Hanzaki

Technical Design Document

# Code Structure

**C#**

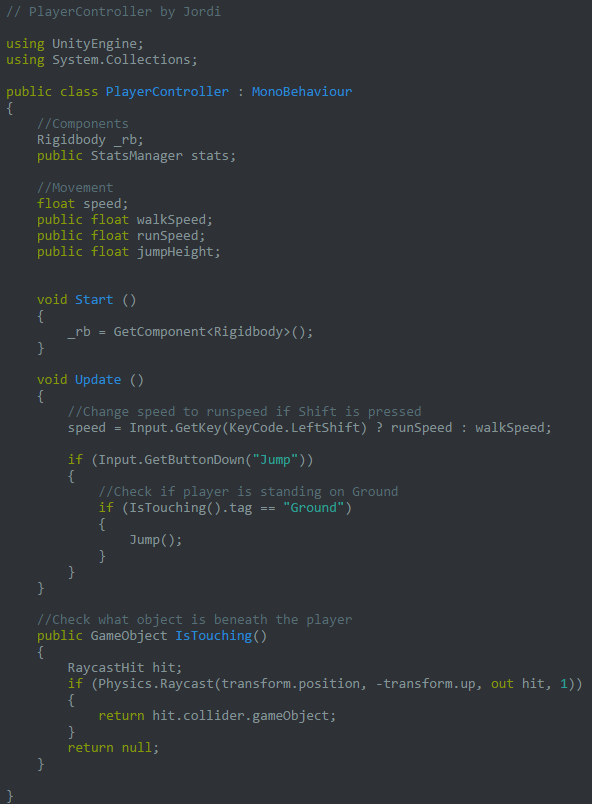
All scripts will be coded in Unity Script (C#)  
We chose this over Js to make optimal use of Unity Script and to create various classes.

**Classes**The script will be planned out completely before they will be added in the game. This will be displayed in a class diagram. The class diagram can be found in this document.

**Scripting Format rules**- Brackets on new lines.  
- Private Component Variables with \_ in front of variable name.  
- Correct use of private vs public variables.  
- Comments on at least all functions. Comments above the commented function.  
- Start a new script with //[Scriptname] by [Scripter] in the first line.  
- Correct usage of capital letters in variables / classes. As taught in the lessons.

**Script validation**All scripts will be checked by the lead developer.

Example Script



# 

# Class Diagram

**Main Camera**

**CameraController**  
The Main Camera will be controlled by the CameraController script. With lerping we will achieve smooth movement for the camera and make restrictions at edges of the level.

**(ImageEffects)**There will be various ImageEffect scripts attached to the Main Camera

**Player**

**PlayerController**Basic variables and functions that are used in both the Side-Scrolling and Top-Down sections will be in this script.

**PlayerSS**Unique Side-Scrolling variables and functions will be in this script. Inherits from PlayerController.  
  
**PlayerTD**Unique Top-Down variables and functions will be in this script. Inherits from PlayerController.

**GameManager**

The GameManager will have a DontDestroyOnLoad function. This object will be instantiated in the BuildScene and keeps existing throughout the game.

**StatsManager**All stats can be accessed from here

**OptionSettings**The options will be saved here.

**PlayerSpawner**When the player leaves a map, the PlayerSpawner will keep track of where he left so her re-enters on the correct location

**QuestManager**We keep track of all quests here and their statuses.

**PickUp**

**PickUpScript**PickUp Objects will have a script. This will function on collision enter with the Player.

**PickUpHeartsScript**Special script for the hearts, these will not be added to the inventory.

**Enemy**

**EnemyBehaviour**The enemy AI will be in this script.

**Weapons**  
All weapons will have their script. In each script the weapon can be controlled.

**Interactables**

**InteractScript**

InteractScript will function as a link between the object that is interactable and the ActionScript that will be activated. Handled by a Boolean to check if it’s currently activated.

**Canvas**

**CutsceneController**The cutscene controller will send text to the UIManager and will control the camera.

**UIManager**The UIManager will handle everything that happens on the HUD

**PauseManager**This manager will display the Pause and modify the timescale

**LocationOnEnter**When the player enters a new level, this script will play the animation that displays the name of that level

**Environment**

**DestructibleScript**Script for assets that can be destroyed. Will trigger a particle effect on hit by sword.

**LevelPortal**Placed at the beginning and end of levels to move between them.

**SteppingStones**Should sink when the player is on them.

**HookScript**This is the object that can be hooked if the player has the grappling hook. This object will light up if the player is close enough.

**Other**

**MainMenuManager**For handling the buttons in the main menu

**OptionsManager**The menu where you can change the options.

**CreditsManager**Scrolling text and a button to get back to main menu.

**QuestClass**Instances of quests with the attributes they have.

**LoadController**Shows the loadPanel while the game is loading.

**IntroCutscene**Plays a video and loads level1 when the video is finished.

# Unity Folder Structure

For every level we will have level-specific assets. These assets will have their folder in the **Art** folder of the project. In those folders are their texture, material, model, etc respectively. There will be subfolders in this folder for Level1, Level2, Level3 and shared.

Scripts will be in the **Scripts** folder.

Sound will be in the **Sound** folder, divided into three formats; BGM, SFX and AMB. These are the background music, the effects and ambience.

Prefabs will be stored in the **Prefabs** folder. These will also be in subfolders similar to the Art folder. So, level1, level2, etc.

All scenes will be stored in the **Scenes** folder. There is also a personal scenes folder for testing. Every member of the team must have their own scene for testing. It is not allowed to modify another members’ scene.

Sprites will have their own place in the **Sprites** folder.

Here is the complete list.

* Art
  + Level1
  + Level2
  + Level3
  + Shared
* Prefabs
  + Level1
  + Level2
  + Level3
  + Shared
* Scenes
  + Personal Scenes
* Scripts
* Sound
  + AMB
  + BGM
  + SFX
* Sprites

# Camera

The Main Camera is not going to be directly attached to the player, but linked through script. This way we can add a smooth lerping function for a more dynamic feel. We can also disable moving it when we are at the end of the level. We can also manipulate it to move the camera to interactables. Think of focusing the camera on an NPC when we are interacting with it.

# Controls and Abilities

Since we are using the keyboard only for the controls we will modify the input manager in Unity to add keys for Z, X, C, V and Left shift.

**Move**We use 3 PlayerController scripts. One general one, one for Side-Scrolling and one for Top-Down. The difference will be in the movement. In the Top-Down section the player should be able to move up by pressing the up-key but in the Side-Scrolling sections the player should jump instead.

**Slash**The player attacks with Z. There are 3 attack animations that will be played if the attack is used in succession.

**Jump**When the player jumps he can hold the button to jump higher. Similar to Mario.

**Shuriken**The player can throw shuriken for a powerful ranged attack. These are consumables that can run out.

**Grappling Hook Ability**For the grappling hook we are going to use a targeting system. If the character is close enough to a hook he can grab into the key will be enabled and the character can hook to that.

**Interact**If there’s a interactable object close to the player then the Z key can be used to interact. This will trigger the camera script to focus on the interacted object.

**Dash**By doubletapping a left or right arrow key fast we can use a dash to evade attacks. During this dash the player is invulnerable.

**SmokeBomb**The player can activate a smoke bomb if he has collected one. This will instantiate the smokebomb prefab that misdirects the aggro’d enemies.

# Enemies / Obstacles and Game Progression

There will be 3 kinds of enemies in the game which all have their own behavior and script. This is because they all have too many different attributies. In this case it is not really a profit to use a general EnemyClass script. We will have a QuestClass. Quests need to be completed to be able to advance to new levels. The quests will be stored in a database on the GameManager. This script won’t be destroyed on load so we can keep track of all quests’ statuses throughout the game.

# UI

The UIManager script will display the stats from the StatsManager on the screen. It is important to us that the player always has enough information about what to do. We will display quests in the top right corner. During cutscenes textboxes will appear. We will provide feedback for every action in the game by UI Graphics and Sound effects.

The Pause will be handled in another script that will freeze the time. Stop the background music. Make options to quit or go to the main menu and continue.

There will be different scripts for the credits menu, main menu and options menu.

On level enter the name of the level will be displayed at the top right of the screen.