Maze Game Design Document

QA Software Development Synoptic Project

Chris Evans

5829035234

Contents

[Introduction 1](#_Toc40432106)

[Purpose 1](#_Toc40432107)

[Requirements 1](#_Toc40432108)

[Design 1](#_Toc40432109)

[Components 2](#_Toc40432110)

[The Player 2](#_Toc40432111)

[The Maze 2](#_Toc40432112)

[Entities 3](#_Toc40432113)

[Class Diagram 4](#_Toc40432114)

[Maze Format Specification 5](#_Toc40432115)

[Definition 5](#_Toc40432116)

[Root Object (Maze) 5](#_Toc40432117)

[Maze Room 5](#_Toc40432118)

[Exits 5](#_Toc40432119)

[Entity 6](#_Toc40432120)

[Enumerations 6](#_Toc40432121)

# Introduction

Olde World Phunne is a game developing company, creating games inspired by the past for a variety of platforms. They wish to provide a free maze game to increase traffic to their website.

## Purpose

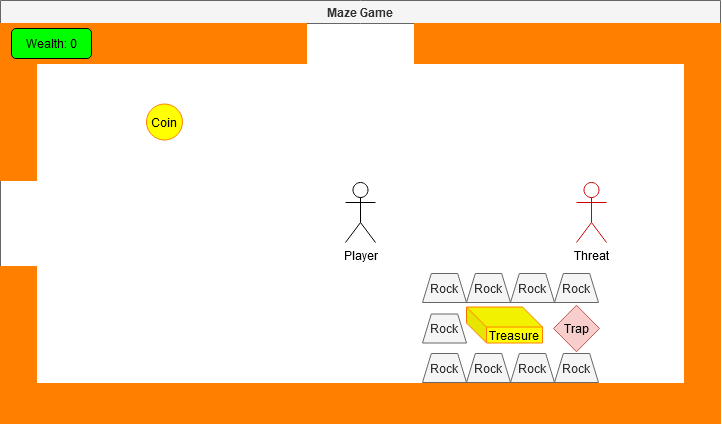
The purpose of this document is to outline a design for a maze game as detailed by Olde World Phunne, including the specification of the format to generate mazes.

# Requirements

* The maze must have a set number of multiple rooms
  + Each room will have between 1 and 4 passages
  + At least 1 passage will be an exit to the room
* Rooms are cleared by eliminating all threats
* Rooms contain collectable treasure which add to the players wealth
  + Coins can be dropped as “markers” which subtract 1 from the players wealth
* 1 Room will contain the exit passage, which will win the game
* Each turn a player performs an “Action” which may include
  + Moving (North, East, South, West)
  + Interacting with entities such as
    - Attacking enemies
    - Disarming traps
  + Dropping Coins

# Design

This is what a typical maze room might look like:



This room contains:

* The Player
* A Threat that needs to be defeated
* A coin dropped by the player
* The player’s wealth in the top-right
* Static Rock objects
  + Statics cannot be interacted with, moved or passed through
* Treasure
* A trap

## Components

The core of the maze game can be thought of in 3 different components:

* The Player
* The Maze
* Entities within the maze

### The Player

The player will be the most rich component as they are what the user playing the game controls. They need to be able to traverse the maze, interact with entities and keep track of wealth accumulated.

### The Maze

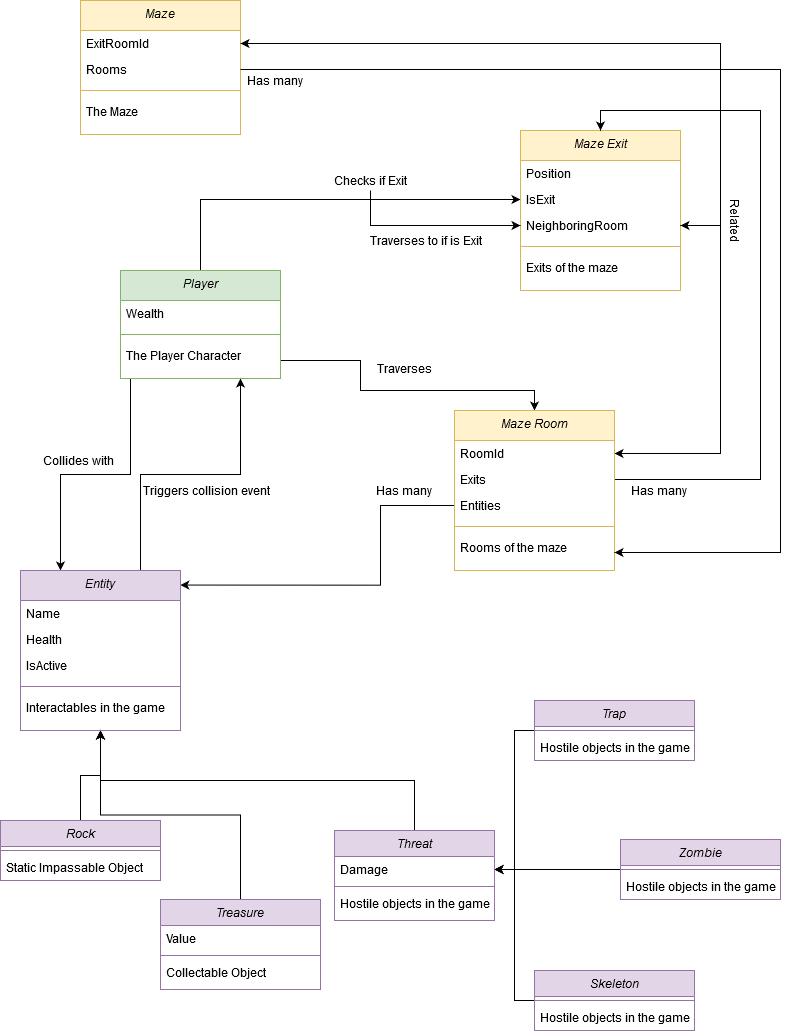
Information of the maze needs to be stored before and throughout the gameplay including:

* The number of rooms
* The positions and states of each entity in each individual room
* Relationships between rooms
* The exit that wins the game

### Entities

I define “entities” as things within the maze that can be interacted with or change depending on the room. For example, if a player interacts with treasure, we want to increase the player’s wealth, but if they interact with a trap, we want something else to happen such as decreasing the player’s wealth or ending the game with a “Game over”.

### Class Diagram



# Maze Format Specification

The maze configuration file is a JSON file named “maze.json”. It is defined based on the classes detailed on the “Components” section.

## Definition

### Root Object (Maze)

|  |  |  |  |
| --- | --- | --- | --- |
| **Fields** | **Data Type** | **Description** | **Requirements** |
| ExitRoomId | Integer 32 | Id of the “Exit” to win the game. | Required, should be number of rooms |
| Rooms | MazeRoom Object Array | Objects defining the individual rooms of the game | Needs at least one room |

### Maze Room

|  |  |  |  |
| --- | --- | --- | --- |
| **Fields** | **Data Type** | **Description** | **Requirements** |
| RoomId | Integer 32 | Unique ID of the room | Required |
| Exits | Exit Object Array | Objects defining each exit of the room | At least 1 is required with the “IsExit” field marked as true and  "NeighbouringRoom" must have a value |
| Entities | Entity Object Array | Objects defining the entities of each room | Not required, recommend to leave blank for room with ID “0” |

### Exits

|  |  |  |  |
| --- | --- | --- | --- |
| **Fields** | **Data Type** | **Description** | **Requirements** |
| Position | “Exit Position” enumeration | Describes position of exit (North, East, South, West) | Required |
| IsExit | Boolean | Describes whether or not the room is an exit | Required |
| NeighbouringRoom | Nullable Integer 32 | Room the exit leads to | Required if “IsExit” is true, otherwise can be null |

### Entity

|  |  |  |  |
| --- | --- | --- | --- |
| **Fields** | **Data Type** | **Description** | **Requirements** |
| Type | “Entity Type” enumeration | Describes the entity type | Required |
| Health | Integer 32 | Health points of the entity | Required |
| Position | Object with “x” and “y” floating point fields to the nearest 0.5 | Position of the entity | Required |
| Active | Boolean | Entity is active or not | Required |

### Enumerations

#### Exit Position

|  |  |
| --- | --- |
| **Index** | **Value** |
| 0 | North |
| 1 | East |
| 2 | South |
| 3 | West |

#### Entity Type

|  |  |
| --- | --- |
| **Index** | **Value** |
| 0 | Rock |
| 1 | Coin |
| 2 | Treasure |
| 3 | Trap |
| 4 | Zombie |
| 5 | Skeleton |