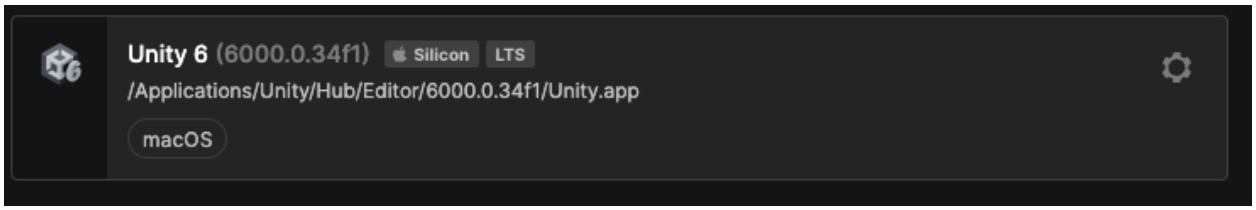
OWL SUITS Work Guide

Development

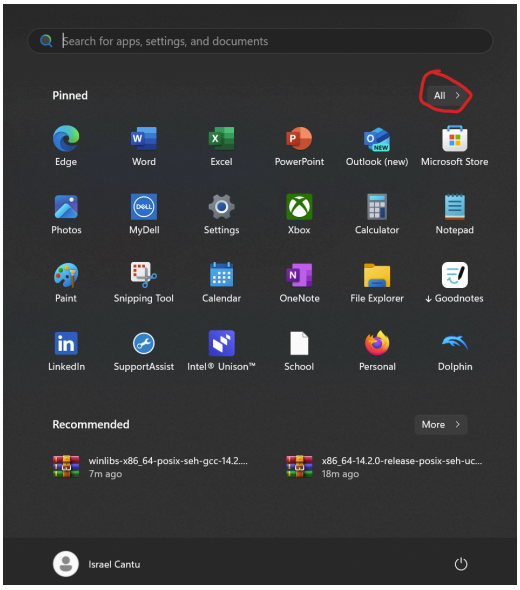
Installation

We will standardize this version of the editor across the board. **Make sure to check "Universal Windows Platform"** when installing (if you are on Windows).

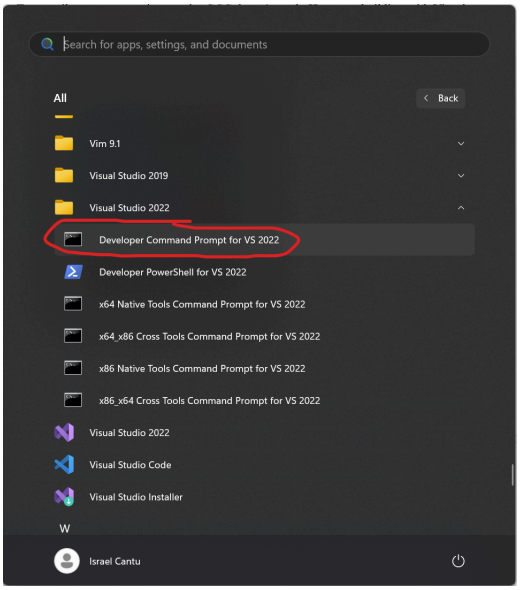


Links

Current Project: https://github.com/OWL-SUITS-2025
Last Year's Project: https://github.com/NASA-SUITS-Daedalus/Daedalus_SUITS_2024
TSS-2025: https://github.com/OWL-SUITS-2025/TSS-2025

- To compile server.exe, using regular GCC doesn't work. However, building with Visual Studio Dev Tools works.
- Make sure you have **Visual Studio 2022** installed on your computer.
- Open the **Visual Studio 2022 Developer Command Prompt**
    - Go to Windows Start Menu>Applications/Folders (see below for button)
    - 

    - Scroll down to Visual Studio 2022 folder (NOT APP)
    - Open **Developer Command Prompt for VS 2022**

- 

- Once the terminal is open, cd to the root of the TSS folder
    - once in the root, run this command to compile the C code to server.exe
        - `cl -g network.c server_data.c server.c cJSON.c -o server.exe -lm`
    - The server.exe should now be in the folder. You can now run the TSS webserver!

GCC Installation:
https://phoenixnap.com/kb/install-gcc-windows
https://www.youtube.com/watch?v=k6juv3mIr9o

Board: Dev

This board will be used for code development. Each section below is scheduled to be **2 weeks** for a total of 12 weeks, completion on April 3, 2025 (date of the **Software Design Review**)

nav = Navigation
tel = Telemetry
map = Mapping
sci = Scientific Data
int = Interoperability
sys = System Core
caw = Caution & Warning
pro = Procedure

Core System Setup:

[sys] Setup basic project structure and TSS integration
[sys] Implement basic UI framework and routing
[sys] Create data store architecture for state management
[sys] Setup basic communication protocols between PR and EV

Navigation & Mapping Foundation:

[nav] Implement basic path planning algorithm
[nav] Create breadcrumb tracking system
[nav] Setup waypoint management system
[map] Create 2D map rendering engine
[map] Implement asset location tracking

Telemetry & Data Handling:

[tel] Setup real-time telemetry data processing
[tel] Create telemetry data visualization components
[tel] Implement biomedical data monitoring system
[tel] Build resource usage tracking system

Scientific & Procedure Systems:

[sci] Build XRF data processing pipeline
[sci] Create sample data management system
[sci] Implement geological data analysis tools
[pro] Create procedure execution engine
[pro] Build step-by-step task tracking system

Safety & Interoperability:

[caw] Implement warning detection system
[caw] Create alert notification system
[caw] Build predictive warning analytics
[int] Develop PR-EV data sync system
[int] Create shared resource management

Advanced Features:

[nav] Implement hazard avoidance algorithm
[nav] Create autonomous navigation system
[map] Add dynamic path recalculation
[sci] Build advanced sample analysis tools
[int] Create real-time video streaming system

Board: UI/UX

This board will be used for UI/UX design development. Each section below is scheduled to be **2 weeks** for a total of 12 weeks, completion on April 3, 2025 (date of the **Software Design Review**)

des = Design System
wir = Wireframe
pro = Prototype
com = Component (3D)
ast = Asset (2D)
usr = User Research

Core System Foundation:

[des] Create design system documentation and guidelines
[des] Define color palette and typography system
[des] Design basic component library structure
[wir] Create basic layout wireframes
[usr] Conduct initial usability research

Navigation & Mapping Interface:

[wir] Design navigation UI layouts and interactions
[wir] Create map interface wireframes
[pro] Build interactive map prototype
[com] Design pin and waypoint components
[ast] Create navigation icons and markers

Telemetry & Data Visualization:

[wir] Design telemetry dashboard layouts
[pro] Create interactive data visualization prototypes
[com] Design biomedical data display components
[com] Build resource tracking indicators
[ast] Create telemetry icons and status indicators

Scientific & Procedure Interface:

[wir] Design XRF data display layouts
[pro] Create sample analysis interface prototype
[com] Design procedure step components
[ast] Create scientific data icons
[usr] Test procedure interface usability

Safety & Communication Systems:

[wir] Design warning notification layouts
[pro] Create alert system prototype
[com] Design warning indicator components
[ast] Create caution and warning icons
[usr] Validate warning system effectiveness

Advanced Interface Features:

[pro] Create autonomous navigation interface
[com] Design advanced mapping components
[com] Build complex data visualization elements
[ast] Create additional scientific icons
[usr] Conduct final usability testing

Using Github

Github Desktop

Please use GUI Github, it is much clearer and easier considering the number of branches we have.

Commit Convention

When creating a commit on GitHub for the related task, the commit message should be as follows:

[Task Card ID] ***space* (Verb in the present tense with the first letter capitalized) rest of the message**

eg. "[nav] Implement new navigation pin designs"
eg. "[tel] Adjust mission timer location"

This will make referencing the commits with tasks much easier

Working with HoloLens 2

PIN: 880088

Hand Simulation in Unity:

https://localjoost.github.io/Using-hand-simulation-with-MRTK3-in-the-Unity-Editor/#in-a-table

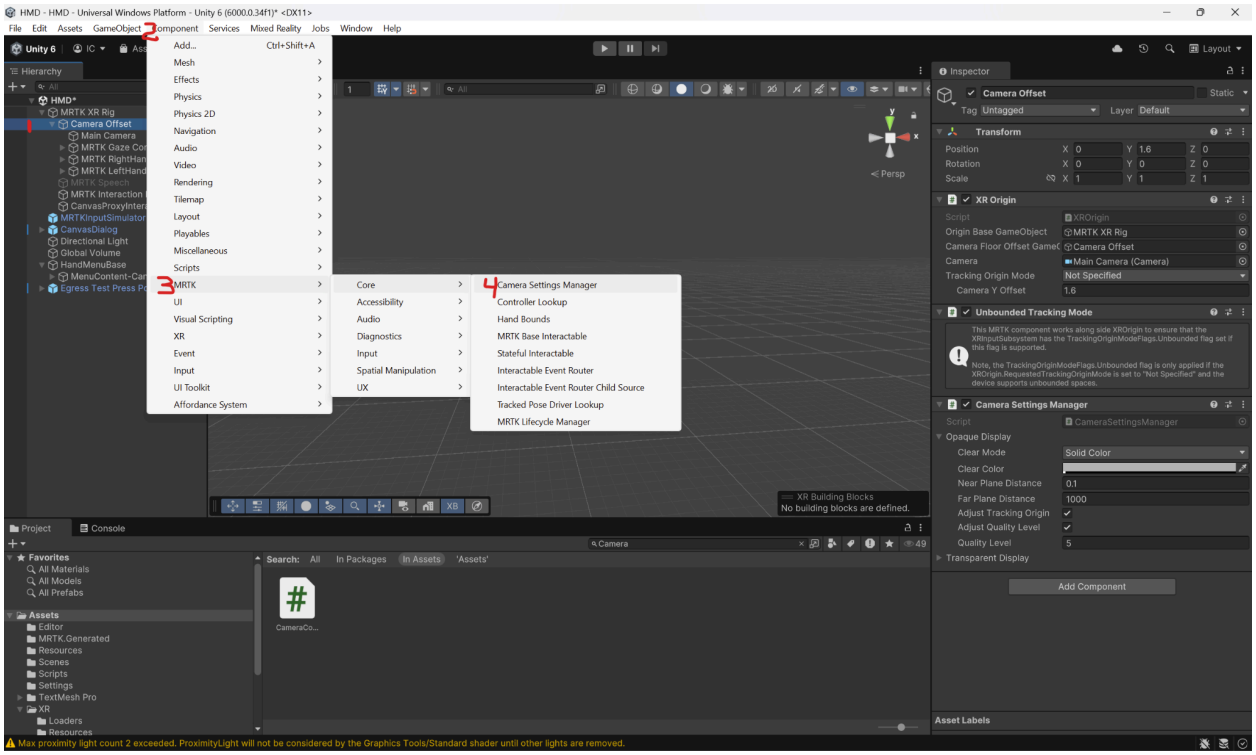
Migration Guide from MRTK2 to MRTK3

https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk3-overview/architecture/mrtk-v2-to-v3

5.1 Setting Up Unity for Hololens

- **Important:** We must set the skybox to transparent, since we don't need it in Mixed Reality (we want to see the IRL world)

- In order to do this, add a **CameraSettingsManager** component somewhere in the **MRTK XR Rig** component in the **HMD** scene.
  I (Israel) added it to the **Camera Offset** Subcomponent.

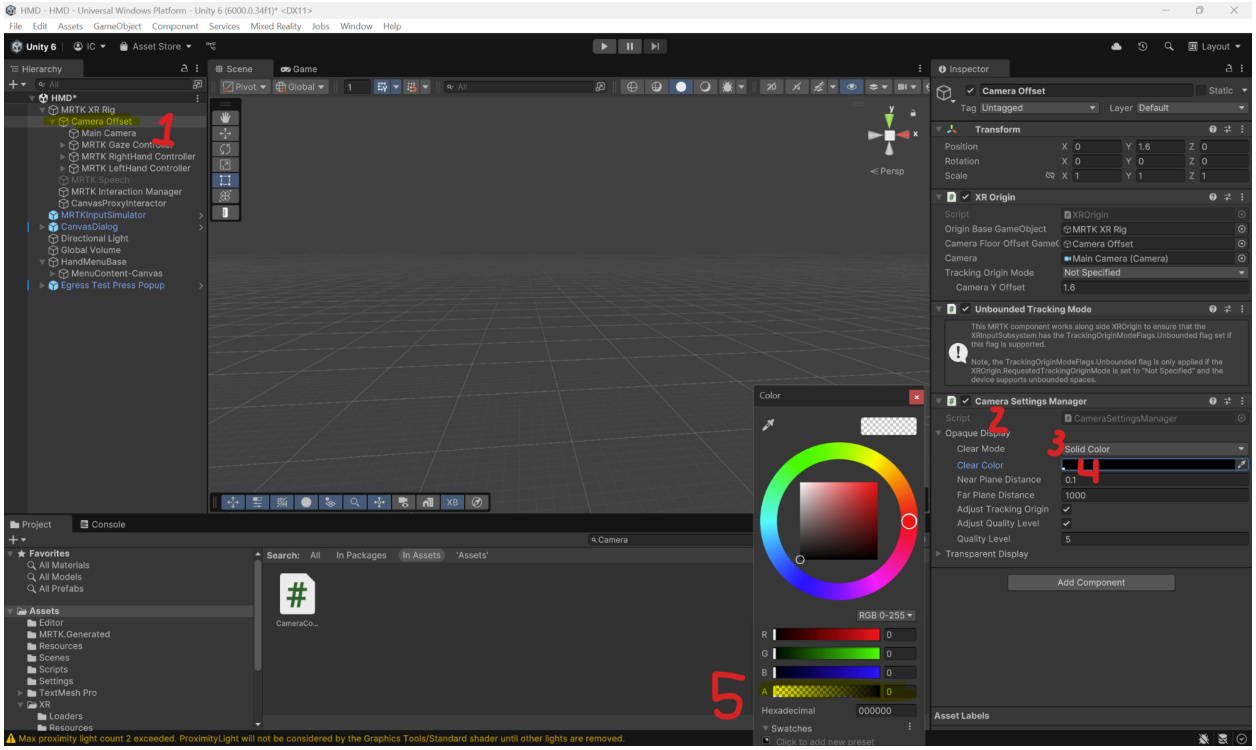


Once the **CameraSettingsManager** component is added, click the **Camera Offset** component, and take a look at it's properties in the **Inspector** tab on the right of the screen (see above screenshot).

Go to the **Opaque Display** section, and change the **Clear Mode** option from *Skybox* to *Solid Color*.

Press the *Clear Color* option below this, and set the **A** (alpha) value to 0 (bottom slider all the way to the left). This will set the skybox color to transparent.

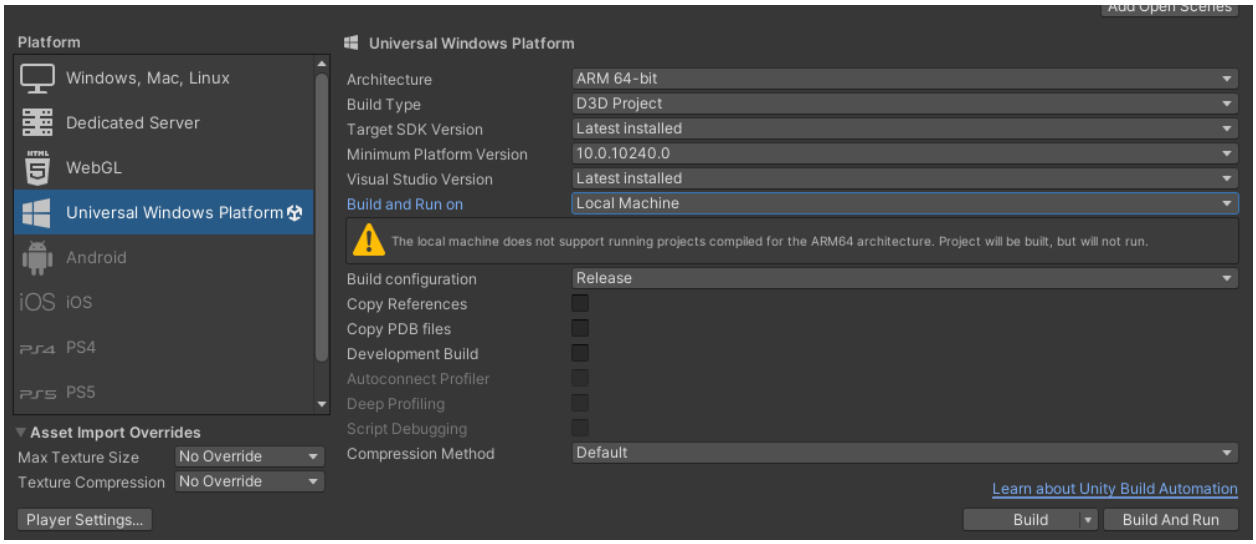
I've already done this in the HMD scene (see 3/6/25 commits), so no need to do this again.



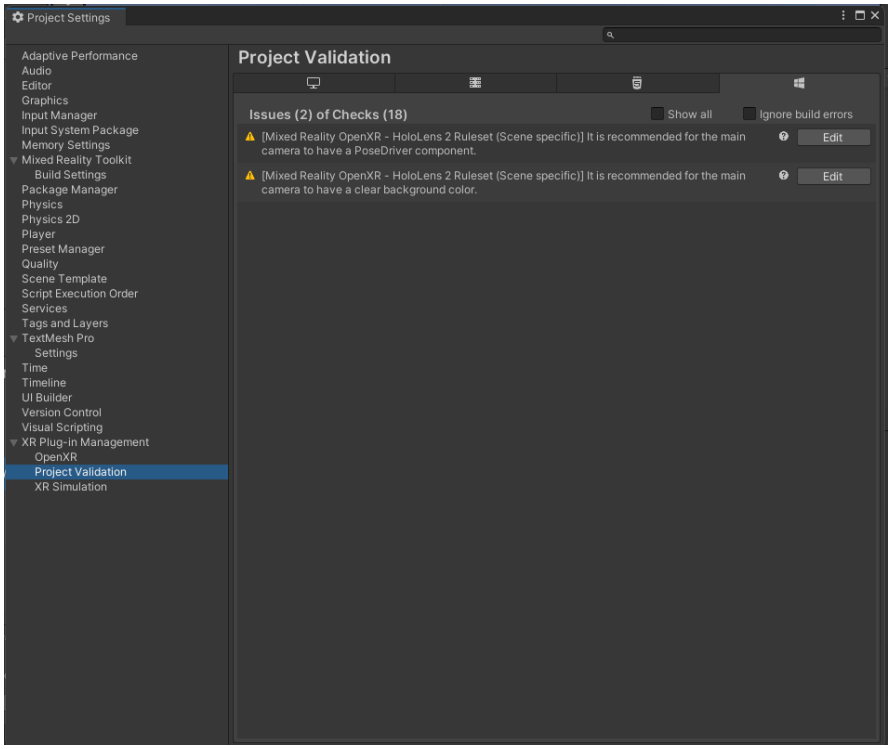
5.2 Building for HoloLens

https://learn.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/using-visual-studio?tabs=hl2

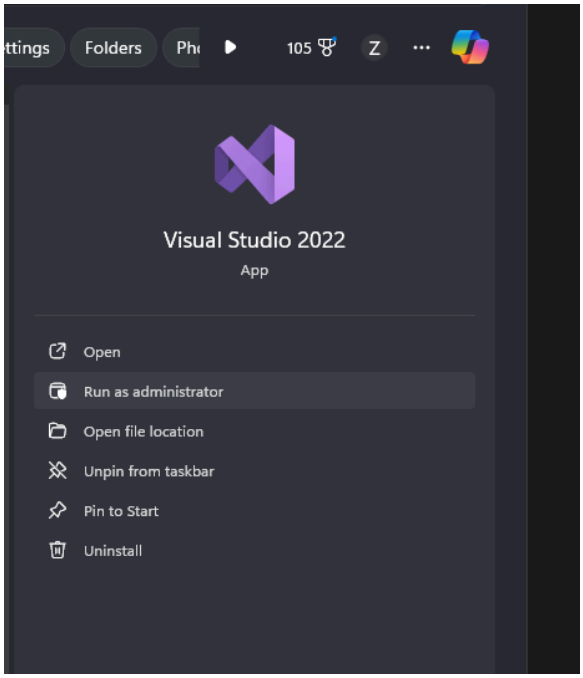
1. Setup build setting in Unity

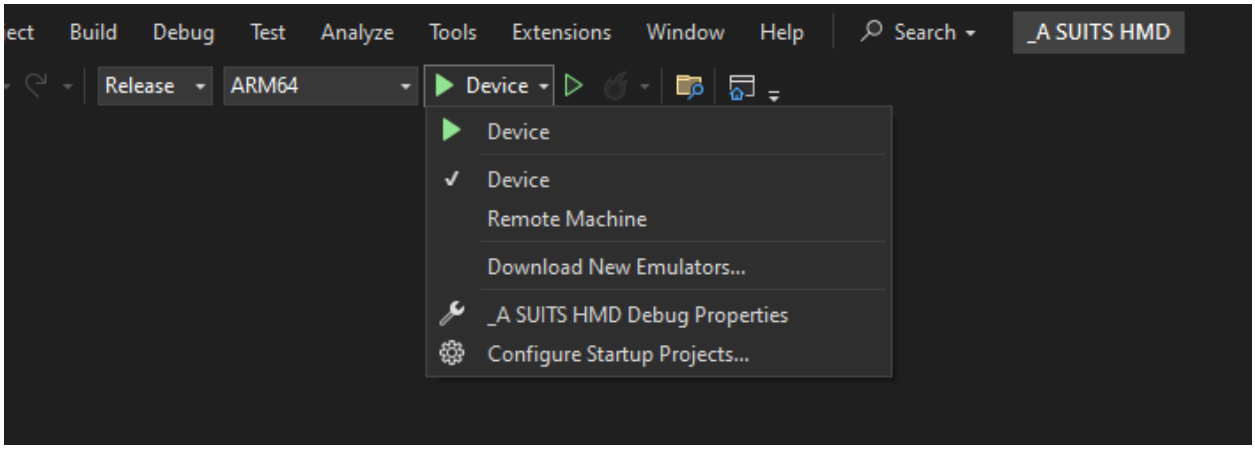


2. Make sure all build errors are solved (you will have to disable holographic remoting)



3. To avoid error: InvalidOperationException: Insecure connection not allowed
    a. **Project settings -->Player --> Other settings --> Allow downloads over HTTP: change this value to Always allowed (yellow exclamation warning icon appears, but it works)**
4. Once built (build it into the Builds folder, make sure it is empty first)

5. 

6. Run as administrator
7. open the build solution
8. 

    a. deploy it as release and not debug
    b. build for ARM64
    c. Visual Studio will prompt you for password,
        i. Do NOT disconnect the hololens
        ii. go to HoloLens Setting,
        iii. “For Developers”
        iv. and press “Pair” to get a password

5.3 Connecting to HoloLens via Window Device Portal (For remote deployment + screenshot/recording purposes)

Enter the IP address in the browser and access the HoloLens 2 through your laptop for live preview of the camera etc.

5.4 Holographic Remoting (KEY FOR DEV!!!)

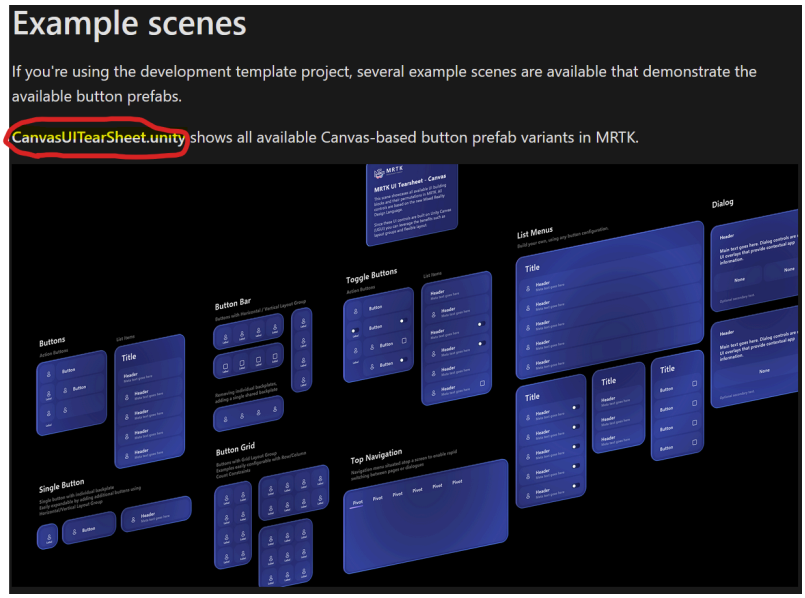
- Remoting feature allows you to deploy Unity play mode directly onto the HoloLens without having to build and deploy repeatedly
- Article: http://www.lancelarsen.com/xr-step-by-step-2023-hololens-2-holographic-remoting-in-unity-2022-solving-skybox-issue/

5.5 Using Sample Components

- The MRTK3 documentation frequently mentions sample scenes to get assets (such as UX Components), but doesn't mention how to import them in Unity. Let's go over this process.

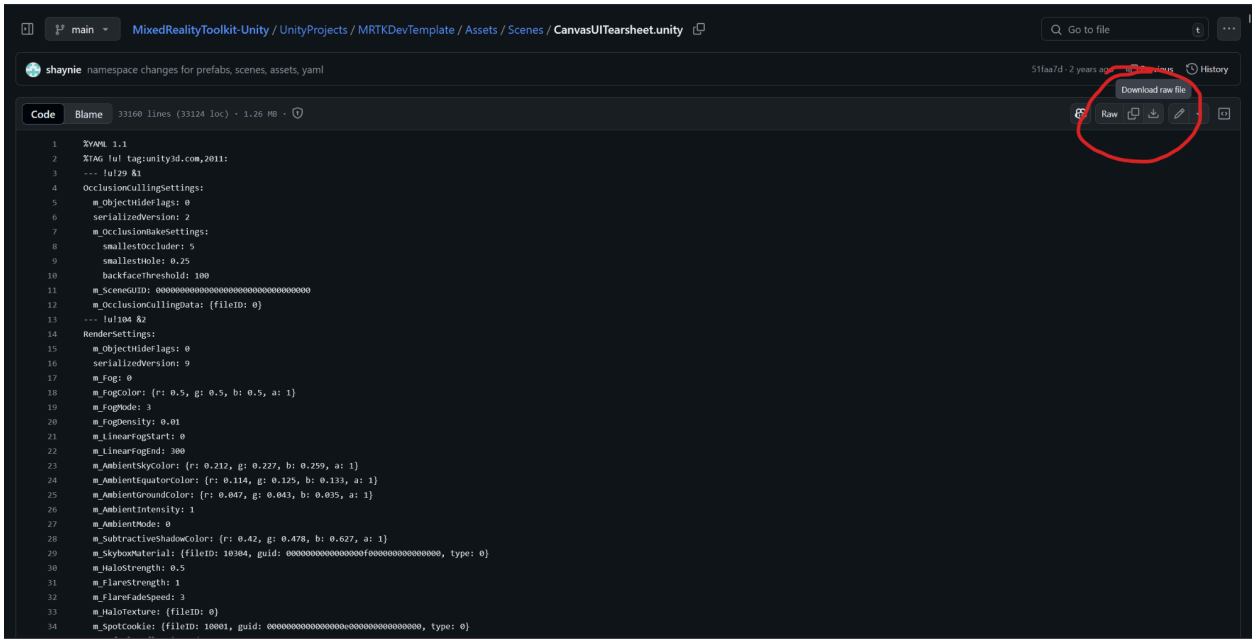
For this example, we'll look at the Button Documentation and try to import a sample button.

1. Usually, the documentation will say which Scene file it is referring to. For instance, this part of the button documentation refers to the **CanvasUITearSheet.Unity** scene.

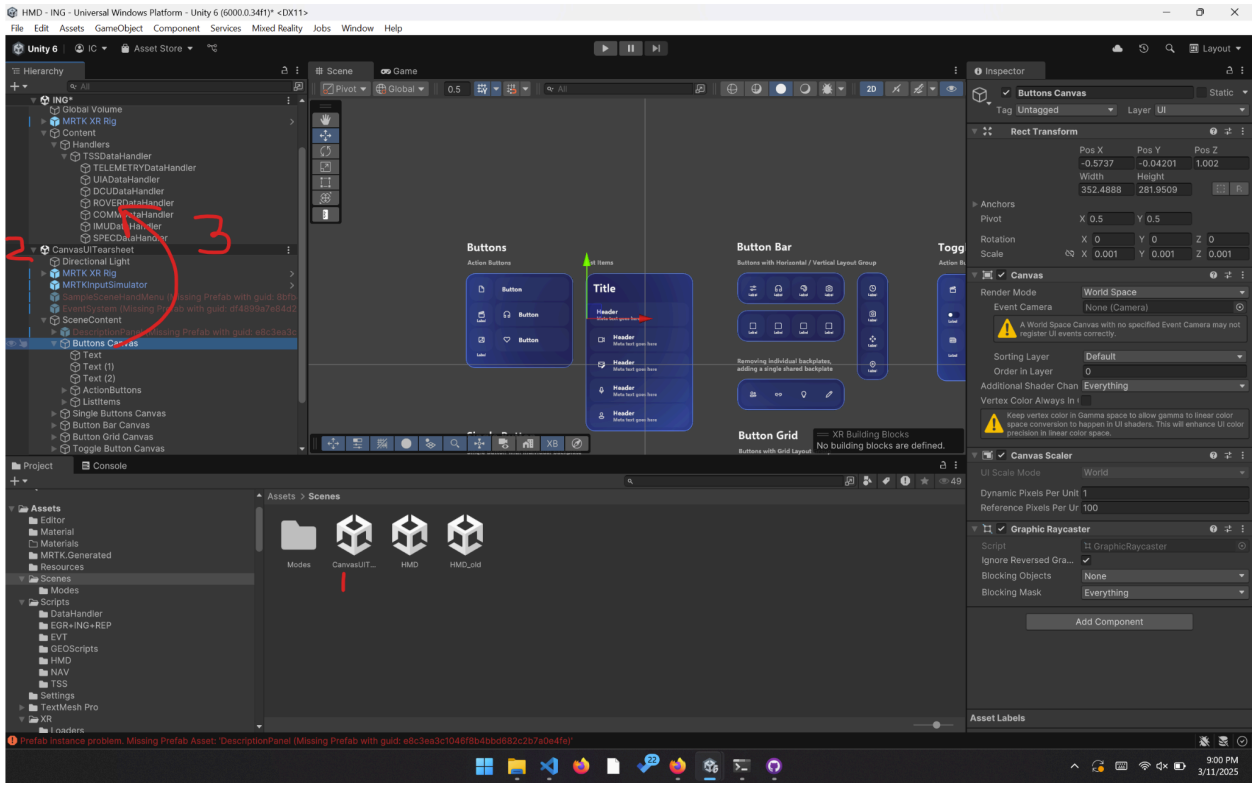


2. To find this file, go to the MRTK3 GitHub repo, and navigate to the **MixedRealityToolkit-Unity/UnityProjects/MRTKDevTemplate/Assets/Scenes/** folder.

3. CTRL+F for the file, click it, then press download button on the right



4. Now, transfer this file into your scenes folder (wherever it is on your hard drive), then you can open the scene in the same project and drag drop stuff into your current working scene,

5. 

5.6 Building/Pushing to HoloLens 2

Performance - MRTK 2 | Microsoft Learn Follow this later to optimize builds