Requirements and Analysis Document for the Tetrix project (RAD)

Contents

1 Introduction 2

1.1 Purpose of application 2

1.2 General characteristics of application 2

1.3 Scope of application 2

1.4 Objectives and success criteria of the project 2

1.5 Definitions and acronyms 2

2 Requirements 3

2.1 Functional requirements 3

2.2 Non-functional requirements 3

2.2.1 Usability 3

2.2.2 Reliability 3

2.2.3 Performance 3

2.2.4 Supportability 3

2.2.5 Implementation 3

2.2.6 Packaging 4

2.2.7 Legal 4

2.3 Application models 4

2.3.1 Use case model 4

2.3.2 Use case priority 4

2.3.3 Domain model 4

2.3.4 User interface 4

2.4 References 4

Appendix 5

Version 1.0
Date 2012-05-21
Author Linus Karlsson, Andreas Karlberg, Magnus Huttu, Jonathan Kara

This overrides all previous versions

1 Introduction

The project aims to create a Tetris inspired game. Generic in the sense that it should be possible to adapt the game to different locations and more, see further below. For definitions and terms of the game see references.

1.1 Purpose of application

Instead of moving the tetrominoes as you would in a normal Tetris game, you move a cannon around a panel in which the tetrominoes are falling towards the bottom. To get points, one should fire bullets with the cannon and shape them in a way so that they are fully covering a row.

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. It will wait for the user to control the cannon. When the player is firing bullets at the falling tetrominoes, the hit part of it will disappear. When they are hitting the bottom of the "Tetris-box", or if they get stacked on top each other, they will "freeze" and won't be shootable. When the tetrominoes are aligned in a way so that they cover a full row, the row will disappear and the player will receive points.

The game will end when the stacked blocks are reaching the top of the window, or possibly be canceled. If the game is cancelled, the player's score will (if good enough) be saved to a high score list.

1.3 Scope of application

The application does not include a computer-player. It is only possible to play the game alone. The application does save the player's name and a list of the ten highest scores.

1.4 Objectives and success criteria of the project

It should be possible to play a game where you can move a cannon and fire bullets at falling blocks in different shapes. There should be at least 2 sets of themes to change between. The cannon should be easy to maneuver with the left and right arrows. You should also be able to choose difficulty and view high scores on the graphical user interface.

1.5 Definitions and acronyms

- Tetris**,** an old invention of a Russian mathematician who wanted to invent something that could bright up in boring times. It's a game where you try to fit different shapes to each other without leaving any space between them. When a full row is filled, it will disappear and the blocks on top will fall down one step.
- Tetrix, a new spiced up version of Tetris where you, instead of controlling the falling tetrominoes, is controlling a cannon and its bullet launcher.
- Cannon, used to shoot down the blocks and will move around the Tetrix box on a fixed track. When a bullet hits a tetromino, the hit square will disappear. One can keep firing bullets until the whole tetromino is gone, or until you are satisfied with the shape.
- Tetromino (block), shootable block in the 7 different shapes. Each shape is made up of four squares. When the shape reaches the ground or if it touches another shape it will freeze.
- Tetris-box, where the tetrominoes will be falling. When the blocks are stacked so that they reach the top of the box, the game will end. The panel is divided into 10x20 cells.

- Freezed, the state where the blocks are unshootable, meaning that hitting them with a bullet won't result in anything.
- Bullet, what the Cannon is firing on the user's input.
- Highscore, a list of the ten best played games.

2 Requirements

In this section we specify all requirements.

2.1 Functional requirements

The player should be able to:

1) Change settings
  a) Determine the volume of the sound effects and background music
  b) Change theme
2) View high score's.
3) Start a new game
  a) Select level (easy or hard)
4) Control the cannon
  a) Move the around the "Tetris box" cannon with the keys.
  b) Shoot laser on the blocks to form them as you want.
5) Exit the application

2.2 Non-functional requirements

2.2.1 Usability

Usability is high priority. Normal users should be able to play the game within a very short period. The application must be self-explanatory in such way that the user instantly feels superior, no matter if he/she is an experienced computer user or not.

2.2.2 Reliability

NA

2.2.3 Performance

Any action initiated by the player should immediately be updated on the screen, meaning that the response time is very low.

2.2.4 Supportability

The application must be implemented so that the GUI is easily modifiable to suit other platforms (Web, Mobile Apps, Pads, etc.). The implementation should prepare for the dividing of the application into a client-server architecture for net based games.

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 needs to be installed on all hosts where it will rusn (possibly downloaded).

2.2.6 Packaging

The application will be delivered as a zip-archive containing;

1. A file for the application code (a standard Java jar-file).
2. All needed resources, internationalization and localization.
3. A README-file documenting installation and start of application.

2.2.7 Legal

There could be legal issues due to the similarity with Tetris and its trademark. This is not covered here.

2.3 Application models

2.3.1 Use case model

See APPENDIX for UML diagram and textual descriptions.

2.3.2 Use case priority

1. New Game
2. Falling Blocks
3. Move
4. Shoot

2.3.3 Domain model

See APPENDIX

2.3.4 User interface

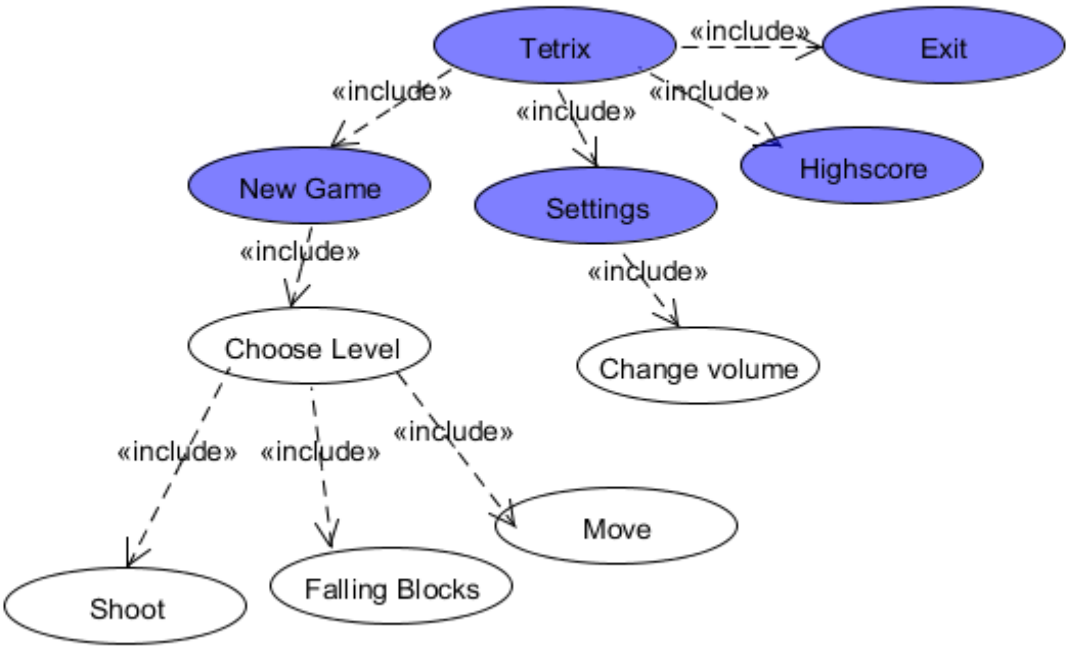
The application will be using a changeable GUI, where you can change things like the theme. The GUI will be offering one screen size only. See APPENDIX for GUI.

2.4 References

1. Tetris: http://en.wikipedia.org/wiki/Tetris
2. Tetromino: http://en.wikipedia.org/wiki/Tetromino

Appendix

Use cases



Use case model

Use case: Move

Short description: How a user moves the cannon.
Priority: High
Extends or Includes: NA
Participating actors: Actual player (AP).

Normal flow of events:

Actor	**System**
AP presses right or left arrow key	
	The cannon move in given direction as long as the key is pressed.

Use case: New Game

Short description: How to a launch a new game.
Priority: High
Extends or Includes: NA
Participating actors: Actual player (AP).

Normal flow of events:

Actor	**System**
AP makes a decision to start a new game	
	The application displays two different levels for the user to choose from – easy and hard.
AP makes his/her choice	
	The game starts with the desired difficulty.

Alternative flow of events:

Actor	**System**
AP has either paused a running game, or is in the Game Over view and decides to play the game once more	
	The application starts with the same difficulty as the previous game.

Use case: Falling blocks

Short description: These are some of the main objects in this game since they can make you lose the game, and win the game.
Priority: High
Extends or includes: Extends StartNewGame.
Participating actors: Tetrix-box

Normal flow of events:

Actor	**System**
A new tetromino is randomly placed at the top of the panel.	
	Tetromino is moving towards the panel's bottom at a given speed.
	The block is either colliding with another shape or reaching the bottom, making the whole tetromino freezed and unshootable.

Alternate flow:

Actor	**System**
A new tetromino is randomly placed at the top of the panel.	
	Tetromino is moving towards the panel's bottom at a given speed.
	The block is either colliding with another shape or reaching the bottom, making the hit square freezed and unshootable while the other part continues to fall like a whole tetromino until it collides with the ground or a freezed object.

Exceptional flow:

Actor	**System**
A new tetromino is randomly placed at the top of the panel.	
	There's a freezed tetromino at the same place, resulting in Game Over.

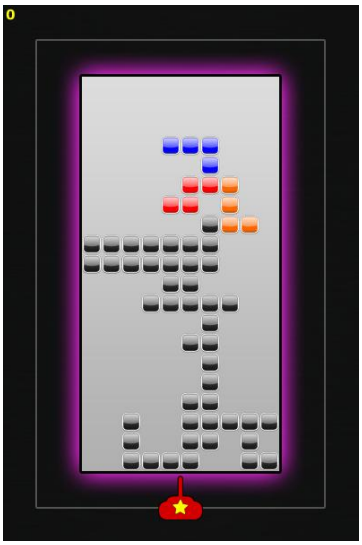
Use case: Shoot

Short description: How a user shoots with the cannon
Priority: Medium
Extends or Includes: NA
Participating actors: Actual player (AP)

Normal flow of events:

Actor	**System**
AP presses down SPACE to shoot	
	The cannon fire a bullet in the direction of the cannon.

GUI



Domain model