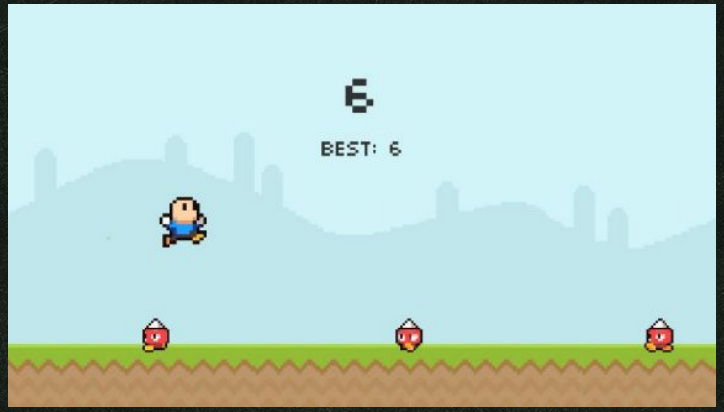
Primer Videojuego en Unity®: Jumping Guy

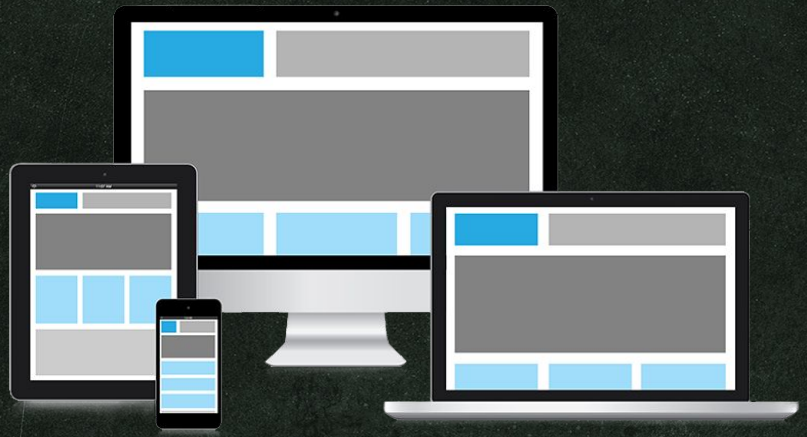
por Héctor Costa Guzmán · hektorprofe.net

Qué aprenderás



- A realizar diseños multiplataforma.
- A generar efectos parallax en capas.
- A gestionar animaciones y colisiones.
- A generar objetos dinámicamente.
- A manipular de ritmo de juego.
- A manejar la interfaz y un marcador de puntos.
- A guardar la puntuación máxima con PlayerPrefs.
- A reproducir música y efectos de audio de forma flexible.
- A exportar el videojuego a Windows, WebGL y Android.
- A programar el videojuego de forma modular, reutilizable y escalable.

Diseño multiplataforma

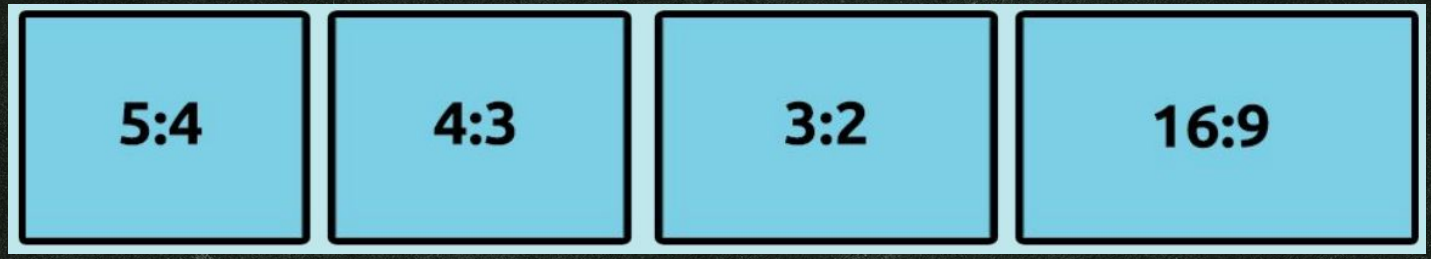


¿Cómo logramos que un juego se vea bien en el máximo de dispositivos?

Relación de aspecto

Cálculo que relaciona el ancho y alto de una imagen:

Relación	Valor	Descripción
5/4	1.25	Estándar en monitores de ordenador prácticamente cuadrados.
4/3	1.33	Estándar del formato PAL para televisores, pantallas de ordenador y iPads.
3/2	1.5	Estándar del formato NTSC para televisores y algunos smartphones.
16/9	1.77	Estándar de pantallas HD, conocido como formato panorámico o widescreen.



Resolución

¿Cómo elegir una? En Unity importa más la relación que la resolución.

Nombre	Resolución
Full HD	1920 x 1080
HD+	1600 x 900
HD	1280 x 720
qHD	960 x 540
nHD	640 x 360

Resoluciones 16/9

Zona segura

Espacio del escenario visible en cualquier dispositivo:

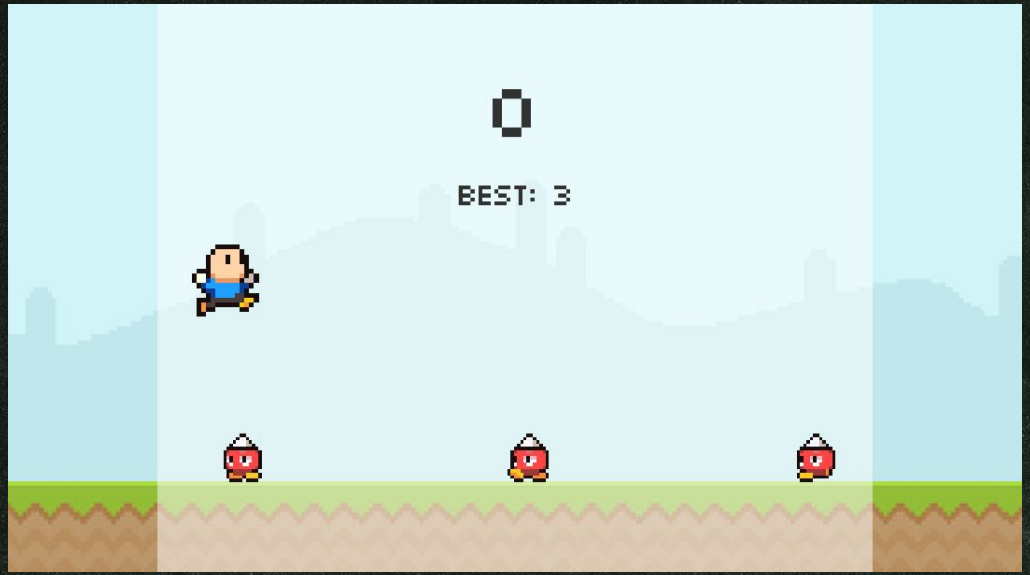
*Ancho = Alto * Relación aspecto mínima*

Resolución	Zona Segura
1920 x 1080	1350 x 1080
1600 x 900	1125 x 900
1280 x 720	900 x 720
960 x 540	675 x 540
640 x 360	450 x 360

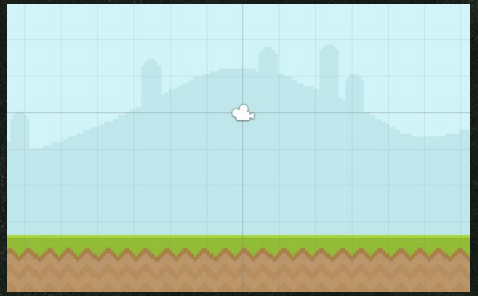
Resoluciones 16/9

Diseño del juego

Los elementos importantes se encontrarán siempre en la zona segura:



Creando la escena



¿Qué vamos a hacer?

- Crear el proyecto y configurar la escena.
- Maquetar el diseño en un *Canvas* con *Raw Images*.
- Limitar la zona segura con un *Panel*.

Objetivo

- Preparar el escenario para el efecto parallax.

Efecto Parallax

¿Qué vamos a hacer?

- Preparar las imágenes para usarlas como texturas.
- Aprender sobre la propiedad *UV Rect* de las *Raw Images*.
- Desarrollar la base de nuestro sistema de juego.
- Modificar la propiedad *UV Rect* desde un script.

Objetivo

- Conseguir un doble efecto de movimiento en el fondo.

Portada animada e inicio

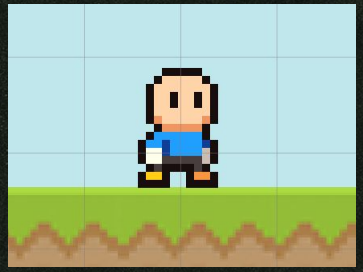
¿Qué vamos a hacer?

- Controlar el efecto Parallax de forma dinámica.
- Crear una estructura para controlar los estados del juego.
- Crear una interfaz con un título y una descripción animada.
- Cambiar entre los estados del juego y esconder la interfaz.

Objetivo

- Mostrar la interfaz e iniciar el juego al presionar una tecla o botón.

Creando al protagonista



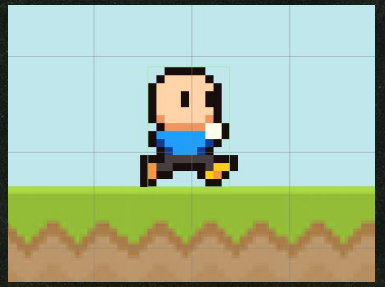
¿Qué vamos a hacer?

- Preparar los sprites del personaje.
- Crear un objeto para el personaje.
- Posicionarlo y darle el tamaño correcto.

Objetivo

- Preparar nuestro personaje para programar su sistema de control.

Animación de correr



¿Qué vamos a hacer?

- Añadir y configurar las animaciones y el animador.
- Desarrollar nuestro propio sistema de control del jugador.
- Llamar a los métodos del sistema desde otros scripts.

Objetivo

- Que el personaje esté parado y al iniciar el juego cambie a correr.

Animación de saltar

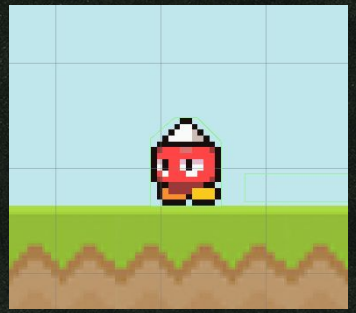
¿Qué vamos a hacer?

- Añadir una nueva animación.
- Controlar la animación dinámicamente.
- Manejar las transiciones entre las animaciones.

Objetivo

- Que el personaje pueda saltar.

Creando al enemigo



¿Qué vamos a hacer?

- Añadir y configurar un nuevo objeto.
- Escalar correctamente el tamaño del sprite.
- Crear su animación inicial y otorgarle movimiento.

Objetivo

- Crear un enemigo animado que se mueva en una dirección.

Generador de enemigos

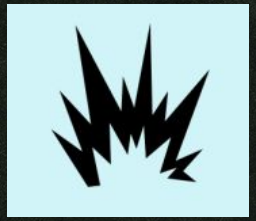
¿Qué vamos a hacer?

- Desarrollar nuestro propio sistema generador de enemigos.
- Utilizar la función *Instantiate* para crear enemigos dinámicamente.
- Utilizar la función *InvokeRepeating* para iniciar el generador.
- Utilizar la función *CancelInvoke* para detener el generador.

Objetivo

- Generar enemigos automáticamente cada cierto tiempo.

Autodestruir enemigos



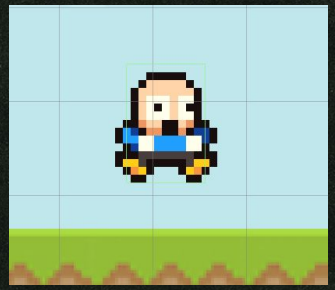
¿Qué vamos a hacer?

- Programar la autodestrucción de los enemigos.

Objetivo

- Liberar la memoria para evitar copias masivas de objetos.

Animación de muerte



¿Qué vamos a hacer?

- Crear la animación y la colisión contra los enemigos.
- Configurar un nuevo estado de juego finalizado.
- Cancelar el generador de enemigos.

Objetivo

- Que el personaje muera al tocar un enemigo y finalice el juego.

Reinicio y cierre de juego

¿Qué vamos a hacer?

- Programar una forma de reiniciar la escena al morir.
- Programar una forma de salir del juego en cualquier momento.

Objetivo

- Permitir al jugador reiniciar y salir.

Dificultad progresiva

¿Qué vamos a hacer?

- Desarrollar nuestro propio sistema de control de velocidad.
- Introducir la propiedad de control de velocidad *Time.timeScale*.
- Crear métodos para iniciar, incrementar y reiniciar la velocidad.

Objetivo

- Aumentar la dificultad del juego cada cierto tiempo.

Marcador de puntos

¿Qué vamos a hacer?

- Desarrollar nuestro propio sistema de control de puntuación.
- Duplicar la capa *UI Idle* para gestionar los datos del marcador.
- Incrementar los puntos con un *trigger* en los enemigos.
- Control la visibilidad del marcador dinámicamente.

Objetivo

- Mostrar el marcador de puntos e incrementarlo al saltar enemigos.

Guardado del récord

¿Qué vamos a hacer?

- Introducir el sistema de datos persistentes *PlayerPrefs*.
- Crear métodos para recuperar, guardar y visualizar el récord en la UI.

Objetivo

- Almacenar el récord de puntos entre partidas y mostrarlo.

Música y efectos de audio

¿Qué vamos a hacer?

- Crear nuestro propio sistema de control de música y efectos.
- Introducir el componente *AudioSource* para reproducir audio.
- Configurar todos los *AudioClips* de forma extensible y reutilizable.
- Llamar a nuestro sistema en los momentos que lo necesitemos.

Objetivo

- Reproducir música y efectos de sonido del juego.

Exportación multiplataforma

¿Qué vamos a hacer?

- Configurar y exportar el juego para *Windows*.
- Configurar y exportar el juego para *WebGL*.
- Configurar y exportar el juego para *Android*.

Objetivo

- Conseguir los distribuibles para cada plataforma.