Requirements and Analysis Document for

‘Tis but a Scratch!

Contents

Version: 1.0

Date: 14-03-30

Author: Anna Nylander, Alma Ottedag, André Samuelsson, Ivar Josefsson

This version overrides all previous versions.

1. Introduction

This section gives a brief overview of the project.

1.1 Purpose of application

The purpose of the game is to create an topdown 2D adventure game in wich two players can cooperate in completing puzzles and fighting enemies located in seperated areas.

1.2 General characteristics of application

The application will be a desktop, standalone and multi-player application with a graphical user interface for the Windows/Linux enviroment.

The application will be a realtime mutliplayer game with network in which the player will be able to move freely over the board, divided in a grid like the classic Pokemon och The Legend of Zelda games. The player will also be able to fight enemies and interact with diffrent objects.

1.3 Scope of application

The application will not be able to save started games or collect any terminated games. The game will only use keyboard and therefore not support mouse or any other controllers. You will only be able to play mutliplayer on seperate PC's, you cannot play multiplayer on the same PC. The project will not implementent possibility for several players on the same computer and it is not within the scope to implement sound either.

1.4 Objectives and success criteria of the project

Multiplayer experience
Defeat enemies
Change rooms
Solve puzzles

1.5 Definitions, acronyms and abbreviations

Player = a user of the program and his or her corresponding representation in-game.

2. Requirements

2.1 Functional requirements

The application should be able to run at 60fps with support for multiplayer gaming on two different computers. The player must be able to move, attack and interact with objects in its environment.
The players shall further be able to:
1: Host a game, excluding kicking players from it.
2: Incapacitate an enemy.
3: Pick up items: either keys that will be stashed, or gear that will be automatically equipped by the player.
4 Go through doors independently of other players.
5 Join a game
6 Take damage, excluding friendly fire.
7 Solve puzzles including for example moving blocks and finding switches.
8 Unlock doors with keys kept by the player.

2.2 Non-functional requirements

2.2.1 Usability

The application will have a high focus on usability. The primary language in the game will be english and it is assumed that the user is somewhat familiar with games and other graphical applications. The game will primarily utilise graphics to convey information and text will be a secondary way to convey information.

The game will implement text instructions in the menu but the game will primarily be user friendly through intuitive controls and interface.

2.2.2 Reliability

NA - the game will have no particular reliability concerns

2.2.3 Performance

Any action performed by a user should have a maximum of 0.2 seconds in worst case scenario, the median response time should be half of that at 0.1

2.2.4 Supportability

The application will be built as a desktop application but should to expand for use on other platforms with relative ease. The graphics library used in the prototype does however already have internal support for use on other platforms, therefore it should be possible to add more definite support for other platforms. Implementing this support will however not be a main concern during the construction.

2.2.5 Implementation

The application will use the Java enviroment, in order to run the application the user will need to have JRE (Java Runtime Enviroment) installed and configured. The application will have an executable file that it can be run through, no installation required.

2.2.6 Packaging and installation

The application will be delivered in a zip archive containing:

1. An executable file for running the program
2. All needed resources (maps, sprites etc.)

2.2.7 Legal

There should be no legal issues

2.3 Application models

2.3.1 Use case model

RunGame, SetGameOptions, QuitGame, StartProgram, TakeDamage, UseEnvironmentalObject, UnlockDoor, Attack, GoThroughDoor, JoinGame, Interact, HostGame, MoveCharacter, KillEnemy, PicUpItem.UML and a list of UC names (text for all in appendix)

2.3.2 Use cases priority

1. Move
2. Attack
3. Pick up item
4. Go through door
5. Unlock door

2.3.3 Domain model

See Appendix.

2.3.4 User interface

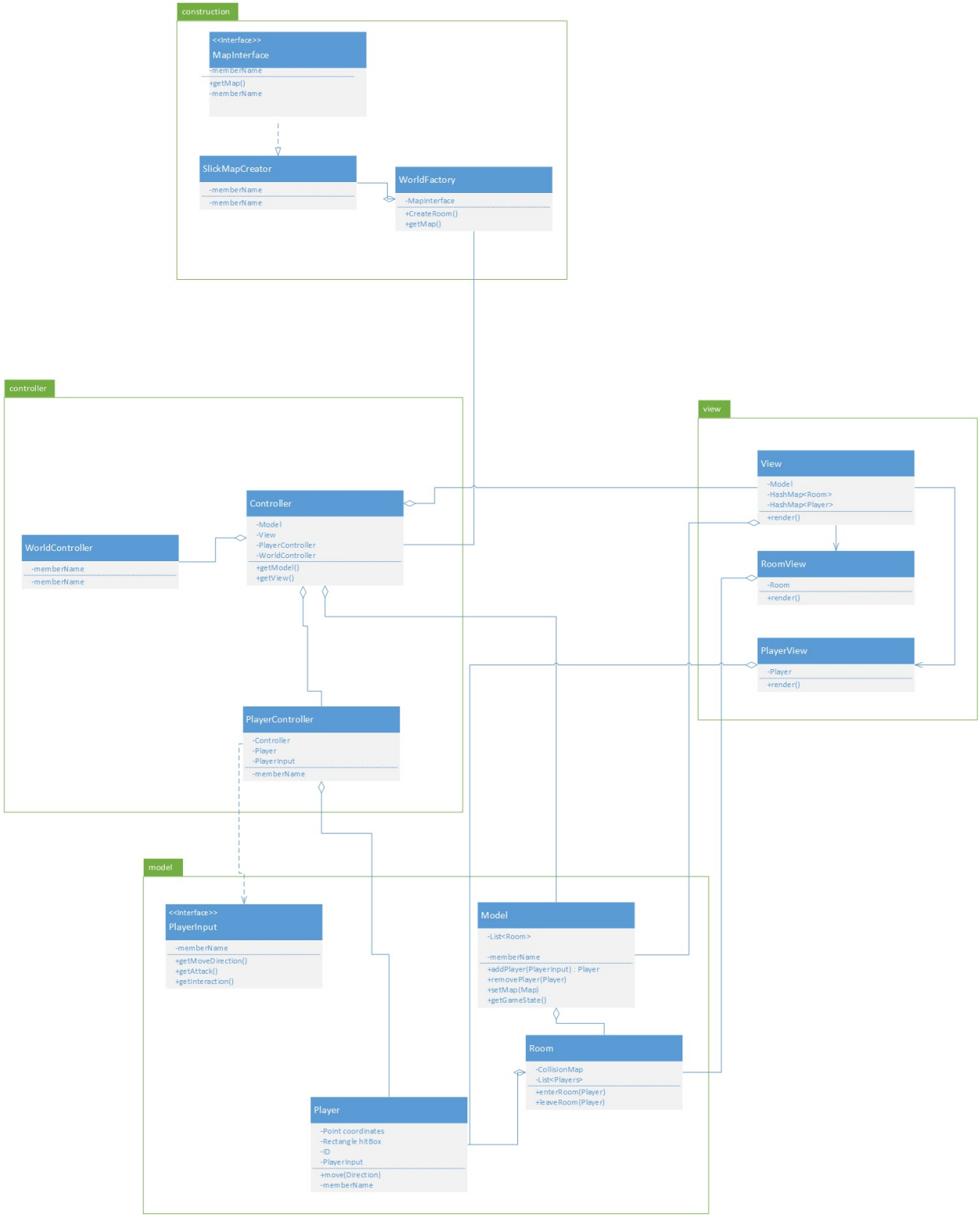
The main window will be seen from above with a projected side-view for characters and objects. Doors will be visible as if the edges of the screen tilt up, such that the player can distinguish a door on all sides of a room. The game set will further be a grid. Each square in the grid will be 120x120px adhering foremost to a screen ratio of 16:9.

2.4 Referenes

APPENDIX

GUI

Domain model



Use case texts