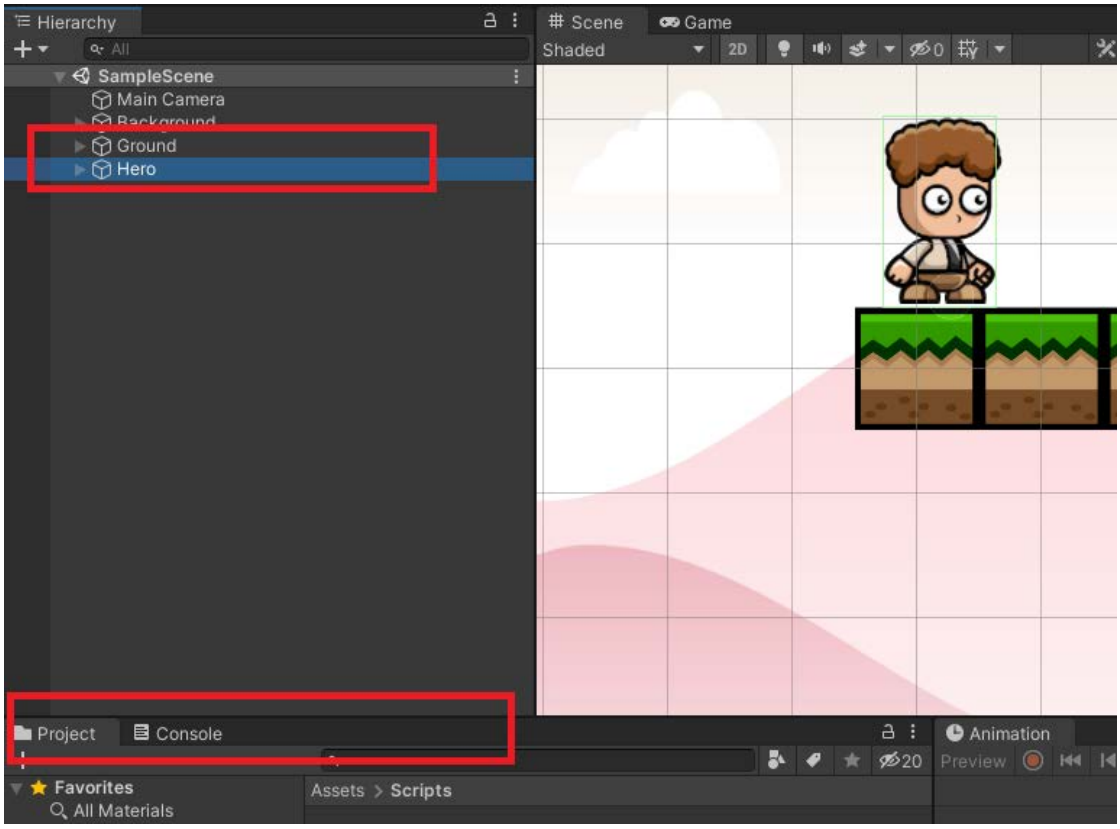
Лабораторная работа 3

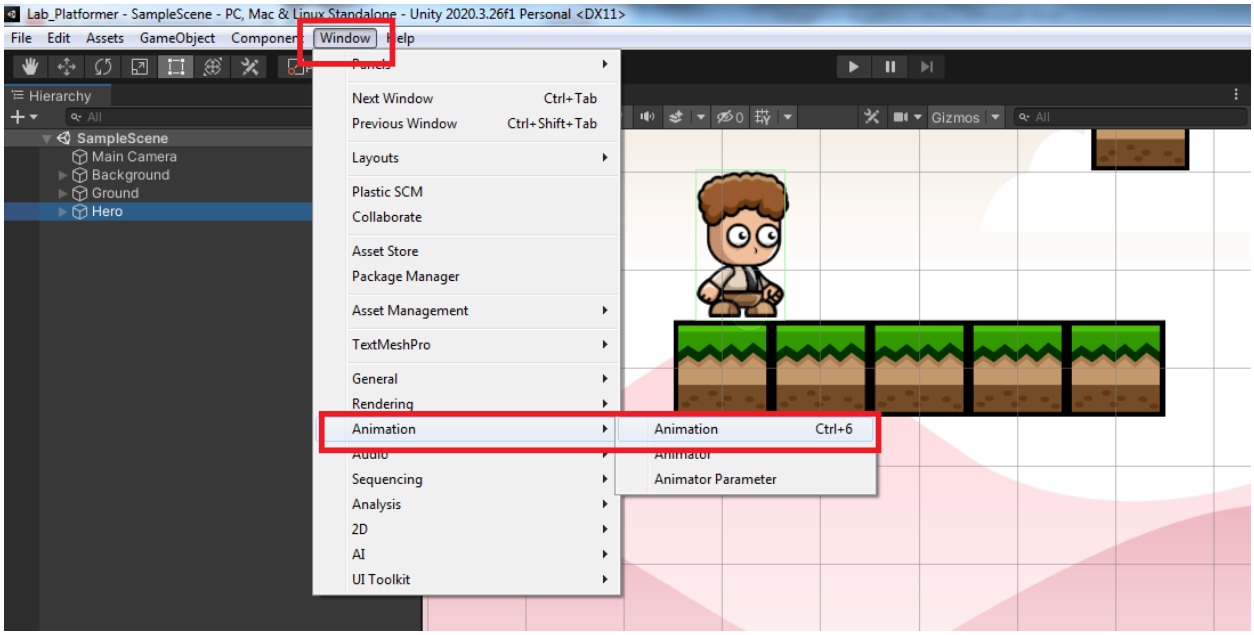
Платформер. Аннимация

1. Создадим покадровую анимацию для персонажа

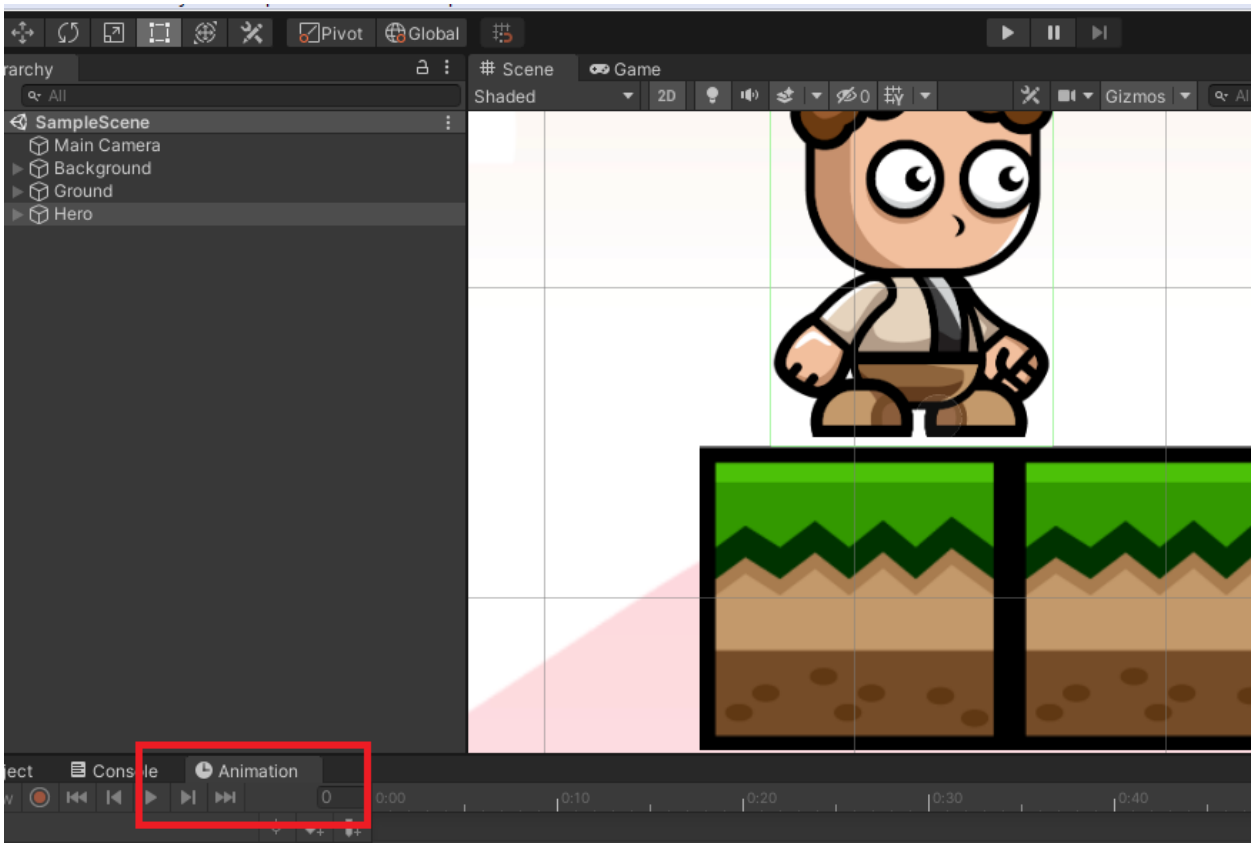
Выделяем Героя



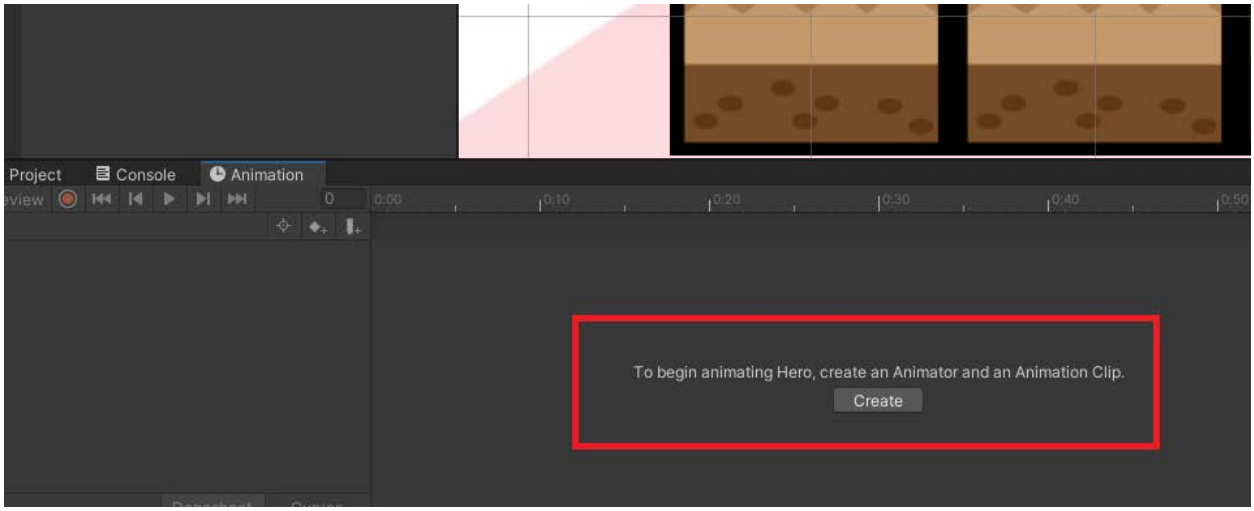
Если нет вкладки Аннимация. Добавляем



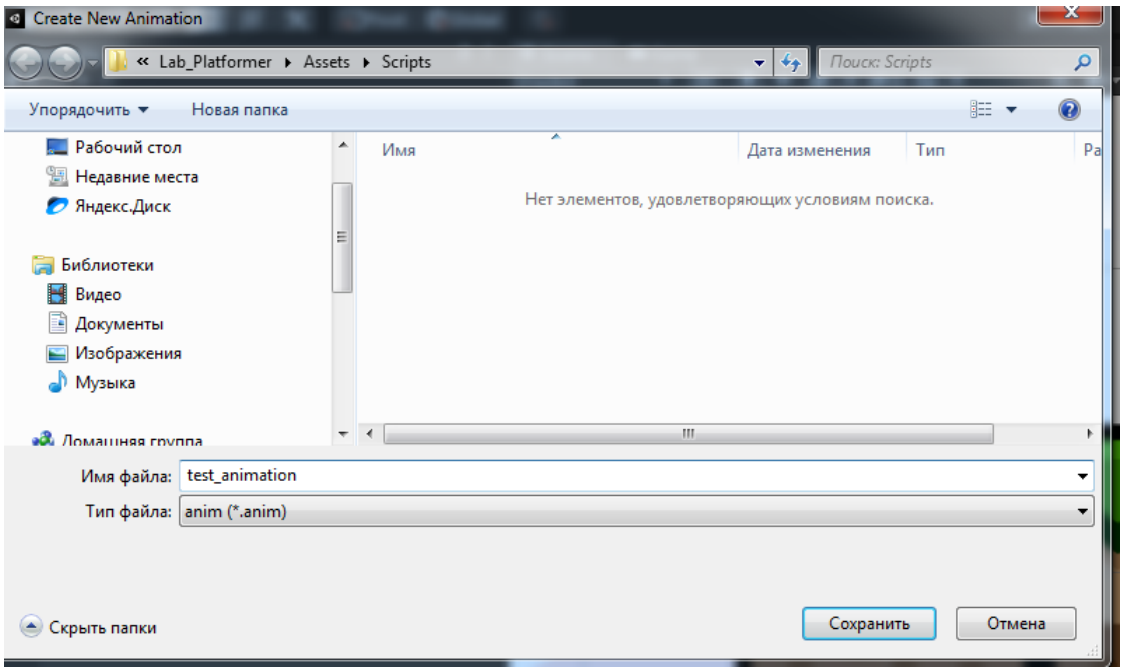
Результат. Добавляется вкладка Анимации



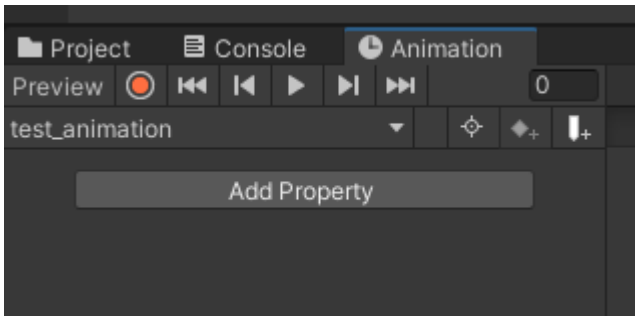
Создаем новый клип



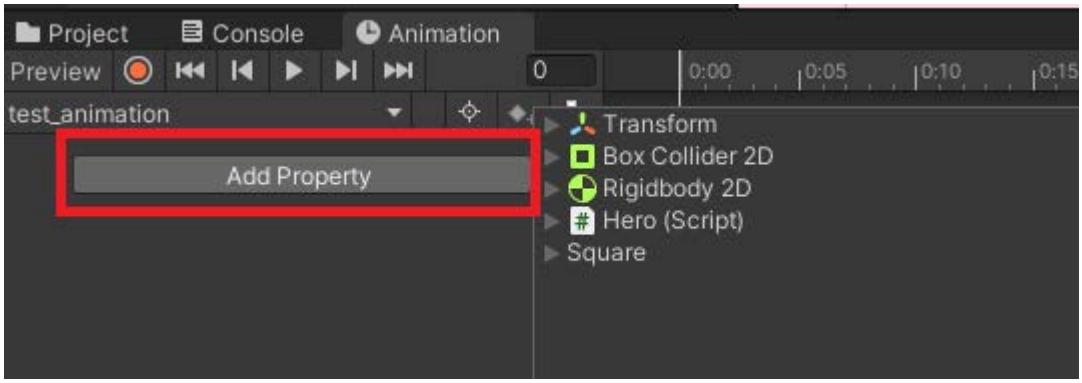
Указываем место. Имя анимации



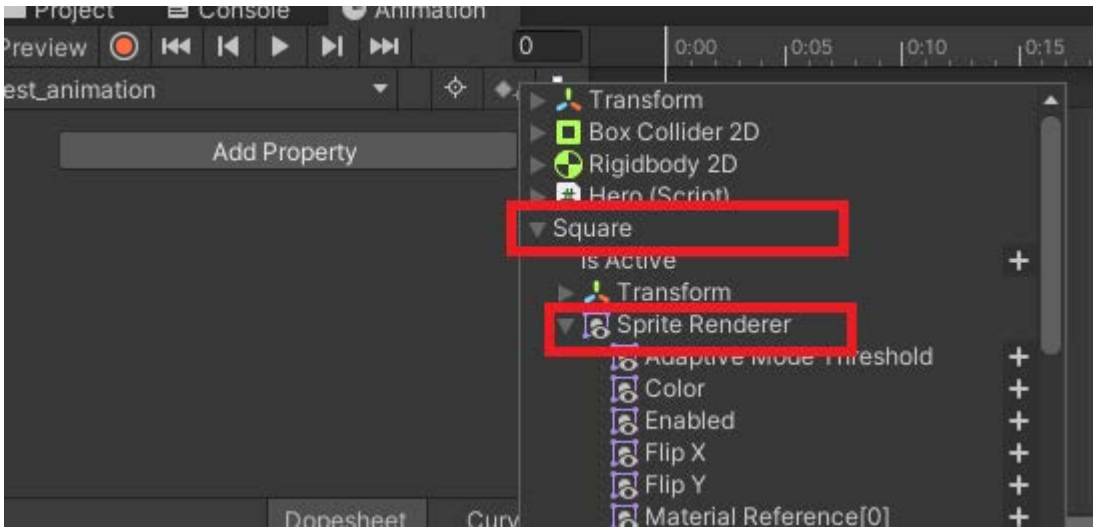
Результат

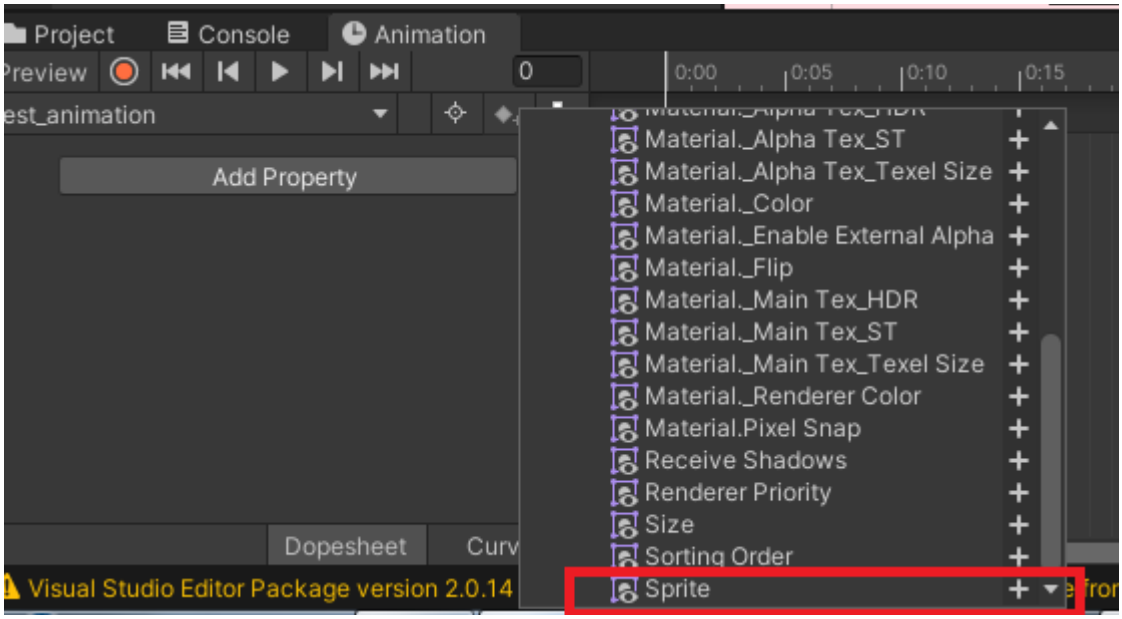


В Add Property можно выбрать свойство

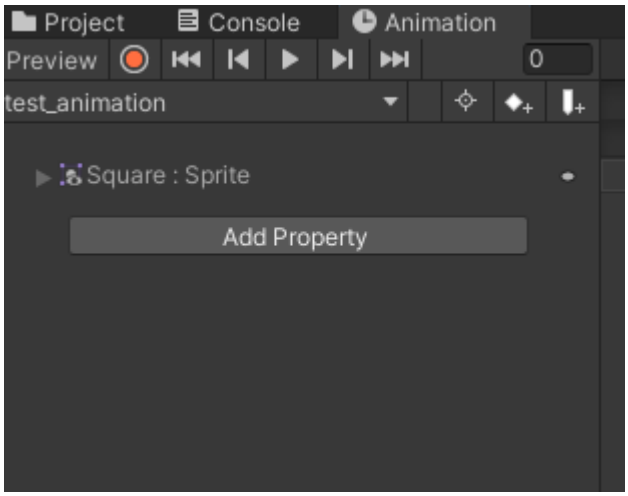


Добавляем Sprite Render -- Sprite


Add Property
Transform
Box Collider 2D
Rigidbody 2D
Square
Transform
Sprite Renderer
Color
Enabled
Flip X
Flip Y
Material Reference[0]

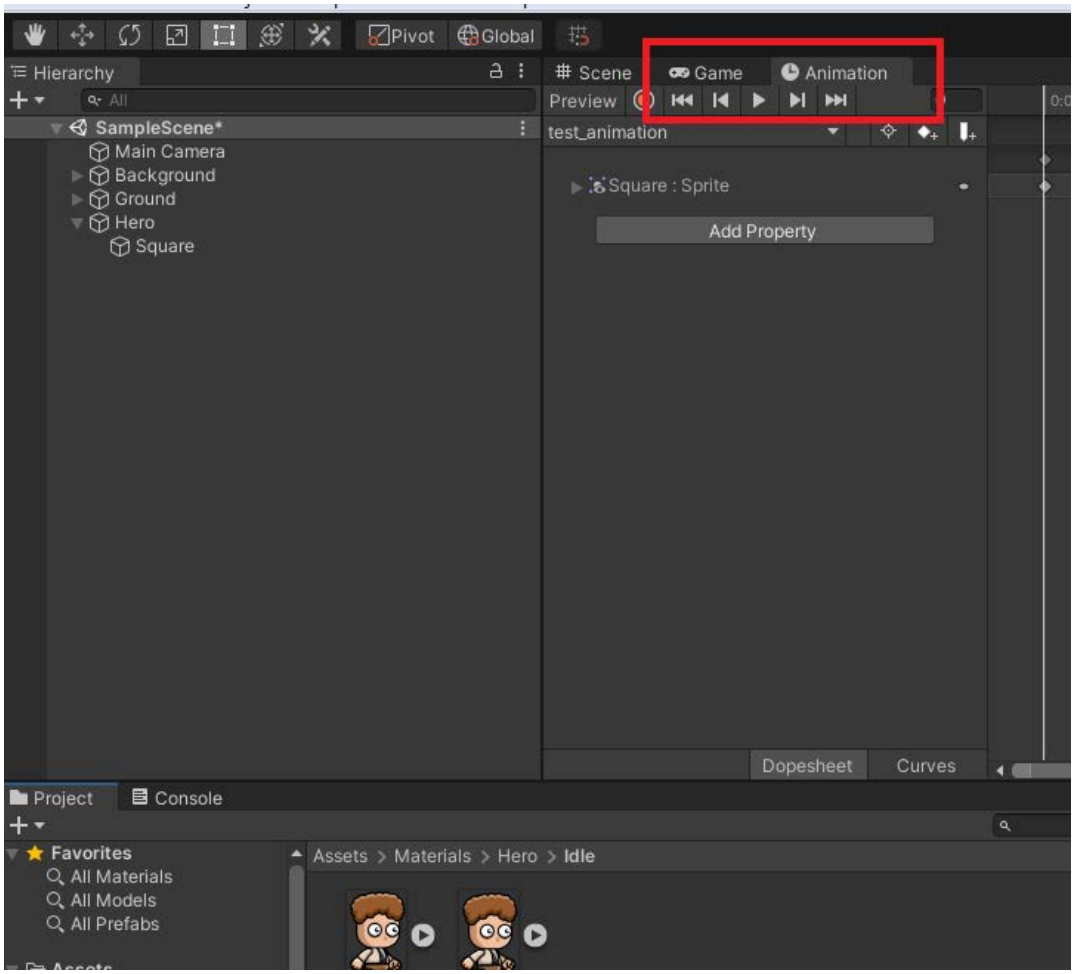

Project
Console
Animation
Add Property
Material._Alpha Tex_ST
Material._Alpha Tex_Texel Size
Material._Color
Material._Enable External Alpha
Material._Flip
Material._Main Tex_HDR
Material._Main Tex_ST
Material._Main Tex_Texel Size
Material._Renderer Color
Material.Pixel Snap
Receive Shadows
Renderer Priority
Size
Sorting Order
Sprite
Dopesheet
Visual Studio Editor Package version 2.0.14

Результат

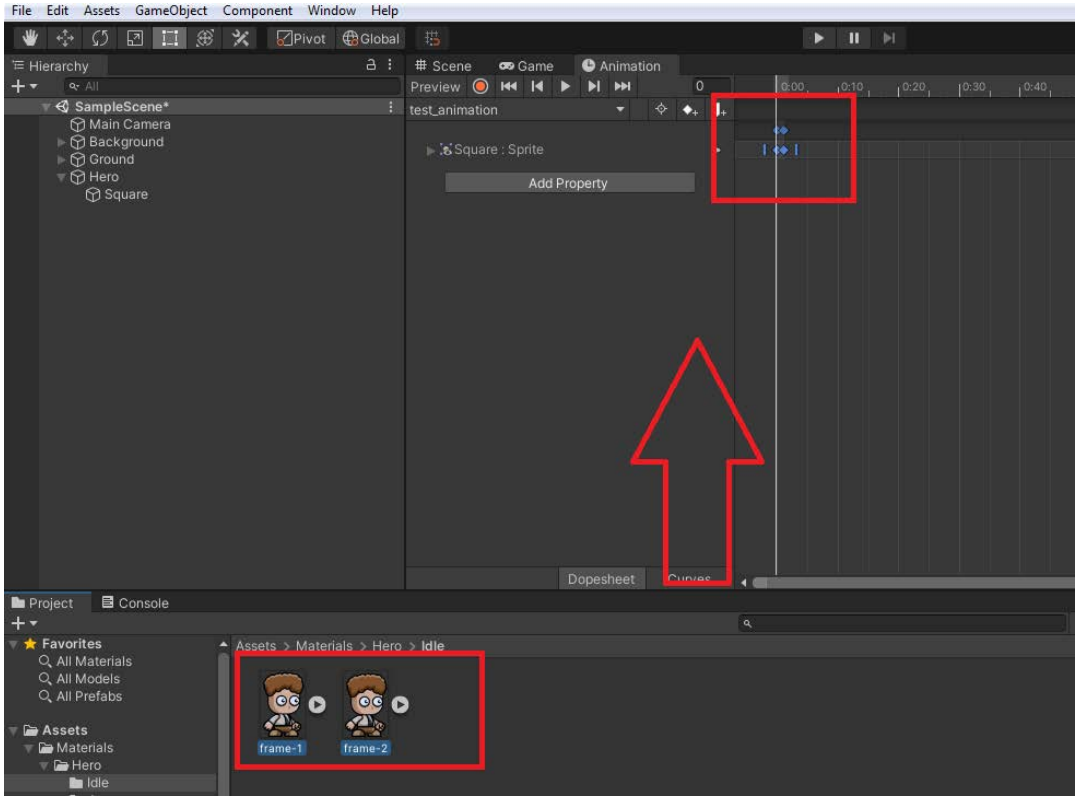

Project
Console
Animation
Square : Sprite
Add Property

Из папки Materials добавляем все наши спрайты анимации в Animation

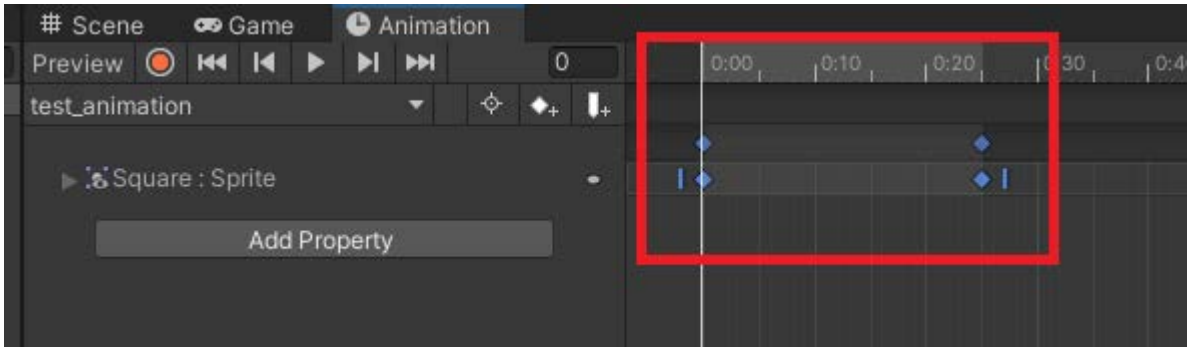
Для удобства работы перетянем рабочую область Animation вверх



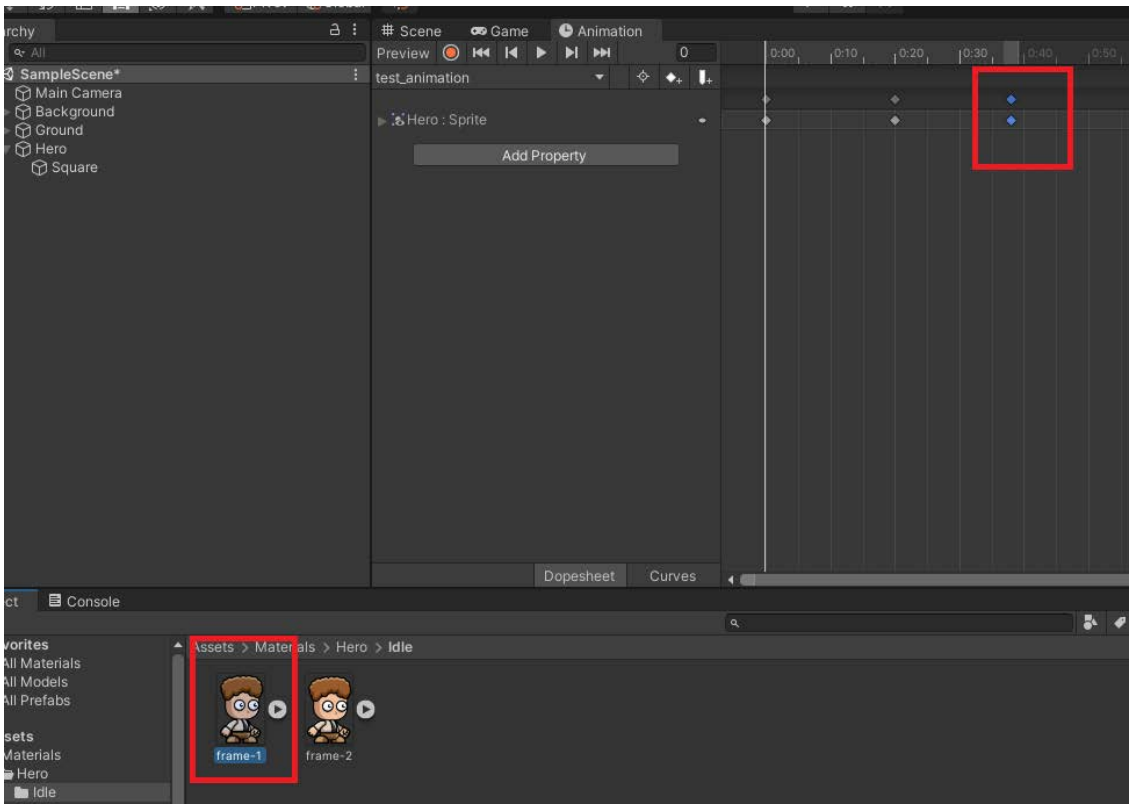
Выделяем спрайты и перетягиваем



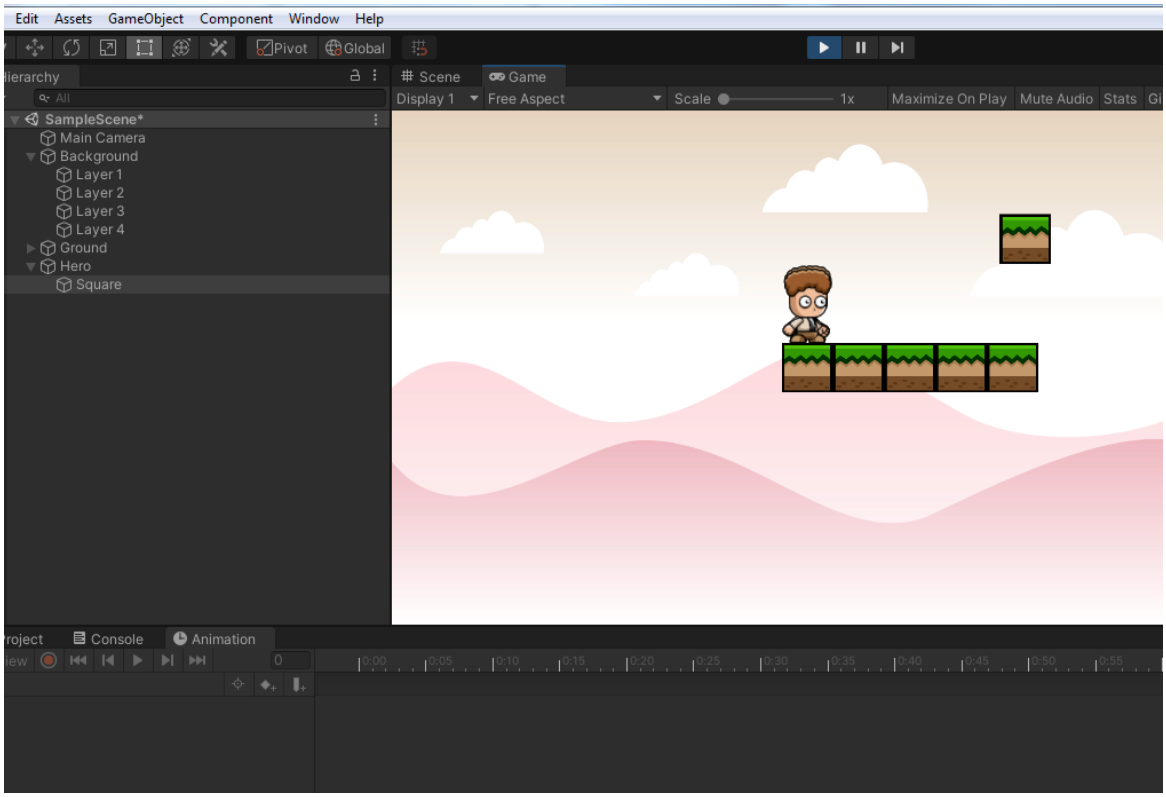
Растягиваем ключи. От этого будет зависеть скорость анимации(первый кадр, второй кадр)



Завершаем анимацию первым кадром. Перетягиваем первый кадр в конец

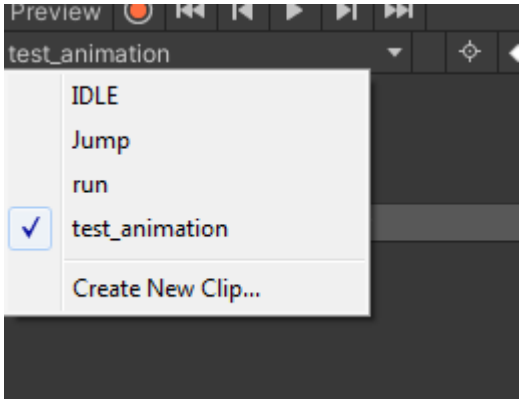


Вкладку анимацию вернем на место и проверим, что наш герой в состоянии покоя имеет анимацию

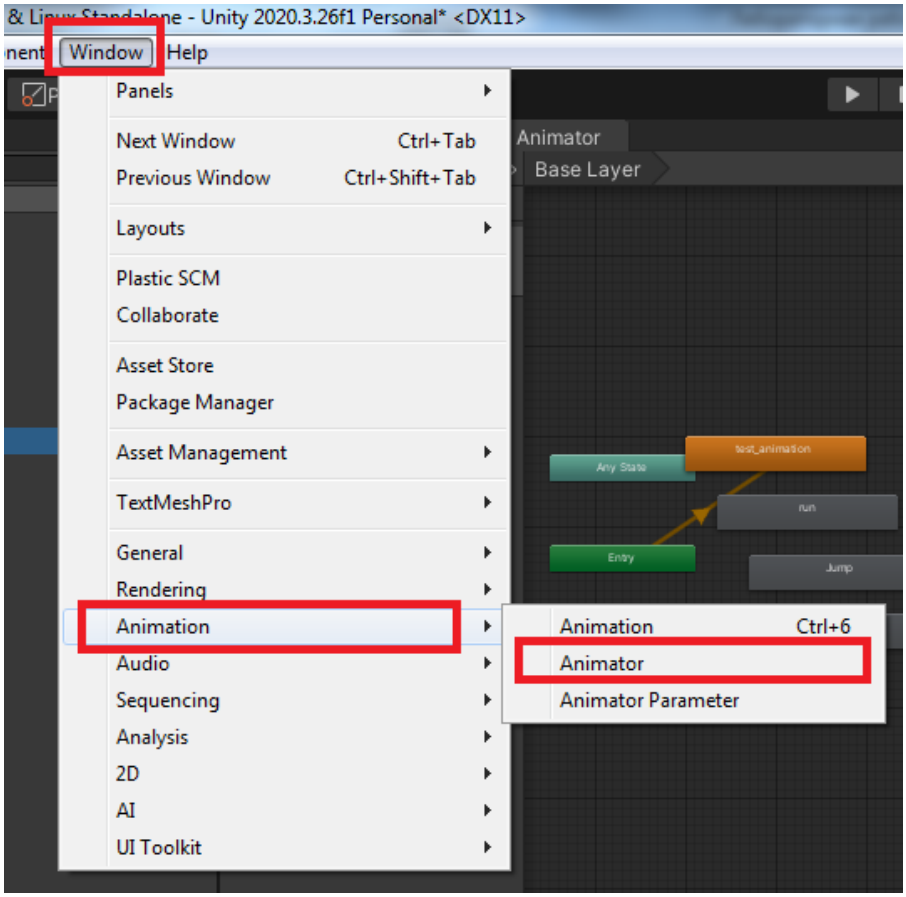


Аналогично используя спрайты анимации создадим

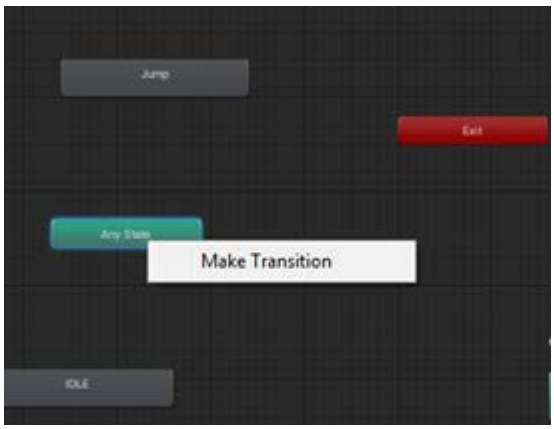
Run, Jump, IDLE

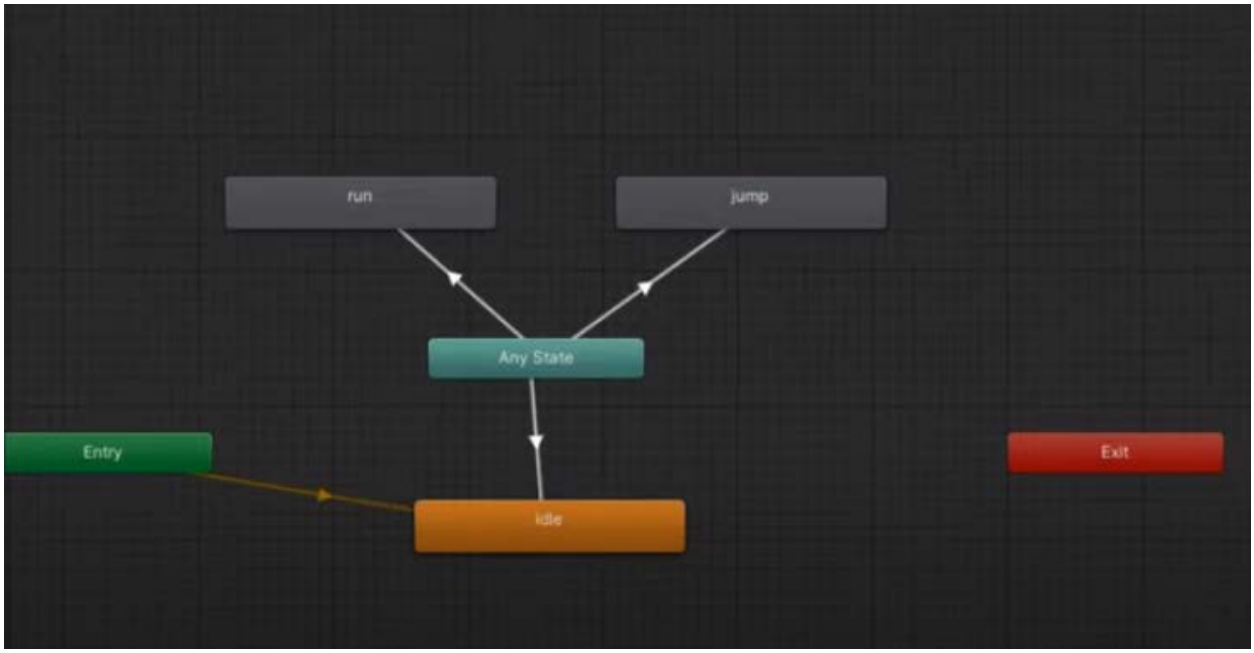


Далее открываем Animator (если его нет на экране добавляем)

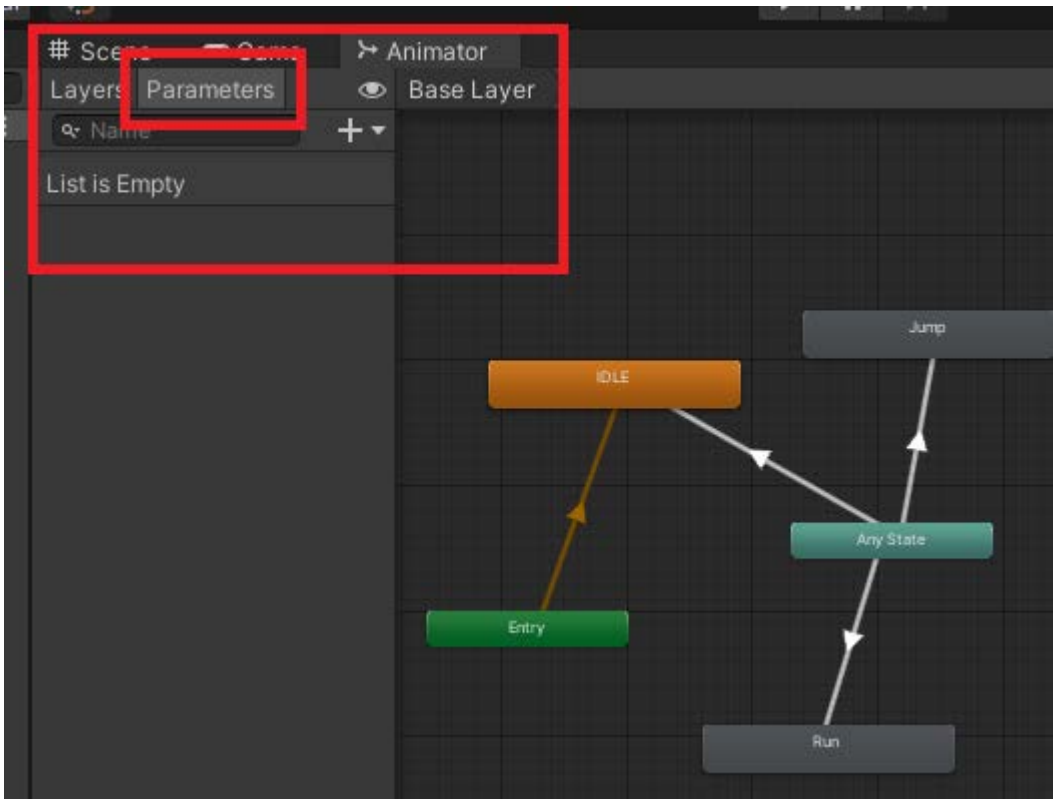


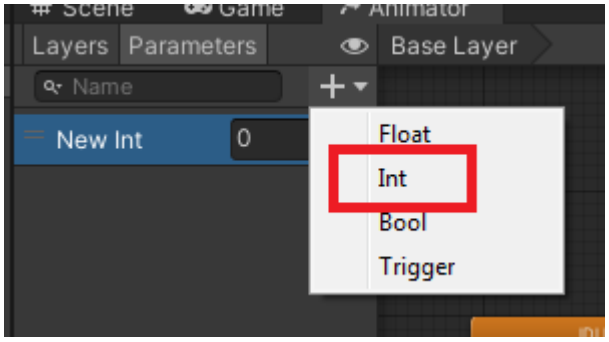
Делаем последовательность переходов



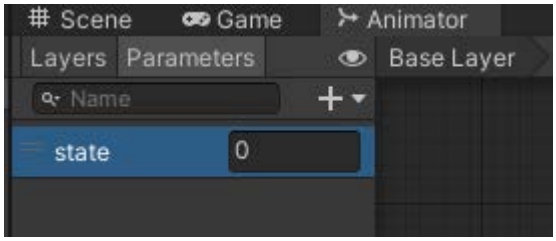


Далее создаем параметр для переключения состояний





параметр – state

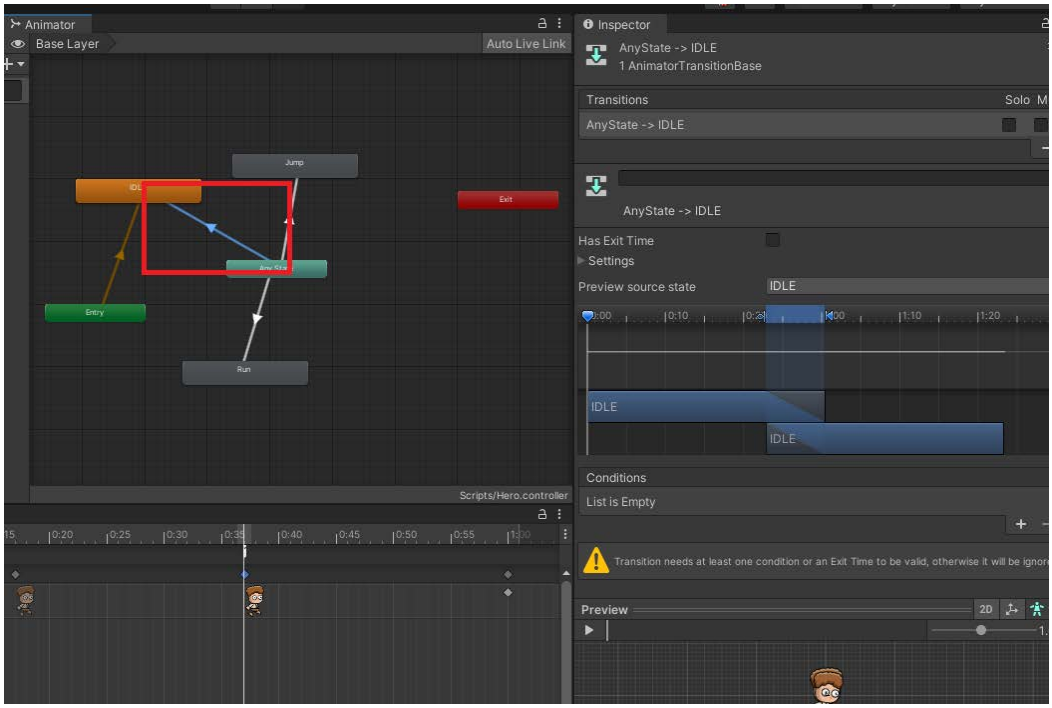


0 –IDLE

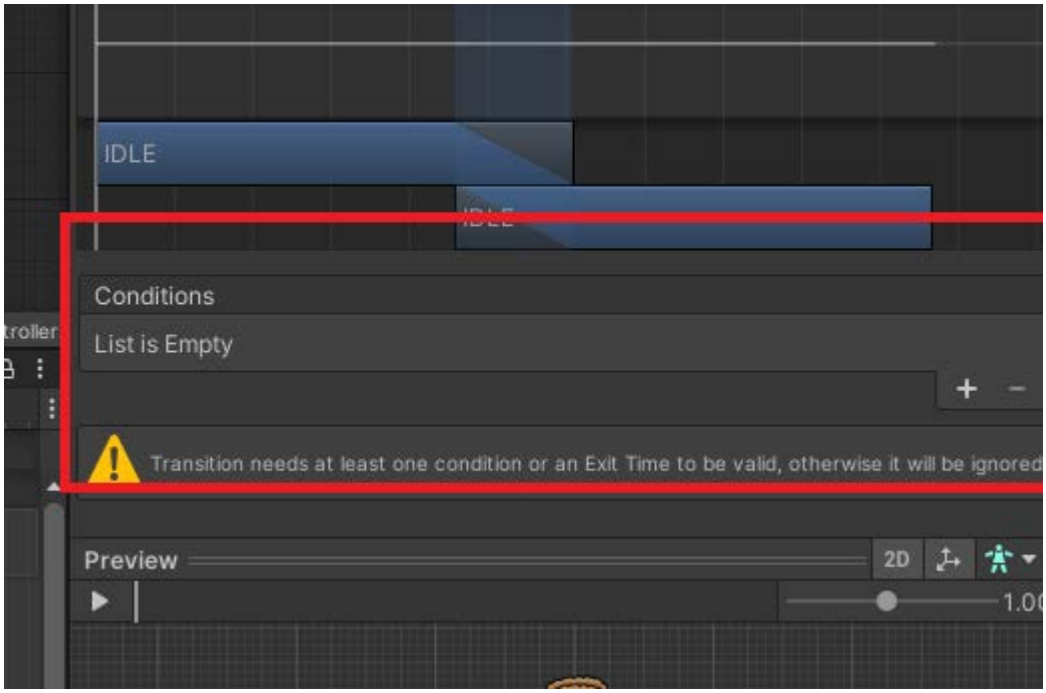
1-Run

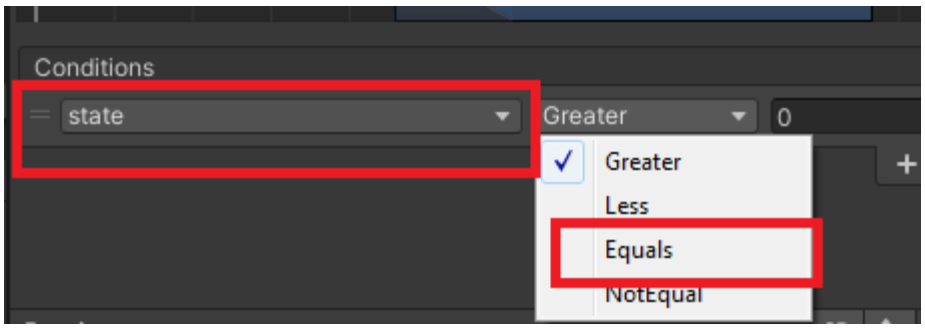
2 –Jump

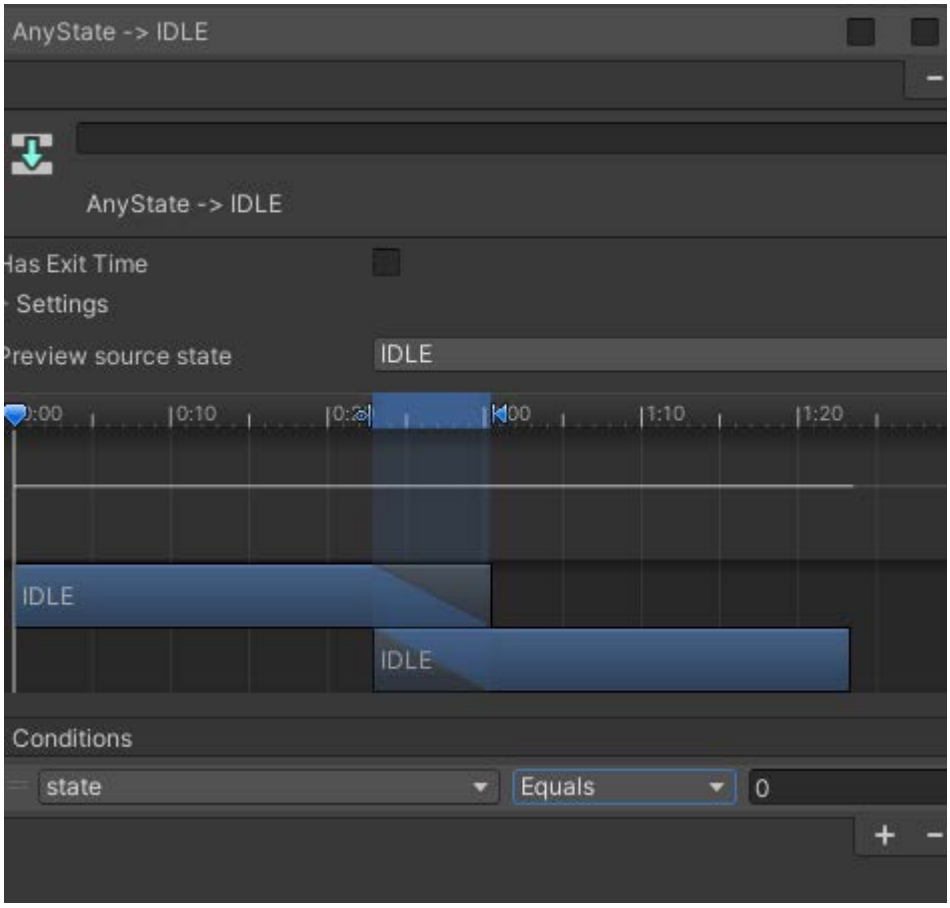
Выделяем стрелочку, которой будем присваивать значение



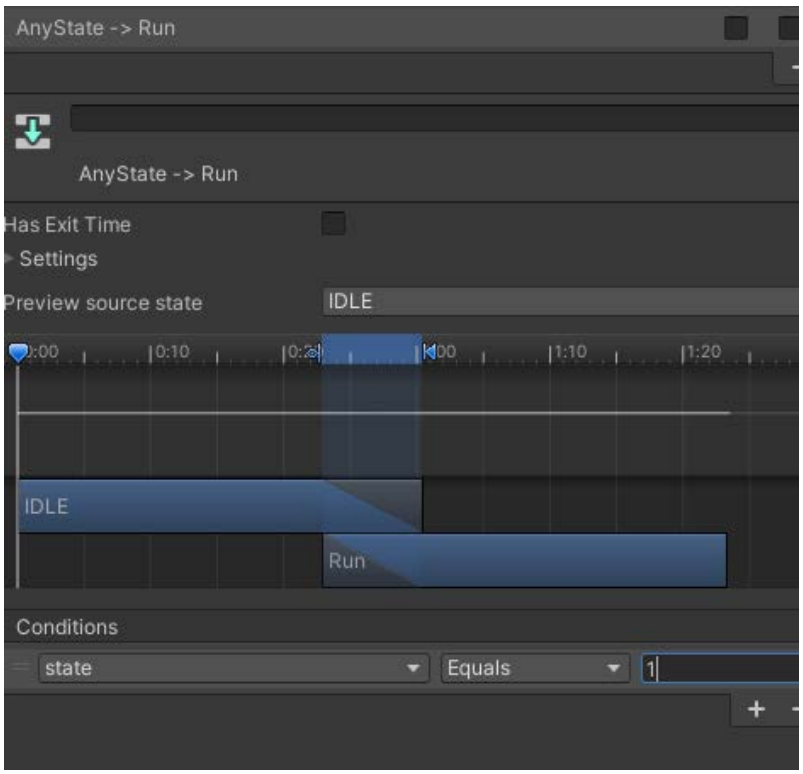
Добавляем в Conditions (нажинаем +)



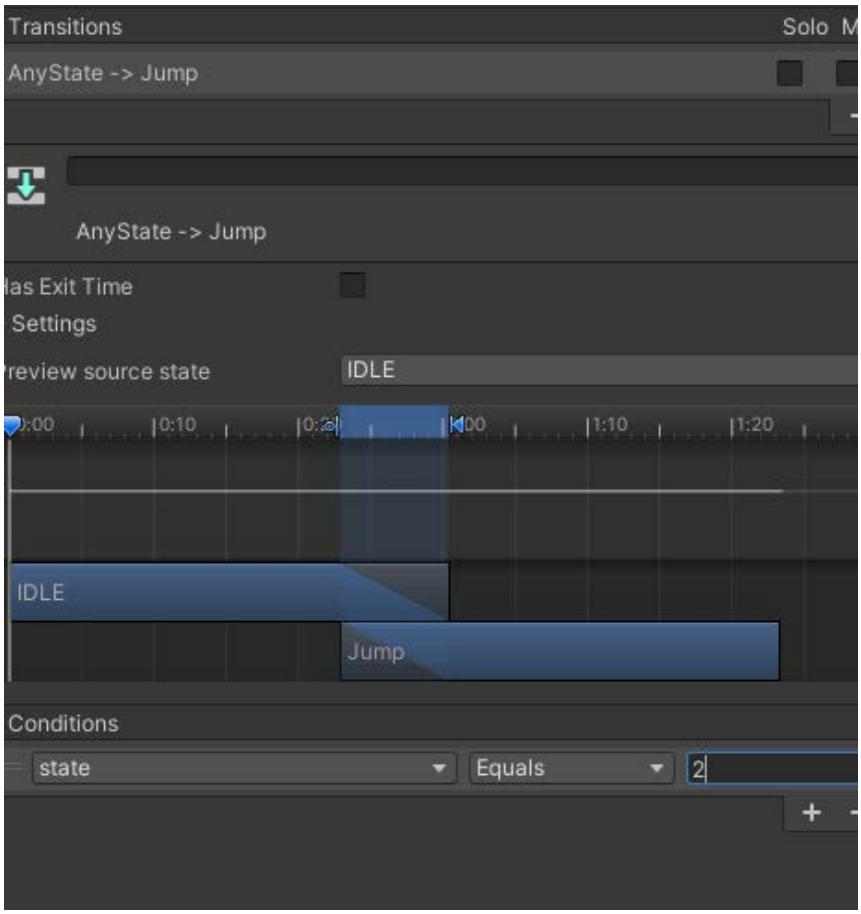




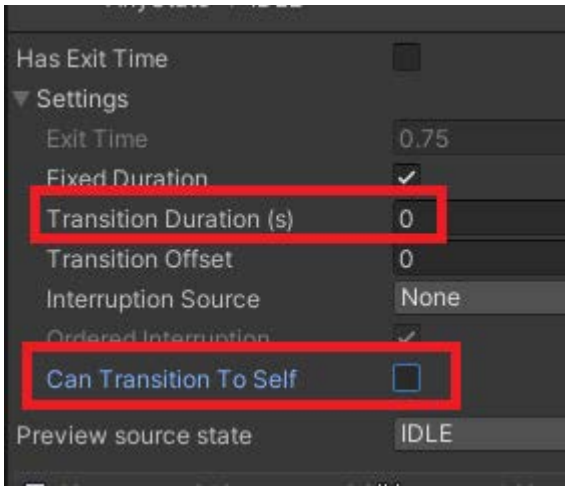
для стрелочки к Run


AnyState -> Run
AnyState -> Run
Has Exit Time
Settings
Preview source state
IDLE
IDLE
Run
Conditions
state
Equals
1

для Jump


Transitions
Solo
AnyState -> Jump
AnyState -> Jump
Settings
IDLE
IDLE
Jump
Conditions
state
Equals
2

Убираем также время задержки после анимации и переход в состояние самого себя



Добавляем в скрипт переменную и компонент.

```
public class Hero : MonoBehaviour
{
    [SerializeField] private float speed = 3f; // скорость
    [SerializeField] private int lives = 5; // количество жизни
    [SerializeField] private float jumpForce = 15f; // сила прыжка

    private bool isGrounded = false;


    private Rigidbody2D rb;
    private Animator anim;
    private SpriteRenderer sprite;

    Сообщение Unity | Ссылок: 0
    private void Awake()
    {
        sprite = GetComponentInChildren<SpriteRenderer>(); //обращение к спрайту
        anim = GetComponent<Animator>();
        rb = GetComponent<Rigidbody2D>();
    }

    ссылка: 1
```

список состояний вне класса, в той последовательности, что описали в аниматоре

```
52      }
53  }
54
    Ссылок: 0
55  public enum States
56  {
57      idle,
58      run,
59      jump
60  }
```

```
private Rigidbody2D rb;
private Animator anim;
private SpriteRenderer sprite;

private States State
{
    get { return (States)anim.GetInteger("state"); }
    set { anim.SetInteger("state", (int)value); }
}

private void Awake()
{
    sprite = GetComponentInChildren<SpriteRenderer>(); //обращение к спрайту
    anim = GetComponent<Animator>();
    rb = GetComponent<Rigidbody2D>();
}
```

get – получает значение из Аниматора

set – устанавливает

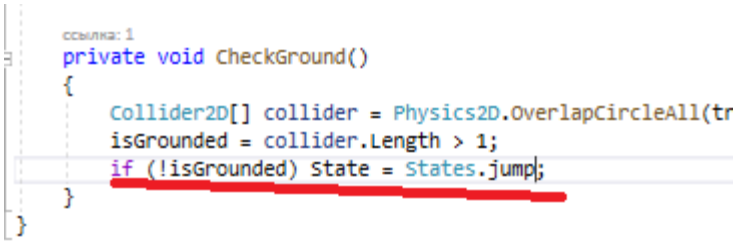
проверяем состояние покоя

```
    Сообщение Unity | Ссылок: 0
private void Update()
{
    if (isGrounded) State = States.idle;
    if (Input.GetButton("Horizontal"))
        Run();
    if (isGrounded && Input.GetButtonDown("Jump"))
```

проверяем состояние бега

```
ссылка: 1
private void Run()
{
    if (isGrounded) State = States.run;
    Vector3 dir = transform.right * Input.GetAxis("Hor
    transform.position = Vector3.MoveTowards(transform
    sprite.flipX = dir.x < 0.0f; // поворот глаз
}
    Сообщение Unity | Ссылок: 0
private void Update()
```

если не на земле, значит прыгаем



```
ссылка: 1
private void CheckGround()
{
    Collider2D[] collider = Physics2D.OverlapCircleAll(tr
    isGrounded = collider.Length > 1;
    if (!isGrounded) State = States.jump;
}
}
```

Проверяйте название аннимаций и списка состояний!!!!

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Hero : MonoBehaviour
{
    [SerializeField] private float speed = 3f; // скорость
    [SerializeField] private int lives = 5; // количество жизни
    [SerializeField] private float jumpForce = 15f; // сила прыжка

    private bool isGrounded = false;


    private Rigidbody2D rb;
    private Animator anim;
    private SpriteRenderer sprite;

    private States State
    {
        get { return (States)anim.GetInteger("state"); }
        set { anim.SetInteger("state", (int)value); }
    }

    private void Awake()
    {
        sprite = GetComponentInChildren<SpriteRenderer>(); //обращение к спрайту
        anim = GetComponent<Animator>();
        rb = GetComponent<Rigidbody2D>();
    }

    private void Run()
    {
        if (isGrounded) State = States.run;
        Vector3 dir = transform.right * Input.GetAxis("Horizontal");
        transform.position = Vector3.MoveTowards(transform.position, transform.position +
dir, speed * Time.deltaTime);
        sprite.flipX = dir.x < 0.0f; // поворот глаз
    }
    private void Update()
    {
        if (isGrounded) State = States.IDLE;
        if (Input.GetButton("Horizontal"))
            Run();
        if (isGrounded && Input.GetButtonDown("Jump"))
            Jump();
    }
    private void FixedUpdate()
    {
        CheckGround();
    }

    private void Jump()
    {
        rb.AddForce(transform.up * jumpForce, ForceMode2D.Impulse);
    }

    private void CheckGround()
    {
        Collider2D[] collider = Physics2D.OverlapCircleAll(transform.position, 0.3f);
        isGrounded = collider.Length > 1;
        if (!isGrounded) State = States.jump;
    }
}

public enum States

{
    IDLE,
    run,
    jump
}
```