Implémentation Réalité Augmentée dans Unity

Corentin Coupry

Février 2023

Résumé

Ce document décrit une méthode d'implémentation et de préparation d'Unity pour l'utilisation de la réalité augmentée.

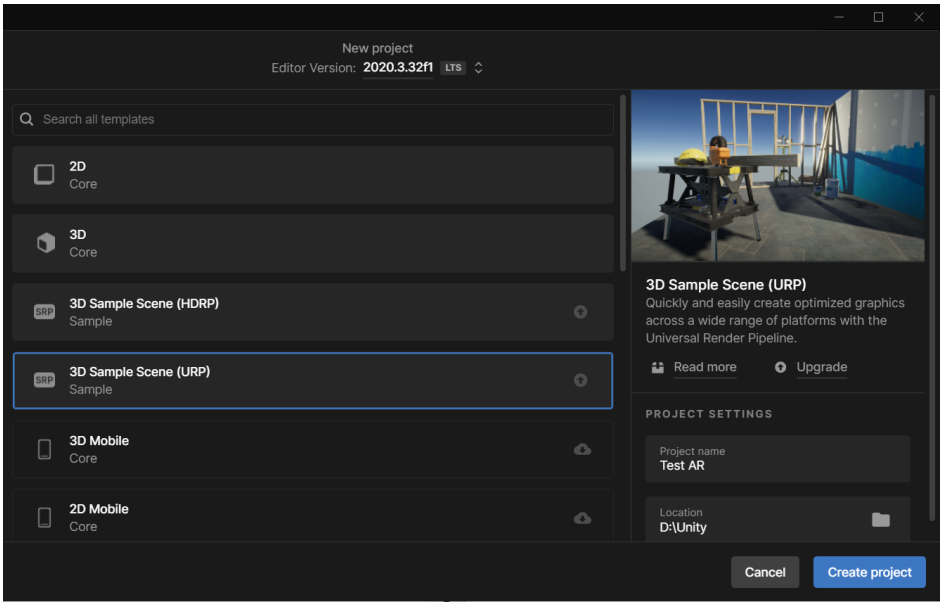
ATTENTION ! ⚠

Cette méthode n'est pas unique ! Il existe plusieurs façons différentes d'implémenter la réalité augmentée dans Unity. Tout dépend de la version du programme utilisée. Ici, nous allons utiliser la version 2020.3 LTS. Vérifiez donc que votre logiciel est bien sous cette version. Sinon, vous pouvez la télécharger ici ou bien en utilisant le UnityHub.

1 Installation

1.1 Création du projet

Créez un nouveau projet en utilisant le template **Universal Render Pipeline**. Donnez-lui un nom et enregistrez-le à un endroit approprié. Faites attention à ne pas créer le projet dans un dossier sur clé USB, pour éviter les problèmes de compilation.



1.2 AR Foundation

Dans cette partie, nous allons explorer l'installation et l'utilisation du package AR Foundation.

1.2.1 C'est quoi AR Foundation ?

Le package AR Foundation est apparu avec Unity 2018.1 et propose un support multi-plateformes pour créer des applications de réalité augmentée. Ce nouveau package permet de créer une application qui fontionnera aussi bien en utilisant ARCore (plugin Android), ARKit (plugin iOs) ou OpenXR (plugin Microsoft Hololens).

1.2.2 Packages nécessaires

La première étape est d'importer les packages qui seront nécessaires au développement. Tout d'abord :

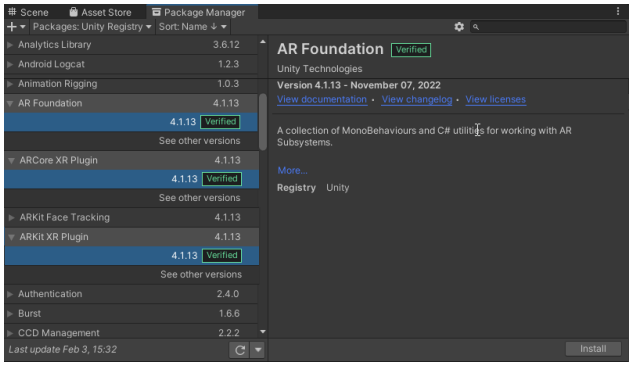
— Ouvrez **Window > Package Manager**

— Dans cette fenêtre, installer les packages suivants :

  — **AR Foundation**

  — **ARCore XR Plugin** (si vous travaillez sous Android, ce qui est notre cas ici)

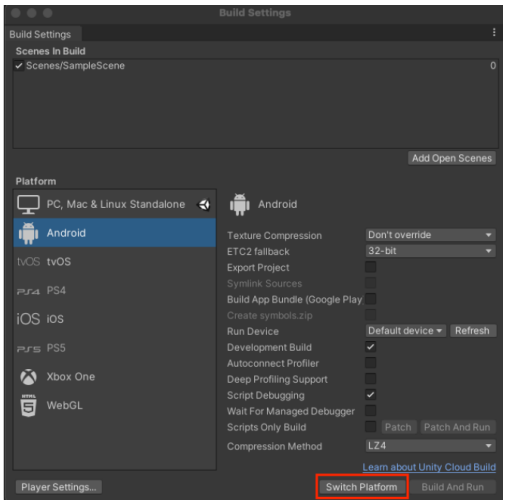
  — **ARKit XR Plugin** (si vous souhaitez travailler sous iOs)



1.3 Changement de paramètres

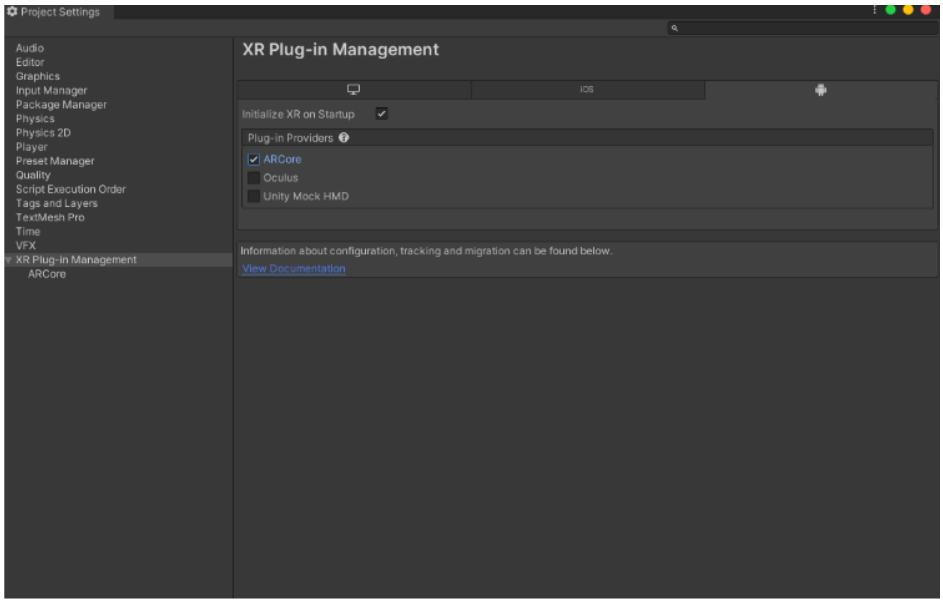
1.3.1 Build Settings

1. Ouvrez **File > Build Settings**
2. Dans la partie **Platform** à gauche, sélectionnez **Android**
3. Si vous le souhaitez, vous pouvez également activer **Development Build** et **Script Debugging** afin de récupérer des informations dans les logs lorsque votre application fonctionne.
4. Cliquez sur **Switch Platform**

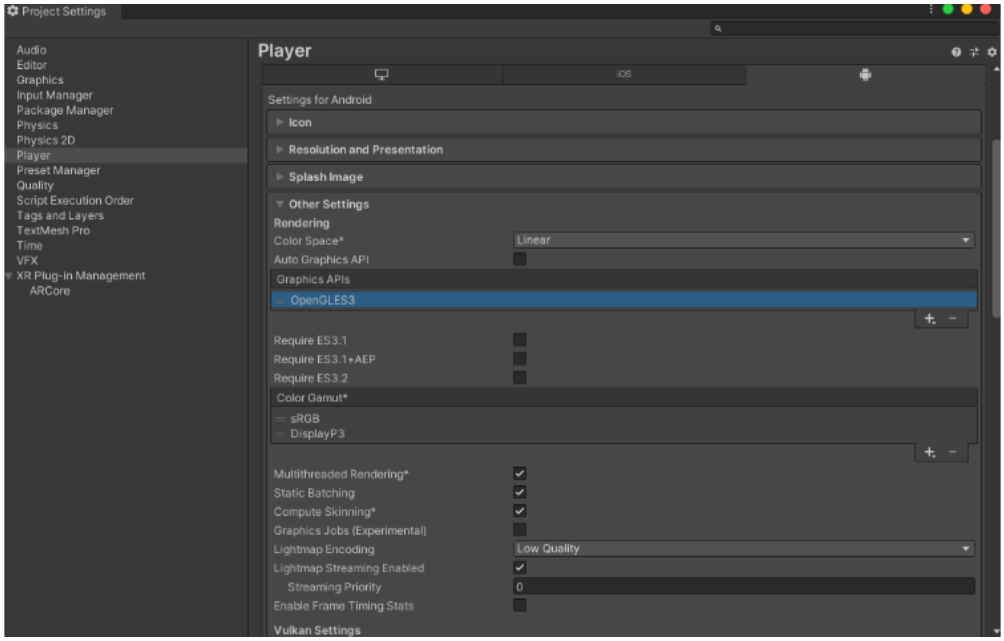


1.3.2 Project Settings

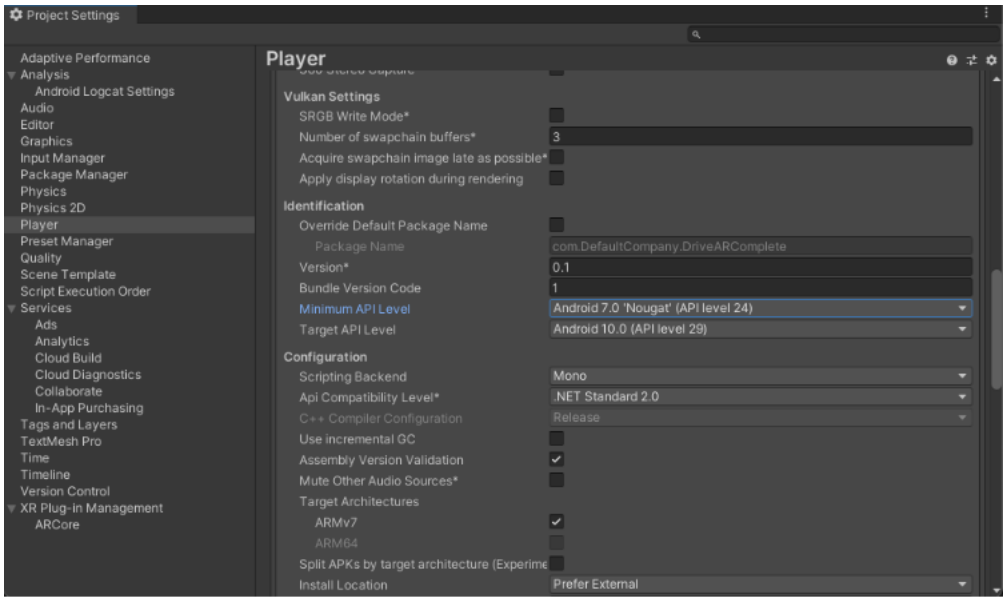
1. Ouvrez **Edit > ProjectSettings...** et sélectionnez la section **XR Plug-in Management**
2. Dans l'onglet **Android**, activez **ARCore**



1. Dans le panneau gauche, cliquez sur la section **Player**
2. Dans l'onglet **Android**, sous **Other Settings**, déselectionnez **Vulkan** de la partie **Graphics APIs**.



Les applications ARCore nécessitent au minimum un niveau API de 24 (*Android 7.0 'Nougat'*). Descendez et trouvez la ligne **Minimum API Level**. Définissez le niveau minimum d'API à 24.

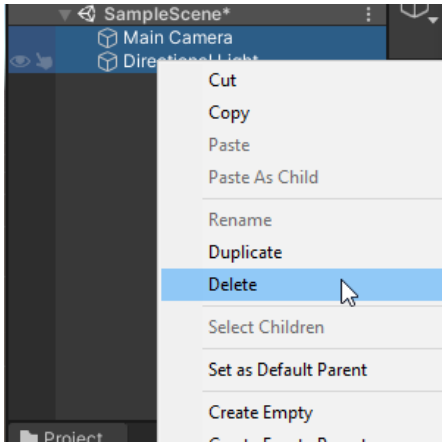


2 Création de la scène

Dans cette partie, nous allons créer une première scène afin de tester l'utilisation de AR Foundation

2.1 Ajout des éléments nécessaires

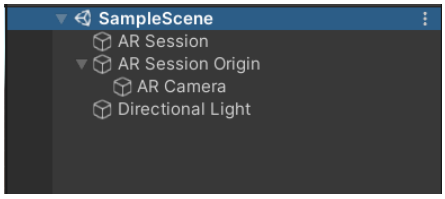
Tout d'abord, nous allons vider la scène des éléments déjà présents. Pour cela, supprimer l'ensemble des éléments de *SampleScene*. Ensuite, nous allons ajouter les éléments de AR Foundation. Faites un clic-droit sur le



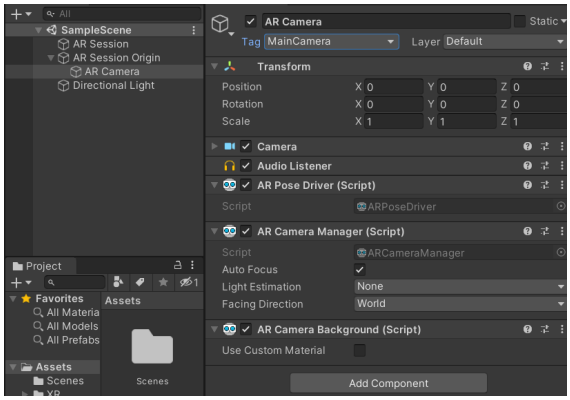
panneau **Hierarchy** et utiliser le menu pour ajouter :

1. **XR > AR Session** : Cet objet contrôle l'ensemble de l'expérience de réalité augmentée
2. **XR > AR Session Origin** : Cet objet transforme les coordonnées AR (celles du monde réel) en coordonnées spécifiques à Unity
3. **Light > Directional Light** : Cet objet permet d'illuminer la scène ainsi que les objets qu'elle contient

Votre scène devrait ressembler à ça :



Ouvrez **AR Session Origin** que vous avez créer et sélectionnez l'objet **AR Camera**. Dans le *Inspector*, changer le tag de l'objet pour ***MainCamera*** si ce n'est pas déjà le cas.

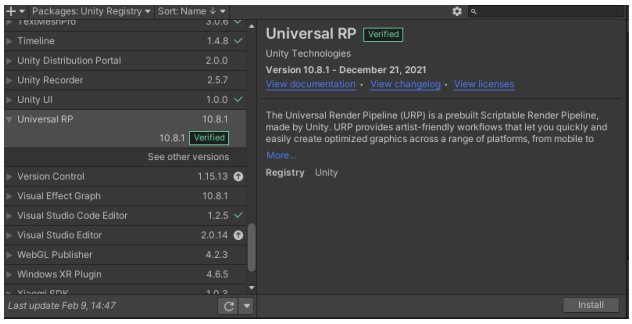


2.2 Changement du Pipeline

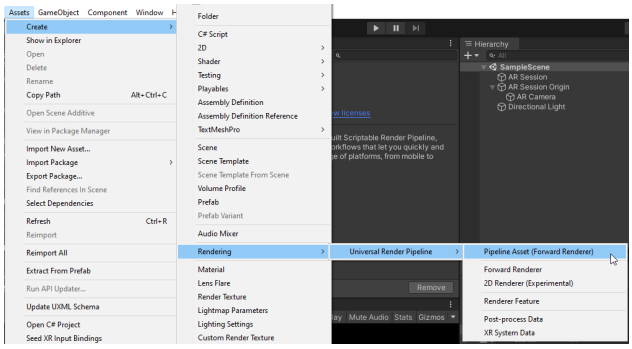
Nous allons, pour ce projet, utiliser le pipeline de rendu graphique **Universal Render Pipeline**. Je n'expliciterais pas plus ce pipeline ici, mais vous pourrez trouver plus d'informations sur le site dédié.

2.2.1 Cas d'un projet Built-in Pipeline

Si votre projet n'a pas été initialisé avec URP, il est toujours possible de l'intégrer sans détruire votre travail. Tout d'abord, il est nécessaire d'installer le package correspondant. Pour cela, ouvrez **Window > Package Manager**.

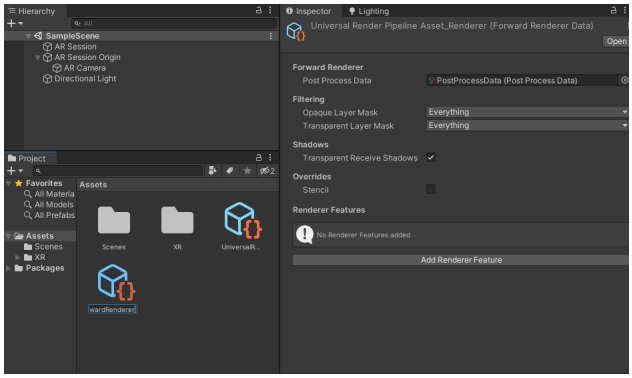


Une fois sur cette fenêtre, il faut installer le package **Universal RP**. Ensuite, une fois le package installé, cliquez sur **Create > Rendering > Universal Render Pipeline > Pipeline Asset (Forward Renderer)**.

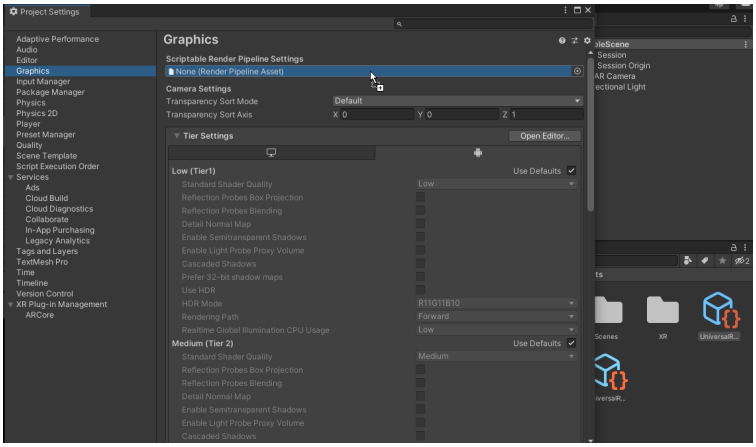


Vous pourrez voir deux nouveaux objets dans la fenêtre **Project** :

- **Universal Render Pipeline Asset**
- **Universal Render Pipeline Asset Renderer** (vous pouvez changer le nom de cet asset en **ForwardRenderer** si vous le souhaitez).



Ensuite, nous allons appliquer l'asset au projet. Pour cela, ouvrez **Edit > Project Settings** dans l'éditeur et sélectionner l'item **Graphics**. Ensuite, assignez l'asset **Universal Render Pipeline Asset** créé dans la case **Scriptable Render Pipeline Settings**



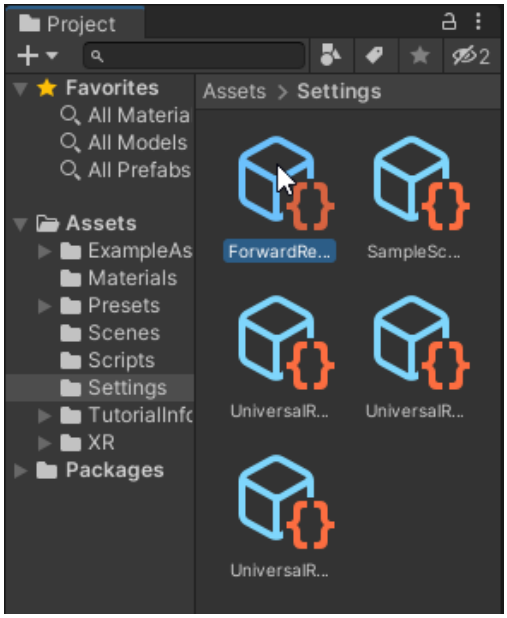
ATTENTION! ⚠

Si certains de vos shaders ont été créés en utilisant le Render Pipeline intégré à Unity, il est nécessaire de les convertir en Shaders URP. Vous pourrez trouver plus d'informations à ce lien.

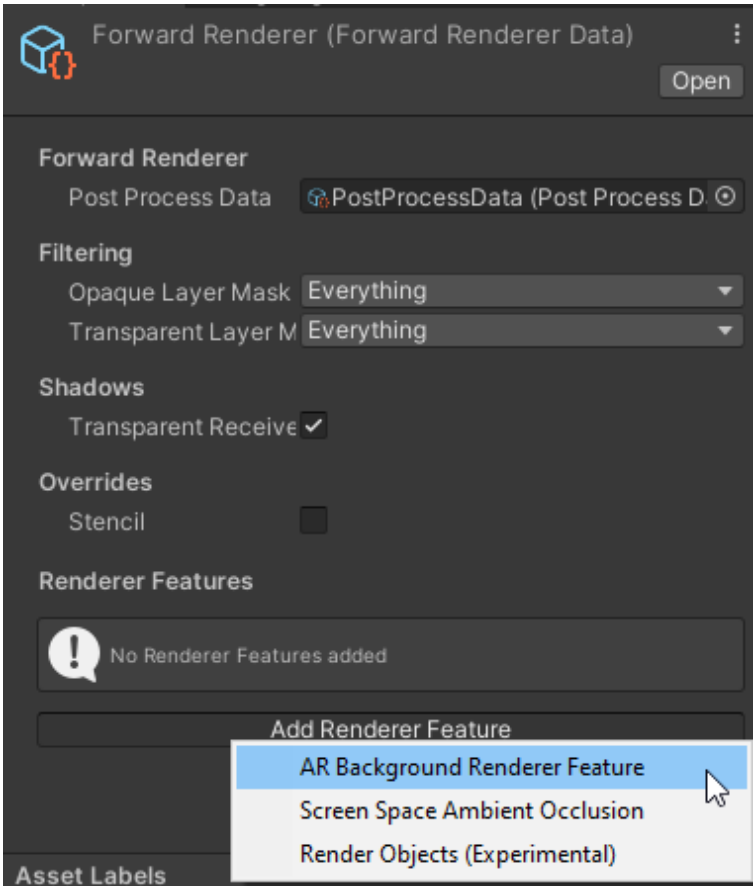
2.2.2 Cas d'un projet URP

Si vous avez créer votre projet en utilisant le **Universal Render Pipeline** ou que vous l'avez intégré, il est temps de le rendre compatible avec AR Foundation. La première étape est de sélectionner, dans le panneau **Project**, l'asset **ForwardRenderer**.

— Si vous avez créer votre projet en utilisant URP, vous pourrez le trouver en allant à **Assets > Settings**
— Si vous l'avez intégré par la suite, vous pourrez le trouver dans le dossier dans lequel vous l'avez rangé



Ensuite, sélectionnez **ForwardRenderer** puis, dans le panneau *Inspector*, utiliser **Add Renderer Feature** pour ajouter un **AR Background Renderer Feature**. Cet élément permet de bien générer le flux caméra dans votre scène.



2.2.3 Vérification de l'installation

Pour voir si l'installation est correcte, nous allons compiler un exemple. Tout d'abord, assurez-vous que votre téléphone est branché à l'ordinateur et que ADB soit correctement installé (voir 3.1). Ensuite, cliquez sur **File > Build and Run...** Si tout se passe bien, une fois compilée, l'application devrait s'installer et vous devriez pouvoir voir le retour caméra sur l'écran de votre téléphone.

3 Installations spécifiques à Android

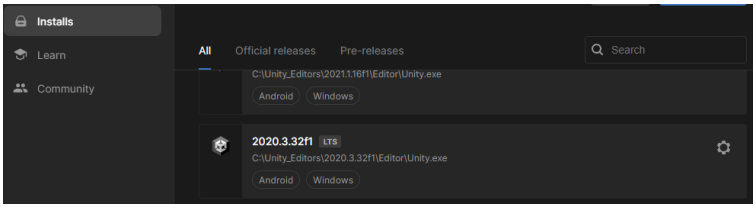
Certains éléments sont nécessaires pour que vous puissiez créer une application sous Android. Nous allons ici vérifier et, le cas échéant, voir comment installer, les éléments indispensables pour un bon développement.

3.1 ADB

La première étape pour efficacement développer une application pour Android est d'installer le **Android Debug Bridge (ADB)**. Il est inclus dans le package *platform-tools* que vous pourrez télécharger à ce lien. Il sera aussi nécessaire d'aactiver le débogage USB sur votre téléphone Android. Pour cela, il vous faudra aller dans les **Options pour les développeurs** sur votre téléphone. Cette option est masquée par défaut pour les versions Android 4.2 et plus récentes. Pous l'activer, vous pouvez aller suivre les instructions concernant à votre version d'Android sur cette page.

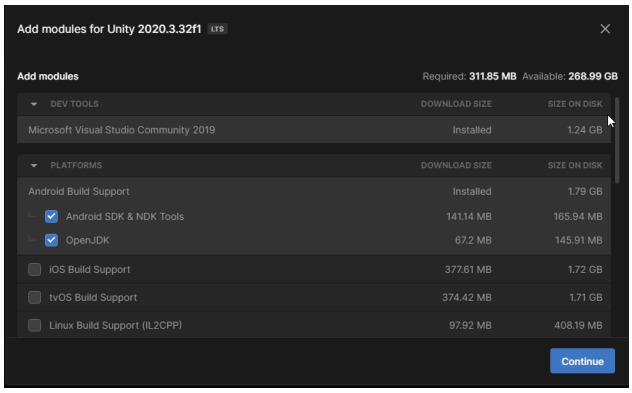
3.2 Installation Android

Ouvrez Unity Hub et, dans l'onglet **Installs**, trouvez l'installation de votre version d'Unity (ici, 2020.3.32f1 LTS).



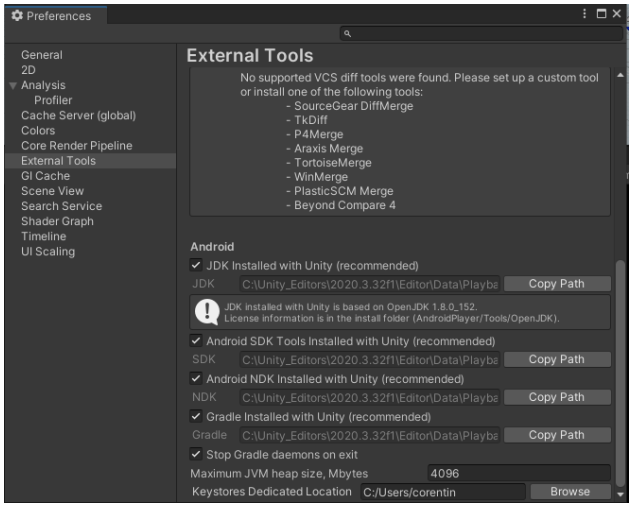
Sélectionnez l'icône de paramètres, sur la droite, et ouvrez la fenêtre **Add modules**. Une fois sur cette page, dans la partie *Platforms*, trouvez la ligne **Android Build Support** et sélectionnez les éléments :

— **Android SDK & NDK Tools**
— **OpenJDK**



3.3 Vérification du projet

Dans votre projet Unity, ouvrez la fenêtre **Edit > Preferences** et allez sur l'onglet **External Tools**. Cherchez la partie *Android* et regardez si les JDK, SDK, NDK et Gradle sont bien paramétrés.



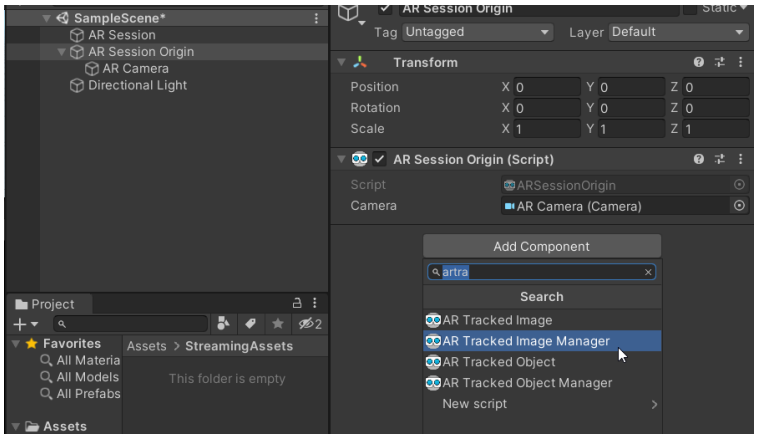
Si jamais les éléments ne sont pas bien installés, il vous suffit de suivre les instructions de la section 3.2.

4 Image Tracking

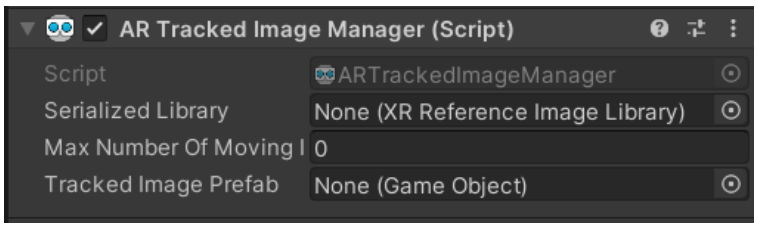
Nous allons ici expliciter un exemple d'initialisation de tracking d'images en utilisant AR Foundation

4.1 Initialisation du tracker

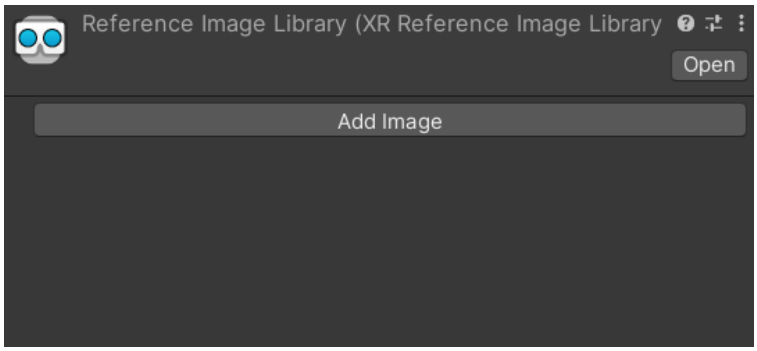
Tout d'abord, cliquez sur **AR Session Origin** puis, dans le panneau *Inspector*, cliquez sur **AddComponent** et ajoutez **AR Tracked Image Manager**



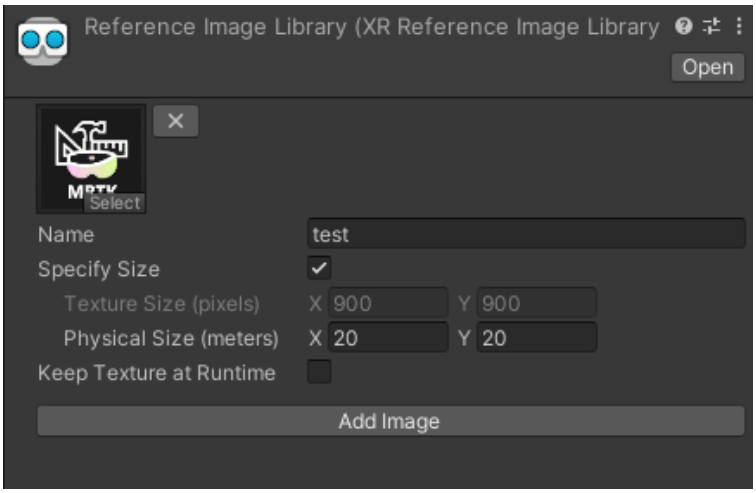
Le script permet de créer des *GameObjects* pour chaque image référencée par *Reference Image Library* détectée dans la scène.



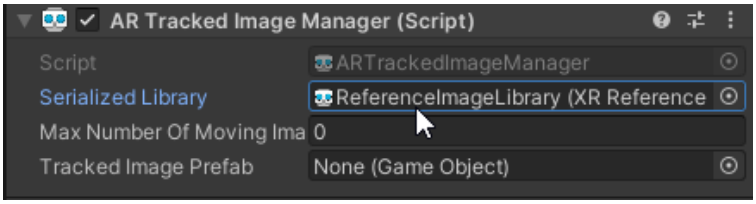
Nous allons ensuite créer notre *Reference Image Library* en faisant un clic droit sur la fenêtre projet, puis en allant sur **Create > XR > Reference Image Library**.



Ensuite, cliquez sur Add Image. Glissez l'image qui vous servira de marqueur dans l'emplacement correspondant. Attention, cette image doit déjà être incluse dans votre projet!

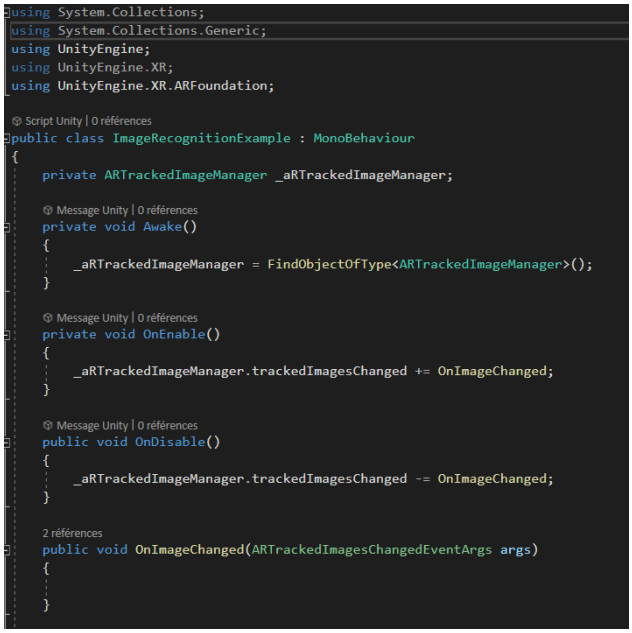


Nommez votre marqueur (ici *test*). Activez ensuite l'option **Specify Size** et spécifiez les dimensions réelles de votre marqueur. Cette information permettra ensuite à Unity de calculer à la fois la distance de votre image par rapport à la caméra, mais aussi les proportions d'affichage de l'objet 3D correspondant, le cas échéant. Une fois la librairie créée, vous pouvez la référencée dans votre **AR Tracked Image Manager** dans l'emplacement **Serialized Library**.



4.2 Création du script

Nous allons ensuite créer un premier script pour tester cette fonctionnalité. Tout d'abord, créez un script appelé *ImageRecognitionExample* et ouvrez-le en utilisant votre éditeur de script préféré.



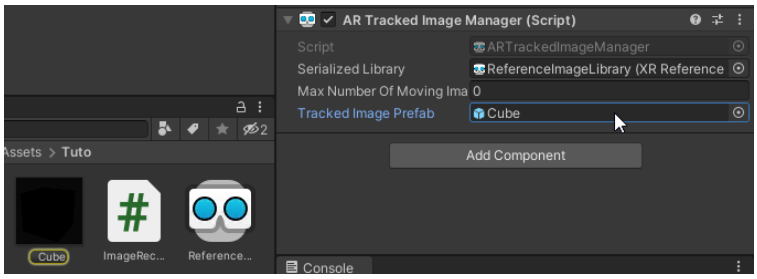
Tout d'abord, nous allons ajouter les lignes permettant d'importer les fonctions spécifiques à la XR.

```
using UnityEngine.XR
using UnityEngine.XR.ARFoundation
```

Ensuite, nous allons ajouter les lignes suivants à notre classe.

```
private ARTrackedImageManager _aRTrackedImageManager; //Reference au script de gestion

private void Awake()
{
    _aRTrackedImageManager = FindObjectOfType<ARTrackedImageManager>();
}
private void OnEnable()
{
    _aRTrackedImageManager.trackedImagesChanged += OnImageChanged;
}
public void OnDisable()
{
    _aRTrackedImageManager.trackedImagesChanged -= OnImageChanged;
}

// Cette fonction permet de determiner ce qui se passe lorsqu'une image est detectee
public void OnImageChanged(ARTrackedImagesChangedEventArgs args)
{
    foreach(var trackedImage in args.added)
    {
        Debug.Log(trackedImage.name);
    }
}
```

Ensuite, créez un cube (ou n'importe quel objet 3D), faites-en un prefab et assignez-le à **AR Session Origin** dans l'emplacement **Tracked Image Prefab**, et compilez votre projet pour le faire tourner sur votre téléphone (Cliquez sur **File > Build and Run...**). Si vous regardez l'image utilisée, vous devriez voir le cube apparaître sur votre téléphone

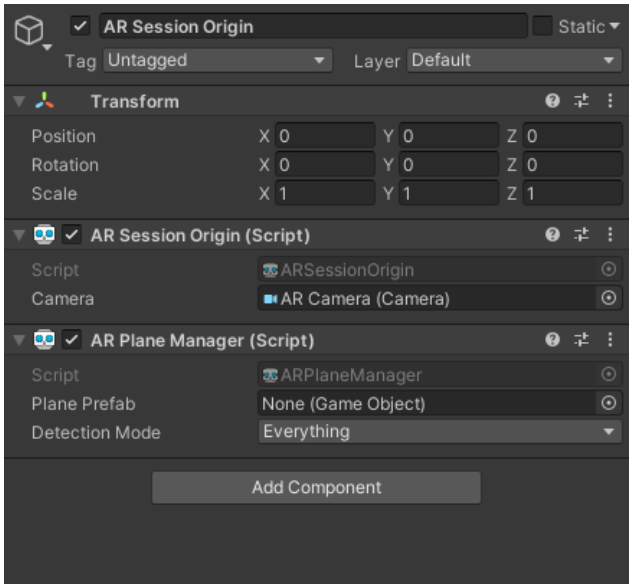


5 Plane Tracking

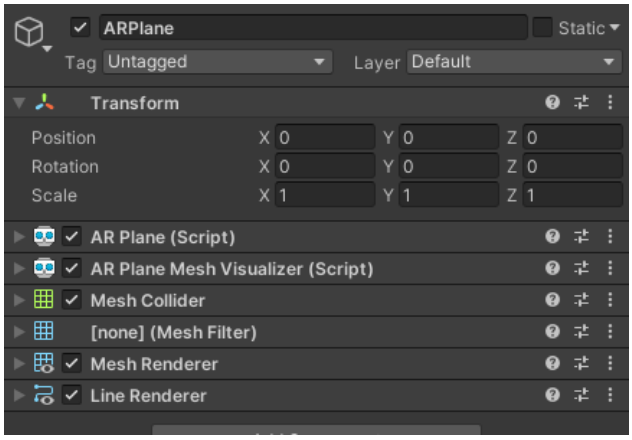
Nous allons ici expliciter un exemple d'initialisation de réalité augmentée sans marqueur utilisant AR Foundation.

5.1 Initialisation du tracker

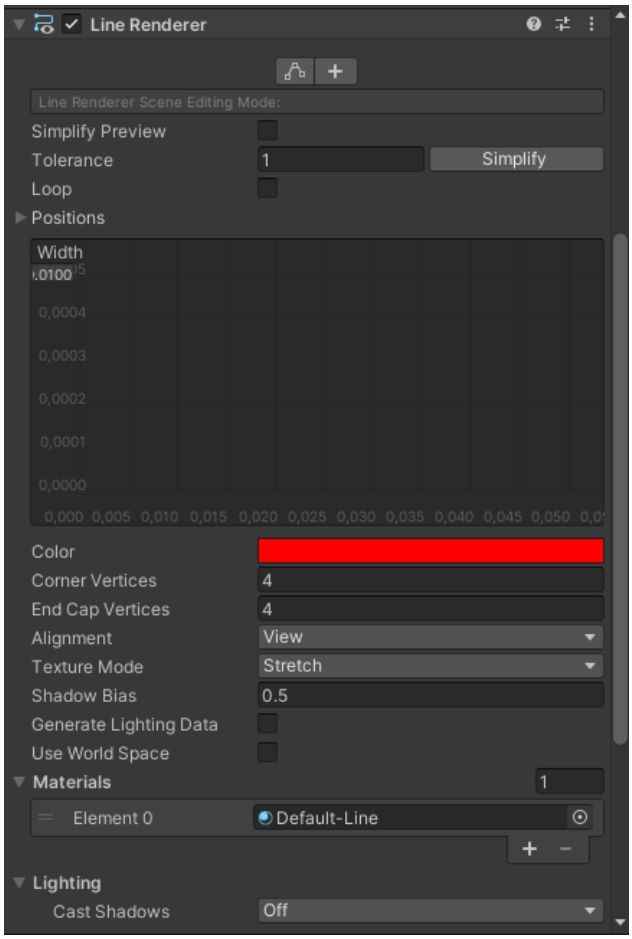
Tout d'abord, cliquez sur **AR Session Origin** puis, dans le panneau *Inspector*, cliquez sur **AddComponent** et ajoutez **AR Plane Manager**



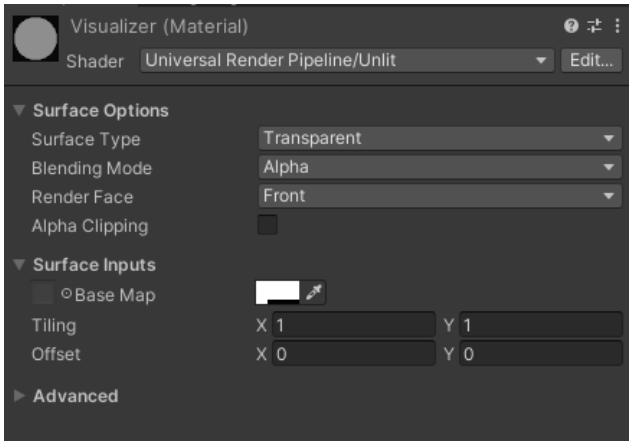
Ensuite, créer un nouveau **GameObject** et renommez-le **AR Plane**. Ajoutez lui les scripts **AR Plane** et **AR Plane Mesh Visualizer**. Ajoutez lui également les composants **Mesh Collider**, **Mesh Filter**, **Mesh Renderer** et **Line Renderer**.



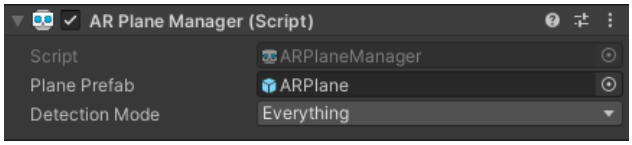
Dans le composant **Line Renderer**, choisissez une couleur visible (ici, rouge).



Créez ensuite un **Material** dans la fenêtre **Project**. Il nous servira à visualiser la surface sur laquelle nous allons travailler. Puis, utilisez-le sur le **Mesh Renderer** de votre **AR Plane**. Faites-en ensuite un prefab.



Sur **AR Session Origin**, ajoutez le prefab **ARPlane** dans l'emplacement **Plane Prefab** de **AR Plane Manager**. Faites un build pour tester.



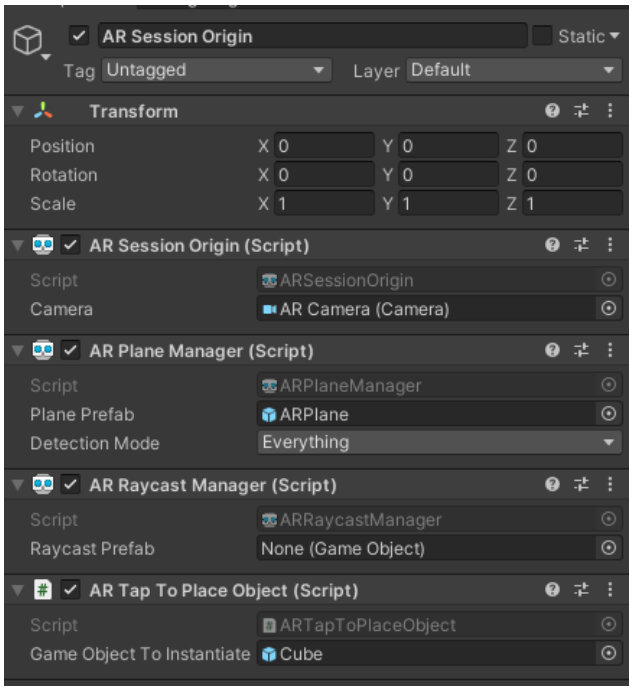
5.2 Toucher pour placer objets

Tout d'abord, il s'agit de créer le script qui permettra de gérer cette fonctionnalité. Ici, nous le nommons **ARTapToPlaceObject** :

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;

[RequireComponent(typeof(ARRaycastManager))]
public class ARTapToPlaceObject : MonoBehaviour
{
    public GameObject gameObjectToInstantiate;
    private GameObject spawnedObject;
    private ARRaycastManager _arRaycastManager;
    private Vector2 touchPosition;
    static List<ARRaycastHit> hits = new List<ARRaycastHit>();

    private void Awake()
    {
        _arRaycastManager = GetComponent<ARRaycastManager>();
    }
    bool TryGetTouchPosition(out Vector2 touchPosition)
    {//Cette fonction permet de d tecter lorsque l'utilisateur touche l' cran
        if(Input.touchCount > 0)
        {
```

```
            touchPosition = Input.GetTouch(index: 0).position;
            return true;
        }
        touchPosition = default;
        return false;
    }

    void Update()
    {
        if (!TryGetTouchPosition(out Vector2 touchPosition)) return;
        if(_arRaycastManager.Raycast(touchPosition, hits,
        trackableTypes:TrackableType.PlaneWithinPolygon))
        {
            var hitPose = hits[0].pose;
            if(spawnedObject == null)
            {
                spawnedObject = Instantiate(gameObjectToInstantiate,
                hitPose.position, hitPose.rotation);
            }
            else
            {
                spawnedObject.transform.position = hitPose.position;
            }
        }
    }
}
```

Ensuite, nous allons placer ce script sur **AR Session Origin** et lui donner un prefab à instancier lorsque l'utilisateur touche l'écran. Ici, ce prefab est un **Cube**. Faites un build pour tester.



6 Conclusion

Nous avons, dans ce document, couvert les principaux éléments qui vous permettront de développer une application mobile de réalité augmentée utilisation AR Foundation. Si vous souhaitez allez plus loins, vous pourrez trouver plus d'informations :

— AR Foundation Sample : lien
— About AR Foundation : lien
— Géolocalisation : lien

De plus, pour votre information, depuis la version 5.0 de **AR Foundation**, il est possible de tester votre application directement avec l'éditeur en utilisant **XR Simulation**. Pour cela, vous pouvez utiliser la versionr **2021.3.15f1** d'Unity.