Senior Project Game Design Document

Caleb Compton

[**I. Concept**](#_x0eipvo651er) **2**

[**II. Platforms and Tools**](#_pufehm4uwruo) **2**

[**III. Major Mechanics**](#_90unfgf3a7qh) **3**

[A. The Combat System](#_5v052qrsx35j) 3

[B. Characters](#_1yk21qskaxbp) 3

[C. Bosses](#_lmvjc0mum0p2) 3

[**IV. Story**](#_z09fw5coxmlo) **4**

[**V. Screens and Control flow**](#_49lrw18okeup) **4**

[A. The Start Menu](#_4c7rt19xk9p0) 4

[B. Load Game](#_936ipwkyt8by) 5

[C. Create Character Screen](#_uayduuklunij) 5

[**VI. Control Systems**](#_r7hmdjkdy4i5) **5**

[**VII. Music and Art Style**](#_p7pic1f2g8yf) **5**

[**VIII. Accessibility Considerations**](#_zatiggve0dy9) **6**

[**IX. Timeline**](#_1bvl64oqtjdo) **6**

[February 28th - Complete Demo](#_4w92ygirnm9x) 6

[March 15th - Implement Animation Class](#_beabthomx9zo) 7

[April 1st - Implement Characters and Skills](#_gu6iqants83r) 7

[April 15th - Implement Map and at least 1 boss](#_uvxguw2ch51i) 8

[April 30th - Improve Visuals + User Interface](#_tp0kk7rf9u27) 8

[Stretch Goals](#_2ysp8t1nb9kf) 8

[**X. Major Components**](#_k9ov7jrsmhgw) **9**

[Class Diagram](#_rnmcg9ai0wrc) 9

# I. Concept

The game can be broadly defined as a puzzle-rpg hybrid. The main gameplay mechanic of the game is to organize and collect spheres falling from the top of the screen to either remove them or use them for attacks. These attacks will be used to defeat enemies (including unique and challenging bosses), and progress from level to level. In order to remove spheres from the screen I am currently planning on simply requiring the player to connect 3 similar spheres in a straight line to remove them. In order to attack opponents, the player will have to collect spheres in a specific area of the screen and then drag them towards the enemy. The types and numbers of spheres that they collected will then determine the type and effectiveness of the attack on the enemy.

# II. Platforms and Tools

I am primarily targeting this application for mobile platforms, especially android devices. It is a relatively simple game that I think would work well as a mobile game, and some of the unique hardware features offered by modern mobile devices, such as integrated gyroscopes and accelerometers, could provide some interesting design space to explore if I am able to complete my primary design objectives.

Despite android devices being my primary target platform, I do want to develop my game in a way that can easily be ported to other platforms, such as IOS or even desktop devices. Because of this, I do not want to program a native android app. For these reasons, I have chosen to develop the game using the Processing language and IDE.

The processing language is a language that I already have some experience with, and it is designed to simplify visual programming. This makes it well suited to simple 2-D games. The processing for android development mode gives developers access to device specific capabilities such as cameras and gyroscopes. Finally, Processing code gets compiled directly into Java code, which can allow it to run natively on Android devices.

There is also a library known as Processing.JS, which allows processing code to be written inside of an HTML5 Canvas element as Javascript. This would allow portability for other types of platforms, such as desktop or IOS.

## Switch to Java

Towards the end of the project I decided to move the project to the Eclipse IDE and finish programming it in Java, using the Processing library. This would still give me access to all of the functions built into the processing language, but with a much more usable IDE. This change required some refactoring of the code, but will not otherwise change the project overall.

# III. Major Mechanics

## A. The Combat System

The most fundamental mechanic of the game is the arrangement of falling spheres on the screen to either remove them or organize them for attacks. This mechanic will make up the bulk of the gameplay, and as the player progresses through the game the difficulty and variety of these sessions will increase. Initially the player will only have to deal with a handful different types of spheres, which can be used to produce a few basic attacks. As the player progresses, they will unlock new attacks and abilities which will both enhance their capabilities as well as raise the overall challenge of the game.

## B. Characters

In addition to the main puzzle-fight sections of the game, there will also be an RPG component. The player will be able to choose from a handful of different starting classes at the beginning of the game, which will affect their starting capabilities. As the player progresses through the game, they will be able to unlock new abilities from the skill-tree. These abilities can include simple things such as raising certain stats, to more complicated skills such as unlocking special attacks that require particular combinations of spheres to create.

## C. Bosses

Finally, in order to add variety to the puzzle sections of the game the player will have to face a number of challenging bosses throughout the game. Each of these bosses will affect the gameplay in unique ways, which the player must adapt to in order to progress. Some boss ideas I have been considering include:

* A boss that freezes some of your spheres for a period of time (the spheres cannot be used or removed from the board while frozen)
* A boss that can control time - making your falling spheres fall much faster or much slower than they normally do, or even making them move backwards.
* A boss that has the ability to change spheres into different types after they have landed
* A boss that prevents your normal method of arranging spheres (the touchscreen / mouse), forcing you to tilt your phone left and right in order to get them to drop in the correct locations.
* A boss that flips the normal control scheme - runes fall up from the bottom instead of down from the top, and the control scheme is flipped (pressing the left button moves right, etc).

If time permits, I may also try to create a VS mode, where two players playing simultaneously try to reduce eachother’s health values to zero. This mode could possibly be on a single device, but more likely will involve players on separate devices connecting to eachother either through a matchmaking service or some sort of password system.

# IV. Story

While I haven’t fully fleshed out a story for the game yet, I do want there to be some level of narrative stakes that push your character forward through the game. Right now I am planning on having a relatively simple story - somebody that the player-character cares about gets kidnapped, and the player-character must set off on a quest to rescue them.

However, when implementing this story I wish to correct a major error I have noticed with this type of narrative structure. In many damsel in distress games, such as Mario or The Legend of Zelda the player really doesn’t get to know the character that they are rescuing until the end of the game, if at all. To rectify this, I am planning on spending some time with the kidnapped character at the beginning of the story and getting to know them first. This way, when they get taken away the player will feel a connection to them and really want to get them back.

# V. Screens and Control flow

The game will have a relatively simple control flow, and only a few different screens to deal with. The first screen is the startup menu, where the player will be able to select which mode they would like to play (campaign mode, VS mode, maybe a “Zen mode” where they just try to get a high score or a “Challenge mode” where they have to complete specific challenges). After selecting a mode, the player will move to the main play screen, which will be fundamentally the same from mode to mode. The only other major screens that will be necessary are the player-customization screen, the pause screen, perhaps a settings screen, and maybe a map screen for the campaign portion of the game.

## A. The Start Menu

The start menu is the first screen the player will see, and will be loaded automatically when the game is loaded. Currently, I want the Start Menu to have 2 different buttons. The first button will be to load a game, while the second button will be to start a new game. If the player has not yet created a character, the “Load Game” button will be dull and inactive. Otherwise, the player will be able to press either of these buttons. In the future I may put a limit on the number of characters that can be saved at one time (to reduce storage) but right now I will not check for this, and allow any number of characters to be created. When the “Load Game” button is pressed the game will then move to the Load Screen, while if the “Start New Game” button is pressed it will go to the “Create Character” screen.

## B. Load Game

The Load Game Screen will have buttons that can be clicked for each character that has been saved, with the most recently saved at the top. When a character is selected, that character file will be loaded from a text file (using either JSON or XML), and the data will be used to populate the Player object. Eventually these buttons may show more information about each character (such as their level, class, or game progress), but for now simply show the character’s name.

## C. Create Character Screen

For now, this screen will simply allow players to choose their character’s name. After choosing the character’s name, it will load a new screen which allows the player to choose their character class (Knight or Wizard). In the future, I plan to add a much larger amount of customization features.

## D. Map Scene

This scene will allow the player to choose what level they want to play. Beating levels will unlock the next levels on the map. The player can move with the arrow keys or simply select the level they would like to play.

# VI. Control Systems

I am planning on implementing a couple of different control systems for the game, and which will be used will depend on platform and player preference. For mobile devices, the primary interface will be the touch screen - players will drag spheres into place, then swipe attacks at their enemies. For desktop versions, the mouse will primarily take the place of the player’s finger to select and drag pieces around the screen. I am also planning on allowing the player to move the pieces with the arrow keys and attack with the space bar. For mobile devices I would emulate this control scheme using an on-screen arrow keys and an attack button, but this will be entirely optional for the player and can be toggled in the settings.

# VII. Music and Art Style

For now I am primarily planning on making the art assets myself, and will likely use some open source / public domain music and sound effects. Because of this, the initial art style is likely to consist of very simple hand drawn 2-D sprites and animations. If I choose to continue development of this game beyond the scope of this senior project I will them get more professional art and music.

# VIII. Accessibility Considerations

Accessibility in game design is very important to me, and for this reason I am going to try and build in a number of accessibility features into the game. One of these features is the option to switch between control schemes - while I plan for most players to swipe and drag the spheres around the screen, having a simple button based option will make it easier for certain players because they will be able to keep their hand relatively stationary.

I am also planning on making the game color-blind accessible. Because color is the main way to organize and distinguish different types of spheres, I am planning on mitigating this difficulty in two ways. Firstly, each sphere will also have a unique “rune” on it which distinguishes it and helps specify its function. Secondly, I am planning on making my color palette accessible, and provide settings for players to change it for various types of color blindness.

Finally, I am planning on designing the entire game to use high contrast colors for text, and to display all text in a large, readable font. I also plan to have options for adjusting the size of the text, as well as the speed, in the settings menu.

# IX. Timeline

## February 28th - Complete Demo

For this section I hope to have a simple prototype version of the system with the most basic functionality. Specifically, the game should have two screens - the start menu and the game screen. The start menu for the demo should simply have a play button which loads the game screen. The game screen itself should have a simple but functional version of the game itself, with the following features

* Spheres fall randomly from the top of the screen
* The player can control the spheres as they fall, and arrange them how they wish
* If three spheres connect in a row (vertically or horizontally) they will all disappear. Spheres that were above the removed spheres will fall down into the lowest available spots.
* Clearing spheres will add to a player’s **Action Bar**, until it is full
* Players will be able to attack enemies by pressing the attack button. This will create an attack based on the spheres in the **Spellbox**.
* Using an attack will reduce a player’s action bar. If the action bar is empty, players will not be able to attack.
* Attacks will usually reduce the enemy health, but could also cause status effects, heal the player, or various other effects.
* When the player wins an option will appear to play again or return to the main menu.
* Occasionally the enemy will attack and reduce the player’s health
* Players can also be damaged if their pile of spheres gets stacked too high
* If the player’s health gets reduced to zero, the player loses. This should bring up an option to play again or return to the main menu

To simplify the demo, I will be using simple shapes, minimal animation, and only three different types of spheres. The enemy and player will both be generic, and enemy attacks will be on a simple timer. The demo will have the following runes

* Health (yellow) - attacks with yellow runes will partially heal the player
* Fire (red) - attacks with red runes will burn the enemy, causing slight damage over time
* Slash (grey) - attacks with grey runes will cause immediate enemy damage

## March 15th - Implement Animation Class

The animation class will track moving objects, effects and animations. These animations can include things such as player and enemy movements, attack animations, and animation effects such as runes glowing before they disappear. The implementation of this class has been completed, and is using sprite sheets. Currently runes have their own form of animation, but I plan to migrate everything to sprite animations over time. Sprite animations allow a lot of flexibility, and the use of sprite sheets results in faster load times and less overall memory usage.

## April 1st - Implement Characters and Skills

Although a simple player class should be implemented by the time the demo is complete, it will be extremely basic and have no customizability. For this section of the project, I want to implement the following features

* A skill tree with several possible skill options for the player. Not all of these options will be available at first.
* A point system that allows players to “unlock” new skills by spending points
* Based on what skills the player has already unlocked, new abilities become available to them
* The player can gain skill points through experience, which results from defeating enemies.
* The game has a system in place to remember different characters and their skills. This will be implemented as a save system, which updates a character file in between different screens, and then loads that file at the beginning of the game.
* Players should be able to have multiple characters with different types of skills

## April 15th - Implement Map and at least 1 boss

The map and boss will help players get a sense of progression as they move through the game. Each level on the map will have slightly different parameters, such as enemy strength, grid size, or the speed the spheres fall at. I want this section of the map to have at least 4 different levels. The first three should ease players into the basic gameplay aspects of the game, and the fourth should be a boss that changes certain fundamental aspects of the game. The first levels should be relatively easy, and give the player opportunities to familiarize themselves with aspects of the game such as the skill tree. The boss should be more challenging, but should still not require more than 2 attempts to defeat.

## April 30th - Improve Visuals + User Interface

For this final section leading up to the public demo, I will mostly focus on improving the appearance of the application. This will most likely involve creating custom sprites and animations for the characters, enemy and bosses, as well as improving the visual appearance of the runes, menus, and the game screen itself. This section may also include fine-tuning of various aspects so that the game plays smoothly at the demo. It is also during this section that I will implement most of the accessibility features of the game, as most of them have to do with adjusting the visual interface of the game. By April 30th, I expect to have a simple but playable game with all of the major features necessary for the campaign mode.

## Stretch Goals

If time permits, there are several other features I would like to implement into the game. These include:

* Story features, such as simple text-based dialogue cutscenes and non-player characters
* More levels and bosses
* Additional game modes, such as free-play or competitive modes.

# X. Major Components

## Class Diagram

