INB381 – Load an OBJ File

Blender

1. Open Blender and delete the cube
2. Add new shape choose ‘Mesh’ and select the ‘Monkey’
3. Switch from ‘Object Mode’ to ‘Edit Mode’ and click ‘Subdivide’ a couple of times to generate more polygons.
4. Switch back to ‘Object Mode’ and choose ‘Smooth’
5. Select ‘Export ‘ and ‘WaveFront” obj.
6. From export options uncheck all and select only ‘Write Normals’, Triangulate Faces’ and ‘Objects as OBJ objects’.
7. Choose directory and filename (monkey.obj) -> click ‘Export’
8. Quit Blender

Demonstration

1. Using the following parser to load the monkey.obj to create an application that renders your monkey, when complete show your tutors.
2. Now export another object shape of your choice from blender and simply change the name of the file.

The function is capable of loading any shape without the need for any change.

Parsing the OBJ (Model)

WebGL does not load meshes. It is your role as a programmer to load the data from a resource file and convert it into vertex data that WebGL can use. You will now create a parser that loads mesh data from a Blender 'WaveFront' object file.

Function

The following function will parse the obj file.

```
function loadMeshData(string) {
    var lines = string.split("\n");
    var positions = [];
    var normals = [];
    var vertices = [];

    for ( var i = 0 ; i < lines.length ; i++ ) {
        var parts = lines[i].trimRight().split(' ');
        if ( parts.length > 0 ) {
            switch(parts[0]) {
                case 'v':  positions.push(
                    vec3.fromValues(
                        parseFloat(parts[1]),
                        parseFloat(parts[2]),
                        parseFloat(parts[3])
                    ));
                    break;
                case 'vn':
                    normals.push(
                        vec3.fromValues(
                            parseFloat(parts[1]),
                            parseFloat(parts[2]),
                            parseFloat(parts[3])));
                    break;
                case 'f': {
                    var f1 = parts[1].split('/');
                    var f2 = parts[2].split('/');
                    var f3 = parts[3].split('/');
                    Array.prototype.push.apply(
                        vertices, positions[parseInt(f1[0]) - 1]);
                    Array.prototype.push.apply(
                        vertices, normals[parseInt(f1[2]) - 1]);
                    Array.prototype.push.apply(
                        vertices, positions[parseInt(f2[0]) - 1]);
                    Array.prototype.push.apply(
                        vertices, normals[parseInt(f2[2]) - 1]);
                    Array.prototype.push.apply(
                        vertices, positions[parseInt(f3[0]) - 1]);
                    Array.prototype.push.apply(
                        vertices, normals[parseInt(f3[2]) - 1]);
                    break;
                }
            }
        }
    }

    return {
        primitiveType: 'TRIANGLES',
        vertices: new Float32Array(vertices),
        vertexCount: vertices.length / 6,
        material: {ambient: 0.2, diffuse: 0.5, shininess: 10.0}
    };
}
```