How to use the Multi-AR package

Introduction

Augmented reality (AR) comes on several different platforms these days. The biggest ones that are also hardware and OS-specific, are Apple AR-Kit for iOS, Google AR-Core for Android and MS Windows Mixed Reality. The goal of the Multi-AR package is to provide you an easy way to deal with the number and specifics of different AR platforms, hence concentrate on the development of your Unity scenes instead. This way you may develop each scene only once and leave the platform-specific stuff to the Multi-AR components and platform-specific interfaces.

Currently Multi-AR works with AR-Kit and AR-Core. Support for Windows Mixed Reality and other AR platforms will come later.

Setup and running

The setup of Multi-AR is pretty straight forward:

1. Download and import the Multi-AR package into new Unity project.
2. You don't need to import the AR-Kit or AR-core Unity packages. They are included in the Multi-AR package.
3. Open a demo-scene from the MultiAR/DemoScenes-folder.
4. Build and run the scene on AR-Kit or AR-core. They should work.
5. Look at the settings of the MultiARManager-component in the scene. You may change some of them and re-run the scene on the AR platform, to see the effect.
6. For initial AR-Kit setup look at 'Setup of Apple AR-Kit'-section below.
7. For initial AR-Core setup look at 'Setup of Google AR-Core' section below.

The core components

Multi-AR consists of several core components: the MultiARManager-component and interface-components for each of the supported AR platforms. Currently these are ARKitInterface and ARCoreInterface components. The core components reside in the MultiAR/CoreScripts-folder, and are components of the MultiARController-game object in the demo scenes. For your convenience, the MultiARController game object along with its components is stored as prefab in the MultiAR/CoreScripts/Prefabs-folder.

How to reuse the Multi-AR package in your Unity projects

To reuse the Multi-AR package in your own Unity project, please do as follows:

1. Import the Multi-AR package into your project and delete MultiAR/DemoScenes-folder. Or, copy the MultiAR/CoreScripts-folder and the platform-specific folders (GoogleARCore & UnityARKitPlugin) from this project to your project.
2. Drag the MultiARController-prefab from the MultiAR/CoreScripts/Prefabs-folder to your scene.
3. Make sure the settings of the MultiARManager-component are OK for your scene.
4. That's it! You are ready to build scene and run it on the supported AR-platforms.
5. Look at the API of MultiARManager. It may be invoked by calling 'MultiARManager.Instance.MethodName()'. For examples and more info, look at the script-components used in the demo scenes. They are located in the MultiAR/DemoScenes/Scripts-folder.

Setup of Apple AR-Kit

To run your project on iOS with AR-Kit, make sure you fulfil the following requirements:

1. The AR-Kit plugin requires Unity 2017.1.0 or later.
2. It requires iOS 11 or later installed on the target device.
3. It requires XCode 9 or later, with iOS SDK that includes ARKit Framework.
4. It requires iOS device that supports ARKit (i.e. iPhone 6S or later, iPad 2017 or later).
5. To build for iOS open the Build settings, select 'iOS' as Platform and press 'Switch Platform'.
6. More information regarding the AR-Kit plugin may be found on this forum: https://forum.unity3d.com/threads/arkit-support-for-ios-via-unity-arkit-plugin.474385/
7. The latest release of the AR-Kit plugin may be found at Unity asset store. Its most current version is available as BitBucket repository here: https://bitbucket.org/Unity-Technologies/unity-arkit-plugin
8. If you want to upgrade Multi-AR to the latest version of AR-Kit Unity plugin, delete the UnityARKitPlugin-folder, then import the package or replace it with the same folder from the latest AR-Kit Unity plugin.

Setup of Google AR-Core

To run your AR project on Android with AR-Core, make sure you follow these steps:

1. The AR-Core plugin requires Unity 2017.2.0 or later.
2. To build for Android devices, you need to have the Android Studio installed. Here is the download link: https://developer.android.com/studio AR-Core requires Android SDK version 7.0 Nougat (API Level 24 or higher). More information can be found here: https://developers.google.com/ar/develop/unity/getting-started#setting_up_your_development_environment
3. Make sure your Android device supports AR-Core. Here is the list of supported devices: https://developers.google.com/ar/discover/#supported_devices
4. Prepare your device: Enable 'Developer options', 'USB debugging' and install the AR-Core service. More information on where to download the AR-Core service and how to install can be found here: https://developers.google.com/ar/develop/unity/getting-started#prepare-device
5. To build for Android open the Build settings, select 'Android' as Platform and press 'Switch Platform'.
6. Press 'Player settings', then disable 'Multithreaded rendering' and under 'Other Settings' set 'Minimum API Level' to 'Android 7.0 Nougat (API level 24)', then under 'XR Settings' enable 'Tango supported'. More information can be found here: https://developers.google.com/ar/develop/unity/getting-started#configure_the_build_settings
7. More information on how to get started with AR-Core can be found here: https://developers.google.com/ar/develop/unity/getting-started The AR-Core Unity forum is here: https://forum.unity3d.com/threads/introducing-arcore-an-android-ar-sdk-for-unity.490929/
8. The latest release of the AR-Core Unity plugin may be found as Git repository here: https://github.com/google-ar/arcore-unity-sdk.git
9. If you want to upgrade Multi-AR to the latest version of AR-Core Unity plugin, delete the GoogleARCore-folder, then replace it with the same folder from the latest AR-Core Unity plugin.

Demo Scenes

The demo-scenes in the package are located in the MultiAR/DemoScenes-folder. Here are their short descriptions:

1. **ObjectSpawnerDemo** – You can see here, as in the other scenes as well, the point cloud and all detected planes. When you tap on a plane, the preconfigured prefab (Teddy bear) will be instantiated, placed and anchored to the plane. This scene utilizes the MultiARmanager-component and AR-platform interfaces to manage the AR-platform data, and the ObjectSpawner-component to instantiate and place the objects.
2. **ObjectControllerDemo** – Here you can select which object to control, i.e. show, hide or move. There are three objects in the scene – Andy, Apple & Teddy with their respective toggle buttons on the left. When you turn on the toggle, you can place and anchor the respective object to a surface, then move it over all detected surfaces. The anchor remains attached to the point of the first placement. When you turn off the toggle, the object hides and its respective world anchor is removed. This scene utilizes the MultiARmanager-component and AR-platform interfaces to manage the AR-platform data, and the ObjectController-component to select and control the objects and their world anchors.
3. **AnchorUnanchorDemo** – This scene demonstrates what happens when you to anchor or unachor an object to a world surface. The Teddy toggle on the left shows or hides the object and allows you to move it over the detected surfaces. The Chain-toggle anchors the object to its current position or removes its world anchor. This scene utilizes the MultiARmanager-component and AR-platform interfaces to manage the AR-platform data, and the ModelAnchorController-component to control the object and its world anchor.

More Information, Support and Feedback

Online Documentation: (yet to come)
Web: http://rfilkov.com
Twitter: https://twitter.com/roumenf
LinkedIn: https://www.linkedin.com/in/rfilkov