bRequirements and Analysis Document for the Beat-to-the-Beat project (RAD)

Contents

Version: 2.1

Date: 2014-05-22

Author: malinth, dhampus, hebjorn, ponerik

This version overrides all other versions.

https://github.com/HampusDahlin/Beat-to-the-Beat.git

1. Introduction

This section gives a brief overview of the project.

1.1 Purpose of application

This project aims to create a computer game where we will utilize music to give the player a dynamic experience. The game is going to be a “Beat ’em up”, so depending on the music’s beat you will have to time your punches in order to beat up the enemies. For definitions, terms and rules of the game see references.

1.2 General characteristics of application

The application will be a desktop, standalone (non-networked), single player application with a graphical user interface for the Windows/Mac/Linux platforms.

The application will feature dynamically real-time generated levels for every song being played. The playable character will not be able to move, instead enemies will be moving towards the player. When an enemy is getting close to the player s/he will have to hit the corresponding arrow-key to attack the incoming npc before taking damage himself. To complete a level one has to defeat all the enemies without dying. In order to die one has to fail several times at hitting the prompted buttons in time, causing the enemy to hit the player instead.

1.3 Scope of application

The application does not support multiplayer. The application will save score at the end of each song played, but is unable to save in the middle of one.

1.4 Objectives and success criteria of the project

1. It should be possible to play through a song killing all of the enemies to the beat of the music.
2. The player should be able to play the game using his or her own music.

1.5 Definition, acronyms and abbreviations

- GUI: Graphical User Interface
- Beat ’em up game: A game where the main focus is beating up enemies.
- NPC: Non Player Character. A character in the game controlled by the computer.
- Java: platform independent programming language.
- JRE: Java Runtime Environment. Software needed to run as Java application.
- Host: a computer where the game will run.

2. Requirements

2.1 Functional requirements

The player should be able to:
1. Start a new song
2. Load a song into the game from a local library.
3. Play the game. During a playthrough the player should be able to:
    - Kill enemies by pressing the prompted button.
    - Win by killing all incoming enemies.
    - Take damage.
    - Regain lost health between songs.
4. Exit the application.

2.2 Non-functional requirements

2.2.1 Usability

Usability is a high priority. Normal users should be able to play the game within a very short time period. The game is simple enough that most players should be able to grasp the idea quickly.

2.2.2 Reliability

NA

2.2.3 Performance

The game has to be responsive to a player's interaction, after a player initiates an action the game must respond at worst in 0.60 sec.

2.2.4 Supportability

NA

2.2.5 Implementation

To achieve platform independence the application will use the Java environment. All hosts must have the JRE installed and configured. The application must be located on each separate host that wishes to run it.

2.2.6 Packaging and installation

The program will not be installed, instead it will be a single file that the user runs to start the game.

2.2.7 Legal

The program will not be distributed with copyright marked music, instead it will come with a small selection of free music samples.

2.3 Application models

2.3.1 Use case model

See APPENDIX.

2.3.2 Use case priority

1. HitButton
2. Start next level
3. Finish level
4. Load song

2.3.3 Analysis model

See APPENDIX.

2.3.4 User Interface

The application will use a GUI following standard conventions.
See APPENDIX for screens and navigational paths.

Appendix:

Usecases:

overview

Use Case Texts

Use Case: Attack

Summary
This is how the player carries out an attack. This UC is always preceded by the Start Game UC.

Priority
High

Extends
-

Includes
UC Kill, UC Miss

Participators

Normal flow of events
Player presses the attack button and hits an enemy.

User...	Computer...
1. Clicks the attack button	
	2. Starts the attack animation
	3. Resolves hit detection from range and direction
4. Player hits an enemy	
	5. UC Kill

Alternate flow of events

Flow 2.2, attack cooldown timer is active

player	computer
	2.2.1. Skips all following steps

Flow 4.2, player misses the enemy

player	computer
4.2.1 Player misses	
	4.2.2. UC Miss

Use Case: Kill

Summary
A player's attack connects, leading to the death of its target.

Priority
High

Extends
-

Includes
-

Participators
Player, NPC

Normal flow of events

player	computer
1. hits the enemy with UC Attack	
	2. removes the enemy
	3. increases the players score by a value determined by the distance to the player and the current combovalue, then increases the players combo by one
	4. gives the player any powerups that she is eligeble for. A player is eligible for a powerup when her score increases above a multiple of the powerups activation threshold.

Use Case: Miss

Summary
The players attack misses, leading to the initiation of an attack cooldown as well as the resetting of the players combo.

Priority
Medium

Extends
-

Includes
-

Participators
Player

Normal flow of events

player	computer
1. Misses with UC Attack	
	2. Starts attack cooldown, making the player unable to attack for a certain amount of time
	3. Sets the players combo to 0

Use Case: Recieve Damage

Summary
An NPC touches the player, leading to the removal of the NPC and damgage being dealt to the player.

Priority
High

Extends
-

Includes
-

Participators
Player, NPC

Normal flow of events

player	computer
1. gets hit by an enemy by getting in contact with it	
	2. deals one damage to the player
	3. sets players combo to 0
	4. removes the NPC

Alternate flow of events

Flow 2.2, HP reaches 0

player	computer
	2.2.1. UC Death

Use Case: Death

Summary
The player takes damage, leading to her HP falling below one and her death. Computer responds by displaying the score screen.

Priority
High

Extends
-

Includes
UC Recieve Damage

Participators
Player

Normal flow of events

User	Computer
1. Dies	
	2. Shows death animation
	3. Displays score screen, where previous high scores is displayed.
4. Press continue	
	5 Returns user to song selection

Use Case: Finish a Level

Summary
The user gets through the song without dying, leading to his score being considered for high score and the display of the score screen.

Priority
High

Extends
-

Includes
-

Participators
User

Normal flow of events

User	Computer
1. Finishes a level.	
	2. Display score screen with previous high scores.
3. Press continue	
	4. Returns to song selection

Alternate flow of events

Flow 2.2, The players score gets onto the high score list

player	computer
	2.2.1 Adds the players score to the high score list and then displays it.

Use Case: Options

Summary
The user goes into the options menu and changes an option.

Priority
Low

Extends
-

Includes
-

Participators
User

Normal flow of events

User	Computer
1. presses the “Options” button	
	2. shows the Options screen where the player is able to change the intensity of the background or return to main menu.
3. changes an option.	
	4. Sets the changed field to its new value and saves to conf file.

Alternate flow of events

Flow 3.2, The players exits the menu

User	Computer
3.2.1. Presses cancel	
	3.2.2 Returns to main menu

Use Case: Pause

Summary
The player pauses the game by pressing escape and then resumes it.

Priority
low

Extends
-

Includes
-

Participators
Player

Normal flow of events

User	Computer
1. Press escape	
	2. pauses music, game timer and shows pause panel with buttons for resuming and quiting.
3 Press resume	
	4 music and game timer resumes and hides pause panel.

Alternate flow of events

Flow 3.2, The players exits the level

User	Computer
3.2.1 Press quit	
	3.2.2 goes to score screen where old high scores are being shown.

Use Case: Upload Song

Summary
The user goes to the song upload screen to choose music from one of her own libraries to add to the game.

Priority
medium

Extends
-

Includes
-

Participators
User

Normal flow of events

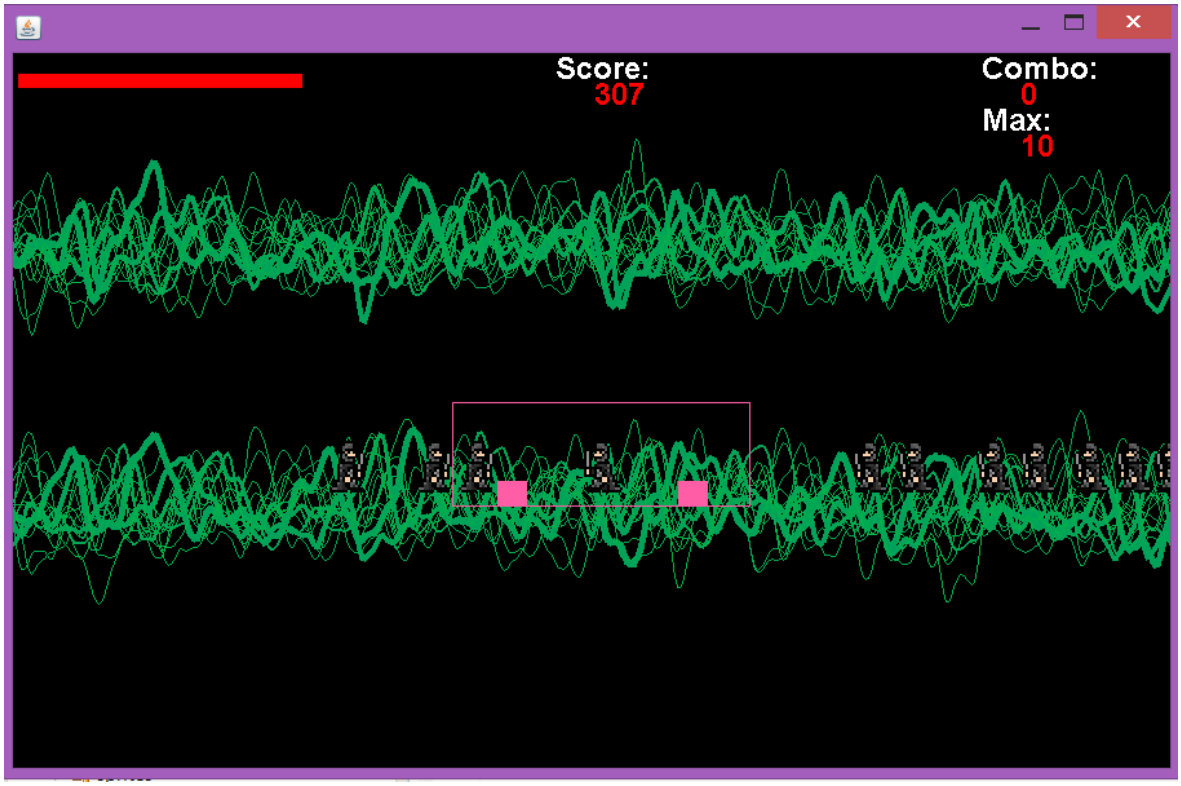
Player	Computer
1. Press the upload song button	
	2. Displays the song upload screen.
3. Press the file selector button	
	4. shows a file selection screen.
5. Chooses a song to load into the game.	
	6. Gets the songs url and adds it to the song list.
	7. Shows song selection screen.

Alternate flow of events

Flow 3.2, The players exits the menu

User	Computer
3.2.1. Presses cancel	
	3.2.2 Returns to main menu

GUI


Score:
307
Combo:
0
Max:
10


Beat
To The
Beat
Play Song
Upload Song
Options
Exit

Analysis Model