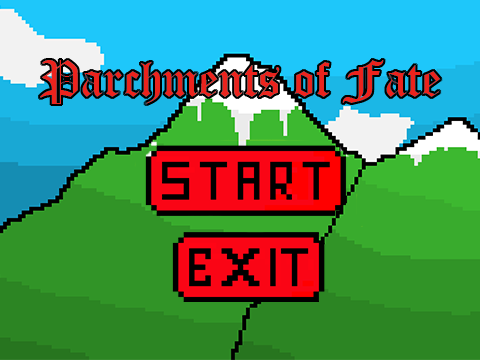
Parchments of Fate

*The best top down casual arcade of 2019!.*



Team 1: Aaron Brent B. Abundo, William Aristotle M. AllosadaRequirements Analysis Document (RAD)

# Introduction

## Purpose of the system

There are countless top down arcade games available to be played. One of the best ones is known as Journey of the Prairie King (Is a minigame from Stardew Valley). For the purpose of this assignment, I will always refer to the game as Stardew minigame).

* [Stardew Valley](https://www.stardewvalley.net/)

You will learn more about this game during the course of this HW1 task. Indeed, your primary goal is to determine (a) the core functionality that must be provided by your system and (b) the numerous possible extensions. This document available today is minimal since the primary task of each team is to fill in the requisite details as required to complete the assignment.

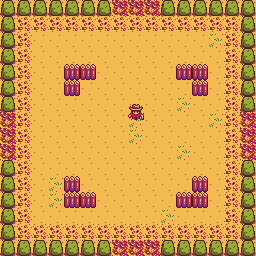


Figure 1: Sample Stardew Minigame layout

## Scope of the system

Parchments of Fate is a game where the player fights off hordes of enemies in an increasing difficulty and numbers. The player will move around a specified location where enemies will spawn around the area. This is a single-player video game where the character you play is the protagonist. The specific rules of the game shall be made at a later date. To keep the game as simple as possible there will not be a network version of the game. The single-player shall use a mouse and keyboard as his/her primary controlling tool.

## Core System Functionalities

The application must be able to:

* Allow one user to play the game. Note that there is no ability to save a game in progress; the game will end once the player dies.
* Provide a GUI that allows players to easily control his/her character.
* Allow the player to restart the game whenever he or she wishes.
* Show the final score once the player dies.

## Objectives and Success Criteria of the Project

The success of the application depends upon meeting the following core set of objectives:

* The design of a command mechanism to encapsulate all changes to the model as a series of commands to be executed on the model.
* The design of a restart-able system where the player can choose to do so whenever he/she wants.
* The design of a GUI to enable the proper playing of a game.
* The use of automatic test cases to validate that 80% of the written code executes according to specification.

## Definitions, Acronyms, and Abbreviations

Important terms and concepts are listed here. More will be added by the project teams.

*Model* A schematic description of a system that accounts for its known or inferred properties [http://www.answers.com]

*View* A visual representation of a model which might also enable a user to manipulate properties of the model

*Game* A game is an instance of Parchments of Fate.

*Sandbox* The location where the player plays.

*Variation* A rule designed to be in effect during game play. There are a [number of potential variations](http://www.angelfire.com/wizard/sandshrew777/faq-oxfest.html) that you must consider.

*Player* Any user that interacts with the system.

*System* Any interactions performed by the application are considered to be performed by the system.

*Score* The number of points earned by each player against other players in the game.

*Submit* The action of the player indicating that they have placed tiles in a configuration for which the player wishes to receive points.

*Restart* The action of the player indicating that they want to restart the game.

*GUI* Graphical User Interface

# Current System

No system is in place but you can visit and play Parchments of Fate given that you have an installer file (Of which anyone can ask for by emailing abundoaaronbrent@gmail.com). I encourage players to do this so they can get a sense of the game behavior and logic.

# Proposed System

## Overview

This section provides a functional overview of the system. This will again be properly be divided into two parts

## Functional Requirements

## The game must be playable for 1 person.

## Players must be able to start a new game when the program is loaded.

## Players must be able to start a new game during the progress of a game with confirmation that they want to cancel the old game.

## Player must be able to eliminate enemies when they spawn.

## Player must be able to restart the game.

## Player must be able to see the score after dying.

## Player must be able to exit out of the game.

## Nonfunctional Requirements

### Usability

* Game must be playable using a mouse and keyboard.
* Player must be easily discernable by the user.
* Score must be shown on screen for the user.
* Easy to locate buttons must be around for ease of use.
* SYSTEM will provide status messages to the player to inform the player of the state of the game.
* SYSTEM must play a sound once the HP of the player is low.
* SYSTEM must play music while the game is running.
* SYSTEM must play a sound once the player has died.
* SYSTEM must play a sound once the game starts.
* SYSTEM must play a sound when a boss spawns in the game.

### Reliability

### Components of the project code will be tested alongside the implementation phase to ensure that they are functional.

### Final, integrated project Code will be tested manually. to ensure that greater than or equal to 80% of the integrated code is covered at run-time, and is functioning properly. The remaining 20% will be inspected through outsourced manual testing to ensure the highest chance of being quality code.

### Performance

### Attacking the enemies must be smooth without graphical lagging.

### Interaction with the GUI must be seamless and easy.

### Death of player and appearances a score must be easy to read.

### Supportability

### The application must not be platform dependent, i.e., it should be able to run on any platform run Unity games.

### Implementation

* Project will be implemented in Unity.
* All project graphical user interfaces will be created using a Unity GUI editor.

## System Models

### Use case model

|  |  |
| --- | --- |
| **Name:** | StartNewGame |
| **Actor:** | PLAYER |
| **Entry**  **Conditions:** | Application is running.  A Game is not currently in progress. |
| **Flow of**  **Events:** | 1. PLAYER initiates new game function  2. SYSTEM presents PLAYER with NewGameForm  3. PLAYER submits NewGameForm |
| **Exit**  **Conditions:** | Game is now in a new state. |

|  |  |
| --- | --- |
| **Name:** | StartNewWhileGameInProgress |
| **Actor:** | PLAYER |
| **Entry**  **Conditions:** | A Game is currently in progress |
| **Flow of**  **Events:** | 1. PLAYER initiates new game function  2. SYSTEM confirms that the PLAYER wishes to continue.  3. SYSTEM presents PLAYER with NewGameForm  4. PLAYER submits NewGameForm |
| **Exit**  **Conditions:** | Game is now in a new state. |

|  |  |
| --- | --- |
| **Name:** | AttackEnemy |
| **Actor:** | PLAYER |
| **Entry**  **Conditions:** | It is initiating while PLAYER is alive.  Player is in game  Game is not finished |
| **Flow of**  **Events:** | 1. PLAYER clicks attack button.   2. SYSTEM shows attack animation. |
| **Exit**  **Conditions:** | Attack animation has now been shown and will deal damage to anything that hits. |

|  |  |
| --- | --- |
| **Name:** | PlayerMovement |
| **Actor:** | PLAYER |
| **Entry**  **Conditions:** | It is initiating while PLAYER is alive.  Player is in game  Game is not finished |
| **Flow of**  **Events:** | 1. PLAYER clicks anything on WASD and will move according to which button is pressed, W is forward, A is left, S is backward, and D is right.   2. SYSTEM shows player moving according to button. |
| **Exit**  **Conditions:** | The PLAYER model has now moved according to user. |

|  |  |
| --- | --- |
| **Name:** | ExitGame |
| **Actor:** | PLAYER |
| **Entry**  **Conditions:** | While the game is running. |
| **Flow of**  **Events:** | 1. PLAYER clicks exit button.   2. SYSTEM closes the game. |
| **Exit**  **Conditions:** | The game is now closed and no game process is running. |

|  |  |
| --- | --- |
| **Name:** | RestartGame |
| **Actor:** | PLAYER |
| **Entry**  **Conditions:** | It is initiating while PLAYER is in game.  Player is in game  Game is not finished |
| **Flow of**  **Events:** | 1. PLAYER clicks the restart button.   2. SYSTEM restarts game instance and resets player HP and score. |
| **Exit**  **Conditions:** | The game has restarted and reset the PLAYER’s HP and score. |

### Object model

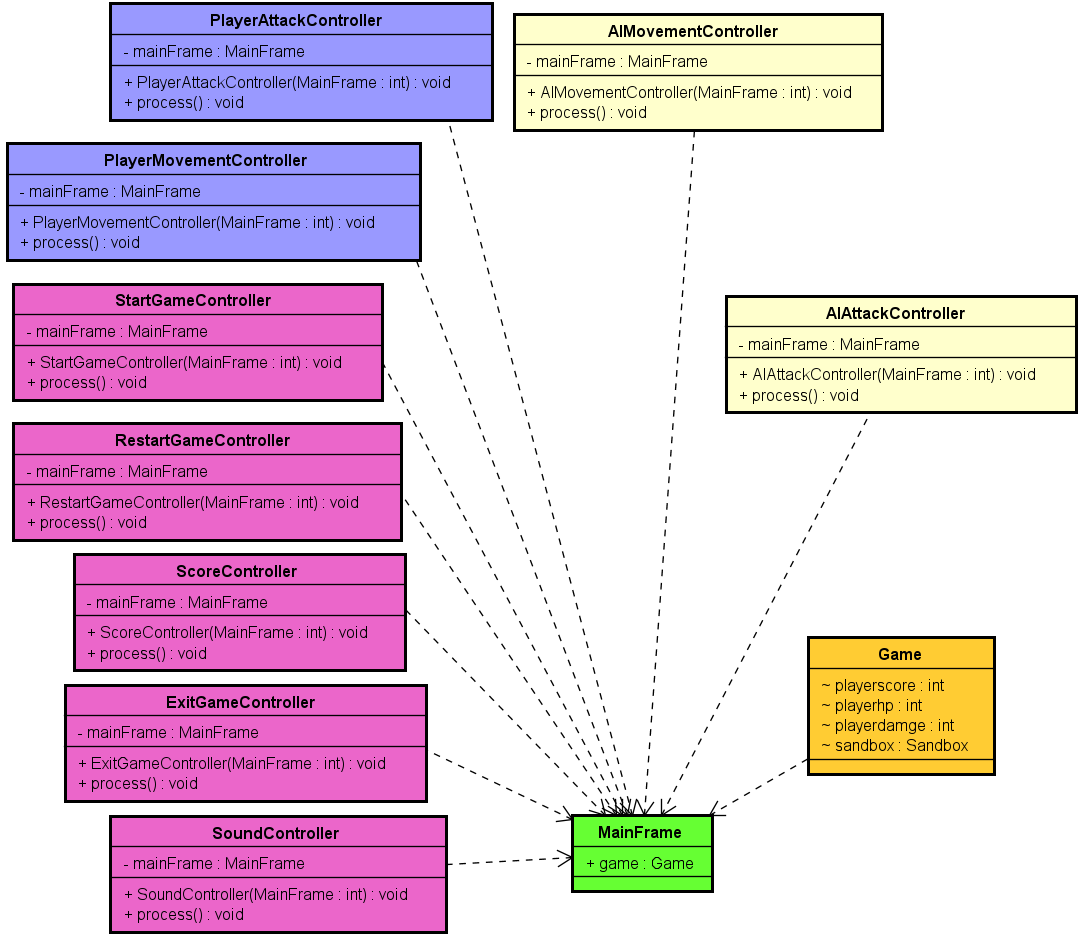


Figure 2.1 - Object Model – Boundaries/Controllers

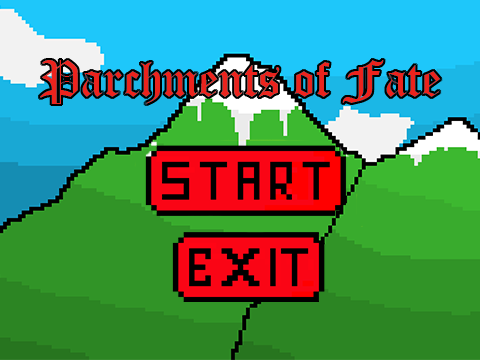
### User Interface – Navigational Paths and Screen Mock-ups



Figure 3 – Main Game

Game Controls:

* Movement
  + Forward (W)
  + Backward (S)
  + Left (A)
  + Right (D)
* Attack
  + Melee Attack (J)
  + Ranged Attack (K)
* System Buttons
  + Restart Game (F1)
  + Exit Game (F2)



**Figure 4 – Main Menu**

Main menu options:

* Start button leads in to the main game page.
* Exit button exits the game.

# Change History

As this document changes, different versions of the game will be saved in a flash drive.

|  |  |
| --- | --- |
| **Version** | **Description** |
| 20191206.001 | Conceptualization |
|  |  |

# References

* Start Button [http://pixelartmaker.com/art/397185d2d931676.png]
* HP icon [https://www.shutterstock.com/video/clip-22752997-pixel-art-retro-game-style-red-hearts]
* Slime [http://pixelartmaker.com/art/34e3be4ae06fbef.png]
* Reset Button [http://pixelartmaker.com/art/191cdb8274cfea9]
* Main Heroine [http://pixelartmaker.com/art/f392a4176916b51.png]
* Stardew Minigame Reference [https://stardewvalleywiki.com/Journey\_of\_the\_Prairie\_King]
* Tentacle Monster [https://www.istockphoto.com/vector/vector-pixel-art-cartoon-gm813145798-131647733]
* Tree [https://www.redbubble.com/people/p04i/works/24275043-pixel-art-tree?p=spiral-notebook]
* Grass Fields [https://opengameart.org/forumtopic/grass-help]
* Start and Exit Button [http://pixelartmaker.com/art/c1b594dc5a6c8ce]
* Top down 2d Unity Basics [https://learn.unity.com/tutorial/live-sessions-on-2d-in-unity-4-3]
* Mountains [http://pixelartmaker.com/art/442f9bc1fdcbf54]

# Appendix: WordSteal Rules (<http://wordsteal.mymesis.com/altpages/wordstealrules.htm>)

Wordox is a fast-paced, competitive word game played by 2 - 4 players. The object of the game is to form words from the given letter tiles, positioning them on the board to out-manoeuvre your opponents - or simply steal your opponent's words (and points).

## Setting up a Game

* To start a new game, click "Join" on any vacant table
* "Set Up" the table for your desired game by selecting "Options
* Select a "Winning Score" value, by default this is set to 50 Points for two player games.
* Select a "Time per Turn" value, by default this is set to 1:30 mins for two player games.
* Select whether you prefer the game to be observed by others or private, by default this is set to observe. (Note: Observers can read conversations between players as well as watching the game unfold.)
* You can join a table that someone has "Set Up" by clicking "Join"

## Basic Game Rules

* Each player plays from a pool of seven tiles selected randomly [from a starting set](http://wordsteal.mymesis.com/altpages/englishtiles.htm).
* The first player to take a turn must drag tiles to the centre of the board to form a word.
* Each tile must touch one another with one tile covering the centre spot. Words can be placed horizontally or vertically in a single row or column.
* Once a player has dragged their tiles to the centre of the board they must hit submit to create their word. If a word is considered as illegal then the tiles are removed from the board and play is passed to the next player.
* In turn each player must place horizontally or vertically a minimum of two tiles on the board to form a word.
* Players submit only one word per turn, although placed words may create more than one word on the board.
* Players' turns are indicated by both their name being highlighted and an audible sound to signify their turn of play.
* When a player's turn is completed and a word has been placed on the board then random generated tiles will appear in the rack.
* The timer will tick down the seconds each player has remaining for each play.

## Scoring

* Players placing words are awarded one point for each tile played. Players can add to their score by stealing points by adding to existing words in play, which in turn will remove points from their opposition.

*Example*

* + Player 1 places the word "help" = 4 points
  + Player 2 places "less" to create "helpless" = 4+4 points
  + Player 1 would have zero points while Player 2 would gain 8 points
  + If Player 1 could add "ness" to create "helplessness" thus scoring 12 points their opposition's score would return to zero.
* Points can be also be achieved by adding words parallel to existing words.

*Example*

READY

HELPLESSNESS

* + Player 2 would lose 3 points while player 1 would gain 8 points. 5 points for the word they placed "ready" and 3 for the additional 2 tile words they created, "re","es" & "as"
* Other points can be gained by placing tiles over orange coloured squares. These additional points are indicated with an icon below players' names and can only be won if you place a tile over a pink coloured square, which clears the board. If someone else clears the board then all bonus points you gained by covering orange squares are lost.

## Miscellaneous

One situation might cause confusion, so is worth clarifying:

* It is legal (and sometimes beneficial) to deliberately skip your turn even if you have a valid move available. Doing this is not considered bad sportsmanship. *Rationale: A 'No Skip' rule would not work, because it would be totally unenforcable, and would likely lead to arguments about whether someone skipping was deliberate or not.*

## Additional Rules

Originally, the game was played without any extra rules, but several variations have developed over time. These additional rules only take effect if the captain of the game selects them in the game Options before the game is started, or for one variation, announces it before the game starts. The captain can choose any combination of one or more of these rules. Players that do not agree with the rules the captain chooses or announces must quit either before the game starts or before they play their first move (ratings are not deducted). If the captain does not choose or mention a rule, it should be assumed it is not in effect.

* **No S**: Also called NS, this is a popular variation of play. No player may extend a word (not even their own word) by adding a single S to the end of the word. Note that it is OK to extend 'CATCH' to 'CATCHES', or 'TOP' to 'STOP', or 'PIN' to 'SPINS', but not OK to extend 'SPIN' to 'SPINS'. Extending 'THESE' to 'THESES' is also disallowed even though the two words are unrelated. However, extending single letters such as 'A' to 'AS' is OK. Any player who breaks this rule automatically loses the turn. Is is enforced by the server.
* **No Pinks**: Also called NP. It is not legal to clear the board. The pink clearing squares have a line in them to indicate they are not valid plays. Any player who plays a tile on a pink square automatically loses the turn. This is enforced by the server.
* **No S, No Pinks**: This combination is very popular. Some players play nothing else.
* **Two Tile/Three Tile**: A few players enjoy this a lot. The captain must announce it ahead of time, and all players must agree to the rule before the game starts. A player must play exactly 2 (or 3) of the 7 tiles on each move (or else skip). Any player who plays more tiles automatically loses the game, or some other penalty announced by the captain before the game begins (but this is currently not enforced by the server).

http://counter.dreamhost.com/cgi-bin/Count.cgi?df=wordstealrules1.dat&pad=F&ft=0&dd=E&istrip=Thttp://counter.dreamhost.com/cgi-bin/Count.cgi?df=wordstealrules1.dat&pad=F&ft=0&dd=E&istrip=T&display=date