Requirements and Analysis Document for

PaintIT

Henrik Lagergren, Markus Pettersson, Aron Sjöberg,
Ellen Widerstrand, Robert Zetterlund

2/10/18

v.1.0

Contents

1 Introduction

Here it should be some kind of major introduction

1.1 Purpose of application

The project aims to create a desktop version of the popular mobile painting-game ”paintSomething” developed by OMGPop. By using the desktop the user is able to use their mouse and their keyboard which allows for a smoother painting experience. While catering for the advanced user, PaintIt is also easy to play for the most inexperienced computer user since it uses UI-design patterns (To read more read appendix 1).

Further, the applications aims to cater to children of developing countries and by being a member of the edu-gaming genre(?) the child will learn basic english words while painting and expressing themselves artistically.

1.2 General characteristics of application

The game is collaborative, meaning the users work together gaining points which is stored in a ”streak” that is viewable in the highscore. When starting the game, one of the users (from now on referred to as the guesser) is advised to look away whilst the other player is presented with a word, (now on be referred to as the painter). The painter's task is to paint the presented word with enough accuracy that the guesser will be able to guess the word using a set amount of Tiles.

The application is an offline desktop game, which follows the target groups prerequisites, users will gather to one computer and take turns painting and guessing. The game is originally designed to consist of two users, however, it is possible to paint and guess collaboratively, meaning the game can be played by virtually any amount of players.

1.3 Scope of application

The scope of the application lies within the hands of the player, meaning the difficulty and entire experience is determined by the painting. What this effectively results in is a game where players rely on each other for a pleasant experience. The Game revolves around a basic loop, (more throughout explanation in the section below):

1. Receiving word
2. Painting word
3. Guessing word

1.4 Objectives and success criteria of the project

1. The game should save the team's streak and add it to the high score.
2. Before starting the game, the players should be able to view the high score.
3. It should be possible for the players to choose their names.
4. Before the players begin to play, it should be possible to get information about how the game works and what they are supposed to do.
5. One player, the painter, should get a word to paint, as well as a painting brush and canvas that he or she can use to paint the word. The other player, the guesser, should be able to see the painter's painting when it is finished and get letters that he or she can use to guess what the painter has painted.

1.5 Definitions, acronyms, and abbreviations

1.5.1 General words used throughout document and development

- MVC - Model View Controller.

1.5.2 Words explaining the game

- **Painter** The player that is being presented a word and that is supposed to paint on the Canvas.
- **Word** The word that is supposed to be painted and guessed.
- **Canvas** The canvas contains the painting of the painter, the canvas is also shown to the guesser.
- **Guesser** The player that is being presented with a painting and eight tiles.
- **Tiles** A tile consists of 1 (one) letter, in total there are 8 (eight) tiles that has n amount of letters being of the words, (where n is and int of the amount of letters in a word)
- **Round** - A round consists of a word being presented, painted and guessed, either incorrectly or correctly.
- **Streak** - The amount of successful rounds, meaning the painting was guessed correctly.

1.5.3 Implementation definitions

- **TopController** - Is a controller that shows and prepares views, unlike every other controller, it does not hold one (1) view, it holds all the views in the application within a list.
- **Canvas** - Not an actual class. The canvas is the umbrella term for the area which you can paint on. The Canvas consists of the CanvasModel, CanvasController and the CanvasView.
  - **CanvasModel** - The actual representation of the canvas. The CanvasModel consists of a 2D matrix of type Color. The model is observed by the CanvasView.
  - **CanvasController** - The CanvasController recieves coordinates from the user via the CanvasView, and changes the model accordingly with regards to the equipped Tool.
  - **CanvasView** - The actual view of the canvas, it extends the JavaFX Class Canvas and uses a pixelWriter to change pixels.
- **Tool** - [REDACTED] Currently an interface which is implemented by Brush, SprayCan, Eraser. Tools is observed by CanvasController and calls the update(int x, int y, Color color).
  - **Brush** -
  - **SprayCan** -
  - **Eraser** -

2 Requirements

Below follows retirements which is to the model what markus is to the group; extremely important.

2.1 Functional Requirements

The players should be able to:

1. Navigate around the main menu.
    a. Be able to read the rules.
    b. Look at the highscore
    c. Go to the game setup screen.
        i. Choose the player names
        ii. Choose playstyle
        iii. Choose timelimit.
        iv. Choose difficulty.
2. Play the game.
    a. The Painter should be able to:
        a. Paint on the canvas using the
            i. Brush
            ii. SprayCan
            iii. Eraser
        b. Choosing color to paint with.
        c. Choosing the radius to paint with.
        d. Clear the entire canvas.
        e. Undo the latest stroke of paint.
    b. The Guesser should be able to:
        a. See the painted canvas.
        b. See a selection of eight tiles.
        c. See an empty word sequence.
        d. Guess the word by:
            i. Moving the given tiles to the word sequence.
            ii. Removing tiles from word sequence.
            iii. Asking for hints.

e. Give up.

c. At the end of the round select to start a new round or end game session.

3. Exit the application and store game-streak.

The application should be able to:

1. ...?

2.2 Non-functional Requirements

Below follows something about something.

2.2.1 Usability

Playing the game requires basic computer knowledge, primarily usage of the mouse to navigate the application and to paint. Further, some experience with drawing programs, msPaint for example, would be beneficial since similar techniques are used throughout the painting process. The program adjusts for different levels of knowledge within the English language, however some English knowledge is required.

2.2.2 Reliability

2.2.3 Performance

The application should not have any delay on a regular computer, except when using the undo and clear functionality while painting. These two buttons redraws a large portion of the canvas, while storing the change, meaning a lot of computer processing power.

2.2.4 Supportability

https://github.com/nahojjjen/OOP-Project-TowerDefence/blob/master/document/RAD.pdf

2.2.5 Implementation

2.2.6 Packaging and installation

2.2.7 Legal

2.3 User Stories

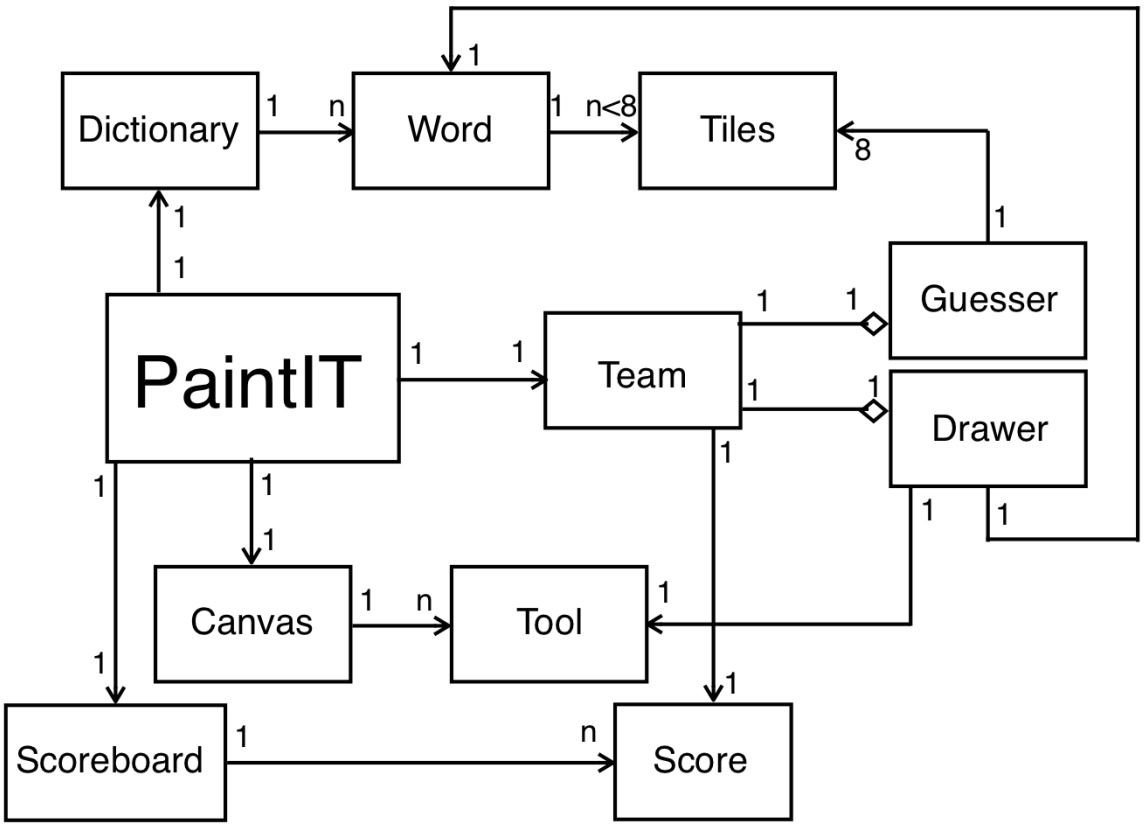
Use the template from the course website and list all user stories here. It is fine to have them in an spreadsheet (or other application) at first, but they must end up here as well.

These user stories should describe what the user will be able to do. Write a the user stories in language of the customer, and give the a unique ID. List the user stories in priority order.

2.4 User interface

Sketches, paintings and explanations of the application user interface (possible navigation).

3 Domain model



3.1 Class responsibilities

Explanation of responsibilities of classes in diagram.

4 References