SWEN1 – Lab02

Use case with formality type “brief” according to C. Larman

Edit a new or existing map: The player wishes to edit a newly created or an already existing map of his. By visiting the “Maps” section from the main menu he is presented with a full list of all available maps. Each element of the list will contain a button to edit the respective map. When the player clicks the button of the map he wants to change, the data file is automatically opened in MS Excel (or whatever standard program is set by the user) and he can edit it.

Use case with formality type “casual” according to C. Larman

Load an existing map:
Main success scenario: The player finds himself in the main menu and wishes to start a new game on an existing map. The player uses the button to begin a new game and is then prompted with a list of all available maps to play on. He or she chooses their preferred level and confirms their choice with a click on the element in the list. The program loads the respective map file, renders it and begins a new game automatically.

Alternate scenario:

If the file is malformed and a map cannot be rendered from it, the program remains in the map list and displays an error message reading “The map file is corrupted and cannot be loaded.”.

Use case with formality type “fully dressed” according to C. Larman

Use Case UC1: Play game

Primary Actor: Player
Stakeholders and Interests:

- Player: Wants a stable framerate with short load times and no crashes to interrupt his experience.

Preconditions: The player has selected a map and started a new game on it.
Post conditions: The player has either defeated all enemy waves and won or his central structure has taken a critical amount of damage and has been destroyed resulting in the player losing the game.

Main success scenario:

1. The player has started a new game and the map is loaded.
2. The player spends his starting currency on building up his defences.
3. The player clicks on begin, indicating they are done preparing and ready for the first enemy wave.

4. Incoming enemies are destroyed by the defensive structures and the player spends the money gained on new defences.

Step four repeats itself so long as there are enemies remaining in the current wave and the main structure has not been destroyed.

5. When all enemies of the current wave have been destroyed, there is an indication that the next wave will be incoming soon.
6. The player has a set amount of time to improve his fortifications before the next wave begins automatically.

Steps four through six are repeated while the last wave has not been defeated and the main structure has not been destroyed.

7. The player has defeated the last wave and a message is displayed indicating that they have won the game.
8. The game automatically returns to the main menu after the message has disappeared.

Extensions:

*a. At any time, the game crashes:

The game shuts down and the player must restart the game if he wishes to continue playing. Any game progress will not be saved.

*b. The player closes the game window:

The current game ends and no game progress is saved.

*c. The Player pauses the game:

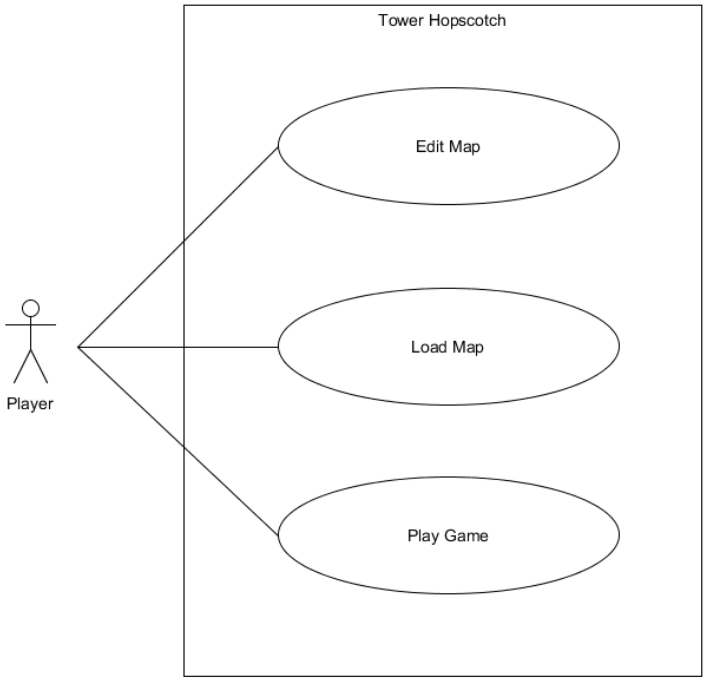
The ongoing game is paused and a menu is brought up allowing the player to leave to leave the game or to resume it.

Special Requirements:

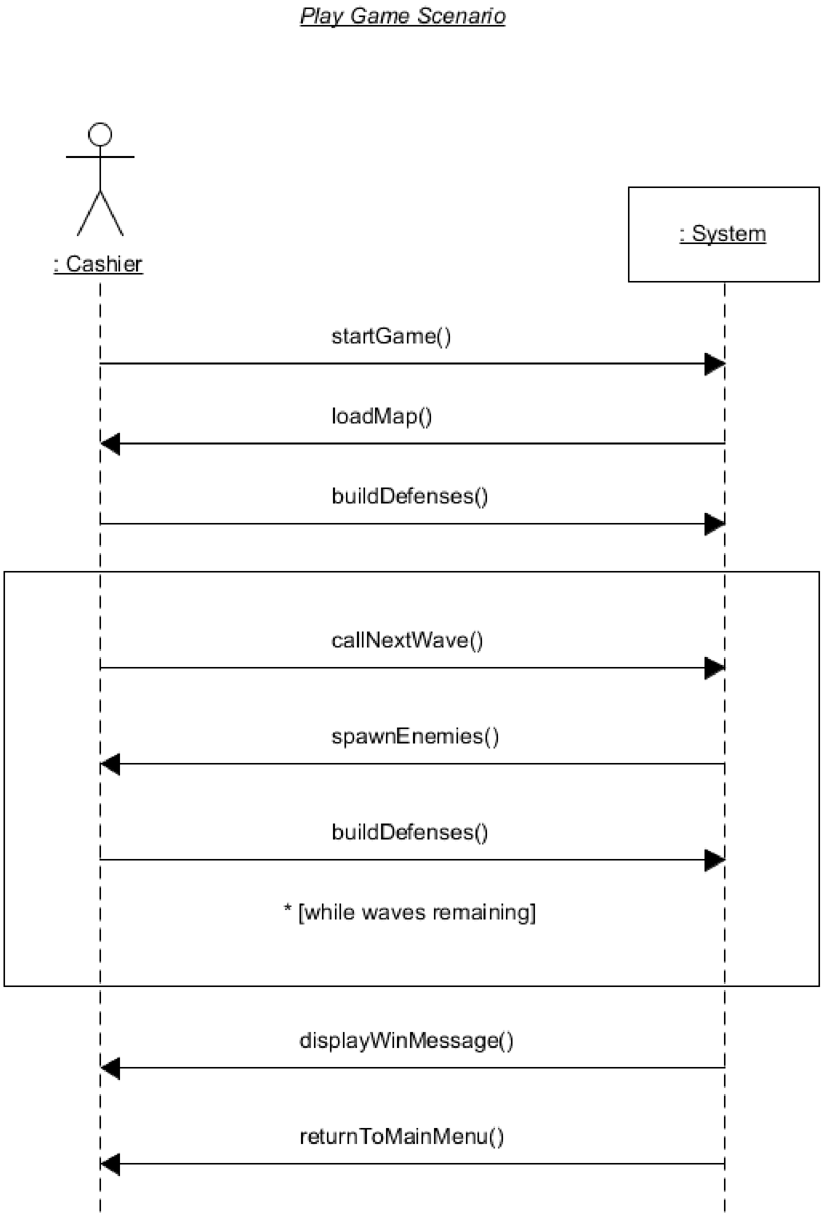
- Windows or Mac computer with Java 8.
- Computer with mouse and keyboard or a touch display.

Frequency of Occurrence: Initiated by player.

Use Case Diagram



Sequence Diagram "Play Game"


Play Game Scenario
: Cashier
: System
startGame()
loadMap()
buildDefenses()
callNextWave()
spawnEnemies()
buildDefenses()
* [while waves remaining]
displayWinMessage()
returnToMainMenu()