INFO 5340 / CS 5650
Virtual and Augmented Reality

Assignment 4

Objective

The objective of this assignment is to design and develop an interactive 3D environment for Virtual Reality. The environment is based on the MakerLAB, with some custom 'stations' integrated. You will start by capturing photos of the lab and applying texture maps to the 3D model. You will be responsible for enabling locomotion so you can travel between stations, then update each station until they fulfill the required functionality. Each station requires a different skill set including: particle systems, sound effects, API requests, and more. Once you're done, it's time to create your own super station!

You will create an interaction on your desktop device as shown in the solution screen recording, ***watch with headphones to hear instructions***. **Note: the desktop audio did not record from Unity, but you can use your imagination (it is explained when you should hear audio).** You can record your gameplay using OBS, which is already installed on the PCs in the Masters' Studio.

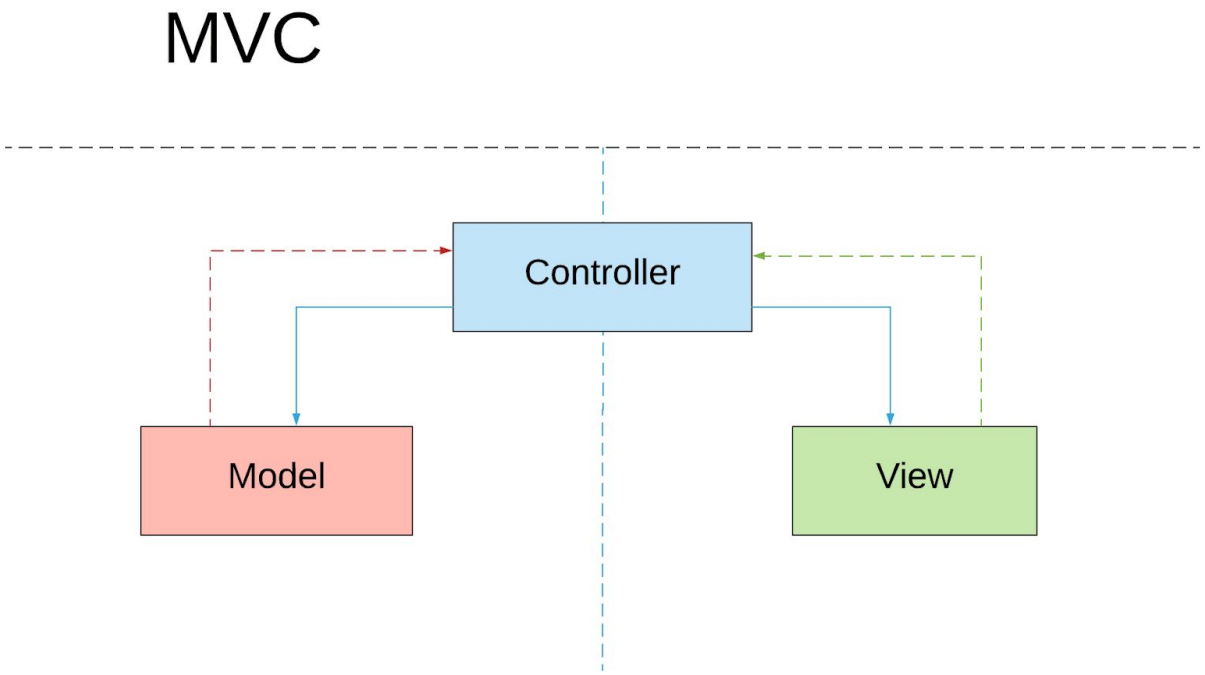
Assignment Task

In this part you will use Unity to build an interactive VR application. Follow the guide below to complete all the tasks. **Tasks are in bold and highlighted in blue.**

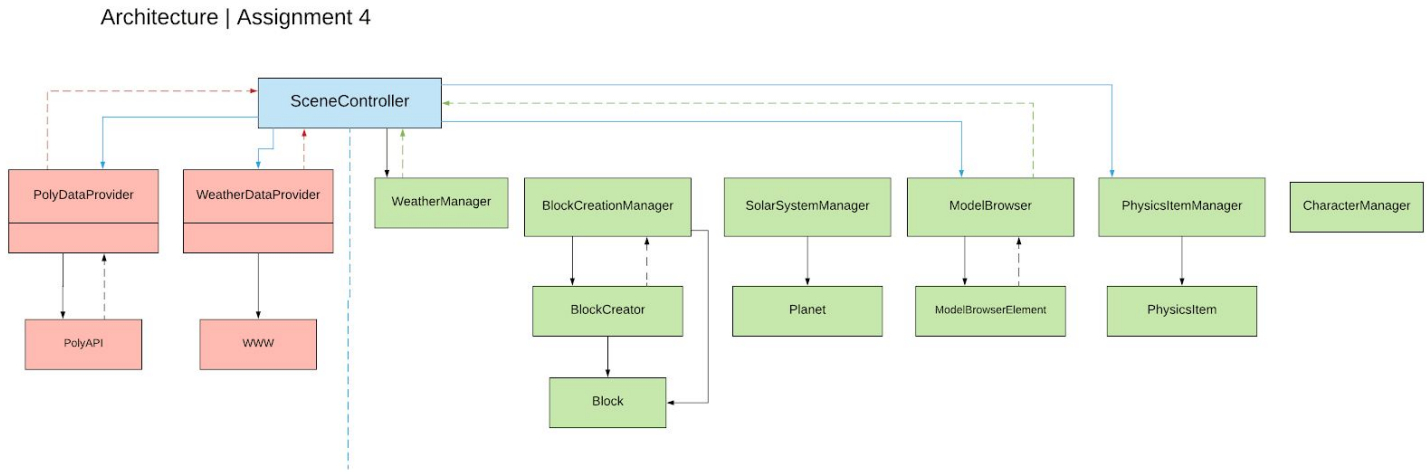
1 | Project Setup and Overview

Log into the computer using your NetID. Clone assignment-4 onto the VR computers in the Masters' Studio and open the Unity Project. Check the console for any errors relating to unsafe code. *If you see an error relating to unsafe code open the project's Player Settings and uncheck then recheck the 'Allow unsafe code' checkbox located under 'Other Settings'*.

Open the MainScene (Assets/Scenes/MainScene), you will only be using this scene for the assignment. The scripts in this scene are set up to follow MVC (Model-view-controller) architecture.



Below is a high-level architectural diagram of the scene's classes:



Solid arrows represent a reference, either by SerializedField or GetComponent(s).

Dotted arrows represent information being 'bubbled up' via Actions/Delegates (without a direct reference). You can read more about actions here.

This information is provided to help you better understand the relationship between classes, you will not need to modify the scene architecture in any way.

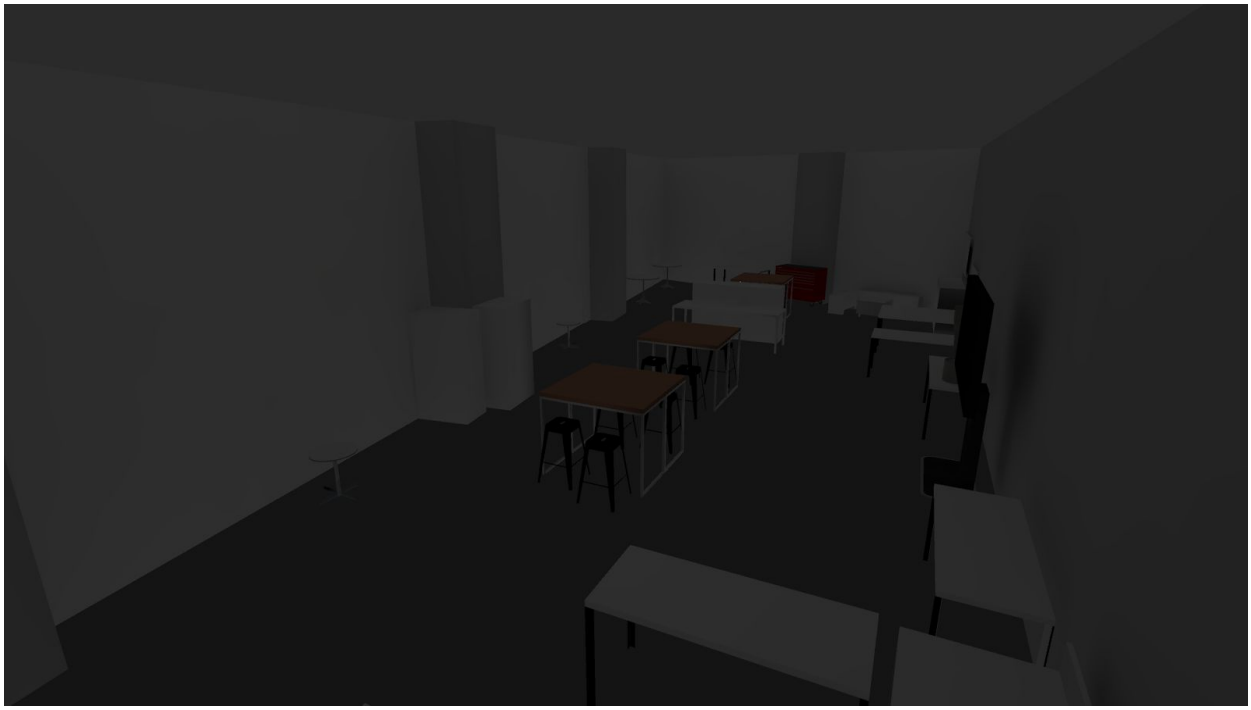
Tip: Reduce the size of the massive 3D gizmos in the scene by click the Gizmos dropdown on the top of the scene window and dragging down the 3D icons slider.

Ensure you have Steam running, sign into your account if you have not done so already. Open SteamVR by clicking the [VR] icon in the top right corner of Steam, or finding SteamVR in your library under 'Tools'.

If SteamVR displays any errors relating to the compositor, restart your computer.

2 | Lighting The Scene

Below is a before and after comparison of the scene with/without **lighting** and **textures**.



Your job in this section is to get your scene to look as close as possible to the second picture. Below are the tasks:

[2.1] Capture and apply textures

Visit the MakerLAB on the second floor of Tata Innovation Center. If you don't have keycard access, find a student who does and plan a visit together.

Use your camera of choice to capture the following four photos:

1) Floor
2) Ceiling
3) West window
4) South window

Try to avoid any undesired objects in the photo (shoes/chairs/walls). If necessary, crop the image in the software of your choice.

Import the photos into the /Assets/Textures/ folder in your Unity project. **Edit the import settings of all the textures so that the "Wrap Mode" is set to "Mirror".** This will create a seamless texture when tiling. Apply the textures to the appropriate materials in the /Assets/Materials/MakerLab/ folder. The texture mapped materials already have lighting 'baked' into the image, so you can apply the texture to the Emission channel rather than the Albedo channel. Adjust the color values of the Albedo/Emission channels to get the desired look. Edit the tiling values to avoid undesired stretching/squishing.

[2.2] Create, position, and bake lights

Create and position lights around the scene to create the desired effect. Baking is set to auto-generate, and you can follow bake progress in the bottom right corner of the Unity Editor.

For more information on lighting read the Unity manual here:
https://docs.unity3d.com/Manual/Lighting.html

Note: Lights will need to have Mode set to "Mixed" or "Baked" to be considered for baking.

[2.3] Create and position a reflection probe

Create and position a reflection probe in the desired position to ensure that reflective surfaces have a captured reflection to render. Ensure the reflection probe's size encompasses the entire scene.

For more information on reflection probes read the Unity manual here:
https://docs.unity3d.com/Manual/class-ReflectionProbe.html

Find the CharacterManager GameObject in the scene and set it to active. Hit play. You'll see a lot of errors relating to the NavMesh.

[3.1] Follow the instructions here to bake a NavMesh.

Hit Play. You should no longer see errors in the console, but your character will only hover in one spot.

[3.2] Duplicate the "DestinationPoint" GameObject in the scene until you have seven total destination points. Move them around to create a series of destinations for your character to traverse (these must be within the NavMesh). Assign references to the destination points in the CharacterManager via the Serialized array.

Hit play. Your character should now traverse the points.

[3.3] Fill in the WaveAtMainCamera and StartWalking functions inside the CharacterManager script so that they implement the functionality described in the function comments. You may find the following documentation useful:

https://docs.unity3d.com/Manual/Animator.html
https://docs.unity3d.com/Manual/class-AnimatorController.html
https://docs.unity3d.com/Manual/class-Animator.html
https://docs.unity3d.com/ScriptReference/Animator.html

The character should now stop at each point, look at the camera in the scene (you), wave, and resume his navigation.

4 | Teleportation

Find the "Teleport" GameObject in the scene and set it to active.

[4.1] Apply a collider to the TeleportArea GameObject so that it encompasses the entire environment.

You should now be able to teleport around the scene by clicking the trackpad.

Note: If the MainCamera GameObject is active, you will have issues teleporting.

5 | Solar System Station`

Find the SolarSystemManager GameObject in the scene and set it to active.

[5.1] Assign the appropriate value to the Planet Type SerializedField on each Planet GameObject.

Open the SolarSystemManager script.

[5.2] Fill in the Awake function so that it implements the functionality described in the function comments. Not sure how to get references to the Planets for the array? Take a look at this documentation.

[5.3] Fill in the OnResetButtonClicked function so that it implements the functionality described in the function comments.

Open the Planet script.

[5.4] Fill in the Reset function so that it implements the functionality described in the function comments.

[5.5] Fill in the UpdateOrbit function so that it implements the functionality described in the function comments. Don't forget to use the helper functions located in the Planet script! You may find the following documentation useful:

https://docs.unity3d.com/ScriptReference/Transform.RotateAround.html

[5.6] Fill in the UpdateRotation function so that it implements the functionality described in the function comments. Don't forget to use the helper functions located in the Planet script!

The solar system should now behave the same as it does in the solution video.

6 | Block Creation Station

Find the BlockCreationManager GameObject in the scene and set it to active.

Open the BlockCreator script.

[6.1] Modify the update function according to the 'TODO' comment.

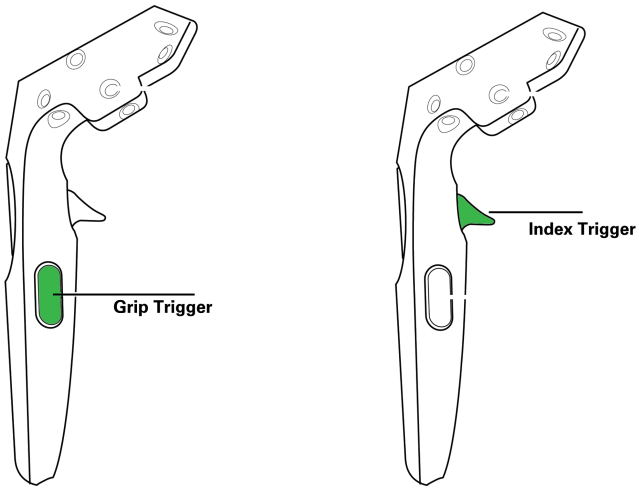
[6.2] Fill in the CheckPosition function so that it implements the functionality described in the function comments.

[6.3] Fill in the CreateBlock function so that it implements the functionality described in the function comments. Having trouble figuring out the syntax to fire an Action? Take a look at

the ModelBrowserElement script for an example. Having trouble figuring out the syntax for playing an audio clip at a certain position? Check out this documentation.

Ensure your Steam controller binding has "CreateBlock" set to receive input from the controller trigger.

You should now be able to create blocks by grabbing the block creators (glue guns) with the **grip trigger**, and squeezing the **index trigger**.



Open the BlockCreationManager script.

[6.4] Fill in the OnDropBlocksButtonPressed function so that it implements the functionality described in the function comments.

When the drop blocks button is pressed it should now cause all floating blocks to drop onto the table/floor.

[6.5] Create two new BlockCreator GameObjects with *different* colors. Position them so they don't overlap with the existing BlockCreators.

[6.6] Create a delete blocks button, implement the necessary code in BlockCreationManager for a reference to be assigned and assign the reference in the scene.

Open the BlockCreationManager script.

[6.7] Fill in the OnDeleteBlocksButtonPressed function so that it implements the functionality described in the function comments.

The block creation station should now behave the same as it does in the solution video.

7 | Weather Station

Find the WeatherManager GameObject in the scene and set it to active.

Open the WeatherManager script.

[7.1] Fill in the SetData function so that it implements the functionality described in the function comments.

[7.2] Fill in the OnCityButtonSelected function so that it implements the functionality described in the function comments.

Clicking the city buttons should now update the weather text.

Find the WeatherConditions GameObject in the scene. You'll see two GameObjects are already created as children, which contain relevant meshes and particle systems.

[7.3] Create new GameObjects as children of the "WeatherConditions" GameObject, one for each of the remaining WeatherConditions (all enum values located in SceneController). In addition to adding/adjusting the particle systems, you will need to include looping audio sources for the Drizzle, Rain, and Thunderstorm conditions. Textures and materials can be found inside the Assets/Textures/Weather/ and Assets/Materials/Weather/ folders. Audio can be found in Assets/Audio/. You may find the following documentation useful:

https://docs.unity3d.com/Manual/PartSysWhatIs.html
https://docs.unity3d.com/Manual/PartSysUsage.html

Here is an example of what your complete scene hierarchy should look like:

[7.4] Assign the necessary references to your new weather conditions GameObjects in the WeatherManager component.

You should now be able to toggle between different weather conditions using the Conditions menu.

[7.5] Add a city button for your hometown (if you were born/raised in one of the included cities please choose a different city). You will need to find the city's ID in the OpenWeatherMap API. You can do so by downloading the file from here (city.list.json.gz). This will require modifying the SceneController script. Assign any necessary references to the new button.

The weather station should now behave the same as it does in the solution video, but with an additional city button.

8 | Super Mode Station

This is your chance to show the world your *Super* Unity skills!

- **Create a new station where you can enable "Super Mode" for your entire scene. It is up to you to decide what that entails!**
  When Super Mode is enabled, you could for example:

- Change the laws of physics (time, gravity, scale)
- Change the viewpoint (from 1st person to 3rd person)
- Alter the locomotion method
- Control the AI character in new ways
- Make every object in the scene interactable
- Apply post-processing effects
- Use audio effects and music
- And many many more!

Note: **This is a required task (not a bonus task)** with an added weight, and will be graded relative to the best student implementation, based on creativity, complexity, and elegance of execution. Make sure you explain and display in detail your Super Mode implementation.

9 | Bonus Task

When any object falls on the floor, have the character walk over to it and pick it up, using the pick up animation.

Good luck!