CS 242 Final Project Proposal

This Healthy Beat: A Rhythm game

Ren-Jay Wang (rwang39), <Namrata Prabhu>
David Fu (davidfu2), <Chaoran Wang>
Kevin Li (kevinli3), <Chaoran Wang>

1. Abstract

1.1. Project Purpose

This project allows users to play a rhythm game similar to osu!

1.2. Background/Motivation

The people in this project enjoy rhythm games.

2. Technical Specifications

2.1. **Platform:** Desktop, Website

2.2. **Programming Languages:** Java/PHP/HTML

2.3. **Stylistic Conventions:** camelCase, Javadoc comments

2.4. **SDK:** LWJGL

2.5. **IDE:** Eclipse

2.6. **Tools/Interfaces:** Any Internet browser

2.7. **Target Audience:** Gamers, musically inclined people

3. Functional Specifications

3.1. Features

-Desktop

- ➢ The game itself, which scores users based on how many notes they hit accurately
- ➢ Scrolling menu with song list
- ➢ Users can upload songs to play

-Website

- ➢ Users must make an account on the website to upload their scores
- ➢ Website will list all songs with high scores
- ➢ Users can download extra songs from website

3.2. Scope of project

-Song sequences/patterns will not be overly complicated

-We will not be able to push updates after the user downloads the application.

-Graphics and animation will not be top quality, as the focus will be on functionality first

4. Timeline:

4.1. Week 1: Desktop: Should be able to see circles moving down corresponding to the notes of a song. Website: The website should show song list and high score chart. Users should be able to download songs from the website.

- Kevin
  - ➢ Make frame for application using LWJGL.
  - ➢ Write a method to send song note circles from the top of the screen to the bottom.
  - ➢ Make music play while notes are moving down.
- Ren-Jay
  - ➢ Parse song into sequence of beats, where beats are indicated by peaks of noise.
  - ➢ Write tests for different kinds of music to check for proper output format.
  - ➢ Create test music.
- David
  - ➢ Make basic website using HTML and PHP.
  - ➢ Includes menu which lists songs to be downloaded and a high score chart for each song.
  - ➢ Add test songs to be downloaded and test information in the chart.

4.2. Week 2: Desktop: Should be able to play the basic game. Notes move down and users press keys to hit them, getting points. The song ends if the user misses too many notes. Website: Users should be able to make accounts.

- Kevin
  - ➢ Make circles respond to key presses.
  - ➢ Create score system, which judges each press (“Perfect”, “Great”, “Good”, “Bad”, “Miss”) based on its timing.
  - ➢ End game loop if there are too many “misses”.
  - ➢ Make week 1 work pretty.
- Ren-Jay
  - ➢ Make different difficulties for songs.
  - ➢ Polish and make sequences more interesting (e.g. adding hold notes). This will be done by manually adding MidiEvents.
  - ➢ Create graphics for circles as well as background.
- David
  - ➢ Create registration system on the website using HTML/PHP
  - ➢ Create user database with SQL.
  - ➢ Test adding users to the database.

4.3. Week 3: Desktop: Add game transitions. Users should be able to choose a song from a menu, play it, be taken to a post-game screen, and then be able to choose another song. Website: User high scores should be uploaded to website.

- Kevin
  - ➢ Create end-game screen which updates user points.
  - ➢ Write code to smoothly transition between menu, game and end-game.
- Ren-Jay
  - ➢ Create scrolling song menu
  - ➢ Allow users to unlock songs with points.

- ➢ Edit song list so users' song lists only list songs they have unlocked.
- David
  - ➢ Upload high scores from the game to the website database after every game.
  - ➢ Test adding scores to the database
  - ➢ Website automatically shows update from database.

4.4. Week 4: Desktop: Polish game. Add musical/visual features. Users should be able to upload their own songs as well as those from the website. Points can be used to unlock songs.

- Kevin
  - ➢ Add musical effects to hitting notes, as well as opening/closing screen.
  - ➢ Create a function where users can request songs. Requests can be an mp3 upload or Youtube link. Requests are sent to database.
  - ➢ Test sending requests to database.
- Ren-Jay
  - ➢ Create login screen and query database for user
  - ➢ Manually parse more songs and find way to store them outside of program
- David
  - ➢ Implement database validation check for login screen.
  - ➢ Create song request database.
  - ➢ Make high score charts look nice and capable of accepting a lot of data values.

5. Future Enhancements

We'd like to create unlockable content as well as perhaps a story/campaign mode to motivate users to play more.