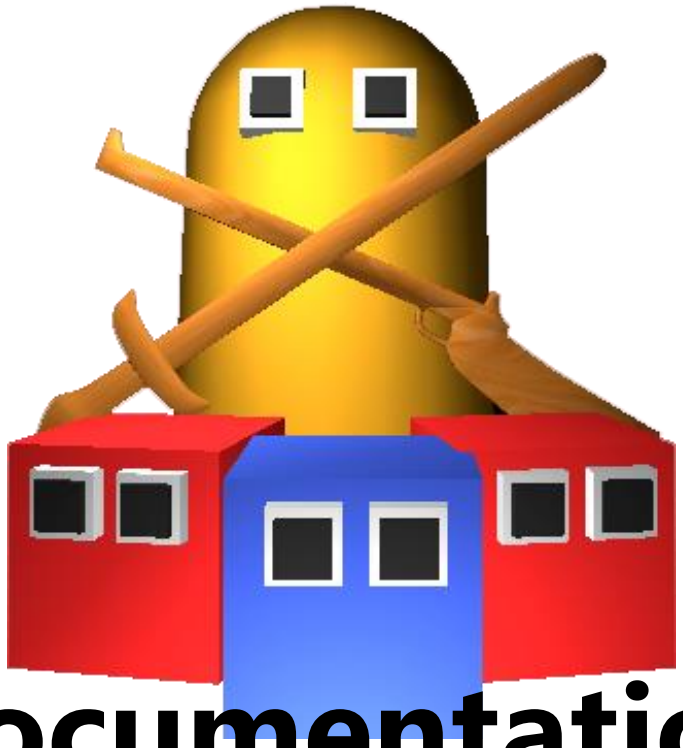

Documentation

By rubengc

Table of contents

Input Manager

This project needs some changes in input manager settings. First, in your Unity, go to Edit->Project Settings->Input and add this new assignements (You need change the size of the inputs assignements to 20):

LockCamera: for assign the key to lock/unlock the camera (I use the space bar)

Spell0, Spell1, Spell2 and **Spell3**: for assign the keys to cast spells (I use Q, W, E and R)

For assign the keys, you only need write in the "**Positive Button**" field the name/letter/number of the key in lowercase (space, q, w, e and r).

Global scripts

In the folder **GlobalScripts** you can found some scripts than you will use in all the scenes that you build with this kit.

Bullet: Need be added to a game object with a **collider**. This script is basically the golbal method to attack of anything object in the scene. It's detect when collides with another game object and try to detect if is an enemy or ally. If is an enemy, then call the method **ApplyDamage()** of his CharacterInformation script.

CharacterInformation: It's provide all the information about an interactuable object, such as health points, team, health and mana regenerations, etc also draws the HP and MP bars and replace the object by its replacement game object.

CleanObject: A little script that destroy a game object after some seconds.

GameManager : Its the responsible of respawn the players (with the method **Respawn()**) and maintains a constant communication with the minions spawner script for restore the dead minions (A small improvement to optimize the game performance was disabled and enable the minions rather than destroy them).

Player

The player need some elements to work correctly. First, his movement is based on follow an instantiate mark (See marks and pings section).

Next, his actions (move and attack) are defined in the **MOBACharacterController** script. If you move the player with attacks marks (right click), he will try to attack any enemy inside his vision range.

In this kit, you can found two different examples of players, one ranged and another melee.

Camera

The camera always follow his target (in all scenes is the player) when It isn't locked (you can lock and unlock with the **LockCamera** defined in **the InputManager**). When you unlock the camera, you can move it at will.

It has attached the **MessageController** script, a little script when you can display messages with sounds at the same time (in the middle of the screen), display a message for the player (such as why you block his actions or some informational message) or display a count down in the middle of the sceen (for the moment I only use this for the respawn count down).

It has attached too the **MarkController** script that allows you move the character or instantiate pings (see marks and pings section).

Marks and pings

The **MarkController** script is the responsible of instantiate the marks used for the movement of the player. You have two differents marks, the first type (with a green material) is used only for move the player, and the second type (with a red material) is used for move the player and and order them to attack an enemy is this is on his vision range.

Of course, the MarkController script support the clicks over objects (such as minions or towers) and you can set a directly target clicking over them.

Another functionallity of this script is the possibility of instantiate **pings** with their respectives sounds. If you ping an enemy or ally, will play the sound and will display a message with information about this ping to the player.

The MarkController can do this actions too clicking inside the minimap.

Minimap

The minimap has a **MinimapGUI** scripts as an example of implementation of a little gui system for this kit. You can implement this over the spells icons (see the spells section) or over the HP and MP bars.

Another important script is the **MinimapIcon**, that defines the icon of the object that has been attached in the minimap with a desired with and height.

Spells

The spell system is the most complex system of this kit. First you have a **SpellsController** script attached to the player. This script manages the cast of the spells and draws their icons and count downs in the screen.

The SpellsController has an array of spell scripts (they are attached to the player too) and it will manage all of them with their count downs or requisites for cast.

A spell script is a script that inherits from the **SpellScript** script. The SpellScript is only a class that define the essentials variables (such as icon, mana cost, cooldown, etc) of a spell and the method **CastSpell()**.

You only need extend your script from this class for inherit all things defined in the SpellScript script. After, you need define the actions performed your spell.

I include some different examples such as heal, fire ball cast or a simple teleport.

Towers

The towers have the simplest AI system, defined in the **TowerAI** script. This script only expects an enemy enter inside our vision range for start an offensive action.

Minions

The minions have a cycle of behavior. First, they are spawned in waves in an interval of time by the **MinionsSpawner** script.

After their **MinionAI** script defines their movement from an array of Transforms called waypoints. If an enemy enter on their vision range, then start an offensive action over them ultil death.