

# Requirements and Analysis Document for AwesomeGame

Version: 1.0

Date: 25/05-18

Author: Farzad Besharati, Therese Sturesson, Philip Nilsson, Linus Wallman

**This version overrides all previous versions.**

## 1 Introduction

The current RPG market on PC is lacking in what could be labeled as “Old School” RPG’s. The popular RPG’s of today are often mechanically too easy or linear in design. Our goal is to design and develop a game that caters to fans of games that became popular during the mid to late 1980’s, such as the original Legend of Zelda for the Nintendo Entertainment System.

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

Some general characteristics:

- It is a one player game.
- The application will be able to save the game, so you will be able to quit the game and come back later.
- It will be an open world game, meaning you will be able to walk around freely on the map.
- The GUI will be simple and easy to use. It will have a start menu and when you start the game it will have a bar with information and items and a box where you play.
- There will be enemies (handled by the computer) to defeat and items to pick up and use.
- Your character will die and the game will end if your health drops to 0.
- There is no time constraints at all in the game.
- To win the game you have to kill the boss.
- The map is going to be divided up into several segments. A segment corresponds to 32x16 “tiles” which also defines what the player can see. In every segment, there are different things to do depending on its contents. For example, there might be obstacles blocking you from picking up a special item.

## 1.2 Definitions, acronyms and abbreviations

Technical definitions:

- JRE: Java Runtime Environment
- PE: Portable Executable, see references

- Jar: package file format used to aggregate Java class files

Most definitions and terms regarding the core game:

- Player: Representative of the person playing the game
- AI: Representative of entities that aren't the Player
- World: The environment in which the game takes place
- Tile: An even-sided square, the world consists of these
- Character: An entity in the world, could be the AI or the Player
- Player Character (PC): A Character representing the Player
- AI Character: A Character representing the AI

## 2 Requirements

### 2.1 Graphical user interface

First off the application will have a fixed size and a fixed GUI that you will not be able to change. The start menu will fill up the whole window. The menu will have four big buttons where you can choose to start the game, Exit the game, or other things. If you click on the "Exit" button the application will stop and close the window.

Start menu:



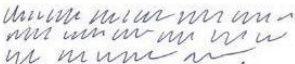
If you click on the "How to play" the menu screen will be replaced by the How to play the game screen. Here you can read about the game and how to play it. There will be a "Go back" button and when you press it you will be go back to the menu screen.

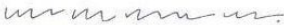
If you click on the "How to play" button:


<Go b2ch

# How to play the game

Messa text här som förklarar hur man spelar spelet. ann an an an an  
an an an an an an an an.

If you click on the “Settings” button you will get to a new screen where you can settings like resolution, music volume and so on. Then you can click on the “Go back” button and you will get back to the menu screen.

If you click on the "Settings" button:

A hand-drawn settings menu. At the top left is a button labeled "< Go back". To its right is the title "Settings" underlined. Below the title are two settings: "Resolution" with a dropdown menu showing "999 x 999 V", and "Game music" with a slider set to the left. Below these is "Game effects" with a slider set to the right. At the bottom is the text "(More settings here if we want to)".

When you click on the “Load game” button the application will change to a new screen where you can see a list of all your saved games. If you click on the “Go back” you will go back to the menu screen. If you click on one of the saved games the game will start with all the saved items, lives and so on.

If you click on the "Load game" button:

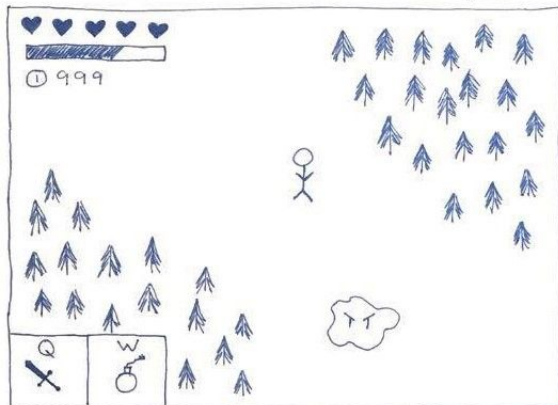
< Go back

## Load game

Therese	26/4-18	△
♥♥♥♥		
Linus	22/4-18	
♥♥♥		
Ferzed	13/4-18	
♥♥♥♥♥♥♥		
Philip	7/4-18	
♥♥		
Ida	29/3-18	▽

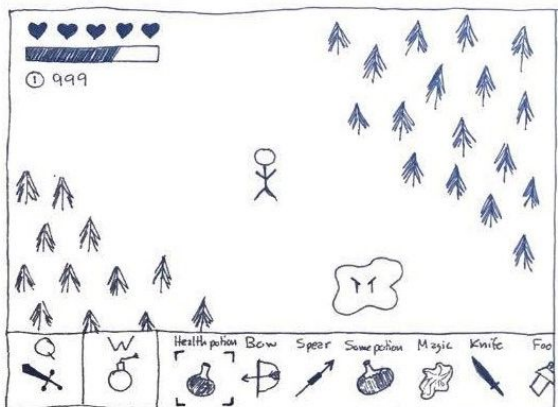
If you have clicked on the “New game”, “Continue” or a saved game i the Load game list you will get to the game screen. Here you will see a lot of things. You will see your character (the player) that you can control and run around with. The background is the world that you run around in. You will also be able to interact with enemies, pick up items and so on. In the left upper corner you will be able to see information like how many lives you have, how many coins you have and so on. In the left lower corner you will have two item slots were you can put two items of your choosing to easier and faster use those two items. It should be possible later to add animations.

If you click on the “Continue” or “New game” button



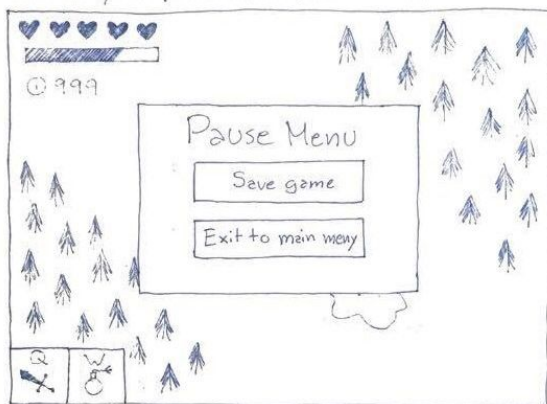
When you press on the inventory button an inventory bar will show up next to the item slots. Here you will be able to see all the items you have and change the items in the item slots from the inventory if you want other items in the item slots.

When you click on the inventory button the inventory bar will show.



If you click on the pause button the ingame menu show up where you can either save your game or exit to the main menu. If you exit to the main menu the first menu will show up and from there you can exit the game.

When you press the pause button



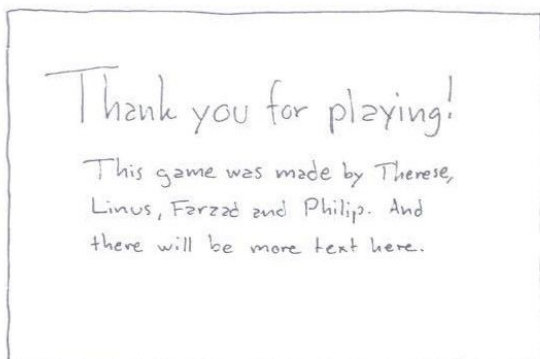
(Allt bakom pause menu kommer mörkuz en zning.)

If you would lose the game (if your health drop to 0) a screen with the text "Game over" will show.

When you lose the game

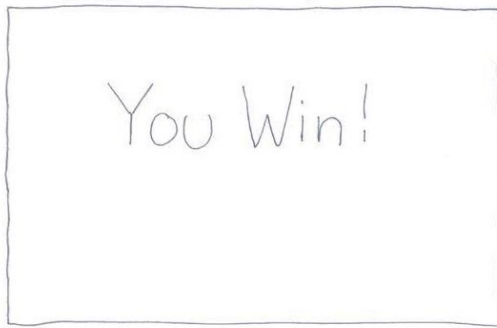


Then a screen with the credits will show.



If you instead win the game the win screen below will show and then the credit screen above.

When you win the game



## 2.2 Functional requirements

- From the menu the user has the ability to...
  - Start the game
  - Top score list
  - Learn how to play the game
  - Change controller or button/key mapping
  - Quit the application
- While inside the game world the user can...
  - Move about freely in the world
  - Pick up items found in the world
  - Transition from one area to another. From the “outside” world into a dungeon, for example
  - Attack AI controlled characters and slay these to acquire rewards
  - Open an inventory screen from which users can...
    - Manage items in inventory by manipulating them
    - Consume items (e.g bombs)
    - Equip items

### Ordering of use cases by priority:

- Start Application
- Start Game
- Exit
- Win game
- Lose game
- Attack
- Damage
- Death
- Item activation
- Inventory Management
- Move
- Item pickup
- How to play
- Settings

## 2.3 Non-functional requirements

### Usability

The game is designed in such a manner that the user should without any trouble immediately be able to get going in the game with only minor descriptions of how to play the game.

### Reliability

There are no critical reliability requirements as it's not a critical system.

### Performance

This is a game after all, which means that there are certain real-time requirements when it comes to rendering and equivalent. This should however not be a problem as this application isn't as complex, and only uses 2D textures.

The user will be most comfortable playing at a stable 60 fps or higher. Therefore, the application will most likely be limited to 60 fps.

### Testability

The underlying game logic should be testable without an accompanying graphical interface.

### Technical

The application must be implemented in a manner such that it is easy to scale, further extend its functionalities, and implement comprehensive features such as a client/server-architecture.

Platform independency is achieved using JRE. To run the application, the user will need to have JRE installed.

For usability sake, it's best to provide a PE (for Windows users) to the user (jar wrapped into PE). This way, we hide information not needed for the user which yields the feeling of opening/executing any other program that they're used to. Same goes for any other data that might confuse the user, such as any library contained in for example dlls.

It's to prefer for sake of usability that an installer is provided which unpackages everything and sets it up nice and tidy for the user. However, it will most likely be an app that you get as-is.

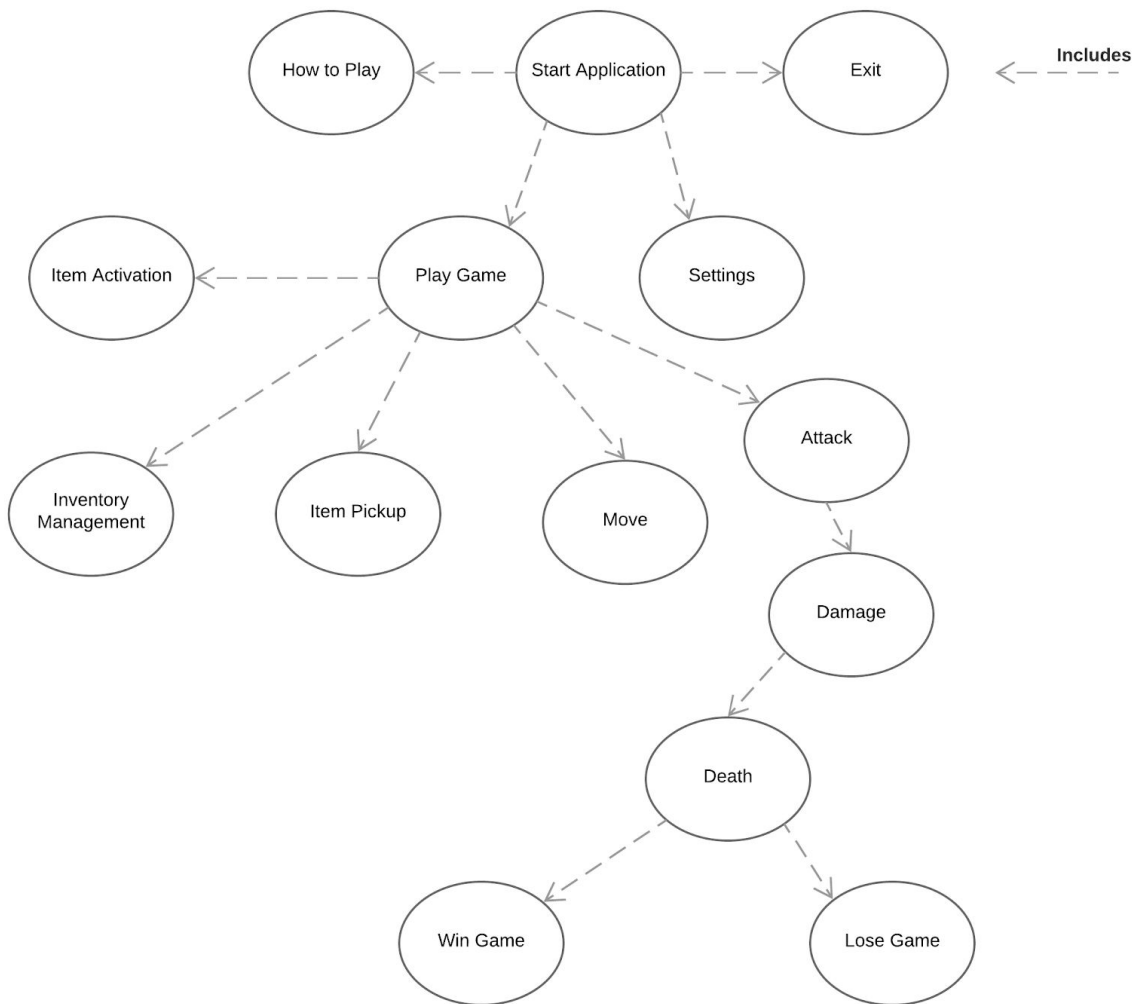
The source code will not be included.

List of what's included in the package:

1. A file that contains the application code (jar file(s) wrapped into a PE)
2. All needed resources (assets, map, external libraries, etc.)

3. A README-file containing short documentation and general information regarding how you execute, configure, and install the application

## 3 Use cases





## 3.1 Use case listing

### UC: 1. Move

Summary: A character moves from one tile in the game world to another

Priority: Mid

Extends: None

Includes: None

Participants: Character

#### Normal flow

Flow 1: Normal movement, move into an empty tile

	Character	System
1.	Character tries to move into another tile	
2.		Character moves one tile in the chosen direction

#### Alternate flow

Flow 2: Movement into impassable tile

	Character	System
1.	Character tries to move into an impassable tile	
2.		Position remains unchanged

Flow 3: Movement into tile occupied by an item

	Character	System
1.	Character tries to move into a tile occupied by an item	
2.		Move Character into tile
3.		Ref. UC: 5. Item pickup

#### Flow 4: Movement into tile occupied by a character

	Character	System
1.	Character tries to move into a tile occupied by another character	
2.		Position remains unchanged

#### Flow 5: Player Character moves between segments

	Player Character	System
1.	Character tries to move beyond the edge of a segment	
2.		Character is transported into a new segment

## UC: 2. Attack

Summary: A Character performs an aggressive action towards another Character

Priority: High

Extends: None

Includes: Damage

Participants: Character

### Normal flow

#### Flow 1: Character attacks a target and hits

	Character	System
1.	Attacks	
2.		Checks if the attack hits the target
3.		Target is hit
4.		Damage is calculated, refer to UC: 3. Damage

## Alternate flow

Flow 2: Character attacks but misses or hits nothing

	Character	System
1.	Attacks	
2.		Checks if the attack hits the target
3.		Attack misses

## UC: 3. Damage

Summary: A source deals damage to a character, damage dealt is deducted from characters hit points

Priority: High

Extends: None

Includes: Death

Participants: Character

## Normal flow

Flow 1: Character takes damage and does not die

	Character	System
1.	Character takes damage from any source	
2.		Character's health decreases by an amount related to the source's attack
3.		Character enters a brief period of invulnerability

## Alternate flow

Flow 2: Character takes damage and dies

	Character	System
1	Character takes damage from any source	
2		Character's health decreases by an amount related to the source's attack

3		Health becomes 0, Character dies, move to UC 4. Death
---	--	---

## UC: 4. Death

Summary: Character death when damage dealt is greater than or equal to current HP

Priority: High

Extends: None

Includes: Lose game, Win game

Participants: Character

### Normal flow

#### Flow 1: AI Character dies

	AI Character	System
1	AI Character has met the requirements for death	
2		Character is removed from World
3		Character drops an item from an item drop table

### Alternate flow

#### Flow 2: AI Character Boss dies

	AI Character Boss	System
1	AI Character Boss has met the requirements for death	
4		Player is presented with Win game, refer to UC: 13. Game won

#### Flow 3: Player Character dies

	Player Character	System
1	Player Character has met the requirements for death	
2		Player is presented with Game Over, refer to UC: 14. Lose Game

## UC: 5. Item Pickup

Summary: A Character picks up an item in the game world

Priority: Mid

Extends: None

Includes: None

Participants: Player, AI

### Normal flow

#### Flow 1: Player Character picks up consumable item

	Player Character	System
1	Player Character stands on top of a consumable item	
2		Check so that the Player hasn't reached the item limit and that the player has enough inventory space for this item
3		Remove item from the world and increment the amount in the Player Character's inventory

### Alternate flow

#### Flow 2: Player Character picks up consumable item, but does not have enough room

	Player Character	System
1	Player Character stands on top of a consumable item	
2		Check so that the Player hasn't reached the item limit and that the player has enough inventory space for this item
3		Player does not have room in inventory, remove item from the world, inventory remains unchanged.

### Flow 3: Player Character picks up direct consumable item

	Player Character	System
1	Player Character stands on top of a direct consumable item	
2		Item's effect is immediately applied to character

### Flow 4: Player Character picks up permanent item

	Player Character	System
1	Player Character stands on top of a permanent item	
2		Check so that Item isn't in the inventory already
3		Remove Item from World and place into Player Character's inventory

### Flow 5: AI Character tries to pick up an item

	AI	System
1	AI Character stands on top of a any item	
3		Item state remains unchanged

## UC: 6. Item Activation

Summary: Player interacts with an item, e.g uses an equipped item.

Priority: High

Extends: None

Includes: None

Participants: Player

### Normal flow

#### Flow 1: Player activates an equipped permanent item

	Player	System
--	--------	--------

1.	Player presses the item's key	
2.		The item's effect activates

## Alternate flow

### Flow 2: Player activates an equipped consumable item

	Player	System
1.	Player presses the item's key	
2.		The item's count is decremented and the item's effect activates

## UC: 7. Inventory management

Summary: Managing your inventory. This includes things such as equipping items.

Priority: High

Extends: None

Includes: None

Participants: Player

## Normal flow

### Flow 1: Player opens/closes the inventory

	Player	System
1.	Presses the inventory key	
2.		The inventory hotbar appears/disappears

## Alternate flow

### Flow 2: Player equips an item in an empty equipment slot

	Player	System
1.	Player presses the directional keys	
2.		The inventory cursor changes position between items
3.	Player presses a key corresponding to an equipment slot	

4.		The item selected by the cursor is put into the corresponding equipment slot
----	--	--

Flow 3: Player equips an item in the same equipment slot as another equipped item

	Player	System
1.	Player presses the directional keys	
2.		The inventory cursor changes position between items
3.	Player presses a key corresponding to an equipment slot	
4.		The item equipped in the slot is replaced with the one selected by the cursor and is put back into the inventory

## UC: 8. Start Application

Summary: A start menu where you can choose to start the game, read about it, and quit the game.

Priority: High

Extends: None

Includes: Start Game, How to Play, Settings, Exit

Participants: Player

### Normal flow of events

Flow 1: Shows the Start Menu

	Player	System
1.	Starts the application	
2.		Shows the start menu screen

## UC: 9. Play Game

Summary: Starts the main game from the Start Menu

Priority: High

Extends: None

Includes: Move, Attack, Item Pickup, Item Activation, Inventory Management

Participants: Player



## Normal flow of events

### Flow 1: Player starts the game

	Player	System
1.	Clicks on the “Start Game” button	
2.		Start menu screen is replaced with the game screen

## UC: 10. How to Play

Summary: Accesses the “How to Play” menu from the Start Menu

Priority: High

Extends: None

Includes: None

Participants: Player

## Normal flow of events

### Flow 1: Player opens and closes the How to Play menu

	Player	System
1.	Click on the “How to play” button	
2.		Start menu screen is replaced with the how to play screen
3.	Click on the “Go back” button	
4.		Player is returned to the main menu

## UC: 11. Settings

Summary: Accesses the Settings from the Start Menu

Priority: High

Extends: None

Includes: None

Participants: Player

## Normal flow of events

### Flow 1: Opening and closing the “Settings” menu

	Player	System
1.	Click on the “Settings” button	
2.		Start menu screen is replaced with the settings screen
3.	Click on the “Go Back” button	
4.		Player is returned to the main menu

## UC: 12. Exit

Summary: Close the application

Priority: High

Extends: None

Includes: None

Participants: Player

## Normal flow of events

### Flow 1: Exit

	Player	System
1.	Click on the “Quit” button	
2.		The game stops and the window disappear

## UC: 13. Win game

Summary: If the player character wins the game

Priority: High

Extends: None

Includes: Death

Participators: Player character

## Normal flow of events

	Character	System
--	-----------	--------

1	The player character kills the boss (enemy)	
2		The game shows the win screen
3		The game shows the credit screen
4		The game show the main menu screen

## UC: 14. Lose Game

Summary: If the player character loses the game

Priority: High

Extends: None

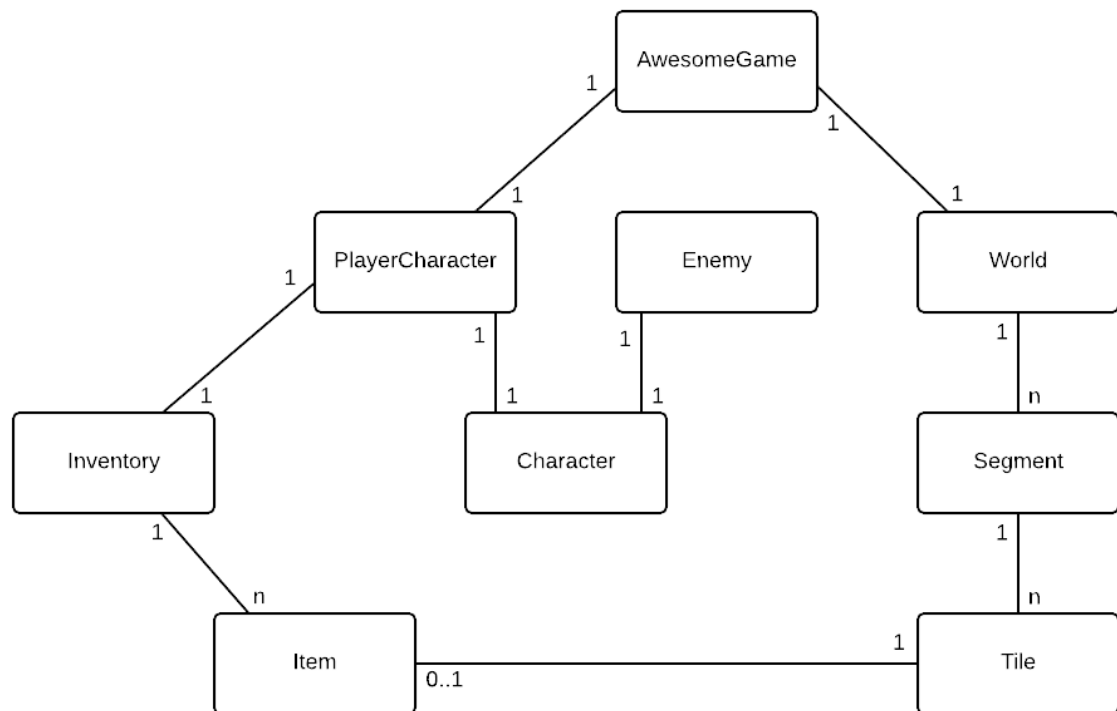
Includes: Death

Participators: Player character

### Normal flow of events

	Character	System
1	The player character have died	
2		The game shows the game over screen
3		The game shows the credit screen
4		The game show the main menu screen

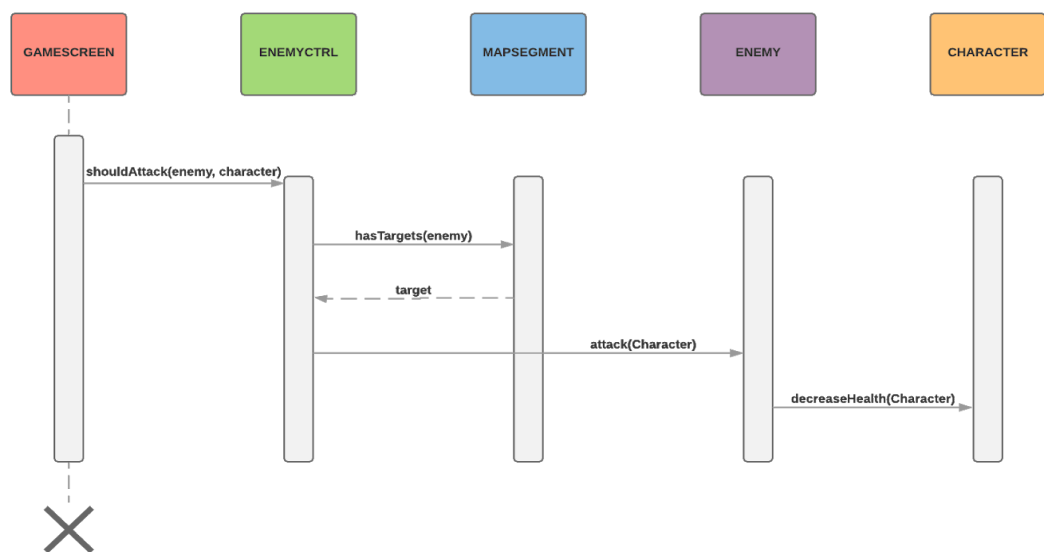
## 4 Domain model



### 4.1 Sequence diagram

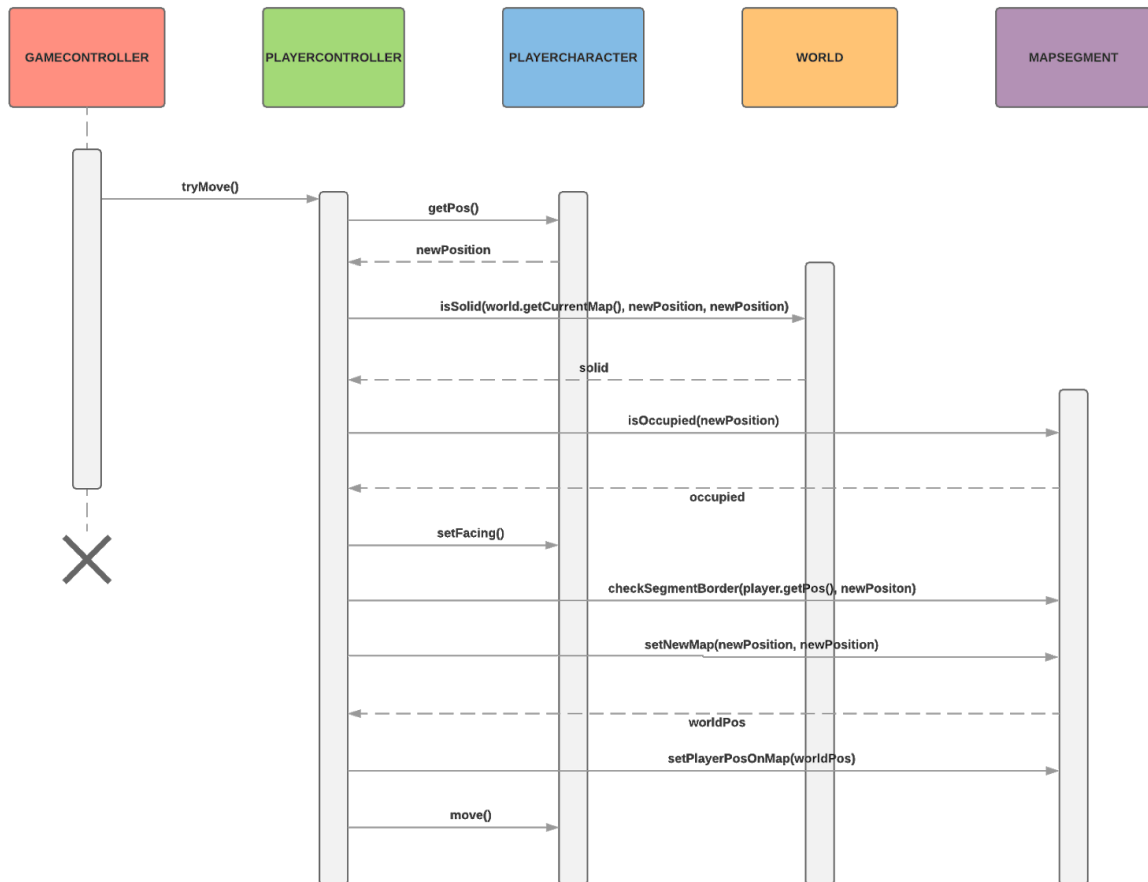
#### SEQUENCE DIAGRAM FOR ENEMY'S ATTACK

Therese | May 23, 2018



## SEQUENCE DIAGRAM FOR PLAYERS MOVE

Therese, Linus, Philip | May 23, 2018



## 5 References

[Portable Executable](#)