Jour 5 - Minecraft AR

Le but de cette journée est de vous faire découvrir le développement en AR au travers d'un jeu. Comme vous l'aurez deviné, vous allez reproduire les principales fonctionnalités du jeu Minecraft, mais en AR.

Configuration

Développer une application sur Unity en AR requiert de posséder un smartphone Android de version 7.0 ou plus avec le mode développeur & le débogage USB activé. Si vous possédez un Iphone, il faut que celui-ci soit de génération supérieure ou égale à l'Iphone SE et un mac est requis pour compiler votre application.

Dans le Unity Hub, ajoutez un module à votre moteur. Il s'agit du module `Android / IOS Build support`.

Créez et ouvrez un nouveau projet. Sélectionnez `File/Build Settings` puis votre plateforme (ici Android) et confirmez en cliquant sur `Switch Platform`.

Dans le panneau en haut, cliquez sur `Add open scenes`. Cette option permet d'ajouter vos scènes actuelles à la pipeline de build du projet.

Ouvrez `Window/Package Manager` et installez les packages suivants :

- AR Foundation
- AR Core (Android)
- AR Kit (IOS)

Ouvrez `Edit/Project Settings` et suivez les instructions suivantes :

- Dans la table `Player`, éditez le nom de la compagnie à ***poc***.

- Dans `Other Settings`, activez l'option `Auto Graphics API`
- Un peu plus bas, pour les options `Package Name` et `Minimum API Level` faites correspondre les valeurs de ces deux champs avec celles recommandées : *com.poc.minecraftearth* et *Android 7.0 Nougat (API Level 24)*
- Dans la table `XR`, créez une instance de configuration (cliquez simplement sur `Create` et sauvegardez le fichier à l'emplacement proposé par défaut). Pensez à mettre à jour l'option `Requirement` de `required` à `optional`

C'est fini pour la configuration, passons au tests pour vérifier que tout fonctionne correctement.

Minecraft Earth

La première étape est de se repérer. Supprimez la caméra par défaut de votre scène et ajoutez à la place une caméra AR.
Pour cela, deux assets sont nécessaires :

- `AR Session Origin` qui contient une caméra pré-configurée pour utiliser les packages AR installés. Comme il s'agit de votre caméra principale, attribuez lui le tag `Main Camera.` Ce tag permet à Unity de déterminer depuis quelle caméra rendre une image à l'écran.
- `AR Session`, le point d'entrée pour la caméra qui lui permet d'utiliser les fonctionnalités de AR Core

Premier build, pour vérifier si tout fonctionne. Branchez votre smartphone à votre PC, et rendez-vous dans la fenêtre de build à `File/Build Settings`. Cliquez sur `Build And Run`.
Attendez que l'application compile, s'installe et se lance sur votre smartphone pour ensuite la tester.

Si tout se lance sans accroc mais que rien ne s'affiche, c'est normal ! Votre appareil détecte les surface mais ne les affiche pas encore.
Votre seconde tâche est donc d'assigner à une surface un prefab. Mais il faut également un moyen d'indiquer à votre AR Session Origin qu'il doit faire cette assignation.
Heureusement, cette fonctionnalité est gérée par un component, `AR Plane Manager`.
Une fois ajoutée, vous devriez jeter un coup d'œil aux prefab à lui attribuer.
Trouvez ce prefab et assignez-le à l'AR Plane Manager.

Relancez le build, vous devriez obtenir un résultat plus probant (attendez une dizaine de secondes au maximum après le lancement de l'application pour voir apparaître la surface détectée).

Ajoutez un indicateur, un pointeur qui s'adapte à la surface détectée. Cet indicateur doit être une image de viseur fourni par vos encadrants.
Afin que l'image s'adapte aux surface, vous devrez écrire un script qui :

- Récupère les surface détectées par votre session AR
- Attachez la position de l'image à la surface pointée par le centre de votre écran.

Puis lorsque l'utilisateur touche son écran, un cube doit apparaître à l'endroit correspondant sur la surface détectée par la session AR et doit lock sa position.

Mais tout comme dans Minecraft, si vous touchez votre écran, et que la surface pointée est une des faces d'un cube déjà existant, le nouveau cube doit être collé au premier.

Vient à présent une fonctionnalité importante du projet : supprimer des cubes.
Pour cela, ajoutez un bouton. Il devra posséder deux modes :

- Create - Lorsque ce mode est actif, une touche de l'utilisateur instancie un cube sur la scène (ce que vous avez fait jusqu'ici)
- Delete - Lorsque ce mode est actif, une touche de l'utilisateur détruit un cube ainsi que ceux qui lui ont été attachés après son instanciation

Maintenant que vous pouvez ajouter et supprimer des cubes, il serait temps de donner un objectif au joueur n'est-ce pas ?

Ajoutez un autre bouton qui permet quant à lui de faire apparaître à l'écran une image d'une forme que l'utilisateur doit reproduire. Si les deux formes correspondent, la partie est gagnée !

La dernière tâche est de proposer des formes procédurales à la place de l'image prédéfinie. Bonne chance à vous ;)