MAHDIA RESEAU INSTITUT SUPÉRIEUR DES ÉTUDES TECHNOLOGIQUES	**I**nstitut **S**upérieur des **E**tudes **T**echnologiques - **M**ahdia Département Technologies de l'Informatique	**Atelier :** Réa. Augmentée & Réa.Virtuelle **Enseignante :** M^{me} Amel Tilouche **Classe :** MDW 5.1

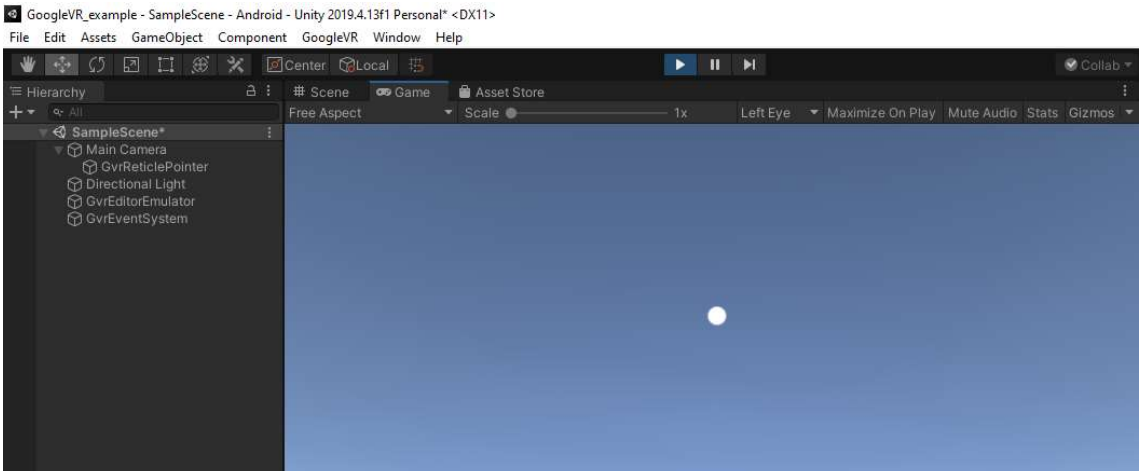
Travaux Pratiques n°4

Initiation à GoogleVR

Travail demandé

1. Créer un nouveau projet 3D.
2. Télécharger le SDK **GoogleVR** et importer son package Unity à votre projet.
   https://github.com/googlevr/gvr-unity-sdk/releases
3. Changer la plateforme en Android.
4. Définir les paramètres suivants :
   - Au niveau du ***Player Settings > XR Settings***,
     - Activer l'option : "***Virtual Reality Supported***"
     - Ajouter le type de casque à utiliser, pour notre cas « Cardboard » au niveau de Virtual Reality SDKs.
   - Au niveau du ***Player Settings > Other Settings***, définir le niveau minimal de l'API à « Android 4.4 'KitKat' (API level 19) » ou supérieur.
5. A partir du panneau ***Project > GoogleVR > Prefabs***, ajouter le Prefab « GvrEditorEmulator » à la vue ***Hierarchy*** de la scène.
   Ce Prefab permettra d'exécuter des programmes sur la machine de développement sans avoir à télécharger sur un appareil Android chaque fois que nous voulons tester quelque chose.
6. A partir du panneau ***Project > GoogleVR > Prefabs > EventSystem***, ajouter le Prefab « GvrEventSystem » à la vue ***Hierarchy*** de la scène.
   Ce Prefab permettra au système GoogleVR de traiter les événements qui peuvent se produire, comme la focalisation du regard sur un objet.
7. A partir du panneau ***Project > GoogleVR > Prefabs > Cardboard***, ajouter le Prefab « GvrReticlePointer » à la vue ***Hierarchy*** de la scène sous l'objet « Main Camera ».
   Ce Prefab affichera un cercle sur l'écran qui nous indiquera où nous regardons.

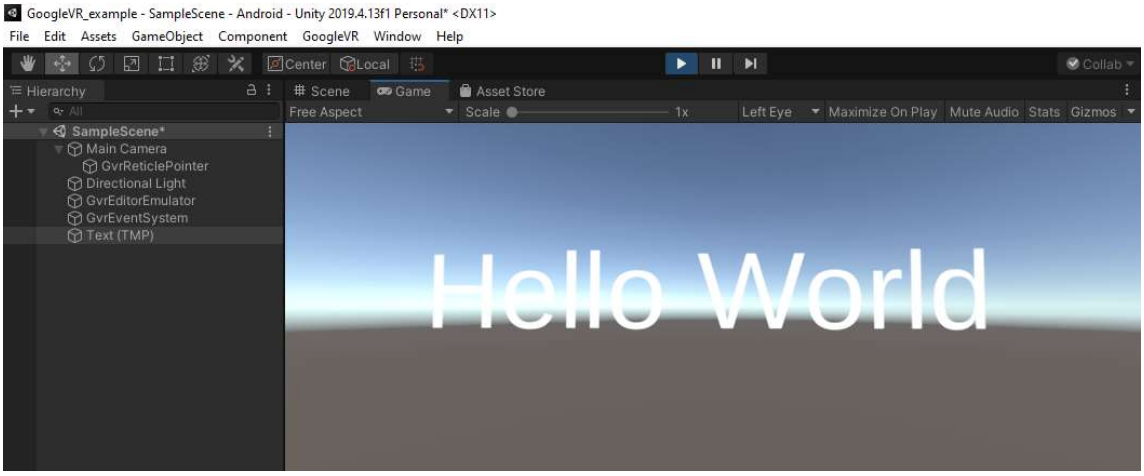
8. Tester votre projet (Play). Faire déplacer la caméra en appuyant sur la touche « Alt » et faisant déplacer la souris. Le cercle indiquant le regard de l'utilisateur sera visualisé.



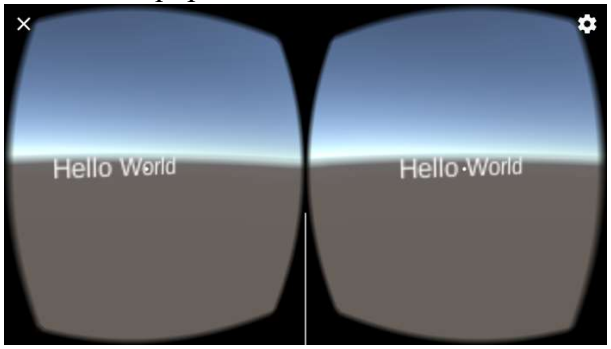
9. Maintenant, on va ajouter un objet à la scène.

- Placer la caméra à la position (0,0,0) pour qu'elle soit focalisée sur le centre de la scène.
- Ajouter un texte à partir du menu ***GameObject > 3D Object***, pour faire ceci, vous devez importer les ressources Text Mesh Professional (TMP).
  Ecrire le texte « Hello World » au niveau de l'input de cet objet.
- Placer cet objet à la position (0, 0, 10), pour qu'il sera visualisé par la caméra.

10. Tester de nouveau votre projet.

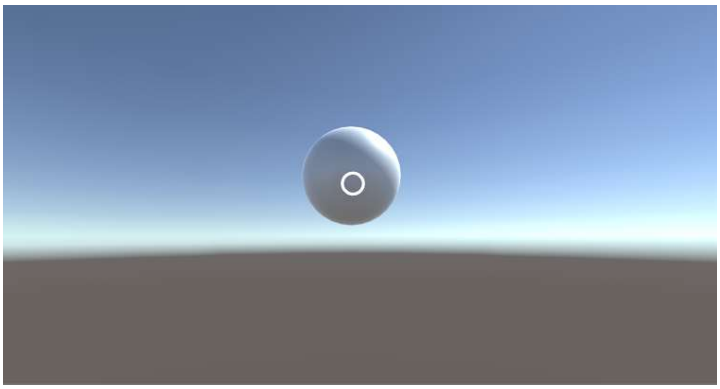


11. Déployer l'application sur un équipement mobile Android et tester.



12. Maintenant, on veut rendre l'application plus interactive.

- Supprimer le texte et ajouter des sphères et des cubes à la scène.
- Pour interagir avec ces objets, on doit définir comment la caméra doit voir ses objets. Sélectionner la caméra et ajouter le composant Script « GvrPointerPhysicsRaycaster » juste en tapant son nom dans la barre de recherche. Ceci va attacher un raycaster à la caméra.
- Pour définir les objets qui déclencheront ce raycaster, ajouter à chaque objet le composant « Event Trigger ».
- Tester l'application. Vous allez remarquer que le cercle s'agrandit à chaque fois que vous regardez un de vos objets.



- Pour que l'objet réagi au moment où il est déclenché par un raycaster, créer un nouveau script « ObjectBehavior » dans lequel vous écrivez le code suivant permettant d'agrandir l'objet. Ajouter ce script aux objets créés précédemment.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;

[RequireComponent(typeof(EventTrigger))]
public class ObjectBehavior : MonoBehaviour {

        Vector3 m_start;
        Vector3 m_target;
        bool m_targeted = false;

        // Use this for initialization
        void Start () {
                // Add triggers.
                EventTrigger trigger = GetComponent<EventTrigger>();

                // EventTrigger entry for PointerEnter.
                EventTrigger.Entry enterEntry = new
EventTrigger.Entry();
                enterEntry.eventID = EventTriggerType.PointerEnter;
                enterEntry.callback.AddListener((data) => {
OnPointerEnter((PointerEventData) data);});
```

```
                // EventTrigger entry for PointerExit.
                EventTrigger.Entry exitEntry = new
EventTrigger.Entry();
                exitEntry.eventID = EventTriggerType.PointerExit;
                exitEntry.callback.AddListener((data) => {
OnPointerExit((PointerEventData) data); });

                // Add the entries to the Event Trigger.
                trigger.triggers.Add(enterEntry);
                trigger.triggers.Add(exitEntry);

                // Set the initial and target scales.
                m_start = transform.localScale;
                m_target = m_start * 1.3f;
        }

        // Update is called once per frame
        void Update () {
                // Figure out which scale to move towards.
                Vector3 target = m_start;
                if (m_targeted) {
                        target = m_target;}

                // Move towards said scale.
          transform.localScale =
          Vector3.MoveTowards(transform.localScale, target,
          Time.deltaTime);
        }

        private void OnPointerEnter(PointerEventData data) {
                m_targeted = true;
        }

        private void OnPointerExit(PointerEventData data) {
                m_targeted = false;
        }
}
```

- Tester de nouveau l'application.