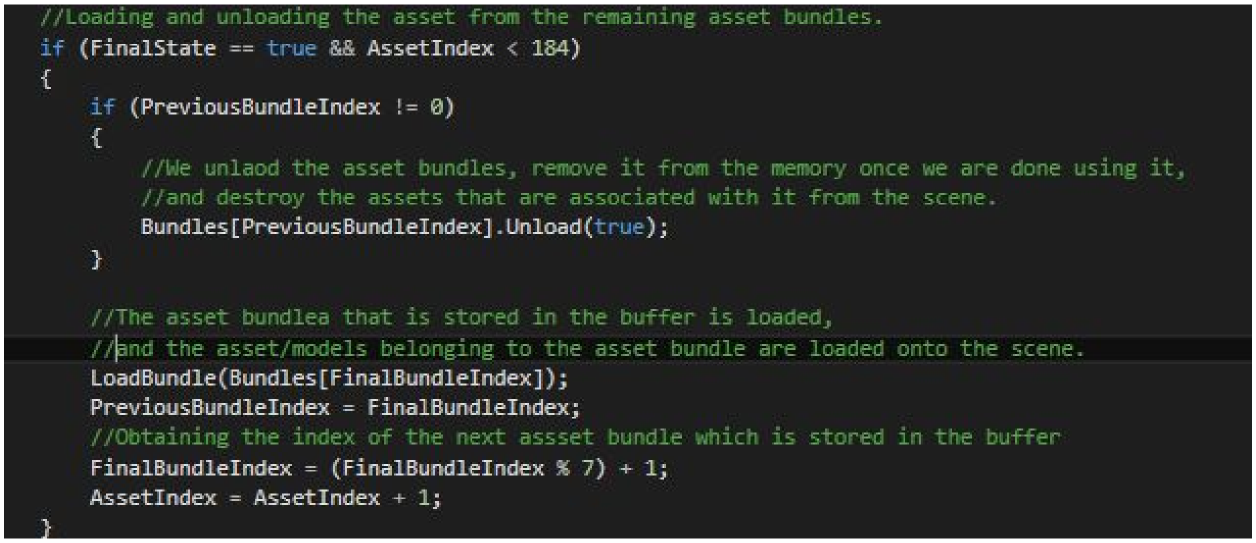
```
//The following function runs during the buffer state and downloads the asset bundles
//which are stored as a buffer and loaded during the final state
//The models from (vortRh_026...) are stored in a buffer of size 7
1 reference
IEnumerator LoadUrl(String url)
{
    WWW www;
    //Begin download (Downloads from the local server links)
    www = WWW.LoadFromCacheOrDownload(UrlNames[UrlIndex], 0);
    //Load the downloaded bundle
    Bundles[BundleIndex] = www.assetBundle;
    yield return www;

    //www.Dispose();
}
```

Figure 75 : LoadUrl() function - Storing asset bundles in the buffer

So the asset bundles are stored in the Bundles array which acts like a buffer, every time an asset is loaded onto the scene an asset bundle is loaded into the buffer. The assets from these asset bundles will be loaded onto the scene in the final state. This process continues until there are no asset bundles left to be loaded into the memory.

The asset bundles in the buffer state and the final state contain only one asset unlike the initial bundle which contained 7.

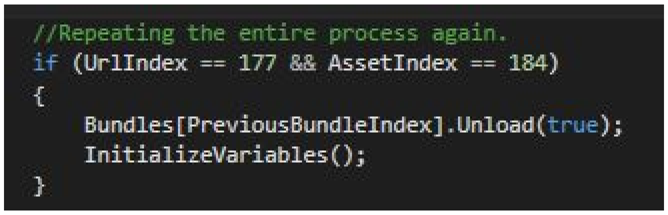
Final State



```
//Loading and unloading the asset from the remaining asset bundles.
if (FinalState == true && AssetIndex < 184)
{
    if (PreviousBundleIndex != 0)
    {
        //We unlaod the asset bundles, remove it from the memory once we are done using it,
        //and destroy the assets that are associated with it from the scene.
        Bundles[PreviousBundleIndex].Unload(true);
    }

    //The asset bundlea that is stored in the buffer is loaded,
    //and the asset/models belonging to the asset bundle are loaded onto the scene.
    LoadBundle(Bundles[FinalBundleIndex]);
    PreviousBundleIndex = FinalBundleIndex;
    //Obtaining the index of the next assset bundle which is stored in the buffer
    FinalBundleIndex = (FinalBundleIndex % 7) + 1;
    AssetIndex = AssetIndex + 1;
}
```

Figure 76 : Final State

We unload the previous asset bundle to free up memory and load the next asset onto the scene from the asset bundle. Finally to continue the simulation we need to keep repeating the entire process, this is done by unloading the final asset bundle and then reinitializing all the variables.

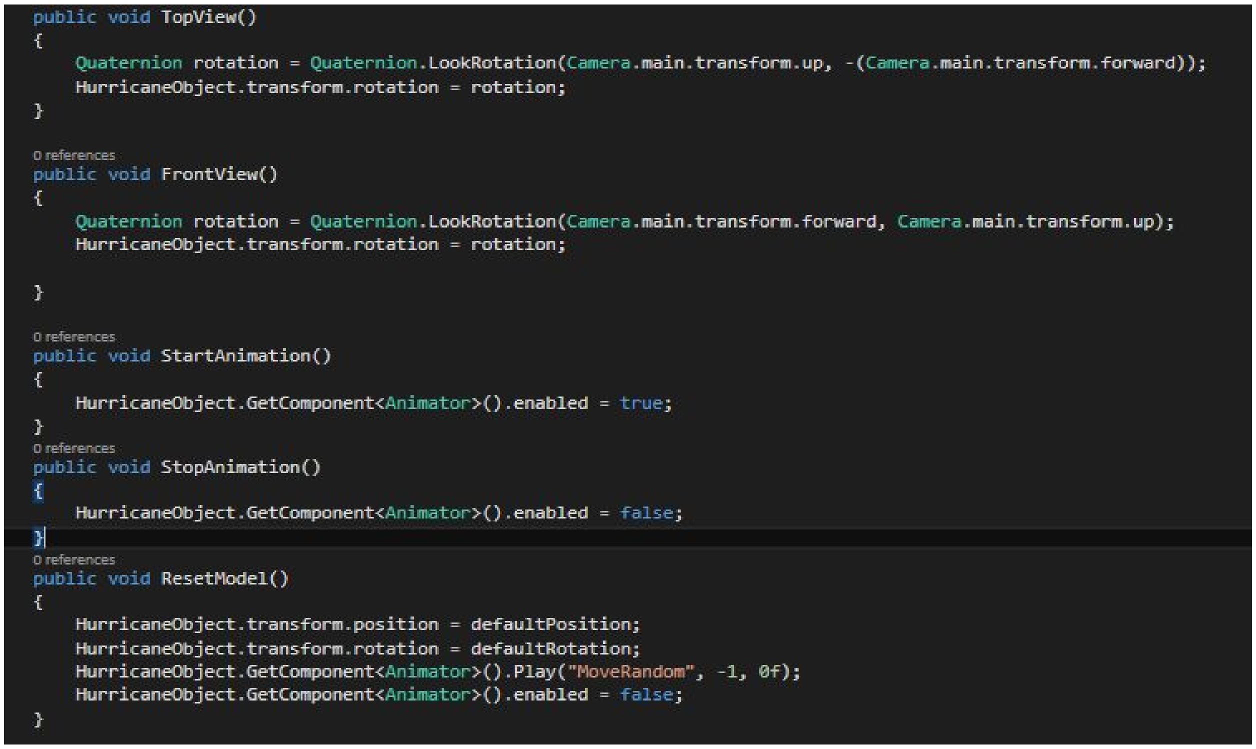


```
//Repeating the entire process again.
if (UrlIndex == 177 && AssetIndex == 184)
{
    Bundles[PreviousBundleIndex].Unload(true);
    InitializeVariables();
}
```

Figure 78 : Repeat the process

HurricaneCapability

This script is responsible for changing the rotation, starting and stopping the animation component. Since the hurricane models are children of the Hurricane gameobject we can manipulate just the Hurricane gameobject and these manipulations will be reflected on the children (the hurricane models).



```
public void TopView()
{
    Quaternion rotation = Quaternion.LookRotation(Camera.main.transform.up, -(Camera.main.transform.forward));
    HurricaneObject.transform.rotation = rotation;
}

0 references
public void FrontView()
{
    Quaternion rotation = Quaternion.LookRotation(Camera.main.transform.forward, Camera.main.transform.up);
    HurricaneObject.transform.rotation = rotation;

}

0 references
public void StartAnimation()
{
    HurricaneObject.GetComponent<Animator>().enabled = true;
}
0 references
public void StopAnimation()
{
    HurricaneObject.GetComponent<Animator>().enabled = false;
}
0 references
public void ResetModel()
{
    HurricaneObject.transform.position = defaultPosition;
    HurricaneObject.transform.rotation = defaultRotation;
    HurricaneObject.GetComponent<Animator>().Play("MoveRandom", -1, 0f);
    HurricaneObject.GetComponent<Animator>().enabled = false;
}
```

Figure 79 : Hurricane capabilities

The Hurricane gameobject is assigned to HurricaneObject. These functions are called using the Keyword Manager (Keyword recognizer) present in the Managers gameobject.

Animation

Creating the animation : Create the animation and save it, the name of our created animation is "MoveRandom".

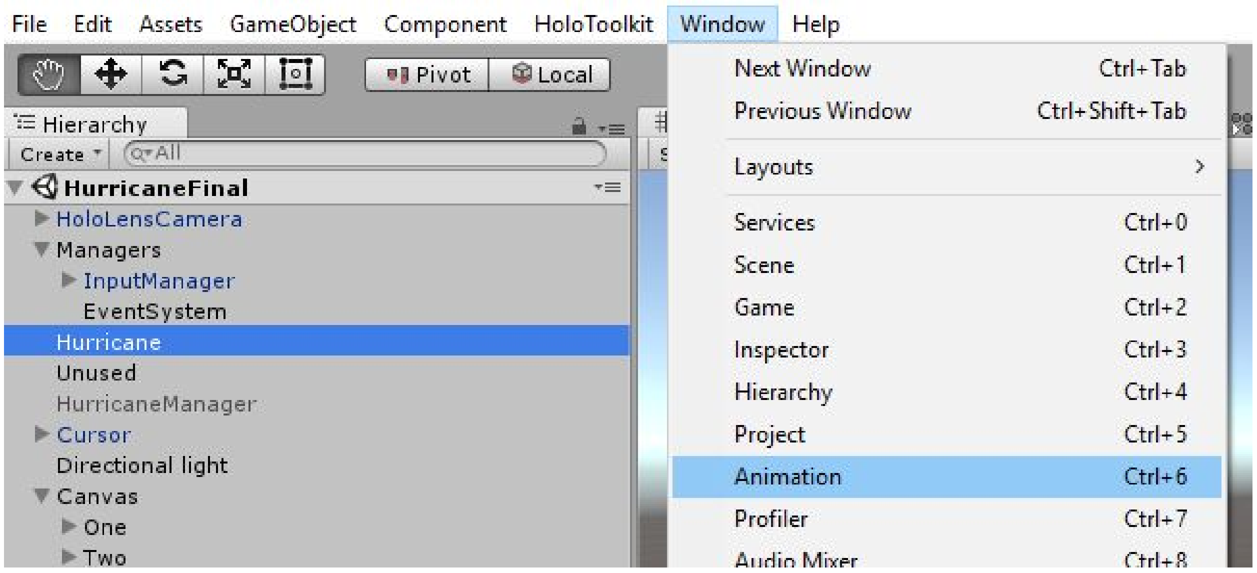


Figure 80 : Creating the animation

Select the Hurricane object and select animation to start creating the animation. A window like this opens up.

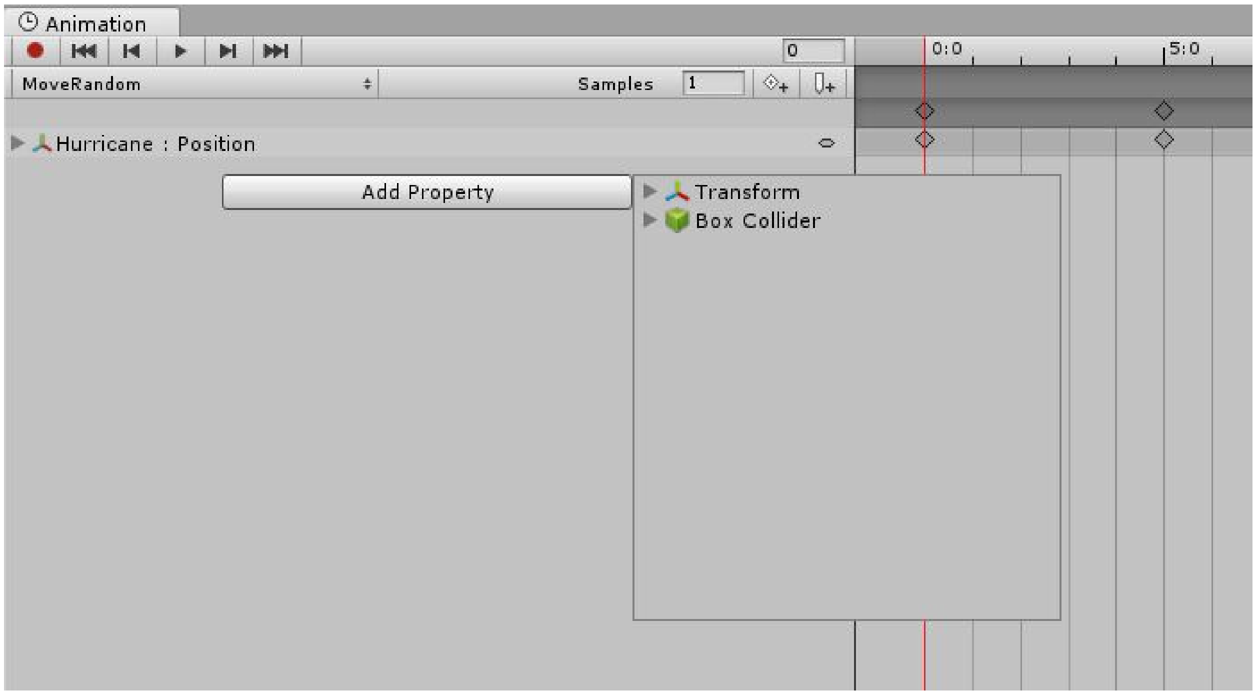
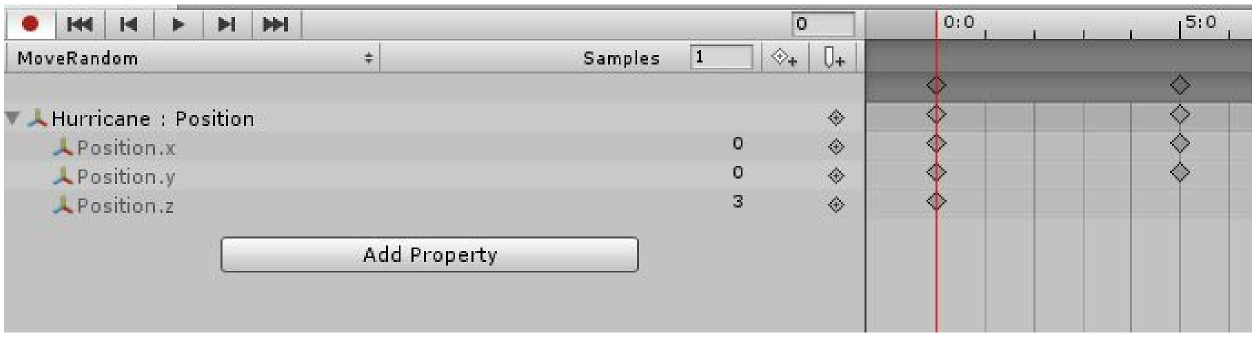


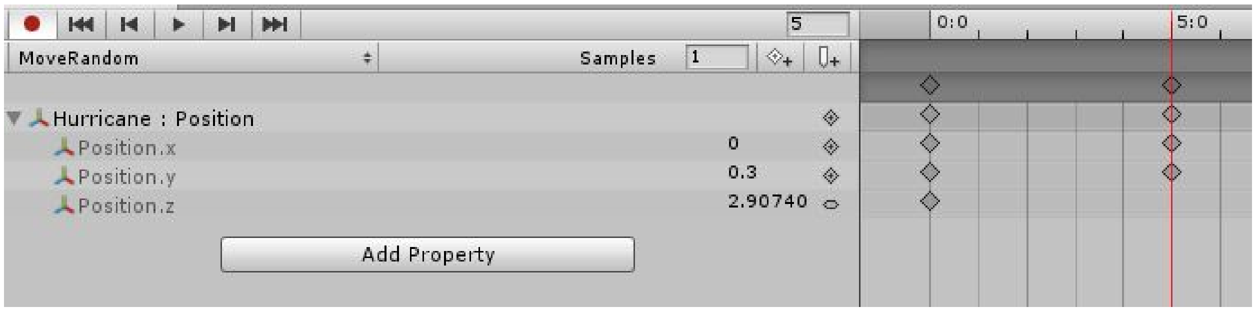
Figure 81 : Animation window, animation properties

In our animation we change the position of the Hurricane object at different time intervals.



Position.x	0
Position.y	0
Position.z	3

Figure 82 : Position at time 0:0



Hurricane : Position

Property	Value
Position.x	0
Position.y	0.3
Position.z	2.90740

Figure 83 : Position at time 5:0

As you can see the position of the hurricane object changes every 5 seconds.

Now we can build and run our application on the hololens. The first time you run the application the hurricane models will load very slowly onto the scene as the asset bundles are being downloaded from the local server. The next time you load the asset bundles and the assets or the next time you run the application the hurricane models will be loaded much faster onto the scene as the asset bundles are being loaded from the cache instead of being downloaded from the local server. (www WWW.LoadFromCacheOrDownload(url, 0);) load from cache or download, if the asset bundle is not present in the cache it will be downloaded.

****

Hurricane models/Assets, Asset Bundles, and a Local Server, were used to build the Hurricane Simulation application for the HoloLens. The interactivity within the application was achieved by using Voice Commands and Animations.

****

REFERENCES

https://unity3d.com/learn/tutorials/topics/scripting/assetbundles-and-assetbundle-manager

https://unity3d.com/learn/tutorials/topics/best-practices/assetbundle-fundamentals

http://www.theappguruz.com/blog/create-and-download-asset-bundle-in-unity

http://www.yudiz.com/getting-started-with-asset-bundles-in-unity/

https://docs.unity3d.com/550/Documentation/Manual/DownloadingAssetBundles.html

https://docs.unity3d.com/550/Documentation/ScriptReference/WWW.LoadFromCacheOrDownload.html

****

THANK YOU