Requirement and Analysis Document for RunningMan

Contents

Version: 0.8
Date: 2015-04-02
Author: Armand Ghaffarpour, Jesper Olsson, Johan Tobin, Simon Lindkvist

This version overrides all previous versions.

1. Introduction

1.1 Purpose of application

The purpose of the application is purely to entertain. It will specifically be a fast paced single player platform game which might resemble Super Mario Bros to some extent. The game should be fairly intuitive for everyone above the age of 15 which also is our target audience.

1.2 General characteristics of application

The application will be a desktop, stand-alone, keyboard-controlled computer game. The user will control one character, and shall, via simple keyboard commands, advance through a level full of various obstacles and enemies. The different levels will be time based, and the user must complete a level in time in order to get a score registered. The application will keep track of the best scores for each level. The more time used, the lower the score. At the end of each level, if reached in time, there will be a helicopter to pick the character up, and thus, ending that level.

1.3 Scope of application

The game is single player only. It will contain two levels. The user can only choose one character. There will be two types of enemies, one which is able to both shoot and move and one that is only able to move.

1.4 Objectives and success criteria of the project

The player should be able to move to the right and to jump. The player should also be able to move left to a certain extent. That is, only in the current scope of the game-view. The scope will change and follow the player as the player moves to the right.

1.5 Definitions, acronyms and abbreviations

GUI - Graphical User Interface
Level - A map/course which the player is supposed to finish
Java - Platform independent programming language
Map- Group of tiles put together in order to create a visual section
Score - Time based score
UML - Unified Modeling Language, a diagram showing relations between classes, packages and their properties
Use Case - List of steps to show how the application will interact with the user to achieve a specific goal
HP - Health Points, is the characters or enemies health measured in points. Zero health points leads to death.

2. Requirements

2.1 Functional requirements

The user should be able to:

1. Start a new game.
   a. Move the character.
   b. Shoot bullets.
   c. Kill enemies by shooting bullets at them.
   d. Pick up power ups.
   e. Move faster by picking up power ups.
   f. Get killed by enemies by collide into them.
   g. Die by falling down into a pit.
   h. Complete the level by moving to right to the end of the level.
   i. Enter the second level by finishing the first level.
   j. Complete the game by completing the second level.
2. Resume a previous game.
3. View the high score.
4. Quit the game.
5. View the main menu.
6. View the ingame menu.

2.2 Non-functional requirements

2.2.1 Usability

2.2.2 Reliability

2.2.3 Performance

2.2.4 Supportability

2.2.5 Implementation

2.2.6 Packaging and installation

2.2.7 Legal

2.3 Application models

2.3.1 Use case model

UML and a list of UC names (text for all in appendix)

2.3.2 Use cases priority

1. Move
2. Finish level
3. Fall down (die)
4. Collide with enemy (die)
5. Shoot
6. Menu
7. New Game
8. View high score
9. Quit
10. Power up
11. Resume game

2.3.3 Domain model

See *Appendix*.

2.3.4 User interface

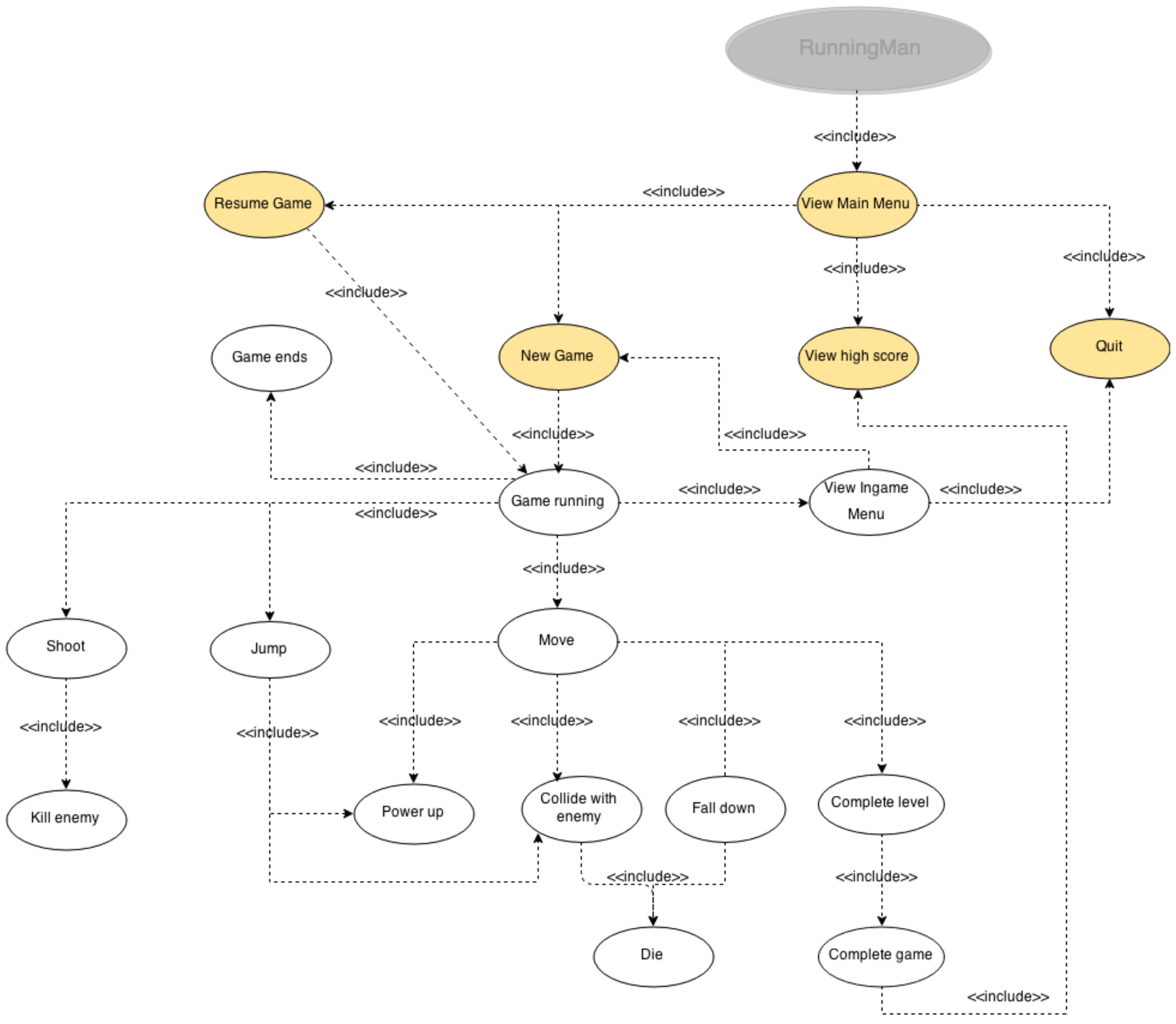
2.4 References

APPENDIX

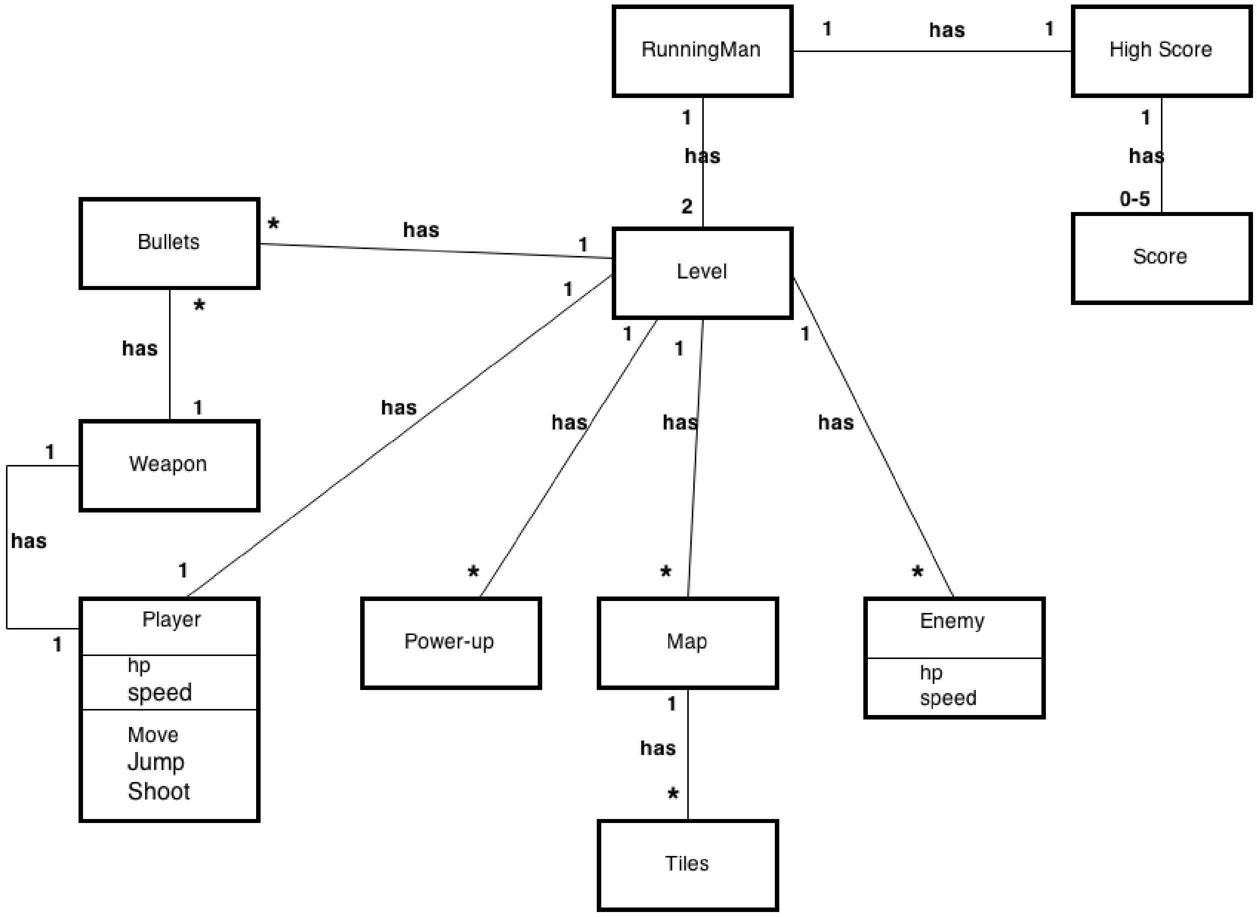
GUI

ToDo: GUI picture

Use Case diagram



Domain Model



Use Cases

User case: Move

Summary: How the player moves
Priority (high,mid,low): High
Extends:
Includes: Power up, Fall down, Collide with enemy

Normal flow:
The player moves around freely

	User	System
1	Press or holds left or right on the keyboard.	
2		The character moves in the given direction.

Alternate flow:
Flow 2.1: The character is next to a wall and tries to move in that direction

	User	System
2.1.1		The character will have a collision with the wall
2.1.2		The character won’t move in that direction

Alternate flow:
Flow 2.2: The character steps on a power up

	User	System
2.2.1		The character picks up the power up
2.2.2		The character gains the given power

Alternate flow:
Flow 2.3: The character falls down

	User	System
2.3.1		The character moves to a point where there's no ground
2.3.2		The character falls down

Alternate flow:
Flow 2.4: The character collides with enemy

	User	System
2.4.1		The character will have a collision with an enemy
2.4.2		The character dies

User case: Jump

Summary: How the player jumps

Priority (high,mid,low): High

Extends:

Includes: Power up, Collide with enemy

Normal flow:
The player moves around freely

	User	System
1	Press space or up on the keyboard.	
2		The character jumps.

Alternate flow:
Flow 2.1: The character tries to jump but is totally blocked from above

	User	System
2.1.1		There is an object blocking directly from above
2.1.2		The character won't jump

Alternate flow:
Flow 2.2: The character tries to jump but is partly blocked

	User	System
2.2.1		There is an object blocking from above
2.2.2		The character jumps and hits the object

Alternate flow:
Flow 2.3: The character jumps and hits a power up

	User	System
2.3.1		The character jumps and picks up the power up
2.3.2		The character gains the given power

Alternate flow:
Flow 2.4: The character jumps and collides with an enemy

	User	System
2.4.1		The character jumps and hits an enemy
2.4.2		The character dies

User case: Complete level

Summary: The user completes the level and moves on to the next one

Priority (high,mid,low): high

Extends: Move

Includes:

Normal flow:

The user reaches the end of a level

	User	System
1	User moves to the right and reaches the final point of the level	
2		A message tells the user that the level is finished
3		Saves the score of the level
4		Shows the score of the level
5	The user clicks on the arrow	
6		The message disappears
7		The next level starts

User case: Complete game

Summary: The user completes the game(the last level)
Priority (high,mid,low): high
Extends: Move
Includes:

Normal flow:
The user reaches the end of the last level

	User	System
1	User moves to the right and reaches the final point of the last level	
2		A message tells the user that the level is finished
3		Saves the score of the level
4		Shows the score of the current level
5		Gets the total score of all of the cleared levels
6		Shows the total score
7		The list of the high score list is displayed with a replay button and menu button
8	The user press replay button	
9		A new game starts

Alternate flow:
Flow 6.1: The user gets a new high score

	User	System
6.1.1		A message displaying the users score and a place to enter the users initials appears.
6.1.2	User enter initials and press enter	
6.1.3		Saves the High score and initals

Alternate flow:
Flow 8.1: The user press menu button button

	User	System
8.1.1	User presses menu button	
8.1.2		The menu appears

User case: Shoot

Summary: The character fires a shot
Priority (high,mid,low): mid
Extends:
Includes: Kill enemy

Normal flow:
The user fires a shot

	User	System
1	Press or hold the CTRL button	
2		Shows the character fires a shot

Alternate flow:
Flow 2.1: The shot hits an enemy whom dies

	User	System
2.1.1		An enemy is hit by the shot
2.1.2		The enemy dies and can no longer harm the user

Alternate flow:
Flow 2.1.1: The shot hits an enemy whom doesn’t die

	User	System
2.1.1.1		The enemy takes damage
2.1.1.2		Calculate new health of enemy

User case: New Game

Summary: Starts a new game at the first level
Priority (high,mid,low): low
Extends:
Includes:

Normal flow:
The game starts from the beginning

	User	System
1	At the main menu or in-game menu, the user selects option “New Game”	
2		Starts a new game from the first level

User case: View High Score

Summary: The user views the high score
Priority (high,mid,low): low
Extends:
Includes:

Normal flow:
The user views all high scores for all levels

	User	System
1	At the main menu, the user selects the option “View High Score”	
2		System displays high score list
3	Press back button	
4		System shows user the menu

User case: Quit

Summary: The user may quit
Priority (high,mid,low): low
Extends:
Includes:

Normal flow: The user quits the game

	User	System
1	At the main menu or the in-game menu the user selects the option“Quit” (or in game quit, close window)	
2		If user is in game, system show confirm dialog
3	User selects “Yes” option	
4		System saves the current state
5		System exits the application

Alternate flow
Flow 3.1: The user selects “No” option and thus not quitting the game

	User	System
3.1.1	The user selects the “No” option	
3.1.2		Resumes the game

User case: View In-game Menu

Summary: During a game the user may look at the menu
Priority (high,mid,low): low
Extends:
Includes:

Normal flow:

	User	System
1	The user presses ESC-button in game	
2		The menu is displayed

User case: Resume game

Summary: If the user has exited in the middle of a game, he should be able to resume from the same place.
Priority (high,mid,low): low
Extends:
Includes:

Normal flow:

	User	System
1	The user presses “Resume Game” from the main menu	
2		The systems restore the state from when the game was last played