Animaciones

Animaciones sencillas

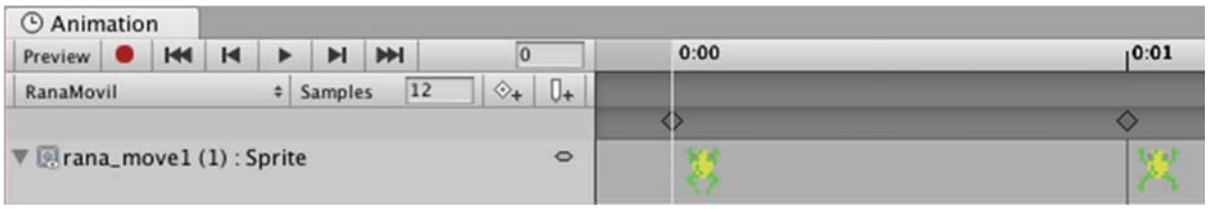
Sprites de la animación

- Localizamos los *frames* del *sprite* que se corresponden con la animación que nos interesa

- Los seleccionamos todos y los arrastramos a la escena
- Unity nos pedirá que pongamos nombre y sitio para guardar la animación en los recursos del proyecto
  - Guardamos la animación en la carpeta *Animations* de *Assets*

Creamos una animación

- Esto ha creado un *game object* con un componente *Animator Controller*
  - El controlador de la animación se ha creado junto con la animación
  - Esta sencilla acción genera un objeto animado
- Podemos abrir el **panel de animación** y ajustar la velocidad de la animación, reduciendo el valor de *Samples* para que vaya más lenta



Aspect Ratio

Espacio de juego

- Nuestro tablero tiene un tamaño de 24 x 29 en un mundo de altura 32
  - Puede dejarse espacio para el HUD, bien arriba, bien a los lados o en todas partes
    - Dejaremos 3 filas arriba y 3 columnas a cada lado
  - Usamos *sprites* que ocupan 8 píxeles por unidad
  - **Creamos** un ***aspect ratio*** para el juego de (30*8*2) x (32*8*2)
    - *Label: RompeMoldes*
    - *Type: Fixed resolution*
    - *Width & Heigth:* **480 x 512**
  - De esta forma acoplamos el ancho del mundo al elegido

Otras alternativas

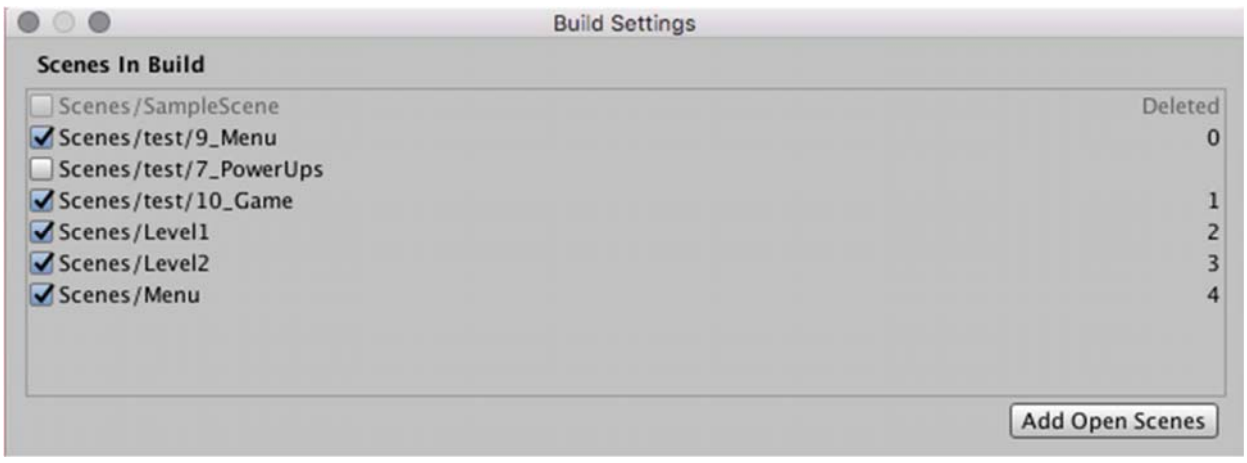
- Podríamos no cambiar el *aspect ratio* y dejar uno estándar
  - El espacio de juego seguiría siendo igual
  - Pero tendríamos las típicas franjas a los lados, al ser el espacio del mundo mayor que el de juego
- Para diseñar la UI es conveniente fijar un tamaño del mundo
  - Así el tamaño del texto y el de las imágenes está ajustado al tamaño elegido

Gestión de escenas

Carga de escenas

- Para poder cargar las escenas en ejecución es necesario añadirlas a las escenas del *Build*

  File >> Build Settings >> Scenes in Build



Carga de escenas

- En los *scripts* que necesiten gestión de escenas hay que importar la librería SceneManagement de Unity

  using UnityEngine.SceneManagement;

- La **carga de una escena** de nombre sceneName se realiza con la instrucción

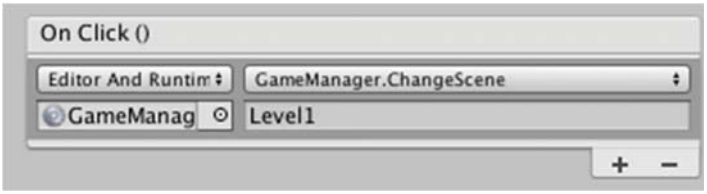
  SceneManager.LoadScene (sceneName, LoadSceneMode.Single);

- El **nombre de la escena activa** se puede conseguir con

  SceneManager.GetActiveScene().name

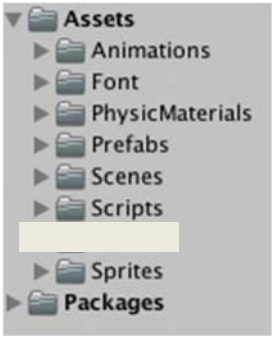
Método asociado a un botón

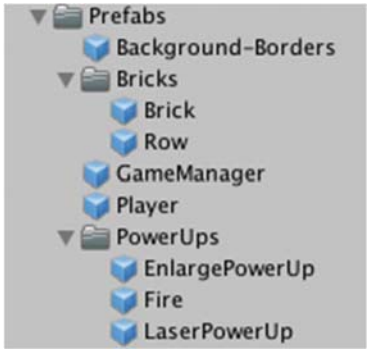
- La acción que nos interesa al pulsar un botón es la de cargar una nueva escena
  - Aprovechamos la omnipresencia del *GameManager,* que ya será un *prefab* a estas alturas, para elegirlo en la configuración de la propiedad *OnClick ()* del botón
  - De entre sus métodos, elegimos el que nos interesa: el método público ChangeScene (string sceneName)
  - Y configuramos el parámetro en el espacio sobrante: el nombre de la escena que queremos cargar

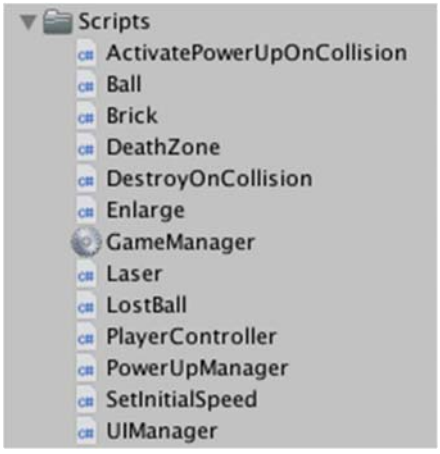


Mini-guía resumen

Recursos del proyecto

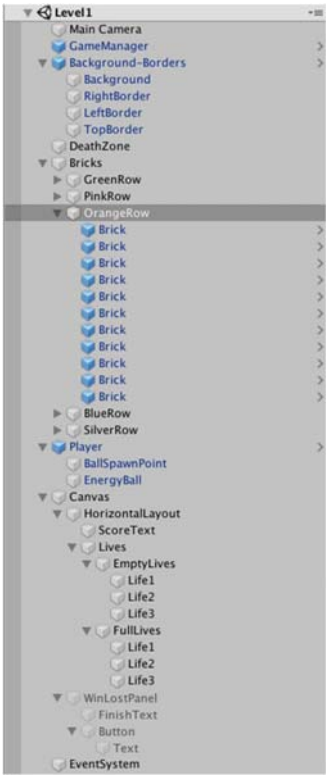




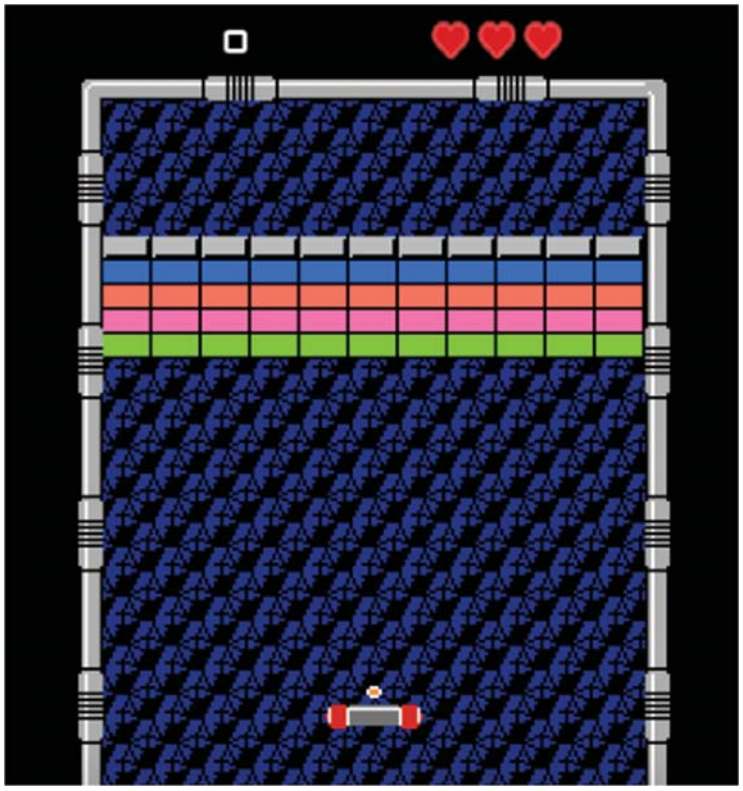


No todos los *prefabs* mostrados son necesarios

Ejemplo de nivel de juego



Ejemplo de nivel de juego



Resumen

- Nombre de clase
  - Nombres de variables **configurables** desde el Inspector
  - Nombres de métodos **públicos**

- Nada que tocar en
  - ActivatePowerUpOnCollision
    - string powerUpName
  - PowerUpManager
    - void ActivatePowerUp (string powerUpName)

Resumen

- Ball
  - float speed
- Brick
- DeathZone
- DestroyOnCollision
  - int golpesAntesDeMorir
  - int puntosGanados
  - GameObject dieObject
- Enlarge
- Laser
  - GameObject fire
  - Sprite vausLaserPower
- LostBall
  - Transform spawnPoint
  - void OnLost ()
- PlayerController
  - float velocityScale
  - float HitFactor (ballPos)
- SetInitialSpeed
  - Vector2 initialVelocity

GameManager

- void AddBrick ()
- void AddPoints (int amount)
- void BrickDestroyed ()
- void ChangeScene (string sceneName)
- void LevelFinished (bool playerWins)
- bool PlayerLoseLife ()
- void SetUIManager (UIManager uim)

UIManager

- Text scoreText, finishText
- GameObject finishPanel
- Image[] lives

- void FinishGame (bool playerWins)
- void LifeLost ()
  - Puede llevar o no parámetro de tipo int, en función de cómo lo implementéis
- void RemainingLives (int remainingLives)
- void UpdateScore (int points)

22/11

Semana 1

- Presentación del juego
  - Física 2D
  - Tablero con bordes
  - Jugador, bola, ladrillos
  - …
- *Scripts*
  - PlayerController
  - SetInitialSpeed
  - DeathZone
  - DestroyOnCollision
  - GameManager

29/11

Semana 2

- Enunciado provisional: ordena el desarrollo
- Escenas de *test* para comprobar funcionamiento por separado
  1. Importación correcta de *sprites* + capas de ordenación
  2. Movimiento de la pala – *PlayerController*
  3. Movimiento de la bola – **Ball**
     - Dos personalidades: cinemática con punto de *respawn*, física
  4. *DeathZone* – *DeathZone*, **LostBall**
  5. Ladrillos – *DestroyOnCollision*
  6. *GameManager* – *GameManager*, **Brick**
  7. *Power-ups* – *SetInitialSpeed*, **Enlarge**, **Laser**

Resto de semanas

- **2/12, 4/12** — Semana 3
  - Capas de colisión, mejora del movimiento de la bola
  - UI (I)
- **11/12** — Semana 4
  - 8/12 Resto enunciado con UI y juego completo
  - UI (II)
  - Implementación de *power-ups*, escala con Enlarge
- **13/12**
  - Laboratorio para temas de UI
- **16/12, 18/12, 19/12** para cerrar flecos — Semana 5
  - Gestión de escenas, animaciones sencillas, *aspect ratio*