Solarmax 3D

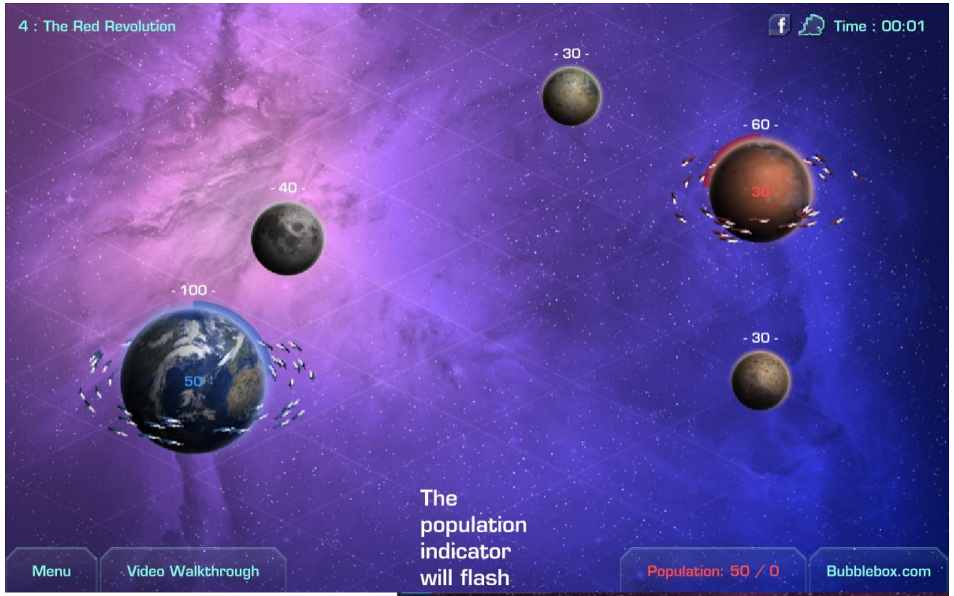
João Fonseca (103154)

Introduction to Computer Graphics – 2022/2023 – Project

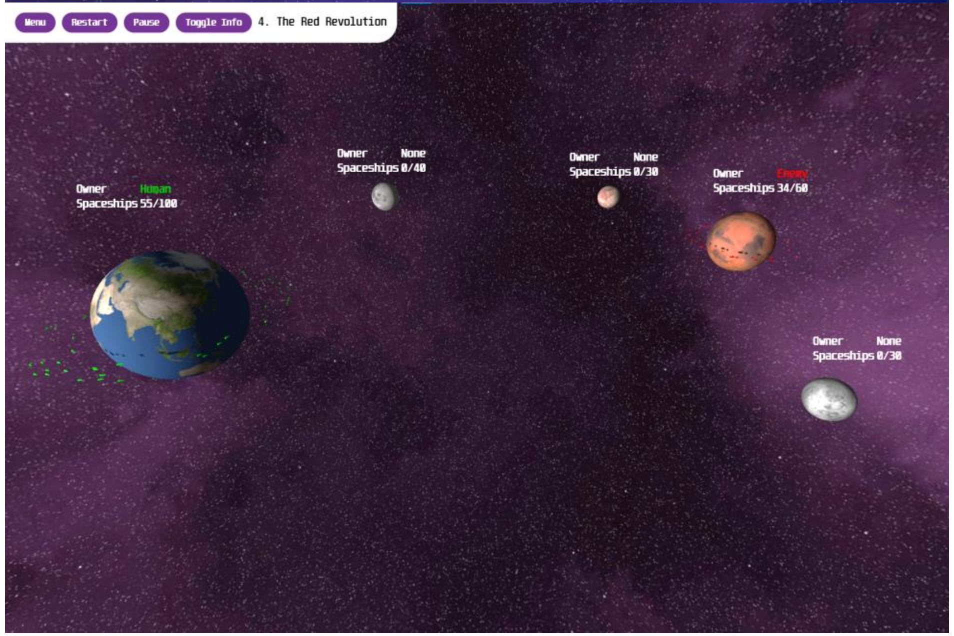
Introduction

- Inspired on **Solarmax (2D)**
- Space-themed strategy game
- Players have a fleet of spaceships
- Goal: colonize all planets in the level

- Created using basic Three.js



The population indicator will flash



Owner Human
Spaceships 55/100

Owner None
Spaceships 0/40

Owner None
Spaceships 0/30

Owner Enemy
Spaceships 34/60

Owner None
Spaceships 0/30

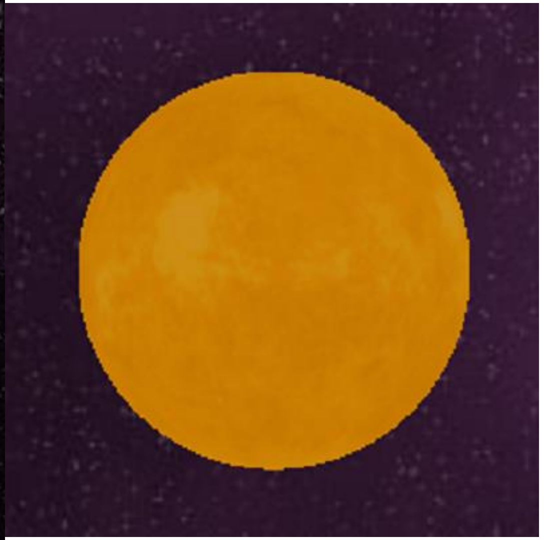
Deployment

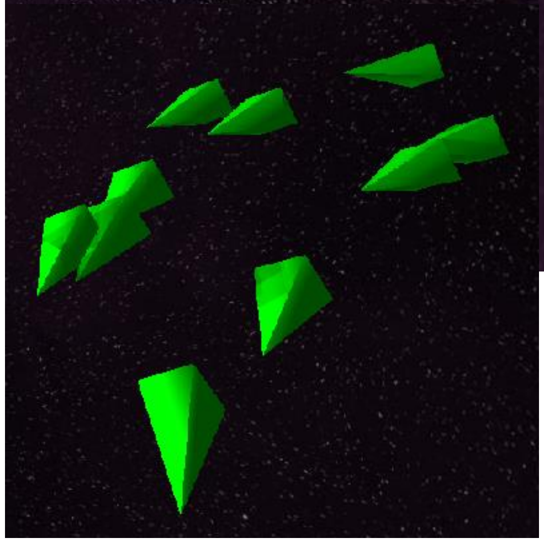
https://joaompfonseca.github.io/icg-project/

Models

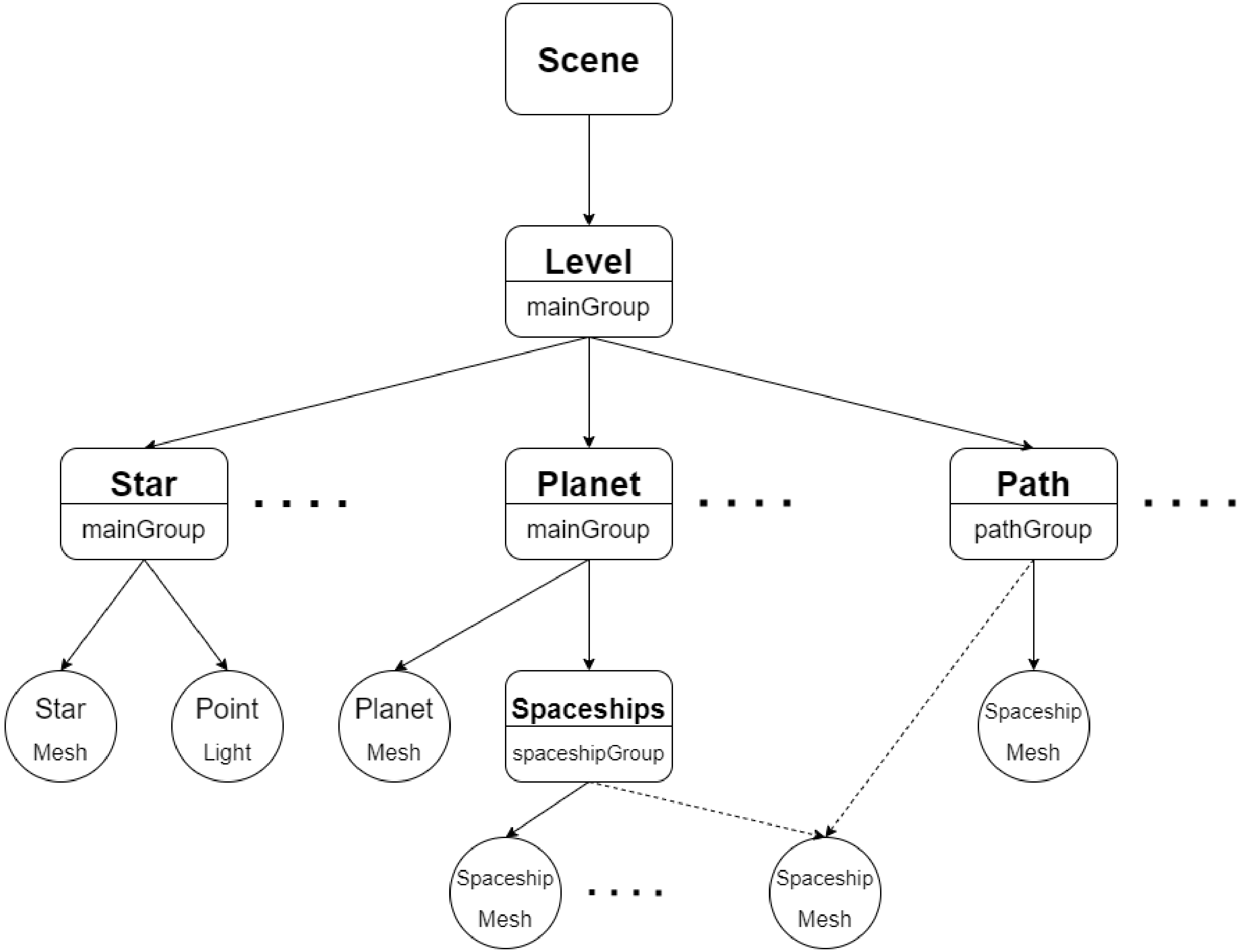
- Planets & Stars
  - SphereGeometry
  - MeshPhongMaterial (+ custom texture)

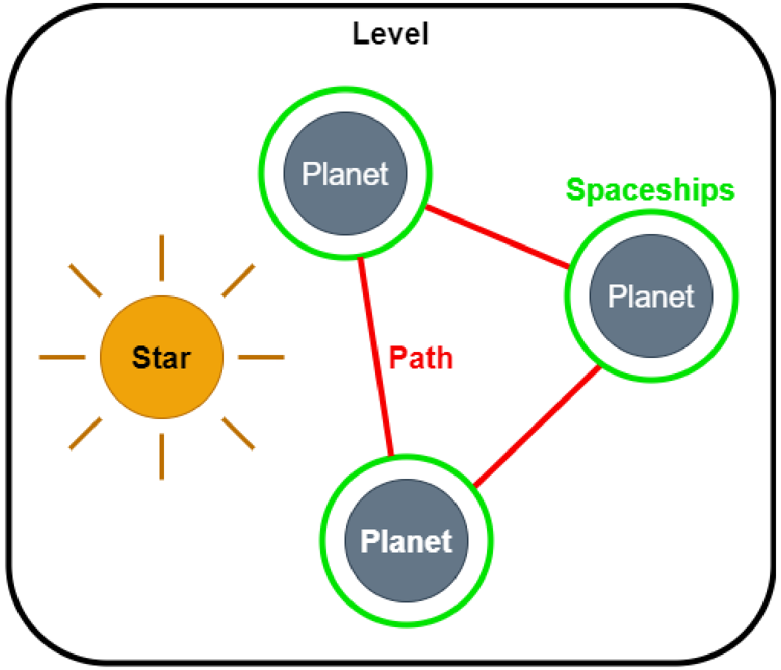
- Spaceships
  - ConeGeometry
  - MeshPhongMaterial (+ emissive of intensity 0.2)





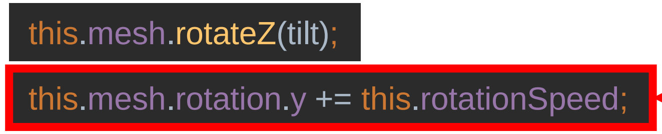
Scene Graph


Scene
Level
mainGroup
Star
mainGroup
. . . .
Planet
mainGroup
. . . .
Path
pathGroup
. . . .
Star
Mesh
Point
Light
Planet
Mesh
Spaceships
spaceshipGroup
Spaceship
Mesh
Spaceship
Mesh
. . . .
Spaceship
Mesh


Level
Planet
Spaceships
Planet
Star
Path
Planet

Animation

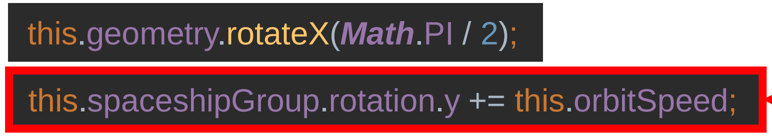
- Planets & Stars
  - Fixed tilt (around Z)
  - Rotation on own axis (around Y)



```
this.mesh.rotateZ(tilt);
this.mesh.rotation.y += this.rotationSpeed;
```

Ball#animate

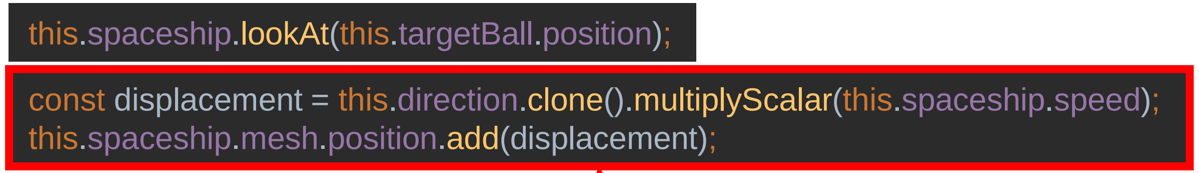
- Spaceships
  - Fixed rotation (around X)
  - Orbit around planets / rotation of group (around Y)



```
this.geometry.rotateX(Math.PI / 2);
this.spaceshipGroup.rotation.y += this.orbitSpeed;
```

Spaceship#animate

- Paths
  - Spacehips points to destination planet
  - Spaceship moves towards planet



```
this.spaceship.lookAt(this.targetBall.position);
const displacement = this.direction.clone().multiplyScalar(this.spaceship.speed);
this.spaceship.mesh.position.add(displacement);
```

Path#animate

- Entities have animate function that is called on each frame

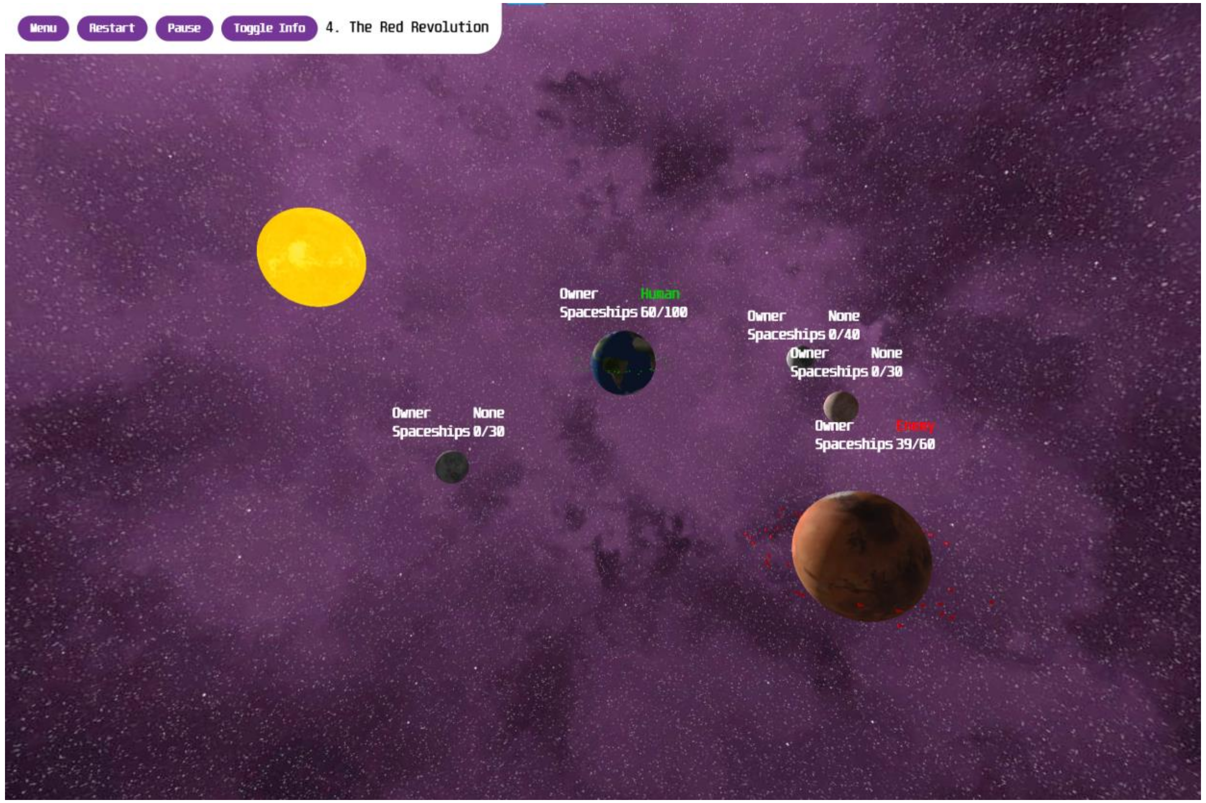
Illumination

- Ambient lighting is present in the environment
  - AmbientLight
  - White light
  - Intensity of 0.1

```
const ambientLight = new THREE.AmbientLight(0xffffff, 0.1);
```

- Star emits light to the entire level
  - PointLight
  - White light
  - Intensity of 0.1
  - Distance of 10000

```
const light = new THREE.PointLight(0xffffff, 1, 10000);
```



Owner Human
Spaceships 60/100

Owner None
Spaceships 0/40

Owner None
Spaceships 0/30

Owner None
Spaceships 0/30

Owner Enemy
Spaceships 39/60

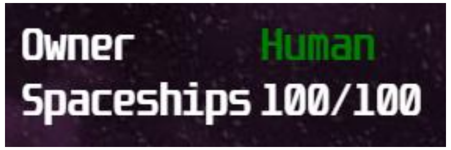
- All objects receive and cast shadows between eachother

User Interaction

- UI Elements
  - Main menu
  - Navbar buttons
  - Labels above planets
  - Colonization progress
  - Information panel

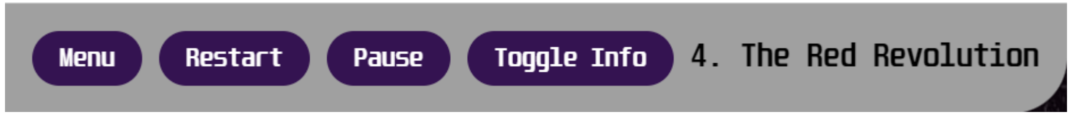
- Mouse controls
  - OrbitControls: rotation, zoom
  - Raycaster & OutlinePass: interaction with planets

- Keyboard shortcuts
  - M: Menu
  - R: Restart
  - P: Pause
  - I: Toggle Info
  - N: Next Level

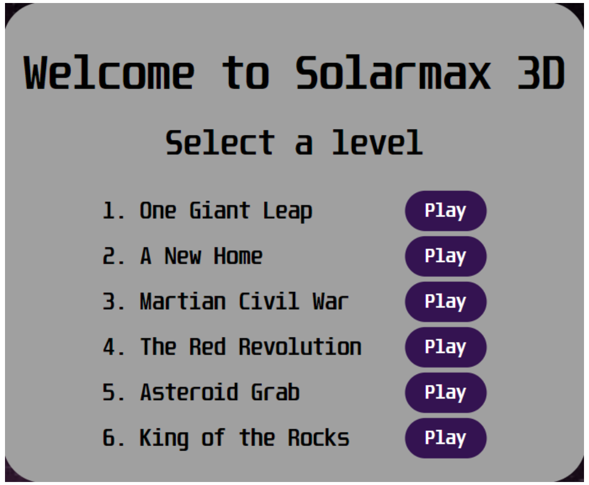


Owner Human

Spaceships 100/100



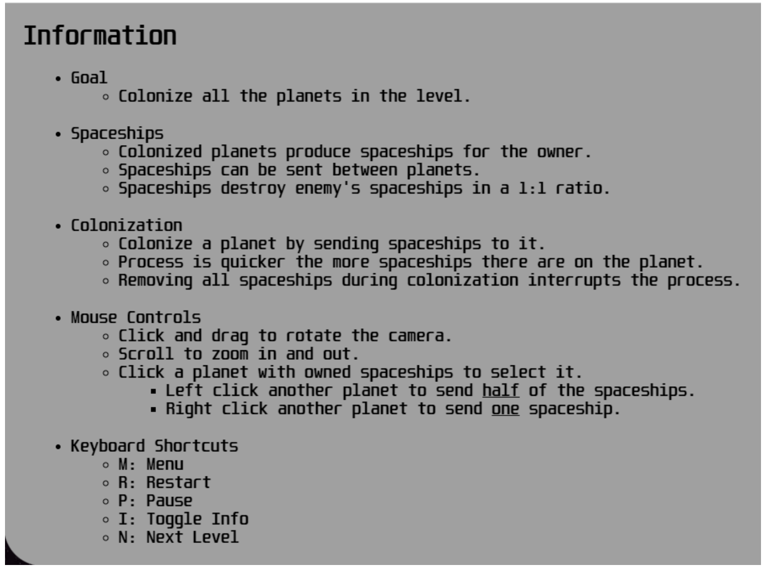
4. The Red Revolution



Welcome to Solarmax 3D

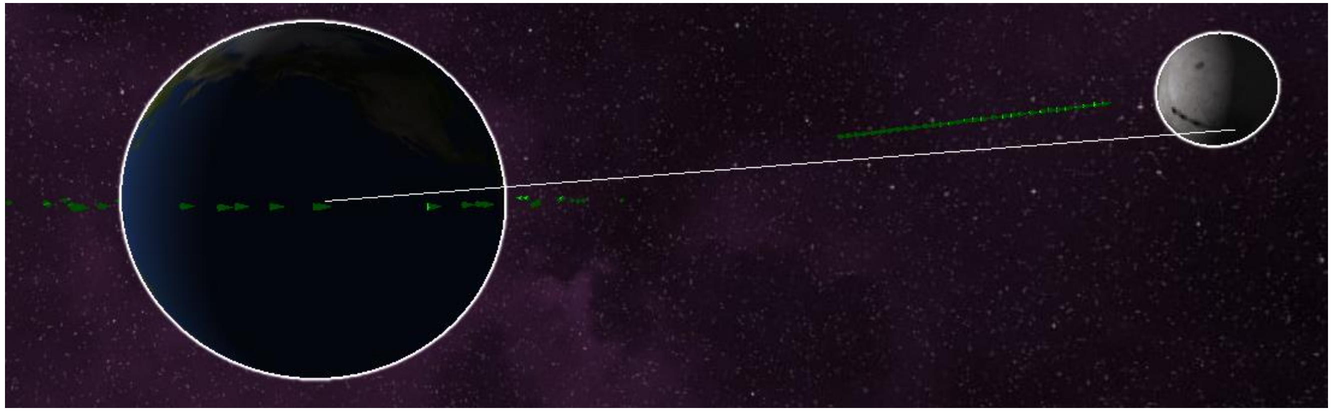
Select a level

1. One Giant Leap
2. A New Home
3. Martian Civil War
4. The Red Revolution
5. Asteroid Grab
6. King of the Rocks



Information

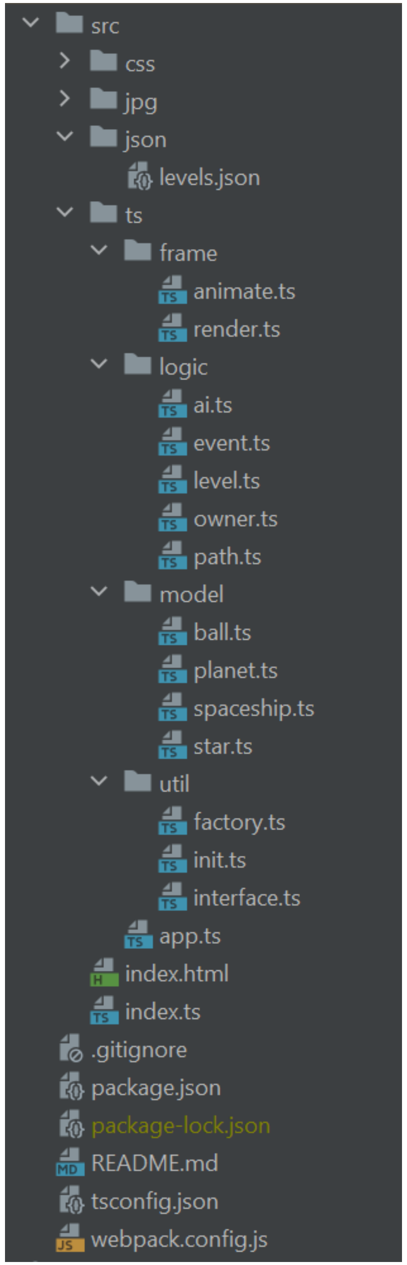
- Goal
  - Colonize all the planets in the level.
- Spaceships
  - Colonized planets produce spaceships for the owner.
  - Spaceships can be sent between planets.
  - Spaceships destroy enemy's spaceships in a 1:1 ratio.
- Colonization
  - Colonize a planet by sending spaceships to it.
  - Process is quicker the more spaceships there are on the planet.
  - Removing all spaceships during colonization interrupts the process.
- Mouse Controls
  - Click and drag to rotate the camera.
  - Scroll to zoom in and out.
  - Click a planet with owned spaceships to select it.
    - Left click another planet to send half of the spaceships.
    - Right click another planet to send one spaceship.
- Keyboard Shortcuts
  - M: Menu
  - R: Restart
  - P: Pause
  - I: Toggle Info
  - N: Next Level



Development

- Coded in **TypeScript** (transpiled to JavaScript)
    - Strong static typing – variables, function parameters and return values
    - ECMAScript support (ES6) – modern JavaScript features
    - Can compile to any JavaScript version
- Follows some OOP practices
    - Classes
    - Design patterns - factory
- Bundled with **Webpack**
    - Optimization of the code
    - **bundle.js ~ 499 KB**
- Deployed in **GitHub Pages**

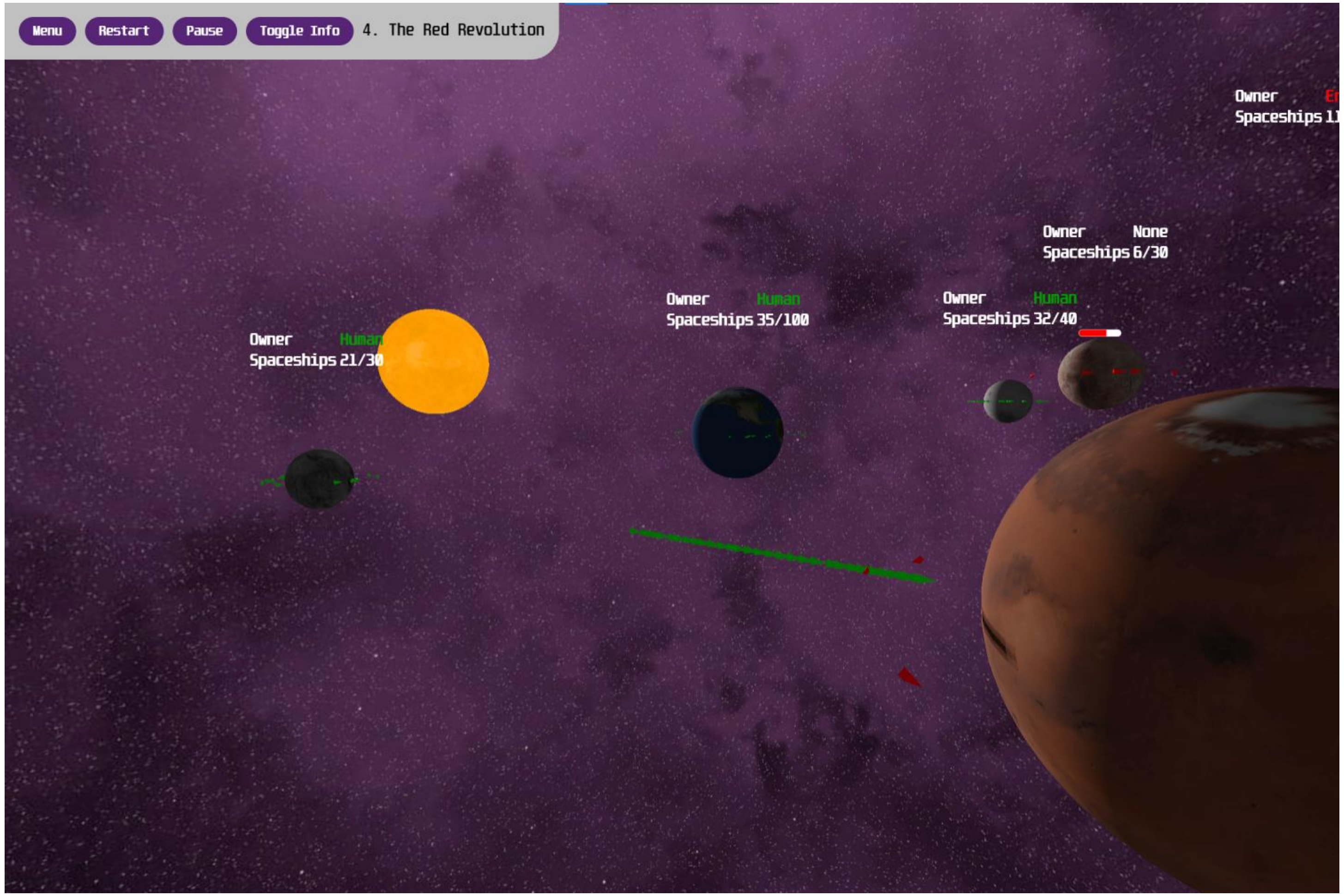
- <u>Problems: creating UI elements (menu, navbar) using basic HTML, CSS and JS</u>



```
src
  css
  jpg
  json
    levels.json
  ts
    frame
      animate.ts
      render.ts
    logic
      ai.ts
      event.ts
      level.ts
      owner.ts
      path.ts
    model
      ball.ts
      planet.ts
      spaceship.ts
      star.ts
    util
      factory.ts
      init.ts
      interface.ts
    app.ts
  index.html
  index.ts
.gitignore
package.json
package-lock.json
README.md
tsconfig.json
webpack.config.js
```

Conclusions

- Understanding of 3D graphics
- Practical experience with Three.js
- Future applications



References

- Examples from ICG practical exercises
    - https://elearning.ua.pt/
- Three.js Documentation
    - https://threejs.org/docs/
- Solar System Textures
    - https://solarsystemscope.com/textures/
- Tutorial: Hosting a Webpack Project with GH-Pages
    - https://learnhowtoprogram.com/intermediate-javascript/team-week/hosting-a-webpack-project-with-gh-pages
- Project repository on GitHub
    - https://github.com/joaompfonseca/icg-project
- Deployment on GitHub Pages
    - https://joaompfonseca.github.io/icg-project/