

# ANGRY PIGS

IMAGEN DIGITAL

PABLO FERNANDEZ GONZALEZ

# ¿Por qué?

- Popularidad
- Controles sencillos
- Cierre del juego en 2023



# Alcance

- Demo jugable con varios niveles disponibles
- Introducción a Unity y C#



Unity 6 | PG | Asset Store | ⚙

Scene

Center Local 1 2D

Hand Selection Move Rotate Scale

Scene Viewport

Game Viewport

Display 1 16:9 Aspect Scale 1x Play Focused

Hierarchy

Level1

- Cameras
- Global Light 2D
- Environment
- SlingShot
- GameManager
- Canvas
- EventSystem
- Blocks
- Piggie (1)

Inspector

GameManager

Tag Untagged Layer Default

Transform

Position X -0.0504414 Y -2.467463 Z 0

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

Game Manager (Script)

- GameManager
- Max Number Of Shots 3
- Seconds To Wait Before Dea... 3
- Restart Screen Object
- Sling Shot Handler
- Next Level Image

Sound Manager (Script)

- SoundManager

Project

- Plugins
- Prefab
- Blocks
- Pigs
- PiggieDeathParticles
- Red
- Resources
- Scenes
- Level1
- Level2
- Level3
- Level4
- LevelS
- Scripts
- AngryBird
- CameraManager
- GameManager

Add Component

# Herramientas

# Tirachinas



```
// Update: Función ejecutada cada frame
0 references
private void Update()
{
    // Si se pulsa en el área con clic izquierdo, suena el sonido de la goma y se actualiza la posición de la cámara
    if (InputManager.WasLeftMouseButtonPressed && _slingshotArea.IsWithinSlingshotArea())
    {
        _clickedWithinArea = true;

        if (_birdOnSlingshot)
        {
            SoundManager.instance.PlayClip(_elasticPulledClip, _audioSource);
            _cameraManager.SwitchToFollowCam(_spawnedAngryBird.transform);
        }
    }

    // Si se mantiene el clic izquierdo, se ha pulsado inicialmente en el área y hay un pájaro en el tirachinas,
    // se dibuja su posición y la orientación del pájaro
    if (InputManager.IsLeftMousePressed && _clickedWithinArea && _birdOnSlingshot)
    {
        DrawSlingshot();
        PositionAndRotateAngryBird();
    }

    // Si se levanta el clic izquierdo y se cumplen las condiciones anteriores, se lanza el pájaro,
    // se reproduce un sonido y se renderizan las nuevas líneas del tirachinas. En caso de quedar pájaros,
    // se instancian en el tirachinas
    if (InputManager.WasLeftMouseButtonReleased && _birdOnSlingshot && _clickedWithinArea)
    {
        if (GameManager.instance.HasEnoughShots())
        {
            _clickedWithinArea = false;
            _birdOnSlingshot = false;

            _spawnedAngryBird.LaunchBird(_direction, _shotForce);

            SoundManager.instance.PlayRandomClip(_elasticReleasedClips, _audioSource);

            GameManager.instance.UseShot();
            //Setlines(_centerPosition.position); ---- Utilizada antes de la creación de la función AnimateSlingshot
            AnimateSlingshot();

            if (GameManager.instance.HasEnoughShots())
            {
                StartCoroutine(SpawnAngryBirdAfterTime());
            }
        }
    }
}
```

# Pájaro



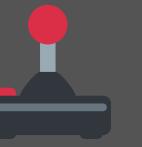
```
// LaunchBird: Cambian las físicas del pájaro y se activa su colisionador.  
// Se añade una fuerza a su lanzamiento  
0 references  
public void LaunchBird(Vector2 direction, float force)  
{  
    ... _rb.bodyType = RigidbodyType2D.Dynamic;  
    ... _circleCollider.enabled = true;  
  
    ... _rb.AddForce(direction * force, ForceMode2D.Impulse);  
  
    ... _hasBeenLaunched = true;  
    ... _shouldFaceVelDirection = true;  
}  
  
// OnCollisionEnter2D: Al chocar con algo, deja de orientarse a una posición definida,  
// se reproduce un sonido y se destruye su instancia (NO DEL JUEGO)  
0 references  
private void OnCollisionEnter2D(Collision2D collision)  
{  
    ... _shouldFaceVelDirection = false;  
    SoundManager.instance.PlayRandomClip(_hitClips, _audioSource);  
    Destroy(this);  
}
```

# Cerdo



```
// DamagePiggie: decremente la salud del cerdo en función del daño realizado  
1 reference  
public void DamagePiggie(float damageAmount)  
{  
    _currentHealth -= damageAmount;  
  
    if (_currentHealth <= 0f)  
    {  
        Die();  
    }  
}  
  
// Die: "Mata" al cerdo (borra su instancia)  
1 reference  
private void Die()  
{  
    GameManager.instance.RemovePiggie(this);  
  
    Instantiate(_piggieDeathParticles, transform.position, Quaternion.identity);  
  
    AudioSource.PlayClipAtPoint(_deathClip, transform.position);  
    Destroy(gameObject);  
}  
  
// OnCollisionEnter2D: si el cerdo recibe una colisión, se calcula la velocidad del impacto  
// en caso de superar el umbral de daño, decrementa la salud del cerdo  
0 references  
private void OnCollisionEnter2D(Collision2D collision)  
{  
    float impactVelocity = collision.relativeVelocity.magnitude;  
    if (impactVelocity > _damageThreshold)  
    {  
        DamagePiggie(impactVelocity);  
    }  
}
```

# Control del juego



```
// WinGame: se desactiva el tirachinas y aparece la interfaz de reinicio y siguiente nivel, si lo hay
2 references
private void WinGame()
{
    _restartScreenObject.SetActive(true);
    _slingShotHandler.enabled = false;

    int currentSceneIndex = SceneManager.GetActiveScene().buildIndex;
    int maxLevels = SceneManager.sceneCountInBuildSettings;
    if (currentSceneIndex + 1 < maxLevels)
    {
        _nextLevelImage.enabled = true;
    }
}

// RestartGame: se reinicia el nivel
1 reference
public void RestartGame()
{
    DOTween.clear(true);
    SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);
}

// RestartGame: se carga el siguiente nivel
0 references
public void NextLevel()
{
    SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
}
```

```
// Awake: detección del gestor de iconos, almacenamiento de cerdos en escena
// y pestaña de cambio de nivel desactivada
0 references
private void Awake()
{
    // Patrón singleton
    if (instance == null)
    {
        instance = this;
    }

    _iconHandler = FindFirstObjectByType<IconHandler>();
    Piggy[] piggies = Object.FindObjectsOfType<Piggy>(FindObjectsSortMode.None);
    for (int i = 0; i < piggies.Length; i++)
    {
        _piggies.Add(piggies[i]);
    }

    _nextLevelImage.enabled = false;
}

// UseShot: actualización del número de lanzamientos y de la UI.
// Se comprueba si es el último tiro
0 references
public void UseShot()
{
    _usedNumberOfShots++;
    _iconHandler.UseShot(_usedNumberOfShots);
    CheckForLastshot();
}
```



# DFIMO

# ¡PRÓXIMAMENTE!

- Más pájaros
- Selección de niveles
- Destruir bloques
- Cajas TNT
- En resumen... recrear el juego

**¡GRACIAS POR VER!**

