**Game Design Document for:**

# Name of Game

**One Liner, i.e. The Ultimate Racing Game**

All work Copyright ©2022 by Woolford Studios Ltd

Version # 1.00

Tuesday, March 01, 2022**Table of Contents**

Name of Game 1

Foreword 4

Project Management 5

Version 1.10 5

Version 2.00 5

Version 2.10 5

Health and Safety in the workplace 6

Copyright Laws 6

Game Overview 7

Philosophy/ theme/ morals 7

Philosophical point #1 7

Philosophical point #2 7

Common Questions 7

What is the game? 7

Who is the target audience? 7

What themes are present in the game? 7

Where does the game take place? 7

What/who do I control? 7

How many characters do I control? 8

What is the main focus? 8

What is different? 8

Feature Set 9

General Features 9

Gameplay 9

The Game World 10

Overview 10

World Feature #1 10

World Feature #2 10

The Physical World 10

Overview 10

Key Locations 10

Objects 10

The World Layout 11

Overview 11

World Layout Detail #1 11

World Layout Detail #2 11

Critical Reflection 16

Extra Miscellaneous Stuff 17

Overview 17

Junk I am working on… 17

“XYZ Appendix” 18

“Objects Appendix” 18

“References/bibliography” 18

“Images (flowcharts, mind-maps, diagrams etc)” 18

“World map” 18

“Classroom notes” 18

“Playable demo” 18

# Foreword

Use flowcharts, tables and other practical means to set out your schedule and other project management needs.

Use the **project management** hub below to include:

* Scheduling
* Deadlines
* Milestones
* Time management

Use the appendix to add classroom notes, references and images used.

**Delete this section before hand-in!**

# Project Management

## Version 1.00

Version 1.00 includes some tuning and tweaking that I did after making my initial pass at the design. Here is what I changed.

1. Idea for game found.
2. Document is created.
3. Necessary headings are added.

## Version 1.10

Version 1.10 includes…

1. Idea for game found.
2. Started writing in the “Game Overview”

## Version 2.00

Version 2.00 includes…

1. Finished the game overview
2. Added text to “Feature Set”
3. Added text to “The Game World”
4. Added text to “The Physical World”
5. Added text to “The World Layout”

## Version 3.00

Version 3.00 includes…

1. Added “Source Control” and images
2. Added “Black Box Testing” overview

# Game Overview

## Common Questions

### What is the game?

This game will be a roguelite themed around the ASCII characters, The idea came from the rules required for a game to be a roguelike that they must have an ASCII display, the spin on this is that the game will physically uses the symbols in the gameplay.

## Inspirations

The game takes inspiration for other popular ‘roguelikes’ such as The Binding of Isaac and Hades which both have different ‘stages’ split into small randomly generated rooms and an overarching goal of defeating enemies and getting far as possible before dying and starting over.

### Who is the target audience?

The targeted audience is for anyone who likes to play roguelike-likes and roguelites which are similar to roguelikes but don’t entirely qualify to be whether that be because of not having turn-based combat or true perma-death.

### Where does the game take place?

The game will take place inside of a magical dungeon where words are given physical form to create paths in the mysterious dungeon.

The dungeon will be randomly generated or ‘shuffled’ every time the player starts a game and will have layered floors with different themes.

### What/who do I control?

The player will control a sole character who is just trying to escape, if the player dies, then they will get sent back to start over retaining none of the collected loot.

### What is the main focus?

The main goal of the game is to escape from the dungeon while collecting items and gear to aid the adventure, when the player dies, they lose everything and are thrown back to the start where all the rooms have changed.

After they escape once, the main goal becomes to beat the high score set by previous runs.

### What is different?

In this roguelite there are missing pieces of the map, so you can use gold gathered in the dungeon to buy map pieces which correlate to their ASCII value, if I can finish this, I can add other purchasable pieces that change other aspects of the game.

A list of ASCII characters and their values:

|  |  |  |  |
| --- | --- | --- | --- |
| **Character** | **Cost** | **Character** | **Cost** |
| ╣ | 185G | ╩ | 202G |
| ║ | 186G | ╦ | 203G |
| ╗ | 187G | ╠ | 204G |
| ╝ | 188G | ═ | 205G |
| ╚ | 200G | ╬ | 206G |
| ╔ | 201G |  |  |

Furthermore, (aesthetically only) the enemies drop gold in the form of their ASCII value, so for example when an enemy dies, they drop a total of 100 gold which could be split down into a: \* (42G), & (38G), ♠ (6G), and two • (7Gx2) in an explosion of characters.

# Feature Set

## General Features

Made in unity

Embeds ASCII symbols into the gameplay

Randomly generated worlds

UI including health, items, and minimap

## Gameplay

Hack-n-slash combat

Perma-death

New map layout each time.

Can purchase and build missing paths

Collectable items and gear

# The Game World

## Overview

The game is set in dungeons that are everchanging, the player character has fallen down into the nethermost layer of the dungeon and must now escape using a new form of magic that uses ASCII symbols as its basis.

## ASCII

As you most likely have realized so far, this game heavily uses ASCII in the gameplay compared to roguelikes which use it just for displaying the game.

The player can use the symbols to bridge across previously uncrossable gaps by purchasing them from the local merchant for their equivalent ASCII value.

## World Feature #2

#########################################################

## The Physical World

### Overview

The player is placed within a randomly generated map of rooms joined by pathways, the game is top down 2D, so most of the time the player can see an entire room at once on their screen. Once the player has cleared all the rooms on a floor, the above floor is unlocked, and the player moves closer to escaping.

### Key Locations

Nethermost Region – As the same suggest it’s the lowest floor, supposed to seem mysterious.

The library – the next floor up, player can learn better attacks here using the symbols.

???? - ######################

#### - ??????????????????????

Ground layer – The end goal location to beat the game.

### Objects

The player uses gold as their currency which can be collected from defeating enemies.

But the more prominent roguelike feature is the collectable items which can have various effects from increased attack, defense, or extremely specific effects such as inflicting poison every 5th attack.

Finally, the player has difference choices of weapon which each have their own pros and cons.

See the “Objects Appendix” for a list of all the objects found in the dungeon

# The World Layout

## Overview

Since the game is ‘randomly generated’ there isn’t a permanent map, but what I can show is an example of what a floor layout could look like as well.

A picture containing text, music, clock, piano

Description automatically generated╬ - Paths, █ - Rooms, • - Missing paths.

One thing I’ve realized now, is that the symbols have a lot more height than width, so I’m conflicted between keeping it like this or matching the ratio.

Alternatively, I’ve realized that if you turn it onto its side, it closer resembles the aspect ratio of a computer screen.

MAP SHOWING LOCATIONS HERE##########################

## World Layout Detail #1

Each level is separated into several floors so that the player doesn’t complete the game after only traversing four or so, this also means the floors can build up to a boss.

## World Layout Detail #2

While the floors are stacked in terms of story, the player can’t actually see the upper/previous floors to avoid clutter, plus it means that unity doesn’t need to worry about keeping past floors and can’t generate the layout when the player first enters.

# Source Control

## Overview

Source control is important for allowing all developers of a team access and edit files over a large geological distance, files can be sorted and have a history of when they were edited.

## Using GitHub

GitHub is a commonly used source control software used primarily for code related projects.

A very basic tutorial to using GitHub:

Graphical user interface, text, application

Description automatically generated

Create a new repository.

Graphical user interface, text, application

Description automatically generated

Add files to the folder.

Graphical user interface, text, application, chat or text message

Description automatically generated

When you go back into GitHub desktop, the “Changes” tab will state the number of changes to documents, you can write about what changed in the dialogue boxes and then “Commit”.

Graphical user interface, application

Description automatically generated

After committing, GitHub will show all the files in the history tab.

Graphical user interface, application

Description automatically generated

Above you can find the “Publish repository” tab which allows you to publish it onto the GitHub website.

Graphical user interface, text, application, email

Description automatically generated

The github website will now show the files which can now be accessed by everyone working on the project.

Graphical user interface, text, application, email

Description automatically generated

Click on the name of the file you want to download, sometimes the file will have a download box, but in the case of a text file it works a bit differently, so next click on the “Raw” tab.

Graphical user interface, text, application, email

Description automatically generated

GitHub will open up a new tab with the contents of the text file, here you can now save the file.

# Black Box Testing

## Overview

Black box testing tests a function such as a game mechanic without the user knowing what is going to make that happen, so a user will input something and an output is received, if the output doesn’t seem right it may need to be tweaked.

This tests how good the function is at taking in unexpected inputs which the designer may not have thought about.

The opposite of black box testing is white box testing, which is the same concept, but the user/designer knows and can see what happens internally.

## Examples

EXAMPLES FROM MY PROJECT HERE

# Production Log

Overview

* + Technical; tools and features
  + Programming fundamentals (variables, if statements, loops, arrays, functions, and classes)
  + Gameplay loop
  + Mechanics

# Extra Miscellaneous Stuff

## Overview

Anything else that is relevant that you may have missed or isn’t essential… maybe the game world has a monetary system and economy that you are dying to add but it doesn’t fit anywhere else.

# “XYZ Appendix”

Provide a brief description of what this appendix is for and then get down to business and provide data to the reader.

Here are a few examples of some of the appendices in my latest design…

# “Objects Appendix”

# “References/bibliography”

[ASCII code - (theasciicode.com.ar)](https://theasciicode.com.ar/extended-ascii-code/box-drawings-double-line-horizontal-vertical-character-ascii-code-206.html)

[Berlin Interpretation - RogueBasin](http://www.roguebasin.com/index.php?title=Berlin_Interpretation)

# “Images (flowcharts, mind-maps, diagrams etc)”

# “World map”

# “Playable demo”

# “Critical Reflection”