HOLOLENS TEMPLATE FOR UNREAL ENGINE 4

VERSION 1.0

AUGUST 1st, 2017

TESTED ON UNREAL 4.16.2 – MIXED REALITY BRANCH

Video found at https://youtu.be/KxvAm2qNJ0Q

What to do:

On the Hololens

- Enable developer mode
- Be sure that latest version 10.0.14393.0 is installed
- Go to your router and note the ip adress of your Hololens

Compiling UE4 from source

Download latest UE4 – Mixed Reality branch from https://github.com/MICROSOFT-XBOX-ATG/MICROSOFT_UWP_UNREAL/tree/dev_MixedReality

Unzip. If using Visual Studio 2015 and Windows 10 SDK < 15063 (valid for HoloLens or Emulator):

- Run GenerateProjectFiles.bat
- Load in VS2015 by double-clicking UE4.sln

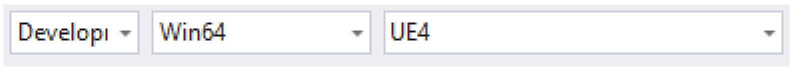
If using Visual Studio 2017 and Windows 10 SDK >= 15063 (required for Mixed Reality immersive devices and the Mixed Reality Portal):

- Run GenerateProjectFiles.bat -2017
- Load in VS2017 by double-clicking UE4.sln

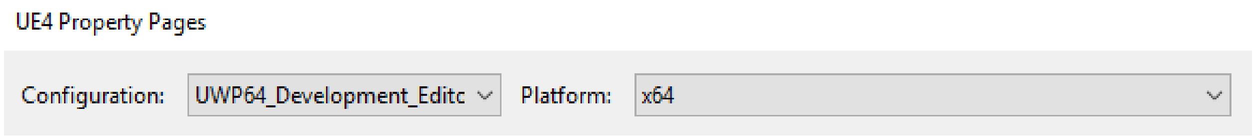
Note: The Xbox-Live SDK is not required for building Windows Mixed Reality projects. If you see an error during 'GenerateProjectFiles' about the Xbox-Live SDK missing from your PC, you can ignore it.

In Visual Studio

- Set solution configuration to Development Editor and solution platform to Win64

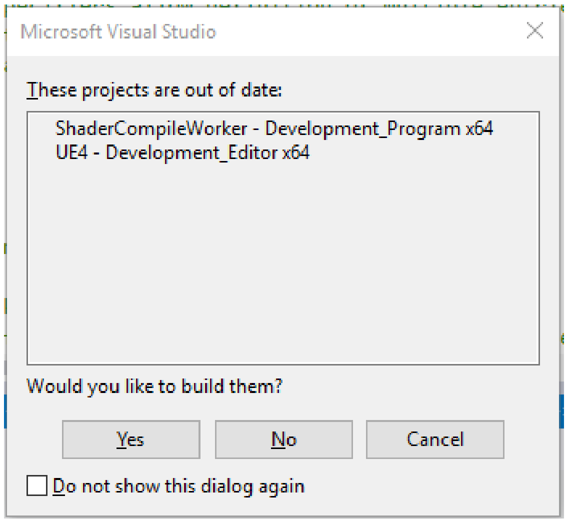

Developı | Win64 | UE4

- Expand the Engine folder under Solution Explorer
- Right-click on UE4 project and select 'Properties'
- Set Configuration to UWP64_Development_Editor and Platform to x64


UE4 Property Pages

Configuration: UWP64_Development_Editc
Platform: x64

- (Close any instance of the Unreal Editor/Launcher currently running.
- Right-click on UE4 project and select 'Build'
    - First build of the Unreal Editor/Engine can take several hours (depending on your development PC). Avoid using the 'rebuild' option for subsequent builds to speed up development.
- Verify that UE4 is set as the StartUp project.
- Launch Unreal Editor from VS by selecting 'Debug > Start without debugging' (or Ctrl+F5).
- You can select 'No' on the following prompt:


Microsoft Visual Studio

These projects are out of date:

ShaderCompileWorker - Development_Program x64
UE4 - Development_Editor x64

Would you like to build them?

- Select an existing project from the Project pane, or create a new one.
- You can also start from install dir/Engine/Binaries/Win64/UE4Editor.exe

How can I install it?

Files can be found at https://1drv.ms/f/s!Av77lIlxt2OY0XGGW8UDwykohjuT

GitHub version at https://github.com/ProteusVR/Hololens *(you need to be logged to Github to open the link)*

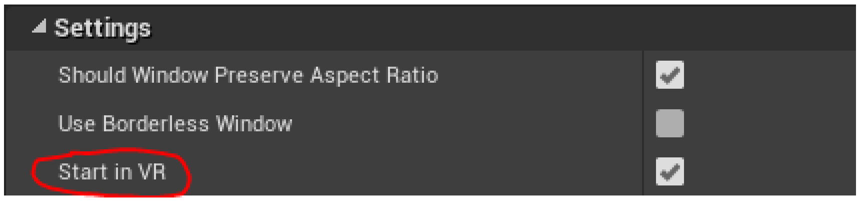
Main infos found in the forum at https://forums.unrealengine.com/showthread.php?151354-Hololens-Template

To install as a template, just unzip into the appropriate templates directory like (install dir)\Templates for launcher version. Launch a new project, and you'll find it in the blueprint section.

To open as a project file, open the project with the launcher or directly from the .uproject file.

In the template

Project – Description / Settings – check 'Start in VR'



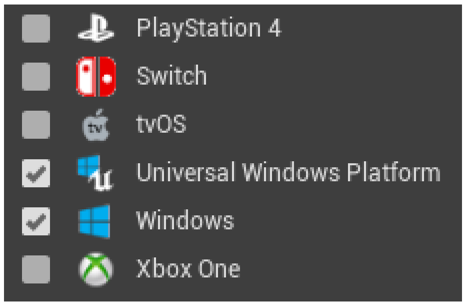
Settings	
Should Window Preserve Aspect Ratio	✓
Use Borderless Window	
Start in VR	✓

Project – Maps and Modes

Default GameMode: GMHolo

Editor/Game Map: HoloMap

Project - Update Supported Platforms



	PlayStation 4
	Switch
	tvOS
✓	Universal Windows Platform
✓	Windows
	Xbox One

- Check 'Universal Windows Platform'
- Uncheck 'Playstation 4'

Project – Target Hardware

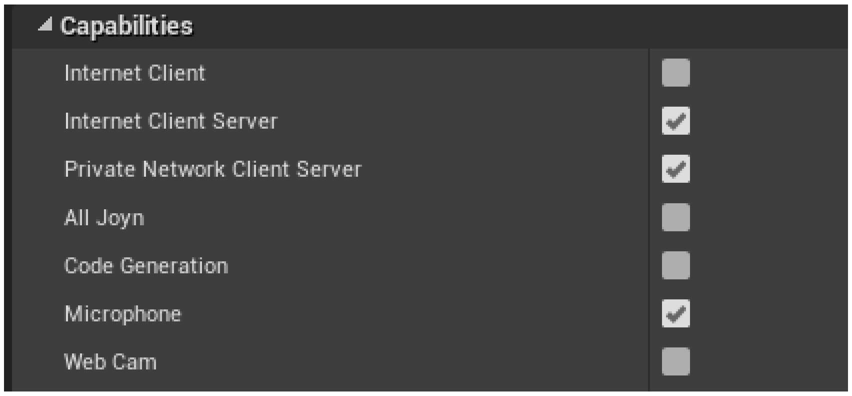
I recommend Mobile/Tablet – Maximum Quality

Project - Packaging

- Set Build Configuration to “Shipping”

Platform - UWP

- Go to the ‘Platforms – UWP’ page under the Project Settings tab.
- Set the ‘Minimum supported platform version’ to 10.0.14393.0
- Set any UWP capabilities that your application will need (ex: Speech Recognition requires the ‘Microphone’ capability, Spatial Mapping requires the ‘Spatial Perception’ capability).



Capabilities

Internet Client	☐
Internet Client Server	✓
Private Network Client Server	✓
All Joyn	☐
Code Generation	☐
Microphone	✓
Web Cam	☐

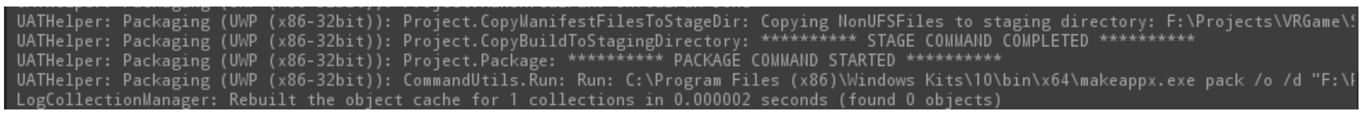
Modify Plugins

- Go to Settings > Plugins
- Under Virtual Reality, enable the ‘Microsoft Windows Mixed Reality HMD’ plugin
- (Optional) Input – enable ‘Windows Speech Recognition Plugin’, if needed
- (Optional) Input Devices – enable ‘Windows Mixed Reality Spatial Input Plugin’, if needed
- (Optional) Remove any plugins not being used (will require a restart of the Editor)

Packaging

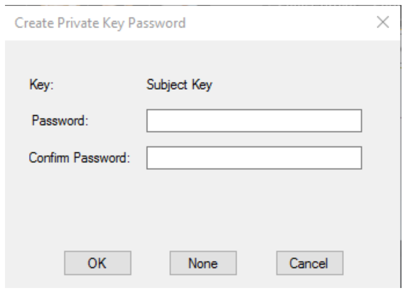
- Be sure to save all of your changes before packaging!
- File > Package Project > Universal Windows Platform > UWP (x86-32bit)

- Towards the end of the build process, makeappx.exe will be called:

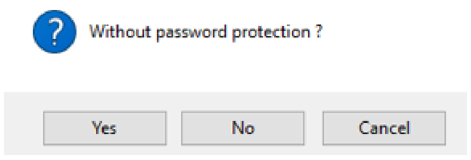


```
UATHelper: Packaging (UWP (x86-32bit)): Project.CopyManifestFilesToStageDir: Copying NonUFSFiles to staging directory: F:\Projects\VRGame\
UATHelper: Packaging (UWP (x86-32bit)): Project.CopyBuildToStagingDirectory: ********** STAGE COMMAND COMPLETED **********
UATHelper: Packaging (UWP (x86-32bit)): Project.Package: ********** PACKAGE COMMAND STARTED **********
UATHelper: Packaging (UWP (x86-32bit)): CommandUtils.Run: Run: C:\Program Files (x86)\Windows Kits\10\bin\x64\makeappx.exe pack /o /d "F:\
LogCollectionManager: Rebuilt the object cache for 1 collections in 0.000002 seconds (found 0 objects)
```

- If this is the first time that you're packaging the project, then a certificate will need to be created.
    - Look for the following prompt (it might be in your taskbar if it does not appear on screen):



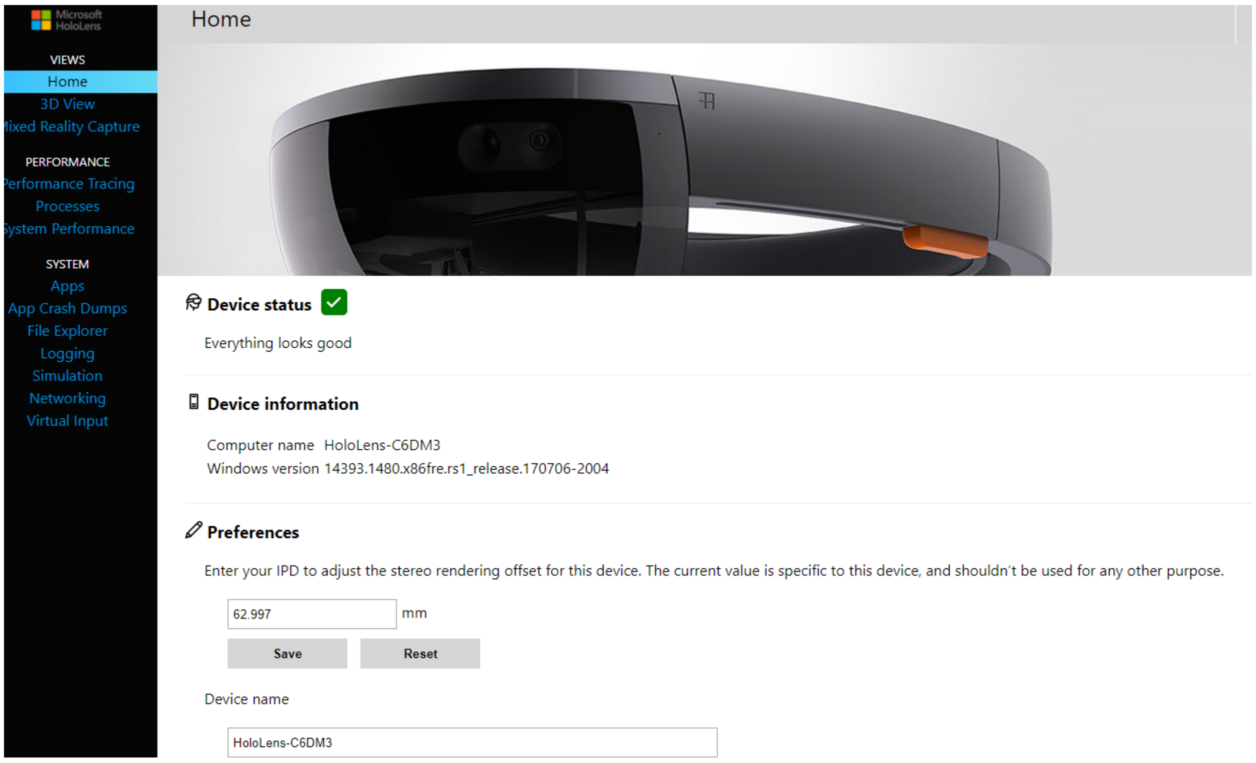
    - Leave the password fields empty and press 'OK'.
    - Select 'Yes' when you see the next prompt to create a test-signed certificate:



Deploy Package to Hololens

Deploy from Windows Device Portal

- Power on the Hololens
- Open the Hololens device portal
    - As an example, if the ip address of the hololens is 192.168.1.100, open https://192.168.1.100 in your browser. You may have permissions to give.



Home

Device status ✓

Everything looks good

Device information

Computer name HoloLens-C6DM3
Windows version 14393.1480.x86fre.rs1_release.170706-2004

Preferences

Enter your IPD to adjust the stereo rendering offset for this device. The current value is specific to this device, and shouldn't be used for any other purpose.

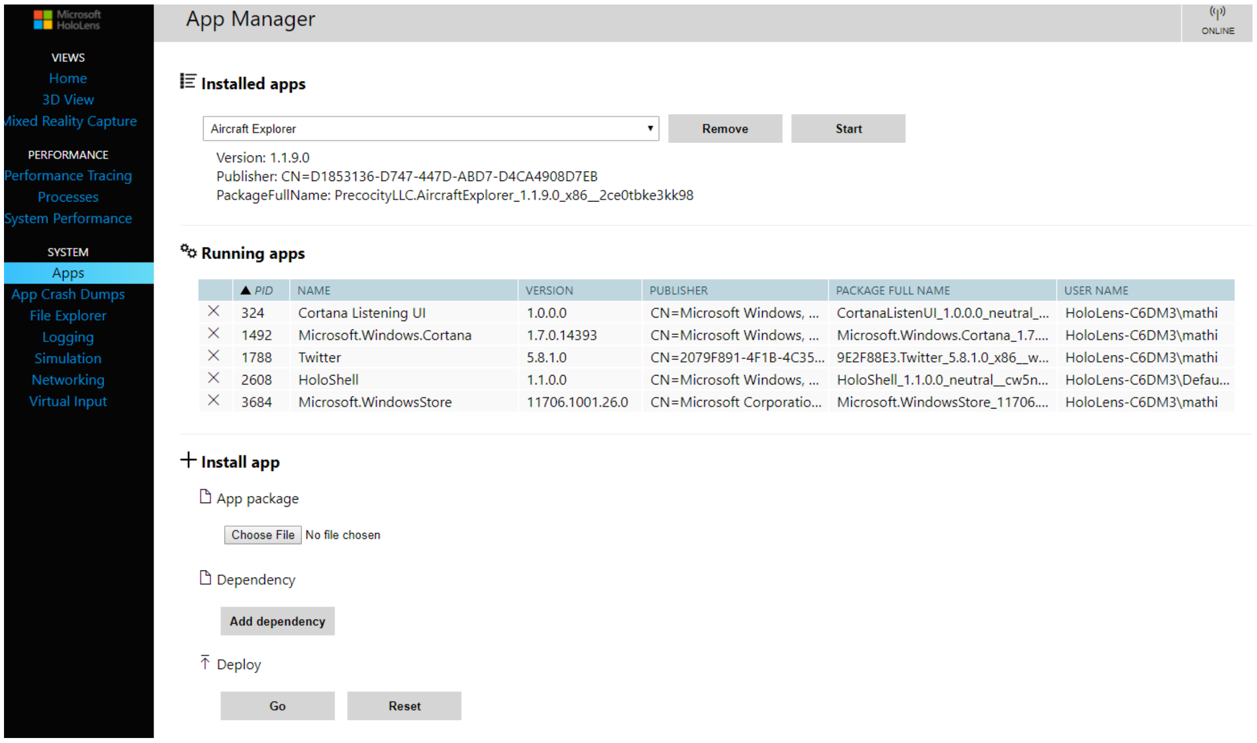
62.997 mm

Save Reset

Device name

HoloLens-C6DM3

- Navigate to the Apps page.



App Manager

ONLINE

VIEWS
Home
3D View
Mixed Reality Capture
PERFORMANCE
Performance Tracing
Processes
System Performance
SYSTEM
Apps
App Crash Dumps
File Explorer
Logging
Simulation
Networking
Virtual Input

Installed apps

Aircraft Explorer | Remove | Start

Version: 1.1.9.0
Publisher: CN=D1853136-D747-447D-ABD7-D4CA4908D7EB
PackageFullName: PrecocityLLC.AircraftExplorer_1.1.9.0_x86__2ce0tbke3kk98

Running apps

	PID	NAME	VERSION	PUBLISHER	PACKAGE FULL NAME	USER NAME
×	324	Cortana Listening UI	1.0.0.0	CN=Microsoft Windows, ...	CortanaListenUI_1.0.0.0_neutral_...	HoloLens-C6DM3\mathi
×	1492	Microsoft.Windows.Cortana	1.7.0.14393	CN=Microsoft Windows, ...	Microsoft.Windows.Cortana_1.7....	HoloLens-C6DM3\mathi
×	1788	Twitter	5.8.1.0	CN=2079F891-4F1B-4C35...	9E2F88E3.Twitter_5.8.1.0_x86__w...	HoloLens-C6DM3\mathi
×	2608	HoloShell	1.1.0.0	CN=Microsoft Windows, ...	HoloShell_1.1.0.0_neutral__cw5n...	HoloLens-C6DM3\Defau...
×	3684	Microsoft.WindowsStore	11706.1001.26.0	CN=Microsoft Corporatio...	Microsoft.WindowsStore_11706....	HoloLens-C6DM3\mathi

Install app

App package
Choose File No file chosen

Dependency
Add dependency

Deploy
Go | Reset

Click Install App, browse to your .appx file just created

- Under ‘Deploy’, select the ‘Go’ button.
- Wait for your app to upload and install on the device.

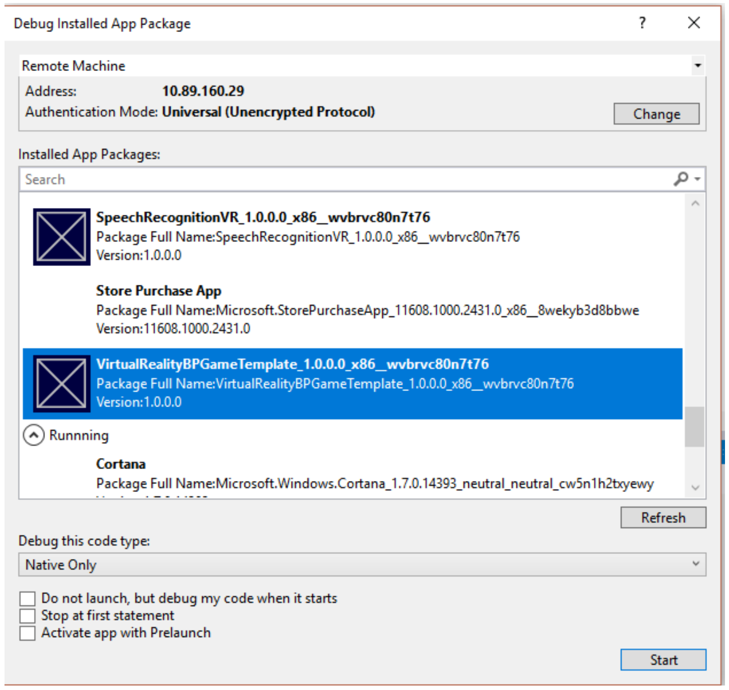
The app will be in the list of your installed apps.

Optionnaly, *you can also start it*

1)Through the app manager, by choosing the app and click start

2)Through Visual Studio in Debug Mode:

- *In VS, select ‘Debug > Other Debug Targets > Debug Installed App Package’.*
- *Change Local Machine to Remote Machine and enter the IPv4 address of your HoloLens in the Address field, then press ‘Select’.*
- *Select your app in the ‘Installed App Packages’ list.*
- *Press the ‘Start’ button and wait for the application to start on the device.*



Debug Installed App Package

Remote Machine

Address: **10.89.160.29**

Authentication Mode: **Universal (Unencrypted Protocol)**

Installed App Packages:

SpeechRecognitionVR_1.0.0.0_x86__wvbrvc80n7t76
Package Full Name:SpeechRecognitionVR_1.0.0.0_x86__wvbrvc80n7t76
Version:1.0.0.0

Store Purchase App
Package Full Name:Microsoft.StorePurchaseApp_11608.1000.2431.0_x86__8wekyb3d8bbwe
Version:11608.1000.2431.0

VirtualRealityBPGameTemplate_1.0.0.0_x86__wvbrvc80n7t76
Package Full Name:VirtualRealityBPGameTemplate_1.0.0.0_x86__wvbrvc80n7t76
Version:1.0.0.0

Running

Cortana
Package Full Name:Microsoft.Windows.Cortana_1.7.0.14393_neutral_neutral_cw5n1h2txyewy

Debug this code type: Native Only

- [ ] Do not launch, but debug my code when it starts
- [ ] Stop at first statement
- [ ] Activate app with Prelaunch

Note: Running with the debugger attached will impact performance!

Remaining Work: UE4 Windows Mixed Reality Port

May 2017

This document breaks down the know areas of work remaining for the Windows Mixed Reality port of Unreal Engine 4. Our current list of TFS items (https://analogdexdev.visualstudio.com/MICROSOFT_UWP_UNREAL/_workitems) document the initial set of expected work items.

Pri 0

Scrub

Before branching the code to the ATG repository, all components require a scrub pass.

Things to remove:

- Unhelpful comments
- Anything "odd" in comments

Things to add:

- Updated UE license header
- Algorithm comments (so that we and others understand where we are taking the code and how it works)

TODO: Update/refactor the creation of the core window creation code. This may be the thing that is preventing creating windows on the HoloLens. Note: It may be necessary to tell Windows to not kill if we exceed the splash screen time. Options include adding the functionality to bring this to the same level of functionality as the existing implementation, or figuring out how to refactor so we can just leverage the existing implementation.

TODO: Create a UFunction Library

- Move public items from the tracking code to a UFunction library.
- Include exposing new RS3 APIs (IsDisplayOpaque, etc…)
- Add Class / method comments that act as tooltips on publicly exposed items
- Note: This should be blueprintable, so need to test that it is all accessible from blue prints

In addition, classes and headers ***should*** be updated to match the UE Coding Standards document (https://docs.unrealengine.com/latest/INT/Programming/Development/CodingStandard/index.html).

Build

Build Targets and Toolsets

Currently, we have UE4 building successfully in the following configurations:

- Development Editor Win64 in Visual Studio 2015
- Development Editor Win64 in Visual Studio 2017
- Development UWP32 in Visual Studio 2015
- Development UWP64 in Visual Studio 2015
- Development UWP32 in Visual Studio 2017
- Development UWP64 in Visual Studio 2017

Build Process Flow

Currently, the only solution to building a Blueprint project is to create the package in the Unreal Editor. Packages must then be side-loaded via the Windows Device Portal (or similar). Debugging requires attaching to the process.

Building, deploying and debugging (F5, Start with Debugging) does not currently work with Blueprint only projects. This has been the state of the UWP fork since we received it from ATG.

TODO: VR Templates:

- Update the VR Template to #ifdef out the Steam specific components, so that the default templates work with Windows Mixed Reality.
- Consider adding support for MR specific features.

Application Manifests

Support for setting application manifest capabilities (uses the Microphone, etc.) is not available when building with Visual Studio 2017. Recent SDKs appear to have removed mobile platform schemas that are expected by the manifest compiler when consuming our generated manifests.

This is actively being investigated and is required to be able to indicate that an application uses specific Windows features, otherwise the feature support will be blocked.

<< This is addressed in a merge which is expected early the week of 6/5. Also, our changes are being considered for merge into the ATG branch

Head Mounted Display

The HMD plugin is up and running in limited testing. The wearer can move about the scene on HoloLens and in an immersive device (Last time we verified, it required keynote team build hacks. We think this is addressed w/ the current build, but are waiting on verification from the Keynote team).

Currently, we do not have a reliable method to ensure we correctly enter exclusive mode without sacrificing the ability to create slate applications. Launching on HoloLens, with or without the exclusive mode hack, is only 100% reliable when a debugger is present. (Note: This is the issues covered above in the create window process).

TODO: Take over the display on tracking lost and render an appropriate image/background. There is currently no tracking loss/attempt to establish indication presented to the user.

Performance

Performance is key for HMDs. We need to investigate UE4 performance settings and options that will help us to hit 60 and 90 fps. It is anticipated that we will need to enable developers to configure the performance options in their projects as well as provide sensible defaults.

No work on performance has begun beyond early discussions and some research (reading UE documentation).

Mainstream PC: We also need to hit 60 FPS on mainsteam PC Specs.

Issue: It is possible to have a perf hit when using Blueprints. There is an option for Blueprint nativization, which we need to test, as the most recent integration has moved this to 'supported'.

TODO: Review the Unreal published "Performance in VR" and figure out what we should be adopting from there.

Store publishing

Currently, there are a number of problematic items in the UWP fork which prevent the app from going to store without direct intervention/hand holding from the ATG team. You need to go into the Engine code, make changes, and rebuild. At a minimum we need to document the process. Ideally, we would honor the 'build for shipping' choice and output a valid UWP.

Blueprint

All components must be reviewed for Blueprint requirements. Everything doable from code needs to be doable from a Blueprint project.

- TODO: Input exposes some unique functionality (gesture settings management) that is unique to our platform for which custom work will be required to support in Blueprint. (This is partially implemented, but needs to be verified)
- TODO: Blueprintize UFunction library (This will capture the HMD items)
- TODO: Spatial Sound will have to be exposed (similar to how it is done in Unity)
- TODO: Spatial Stage will need to be exposed
- FUTURE: Sharing will have to be exposed when we add it
- FUTURE: Spatial Mapping will have to be exposed (P2/Hololens work)
- Expected: There are likely additional items which needs to be exposed to blueprint
- Done: Speech is already blueprintized.

Testing

To date, minimal formal testing has been performed on our UE4 work. The amount of testing has varied by component. We need to ensure that our plugins are stable, performant and work as expected as well as verify that we have not introduced regressions to any part of the code that we modified.

Specific test projects we want to create:

- Work from a blank project and MRize it
- Get an Origami equivalent project up and running against Unreal
    - As a script project

- o As a blueprint project

Pri 1

Input

Spatial Input v1 API (HoloLens gesture) is pretty much done, code wise. One exception is that we have not yet determined how to best expose the ability to start the plugin with the desired gesture support. Related to this, we have not exposed (to Blueprint) methods to query and/or change the currently supported gestures.

Some confusion is likely to come from having all possible gesture “controls” exposed to Blueprint when some will be non-functional (ex: Manipulation is not supported when Navigation is enabled).

No work has yet been done to add support for the version 2 Spatial Input APIs (Motion Controller support).

It is important, when adding support for the newer APIs, that we do **NOT** break support for HoloLens.

This module also requires a good deal of testing. Only minimal, debugger based tests have occurred to date.

FUTURE: Spatial Mapping

There has been no work started on a Spatial Mapping plugin. This feature is key when supporting HoloLens.

Planning and costing for this work is also required.

Spatial Stage (Chaperone)

There has been no work put into Spatial Stage APIs (floor, boundaries, visualization). This feature will need to be implemented as part of the HMD plugin.

Planning and costing for this work is also required.

RS3 Features and Changes

The next release of Windows (RS3) will be delivering new features (sharing API, text plane rendering, Live Cube manifests, glTF controller models) and it is anticipated to see others taking changes (we have heard mention of a cleaner way to get the rendering pipeline and some possible input enhancements).

The magnitude of this work is currently unknown (early WAG is that it will be significant). One known is that we ***must*** continue to retain HoloLens compatibility.

Pri 2

Spatial Sound (Our team will definitely own this)

Spatial Sound is essentially not started. While we do have a skeleton plugin class, there is no functional code backing it. This is due to Unreal Engine having an old version of XAudio2 (2.7) implemented.

To support Spatial Sound, XAudio2 version 2.9 will need to be implemented in the UE4 engine. This appears to take the form of a plugin-like component and likely could be based on the 2.7 implementation.

Once the updated XAudio 2 implementation is available, the Spatial Sound plugin needs a functional implementation and Blueprint exposure (per sound source) for:

- Min and max distance
- Unity gain distance
- Room model

The HRTF API that form the basis of Spatial Sound has a hard requirement for 48 kHz, 32-bit float audio data as input as well as output. We must endure that the UE4 engine can provide this format. If it cannot, we will need to implement support in the engine.

Engine
To date, we have found portions of the UWP application class that were missing or not fully implemented (ex: support for input plugins). It is expected that we are likely to encounter other unimplemented portions as we add support for Windows Mixed Reality. Performance settings are one area where there is an expectation of us needing to do engine work.

Functional Libraries
Functional libraries are a collection of static functions that are called by Blueprint code. Libraries need to be built for the HMD at a minimum.

It is expected that other plugins are likely to need functional libraries as well. Likely candidates are:

- Input
  - Expose plugin constructor that allows caller to specify gesture settings
  - Expose methods to Query and Change current gesture settings
  - Expose method to cancel outstanding gestures
- Spatial Sound
  - Expose key MS HRTF parameters; Min/Max Gain, Unity Gain Distance, Room Size

Pri 3

Speech
Speech is pretty much done, code wise.

Blueprint has been tested. Code / script testing is planned.

Editor
1) Run from editor is a key feature, and will require some significant work.
2) On Hololens (Future) Remoting from editor will need to be implemented at a later time

Deployment
To make the development cycle efficient, we need to add support for deploying and launching to a remote device (HoloLens or PC), as well as the run from editor functionality described above. The Device

Portal and it’s open source wrapper are good candidates for forming the basis of this, since UE4 has the ability to compile and package the appx file. There will be UI requirements for this as well. We have not started on any of this work.

Some investigation is warranted to determine if it would be to implement remoting within UE4 so that the editor reflects on-device behavior. Assuming the investigation returns positive results, work would be required (Engine? Editor?) to implement.

Pri 4

Documentation

At the very least, we need to document how to make your application Mixed Reality friendly; feature use, performance settings, etc.

Additionally, we are adding some functionality to UE4 that may require special considerations to use (ex: specifying gesture flags). Documentation describing how and why to use this functionality will need to be written.

Proof of concept sample

There must be at least one sample/demo application which covers the entire MR feature set and runs against in each perf environment at framerate. One option for this would be to port the existing Unity based Holographic academy document.