AWE

Estimote User Guide

Welcome

Welcome to the AWE Estimote User Guide. This asset will make your project able to retrieve Estimote beacons and their live telemetry into a Unity project. The asset functions as a wrapper for the Estimote SDKs for Android and iOS, and the asset is non-official. AWE is not affiliated with Estimote in any way, and this asset is for the developers to save resources coding it on their own. For more information on the asset, go to the AWE website. For more information on the Estimote beacons, go to the official website.

Note

This asset only functions with Estimote Location Beacons and possibly Estimote Proximity Beacons (not tested). This asset does not work with other types of iBeacons or Bluetooth beacons.

You need an Estimote account to create apps in the cloud.

Scanning for data only works in Unity when screen is turned on. Please design the app with this in mind.

Refer to the Estimote SDK Documentation for help.

This asset will no longer be updated for future Estimote SDK versions

Trademarks

The Estimote trademark is the property of the trademark holder.

The Bluetooth© word mark and logos are owned by the Bluetooth SIG, Inc. and any use of such marks by AWE is under license.

Requirements

- Unity 5.6.0 or higher
- Estimote Location Beacons (“Long Range Location Beacons”). Possibly works with Estimote Proximity Beacons as well but this has not been tested

- **For Android builds:**
  - Compatible Android device (Android versions 4.4+)
  - Estimote Android SDK 1.0.3

- **For iOS builds:**
  - Apple Mac
  - XCode 7.1 or higher
  - iOS Device with iOS7.0 or higher
  - Estimote iOS SDK 4.11.0

Content and Setup

In this package you will find everything you need to set up and get started working with Estimote Location Beacons in Unity. The AWE Estimote asset makes it possible to use both iOS and Android devices with your Unity Estimote projects all nicely wrapped into a single script interface.

1. Import the AWE Estimote asset into your Unity project
2. This imports a folder, “AWE Estimote” in which there are several folders.
3. In Project view, drag the Editor folder on top of the Assets folder (or drag the content of the Editor folder to your Editor folder in Assets folder).
4. Repeat this for Plugins and Resources folders.

Here follows a description of the important files in this asset:

/Assets/AWE Estimote/Scripts/AWE_Estimote.cs

This c# script contains the core functionality of communicating with the Estimote beacons and cloud. Remember to always have this file in your project. If you need to start a setup with AWE Estimote, make another script, instantiate AWE_Estimote class and call *awe_estimote.Setup();*. To access one beacon’s data, parse through the beacons list eg. awe_estimote.beacons[0].RSSI.

- **Setup()** - Setup the AWE Estimote asset, ready to scan for beacons
- **StartScan()** - Begin scanning after initialization has completed. Can be invoked automatically by changed beginScanAtStart (bool) to true
- **StopScan()** - Stop scanning for beacons.
- **XXXCallback()** - All methods ending with “callback” are called from native code to send data to Unity. Do not call these methods

/Assets/AWE Estimote/Scripts/AWE_Estimote_Beacons.cs

This class represents an Estimote Beacon with almost all live data possible. The following is a list of available data of each beacon:

- **Name** - Name of the beacon, retrieved from Estimote Cloud. Replaces temporary “Minor: xxxxx” name when retrieved
- **MACAddress** - MAC Address of this beacon
- **ProximityUUID** - UUID of the beacon
- **Major** - Major ID of the Beacon
- **Minor** - Minor ID of the Beacon
- **RSSI** - Relative Signal Strength Indicator of the Beacon
- **MeasuredPower** - Measured power of the Beacon at 1 meter's distance. Only for Android, on iOS this will be 0
- **CalculatedStrength** - Calculated strength of the Beacon by subtracting RSSI from Measure Power. Only for Android, on iOS this will be 0
- **Distance** - Calculated distance to the beacon in meters, algorithm by Estimote
- **Color** - Color of beacon
- **BattefyLifeInDays** - Battery life expectancy in days

- **Accelerometer** - Current accelerometer output in a 3D vector
- **Pressure** - Current air pressure in pascal, as of Estimote SDK 0.13.0 (Android) and 4.11.0 (iOS) air pressure output doesn't work
- **Light** - Current ambient light level in lux
- **Temperature** - Current temperature in Celcius

/Assets/AWE Estimote/Scripts/TestScene/AWE_Estimote_Example.cs

This class is an example of how to handle the AWE Estimote asset and has everything already set up to start a connection with Estimote beacons and cloud, and receive all available data from the beacons. You can modify this file if you like and use it as a template.

- **ToggleScanning()** - Start and stop scanning
- **ClearBeaconsList()** - Clear the UI scroll view list of beacons and remove the beacons

/Assets/AWE Estimote/Scripts/TestScene/AWE_Estimote_Beacon_Handler.cs

This class handles UI updates in the list view and updates the data shown for each beacon. This class is a part of the example only, and you don't have to use this class in your own projects.

/Assets/AWE Estimote/Scripts/TestScene/OrderScrollContent.cs

This class orders the beacons in the UI list by distance, so those closest are on top of the list. This class is a part of the example only, and you don't have to use this class in your own projects.

/Assets/AWE Estimote/Materials/

This folder contains all materials used for the demo scene. You may use the materials, but make sure the Apache license follows your use.

/Assets/Resources/awe-logo

AWE logo for the custom editor

/Assets/AWE Estimote/Prefabs/Beacon

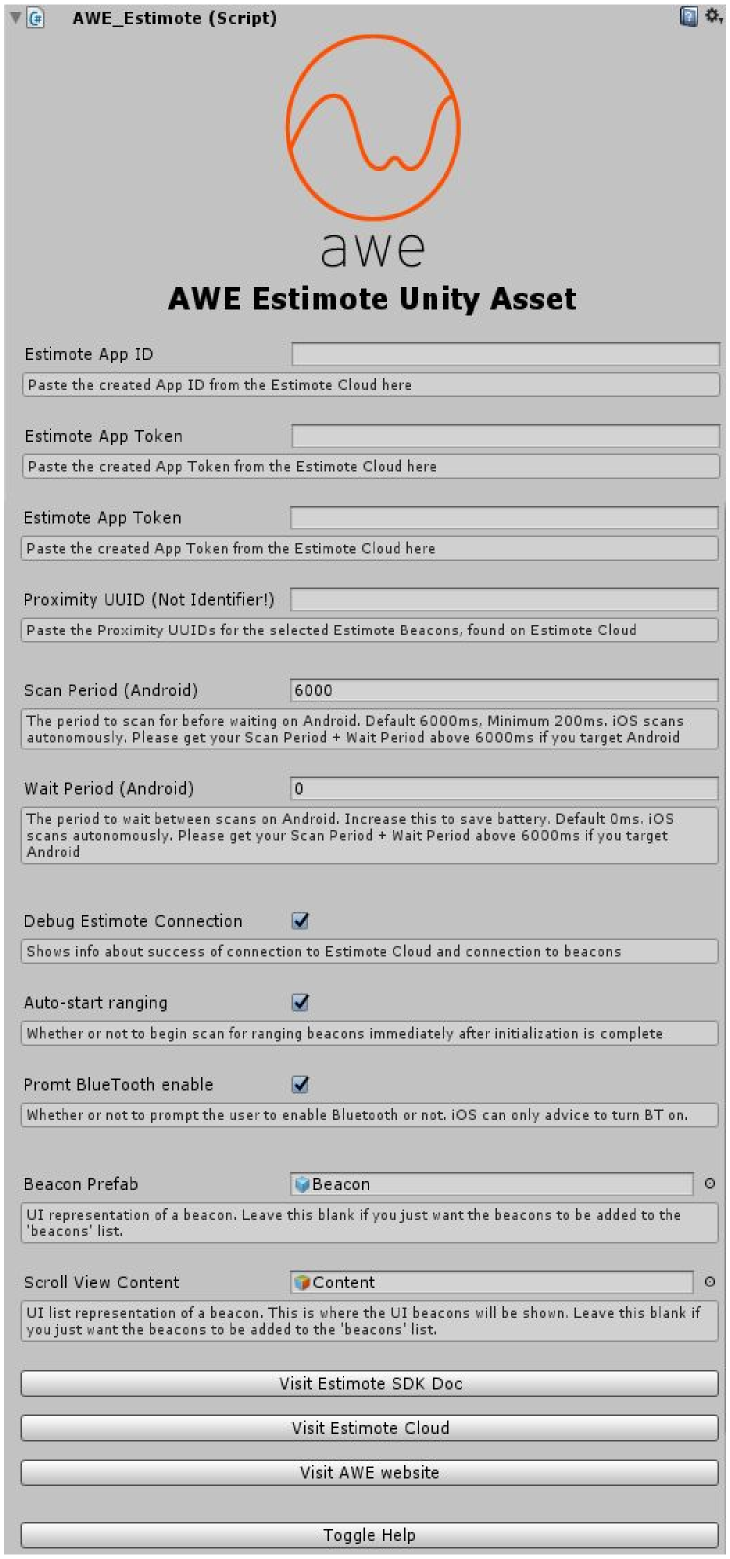
UI representation of a beacon. Used in AWE_Estimote.cs.

/Assets/AWE Estimote/Scene/EstimoteTestScene.unity

This demo scene runs the AWE_Estimote.cs file and displays all data from each found Estimote beacon. You can modify this scene and use it as a template if you like.

/Assets/Editor/AWE_Estimote_Editor.cs

This editor script generates a custom editor for AWE_Estimote.cs. It looks like this:



AWE_Estimote (Script)

awe

AWE Estimote Unity Asset

Estimote App ID
Paste the created App ID from the Estimote Cloud here

Estimote App Token
Paste the created App Token from the Estimote Cloud here

Estimote App Token
Paste the created App Token from the Estimote Cloud here

Proximity UUID (Not Identifier!)
Paste the Proximity UUIDs for the selected Estimote Beacons, found on Estimote Cloud

Scan Period (Android) 6000
The period to scan for before waiting on Android. Default 6000ms, Minimum 200ms. iOS scans autonomously. Please get your Scan Period + Wait Period above 6000ms if you target Android

Wait Period (Android) 0
The period to wait between scans on Android. Increase this to save battery. Default 0ms. iOS scans autonomously. Please get your Scan Period + Wait Period above 6000ms if you target Android

Debug Estimote Connection ✓
Shows info about success of connection to Estimote Cloud and connection to beacons

Auto-start ranging ✓
Whether or not to begin scan for ranging beacons immediately after initialization is complete

Promt BlueTooth enable ✓
Whether or not to prompt the user to enable Bluetooth or not. iOS can only advice to turn BT on.

Beacon Prefab Beacon
UI representation of a beacon. Leave this blank if you just want the beacons to be added to the 'beacons' list.

Scroll View Content Content
UI list representation of a beacon. This is where the UI beacons will be shown. Leave this blank if you just want the beacons to be added to the 'beacons' list.

Guide for building for Android

1. Get the Estimote SDK files needed for Android. Download the version 1.0.3

2. Open the zip file, and then enter the Android-SDK-1.0.3\EstimoteSDK folder

3. In the EstimoteSDK folder extract the *estimote-sdk.aar* file to the */Assets/AWE Estimote/Plugins/Android folder.*

4. Now your Android setup is ready and you can build the app and scan for Estimote beacons with an Android device.

Guide for building for iOS in Xcode

1. Export the Unity project
Build the project for iOS. Save (or “Replace”) from Unity to Xcode - When the Xcode has not previously been setup - then open up the XCode project.

2. Get the Estimote SDK files needed for iOS. Download the Estimote iOS SDK 4.11.0.

3. Press the “Clone or Download” button and then “Download Zip”

4. Implement the plugin
Open the zip file downloaded. Locate the IHS.Framework inside the folders iOS-SDK-master -> EstimoteSDK.

5. Extract the “EstimoteSDK.framework” into the xcode project folder first and place it in the root folder (often the folder is called “Xcode”). IMPORTANT: When handling the download and moving of the framework, always do this with OSX - not Windows, as Windows apparently alters the framework into a regular folder.

6. Add the newly placed Estimote framework into Xcode under “Frameworks”. Drag the framework from the Xcode folder into the Xcode application and drop it in the Frameworks group.

7. Make sure to “Add to target”, check your project (not the tests though) and “Copy items if needed”.

8. Within the XCode project make sure the following frameworks are present under General -> “Linked Frameworks and Libraries” - if not add them

- CoreLocation.framework
- CoreBluetooth.framework
- EstimoteSDK.framework

9. In the info.plist add "NSLocationAlwaysUsageDescription" as a row and add the string a la: "This app requires Location services to provide information on Estimote beacons".

> Also in the info.plist add "NSBluetoothPeripheralUsageDescription" with a description a la: "This app requires Bluetooth services to provide information on Estimote beacons"

10. Possibly add Developer Provision Profile. Also go to Build settings -> Code Signing and change all versions to your team name

11. Disable Bitcode in Build Settings

12. Do a total clean After this: Cmd + Alt + Shift + K and press Clean

13. Press Cmd + R, to run the Unity project on the iOS device.

For specific build settings refer to the Estimote SDK documentation.

"Append" from Unity to Xcode - When the Xcode has already been set up

1. Delete the reference to the Estimote Framework in the Project Navigator sidebar

2. Add the Estimote Framework folder again, by locating it in the project's Xcode folder. Drag it into the Framework group folder in the Project Navigator. A popup window appears, in which you must make sure the "Add to targets" has the project selected and "Copy items if needed" is checked.

3. Hit the Play icon, or Cmd+R to build and run the app.

AWE Estimote Asset Changelog

- Version 1.0
  - First release

- Version 1.1
  - Added checks to scan for if beaconPrefab has been added to AWE_Estimote script
  - Added check and visual clarification to let users know that Proximity UUID is not the Identifier
  - Changed default scan period to 6000 ms

Support and help

Contact AWE for questions or support at egil@theawe.dk