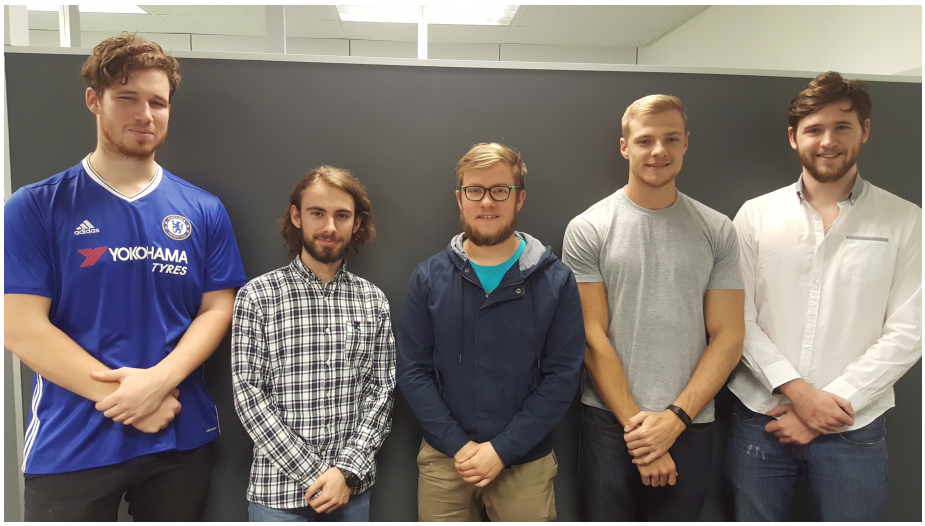

Absolute|0|

Voxc.js
Requirements Specification

Name	Student Number
Chris Dreyer	15072623
HD Haasbroek	15046657
Cameron Trivella	14070970
Pearce Jackson	14044342
Idrian van der Westhuizen	15078729

Contents

1 Introduction **2**
- 1.1 Purpose . . . 2
- 1.2 Product Scope . . . 2
- 1.3 References . . . 2

2 External Interface Requirements **3**
- 2.1 User Interfaces . . . 3
- 2.2 Software Interfaces . . . 3
- 2.3 Communications Interfaces . . . 3

3 System Features **4**
- 3.1 File upload with rules feature . . . 4
  - 3.1.1 Description and Priority . . . 4
  - 3.1.2 Stimulus/Response Sequences . . . 4
  - 3.1.3 Functional Requirements . . . 5
- 3.2 File upload without rules feature . . . 5
  - 3.2.1 Description and Priority . . . 5
  - 3.2.2 Stimulus/Response Sequences . . . 5
  - 3.2.3 Functional Requirements . . . 5

4 Other Nonfunctional Requirements **6**
- 4.1 Performance Requirements . . . 6
  - 4.1.1 Time to respond to an uploaded file . . . 6
  - 4.1.2 Respond to user interaction . . . 6
  - 4.1.3 Reliability . . . 7
  - 4.1.4 Maintainability . . . 7
  - 4.1.5 Portability . . . 7
  - 4.1.6 Scalibilty . . . 7
  - 4.1.7 Usability . . . 7
- 4.2 Security Requirements . . . 7
- 4.3 Quality Requirements . . . 8

1 Introduction

1.1 Purpose

This SRS document aims to stipulate the requirements of the voxc.js library to aid in the development process and to ensure that a functional and usable product is delivered.

1.2 Product Scope

Voxc.js is meant to be an easy to use and easy to maintain JavaScript library, similar to how three.js is a library for WebGL. The purpose of the project is to provide users a way to import Voxel models into a webpage that would be using the Voxc.js library and use these Voxel models as a coordinate system to create newly genereated mesh object according to a rules file with a specified structure. The user will then be able to export the textured and rendered object.

1.3 References

http://www.oskarstalberg.com/game/house/Index.html

https://voxel.codeplex.com/

https://pages.github.com/

http://threejs.org/

http://coffeescript.org/

http://es6-features.org/#Constants

http://www.typescriptlang.org/

http://www.codebelt.com/typescript/typescript-es6-modules-boilerplate/

http://giacomotag.io/typescript-webpack/

2 External Interface Requirements

2.1 User Interfaces

The users should be able to interface with voxc.js through our web interface that will be hosted on Github Pages. They will mainly be using a laptop or a desktop to interface with the library. However the library itself should be able to interface with any website on any device with a web browser that has support for WebGL.

This library should not require user profiles as their voxel objects are stored locally on browser storage and manipulated by the library. This implies that the user should only be led to one screen where they will be allowed to upload a voxel object file and a rules file if they choose to do so.

2.2 Software Interfaces

The system should incorporate several different languages in order to function. For the rules file, JSON objects should be used so that the users can manipulate the file in any text editor based on set conventions and structure.

MagicVoxel should be used for the creation and manipulation of voxel objects. The library should be able to handle the .obj filetype for the voxel objects.

The library should use TypeScript as the JavaScript variant to enable future developers and current developers to debug easier. The library should run on all major internet browsers that support WebGL.

2.3 Communications Interfaces

HTTP will be used to handle GET and POST requests and FTP will be used for file uploads and downloads.

3 System Features

3.1 File upload with rules feature

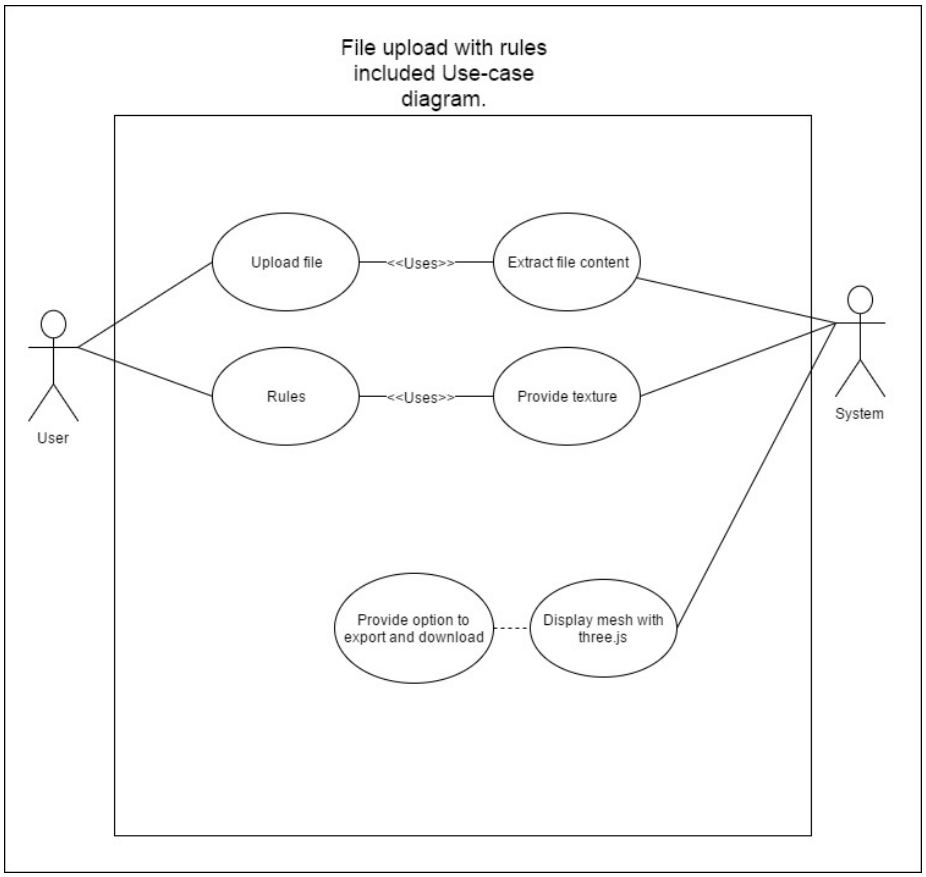

Figure 1: Use Case Diagram for User upload with rules

3.1.1 Description and Priority

This feature is of high priority since the user will not be able to have their exported file if they can not use the upload feature. The penalty would be a 9 out of 10. The systems response is also critical because if no textures are provided then the program fails to do what it is meant to do.

3.1.2 Stimulus/Response Sequences

Stimulus: The user uploads his file with the rules file included.
Response sequences: The system responds by using the files the user uploaded

to extract them.

The system then uses the user rule file to provide the textures. Next the system responds by displaying the mesh with three.js and then providing the user with the option to download it.

3.1.3 Functional Requirements

REQ-1: The user needs to be able to upload a file with rules.
REQ-2: The system must be able to export the files from the users upload.
REQ-3: The system has to provide the textures based on the users rule file.
REQ-4: The system needs to display the mesh with three.js.
REQ-5: The system must provide the functionality to download the mesh.

3.2 File upload without rules feature

3.2.1 Description and Priority

This feature is of high priority since the user will not be able to have their exported file if they can not use the upload feature.It is also of high priority to provide a rule file if none was uploaded otherwise there would be no rules to provide textures for.The penalty would be a 9 out of 10.The systems response is also critical because if no textures are provided then the program fails to do what it is meant to do.

3.2.2 Stimulus/Response Sequences

Stimulus: The user uploads his file with the rules file excluded.
Response sequences: The system responds by using the files the user uploaded to extract them.

The system then uses a rule file provided by the system to get the textures. Next the system responds by displaying the mesh with three.js and then providing the user with the option to download it.

3.2.3 Functional Requirements

REQ-1: The user needs to be able to upload a file without rules.
REQ-2: The system must be able to export the files from the users upload.
REQ-3: The system has to provide a rule file for the textures.
REQ-4: The system needs to display the mesh with three.js.
REQ-5: The system must provide the functionality to download the mesh.

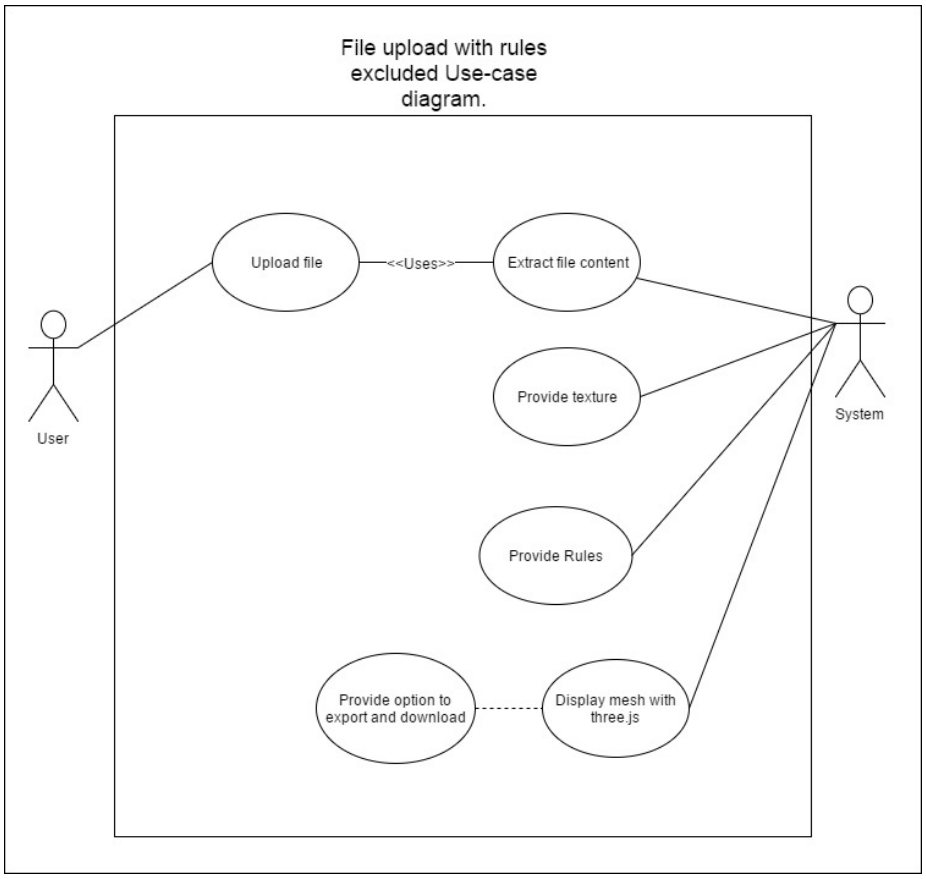

Figure 2: Use Case Diagram for User upload with rules

4 Other Nonfunctional Requirements

4.1 Performance Requirements

4.1.1 Time to respond to an uploaded file

A user is required to upload a voxel object to the library and then should be allowed to mainpulate a rules file if they choose to do so or they should be able to use a predefined rules file such that the rules are applied to the object. The library must not delay once a file is uploaded, this means the time it takes to respond to a uploaded file should be proportional to the files size.

4.1.2 Respond to user interaction

The system should respond to user interactions in real time.

4.1.3 Reliability

The system should never cease working completely unless the error is caused by external systems outside our control (operating system, web APIs, etc). Ideally an entire system uptime (per month) of 99.0% must be reached, meaning that the system should have validation and error checking to prevent unwanted results.

4.1.4 Maintainability

The library's code must be well documented, both by means of in-code comments and external documentation, to aid in maintaining the system. A user manual should also be provided to make it easier to understand and ultimately maintain the library.

4.1.5 Portability

This library should be accessible across all major internet browsers that support WebGL. These include Chrome, Firefox etc. The library should also be able to be imported to any website.

4.1.6 Scalibilty

The system should be able to handle the majority of web browsers and file sizes, by file size we mean that the system should not necessarily struggle or outright reject a file because it was too large. Realistically we cannot handle all file sizes, but the system should aim to handle a large as possible file size.

Additionally the library should be able to work with any other webpage as a simple imported library, therefore users should be able to alter positions and perhaps even sizes of the dom elements used to upload, download and display the objects.

4.1.7 Usability

The Voxel library should be easy to operate and understand. The core functions of the system shouldn't take the user more than a minute to access and understand. A user manual should be provided to aid in the use of the library and to explain some advanced functions or settings.

4.2 Security Requirements

This library should be an open source project and so the code will be freely available to anyone visiting the Github repository. The actual web interface hosted on Github will act as a demo of the library and should be protected, the only editing that a user should be able to do is uploading of objects and editing their own rules file.

4.3 Quality Requirements

Stars for reviewing