# GemLabyrinth

## Introduction

GemLabyrinth is an Unreal Project (currently 5.1) which provides a platform for student assessment in Unreal on KF4010 Introduction to Game Development. It has a meta-world which will contain many portals to minigames, each of which is a student-developed assignment. The meta-game includes two basic concepts: **lives** and **gems.**

## Game Overview

The player starts with 3 lives. They enter minigames (created by students) in order to gain gems. Each minigame should have 4 possible outcomes:

1. Fail – the player loses a life and is taken back to the meta-world
2. Quit – the player choses to bail out and is taken back to the meta-world with no consequences.
3. Success – the player gains a gem and is taken back to the meta-world. The minigame should now be locked against further attempts.
4. Super-success – the player succeeds with extra criteria, e.g. within a fast time, collecting an extra hard-to-reach artefact, and so on. The player gains a gem and a life and is taken back to the meta-world. The minigame should now be locked against further attempts.

## File Structure

A template will be given to students which contains the meta-world with only one mini-game portal, and the bare bones of the mini-game map. The meta-world assets are in the GemLabyrinth folder, and the minigames are in the Minigames folder.

Graphical user interface

Description automatically generated with medium confidence

Inside the Minigames folder is a minigame template:

Graphical user interface, text, application

Description automatically generated

This contains an almost empty level an currently a floor asset.

Graphical user interface, application, website

Description automatically generated

The procedure for students is as follows:

1. Copy the whole template project
2. Rename the “RenameToYourID” folder to their student id
3. Rename the “RenameMe” level to something unique. This name will be used to connect the meta-game portal to the minigame.
4. Create the minigame of their dreams in that level
5. Submit just the minigame folder
6. All submissions are then collected into the Minigames folder for the final showcase
7. Each minigame will need a portal in the meta-world which links to it. This is easily done (see section below on portals)

## Continuity between the meta-world and the minigames

Continuity is achieved using the glGameInstance blueprint. This is one reason why the students are required to use this project template – it is already plugged in correctly. The glGameInstance currently has the following persistent variables:

* Gems (integer, initialised at 0)
* Lives (integer, initialised at 3)
* Metaworld (Name, used to go back to the metaworld)
* Showlives, Showgems, Showtimer (bools, used by the UI to determine what is shown and what is not)
* InStudentWorld (bool) so status can be queried
* UserBool1, UserBool2, UserInt1, UserInt2 – variables the students can use if they need them.

The timer variable is not yet implemented.

There are also the following functions:

* Fail
* Quit
* Success
* Super-success – use these 4 to register a result and go back to the meta-world
* GoMeta – utility to jump back to the meta-world
* ResetUserVariables – resets them to 0s and falses
* GoStudentWorld – used to go from meta-world to a minigame
* ToggleCheat (=)
* CheatFail (0)
* CheatQuit (1)
* CheatSuccess (2)
* CheatSuper (4) – these last 5 are part of the cheat system. They are also keymapped as given. The last 4 only work if cheat mode is enabled, and can be used in the meta-world also to increase gems and increase or decrease lives.

Access to these variables and functions from the students’ functionality is pretty easy. Here is an example:

A screenshot of a video game

Description automatically generated

## Teleport Pads

There are 5 teleport pad blueprint assets created, most of which are for use in the minigames.

Graphical user interface, application

Description automatically generated

* glTeleport is for use in the meta-world. It takes a name parameter, and will teleport the player into that named minigame
* glQuitPlate, glSuccessPlate and glSuperPlate can be used in the minigames as an easy way to implement quit, success and super-success outcomes. They are coloured bronze, silver, and gold appropriately. It is recommended that all minigames have a quit plate near the player spawn in the level. These do not have to be used to get the outcomes.
* glTester is a utility version I have been using to trigger things I want to test. I am unsure at the moment whether to make it available for students.

## Functionality that is currently incomplete

* The timer is not yet implemented, this is intended to be an easy-to-use timer that students can use if they want to implement a time limit.
* The HUD functionality is only partially realised at the moment
* There is no consequence of getting to 0 lives (death)
* There could do to be meta-game menus
* The meta-world could do to be much nicer, more spacious and nice-looking, to accommodate all of the minigame portals.
* Transitions from meta-game to mini-game could have pretty effects and maybe sounds
* The teleport pads could be prettier, I think a sparkly particle effect over them would be nice
* It would be a good idea to try some varieties of minigame, to give the students ideas, and to iron out bugs.