СОЗДАНИЕ ИГРОВОГО ПРИЛОЖЕНИЯ НА ANDROID В СРЕДЕ UNITY

Чиркун Артур Леонидович

Студент факультета математики и технологий программирования
УО «Гомельский государственный университет имени Ф. Скорины»

Игровое приложение представляет собой компьютерную программу, служащую для организации игрового процесса, связи с партнёрами по игре, или саму выступающую в качестве партнёра. Игровые приложения часто создаются на основе фильмов и книг. Специальные разработанные игры позволяют использовать игроков в научно-исследовательских работах. Составляющими компьютерной игры являются:

– сеттинг – это среда, в которой происходит действие игрового приложения; место, время и условия действия;

– геймпле́й – игровой процесс с точки зрения игрока;

– музыка в компьютерных и/или видеоиграх – это любые мелодии, композиции или саундтреки видеоигр.

В настоящем проекте осуществляется разработка игрового приложения в жанре "Mario", с использованием языка C# [1]и среды разработки Unity [2].

Основной чертой игрового процесса является прыгание по платформам, лазанье по лестницам, собирание предметов, обычно необходимых для завершения уровня.

Противники, всегда многочисленные и разнородные, обладают примитивным искусственным интеллектом, стремясь максимально приблизиться к игроку, перемещаясь по круговой дистанции или совершая повторяющиеся действия. Соприкосновение с противником обычно отнимает жизненные силы у героя или вовсе убивает его. Иногда противник может быть нейтрализован либо прыжком ему на голову, либо из оружия, если им обладает герой. Смерть живых существ обычно изображается упрощённо: существо исчезает или проваливается вниз за пределы экрана.

Ниже кратко опишем логику персонажа:

– задание очков здоровья, скорости передвижения и силы прыжка персонажа:

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

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

– метод для бега:

```
  private void Run()
  {
    Vector3direction = transform.right * Input.GetAxis
("Horizontal");
    trans-
form.position=Vector3.MoveTowards(transform.position, trans-
form.position + direction, speed * Time.deltaTime);
    sprite.flipX = direction.x < 0.0F;
    if (isGrounded) State = CharState.Run;
  }
```

– метод для прыжков:

```
  private void Jump()
  {
    rigidbody.AddForce(transform.up * jumpForce, Force
Mode2D.Impulse);
  }
```

– метод для стрельбы:

```
  private void Shoot()
  {
    Vector3 position = transform.position; position.y +=
0.8F;
    Bullet newBullet = Instantiate(bullet, position, bul-
let.transform.rotation) as Bullet;
    newBullet.Parent = gameObject;
    newBullet.Direction = newBullet.transform.right *
(sprite.flipX ? -1.0F : 1.0F);
  }
```

– метод для получения урона:

```
  public override void ReceiveDamage()
  {
    Lives--;
    rigidbody.velocity = Vector3.zero;
    rigidbody.AddForce(transform.up * 8.0F, ForceMode2D.
Impulse);
    Debug.Log(lives);
  }
```

– метод для проверки на поверхности персонаж или нет:

```
private void CheckGround()
{
  Collider2D[]  colliders  =  Physics2D.OverlapCircleAll
(transform.position, 0.3F);
  isGrounded = colliders.Length > 1;
  if (!isGrounded) State = CharState.Jump;
}
```

Далее создаются неподвижные и подвижные монстры.

Таким образом, были реализованы алгоритмы для создания игры в жанре " Mario ".

Литература

1 Шилдт, Г. Java 8. Полное руководство 9-е издание / Г. Шилдт М.: Вильямс, 2015. – 1376 с.

2 Hellman, E. Android Programming: Pushing the Limits / E. Hellman B.:Blin-Rite, 2014. – 423 с.