Golpe WebXR

José Luis Carreño Arteaga
jcarreno53@yahoo.com.mx

Permiso de los módulos de Grupo Web

Definición de variables XR y WebGL

```
// XR globals.
let xrButton = null;
let xrRefSpace = null;
let xrViewerSpace = null;
let xrHitTestSource = null;

// WebGL scene globals.
let gl = null;
let renderer = null;
let scene = new Scene();
scene.enableStats(false);

let arObject = new Node();
arObject.visible = false;
scene.addNode(arObject);

let flower = new Gltf2Node({url: 'media/gltf/sunflower/sunflower.gltf'});
arObject.addNode(flower);

let reticle = new Gltf2Node({url: 'media/gltf/reticle/reticle.gltf'});
reticle.visible = false;
scene.addNode(reticle);
```

Modelo a ser desplegado con
El hit de la pantalla

Retícula guía

Definición de variables XR y WebGL

```
// Tener una sombra debajo de un objeto ayuda a anclarlo
// en el mundo sin agregar complejidad
let shadow = new DropShadowNode();
vec3.set(shadow.scale, 0.15, 0.15, 0.15);
arObject.addNode(shadow);
```

Sombra de anclaje,
El objeto tendrá una escalar de 0.15

```
const MAX_FLOWERS = 30;
let flowers = [];
```

Máximo de objetos a ser colocados en la escena

Arreglos de objetos funcionando como cola de información

```
// Asegurar que el background es transparente para realidad aumentada.
scene.clear = false;
```

Funciones

```
function initXR() {
    xrButton = new WebXRButton({
    onRequestSession: onRequestSession,
    onEndSession: onEndSession,
    textEnterXRTitle: "START AR",
    textXRNotFoundTitle: "AR NOT FOUND",
    textExitXRTitle: "EXIT  AR",
  });
  document.querySelector('header').appendChild(xrButton.domElement);


  if (navigator.xr) {
    navigator.xr.isSessionSupported('immersive-ar')
                .then((supported) => {
      xrButton.enabled = supported;
    });
  }
}

function onRequestSession() {
  return navigator.xr.requestSession('immersive-ar', {requiredFeatures: ['local', 'hit-test']})
                    .then((session) => {
    xrButton.setSession(session);
    onSessionStarted(session);
  });
}
```

Funciones

```
function onSessionStarted(session) {
        session.addEventListener('end', onSessionEnded);
        session.addEventListener('select', onSelect);
         if (!gl) {
            gl = createWebGLContext({
                 xrCompatible: true
            });


            renderer = new Renderer(gl);
            scene.setRenderer(renderer);
         }


         session.updateRenderState({ baseLayer: new XRWebGLLayer(session, gl) });
         // Queremos proyectar un rayo desde la posición del observados
         // y renderizar una reticula en donde se intersecta con una superficie real.
         // Para hacer esto, obtenemos el espacio del observador y luego creamos un
         // hitTestSource que lo rastrea.
         session.requestReferenceSpace('viewer').then((refSpace) => {
           xrViewerSpace = refSpace;
           session.requestHitTestSource({ space: xrViewerSpace }).then((hitTestSource) => {
              xrHitTestSource = hitTestSource;
           });
         });


         session.requestReferenceSpace('local').then((refSpace) => {
           xrRefSpace = refSpace;


           session.requestAnimationFrame(onXRFrame);
         });
}
```

Funciones

```
function onEndSession(session) {
  xrHitTestSource.cancel();
  xrHitTestSource = null;
  session.end();
}


function onSessionEnded(event) {
  xrButton.setSession(null);
}


// Agregamos un nuevo objeto a la escena
// en la transformacion especificada
function addARObjectAt(matrix) {
  let newFlower = arObject.clone();
  newFlower.visible = true;
  newFlower.matrix = matrix;
  scene.addNode(newFlower);


  flowers.push(newFlower);


   // Por razones de rendimiento, si agregamos demasiados objetos,
    //comenzamos a eliminar los mas antiguos para evitar que la
  // complejidad de la escena crezca demasiado.
  if (flowers.length > MAX_FLOWERS) {
    let oldFlower = flowers.shift();
    scene.removeNode(oldFlower);
  }
}
```

La función shift genera una cola de información

Funciones

```
let rayOrigin = vec3.create();
let rayDirection = vec3.create();
function onSelect(event) {
    if (reticle.visible) {
        // La reticula ya debera estar posicionada en el ultimo punto de impacto,
        // asi que podemos usar su matriz para evitar una llamada innecesaria a
        // event.frame.getHitTestResults.
          addARObjectAt(reticle.matrix);
    }
}


// Se llama cada vez que una XRSession solicita
// que se dibuje un nuevo cuadro.
function onXRFrame(t, frame) {
     let session = frame.session;
     let pose = frame.getViewerPose(xrRefSpace);
     reticle.visible = false;
     // Si tenemos una fuente de prueba de impacto, obtenemos sus resultados para el fotograma
     // y usamos la pose para mostrar una reticula en la escena.
    if (xrHitTestSource && pose) {
        let hitTestResults = frame.getHitTestResults(xrHitTestSource);
        if (hitTestResults.length > 0) {
          let pose = hitTestResults[0].getPose(xrRefSpace);
          reticle.visible = true;
          reticle.matrix = pose.transform.matrix;
        }
    }
      scene.startFrame();
      session.requestAnimationFrame(onXRFrame);
      scene.drawXRFrame(frame, pose);
      scene.endFrame();
}
```

Se inicia realidad aumentada

```
    // Iniciamos la aplicacion XR
    initXR();
  </script>
 </body>
</html>
```

Referencias Bibliográficas