JSTanks - Design Document

Jiahao Li	Pavithran Pathmarajah	Viren Patel
LI577	PATHMAP	PATELVH3
001416646	001410729	001419057

December 8, 2016

Contents

1 Revision History

Table 1: Revision History

Date	Developer	Change	Revision
November 6	Jiahao Li	Initial Draft	0
November 6	Pavithran Pathmarajah	Initial Draft	0
November 6	Viren Patel	Initial Draft	0
December 7	Viren Patel	Final Draft	1

2 Introduction

Module Decomposition is a good practice when it comes to developing applications involving many different data structures. Following this can benefit the program in many ways including the implementation process, maintenance, testing, and understanding of the program to new entities.

This document is a Module Guide for JSTanks. JSTanks has followed the process the Module Decomposition greatly and can be seen in this doument.This document goes in to detail to show how the things mentioned in the Software Requirements Specifications document are accomplished in the program. It shows how the functions of the program are executed as the modules interact with eachother and tracing each function to its corresponding module(s).

This document has been divided into the following sections: Anticipated and Unlikely Changes of the software requirements, Module Hierarchy, Connection between Requirements and Design, Module Decomposition, Traceability Matrix, and Use Hierarchy between Modules.

3 Anticipated and Unlikely Changes

This sections contains possible changes to the project which may occur. The changes are placed into two categories anticipated and unlikely, depending on how likely a change is to occur.

3.1 Anticipated Changes

Anticipated changes are sources within modules which may be modified throughsout the products life time. The changes will be made to specific modules in such a way that it will not affect how modules interact, or respond, but how they work.

AC1

Each tile pre-render's it's own image when it is created. Will be changed such that shared images are pre-rendered on startup

AC2

Currently a single thread runs procedurally. Updated game will have a sole update and render thread in the background. Preventing browser lock up.

AC3

Level and map specifications may be moved to an external text or Html file to be parsed by the game dynamically. Allowing for the addition of new maps without having to hardcode scripts.

AC4

Artificial Intelligence of bots, to be replaced with an algorithm to try and attack the player tank and the home-base depending on what is closer.

AC5

The strength of projectiles to go down depending on number of grid spots away from the tank they have travelled.

AC6

Only re-rendering changed entities instead of the current system where all entities are re-rendered each frame.

3.2 Unlikely Changes

Unlikely changes are the changes that will most likely not occur, for the affects of these changes would require many modifications to many modules for it would have an effect on how modules interact and respond.

UC1

The data that tile-entities store and provide will remain at a minimum of health information and graphics renders.

UC2

The game will continue to be purely, HTML, CSS, an JavaScript such that it modules will not have to be ported to another language or ecosystem

UC3

All interfacing between entities and the user will continue to propagate through the gameBoard class.

UC4

Game graphics will not be changed from a three level render stack to, a two level stack. Since it would require changes across all modules.

4 Module Hierarchy

This section provides a simple overview of the module design. The modules are hierarchically labelled M1 through M13 below and are decomposed in the table below that.
Hierarchy. These are the modules which will be implemented.

M1: Game Module
M2: GameBoard Module
M3: Projectile Queue Module
M4: Create Game Module
M5: Player Module
M6: Bot Module
M7: Tank Module
M8: Empty Tile Module
M9: Projectile Module
M10: Home-Base Module
M11: Wall-Module
M12: Steel-Wall Module
M13: Overlay Module
M14: JSTanks Web Module
M15: JSTanks Style Module

Level 1	Level 2
Game Module	- Set graphics area
	- Create graphics context
	- start main game loop
	- interface between key presses, Overlay menu and GameBoard
	- GameBoard Module

	- Overlay menu
GameBoard Module	- Interface between user and game
	- update board graphics
	- update projectiles
	- move entities on request
	- Create Game Module
	- Projectile QueueModule
	- Player Module
	- Bot Module
	- Wall Module
	- Steel-Wall Module
	- Home-Base Module
	- Empty Tile Module
Projectile Queue Module	- Array of Projectiles
	- draw projectiles on board
	- Projectile Module
Create Game Module	- Place tiles on board
	- Player Module
	- Bot Module
	- Wall Module
	- Steel-Wall Module
	- Home-Base Module
Player Module	- Keyboard interfacing with tank
	- Tank Module

Bot Module	- Artificial Intelligence interfacing with tank - Tank Module
Tank Module	- tank movement - fire projectiles - tracks tank health
Empty Tile Module	- No strength factor - Can be moved to - replaces destroyed entities
Projectile Module	- Damage tile entities - identify tile hit
Home-Base Module	- Player's home base will trigger end game on destroy - tracks base health
Wall Module	- A weak wall, that can be destroyed - tracks wall health
Steel-Wall Module	- A stronger wall, that can take more damage - tracks steel-wall health
Overlay Module	- Pauses game loop - overlay's pause menu
JSTanks Web Module	-Displays the whole game on the website
JSTanks Style Module	-Styles the game's webpage

5 Connection between Requirements and Design

The requirements included in the Software Requirements Specifications document are implemented into the modules listed in the "Module Hierarchy" 4 section of this document. The corresponding modules of all function and non-functional requirements can be found in Table 4 .

6 Module Decomposition

Modules are decomposed according to the principles of "information hiding" proposed by Parnas ed el (1984). We have made the design decision to not implement traditional information hiding, such that the project may be used by others whom are starting out in game design or programming, whom will be able to easily decipher our project and build their own web based game. By hiding how our key modules function internally, these new-comers will not be able to fully understand how to develop a basic HTML, CSS, JavaScript game; for this reason JSTanks has opted out from information hiding. All module decomposition can be found in the Module Interface Specification.

7 Traceability Matrix

Requirement	Module(s)
	Functional Requirements
FR1	JSTanks Web
FR2	JSTanks Web
FR3	JSTanks Web, JSTanks Style, Game, Game Board, Home-Base, Wall, Steel-Wall, Create Game, Tank, Bot, Overlay
FR4	Overlay.
FR5	Overlay
FR6	Overlay
FR7	Overlay
FR8	Overlay
FR9	Overlay
FR10	Overlay
FR11	Overlay
FR12	Overlay
FR13	Bot, Projectile Queue, Tank
FR14	Bot
FR15	Bot
FR16	Overlay
FR17	Player, Tank
FR18	Projectile Queue
FR19	Projectile Queue, Tank, Bot
FR20	Game Board, Home-Base, Wall, Steel-wall, Projectile Queue
FR21	Game Board, Home-Base, Wall, Steel-wall, Projectile Queue
FR22	Game Board, Home-Base, Wall, Steel-wall, Projectile Queue

FR23	Projectile Queue, Tank, Player
FR24	Overlay
FR25	Overlay
Non-functional Requirements	
NF1.1	JSTanks Web, JSTanks Style, Game, Game Board, Home-Base, Wall, Steel-Wall, Create Game, Tank, Bot
NF1.2	JSTanks Web, JSTanks Style, Game, Game Board, Tank, Bot, Overlay
NF2.1	Tank, Player, Projectile
NF2.2	–
NF2.3	JSTanks Web
NF2.4	Overlay
NF2.5	–
NF3	–
NF4	JSTanks Web, JSTanks Style
NF5	Game, Game Board, Tank, Player, Bot, Overlay, Home-Base, Wall, Steel-Wall, Create Game, Projectile Queue
NF6	–
NF7	Overlay
NF8	–
Anticipated Changes	
AC1	Game Board, Game
AC2	Game, JSTanks Web
AC3	Create Game, Game Board
AC4	Bot
AC5	Projectile
AC6	Game Board, Game

Table 4: Trace Between Requirements and Modules arc@@@arcblack!50

8 Use Relations

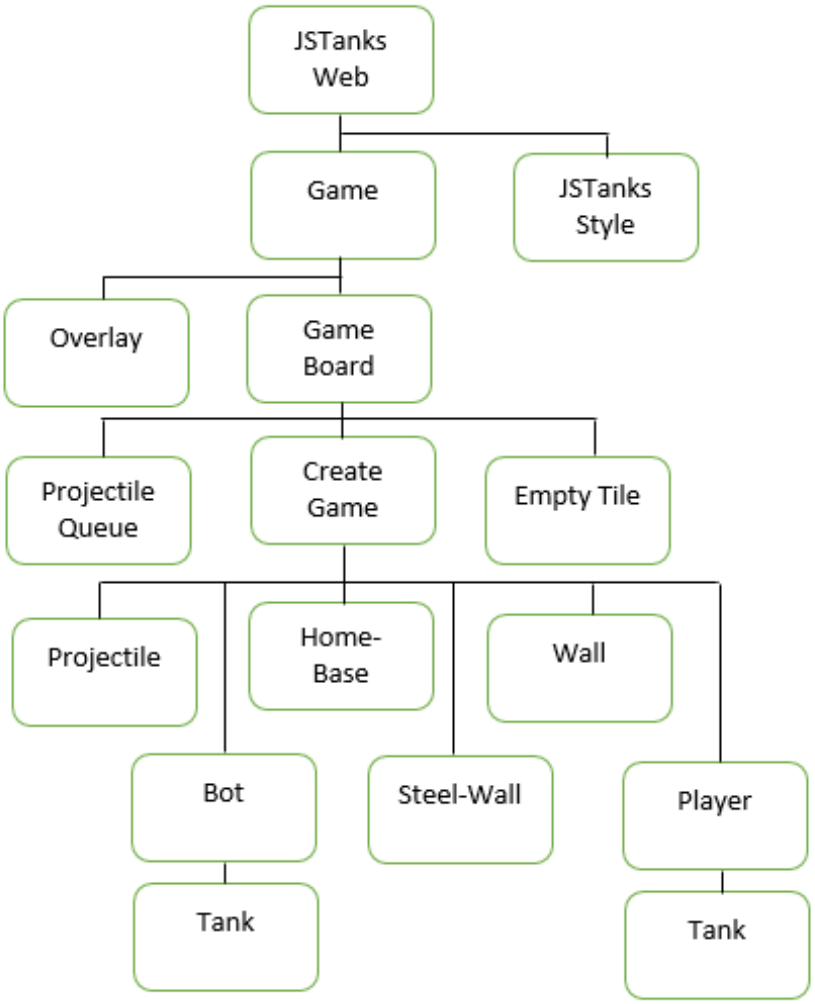

Figure 1: Uses Hierarchy

Module		Use Relation
JSTanks Web	uses	JSTanks Style, Game
JSTanks Style	uses	–
Game	uses	Game Board, Overlay
Game Board	uses	Projectile Queue, Create Game, Player, Bot, Empty Tile, Home-Base, Wall, Steel-Wall
Projectile Queue	uses	Projectile
Create Game	uses	Player, Bot, Empty-Tile, Home-Base, Wall, Steel-Wall
Player	uses	Tank
Bot	uses	Tank
Tank	uses	–
Empty Tile	uses	–
Projectile	uses	–
Home-Base	uses	–
Wall	uses	–
Steel-Wall	uses	–
Overlay	uses	–

Table 5: Uses Relationship between Modules

List of Tables

List of Figures