Player movement 2D

In this tutorial we'll create a simple Player control for 2D. You will learn to code:

- Controller2D
- Player
- Player Controller
- Player Input
- Run

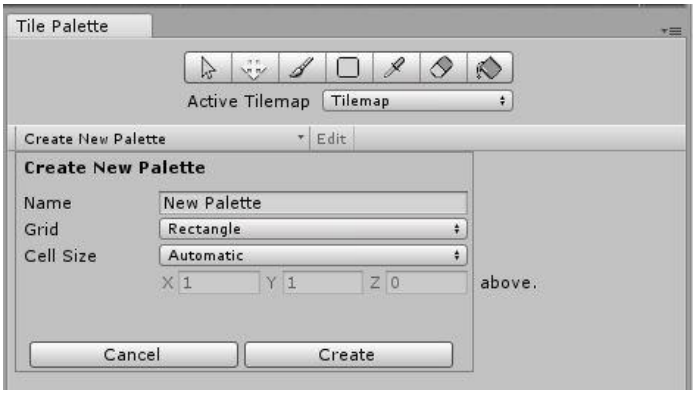
I would assume you know the basics of Unity editor as well we will be tutorial Unity 5/2018.

Game Design

Before we begin, we start by creating a simple level design that we can use to test out the game machine. It is going to be a 2D side scroller platformer. For the gameplay we will have a running action and a jumping action for the player to use.

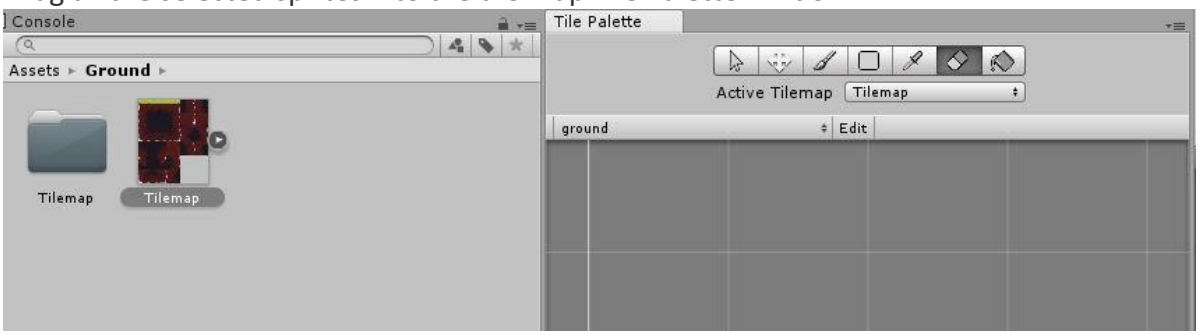
Set up

- To add a character sprite to game, simply drag and drop it from the Project folder into the Scene view. To see how easy the process is, select the Scene view, then drag the character sprite from the Sprites folder into your Scene view.
- We start with opening a new project without any packages and change 3D to 2D. Let get started with creating three folder: Player, Tile Map, Ground, player Tile map that makes it fast and easy to create the level design. So, in the editor, click Window -> Tile Palette to open the 2D Tile Palette window.

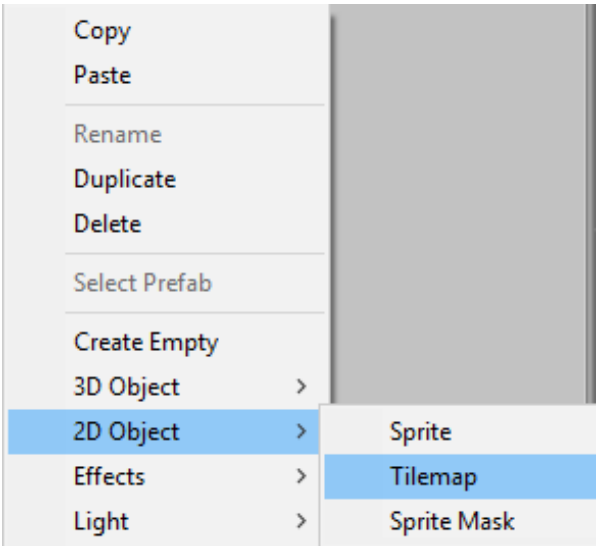


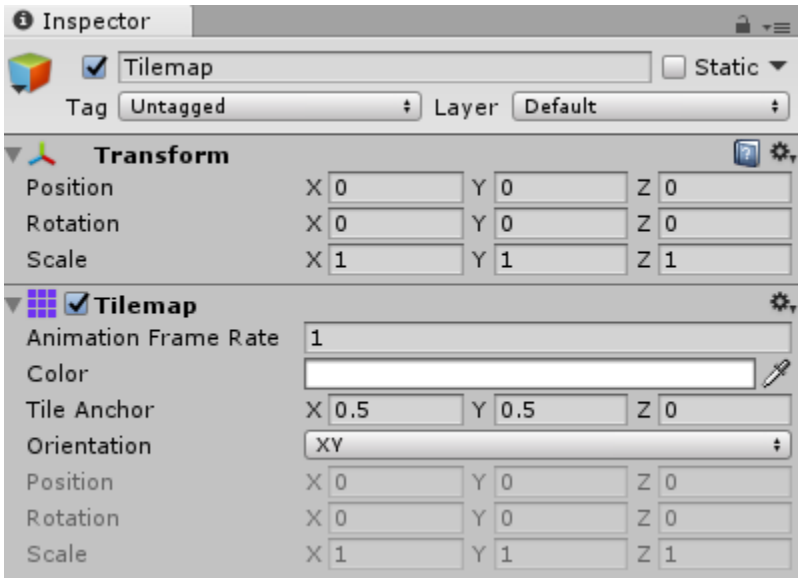
- Click in the middle what say “Create New Palette and name it Ground. Leave the grid and cell options as their defaults. This selector allows you to create brushes with different. When Clicking to Create and, prompted, choose to store the new palette in the project’s Assets\Tile map folder. Underneath this, folder called Ground.

- Drag all the selected sprites into the tile map Tile Palette window

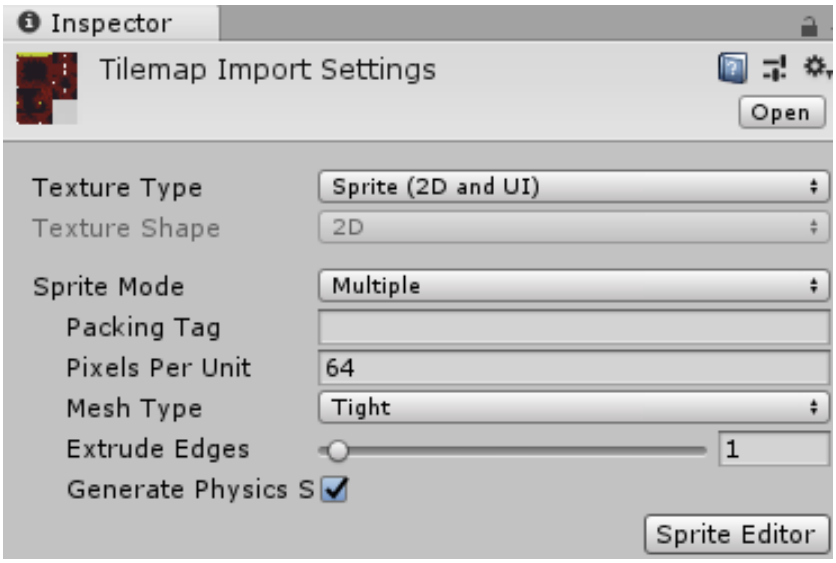


- Before we start painting, we need to create something called a “Tile map” in our scene. Right-click in the hierarchy and go to 2D Object -> Tile map.
- This has created two things: a “Grid” and a “Tile map”. This is how the Tile map Editor works, there is only one Grid in the scene, but there can be several Tile maps.





- The value we need to look at is the “pixels per unit” value. A larger value decreases the size of our image and, conversely, a smaller value increases the size. We have to employ some trial and error here in order to find the correct value.

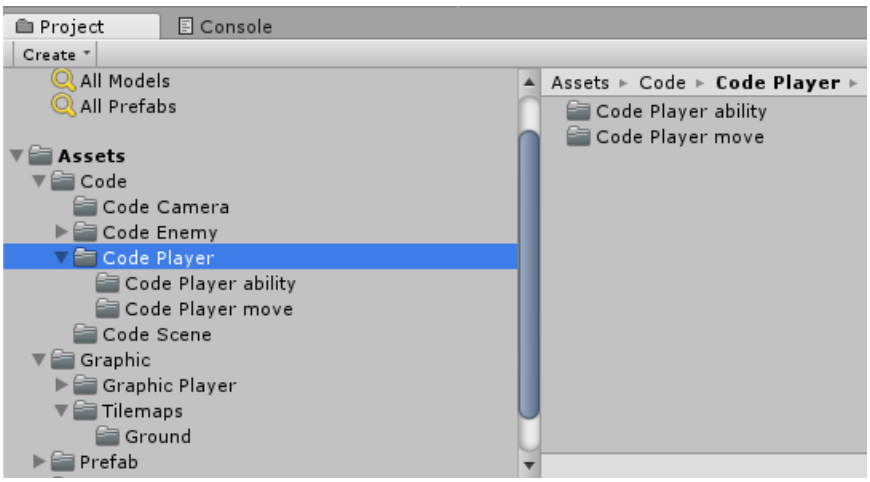


- Select all of the tile images that you used, change the “pixel per unit” value, and then click “apply that shouldn’t be overlapping. The very last part of this tutorial is about Tile Colliders. On any tile map, click Add Component and search Tile map Collider. Now what we are going to be using is the “Brush” to create other simply level design for 2d side scroller platformer.

Coding

Step 1

Create a three folder for the player, this first code player holds the other two folder, the code player moves, and code player ability as well create one folder for the camera



Step 2

Create a new C# Script, called it Controller2D, Player, PlayerController, PlayerInput and put it inside the code Player move folder and attach it to the main character of the game and for the CameraFollow attach the code to the main Camera.

Writing down the following code and some of the code will be in notepad to look at:

- Controller2D:In notepad due to how big to fit in
- Player : In notepad due to how big to fit in
- PlayerController: In notepad due to how big to fit in
- CameraFollow: In notepad due to how big to fit in
- PlayerInput : Using UnityEngine;

```
[RequireComponent(typeof(Player))]
public class PlayerInput : MonoBehaviour{
private Player;
private void Start(){
player = GetComponent<Player>();}
private void Update(){
Vector2 directionalInput = new Vector2(Input.GetAxisRaw("Horizontal"),
Input.GetAxisRaw("Vertical"));
player.SetDirectionalInput(directionalInput);
if (Input.GetButtonDown("Jump")){
player.OnJumpInputDown();}
if (Input.GetButtonUp("Jump")){
player.OnJumpInputUp();}
}
}
```

- Run: using UnityEngine;

```
public class Run : MonoBehaviour {
Rigidbody2D rb;
float movespeed = 5f;
float dirX;
// Use this for initialization
void Start () {
rb = GetComponent < Rigidbody2D > ();}
// Update is called once per frame
void Update() {
if (Input.GetKey(KeyCode.LeftShift))
movespeed = 10f;
else
movespeed = 5f;
dirX = Input.GetAxis("Horizontal") * movespeed;}
private void FixedUpdate(){
rb.velocity = new Vector2(dirX, rb.velocity.y);}
}
```