Joanne Wardell

***Finding Teapots Final Project***

Finding Teapots is a space explorer game in which the player is to find as many teapots as possible. The organization of this program and its components consist of an index.html landing page, the main.js file, the Utah Teapot Geometry file, the stats.js file, and the THREE.js API file. Although the game is small, it allows the user to move freely and explore the graphical space with no boundaries.

The game begins by displaying a loading screen as long as the document ready state is not complete. The entire game has been enclosed within the window onload function to avoid global variables. In fact, all computations and initializations are enclosed in functions to make debugging and selectivity easier. The alert message is displayed at first. Then, the THREE.js scene is initialized. The planet-like spheres are drawn and pushed into an array which are then added to the scene. Next, the teapot is rendered and added to the scene along with five surrounding point light sources to increase visibility. Four spheres with textures are loaded and added to the scene. Finally, the render loop checks game logic and asynchronously renders the scene. Event listeners are set for mouse movements, key presses, and key releases.

The game logic determines if the user is close to the teapot by comparing the camera and teapot positions. If the camera comes within a certain epsilon of the teapot's vertices, the teapot has been found and moves to a random location on the map. The teapot's color is a new random color for each new game session. The event handlers update the camera side movements, or strafe, and the camera pitch and yaw, independently and in that order. The camera lookat and up vectors are also managed and updated when the mouse is moved. The stars are spheres in a THREE point cloud with randomly placed stars within the spherical cloud. The planets are not part of the cloud, but are placed in random locations in the same range as the stars in the point cloud.

Rendering the teapot presented itself as a challenge in the beginning because most of the files that I found for its vertex information were json files. Loading the json version of the teapot was troublesome and often would not work. Luckily, the TeapotBufferGeometry.js file made things easier and rendered a beautiful teapot. The textures were hard to work with because of issues with using cross-origin data and CORS policy. I ended up developing everything locally and this resolved the issue. Finding a nice balance of randomness was also annoying, but a decent scaling factor for Math.random was figured out.

I think this game has a lot of potential and I would like to expand upon it by making more objectives and increasing difficulty. There are other steps that I could take to increase performance such as less expensive graphics for space materials, better mouse controls, and the ability to save state for later gameplay. Personally, I would like to see this game develop into some sort of light weight role-playing game.

***My suggestions and disclaimers:***

I developed this game on three different operating systems and depending on the speed at which its rendered, it looks somewhat different per machine. Some adjustments might need to be made to the size of the point cloud as the screen gets larger. I did the most development of this game on Ubuntu in Chrome/Chromium and this is where I have noticed the best results.

Please run this locally as the textures will not load otherwise. With chrome configured in the PATH environment variable, run **`chrome --allow-file-acces-from-files`** in the command prompt or terminal.
One can also make a simple http server on port 8000 using python. In the terminal issue the command,

```
python -m http.server 8000
```

Another way to do this with node.js is to install http-server:

```
npm install http-server -g
http-server . -p 8000
```

while you are in the ./teapot directory.
Navigate to http://localhost:8000