Лабораторная работа № 25

Тема: разработка игрового проекта Bounce

Цель: приобрести навыки в разработке игрового проекта Bounce

Ход работы

1. Выполнение работы

Создание игрового проекта Bounce

1. Создал объект Ball: добавил Circle Collider 2D и RigidBody 2D: выставил Collisin Detection – Continious и Interpolate – InterPolate; добавил компонент скрипта Ball, прикрепил тэг Player. Создал дочерний New Sprite: прикрепил спрайт, изменил цвет. Выставил белый цвет Main Camera. Создал объект Platform: прикрепил тэг JumpingBlock, добавил компонент Box Collider 2D: выставил Offset – Y=0.26 и Size – X= 6.8, Y= 0.09; выставил Y= -5. В дочерний New Sprite: добавил спрайт Platform и Box Collider 2D, в котором выставил Offset – Y=-0.005 и Size – X=0.04, Y=0.03; выставил черный цвет.

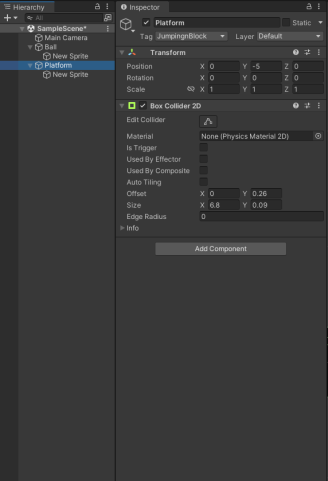
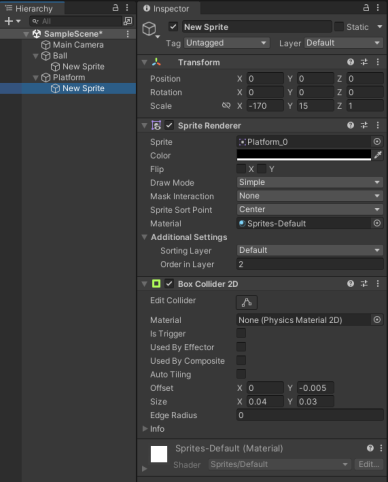
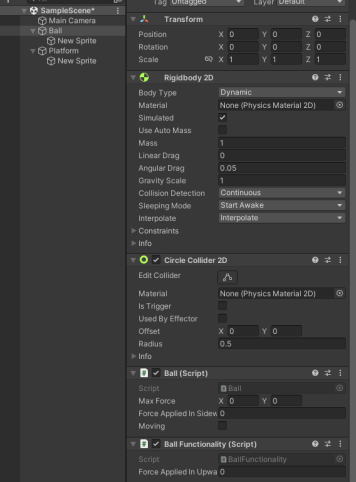


Рис. 25.1 Inspector объектов Ball, Platform и New Sprite

1. Листинг Ball.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Ball : MonoBehaviour

{ Rigidbody2D rgbd;

public Vector2 maxForce;

public float forceAppliedInSidewaysDirection;

public bool moving;

float sidewaysMovement;

BallFuctionality ballFuctionality;

// Start is called before the first frame update

void Start()

{

rgbd = GetComponent<Rigidbody2D>();

ballFuctionality = GetComponent<BallFuctionality>();

}

// Update is called once per frame

void Update()

{

// clampVeclocity();

// Debug.Log(rgbd.velocity);

movement();

}

public void FixedUpdate()

{

if (!moving)

{

Vector2 vel = rgbd.velocity;

vel.x = Mathf.Lerp(vel.x, 0, 0.2f);

rgbd.velocity = vel;

}

}

void clampVeclocity()

{

Vector2 vel = rgbd.velocity;

if (Mathf.Abs(vel.x) >= maxForce.x)

{

vel.x = maxForce.x \* Mathf.Sign(rgbd.velocity.x);

}

if ((vel.y) >= maxForce.y)

{ if (ballFuctionality.jumpHigher) return;

vel.y = maxForce.y;

}

rgbd.velocity = vel;

}

void movement()

{

if (Input.GetKey(KeyCode.LeftArrow))

{

rgbd.AddForce(new Vector2(-forceAppliedInSidewaysDirection, 0));

}

if (Input.GetKey(KeyCode.RightArrow))

{

rgbd.AddForce(new Vector2(forceAppliedInSidewaysDirection, 0));

}

if (Input.GetKeyDown(KeyCode.RightArrow) || Input.GetKeyDown(KeyCode.LeftArrow))

{

moving = true;

}

if (Input.GetKeyUp(KeyCode.RightArrow) || Input.GetKeyUp(KeyCode.LeftArrow))

{

moving = false;

}

}

}

1. Листинг BallFunctionality

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class BallFuctionality : MonoBehaviour

{

Rigidbody2D rgbd;

public float forceAppliedInUpwardDirection, higherForceAppliedInUpwardDirection;

public bool jumpHigher;

Manager manager;

public bool hasElectricity;

SpriteRenderer sr;

public Color electricitySprite;

Color originalColor;

Score score;

// Start is called before the first frame update

void Start()

{

rgbd = GetComponent<Rigidbody2D>();

sr = GetComponentInChildren<SpriteRenderer>();

score = FindObjectOfType<Score>();

originalColor = sr.color;

manager = FindObjectOfType<Manager>();

}

public void OnCollisionEnter2D(Collision2D collision)

{ if (!manager.startGame) return;

if (collision.collider.tag == "JumpingBlock")

{

if (jumpHigher)

{

Debug.Log("Applying higher force");

rgbd.AddForce(new Vector2(0, higherForceAppliedInUpwardDirection));

jumpHigher = false;

}

else

{

rgbd.AddForce(new Vector2(0, forceAppliedInUpwardDirection));

}

}

if (collision.collider.tag == "ElectricityBlock")

{ //gameover

if (hasElectricity) {

rgbd.AddForce(new Vector2(0, forceAppliedInUpwardDirection)); return; }

else

{

Debug.Log("Dead");

manager.RestartTheGame();

}

}

}

public void OnTriggerEnter2D(Collider2D collision)

{

if (!manager.startGame) return;

if (collision.tag == "spring")

{

jumpHigher = true;

Destroy(collision.gameObject);

}

if (collision.tag == "TimeSlower")

{

//call function to slow the game

manager.slowTimerStart();

Destroy(collision.gameObject);

}if (collision.tag == "TimeFaster")

{

//call function to slow the game

manager.FastTimerStart();

Destroy(collision.gameObject);

}if (collision.tag == "ElectricityPower")

{

//change the sprite to charge sprite

//screen flashes

Destroy(collision.gameObject);

StartCoroutine(electricity());

}

if (collision.tag == "PointObject")

{

//add point

if (!collision.GetComponent<PointsObject>().hasBroke)

{

score.AddScore();

}

collision.GetComponent<PointsObject>().Explode();

}

}

IEnumerator electricity()

{

hasElectricity = true;

sr.color = electricitySprite;

yield return new WaitForSeconds(2f);

hasElectricity = false;

sr.color = originalColor;

}

}

1. В компоненте RigidBody 2D объекта Ball выставил Angular Drug = 0, Gravity Scale = 2, Mass = 1, Force Applied InUpwaysDirection = 500 и включил Freeze Rotation. Создал объект MovingBlock: прикрепил тэг JumpingBlock, добавил компонент скрипта BlockScript: включил MoveBlock; добавил Box Collider 2D и RigidBody 2D, где выставил Kinematic и включил Freeze Rotation. Скопиовал MovingBlock, назвал FallingBlock и поменял цвет. У объектов Platform, MovingBlock и FallingBlock выставил порядок JumpingBlock. В окне Project Setting- Physics 2D отключил JumpingBlock.

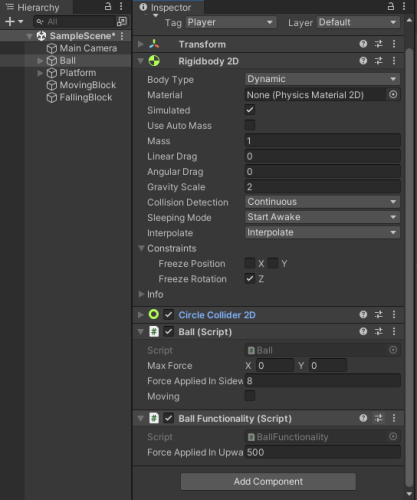
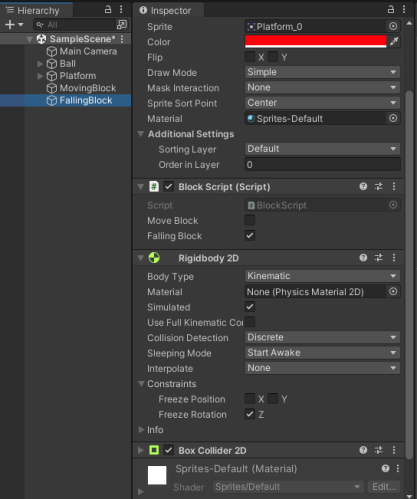
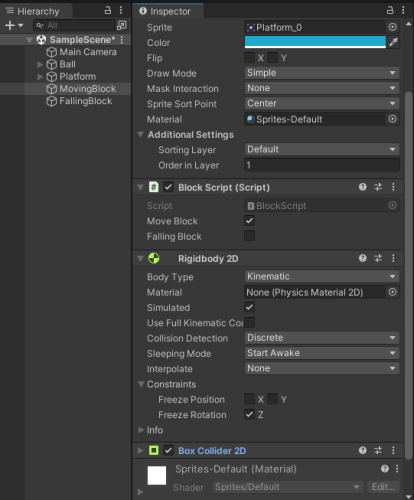


Рис. 25.2 Inspector объектов Ball, MoveBlock и FallingBlock

1. Листинг BlockScript.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class BlockScript : MonoBehaviour

{

public bool moveBlock, fallingBlock;

bool startMovingBlock, startfallingBlock;

Rigidbody2D rgbd;

bool moveLeft;

// Start is called before the first frame update

void Start()

{

rgbd = GetComponent<Rigidbody2D>();

if(Random.value > 0.5)

{

moveLeft = true;

}

}

// Update is called once per frame

void Update()

{

if(startMovingBlock && moveBlock)

{

if(moveLeft)

{

rgbd.velocity = new Vector2(-3, 0);

}

else

{

rgbd.velocity = new Vector2(3, 0);

}

}

}

private void OnCollisionEnter2D(Collision2D collision)

{

if(collision.collider.tag == "Player")

{

if(moveBlock)

{

startMovingBlock = true;

}

else if(fallingBlock)

{

rgbd.bodyType = RigidbodyType2D.Dynamic;

}

}

}

}

1. Настроил иерархию и сделал следование камеры за игроком

Листинг CameraController.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class CameraController : MonoBehaviour

{

public Transform player;

// Start is called before the first frame update

void Start()

{

}

// Update is called once per frame

void Update()

{

if (player.position.y > transform.position.y)

{

transform.position = new Vector3(transform.position.x, player.position.y, -10);

}

}

}

1. Листинг Manager.cs

using System.Collections;

using UnityEngine.SceneManagement;

using UnityEngine;

public class Manager : MonoBehaviour

{

public bool slowTime, fastTime;

public float slow;

public bool startGame;

public void slowTimerStart()

{

StartCoroutine(slowTheTime());

}

IEnumerator slowTheTime()

{

slowTime = true;

Time.timeScale = 1 / slow;

Time.fixedDeltaTime = Time.fixedDeltaTime / slow;

FindObjectOfType<BallFuctionality>().forceAppliedInUpwardDirection \*= 2;

yield return new WaitForSeconds(2 \* slow);

Time.timeScale = 1;

Time.fixedDeltaTime = Time.fixedDeltaTime \* slow;

FindObjectOfType<BallFuctionality>().forceAppliedInUpwardDirection /= 2;

slowTime = false;

}

public void FastTimerStart()

{

StartCoroutine(fastTheTime());

}

IEnumerator fastTheTime()

{

fastTime = true;

Time.timeScale = 1 \* slow;

Time.fixedDeltaTime = Time.fixedDeltaTime \* slow;

FindObjectOfType<BallFuctionality>().forceAppliedInUpwardDirection /= 2;

yield return new WaitForSeconds(2 \* slow);

Time.timeScale = 1;

Time.fixedDeltaTime = Time.fixedDeltaTime / slow;

FindObjectOfType<BallFuctionality>().forceAppliedInUpwardDirection \*= 2;

fastTime = false;

}

public void StartTheGame()

{

startGame = true;

FindObjectOfType<BallFuctionality>().GetComponent<Rigidbody2D>().bodyType = RigidbodyType2D.Dynamic;

}

public void RestartTheGame()

{

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);

}

}

1. Создал объект JumpHighter: добавил спрайт и Box Collider 2D, где включил Is Trigger, выставил цвет и прикрепил тэг Spring; создал дочерний объект, куда добавил спрайт Circle 6296\_3. Создал объект SlowMotion: добавил SpriteRenderer, куда добавил спарйт, и Box Collider 2D, где включил Is Trigger; и прикрепил тэг TimeSlower. Скопировал и переименовал в FastMotion с заменой тэга.

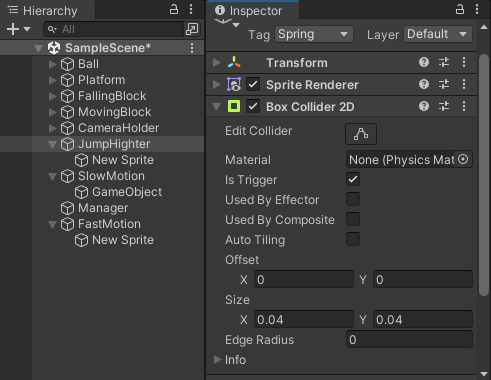
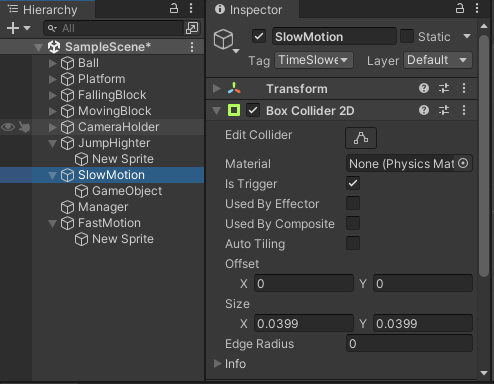
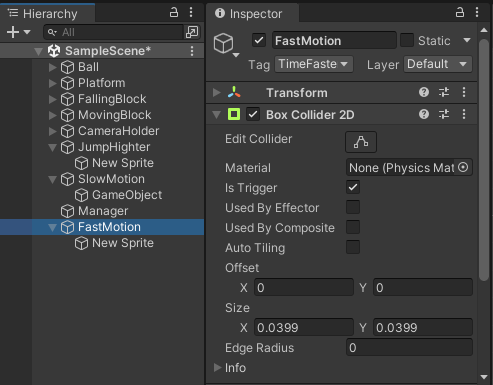


Рис. 25.3 Inspector объектов FastMotion, SlowMotion и JumpHighter

1. Создал объект TextMeshPro, дочерний объекту fastMotion: выставил масштаб, Font Size =60, выравнивание текста посередине и цвет черный, прописал текст FAST. Скопировал его для разных объектов с заменой текста. Создал объект ElectricPower: добавил Box Collider 2D, в котором включил Is Trigger, и SpriteRenderer, куда прикрепил спрайт. Скопировал TextMeshPro для разных объектов с заменой текста.

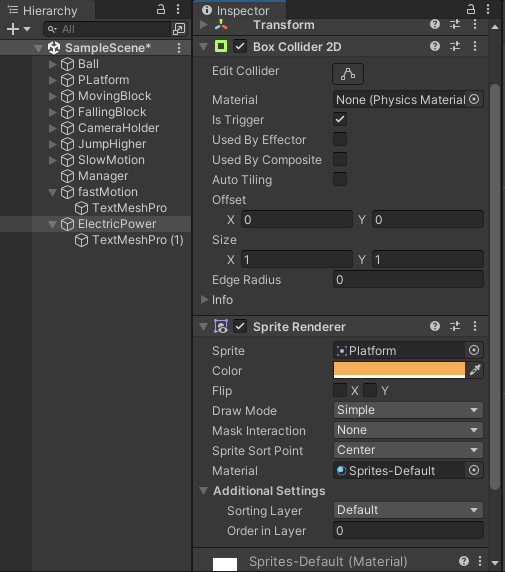
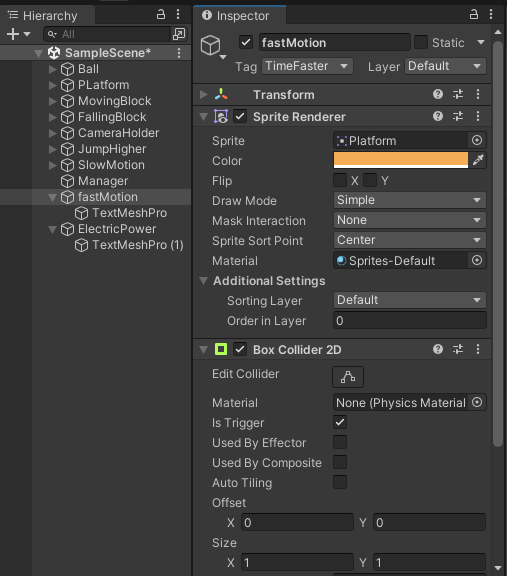


Рис. 25.4 Inspector объектов fastMotion и ElectricPower

1. Создал объект PointObject: в дочернем Line в объекте block добавил спрайт, уменьшив масштаб по Y, выставил голубой цвет, добавил Box Collider 2D. Скопировал объект block, расположив его на сцене в одну линию. В дочернем Line создал объект через 3D Object – TextMeshPro- Text: уменьшил размер, приписал текст, выставил выравнивание текста посередине и цвет как у объекта block. Добавил компонент скрипта PointsObject объекту Line: добавил объект +1Text в Points1, выставил Force = 40 и прикрепил тэг PointsObject. Добавил RigidBody 2D объекту +1Text, выставив Kinematic.

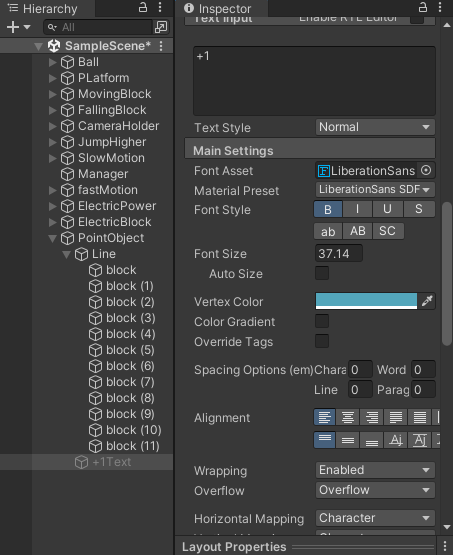
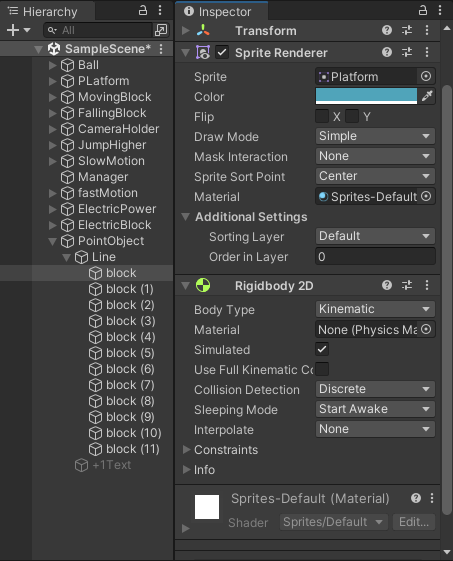
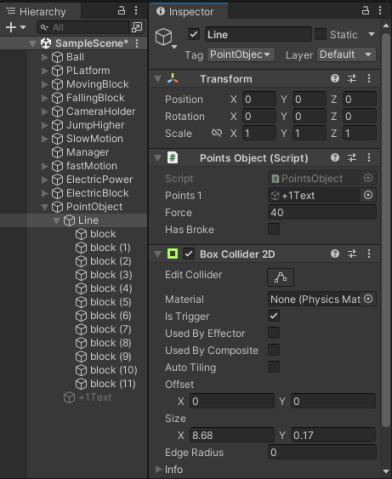


Рис. 25.5 Inspector объектов block, +1Text

1. Листинг PointsObject.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PointsObject : MonoBehaviour

{

Transform[] childObj;

public GameObject Points1;

public int force;

public bool hasBroke;

public void Explode()

{

hasBroke = true;

Points1.SetActive(true);

Points1.GetComponent<Rigidbody2D>().bodyType = RigidbodyType2D.Dynamic;

Points1.GetComponent<Rigidbody2D>().AddForceAtPosition(Vector2.up \* force \* 3 / 2, Points1.transform.position);

foreach (Transform t in transform)

{

Rigidbody2D rgbd = t.GetComponent<Rigidbody2D>();

if (rgbd != null)

{

//shake camera

rgbd.bodyType = RigidbodyType2D.Dynamic;

rgbd.AddForceAtPosition(Vector2.up \* force,

new Vector2(Random.Range(t.position.x - 2, t.position.x + 2), t.position.y));

//play sound

}

}

}

}

1. В PointObject создал ScoreCanvas и TextMeshPro-Text: прописал текст, выставил выравнивание посередине, цвет шрифта белый; создал объект Image, куда добавил спрайт Platform, выставил цвет бирюзовый. У этих объектов в окне Anchor Presets выставил выравнивание к левому верхнему углу. Прикрепил Score.cs объекту ScoreCanvas и добавил в Score Text - TextMeshPro-Text.

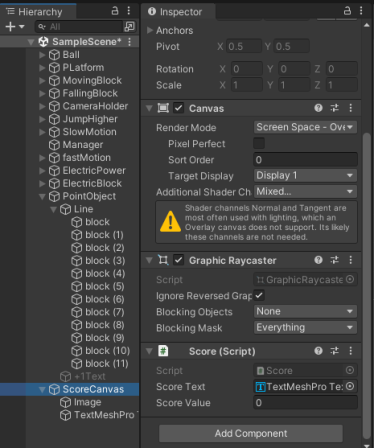
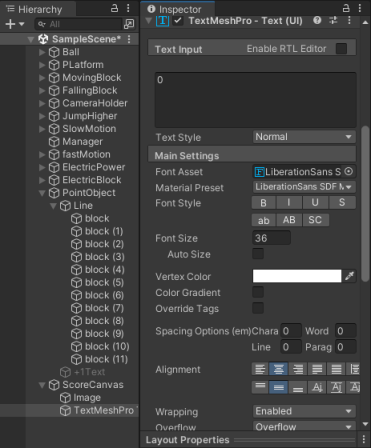
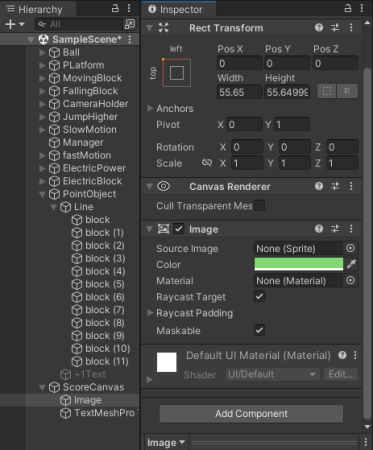


Рис. 25.6 Inspector объектов Text, Image и ScoreCanvas

1. Листинг Score.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Score : MonoBehaviour

{

public TMPro.TextMeshProUGUI scoreText;

public int ScoreValue;

public void AddScore()

{

ScoreValue++;

scoreText.text = ScoreValue.ToString();

}

}

1. Создал объект StartCanvas и настроил дочерние объекты.

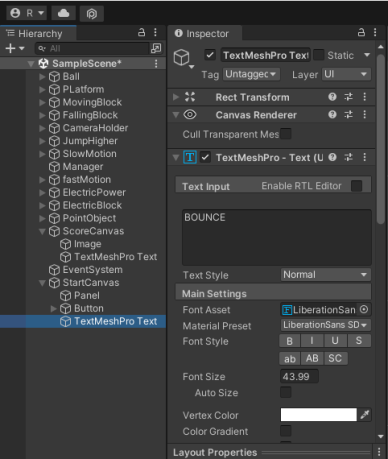
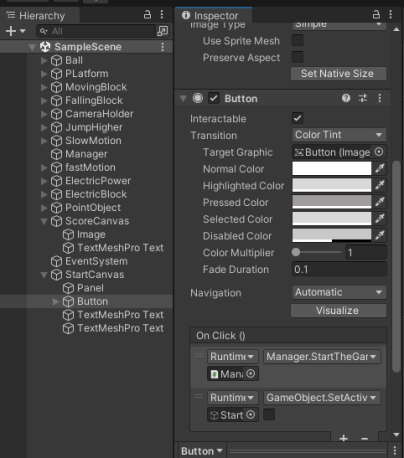
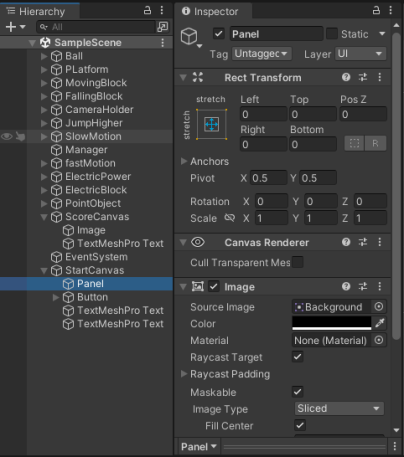


Рис. 25.7 Inspector объектов Panel, Button и TextMeshPro-Text

1. Вывод

В ходе проделанной работы были приобретены навыки в разработке игрового проекта Bounce.