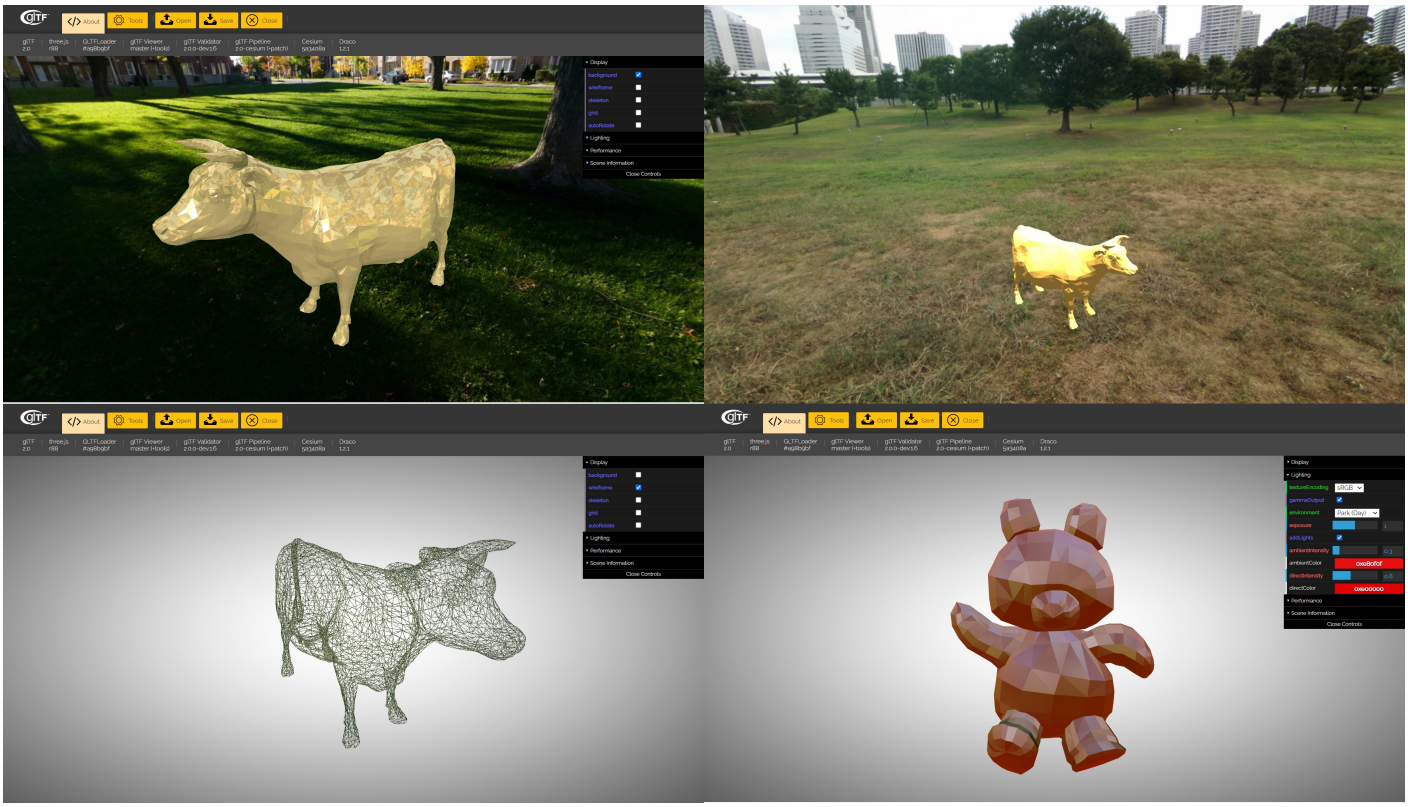
CS460 Fall 2021
Name: Tung Duong
Github Username: TungDuong2
Due Date: 12/06/2021

Assignment 10: glTF!

We will load our favorite mesh from a file and then convert it to a valid glTF file. You can choose if you want to do this assignment in JavaScript or in Python. In class, we will use Python (see example colab `https://cs460.org/shortcuts/33/`).



Starter code for assignment 10. After pulling from upstream, there is the folder `10` in your fork. This folder contains an `index.html` file that uses JavaScript to make glTF JSON. This folder also contains a `gltf.py` script that you can run with `python gltf.py` to output the glTF JSON. As a start for this assignment, both versions create an identical valid glTF JSON structure holding a single triangle (see screenshot above).

Part 1 (1 points): Please decide which language you will use: JavaScript or Python. Python might be a bit easier to load and parse an existing file—with JavaScript we need to use Ajax to load the existing mesh and parse it (or as option 3: use a Three.js loader and grab the vertices/indices from there). For parsing files with Python look here: `https://tutorial.eyehunts.com/python/python-read-file-line-by-line-readlines/` For using Javascript and Ajax look here: `https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/Using_XMLHttpRequest`.

Part 2 (15 points): Load a mesh from an external file. A .PLY or .OBJ file might be the easiest to parse.

Part 3 (20 points): Parse all vertices from the loaded mesh and create the VERTICES array and base64 code.

Part 4 (20 points): Parse all indices from the loaded mesh and create the INDICES array and base 64 code.

Part 5 (10 points): Calculate all required fields for the glTF file (as we did in class) and generate the glTF JSON code. Store the glTF JSON code in a glTF file.

Part 6 (5 points): Please make sure the glTF file is valid using `http://github.khronos.org/glTF-Validator/`.

Part 7 (5 points): Visualize the glTF file using `https://gltf.insimo.com/`. You might have to choose the wireframe display option since the glTF file does not include material (Display -> Wireframe, in the dat.GUI). **Please replace the screenshot above.**

Part 8 (5 points): Add the glTF file to your fork.

Part 9 (10 points): Choose a final project—either an existing one from `https://cs460.org/assignments/final/` or a new one. Please list the project here and in the link. If working as a team, assemble your team and list the team members below and in the link.

I decided to create a Spacecraft game (demo) on THREE.js by applying relevant knowledge from this course such as: OOP, sky box, glTF, camera, lights, etc.

`https://tungduong2.github.io/cs460student/final/`

Part 10 (9 points): Make sure this PDF and your glTF file are in your fork on github. Then, please send a pull request.

Bonus (33 points):

Part 1 (15 points): Please add any kind of material to the glTF file. For this, you would have to read the specs or google for examples :)

gltf.py

```
],

"materials": [
    {
        "name": "gold",
        "pbrMetallicRoughness": {
            "baseColorFactor": [1.000, 0.766, 0.336, 1.0],
            "metallicFactor": 0.5,
            "roughnessFactor": 0.1
        }
    }
],

"meshes": [
    {
        "primitives": [{
            "mode": 4,
            "attributes": {
                "POSITION": 0
            },
            "indices": 1,
            "material": 0
        }]
    }
],
```

Part 2 (18 points): Write THREE.js code that displays your glTF file using the `THREE.GLTFLoader`.

```
// Instantiate a loader
const loader = new THREE.GLTFLoader();

// Load a glTF resource
loader.load(
  // resource URL
  'cow-nonormals.gltf',
  // called when the resource is loaded
  function ( gltf ) {

    const model = gltf.scene.children[0];
    model.scale.setScalar(5);
    model.position.set(0 ,-20, 0);

    scene.add( gltf.scene );

    gltf.animations; // Array<THREE.AnimationClip>
    gltf.scene; // THREE.Group
    gltf.scenes; // Array<THREE.Group>
    gltf.cameras; // Array<THREE.Camera>
    gltf.asset; // Object

  },
  // called while loading is progressing
  function ( xhr ) {

    console.log( ( xhr.loaded / xhr.total * 100 ) + '% loaded' );

  },
  // called when loading has errors
  function ( error ) {

    console.log( 'An error happened' );

  }
);
```