

Worldview AR Documentation

About:

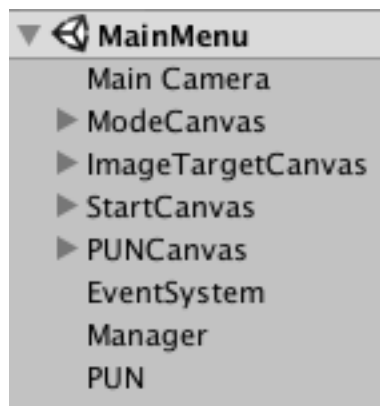
Worldview AR is an interactive augmented reality prototype to visualize NASA earth observing satellite imagery. The functionality to ingest tiles from GIBS into Unity3D is used from the EVRTH project, documentation for EVRTH found in the EVRTH folder of Assets. Through this application, users can visualize and interact with satellite imagery in novel ways by utilizing augmented reality. Additionally, a game mode is provided to give players a guided experience within the application. A networked multiplayer mode is also provided to enhance engagement.

<https://github.com/nasa-gibs/worldview-ar>

Main Components Overview

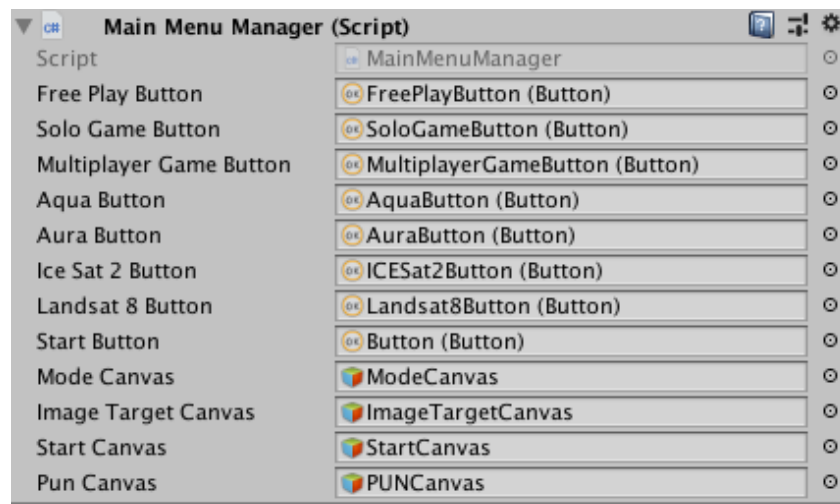
Main Menu Scene:

The main menu scene includes all of the UI elements and functionality to set up the application with the right mode. The hierarchy contains all of the UI elements separated by canvas. The Manager contains the Main Menu Manager, which controls the functionality of the menu. “PUN” contains the Launcher script used for multiplayer.



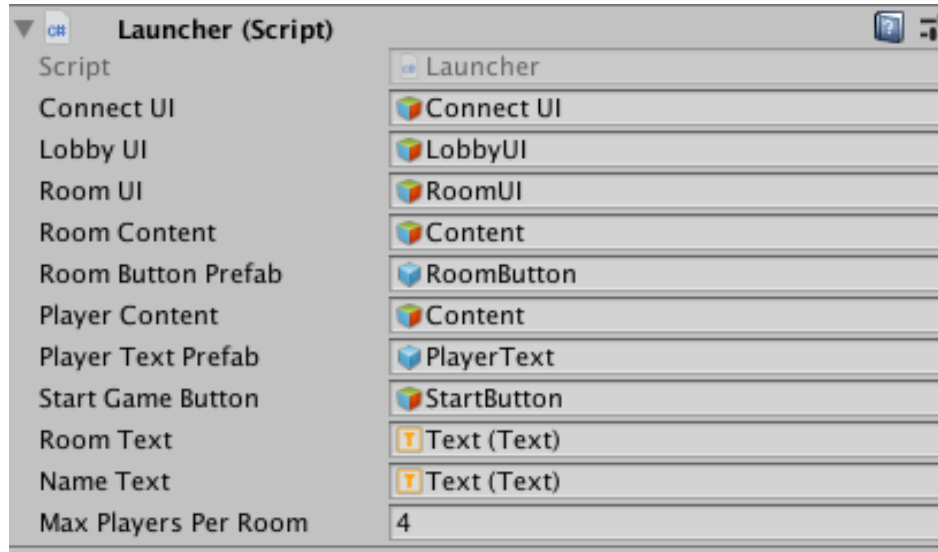
Main Menu Manager:

Most of the UI elements are assigned to the Main Menu Manager, as shown below. This manager controls the flow of the menu, displaying the correct UI and taking input. Essentially, it just turns on and off the canvases in the correct order to gather all the necessary information from the user.



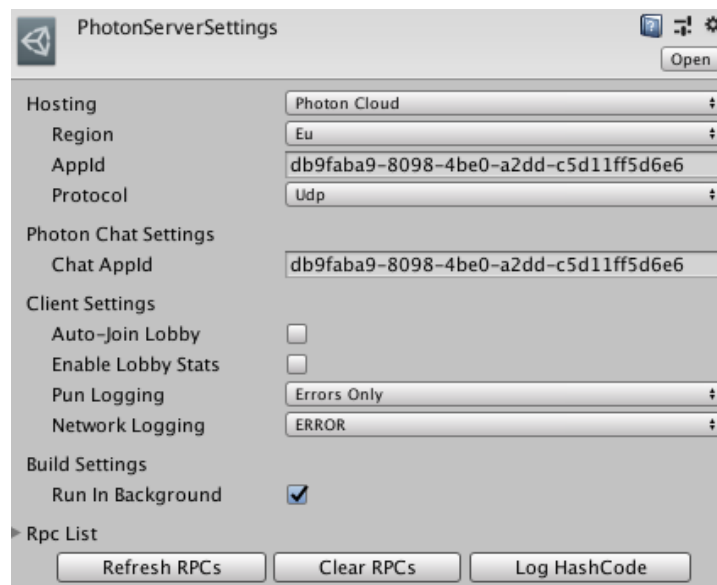
Launcher:

The Launcher is activated when the user choses the multiplayer mode. This script contains the functionality to connect to Photon. The related UI elements are assigned to this script to receive input and display the necessary information. This was a quick implementation to get devices networked. It may be beneficial to expand the connection process to display more information to the user about the connection progress as well as provide fallbacks for connection failures along the way.



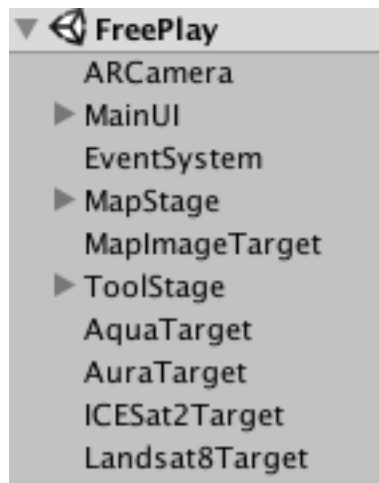
PhotonServerSettings:

The PhotonServerSettings contains some of the main settings for Photon. The AppId is the only thing that may need to change.



Free Play Scene:

The free play scene contains all the functionality to allow users to explore various preset data sets and dates. The MainUI contains all of the on screen UI elements for the scene. The MapStage contains the EVRTHManager, Globe and Map gameobjects. An Image Target Stage script is attached to mirror the Map Image Target and provide additional functionality. The ToolStage contains all of the interaction tools for the application. An Image Target Selector script is used to assign the stage to the correct image target for interaction.



ARCamera:

The ARCamera is a prefab for the Vuforia API. This camera is what is used for the scene and contains the Vuforia Behavior script. The Vuforia configuration can be opened from the inspector when selecting the ARCamera. This configuration provides many of the settings and key for Vuforia.

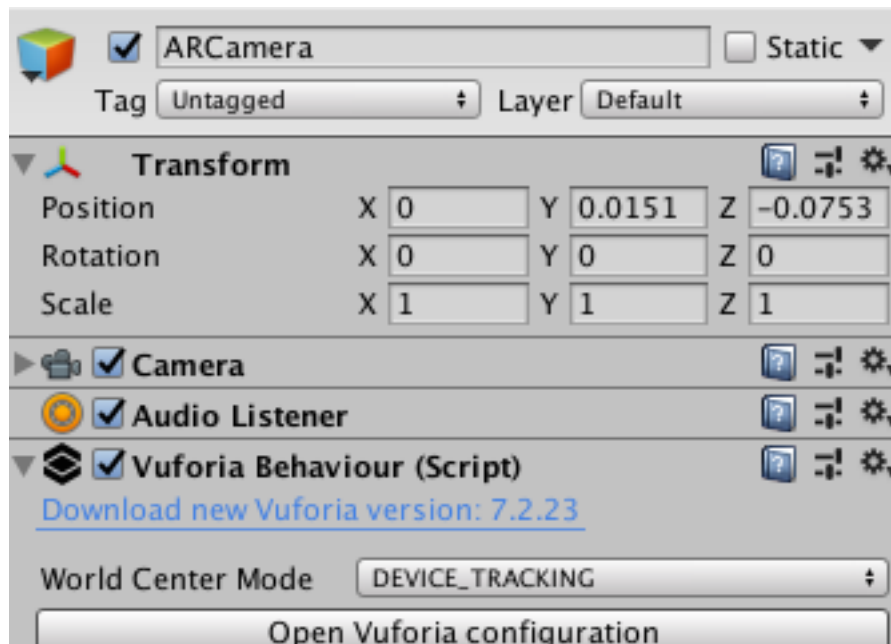
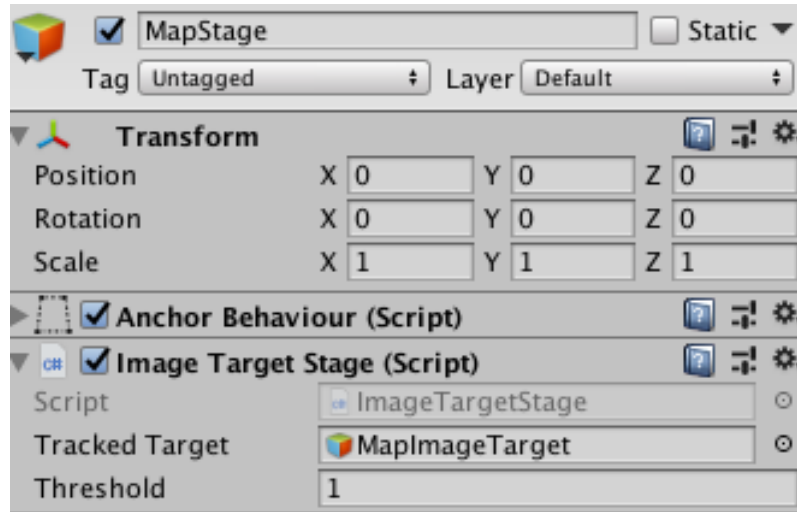


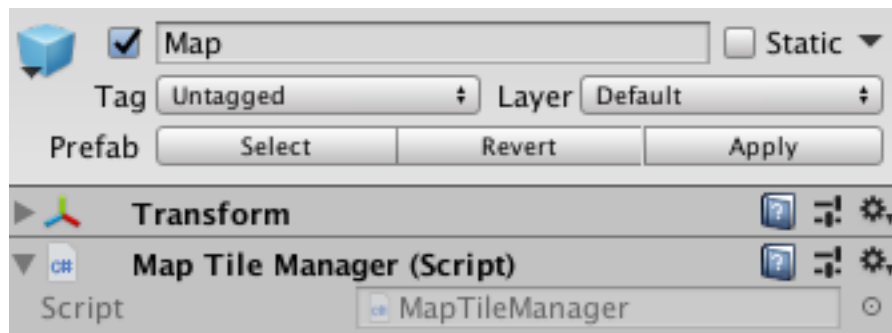
Image Target Stage:

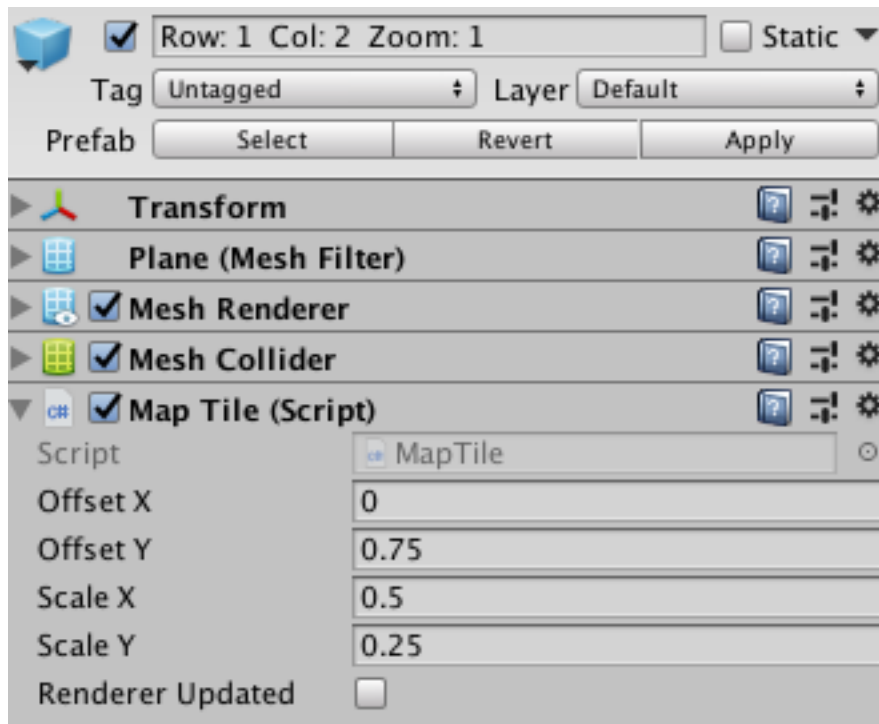
The Image Target Stage is a wrapper around the Vuforia image target tracking. This script mirrors the transform of the image target. However, the orientation is compared against a set orientation to ensure a false positive does not cause errors. Currently the script is set to have the image target laying flat horizontally. The threshold is how many degrees, in angles, away from the set orientation the image target will update with.



Map Tile Manager & Map Tile:

The Map Tile Manager controls all of the individual Map Tiles. This is crucial to easily call functions on all of the Map Tiles. Each Map Tile finds the associated Globe Tile and copies its texture. This allows us to only load in a tile from GIBS once for both the Globe and Map. These scripts also provide functionality for the AB Comparison Tool. The offset and scale for the tile must also be declared for each Map Tile. Note: The Map gameobject holds all of the tiles pre-generated to save on computational power at run time. The globe is generated at run time and creating meshes during run time is relatively expensive.





Marker Trackable Event Handler:

The Marker Trackable Event Handler is very similar to Vuforia's Default Trackable Event Handler. The only modification is that "Tracked" is exposed as a public variable so other scripts can check if the image target is currently in view of the camera.

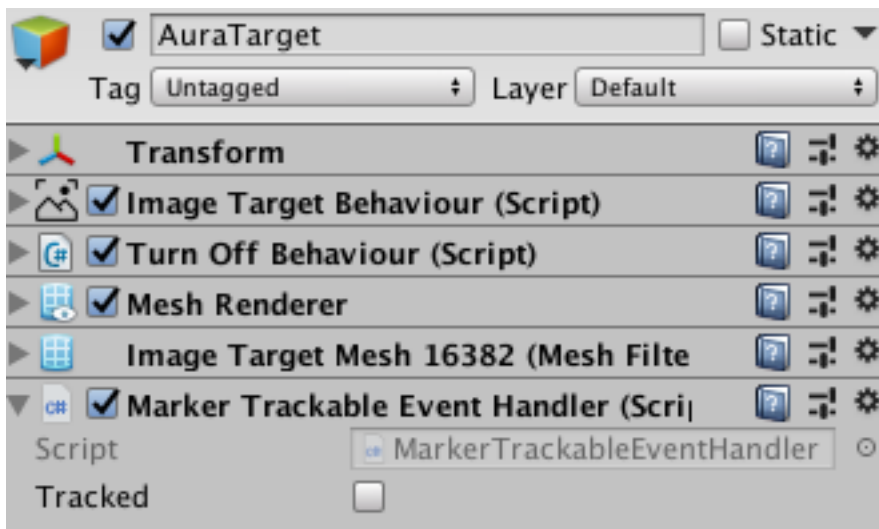
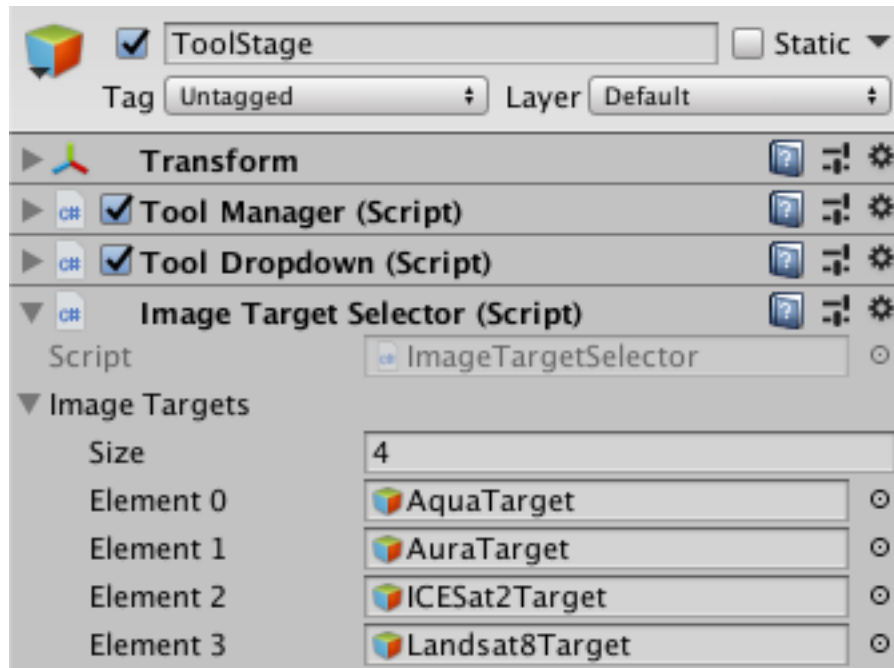


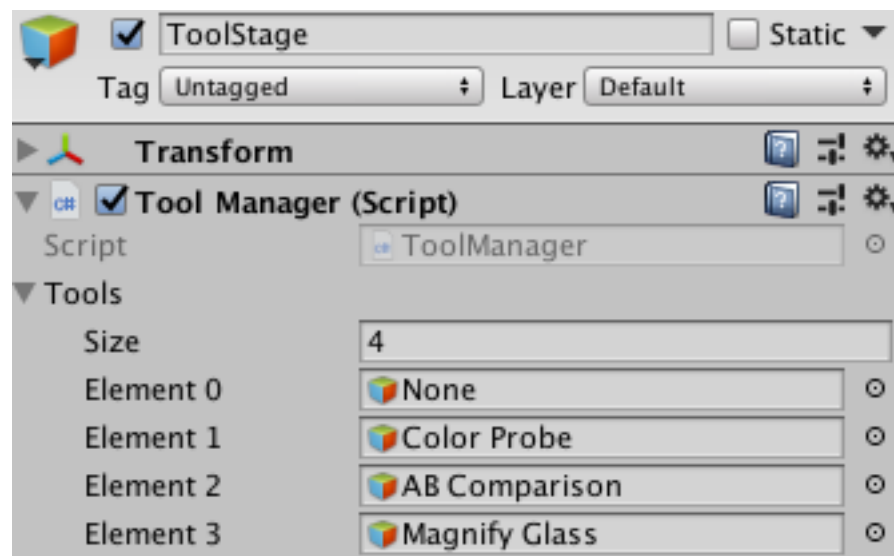
Image Target Selector:

The Image Target Selector takes in all of the available image targets. Based on the image target chosen in the main menu, the ToolStage is assigned to and becomes a child of that image target. The other image targets are destroyed to save on computational resources.



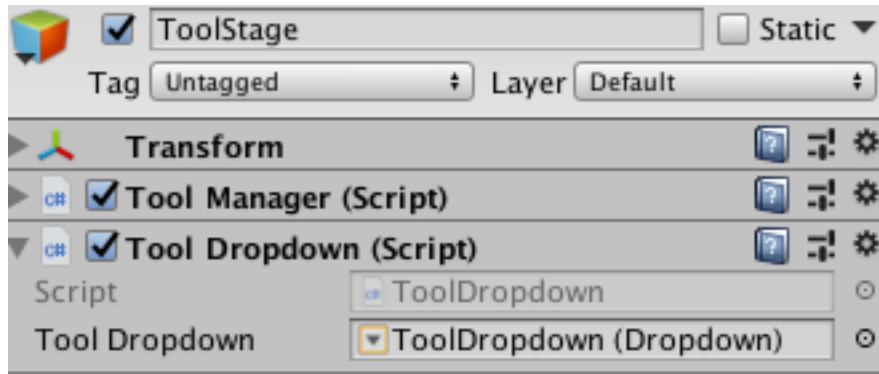
Tool Manager:

The Tool Manager provides access to toggle the various interaction tools. Only one interaction tool can be active at a time.



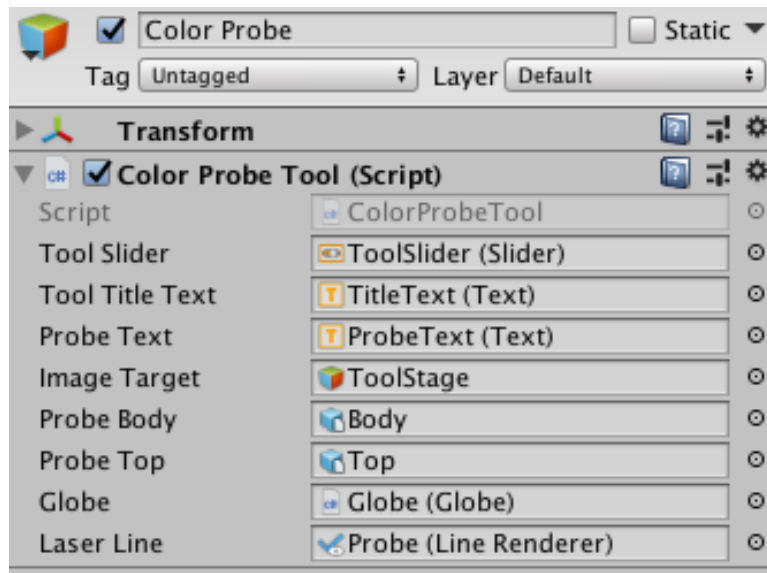
Tool Dropdown:

The Tool Dropdown allows for the interaction tools to be selected from a dropdown UI element.



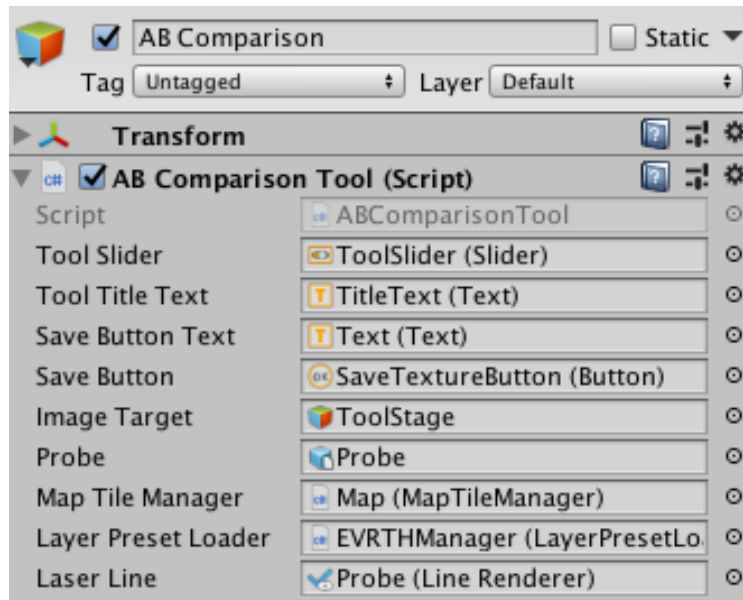
Color Probe Tool:

The Color Probe Tool performs a raycast in the downward direction of the tool image target. If the raycast hits a collider, the rgb value is returned for that specific location on the texture. This rgb value is compared against the color map key to find the associated value.



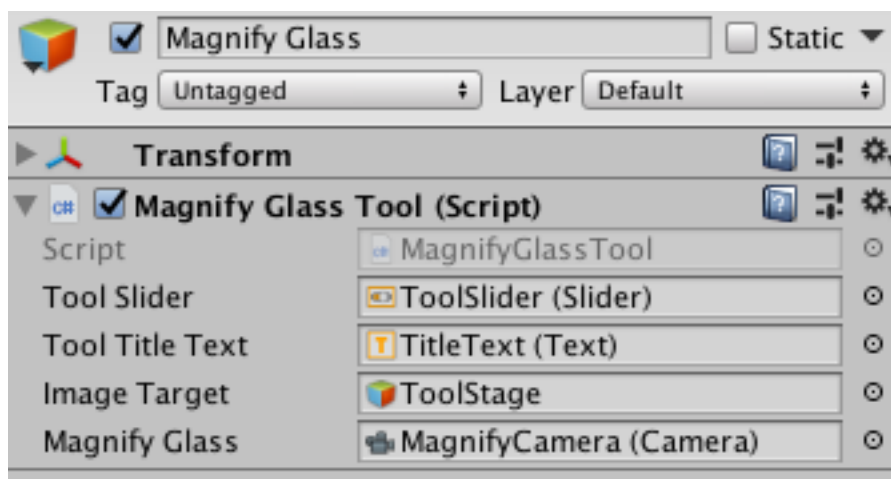
AB Comparison Tool:

The AB Comparison Tool performed a raycast in the downward direction of the tool image target. If the raycast hits a collider, that position is passed to the shader of the Map and Globe Tiles. This sets the spy position to control where the AB comparison spy radius is located.



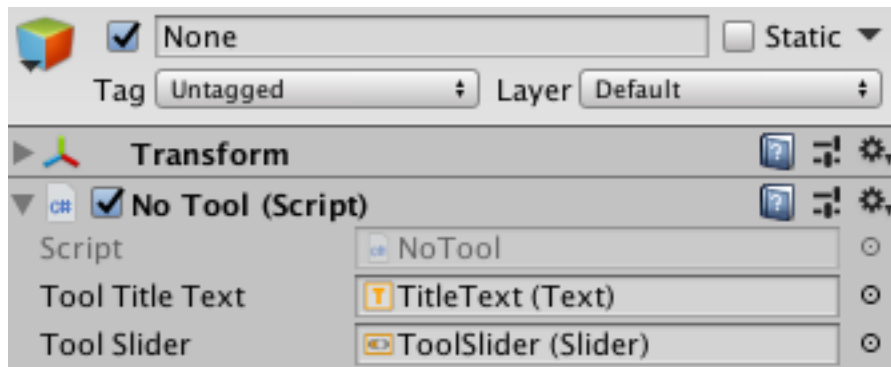
Magnifying Glass Tool:

The Magnifying Glass Tool utilizes a second camera and render texture functionality. A camera is facing in the downward direction of the tool image target and the image is rendered to a cylinder above it. The zoom level can be altered with field of view of the camera.



No Tool:

No Tool provides very little functionality. Basically this script is just to control various UI elements when no tool is selected.



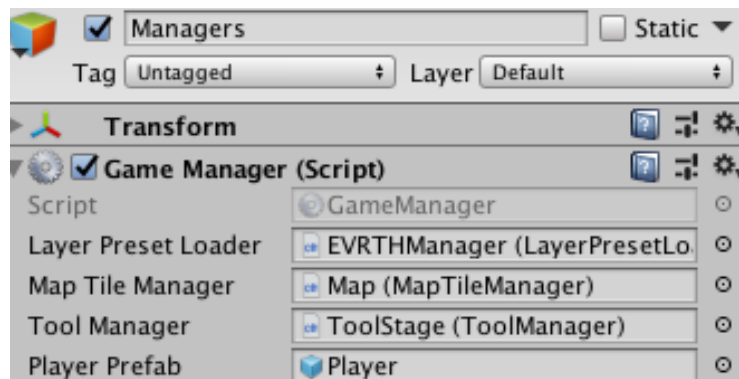
Game Scene:

The Game Scene contains much of the same functionality as the Free Play Scene. The main difference here is the Managers gameobject containing the scripts for Game Manager, Question Manager, and Game UI Manager.



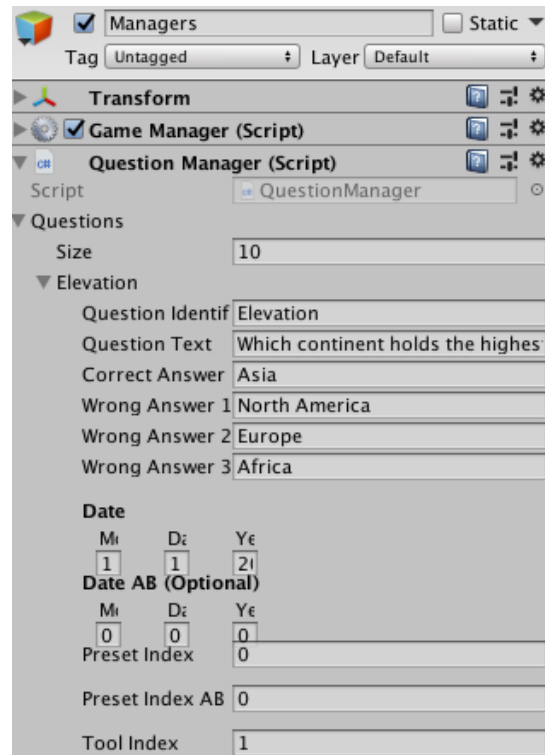
Game Manager:

The Game Manager script contains the base functionality for the game mode including methods that will adapt for either solo offline or multiplayer mode. A basic state machine is used to drive the game and is controlled by the master client. The master client will send out events to the other instances in the room.



Question Manager:

The Question Manager is used to author all of the questions used in the game. From the inspector you can fill in the information requested for each question. The AB fields are only required if you are using the AB Comparison Tool for that question.

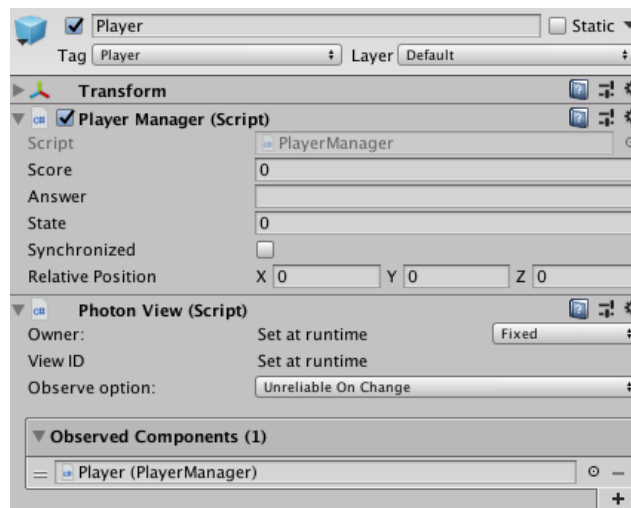


The screenshot shows the Unity Inspector for the 'Question Manager (Script)' component. The 'Script' dropdown is set to 'QuestionManager'. The 'Questions' section is expanded, showing a single question with the following fields:

- Size: 10
- Elevation: (empty)
- Question Identif: (empty)
- Question Text: Which continent holds the highest elevation?
- Correct Answer: Asia
- Wrong Answer 1: North America
- Wrong Answer 2: Europe
- Wrong Answer 3: Africa
- Date: (empty)
- Date AB (Optional): (empty)
- Preset Index: 0
- Preset Index AB: 0
- Tool Index: 1

Player Manager:

The Player Manager is attached to the Player prefab found in WorldviewAR -> Prefabs -> Networking -> Resources -> Player. A Photon View component must be attached to the player prefab for networking capabilities. Additionally, when using OnPhotonSerializeView, the order you send information has to be the same order you receive it.



The screenshot shows the Unity Inspector for the 'Player' prefab. The 'Player Manager (Script)' component is selected, showing the following fields:

- Score: 0
- Answer: (empty)
- State: 0
- Synchronized: (checked)
- Relative Position: X 0, Y 0, Z 0

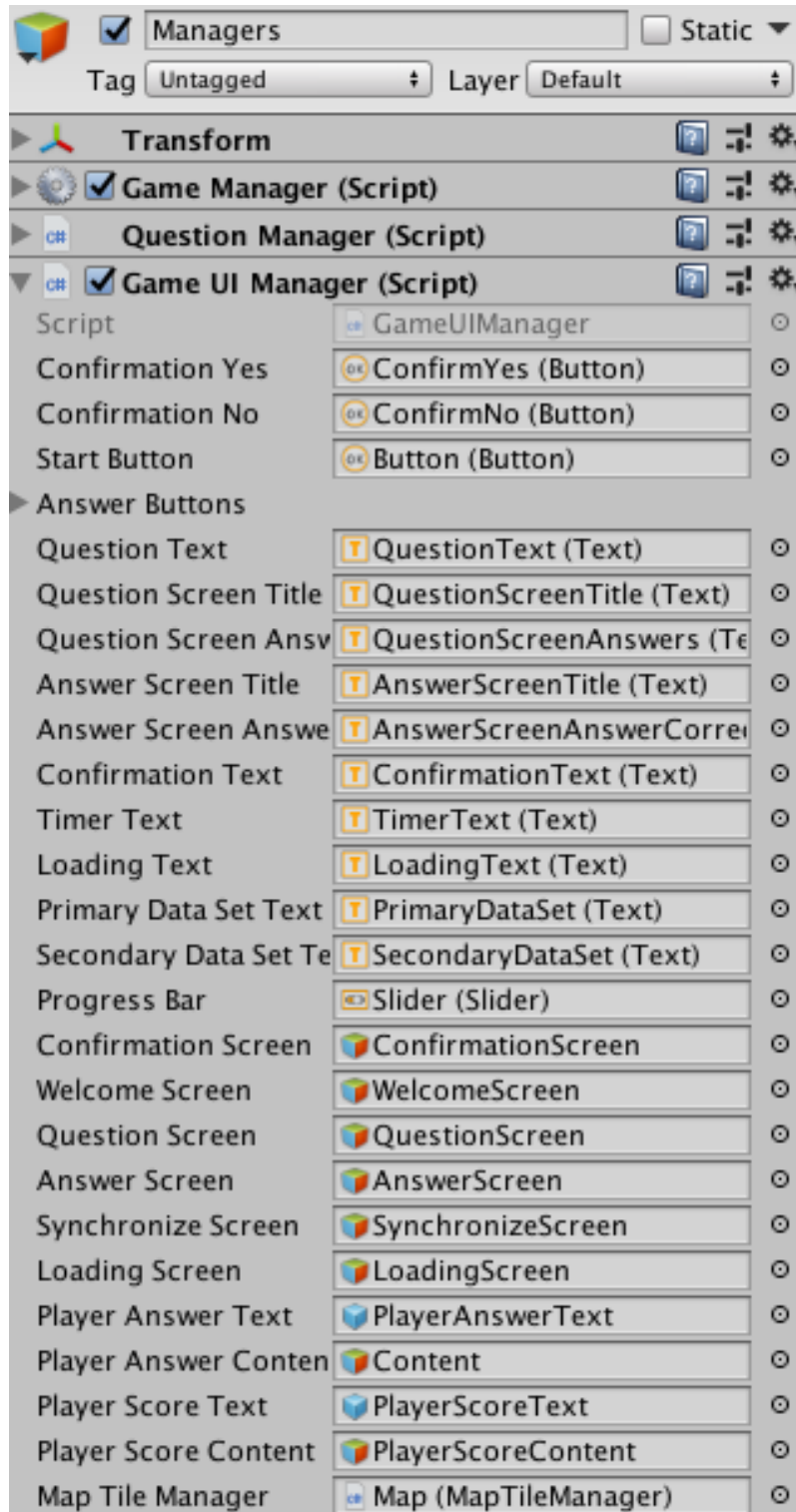
The 'Photon View (Script)' component is also visible, showing the following fields:

- Owner: Set at runtime
- View ID: Set at runtime
- Observe option: Unreliable On Change

The 'Observed Components (1)' section shows the 'Player (PlayerManager)' component.

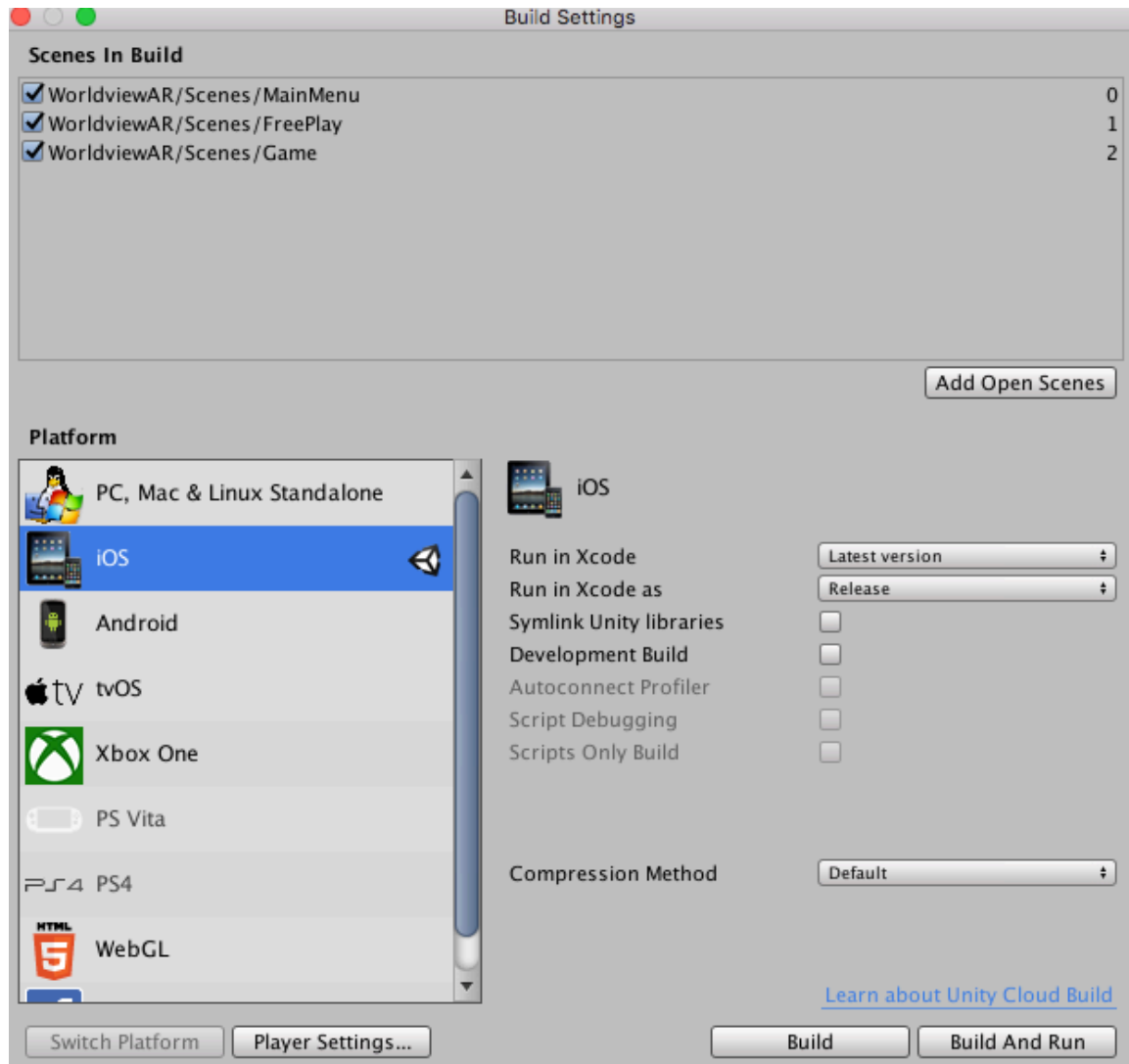
Game UI Manager:

The Game UI Manager registers all of the UI elements and provides public functions to access the functionality. The Game Manager primarily uses this to get input and update various text fields.



How to Build iOS:

Unity Project Opened -> File -> Build Settings -> Build and Run



An Xcode project will be generated in the folder you created -> Open the Xcode Project

WorldviewAR_v1	Today at 4:56 PM	--	Folder
build	Today at 4:56 PM	--	Folder
Classes	Today at 4:56 PM	--	Folder
Data	Today at 4:56 PM	--	Folder
Info.plist	Today at 4:56 PM	2 KB	Property List
LaunchScreen-iPad.png	May 8, 2018 at 12:06 PM	128 bytes	PNG image
LaunchScreen-iPad.xib	May 8, 2018 at 9:05 AM	2 KB	Interfa...cument
LaunchScreen-iPhone.xib	May 8, 2018 at 9:05 AM	2 KB	Interfa...cument
LaunchScreen-iPhoneLandscape.png	May 8, 2018 at 12:06 PM	128 bytes	PNG image
LaunchScreen-iPhonePortrait.png	May 8, 2018 at 12:06 PM	128 bytes	PNG image
Libraries	Today at 4:58 PM	--	Folder
MapFileParser	May 9, 2018 at 11:14 AM	173 KB	Unix executable
MapFileParser.sh	May 8, 2018 at 9:05 AM	1 KB	Shell Script
process_symbols.sh	May 8, 2018 at 9:05 AM	373 bytes	Shell Script
Unity-iPhone	Today at 4:56 PM	--	Folder
Unity-iPhone Tests	Today at 4:56 PM	--	Folder
Unity-iPhone.xcodeproj	Today at 4:56 PM	181 KB	Xcode Project
UnityData.xcassets	Today at 4:56 PM	--	Folder

Once you open the project, there will be no provisioning profile set. Select Automatically manage signing. After that, you should be able to find your Provisioning Profile under the dropdown.

▼ Identity


Display Name	Worldview AR
Bundle Identifier	gov.nasa.neacc.WorldviewAR
Version	1.0
Build	0

▼ Signing

☐ Automatically manage signing
Xcode will create and update profiles, app IDs, and certificates.

▼ Signing (Release)

Provisioning Profile	None
Team	None
Signing Certificate	None

Status  "Unity-iPhone" requires a provisioning profile.
Select a provisioning profile for the "Release" build configuration in the project editor.

Finally, at the top of the Xcode project select your device. Then go to Product -> Run and the application should successfully be built to your device!

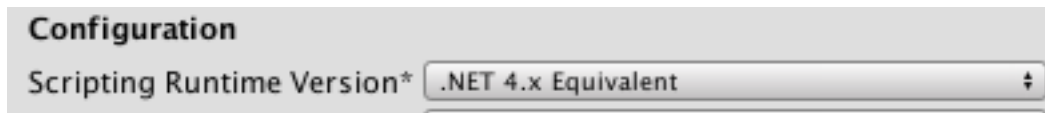


Possible iOS Build Issues:

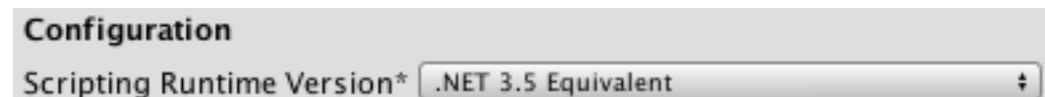
The following issues may or may not occur depending on how your environment is set up. Additionally, any other issues not listed here may need to be googled, nothing should be too serious or prevent you from completing a build.

Scripting Runtime Version:

The Scripting Runtime Version for this project needs to be set at .NET 4.x Equivalent. However, some machines may fail to build this.



To overcome this issue, first build with .NET 3.5 Equivalent. This will provide a successful build. From here switch back over to .NET 4.x Equivalent and this time append the previous build instead of replacing it.

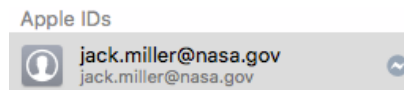


Can't Find Provisioning Profile:

If you cannot find your provisioning profile under the dropdown in Xcode you need to first be sure you have created an apple developer account ->

<https://developers.google.com/vr/develop/unity/get-started-ios>

After this, go to Xcode -> Preferences -> Accounts and add in your developer account.



Conflicting Provisioning Profile Settings:

If you receive the error that Unity-iPhone has conflicting provisioning settings, we just need to change a few settings in the Xcode project.

▼ Signing

☒ Automatically manage signing

Xcode will create and update profiles, app IDs, and certificates.

Team Jack Miller (Personal Team)

Provisioning Profile Xcode Managed Profile

Signing Certificate iPhone Development

Status ! **Unity-iPhone has conflicting provisioning settings.**

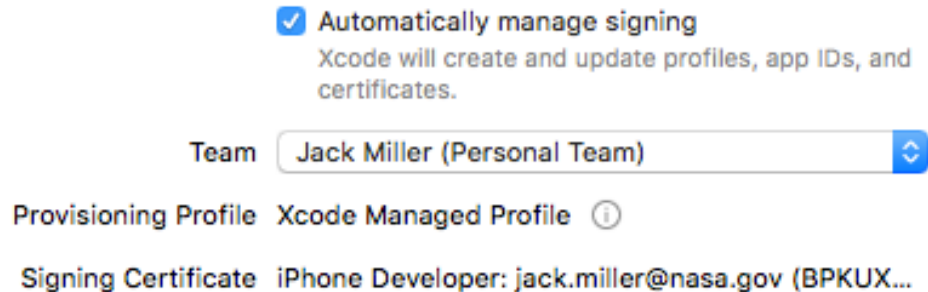
Unity-iPhone is automatically signed, but code signing identity iPhone Development has been manually specified. Set the code signing identity value to "iPhone Developer" in the build settings editor, or switch to manual signing in the project editor.

With the Xcode project open go to Build Settings and scroll down to Signing. Change all of the dropdown options to iOS Developer as shown below.

▼ Signing	
Setting	Unity-iPhone
Code Signing Entitlements	
▼ Code Signing Identity	iOS Developer
Debug	+ iOS Developer
Any iOS SDK	iOS Developer
Release	iOS Developer
Any iOS SDK	iOS Developer
ReleaseForProfiling	iOS Developer
Any iOS SDK	iOS Developer
ReleaseForRunning	iOS Developer
Any iOS SDK	iOS Developer
Code Signing Inject Base Entitlements	Yes
Code Signing Style	Automatic
Development Team	Jack Miller (Personal Team)
Other Code Signing Flags	
Provisioning Profile	Automatic
Provisioning Profile (Deprecated)	Automatic

Then go back to general and you will see that there are no longer any errors. Proceed to build the application to your device.

▼ Signing



How to Build Android:

While I have not tested this specific project for android, the build process should be relatively simple.

Unless you have done so already, you will have to setup the android environment for Unity. The instructions to do so can be found here.

<https://docs.unity3d.com/Manual/android-sdksetup.html>

- Choose step 2b instead of 2a and download / install Android Studio
- The path to the android sdk can be found here
 - C:\Users\”YourName”\AppData\Local\Android\Sdk
 - You will have to show hidden folders to see AppData*

Common Issues:

- If you receive “Unable to list target platforms. Please make sure the android sdk path is correct. See the Console for more details.” when trying to build the application, see the following link.
 - <https://answers.unity.com/questions/1323731/unable-to-list-target-platforms-please-make-sure-t.html>
- If you receive “Gradle Failure” when trying to build the application, see the following link. You may have to downgrade your jdk.
 - <https://answers.unity.com/questions/1419389/how-to-fix-android-gradle-failure.html>

After your environment is properly set up, change the Build Settings in Unity to the Android platform and select build and run. With your device plugged in, it will first build an apk in a folder you choose and then it will deploy it to your device.

Future Work and Polish

UI (All Scenes)

Many of the UIs in this application were quickly designed just to be functional. It would be nice to design a theme for the application and apply it to all of the UI elements for a more professional look.

Photon App ID

The Photon App ID needs to be updated. Ideally, create a photon account that everyone can access and generate a free development ID for the application. The current ID is attached to Jack Miller's NASA log in so that needs to be removed.

Vuforia App License Key

The Vuforia App License Key needs to be updated. Ideally, create a Vuforia account that everyone can access and generate a free development key for the application. The current key is attached to Jack Miller's NASA log in so that needs to be removed.

Update Unity and Vuforia Version

As long as there are no known major bugs I would suggest keeping these updated as you go. Useful functionality for both are always being added and its good to keep things up to date. I didn't have time this summer to keep up as I was rapidly adding in new features but just be sure nothing breaks between updates.

Launcher (script) and Connecting to Photon

The current implementation of Photon was developed to get Photon working as quickly as possible. The Launcher script provides all the functionality to get connected but this could be polished quite a bit. I would give the user more feedback into the connection status and provide better fallbacks when things fail.

FreePlay and Game Scene MainUI

My initial intent was to keep the screen space UI limited so I just created a single MainUI canvas to hold all of the UI elements. However, as the project progressed more and more elements were added so it got fairly bulky. It may be beneficial to break up this UI into sub parts for organizational purposes but functionally it's fine.

EVRTH Tile Ingest Hangs

If possible, try to discover why the tile ingest will occasionally indefinitely hang for long periods of time. It appears that this happens mostly when the connection is dropped but it doesn't seem to restart when the connection is regained. Switching to another layer or date will overcome the hang up.

Image Target Stage

The current Image Target Stage compares its orientation relative to a flat horizontal surface. It may be beneficial to allow the user to select what the correct orientation should be or perhaps take the median of many readings to automatically establish what the correct orientation should be.

Interaction Tools

Initially it wasn't planned to have all of the interaction tools, they were added in over time when we thought something would be useful. Currently, they are all separate scripts but share much of the same functionality. Creating a tool base class that all of the interaction tools derive from would be a good idea.

Game Manager

There are two timeout functions that could fairly easily be combined. The only difference is that one is modified for AB comparison since it has two data sets to load.

Game UI Manager

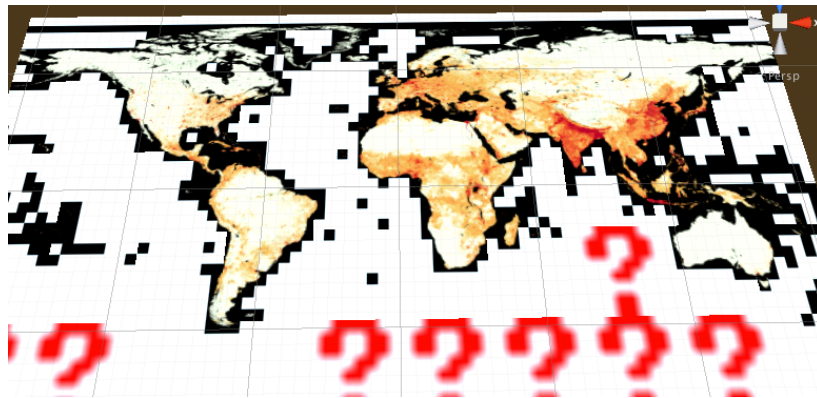
This script became much heavier than anticipated. This could probably be broken up into pieces for the sake of organization. I would also remove its ability to change states and keep that all in the Game Manager.

Player Manager

The name display of other players devices could be extended into cool special effects for players when they answer questions right and things of that nature. Additionally, you could place avatars in that position when players are competing in completely different areas.

Tile Ingest Format Issues

Unity can only process JPG and PNG image formats. When Unity tries to generate a texture from an unknown format you get the red question marks shown below. These red question marks always have a size of 8x8 so a hacky fix was implemented to look for this and change it to an all black texture when it happens (See DownloadManager). Unity doesn't provide any call back when the texture generation fails in that manner. Strangely, the some tiles, like the ocean shown here, appear to have a white or different colored even though they behave like black pixels when you blend them with other layers.



Get Capabilities Update

The EVRTH project embedded an old version of Get Capabilities within the project. I tried to just swap in updated copy but some format differences were giving me issues. However, for now you can just copy in new layers into the existing Get Capabilities as needed. In the future this should be fixed.

Adding in New Image Targets

Log into developer.vuforia.com -> Target Manager

Add new Database or select an existing Database

Add Target -> Upload picture with other required information

Make sure the width is correct or there will be issues with tracking

Select Download Database and chose Unity for the development platform

Import newly created Unity package into the project

In the hierarchy, select the image target and change the image target under Image Target Behavior

It is possible that if the new image target is a different size than the old one there may be strange scaling effects happening. If the app is not functioning right when changing image targets, look more into this. However, I believe with how I set it up, this shouldn't be a problem.

Other

There was a ton that had to get done this summer with very little time so some things may have not been written as cleanly as I would have liked. Feel free to polish anything that can be improved upon. If there is anything I wrote that you are confused about please contact me. I tried to comment everything that wasn't obvious. Additionally, feel free to change anything else you find along the way, I would love to see where this project goes!