Final RPG project

Contact :
pondomaniac@outlook.com
Contact :
http://pondomaniac.wix.com/pondomaniac

Table des matières

Introduction:

This is a complete RPG project. It's using the latest Unity features like Sprites, 2D features and mores.

In this guide we will show you how to create your own RPG with your own assets and also we will expose some key features to help you personalize and enhance the game experience.

Unity 5 and 2d:

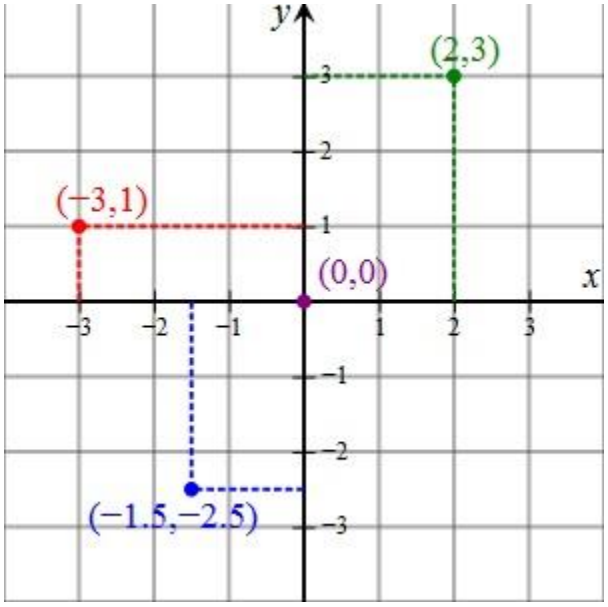
Unity 5 introduces some new 2d game creation tools, the main idea behind this is that creating a 2d game inside a 3d tools is not that simple, so the unity team introduced some new features to make it more easy: 2d view, Sprites, 2d collision, 2d physics...

For more information please refer to the following link :

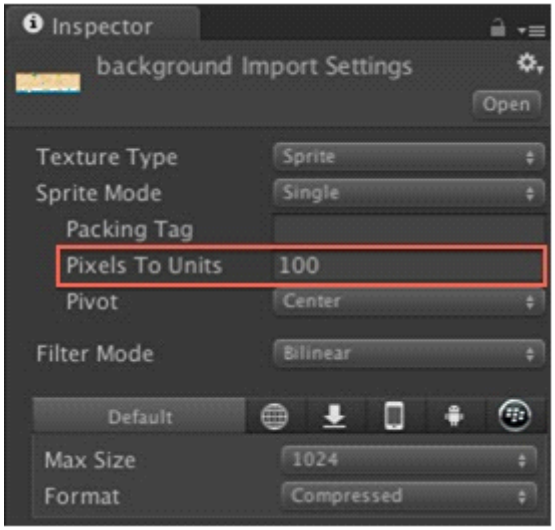
unity3d.com/pages/2d-power

Unity 3d Coordinate system and units:

The Unity coordinate system represents how Unity locate objects in the world, since we have make the choice of a 2d project we will focus just on the X and Y axis ,The (0,0) represent the center of the scene, each square represent a Unit.



When you import a 2d asset you can define in the asset inspector how many pixels correspond to a unit, in the example bellow 100 pixels represent 1 unit:



The organization of the project:

You can find below a print screen of the project Tree:

Animations: Contains all the animations used in the project.

Documentation: Contains the documentation of the project.

GameObjects : All the game objects used in the project.

Maps: All the TMX maps used in the project.

Materials: Materials used by the UI

Musics : All the static music used in the project (there is also some sounds presents in the Resources folder).

Ressources: It contains all the “Prefabs”, “Sounds” and “Sprites” used dynamically in the project.

Scenes: All the scene of the project.

Script: The project’s code.

SimpleParticlePack: It contains the particle effect used in the battle.

Sprites: It contains all the project's sprites.

Tiled2Unity: It contains the code that uses Tiled maps.

How the scenes are organized ?:

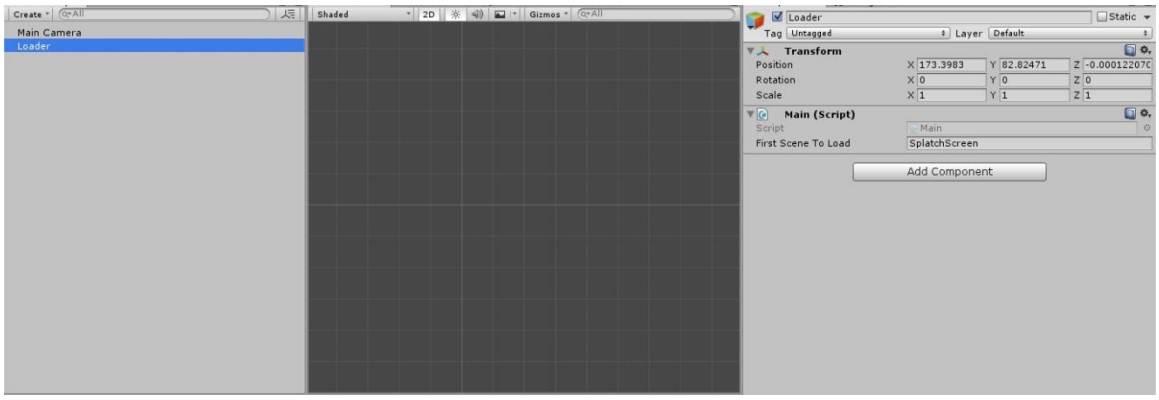
There are 6 types of scenes:

-Loader scene
-Splatch screen scene
-Main Menu Scene
-Character Name scene
-Exploration scenes
-Battle Scenes

We will describe each type of scenes and its main components:

Loader scene:

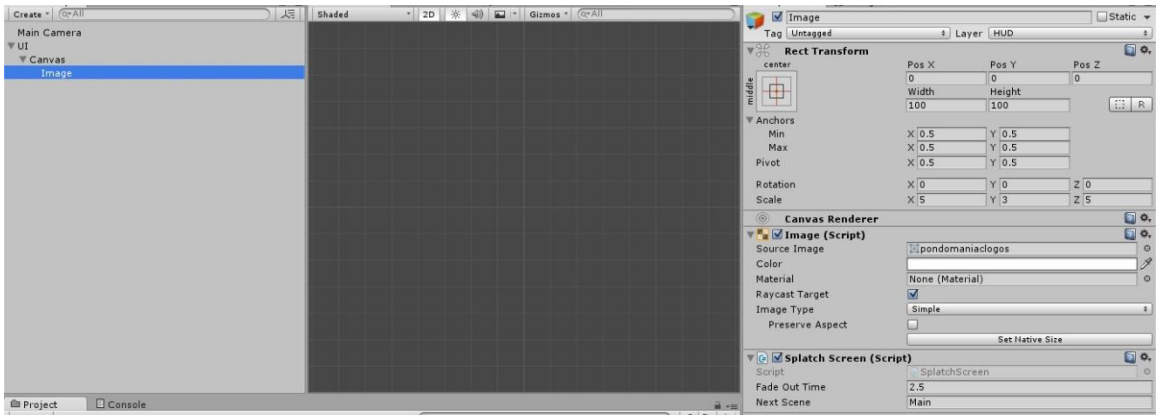
This is the first scene to load, it's an empty scene and it's used to load all the static objects that will be used and shared by the Game scenes. There is a "Main" script attached to the "Loader" game object, the "Main.cs" script contains for example all current characters, objects, current scene, and previous scene and more, you can open the script to have a better idea.



First Scene to Load: The scene that will be loaded when the game launch.

Splatch screen scene:

This scene contains the splatchscreen logo; you can replace the image by your own logo by changing “Source Image” on the Image component.

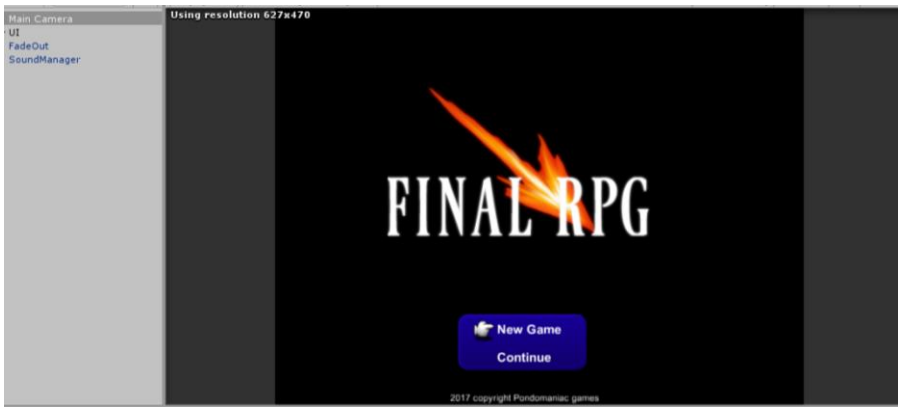


Next Scene: The next scene to load.

Fade Out Time: The duration of the animation.

Main Menu screen scene:

This scene contains the main menu UI where you choose “New Game” or “Continue”, Here is a description of the objects the scene:



Main Camera: The orthogonal camera used by the scene.

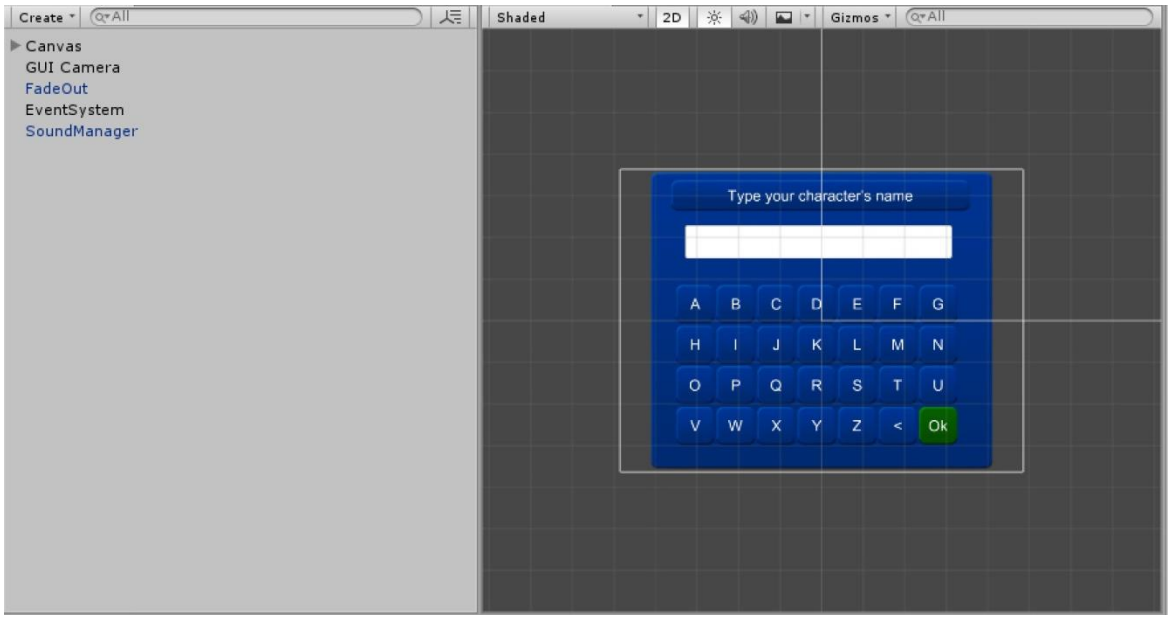
UI: The buttons that redirect to different scene you can take a look at the script that will just load a new game or Load the last saved file.

Sound Manager: This game object plays the music of the scene and also contains several static functions that play different sound effects.

Fade Out: Play a fade out animation you can parameter the duration.

Character Name scene:

This scene represents a keyboard that allows the player to type his name, you can also use physical keyboard to do it.



Canvas: Represent all the buttons of the screen

GUI Camera: The orthogonal camera used by the scene.

Fade Out: Play a fade out animation you can parameter the duration.

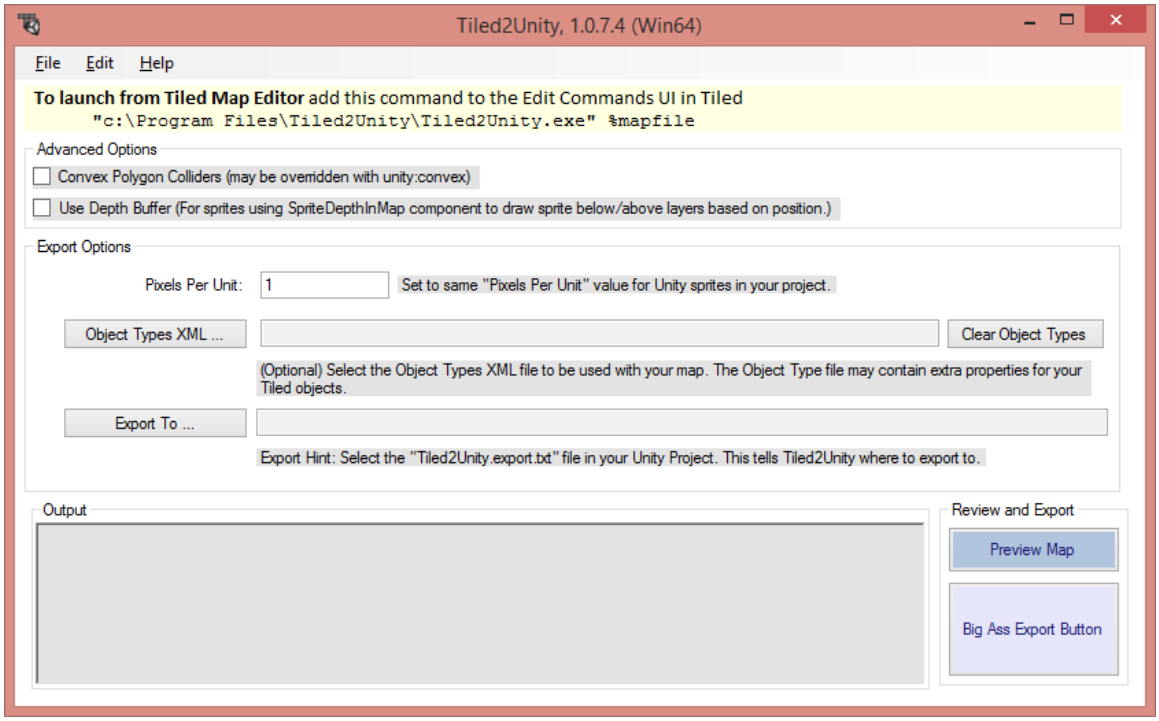
Event System: Tis game object catch all the events of the scene.

Sound Manager: This game object plays the music of the scene and also contains several static functions that play different sound effects.

Exploration scenes:

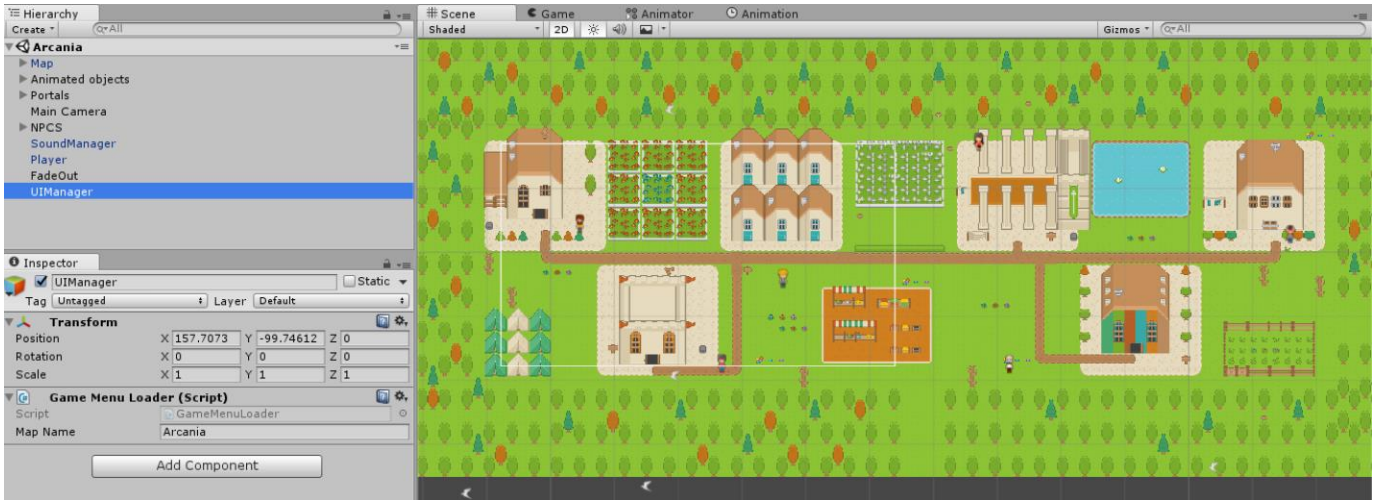
This type of scene is used when you want the player to discover a map, place, village or a world map to travel from place to place. It contains sometimes some NPC (Non player character) with whom you can speak, interact, and buy stuffs like equipment or items. In certain area you can also be attacked by monsters.

The maps are created with “Tiled” (http://www.mapeditor.org/) that is the best map editor used by famous game developers,
The maps are converted to unity using “Tiled2Unity” (http://www.seanba.com/tiled2unity), actually Tiled2Unity convert all tiles and shapes into Game Objects with collisions inside Unity using a program, it locates the file that you created using Tiled and by pressing “Big Ass Export Button” it transfers it to Unity.



To understand how Tiled2Unity works and to be up to date I advise you to look at the marvelous tutorial series made by Seanba that starts here:
https://github.com/Seanba/Tiled2Unity/blob/master/doc/getting-started.md .

Let’s return to our scene, we will describe bellow the print screen each component in the scene:



Maps: The maps generated by TiledToUnity.

Animated objects: The animated objects of the scene like birds, fire camps, waves.

Portals: It represents the doors to different scenes and also to leave or enter the village/Caves...

Main Camera: The orthogonal camera used by the scene.

NPCS: All the Non player characters of the scene.

Battle Area: The area where the player can be attacked, you can define which battle scene to load and also the rest time so the player can't be attacked in them.

Sound Manager: This game object plays the music of the scene and also contains several static functions that play different sound effects.

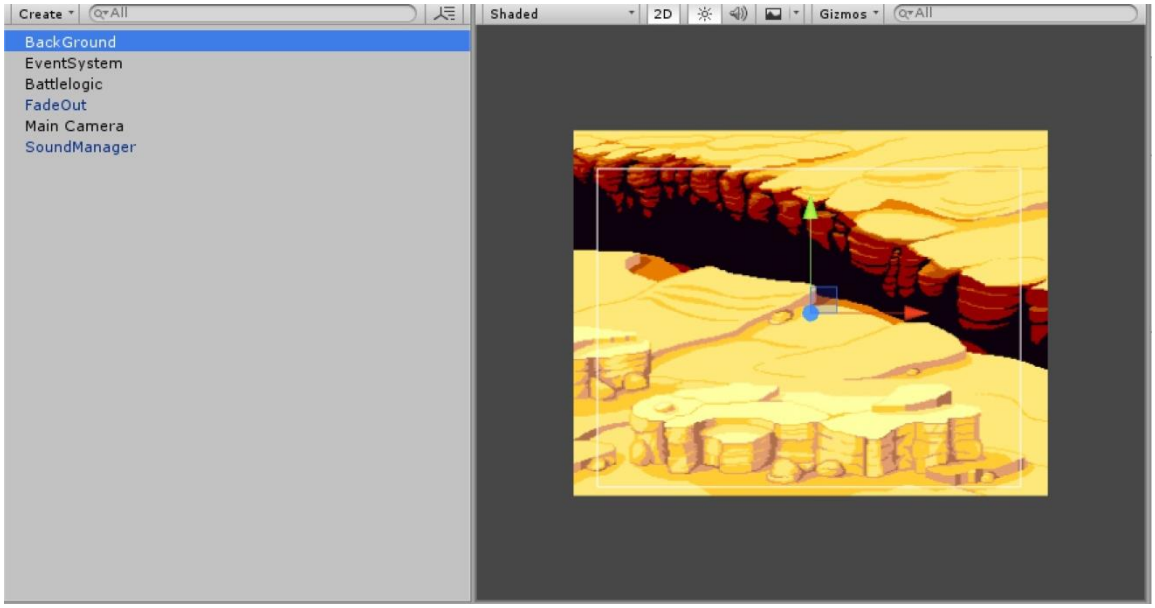
Player: The player game object that you can control with keyboard.

Fade Out: Play a fade out animation, you can parameter the duration.

UIManager : It loads the UI gameobject from the resources folder .

Battle scenes:

This type of scene is used for battles, all battle scenes names begin with "Battle" prefix, it's coupled with BattleArea gamebject (see the Exploration scene) so you redirect the battle area to your Battle scene.
You can parameter the battle scene to generate a certain type of monsters and also the number of the generated monsters.



BackGround: The background of the battle scene.

Event System: Tis game object catch all the events of the scene.

Battlelogic: The game logic of the battle scene.

Fade Out: Play a fade out animation, you can parameter the duration.

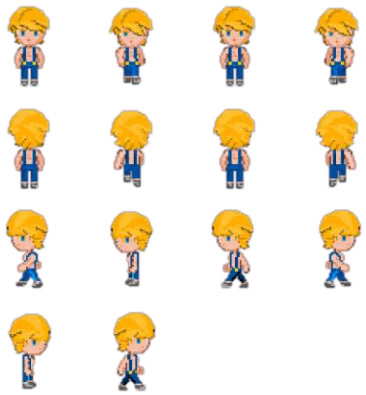
Main Camera: The orthogonal camera used by the scene.

Sound Manager: This game object plays the music of the scene and also contains several static functions that play different sound effects.

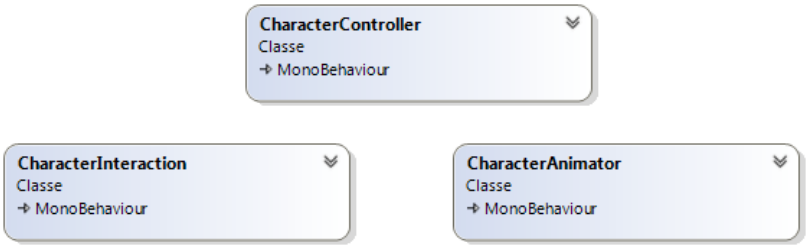
Behind the scenes:

Player:

We will describe in this section the player Game Object and how it move, get animated and interact with others Game Objects, it's the same concept for NPC character that we will cover later :



Tree components are used to move and animate the character, we invite you to take a look at the parameters of each script:



CharacterController : it coordinate between character movement and animation

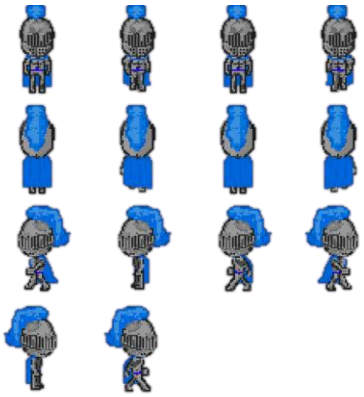
CharacterAnimator : Contains a dictionary of all characters movements, and play the corresponding animation base on the Animator component that is attached to the game object. Here is tutorial to make your own animation from your tileset (image that contains all 2d animations):

https://unity3d.com/learn/tutorials/projects/2d-roguelike-tutorial/player-and-enemy-animations

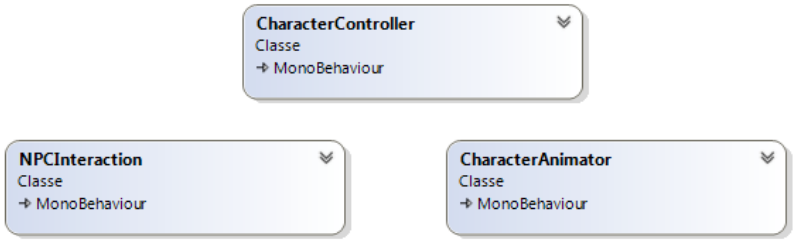
CharacterInteraction : Contains movement, Chat and market interaction, you can change the movement speed and also the smoothness

NPC Non player character:

The mechanism of non-player character is like the player mechanism, there just some differences, like the IA movement and auto interaction with the player:



Tree components are used to move and animate the character; we invite you to take a look at the parameters of each script:



CharacterController : it coordinate between character movement and animation

CharacterAnimator : Contains a dictionary of all characters movements, and play the corresponding animation base on the Animator component that is attached to the game object. Here is tutorial to make your own animation from your tileset (image that contains all 2d animations):

https://unity3d.com/learn/tutorials/projects/2d-roguelike-tutorial/player-and-enemy-animations

NPCInteraction : Contains IA, movement, Chat and market interaction, you can change the movement speed, Movement area, side and also the smoothness of the movement

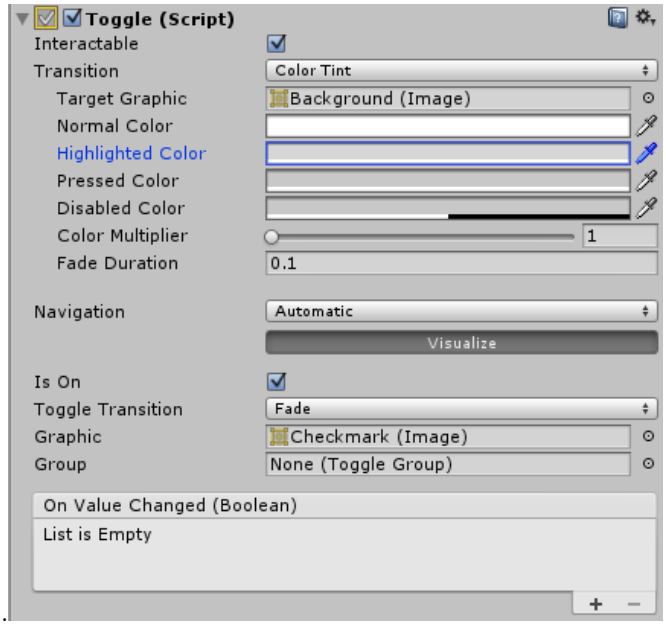
UI (User Interface):

Game Menu:

The GameMenu represent the in game menu where you can pause the game,change equipements, spells, use items, save , exit the game …

The game menu is a game object called “UI” that islocated in the resources folder, it’s instantiated by the UIManager when you play the the game.

The mechanism consist of a series of Toggle components that’s launch a function when the OnValueChanged is triggered. You can take a look at the script corresponding in Scripts/UI/ GameMenuUI.



Battle Menu:

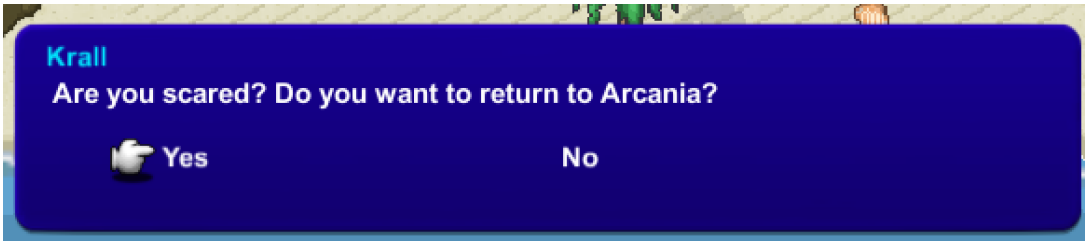
The battle menu that is called “BattleUI” is like the GameMenu but contains different information and action like Weapon to use, Spell to launch, item to use in battle you can take a look at the script corresponding in Scripts/UI/BattleUI.

Dialogs:

The dialog uses the 2DCollision system to launch the dialog UI. The text of the dialogs come from the DialogData present on each talkable NPC. There is 2 types of dialog: Dialog without decision,Dialog with decision, the main difference is the a decision to make at the end as you can see bellow :

-Dialog without decision:

-Dialog with decision :

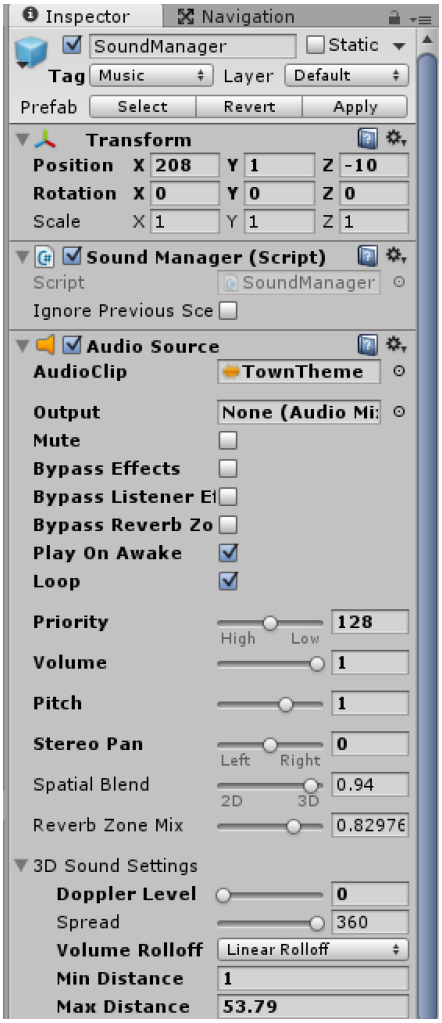


Market:

The market decision works like dialog data, using the market data component on the npcY you should specify the items id for displays it in the panel.

Sounds:

The sound is controlled by the SoundManager present on every scene, if the music is the same as the previous scene the music will continue to play even if the scene change, if you want to restart the music check “IgnorePreviousSceneMusic”. The SoundManager class contains also different useful static functions to play different sounds, please take a look if you want to understand more.



Datas:

The Datas in the game are static, it means that we don't use a database the datas are stored inside a file "Datas.cs" with fixed values, the main reason behind this is to make the game more compatible with different platforms, for example in the file you have the following:

```
var spell = new SpellsData();
spell.Name = "Fire ball";
spell.Description = "Send a fireball on your ennemies";
spell.AllowedCharacterType = EnumCharacterType.Wizard;
spell.PicturesName  = "S_Fire03";
spell.ManaAmount = 5;
spell.Attack = 10;
spell.ParticleEffect = "FireBall";
spell.SoundEffect = "foom_0";
SpellsData[1]= spell;
```

As you can see the datas are written inside the code, for ParticleEffect, SoundEffect,picturesName… You can store the following datas :

SpellsData : For spells and magic
ItemsData : For equipment and potions
CharactersData : For the characters in the team

Settings:

By setting we mean the general setting of the game used by the code, for example you will find the path to different folders,musics, speed of dialogs, dialog max characters, directional keys to move the player, the name of the saved file … we invite you to take a look at "Settings.cs" file to see all the general settings.

Next step:

This tutorial shows you just the main mechanism of the game, we invite you to open the project change the sprite and assets with your own assets and make your own game.

Don't forget to contact us at: pondomaniac@outlook.com
If you need some help, think that something is missing or need a new feature.

Credits:

We would like to thanks you and the following persons, for their assets or help in this project.

Graphics

Kenney.nl

Map generation

http://www.seanba.com/tiled2unity

Thank you!