Одеська національна академія харчових технологій
Кафедра комп’ютерної інженерії

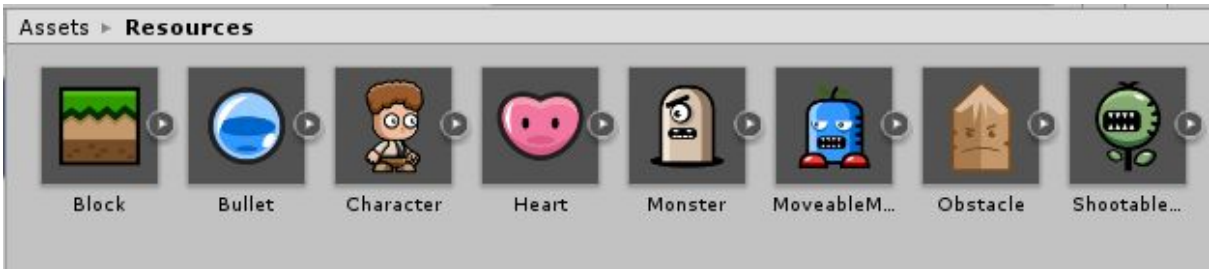
Лабораторна робота №3

з дисципліни «Проектування ігрових систем»

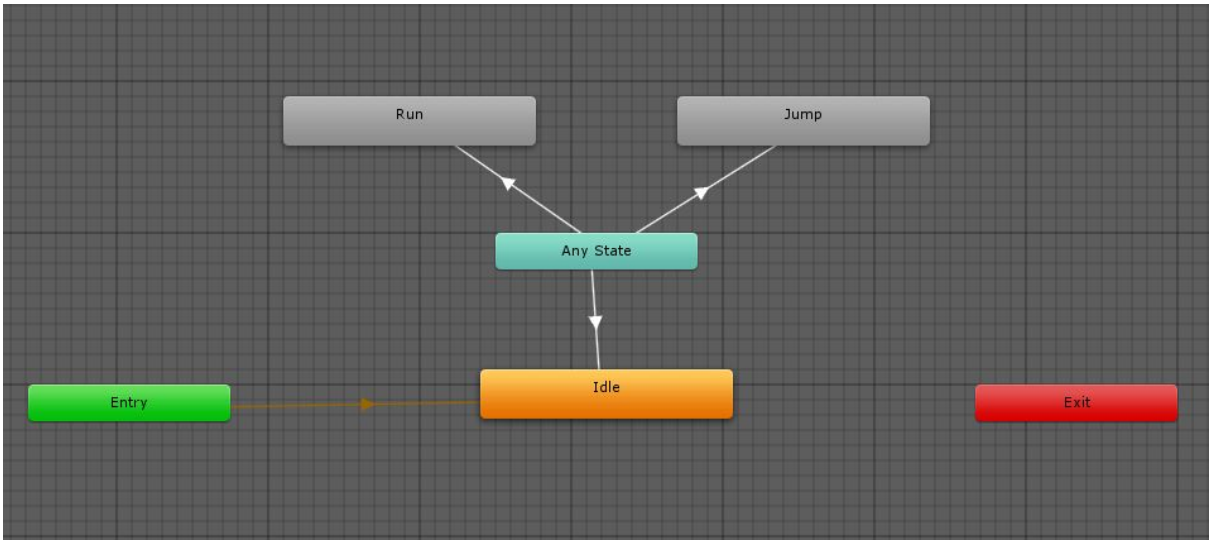
Виконав: студент групи 542
Погосян Михайла
Перевірив: Сіренко О.І.

Одеса 2017 р.

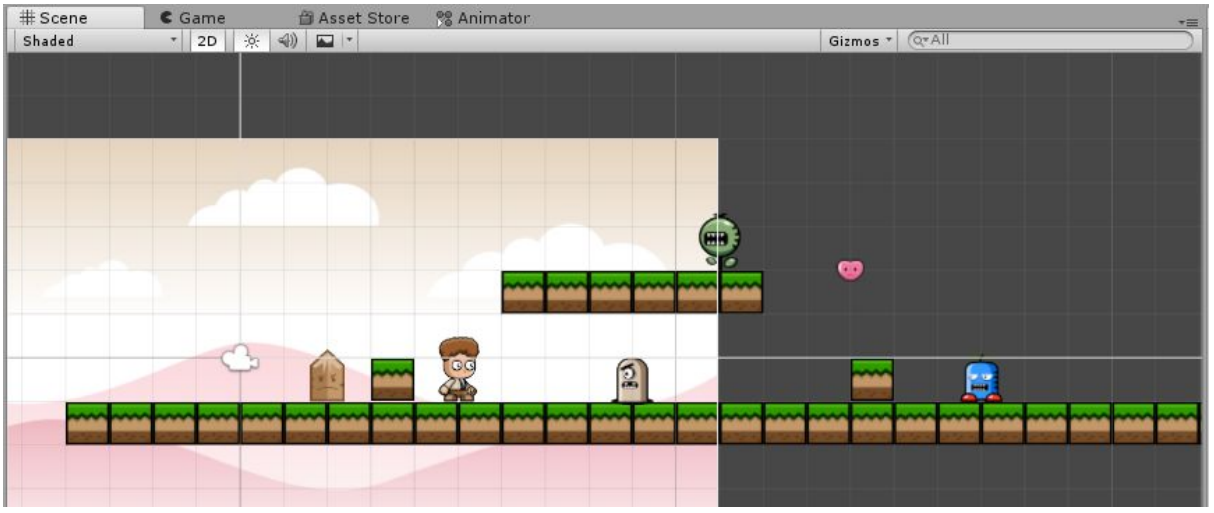
Спрайты для сцены:



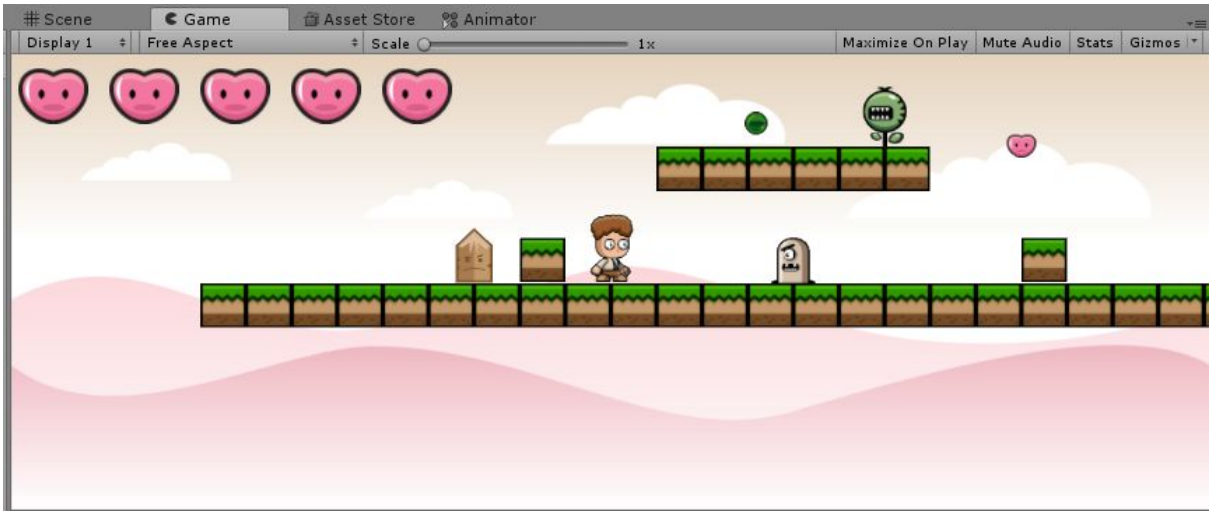
Анимация:



Финальная сцена:



Запущенная игра:



Скрипты:

Для героя:

```
using UnityEngine;
using System.Collections;

public class Character : Unit
{
[SerializeField]
private int lives = 5;

public int Lives
{
get { return lives; }
set
{
if (value < 5) lives = value;
livesBar.Refresh();
}
}
private LivesBar livesBar;

[SerializeField]
private float speed = 3.0F;
[SerializeField]
private float jumpForce = 15.0F;

private bool isGrounded = false;

private Bullet bullet;

private CharState State
{
get { return (CharState)animator.GetInteger("State"); }
set { animator.SetInteger("State", (int)value); }
}

new private Rigidbody2D rigidbody;
private Animator animator;
private SpriteRenderer sprite;

private void Awake()
{
livesBar = FindObjectOfType<LivesBar>();
rigidbody = GetComponent<Rigidbody2D>();
```

```
animator = GetComponent<Animator>();
sprite = GetComponentInChildren<SpriteRenderer>();

bullet = Resources.Load<Bullet>("Bullet");
}

private void FixedUpdate()
{
CheckGround();
}

private void Update()
{
if (isGrounded) State = CharState.Idle;

if (Input.GetButtonDown("Fire1")) Shoot();
if (Input.GetButton("Horizontal")) Run();
if (isGrounded && Input.GetButtonDown("Jump")) Jump();
}

private void Run()
{
Vector3 direction = transform.right * Input.GetAxis("Horizontal");

transform.position = Vector3.MoveTowards(transform.position, transform.position + direction, speed * Time.deltaTime);

sprite.flipX = direction.x < 0.0F;

if (isGrounded) State = CharState.Run;
}

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

private void Shoot()
{
Vector3 position = transform.position; position.y += 0.8F;
Bullet newBullet = Instantiate(bullet, position, bullet.transform.rotation) as Bullet;

newBullet.Parent = gameObject;
newBullet.Direction = newBullet.transform.right * (sprite.flipX ? -1.0F : 1.0F);
}

public override void ReceiveDamage()
{
Lives--;

rigidbody.velocity = Vector3.zero;
rigidbody.AddForce(transform.up * 8.0F, ForceMode2D.Impulse);

Debug.Log(lives);
}

private void CheckGround()
{
Collider2D[] colliders = Physics2D.OverlapCircleAll(transform.position, 0.3F);

isGrounded = colliders.Length > 1;

if (!isGrounded) State = CharState.Jump;
}

private void OnTriggerEnter2D(Collider2D collider)
{
```

```
Bullet bullet = collider.gameObject.GetComponent<Bullet>();
if (bullet && bullet.Parent != gameObject)
{
ReceiveDamage();
}
}
}


public enum CharState
{
Idle,
Run,
Jump
}
```
Для камеры:
```
using UnityEngine;
using System.Collections;

public class CameraController : MonoBehaviour
{
[SerializeField]
private float speed = 2.0F;

[SerializeField]
private Transform target;

private void Awake()
{
if (!target) target = FindObjectOfType<Character>().transform;
}

private void Update()
{
Vector3 position = target.position; position.z = -10.0F;
transform.position = Vector3.Lerp(transform.position, position, speed * Time.deltaTime);
}
}
```
Для монстра:
```
using UnityEngine;
using System.Collections;

public class Monster : Unit
{
protected virtual void Awake() { }
protected virtual void Start() { }
protected virtual void Update() { }

protected virtual void OnTriggerEnter2D(Collider2D collider)
{
Bullet bullet = collider.GetComponent<Bullet>();

if (bullet)
{
ReceiveDamage();
}

Character character = collider.GetComponent<Character>();

if (character)
{
character.ReceiveDamage();
}
}
```

```
}
```

Для движущегося монстра:

```
using UnityEngine;
using System.Collections;
using System.Linq;

public class MoveableMonster : Monster
{
[SerializeField]
private float speed = 2.0F;

private Vector3 direction;

private SpriteRenderer sprite;

protected override void Awake()
{
sprite = GetComponentInChildren<SpriteRenderer>();
}

protected override void Start()
{
direction = transform.right;
}

protected override void Update()
{
Move();
}

protected override void OnTriggerEnter2D(Collider2D collider)
{
Unit unit = collider.GetComponent<Unit>();

if (unit && unit is Character)
{
if (Mathf.Abs(unit.transform.position.x - transform.position.x) < 0.3F) ReceiveDamage();
else unit.ReceiveDamage();
}
}

private void Move()
{
Collider2D[] colliders = Physics2D.OverlapCircleAll(transform.position + transform.up * 0.5F + transform.right *
direction.x * 0.5F, 0.1F);

if (colliders.Length > 0 && colliders.All(x => !x.GetComponent<Character>())) direction *= -1.0F;
transform.position = Vector3.MoveTowards(transform.position, transform.position + direction, speed * Time.deltaTime);
}
}
```

Для монстра что стреляет:

```
using UnityEngine;
using System.Collections;

public class ShootableMonster : Monster
{
[SerializeField]
private float rate = 2.0F;
[SerializeField]
private Color bulletColor = Color.white;

private Bullet bullet;
```

```
protected override void Awake()
{
bullet = Resources.Load<Bullet>("Bullet");
}

protected override void Start()
{
InvokeRepeating("Shoot", rate, rate);
}

private void Shoot()
{
Vector3 position = transform.position; position.y += 0.5F;
Bullet newBullet = Instantiate(bullet, position, bullet.transform.rotation) as Bullet;

newBullet.Parent = gameObject;
newBullet.Direction = -newBullet.transform.right;
newBullet.Color = bulletColor;
}

protected override void OnTriggerEnter2D(Collider2D collider)
{
Unit unit = collider.GetComponent<Unit>();

if (unit && unit is Character)
{
if (Mathf.Abs(unit.transform.position.x - transform.position.x) < 0.3F) ReceiveDamage();
else unit.ReceiveDamage();
}
}
}
```

Для пенька:

```
using UnityEngine;
using System.Collections;

public class Obstacle : MonoBehaviour
{
private void OnTriggerEnter2D(Collider2D collider)
{
Unit unit = collider.GetComponent<Unit>();

if (unit && unit is Character)
{
unit.ReceiveDamage();
}
}
}
```