Immersive Unity SDK (iUS)

Overview

The script and prefabs in this directory provide the support for Immersive Spaces from a Unity3D project. Provided are a set of Camera Prefabs which can be place in you scene and will automatically provide support for Immersive Spaces of different shapes, sizes and layouts as well as an advanced Hotspot system (as detailed in the file Hotspots README) which allows for the creation of complex experiences without any programming.

These camera prefabs support "Virtual Room Mode". This allows content to be tested in simulated rooms with different sizes and and layouts.

Getting Started

Once you have imported the UnityPackage Immersive Unity SDK you must manually import TextMesh Pro by going to Packages > TextMesh Pro > Package Resources and double click on TMP Essential Resources.

It is also recommended to change the Player settings. Go to Edit > Project Settings > Player. In the top right corner choose slider button next to the gear and select the Player setting preset provided. This will change these settings to be optimal for an Immersive Experience. You should leave the Product name as Immersive Application.

When creating new scenes, there are shortcuts which will automatically instantiate scenes with an Immersive Camera System automatically present. To access these open Immersive Interactive from the menu bar. Select New 2D Scene for a 2D cameras system or New 3D Scene for a 3D Camera system.

Run Shortcuts

When testing within the Unity Editor you can use Editor Only Settings to determine how you wish to test software. When running software from the Immersive Player it will automatically tell the program about the room layout and surface sizes and will be displayed correctly. If however you wish to test your software in an actual Immersive Space without running it from the Immersive Player, a series of batch files which act as run shortcuts are provided which will run the experience targetting different layouts and surface sizes.

These can be accessed from your build folder inside the folder entitled Run Shortcuts.

Note: You should leave the Product Name as Immersive Application.

Immersive Camera Systems

Immersive camera systems are prefabs which will contain all the components required to display images in an Immersive Space. There are different Immersive Camera Systems which allow you to create different types of content. The two provided camera systems are 2D Immersive Camera and 3D Immersive Camera. In addition to the Cameras in a system, there are also a series of Unity canvases which allow you to have objects and buttons which are repositioned based on the object size.

Virtual Room

In order to make it easy to test Immersive Experience, you can test in an Virtual Immersive Room. You can move around this room using the WASD buttons and you can rotate by with the mouse while holding down the right mouse button. Using the mouse you can simulate touches on the walls or floor.

Layouts and Surface Sizes

There are a variety of different Immersive Space layouts and sizes. The size and aspect ratio of individual surfaces can vary. Common surface apsect ratios are 16x9, 16x10 and 4x3. In Standard spaces all surfaces are the same width. In Wide spaces there are double wide surfaces on each wall. Wide Front spaces have a double wide center wall and normal side walls.

Additionally, not all spaces have all surfaces. Common layouts are: + Powerwall - Center Surface Only + Story Corner - Left and Center Surfaces + Standard - Left, Center and Right Surfaces + 4 Wall - Left, Center, Right and Back Surfaces

Any of these variations can be combined with a floor.

Canvases

Each Immersive Camera System contains a canvas for each surface. You can use these canvases to place objects or buttons which will scale their position and/or size relative to the surfaces size and shape. The following tutorials provide a good overview as to how to use this: https://unity3d.com/learn/tutorials/s/user-interface-ui

These canvases are located at Immersive Camera > UI > Canvas in the hierarchy of the camera prefab and are located to the right in the scene view. There is a canvas for the left, center, right, back and floor canvases. The center canvas will always be visible in all layout options however the left, right, back and floor canvases will be deleted if their corresponding surfaces do not exist.

You can add UI elements to each of these objects, and, place and scale them using the Rect Transform tool as demonstrated in the tutorial above. The canvases will scale horizontally but not vertically.

The example scene Layout Sample demonstrates how this can be used.

Generic Camera System Settings

Editor Only Settings are settings which control how your experience is displayed when played in the Unity Editor: + Virtual Room Mode - If true, then when playing, scene will open in a Virtual Room. If not true, then the scene will be displayed as a single spanning image. + Use Virtual Room In Build - If true then when project is built it will run in a Virtual Room. If false then Virtual Room will only be used in Editor. + Screen Size - A choice of preset surface sizes to use while in Edit + Active Walls - Select which walls should be active for testing.

Interaction Settings control the type of interactions available to the user: + Interaction On - Whether the scene will register any interaction at all. + Point Touches On - Whether small touches should be registered as discrete touches for interacting with objects and buttons. + Display Touch Points - Whether an icon should be displayed where the user is touching. + Area Touches On - Whether large touches, such as a full arm should be

registered as area touches which enable interactions such as catching falling snow flakes on the users arm. + Display Area Touches - Whether to display an outline of where area touches are registered.

2D Immersive Camera

The 2D Immersive Camera uses orthographic cameras to create a continuous spanning image.

- **Stage** All objects that you wish to appear on the walls (exluding canvas objects) should be placed in the Empty Gameobject called "Stage". When you create a new Immersive Camera the Stage object will be automatically added to the scene. The Stage will control scaling of objects when the Aspect Ratio does not equal the Target Aspect Ratio. The Stage also provides shortcut buttons to create common prefabs including a Hotspot Controller.
- **Floor** In addition to the Stage there is also a Floor object in which you should place all you objects you wish to appear on the floor.
- **Floor Camera** The floor camera in the 2D Immersive Camera points directly down. The means the floor content is perpendicular to the wall contents so it will never interact.

3D Immersive Camera

The 3D Immersive Camera uses perspective cameras point at 90 degree rotations in order to display 3D Unity scenes in an Immersive Space. + **Scaling Mode** There are two scaling modes which handle spaces with wide surfaces in different ways. + Fixed Horizontal FOV - Each camera has a 90 degree horizontal field of view. This has the advantage that each surface displays content in the same way however this also means that a space with a wide surface, the vertical field of view is very small. + Fixed Vertical FOV - This mode prioritises the center surface. The vertical field of view of the center screen is the same as that of 16x9 surface with 90 degree horizontal field of view. This means that horizontal field of view of the center surface is greater than 90 degrees. The horizontal field of view of the side walls is 90 degrees. The frustrums of these cameras are assymetrical and are shifted so they align with the center camera. The advantage of the technique is that the center surface has a high vertical field of view, however this is at the expense of the field of view of the side walls.

Custom Immersive Camera

Both the 2D and 3D Immersive Cameras inherit from the class AbstractImmersiveCamera. This provides a lot of the base functionality for supporting Immersive Rooms including setting up the correct canvases and cameras for the current space. It also provides Virtual Immersive Room functionality which is a useful feature when testing. This also defines several abstract methods which subclasses must implement for the specific camera type you want. These are:

- PositionCamera() - This is where the position and rotation of each camera should be set.
- RotateCameraLeft() - This should rotate the camera, one width of the center surface, to the left.
- RotateCameraRight() - This should rotate the camera, one width of the center surface, to the right.

The physical properties of the Cameras (eg. orthographic/perspective) should be set manually by change the

referenced object "Main Camera".

Interaction

There are several ways add touch interactions to objects in your scene.

- **Interactable Object** The easiest way to make an object interactive is to attach a script to the object which inherits from the abstract class InteractableObject. Four methods must be implemented, OnRelease(), OnPress(), OnTouchEnter() and OnTouchExit().
    - OnPress is called the frame the user touches an object.
    - OnRelease is called the frame the after the user has released their touch from an object.
    - OnTouchEnter is called when the user is touching an object.
    - OnTouchExit is called when the user is still touching the screen but are no longer touching the object.

This is the underlying functionality that the Hotspot system uses.

- **Touch Listeners** Every time a surface is touched, an event is raised. You can subscribe to these events from any script in the project.

  Whenever any surface is touched the event AnyWallTouched is raised. Additionally, a event specific to the individual surface is also raised. These are called LeftScreenTouched, CenterScreenTouched, RightScreenTouched, BackScreenTouched. Each event has information about the touch associated with it:

    - A Vector2 representing the pixel position of the touch.
    - The camera associated with the surface the touch is on.
    - The phase of the touch, eg Began or Ended.
    - The index of the touch.

  In order to listen to one of these events you need a method which takes all of the arguments described above. You at a method, New_Method, to listen for an event by calling AbstractImmersiveCamera.WallTouchedEvent.AddListener().

Weather Effects

Weather effects can be added to the screen, for example rain or frost. These are based on the Asset Store toolkit *RainDropEffect2*. In order to add a weather effect to scene, add the Weather Effect component to the camera in that scene. The component allows you to provide an Effect Prefab. A library of these can be found in the folder Immersive Camera > Weather Effects. Further effects can be constructed using the *RainDropEffect2* toolkit. You may also provide reference to a game object with an AudioSource attached. If this is the case the audio source will be turned on and off when the weather effect is turned on or off. Finally there is a public variable EffectOnOff which can be used to enable of disable the effect.