

Examen

-Partie I

1/a-Définir un shader

- b- A quoi sert l'utilisation des shader
- c- Quels sont les types de shader utilisé en WebGL.
- d- Quel est le langage de shader.

2/-a- A quoi sert les rendu

- b-Comment faire le rendu en WebGL(Three.js),
- c-Comment faire le rendu en Blender

3/-donnez l'historique des langages et les formats 3D sous forme d'un graphe hiérarchique

4- Quels sont les prototypes de la bibliothèque three.js

5- A quoi sert OrbitControls.js

6-A quel langage appartient la fonction requestAnimationFrame : WebGL, Html5, Javascript...

7- Voici le code suivant

```
Shape {  
    geometry Box {  
        size 4 8 3  
    }  
    appearance Appearance {  
        material Material {  
            diffuseColor 1 0 1  
        }  
    }  
}
```

- Quel est le langage utilisé.
- Donner le résultat de ce code .
- Ecrire le code équivalent en three.js.

8-Quels sont les thèmes des exposés abordés dans ce module

9-Quel est votre thème

10-Donnez les grands points abordés dans chaque thème.

II-Partie TP Voici le code suivant : **Ajouter les instructions qui manque**

<!DOCTYPE

E html>

```
<html lang="en">  
    <head>  
        <title>three.js webgl - io blender - vertex colors</title>  
        <meta charset="utf-8">  
        <meta name="viewport" content="width=device-width, user-scalable=no,  
        minimum-scale=1.0, maximum-scale=1.0">  
    </head>  
    <body>  
        <div id="container"></div>  
        <div id="info"><a href="http://threejs.org" target="_blank"  
        rel="noopener">three.js</a> webgl - io blender - vertex colors</div>
```

```

<script src="../build/three.js"></script>
<script src="js/Detector.js"></script>
<script src="js/libs/stats.min.js"></script>
<script>
if ( ! Detector.webgl ) Detector.addGetWebGLMessage();

    var container, stats;
    .....(1)

    var mesh, mesh2, mesh3, light;
    var mouseX = 0, mouseY = 0;
    var windowHalfX = window.innerWidth / 2;
    var windowHalfY = window.innerHeight / 2;
    init();
    animate();

function init() {
container = document.getElementById( 'container' );
.....
....(2)
camera.position.z = 1800;
.....(3)
light = new THREE.DirectionalLight( 0xffffff );
light.position.set( 0, 0, 1 ).normalize();
.....(4)
.....(5)
loader.load( "models/json/cubecolors/cubecolors.json", createScene1 );
loader.load( "models/json/cubecolors/cube_fvc.json", createScene2 );
renderer = new THREE.WebGLRenderer( { antialias: true } );
renderer.setPixelRatio( window.devicePixelRatio );
renderer.setSize( window.innerWidth, window.innerHeight );
container.appendChild( renderer.domElement );
stats = new Stats();
container.appendChild( stats.dom );
document.addEventListener( 'mousemove', onDocumentMouseMove,
false );

                                //
window.addEventListener( 'resize', onWindowResize, false );

                                }

function onWindowResize() {
windowHalfX = window.innerWidth / 2;
windowHalfY = window.innerHeight / 2;
camera.aspect = window.innerWidth / window.innerHeight;
camera.updateProjectionMatrix();

```

```

renderer.setSize( window.innerWidth, window.innerHeight );
    }

    function createScene1( geometry, materials ) {
    mesh = new THREE.Mesh( geometry, materials );
    mesh.position.x = 400;
    mesh.scale.x = mesh.scale.y = mesh.scale.z = 250;
    .....(6)
    }

    function createScene2( geometry, materials ) {
    .....(7)

    mesh2.position.x = - 400;
    mesh2.scale.x = mesh2.scale.y = mesh2.scale.z = 250;
    scene.add( mesh2 );
    }

    function onDocumentMouseMove( event ) {
    mouseX = ( event.clientX - windowHalfX );
    mouseY = ( event.clientY - windowHalfY );
    }

    //

    function animate() {
    .....(8)

    render();
    stats.update();
    }

    function render() {
    camera.position.x += ( mouseX - camera.position.x ) * 0.05;
    camera.position.y += ( - mouseY - camera.position.y ) * 0.05;
    camera.lookAt( scene.position );

    if ( mesh ) {
    mesh.rotation.x += 0.01;
    mesh.rotation.y += 0.01;
    }

    if ( mesh2 ) {
    mesh2.rotation.x += 0.01;
    mesh2.rotation.y += 0.01;
    }
    .....(9)
    }
    </script>
</body>
</html>

```

