

Ultimate 2048

Software Requirements Specification

Team 4
Slot-11 Group-18

Prepared for
CS 101 course project
Instructor : Prof. D.B. Phatak,
Prof. Supratik Chakraborty
Autumn 2014

INDEX

Contents	Page #
1.0. Introduction.....	3
1.1. Purpose of Document.....	3
1.2. The Project.....	3
1.3. Scope of Project.....	3
1.4. References for SRS.....	4
1.5. References for Project.....	4
2.0. Requirements Specification.....	4
2.1. Functional Requirements / Modules.....	4
2.2. Non-Functional Requirements	6
2.3. User Characteristics.....	6
3.0. Developer-Defined Functions in App.....	7
3.1. Various Modules.....	7
3.2. User Interface.....	7
4.0. Solution (Product) Overview	8
4.1. End Product Environment	8
4.2. Assumptions and Dependencies.....	8
5.0. Future Prospects.....	9

1.0. Introduction

1.1. Purpose of Document

The purpose of this document is to present a detailed description of the Ultimate 2048 game that the team plans to make. The following sections of the document explain the purpose and features of the game, the interface of the game, and the constraints under which it must operate.

1.2. The Project

Ultimate 2048 will be a feature packed version of the original 2048 game. Ultimate 2048 is played on a NxN grid(default being 4x4), with numbered tiles that slide smoothly when a player moves them using the four arrow keys. Every turn, a new tile will randomly appear in an empty spot on the board with a value of either 2 or 4. Tiles slide as far as possible in the chosen direction until they are stopped by either another tile or the edge of the grid. If two tiles of the same number collide while moving, they will merge into a tile with the total value of the two tiles that collided. The resulting tile cannot merge with another tile again in the same move.

1.3. Scope of Project

This game is meant for pure entertainment for people who enjoy playing puzzle-games and will attract more fans like people on the go. This game can be even made an online in the future. It can be also replicated easily onto any mobile platform. For now it will be employed as a PC mini-game.

1.4. References for SRS

IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications*. IEEE Computer Society, 1998. (Present on www.wikipedia.com as an article)

1.5. References for Project

Library Listings for C++. www.cplusplus.com

SFML Tutorials. sfml-dev.org/tutorials/2.1/

Gamesfromscratch. gamefromscratch.com

Object Oriented Programming with C++ by E Balagurusamy.

Discussions on sites stackexchange.com & stackexchange.com

2.0.Requirement Specifications

2.1.Functional Requirements / Modules

➤ *GAMESELECT SCREEN:*

It has three options:

- ❖ Autosolver :- This takes you to an auto-solver, which runs on pressing the “comma” key. This works only for a 4x4 grid and classic mode.
- ❖ Manual 2048 :- This takes you to homescreen, where you can choose to customize various options and play 2048 game.
- ❖ Virahanka 2048 :- This takes you a 4x4 grid, on which consecutive Virahanka numbers can be added using the arrow keys. This also is only for classic mode.

➤ *Homescreen :*

It consists of three options:

- ❖ Play :- Selecting this will take to the main game screen where you can play the game.
- ❖ Game Options :- Selecting it will lead you to options screen, where you can choose the size of grid, colour theme and the game mode.
- ❖ Read instructions : Selecting it will take you to instructions screen, from where you can review the rules & objective of the game.

➤ *Game Options Screen :*

Here you can choose:

- ❖ Size of Grid :- The Grid size can be set to 4x4, 5x5, 6x6 or 7x7 (default being 4x4)
- ❖ Colour Theme :- Colour Theme can be set either Red, Green or Blue.
- ❖ Game mode :- Here you can choose from the following game modes:
 - X-tile : This mode has one X-tile which cannot be added to any other tile.
 - Survival : In this mode 2 or 4 numbered tiles keep popping out randomly after a fixed interval of time. If you fail to add them and the screen gets filled completely then the game is over.

- **Classic :** This mode is the normal 2048 game without any modifications.

➤ **Instructions Screen :**

It displays the game rules for your reference.

➤ **Game Screen :**

It consists of:

- ❖ **Grid :-** It is the space where all the tiles are situated and are moved.
- ❖ **Tiles :-** Tiles are coloured square with numbers on them.
- ❖ **Scoreboard :** It shows the current score and high score.

2.2 *Non-Functional Requirements*

- **General Guidelines :** Priority should be given to performance, adaptability, maintainability, and usability.
- **Operating Constraints :** This program requires the machine to have GNU-GCC compiler and the computer to have C++ Redistributable package in case of Windows and C++ Runtime Environment in case of Linux.
- **Documentation :** The User Manual will be made for the application in addition to the "Read Instructions" included inside the 'Main Menu' of the game.
- **Portability :** The program will run on Windows XP/Vista/7/8/8.1, and Ubuntu 10.04 or higher.

- **Reliability :** As the game is for pure recreation and involves no user data, reliability is of low importance.
- **Deployment :** The game will be uploaded online in the form of a C++ executable file.

2.3 User Characteristics

The user is expected to be literate in the computer basics and be able to use a keyboard or a mouse.

3.0. Developer-defined Functions in App

3.1. Various Modules

- **Generation of the tiles:** Initialising and updating the game after each move involves generation of '2' or '4' tile randomly. The tile must be generated on an empty space on the grid.
- **Movement of the tiles:** This function involves input from the user. Direction of movement is provided by user with keyboard arrow keys corresponding to up, down, left, right. All tiles are moved in the specific direction till they encounter another tile or a wall(end of grid).
- **Updating the tiles on Grid:** If any tile encounters another tile of same value during movement, then the 'front' tile will contain twice the value of the tiles and the 'back' tile will be erased. Front is in the

direction of the movement. After this the score is incremented by the sum of values of all new updated tiles.

3.2. User Interface

A GUI Window as well as a console output window are there when the game is executed. User shall provide input using mouse clicks and key presses while the GUI windows is active. The controls are mentioned in the user manual. The game screen will have a grid of NxN and coloured tiles having text on them specifying their value. These tiles have to be moved and updated according to user input with constraints from the game rules.

4.0. Solution (Product) Overview

4.1 End Product Environment

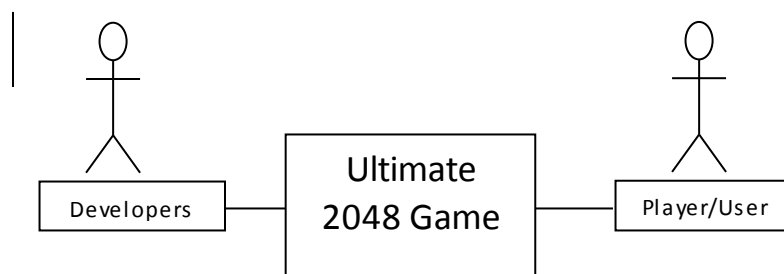


Figure 1 – End Product Environment

The end product has two active actors. The main active part in this system will of course be the user of this end product. But even the developers will remain active in the sense that they will help out in sorting out errors that may crop up and updates that may be required even after the project has been completed.

4.2 Assumptions and Dependencies

- The application is not internationalized; all menus and messages are in English.
- The application will run on Windows XP/Vista/7/8 and Linux platforms with the C++ Runtime Environment installed.
- The application does not require an internet connection, other than to download the application from the product website.

5.0. Future Prospects

This game can be even made an online in the future. The game can also be ported as a mobile game and made available on tablets after suitable changes are made for increased adaptability.