# Software requirements specification for project

# Blood, sweat, and voxels

## Authors

πVoxelTeam:

Arseniy Morozov

Denis Enes

Maxim Sharapov

Vladimir Parnachev

## Introduction

We want to improve the user's experience of interacting with the game world and make this world feel more realistic. That’s why we decided to use the voxel world representation. Our final library will be used as a graphic base for other more complicated applications.

## Glossary

Voxel - (**vo**lumetric + pi**xel**) - a volumetric image element containing the value of a raster element in three-dimensional space. Voxels are analogous to 2D pixels in 3D space.

Graphics engine - a program that performs a core or essential function for developing other applications involving graphics.

Vizualizator - a part of a program that will draw stuff on screen.

Scene - computer model for representing virtual space.

Voxelization - translating polygonal model to voxel mesh

Object - voxel mesh

## Actors

Software developer - wants to develop his own application. He chooses our engine and works with voxel-based graphics.

## Functional requirements

### Strategic Use-cases

[Optional. White-level use-cases. This section is useful when there are too many blue-level use-cases and they should be grouped somehow.]

### Use-cases for developer

#### *Use-case <UC-1-1>*

**Actors:** software developer

**Goals:** load and render scene, constructed with voxels

**Mains success scenario:**

1. Developer initiates loading protocol for his own scene
2. Our library loads it
3. Library starts rendering process
4. Library shows window with rendered scene in-Real-Time
5. Library keep rendering scene until the user stops execution

#### *Use-case <UC-1-2>*

**Actors:** software developer

**Goals:** manually add / delete voxels in the certain place

**Mains success scenario:**

1. Library runs rendering process in the window
2. Developer presses the special button on the keyboard/mouse in order to add/delete voxel
3. Library reads inputs from devices connected to PC
4. Library reacts on input and changes the world according to the input command.

#### *Use-case <UC-1-3>*

**Actors:** software developer

**Goals:** to save a custom scene

**Mains success scenario:**

1. Developer presses the combination of buttons on the keyboard
2. Library reads inputs from devices connected to PC
3. Library saves current scene in the special file

#### *Use-case <UC-1-4>*

**Actors:** software developer

**Goals:** to look around the scene

**Mains success scenario:**

1. Developer wants to move the camera
2. Library reads inputs from devices connected to PC
3. Library rerenders the scene in-Real-Time according to changes made to camera position

#### *Use-case <UC-1-5>*

**Actors:** software developer

**Goals:** create a voxel-based model from a regular polygonal model

**Mains success scenario:**

1. Developer loads a polygonal model to the side program of ours
2. Side program translates a polygonal mesh to voxel mesh
3. Developer can save result as file

#### *Use-case <UC-1-6>*

**Actors:** software developer

**Goals:** set positions of light sources

**Mains success scenario:**

1) Developer calls a special function that set position of light source

2) Then he compiles project and runs it

3) Our library renders scene according to parameters of light source that developer has set before.

### 

### System-wide functional requirements

*[Optional. System-wide functional requirements that weave with multiple use-cases. Examples: authorization, audit]*

## Non-functional requirements

*[All the subsections are optional.]*

1. Stable quite high fps (>= 30)
2. High screen resolution ( >= 1048576 pixels)
3. Huge amount of voxels that our render can proceed in real-time
4. Simple and functional API
5. Support various properties of voxels (colors, textures, reflectivity etc)

### Environment

*[Environment requirements are limitations for hardware and software usage including supported hardware platforms, networking infrastructure and protocols, programming languages, libraries and external services]*

1. Operating system: Windows
2. Nvidia GPU

### Performance

*[The performance characteristics of the system should be outlined in this section. Examples are response time, throughput, capacity and startup or shutdown times.]*

### Reliability

*[Reliability includes the product and/or system's ability to keep running under stress and adverse conditions. Specify requirements for reliability acceptance levels, and how they will be measured and evaluated. Suggested topics are availability, frequency of severity of failures and recoverability.]*

### Extensibility

*[This section indicates requirements that will enhance the extensibility including extension points, compatibility, scalability, configurability]*