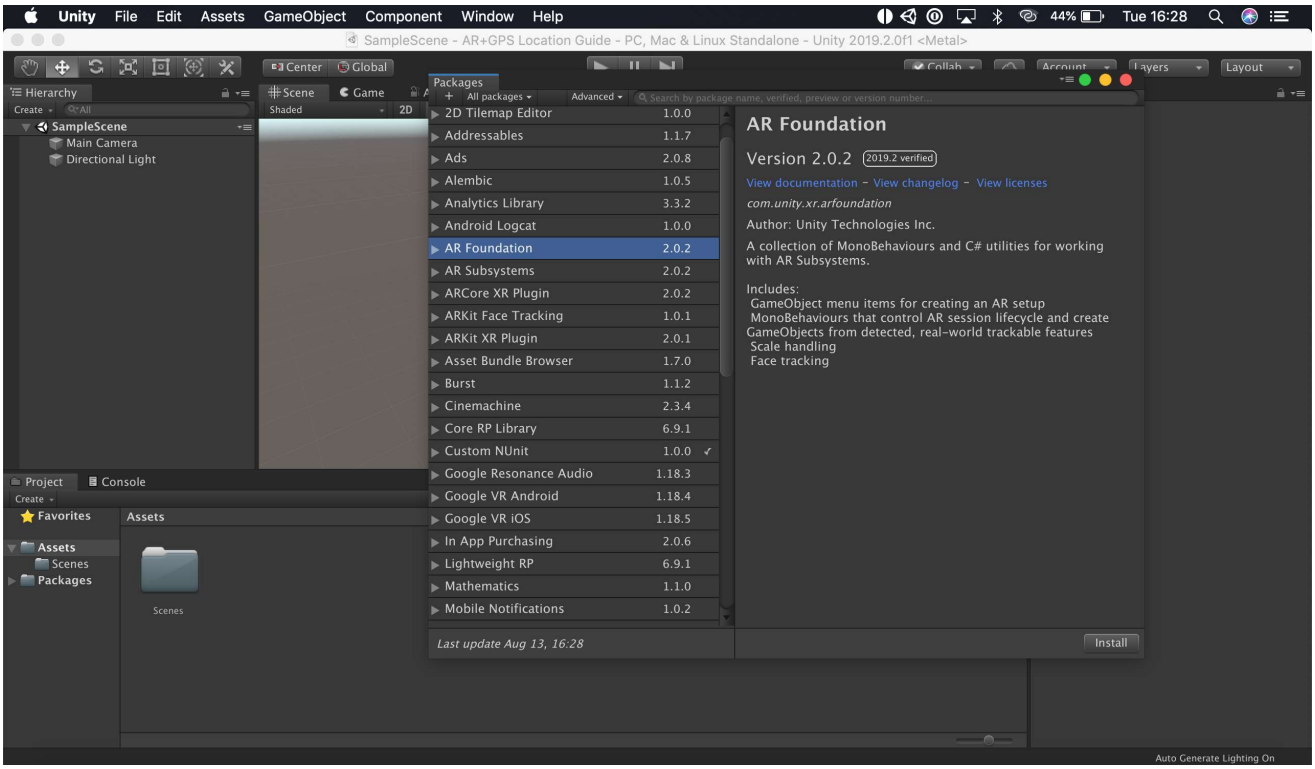
Guide

Quickstart

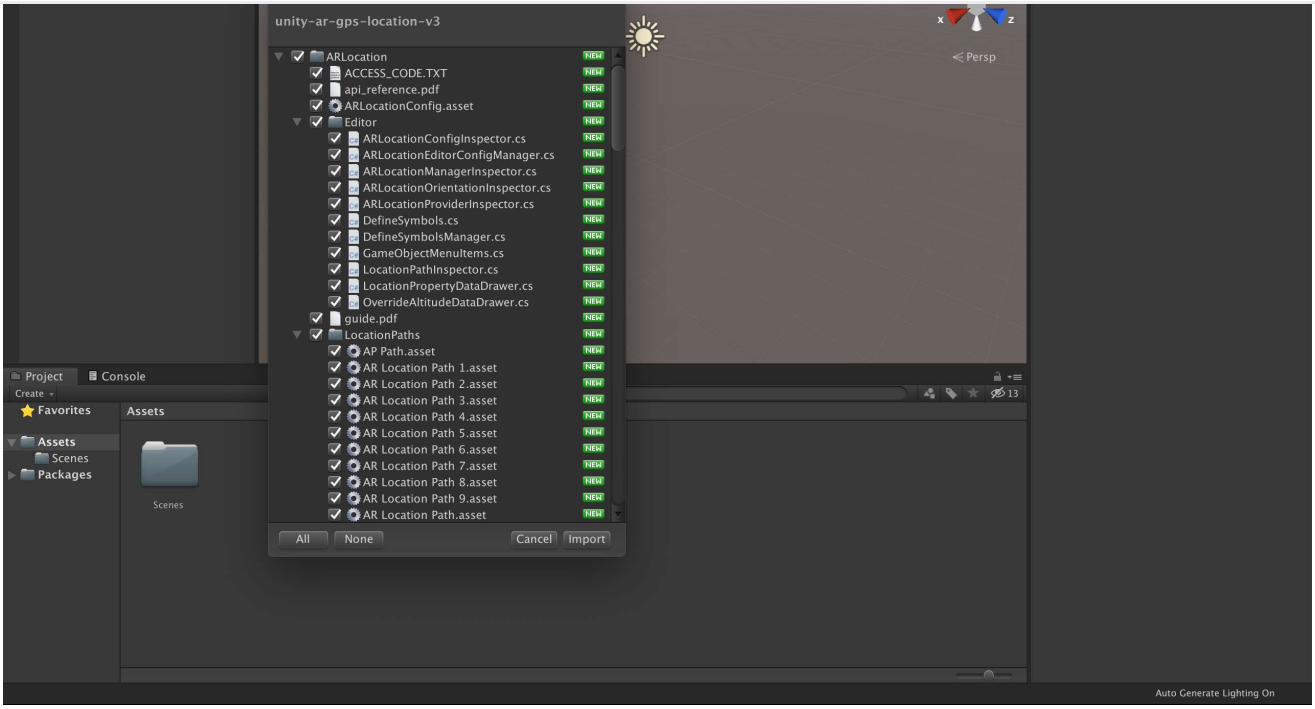
AR Foundation

Either starting from an empty project or an existing one, open the `Package Manager` window by clicking on `Window -> Package Manager` and install the `AR Foundation` package, as well as the `AR Core XR Plugin` and/or the `AR Kit XR Plugin`, depending on which platforms you are targeting.



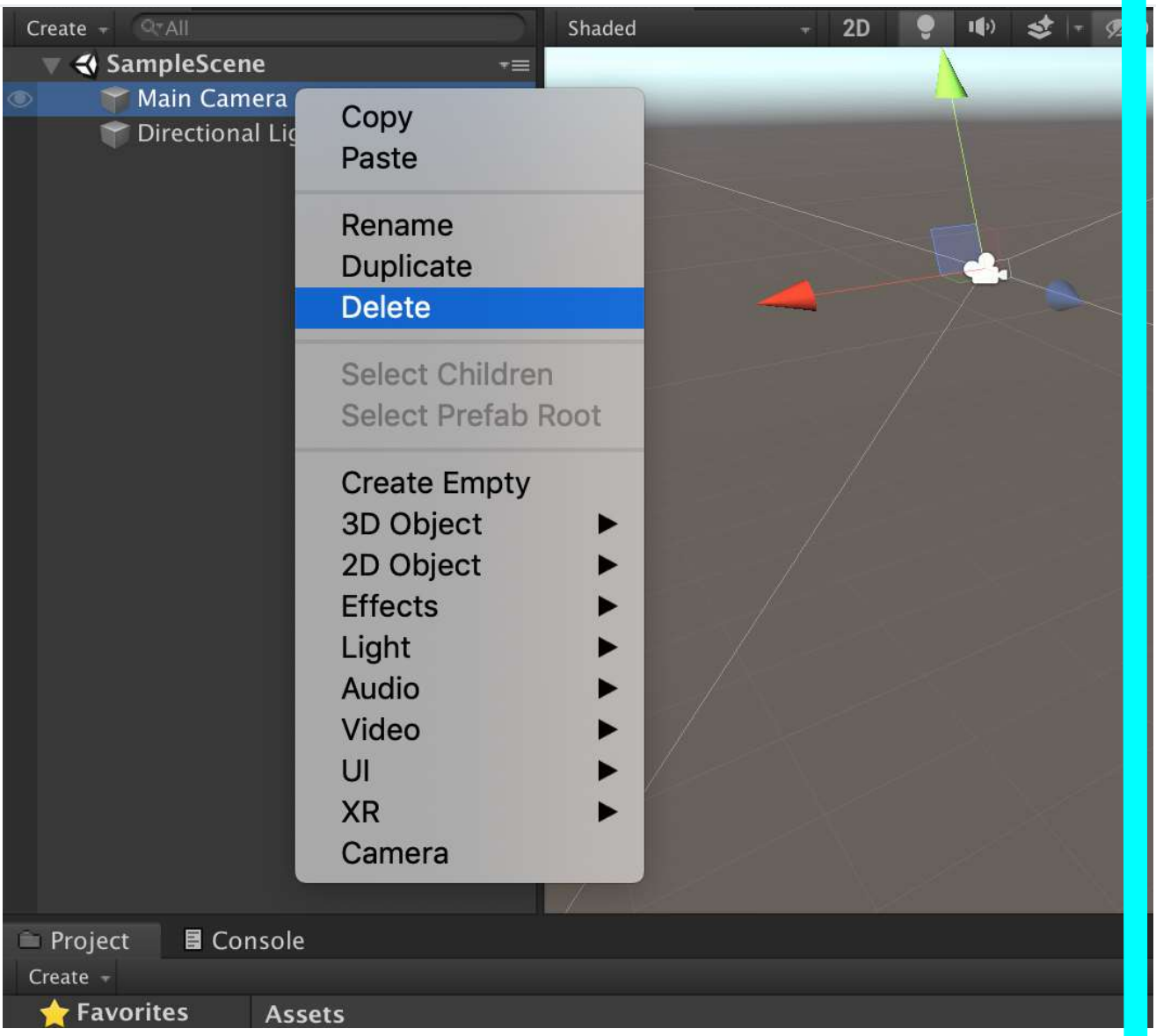
Next, import the `Unity AR+GPS Location` package, either from the asset store, or from a `.unitypackage` you have downloaded from our github repository.

Unity AR+GPS Location Docs (v3.0+)



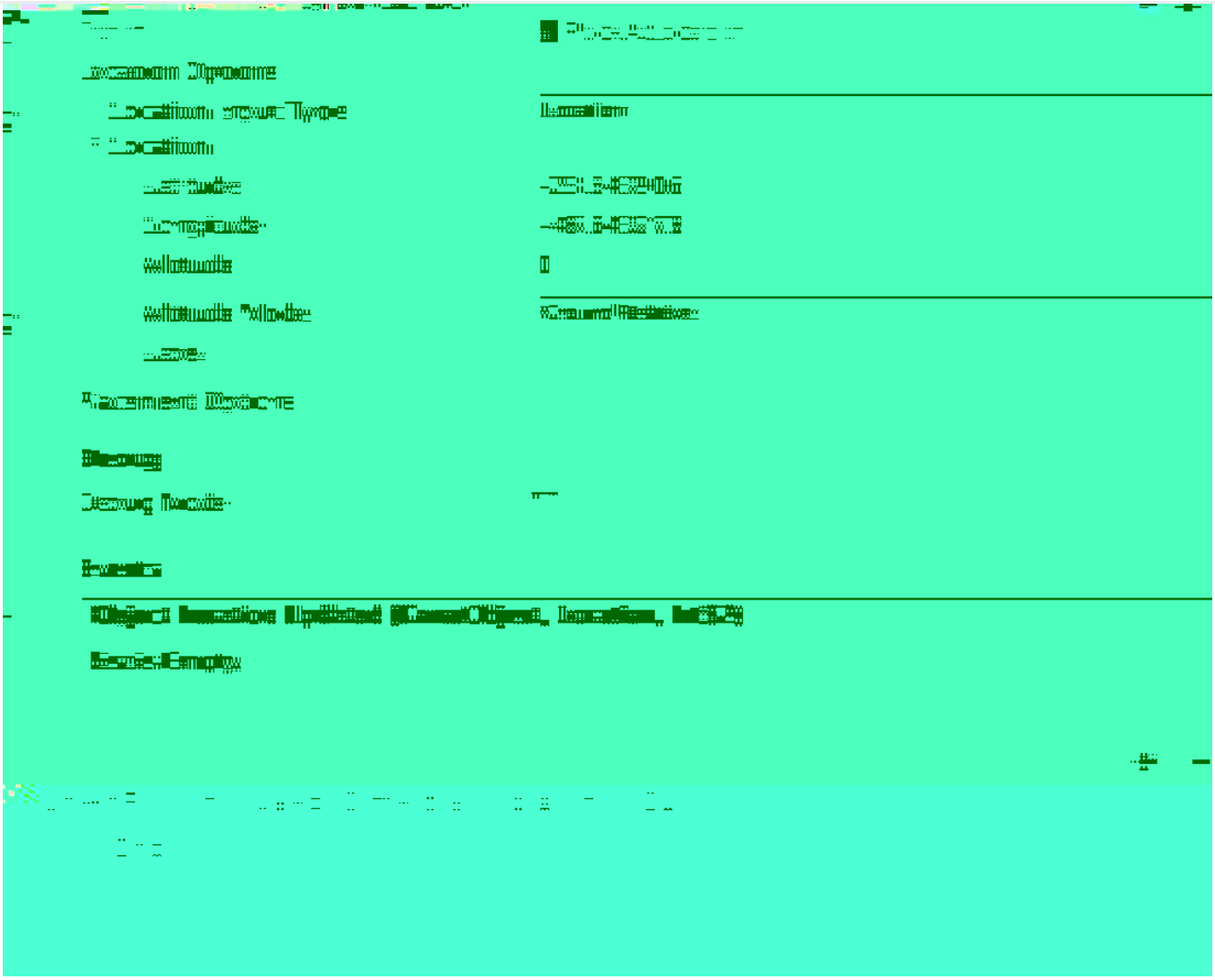
Then, remove the default 'Main Camera' from the scene.

Unity AR+GPS Location Docs (v3.0+)



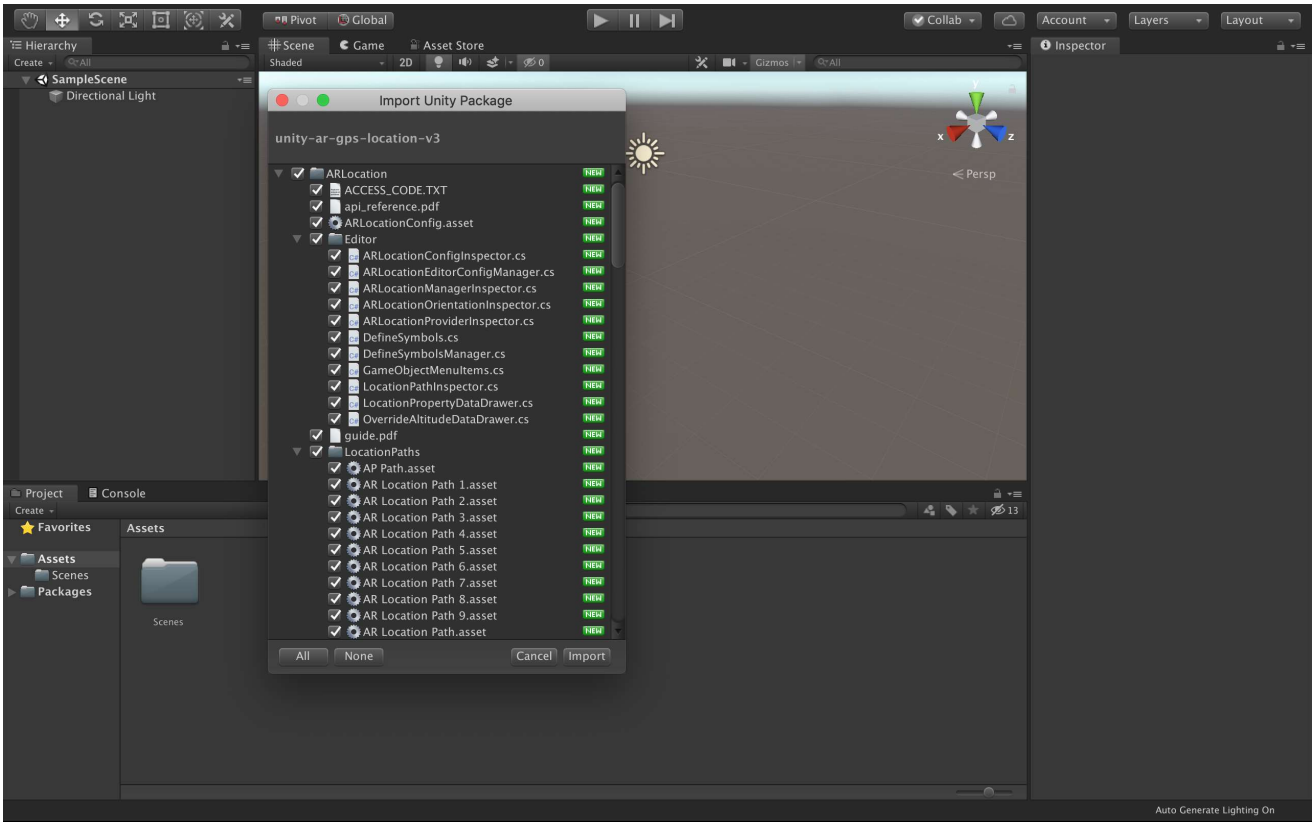
Right-click on the `Hierarchy` and go to `AR+GPS -> Create Basic Scene Structure` .

Unity AR+GPS Location Docs (v3.0+)

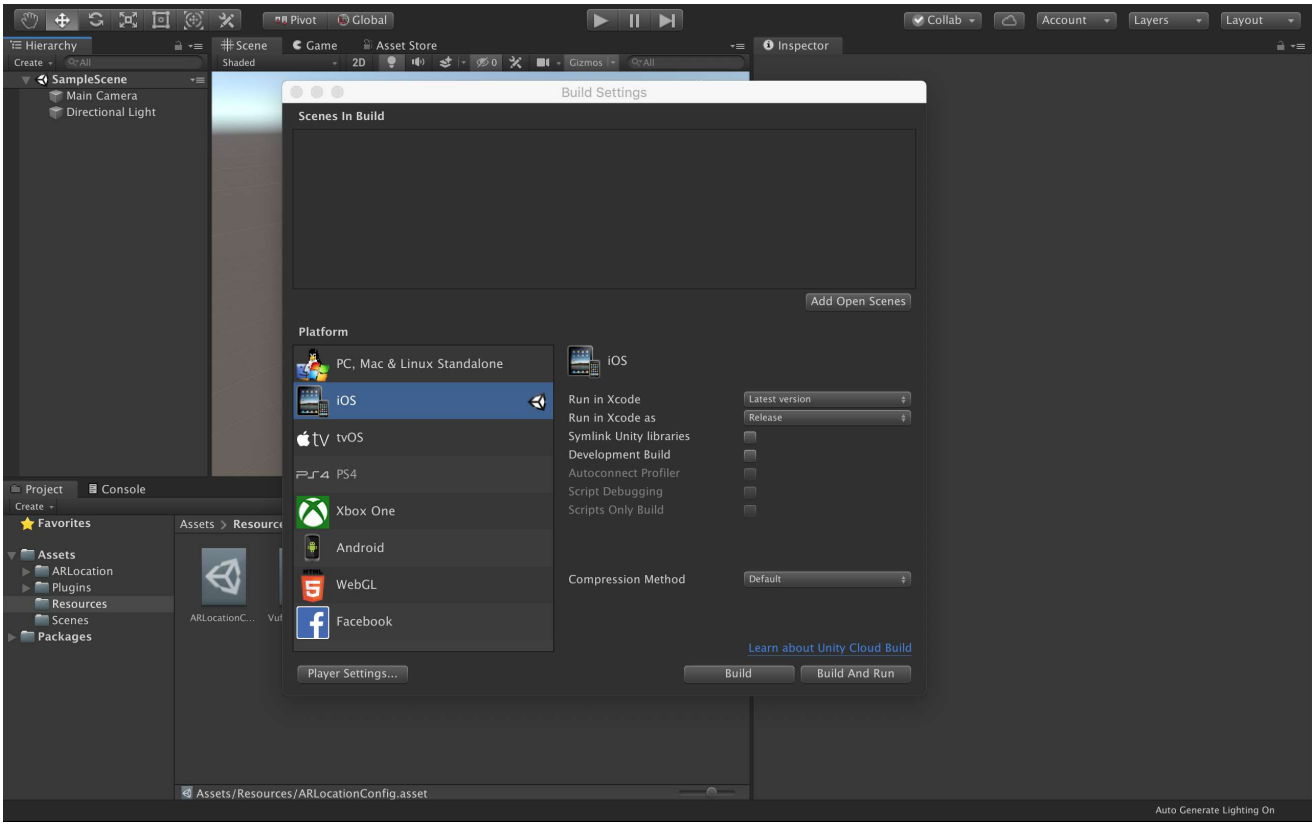


That's about it! Now you just need to build the project. On iOS just remember to set the `Camera Usage Description` and `Location Usage Description` strings on the Player Settings, and set the `Achitecture` to `ARM64` .

Unity AR+GPS Location Docs (v3.0+)

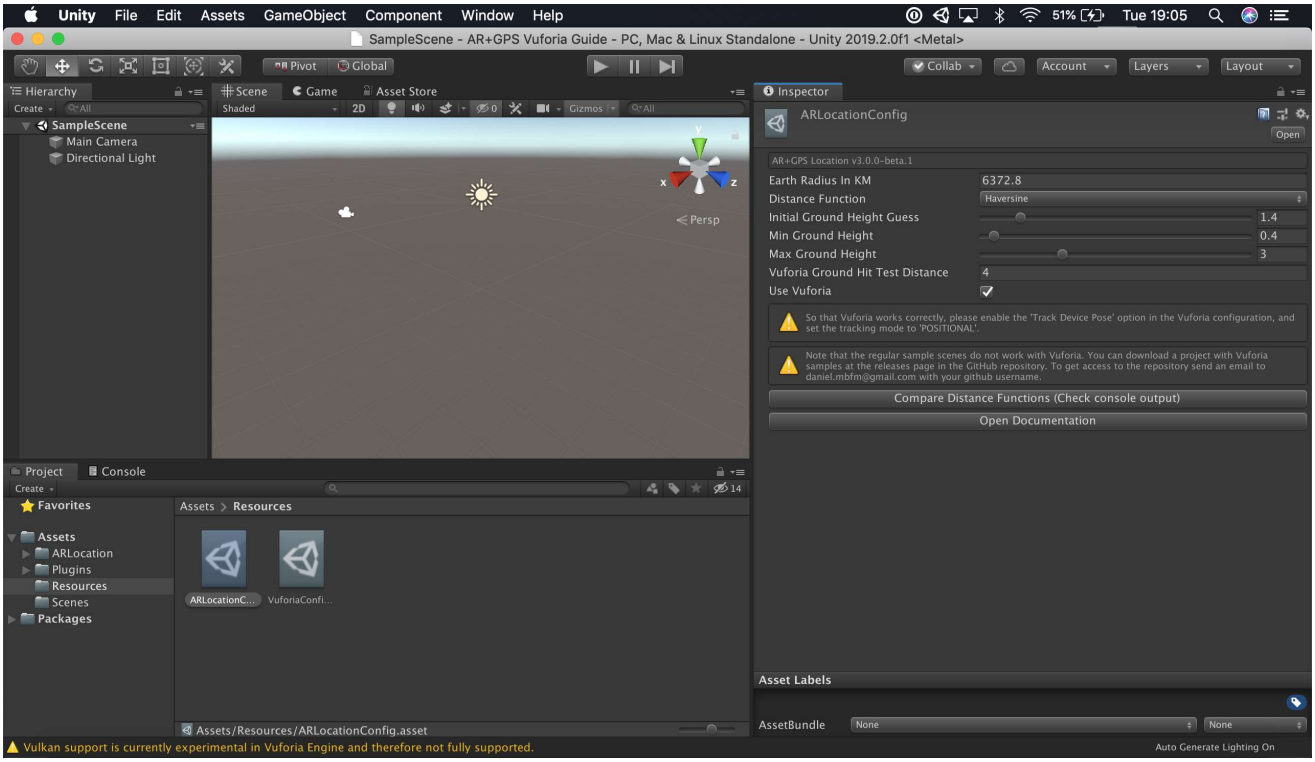


Make sure the current platform is set to either `Android` or `iOS` in the `Build Settings` window.



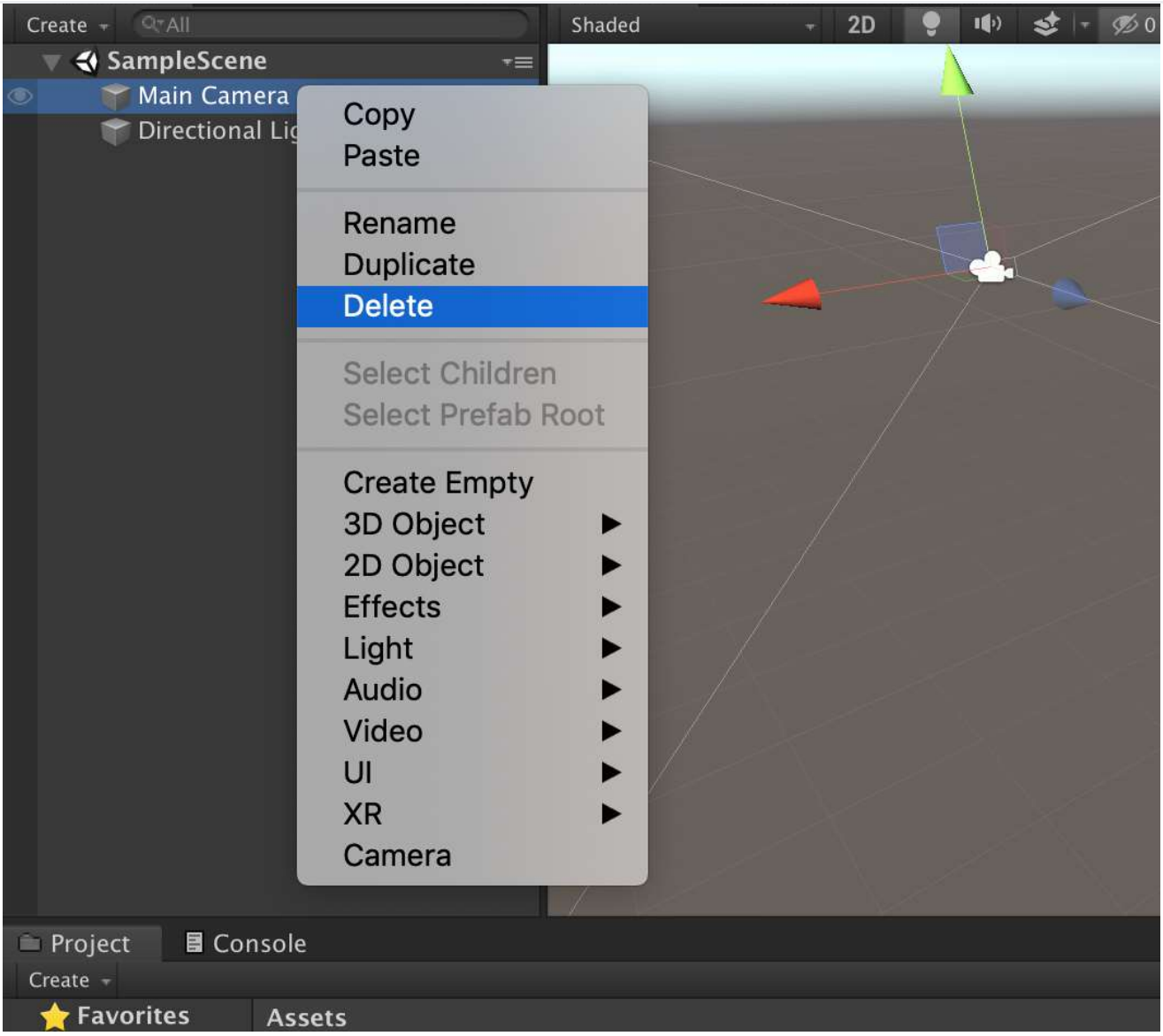
Unity AR+GPS Location Docs (v3.0+)

the project.



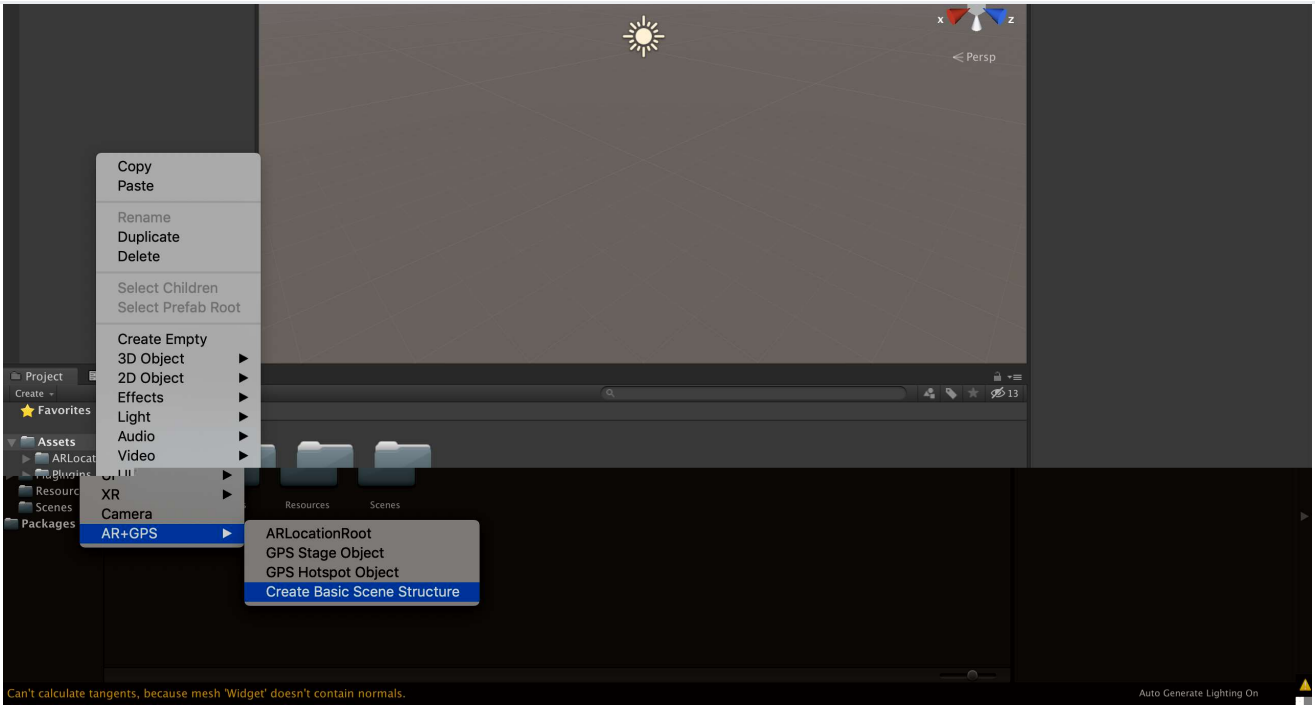
Then, remove the default 'Main Camera' from the scene.

Unity AR+GPS Location Docs (v3.0+)

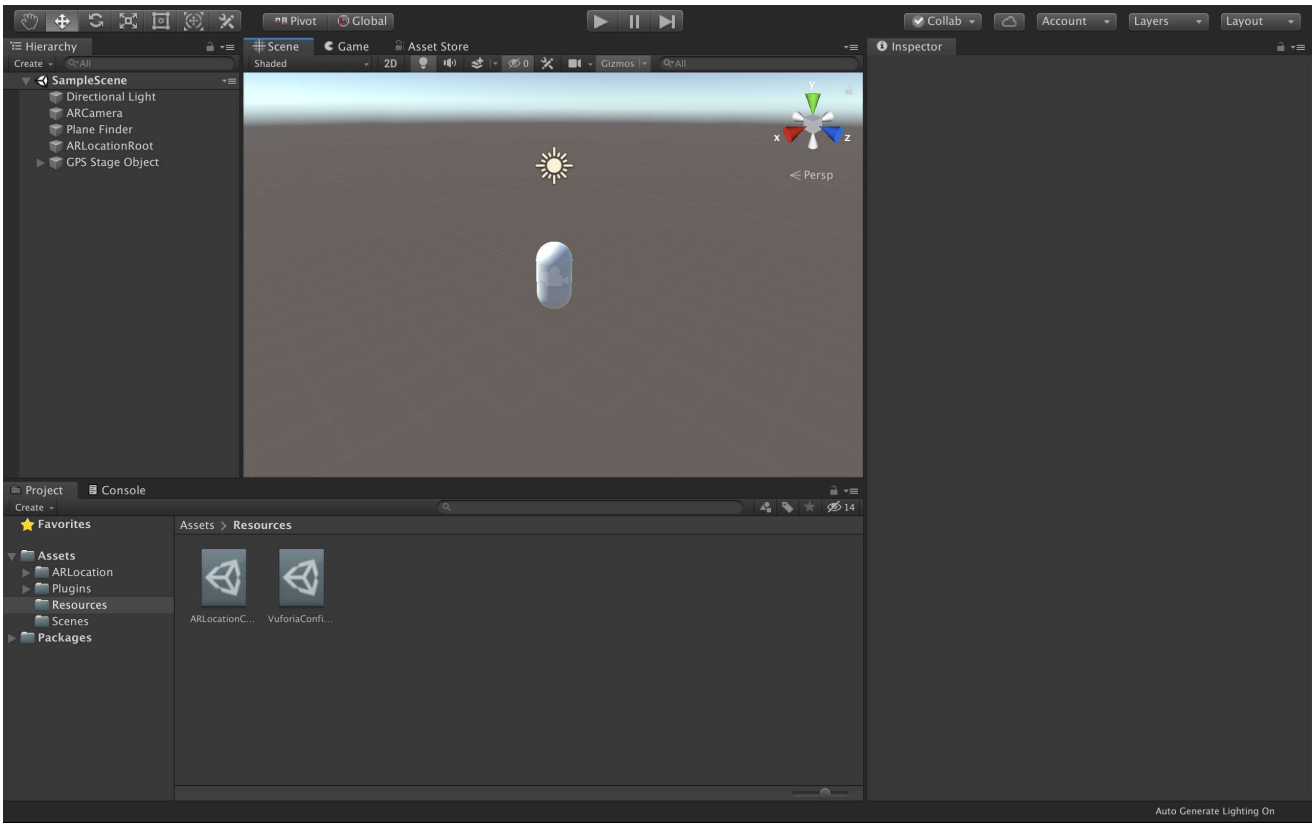


Right-click on the `Hierarchy` and go to `AR+GPS -> Create Basic Scene Structure .`

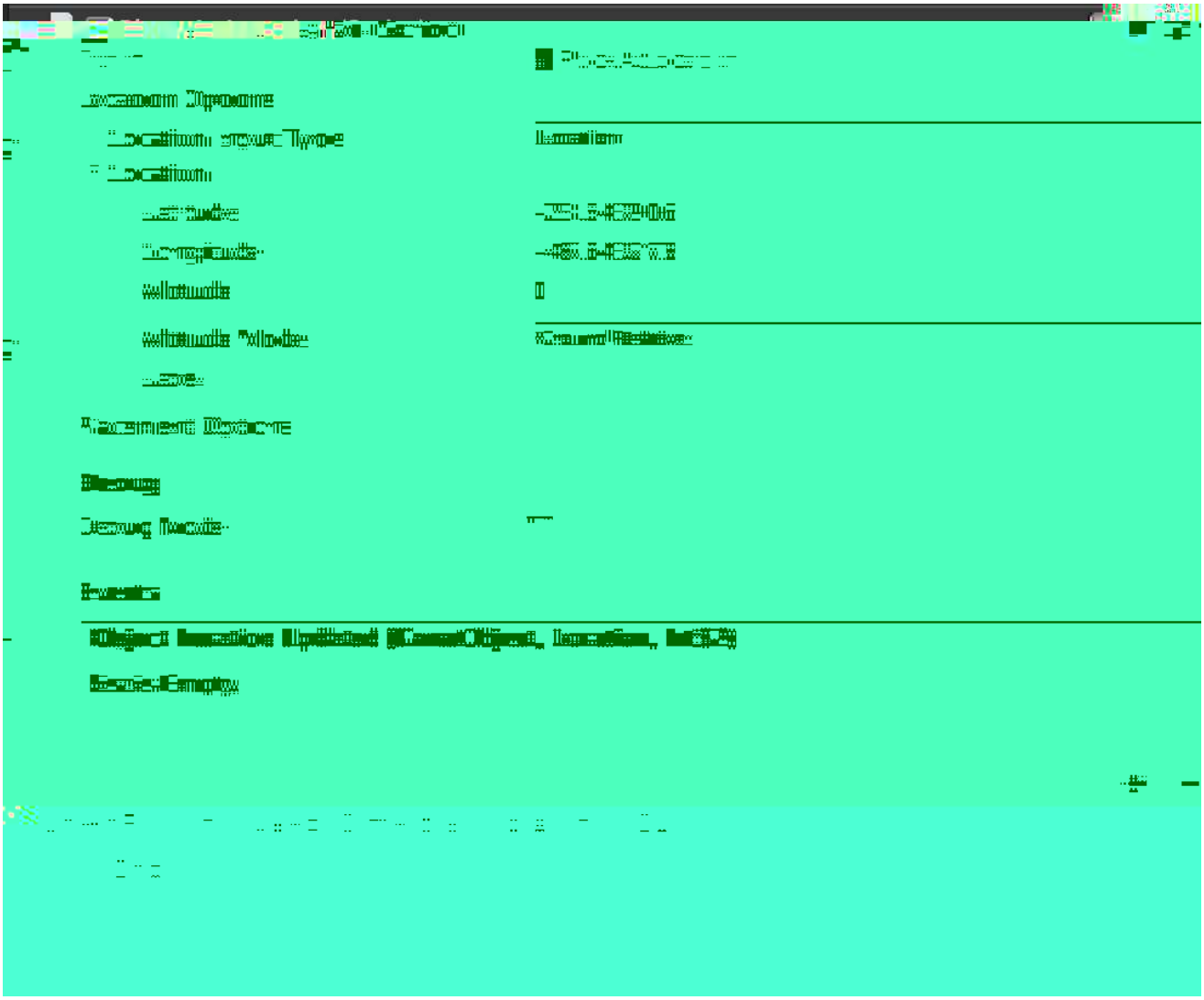
Unity AR+GPS Location Docs (v3.0+)



This will automatically populate the scene with all the Game Objects and Components for a basic AR+GPS experience!

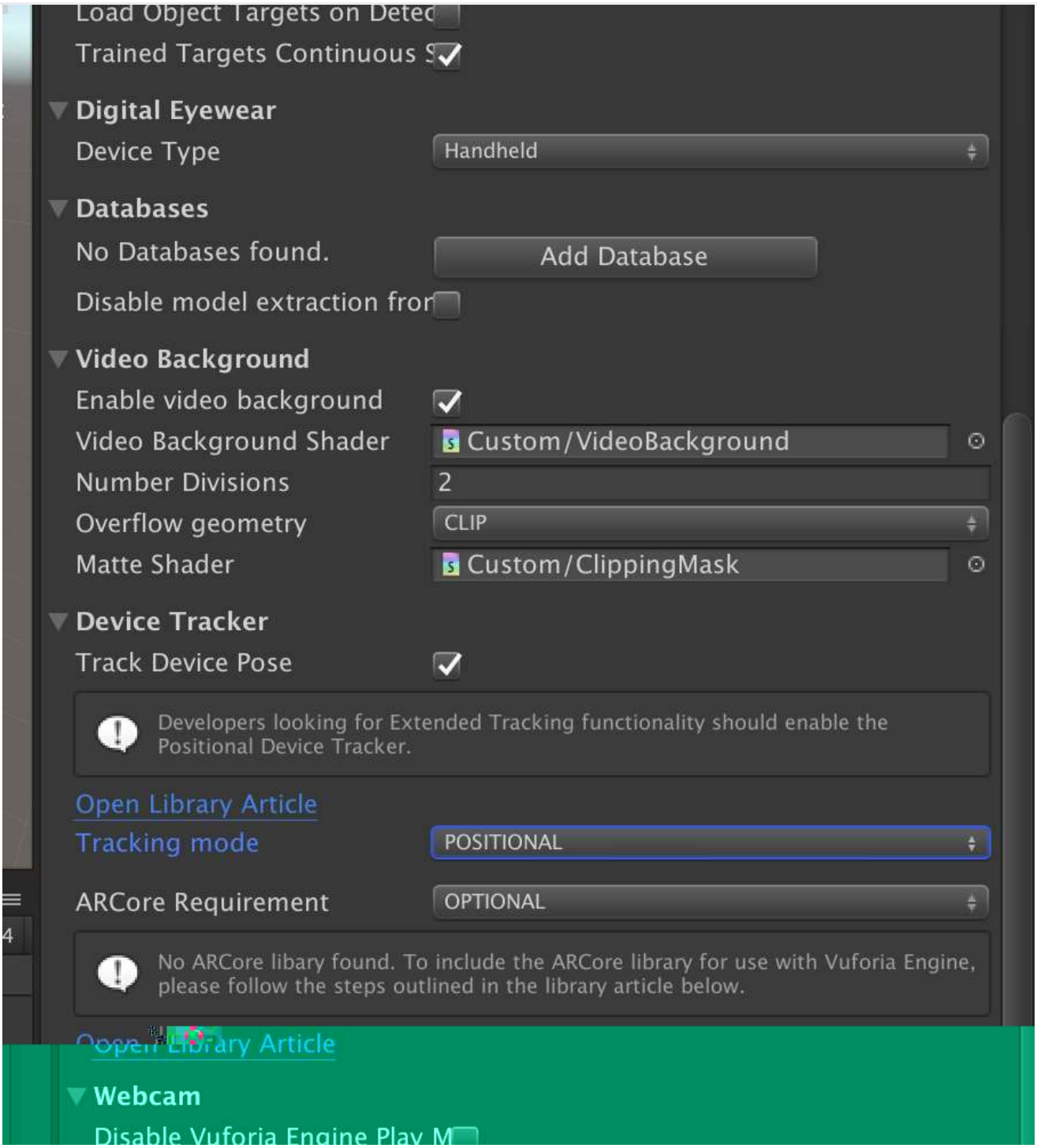


Now just click on the `GPS Stage Object` and on the `Inspector` window, go into `Location Options -> Location`, and put the geographical coordinates of the location you want the object to appear at.



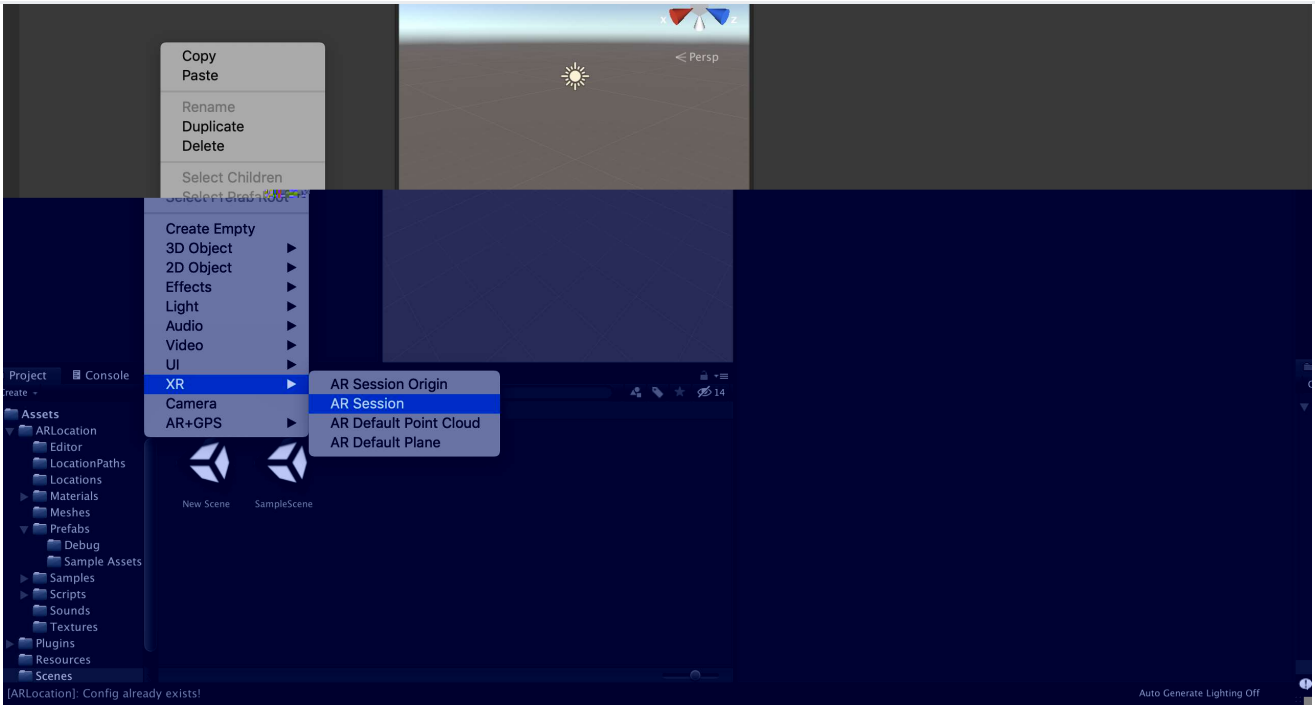
Go to Res

Unity AR+GPS Location Docs (v3.0+)



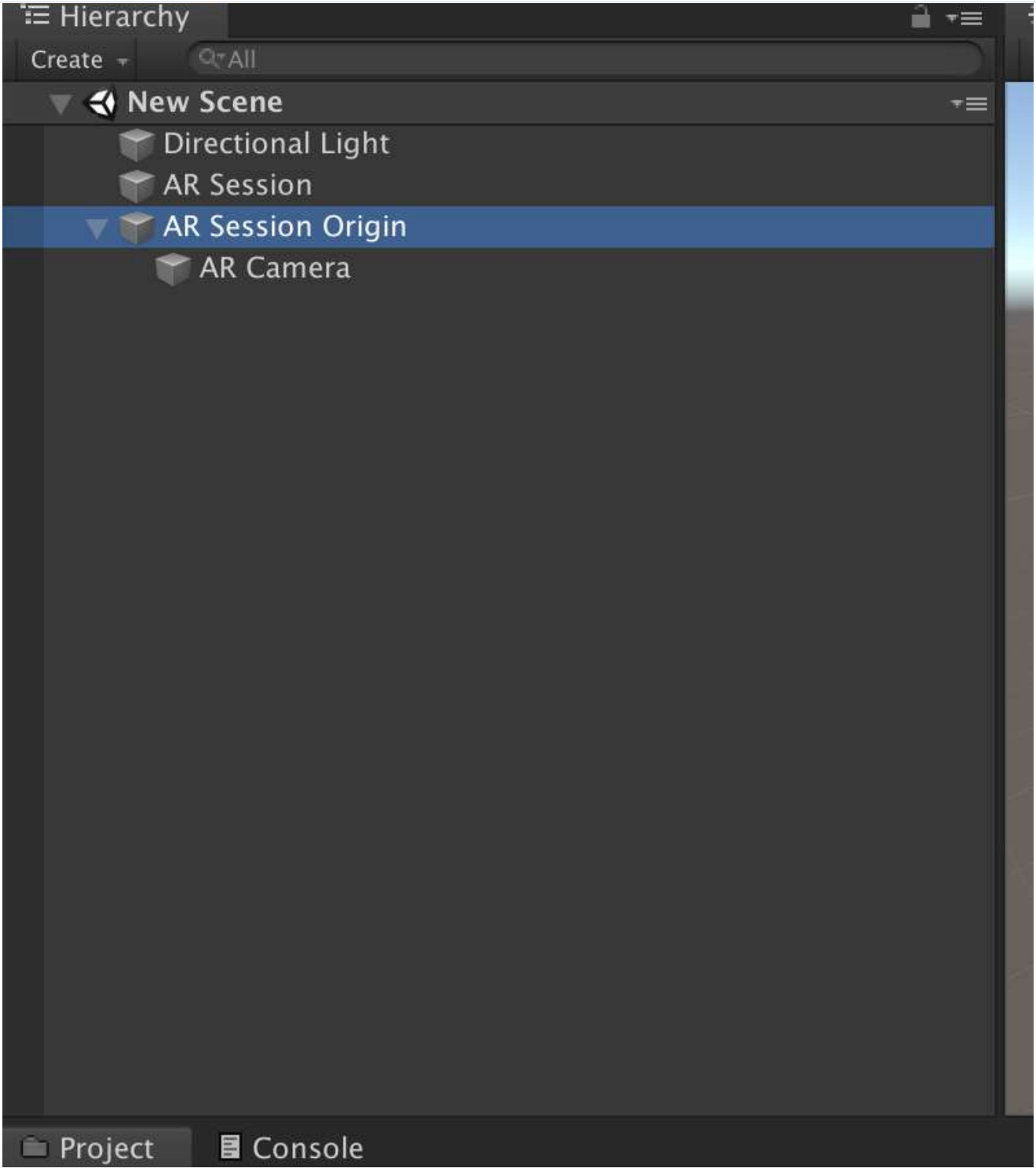
That's about it! Now you just need to build the project. On iOS just remember to set the `Camera Usage Description` and `Location Usage Description` strings on the Player Settings.

Unity AR+GPS Location Docs (v3.0+)



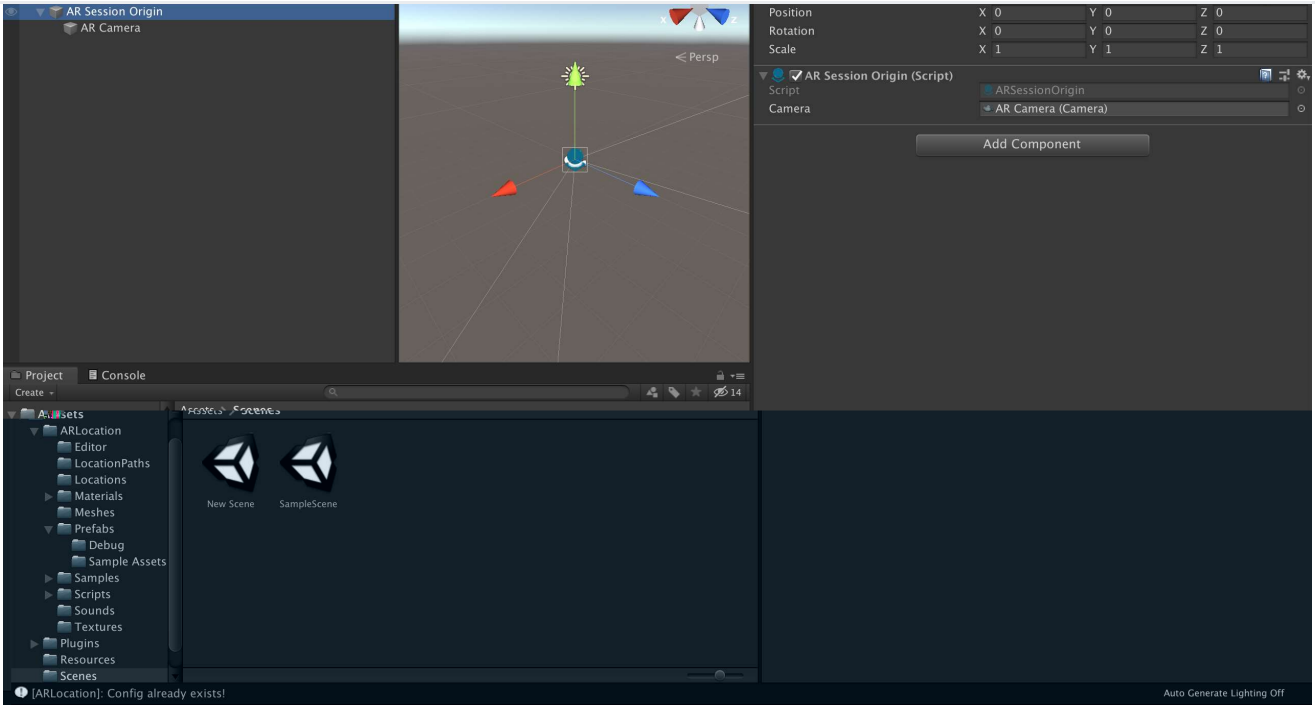
Resulting in the following scene structure:

Unity AR+GPS Location Docs (v3.0+)

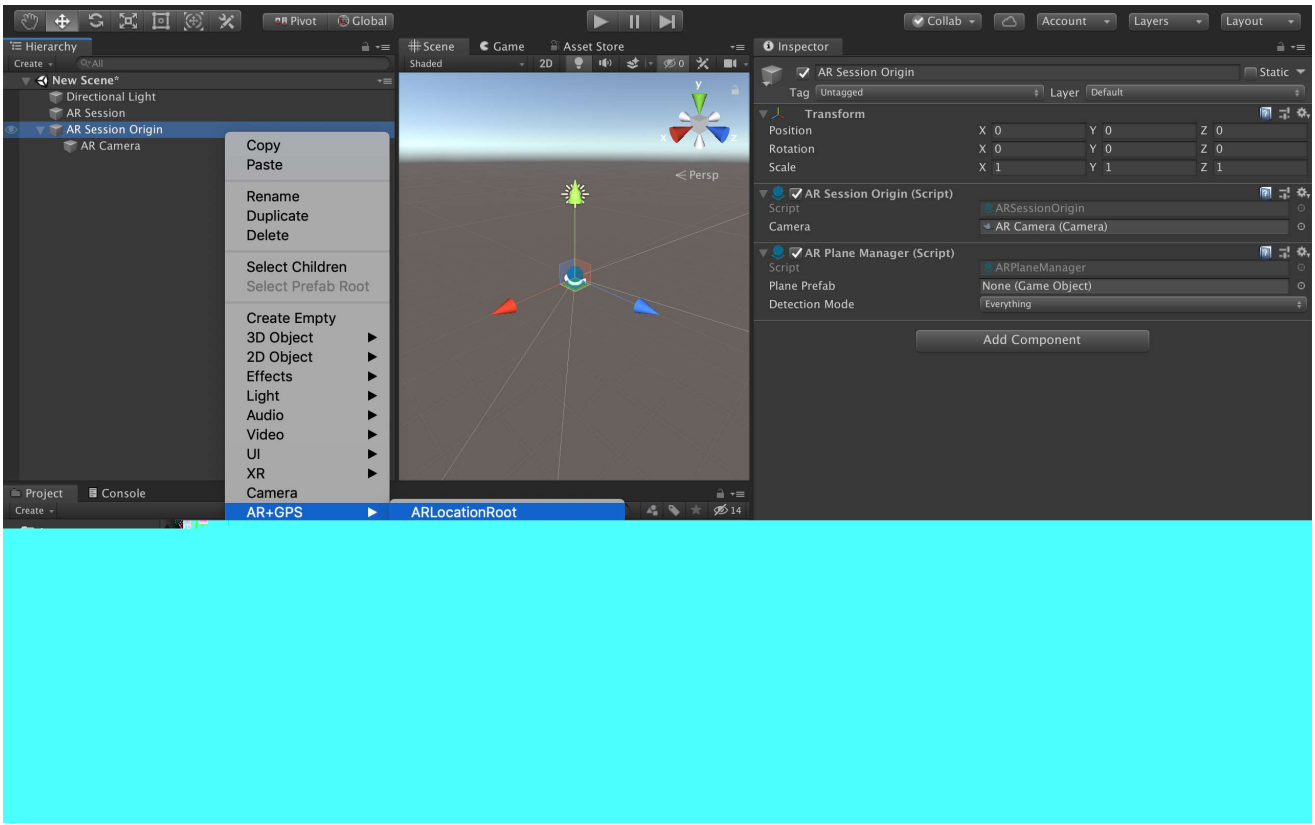


Next, click on the `AR Session Origin` object, and add a `ARPlaneManager` component to it.

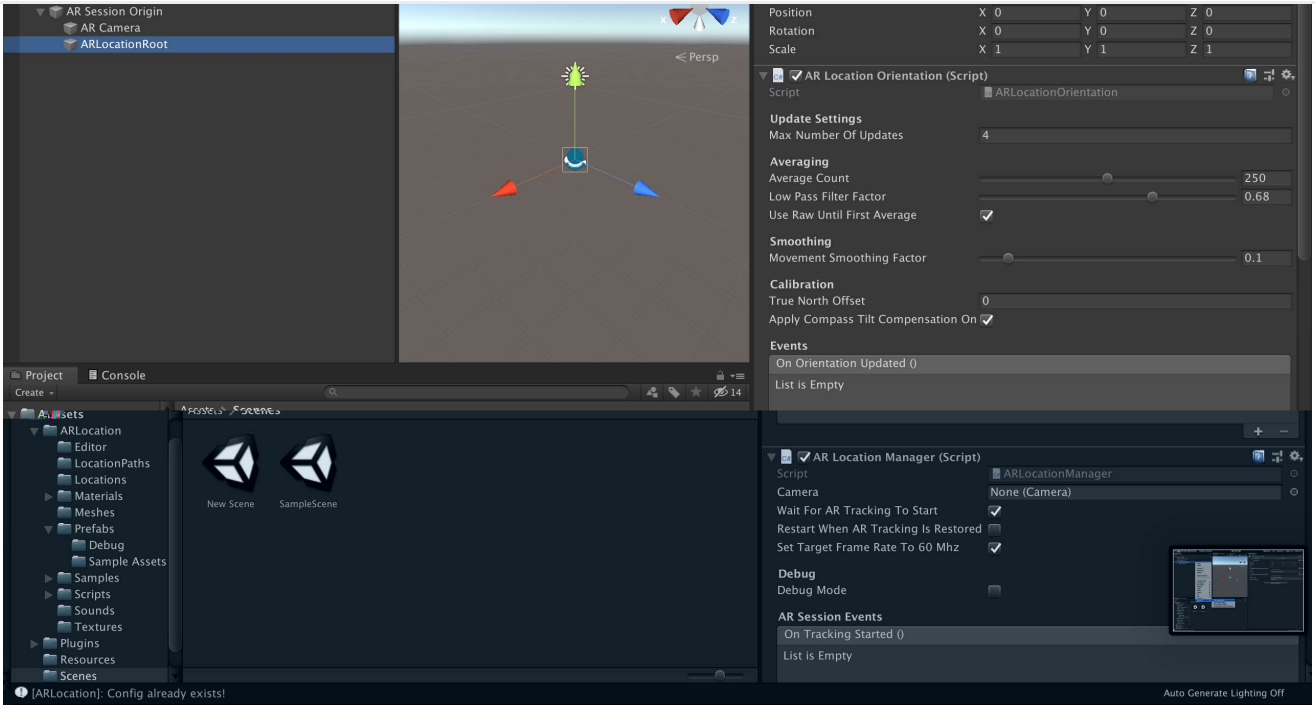
Unity AR+GPS Location Docs (v3.0+)



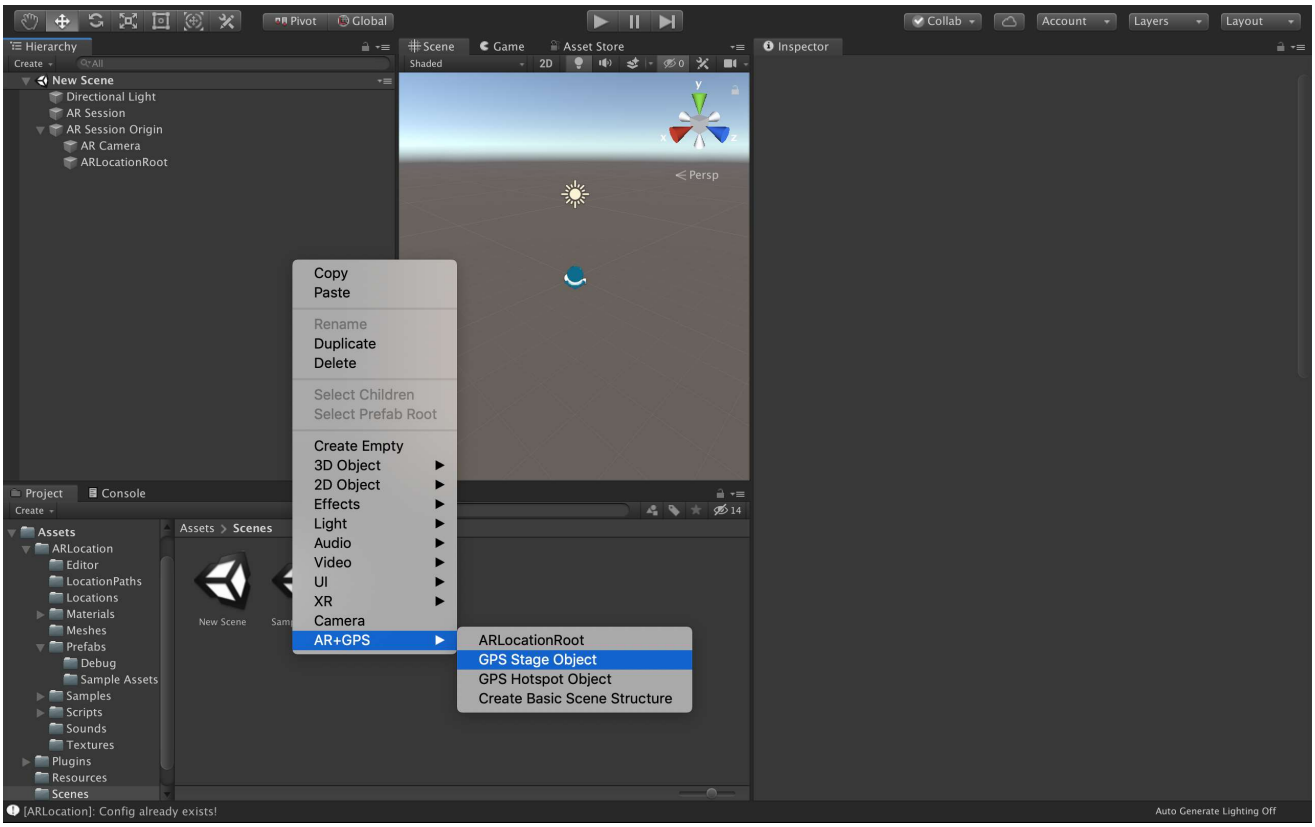
Then create a `AR+GPS -> ARLocationRoot` object as a child of the `AR Session Origin` .



Unity AR+GPS Location Docs (v3.0+)

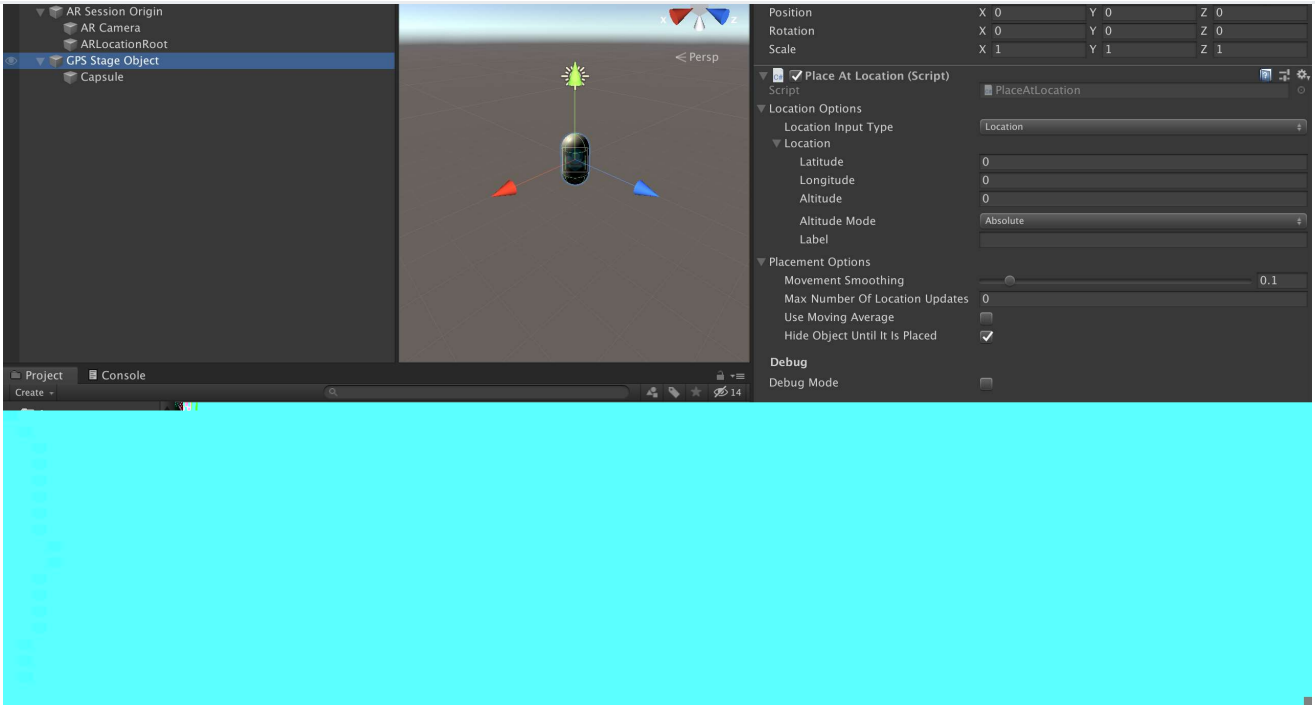


Ok, now the basic `AR+GPS` structure is laid out! All we need now is an object to be positioned in a given geo-location. There are a few components that will do that in different ways, as we will see later. For now, let's create a `AR+GPS -> GPS Stage Object`.



This creates an empty GameObject with a `PlaceAtLocation` component added to it. So, for something to actually appear, we should, for instance, child it to this component. Here we just create a `Capsule` as a child.

Unity AR+GPS Location Docs (v3.0+)

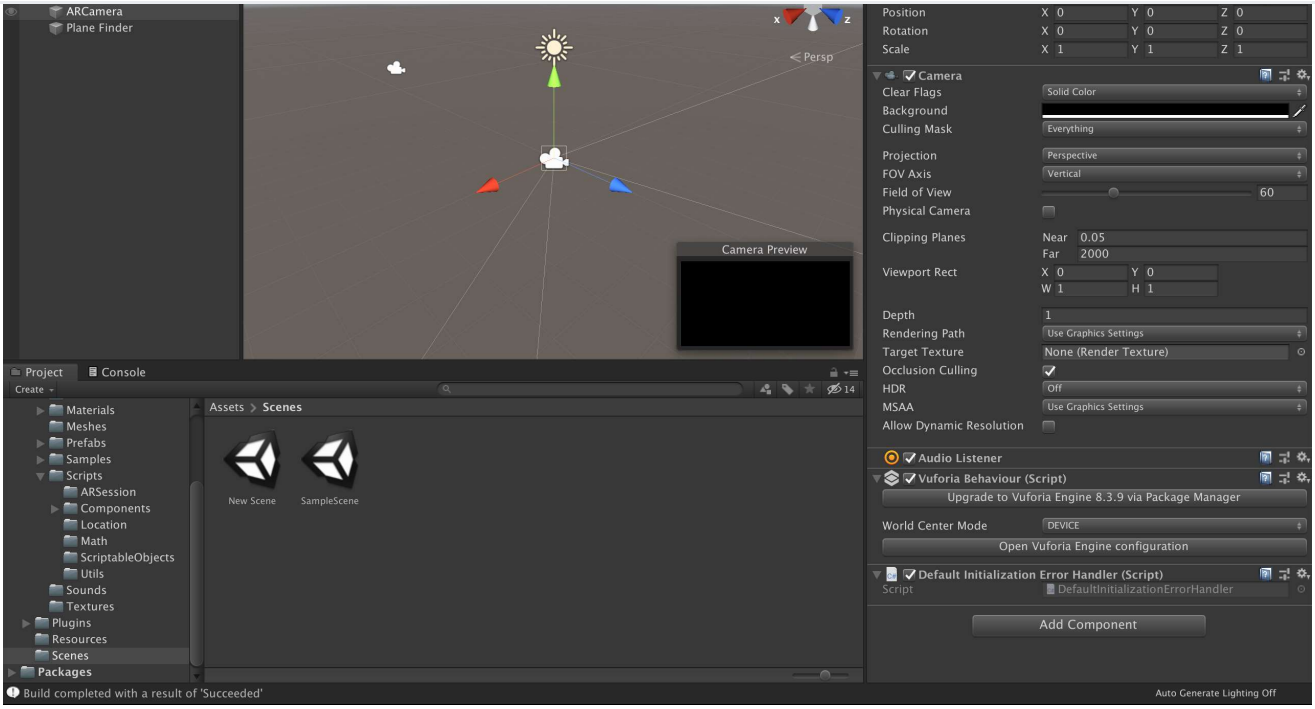


Again, all is needed is to set the desired geographical coordinates in the inspector for the `PlaceAtLocation` component of the `GPS Stage Object` to place the object in the correspond location!

Vuforia

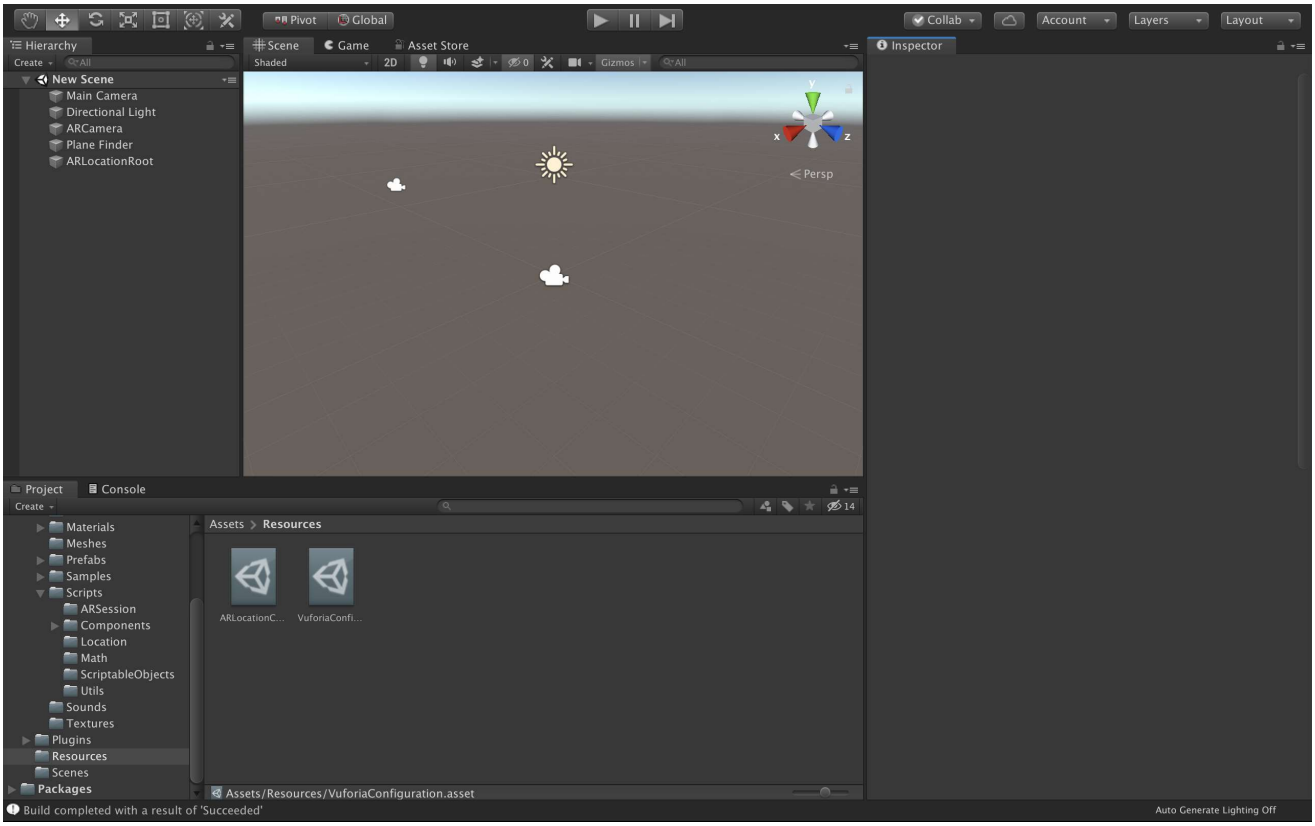
Again, starting with a new scene, delete the old `Main Camera`. Then, create a `Vuforia Engine -> AR Camera` and a `Vuforia Engine -> Ground Plane -> Plane Finder`, to create the basic Vuforia structure, like bellow.

Unity AR+GPS Location Docs (v3.0+)

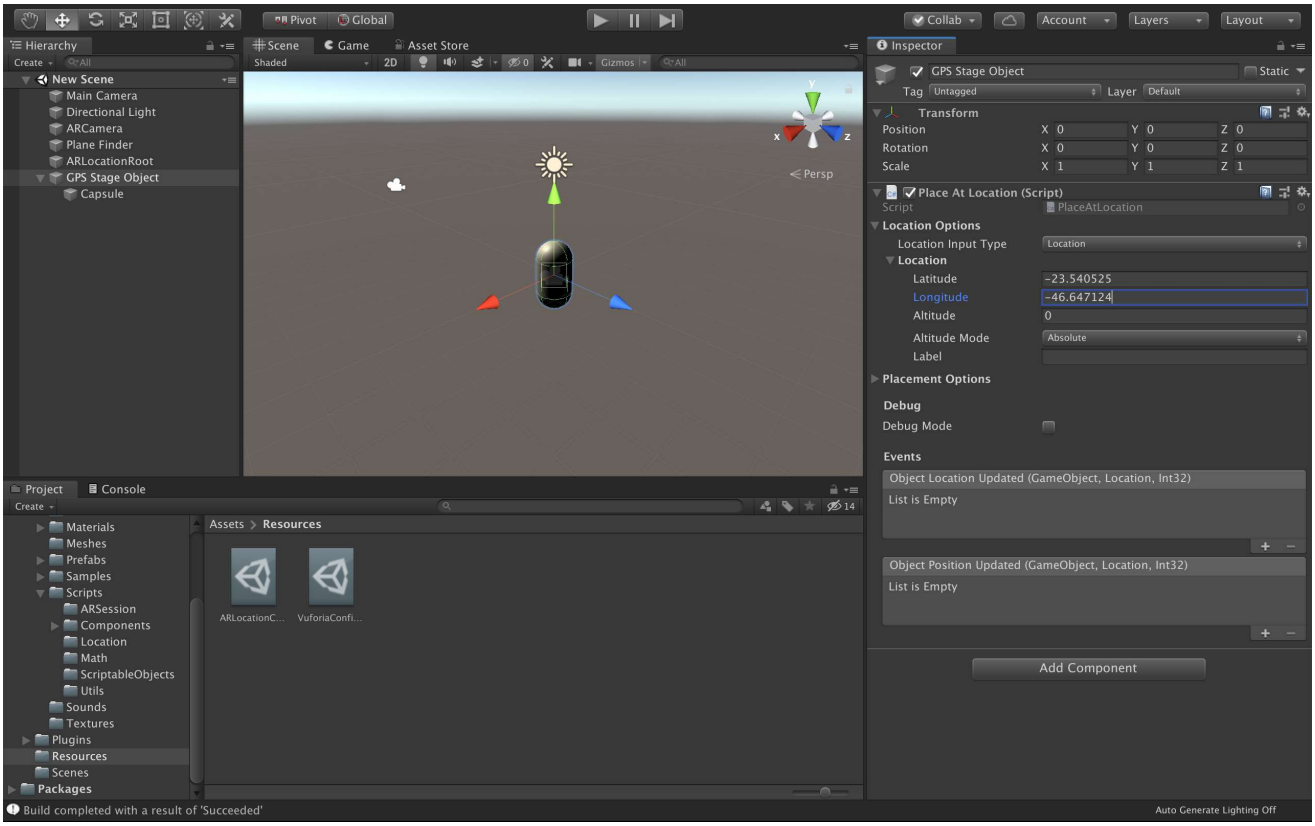


As mentioned in the last section, check that Vuforia is properly configured for positional tracking.

Now, create a `AR+GPS -> ARLocationRoot` object at the root of the scene, and the basic structure is complete.



You can add a `AR+GPS -> GPS Stage Object` and add some renderable object as it's child, setting the desired geographical coordinates at the `PlaceAtLocation` inspector for the `GPS Stage Object`.



Concepts

The `AR+GPS` system works by mixing the incoming GPS data from the device with the underlying AR tracking done by AR Foundation of Vuforia.

Both in AR Foundation and Vuforia, the AR tracking moves the camera (which starts at the origin of the world coordinates) around so that it matches the real movement of the device.

What the `AR+GPS` system does it to calculate the position and orientation of the gps-positioned objects relative to the user/camera, both in terms of position and orientation.

For that, it is essential that at `ARLocationRoot` object exists in the scene, as a sibling of the

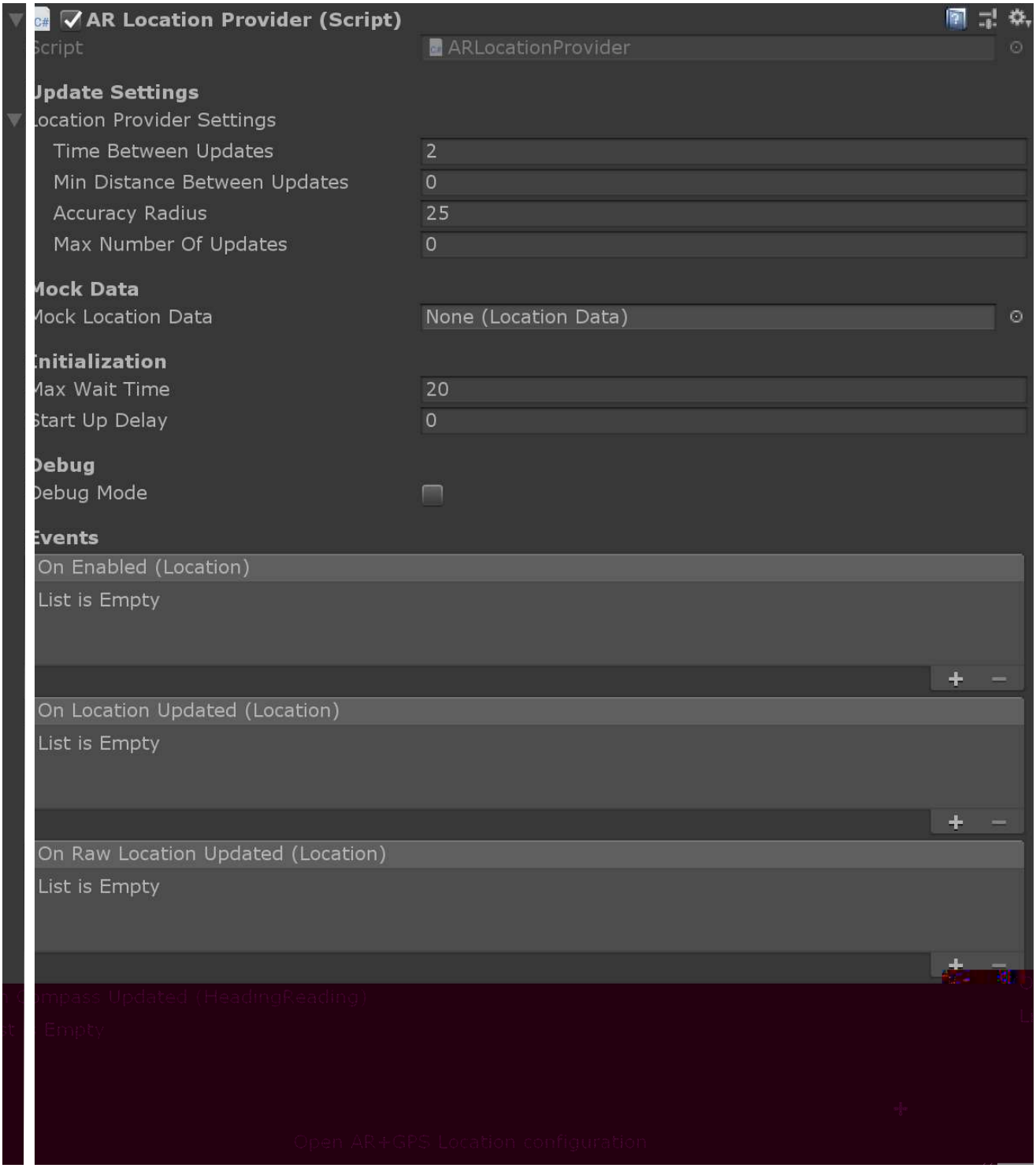
`ARLocationRoot` game object is aligned with the geographical directions, so that the objects are positioned correctly.

The `ARLocationProvider` handles all the GPS data incoming from the device. It can be placed anywhere in the scene. This data is filtered via the used configuration used by the `ARLocationP`

d when the AR tracking starts for the first time.

`On Tracking Lost` henever the AR tracking is lost.

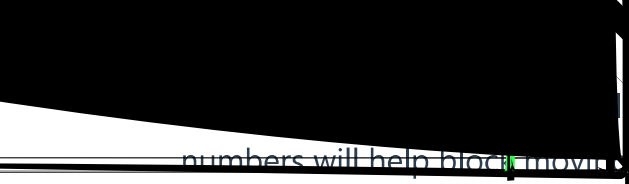
`On Tracking Restored` ed whenever the AR tracking is restored after being lost.



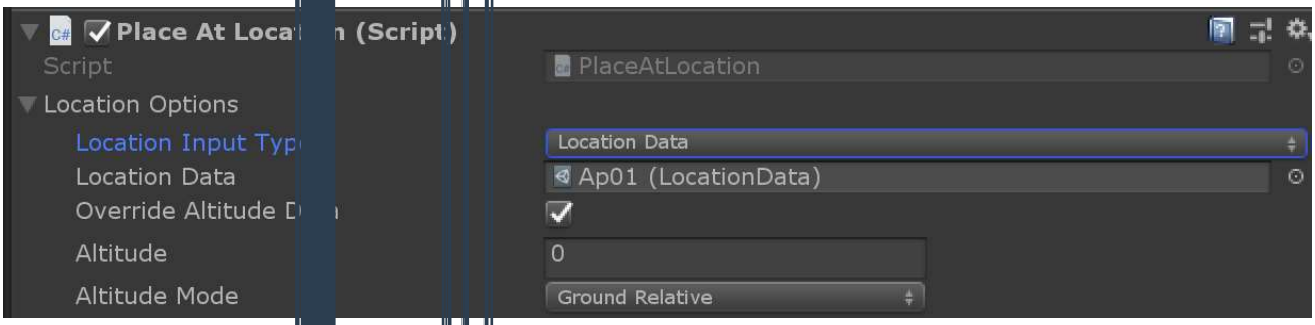
This component manages all the data incoming from the GPS and magnetic sensors. The data is filtered according to the settings used, but you can also have access to the raw data. This component is required for the `AR+GPS` system to function.

Properties:

-



numbers will help block movi



This component will place the Game Object it is attached to at a given geographical location. This is the easiest way to place gps-positioned AR content.

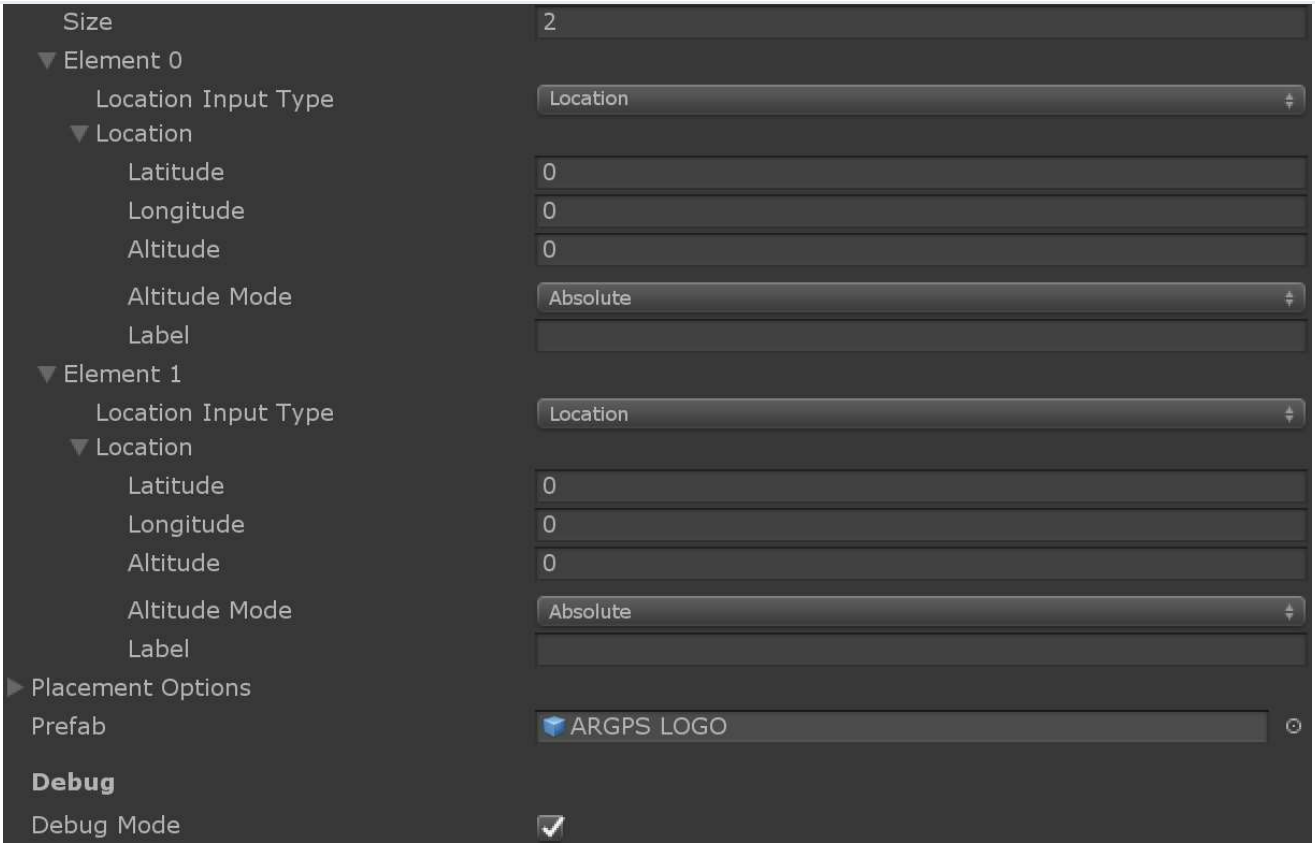
Properties:

- `Location Input Type` The type of location coordinate input used. Either 'Location' to directly input location coordinates, or 'LocationData' to use a ScriptableObject.
- `Location` Input the desired GPS coordinates here.
- `Location Data` A LocationData ScriptableObject storing the desired GPS coordinates to place the object.
- `Override Altitude Data` If true, override the LocationData's altitude.
- `vement Smoothi` The smoothing fa

n updates. Zero means no limits are imposed.

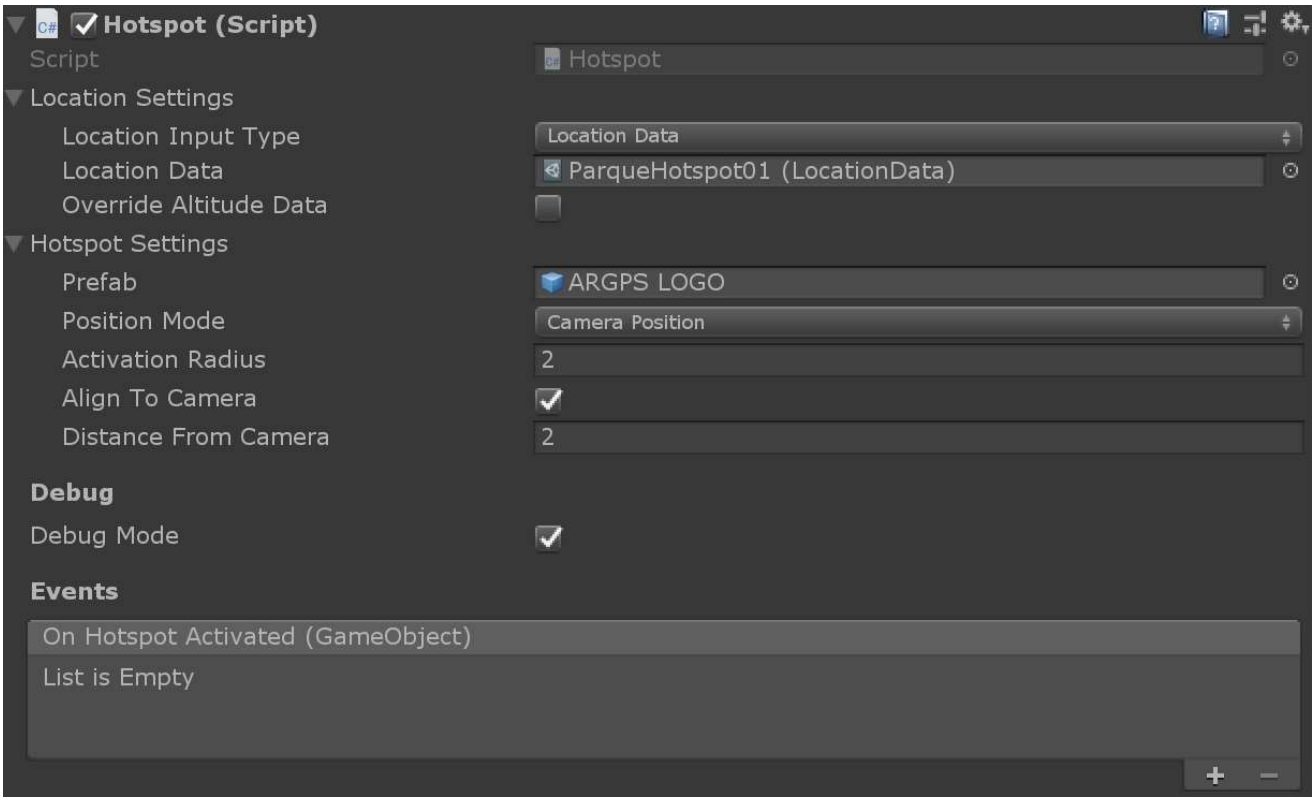
- `Use Moving Average` If true, use a m

Unity AR+GPS Location Docs (v3.0+)



This component will take an array of `Locations` and instantiate a given `Prefab`, placing the instances in the given locations.

Hotspot

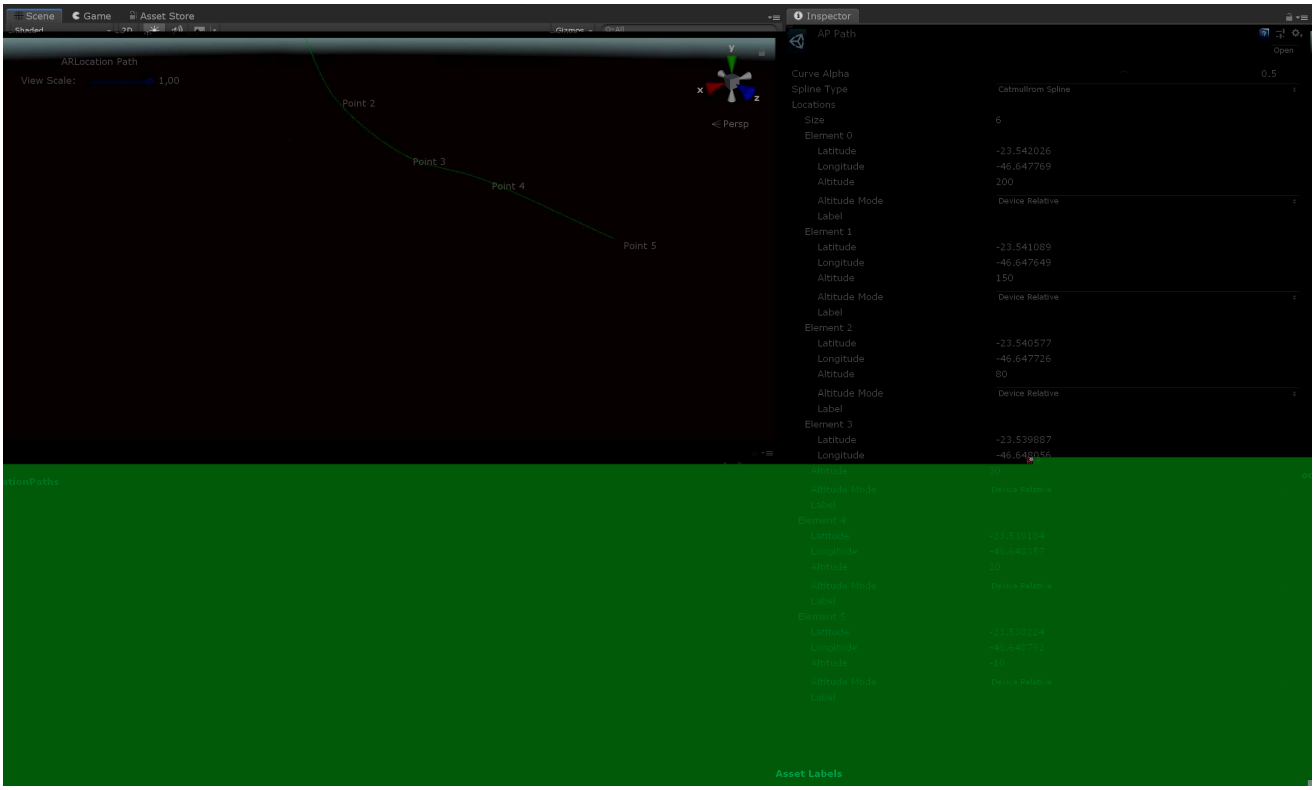


Prope

- P
- P
- e number of points-per-segment used to calculate the spline.
- `Movement Smoothing` The smoothing factor for movement due to GPS location adjustments; if set to zero it is disabled.
- `Max Number Of Location Updates` The maximum number of times this object will be affected by GPS location updates. Zero means no limits are imposed.
- `Use Moving Average` If true, use a moving average filter.

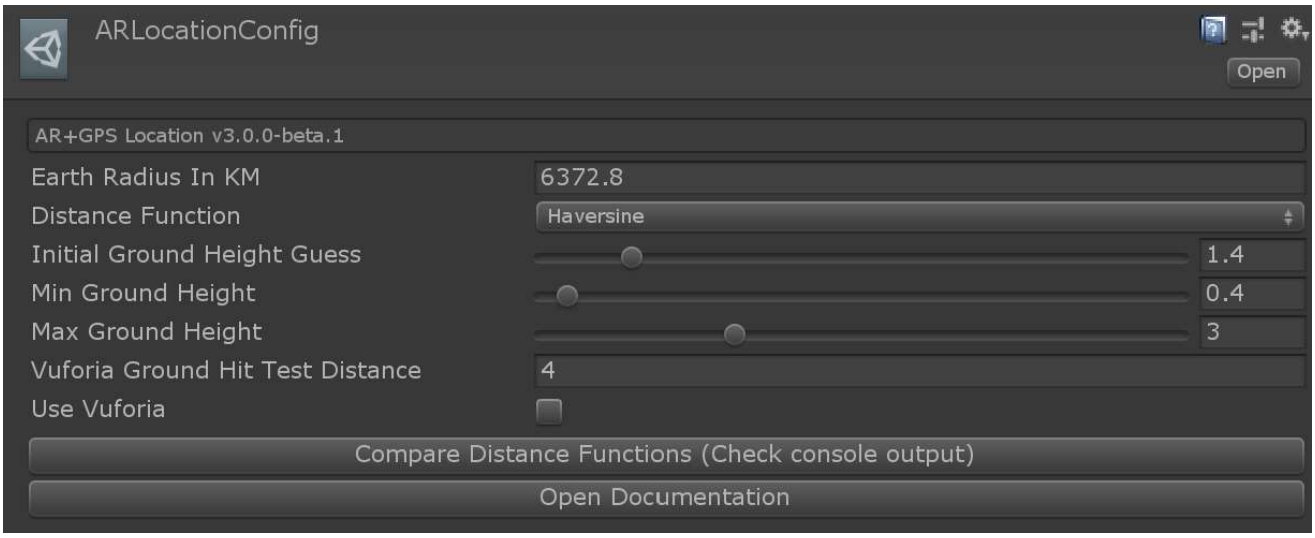
Unity AR+GPS Location Docs (v3.0+)

LocationPath



Holds data for a path/curve in space, defined by a set of geo-coordinates. To create, go to `Create -> AR+GPS -> Path` .

ARLocationConfig



This holds the global configuration for the `Unity AR+GPS Location` . The plugin will automatically create a configuration file at `Resources/ARLocationConfig` .


lue,
Altitude = ltValue,
AltitudeMo AltitudeMode.GroundRelative
};
var opts = new aceAtLocation.PlaceAtOptions()
{
HideObject tilItIsPlaced = true,
MaxNumber pdates = 2,
MovementSmoothing = 0.1f,
UseMovingAverage = false
};
PlaceAtLocation.AddPlaceAtComponent(g
https://5d538925d91704000ae5fee

ly loop trough all of them, apply

laced object

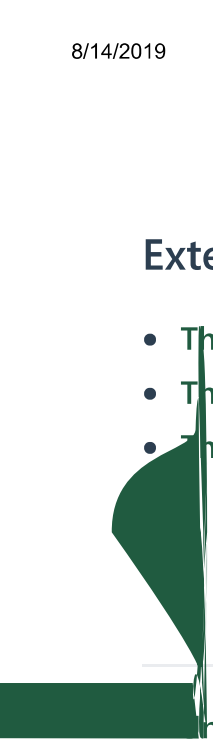
ne `PlaceAtLocation` component, you can monitor
by calling

between the game object and the camera. You can
by doing

s based on the distance of the user to the object.

External Resources

- The ARFoundation manual .
- The Unity AR Handheld forums .
- The Vuforia Unity Forums .



nity AR+GPS Lo