**Alex Jiménez** **M09-UF3** **21/04/2025**
**Pau Alcaraz**

# Juego Multijugador Unity

A part d'entregar el projecte, entrega un document de memòria amb aquesta informació:

## 1. Links Juegos

- Link GitHub: https://github.com/userAl3x/JuegoMultijugadorUnity
- Link Juegos Drive: https://drive.google.com/drive/folders/1HI4A9k9VGkUaLxNvP-FRrmCFwV31Cog_?usp=sharing

## 2. Identifica quina informació cal enviar

- Nombres de los jugadores (usando **PhotonNetwork.NickName**)
- Posición y movimiento de los jugadores (usando **Rigidbody2D** y **PhotonView**)
- Contador de monedas (variable coins en PlayerMovement.cs)
- Información de las salas (**nombres de sala mediante createInput y joinInput**)

## 3. Quins mètodes s'han d'utilitzar per enviar i rebre la informació

Para enviar:

- Para sincronizar nombres de jugadores: **view.RPC("SetPlayerName", RpcTarget.AllBuffered, PhotonNetwork.NickName**)
- Para actualizar la UI de monedas: **view.RPC("UpdateCoinUI", RpcTarget.AllBuffered)**
- Movimiento del jugador mediante **Rigidbody2D** sincronizado con **PhotonView**

Para recibir:

- Para recibir y mostrar nombres de jugadores: **[PunRPC] void SetPlayerName(string name)**
- Para manejar cuando un jugador se une a una sala: **OnJoinedRoom()**
- Para detectar colisiones con monedas: **OnTriggerEnter2D**

Nuestro juego utiliza un sistema de salas donde:

- Los jugadores pueden crear salas (CreateRoom())
- Los jugadores pueden unirse a salas existentes (JoinRoom())
- Los nombres de los jugadores se guardan en PlayerPrefs
- Cada jugador tiene su propia cámara que solo se activa para el jugador local

## 4. Funcionamiento del Juego Multijugador

1. **Sistema de Nombres de Jugadores**:
   - Cuando un jugador inicia el juego, puede introducir su nombre en un campo de texto (nameInput)
   - Este nombre se guarda en PlayerPrefs para recordarlo en futuras sesiones
   - El nombre se asigna a PhotonNetwork.NickName para que todos los jugadores lo vean
   - Cuando un jugador se une, su nombre se sincroniza con todos usando SetPlayerName a través de RPC

2. **Sistema de Salas**:
   - Los jugadores pueden crear nuevas salas escribiendo un nombre en createInput
   - También pueden unirse a salas existentes escribiendo el nombre en joinInput
   - Cuando un jugador se une a una sala, automáticamente se carga la escena del juego (SampleScene)
   - Cada sala puede tener múltiples jugadores interactuando entre sí

3. **Sistema de Movimiento del Jugador**:
   - Cada jugador tiene un Rigidbody2D para el movimiento físico
   - El movimiento se controla con las teclas WASD o flechas
   - Solo el jugador local puede controlar su personaje (verificado con view.IsMine)
   - El movimiento se sincroniza automáticamente con otros jugadores a través de PhotonView

4. **Sistema de Cámaras**:
   - Cada jugador tiene su propia cámara
   - La cámara solo se activa para el jugador local (playerCamera.SetActive(true))
   - Las cámaras de otros jugadores se mantienen desactivadas
   - Esto evita que veas el juego desde la perspectiva de otros jugadores

5. **Sistema de Monedas**:
   - Los jugadores pueden recoger monedas que tienen el tag "Coin"
   - Cuando un jugador toca una moneda, su contador (coins) aumenta
   - La UI se actualiza para mostrar las monedas recolectadas
   - La actualización de la UI se sincroniza con todos los jugadores usando RPC

6. **Sistema de Sincronización**:
   - Se usa PhotonView para mantener sincronizados los objetos importantes
   - Los RPCs se usan para eventos específicos como actualizar nombres o monedas
   - El movimiento se sincroniza automáticamente a través del sistema de red
   - Se verifica constantemente view.IsMine para asegurar que cada jugador solo controla su personaje

7. **Flujo del Juego**:
   1. El jugador introduce su nombre
   2. El jugador crea o se une a una sala
   3. Se carga la escena del juego
   4. El jugador aparece en el mundo con su nombre sobre su cabeza
   5. El jugador puede moverse y recoger monedas
   6. Todos los jugadores ven las acciones de los demás en tiempo real

8. **Características de Seguridad**:
   - Los nombres se validan antes de ser asignados
   - Solo el jugador local puede controlar su personaje
   - Las acciones importantes se verifican con view.IsMine
   - Los datos se sincronizan de manera segura a través de Photon

Esta implementación permite que múltiples jugadores interactúen en el mismo mundo, vean los movimientos de los demás, compartan el mismo espacio y compitan por recoger monedas, todo mientras mantienen sus propias perspectivas y controles individuales.

## 5. Anota tots els recursos, tutorials, documentació que fas servir per aquesta pràctica.

Webgrafia:


- 9 EASY Steps to create a multiplayer game with Unity & Photon – Tutorial: https://www.youtube.com/watch?v=93SkbMpWCGo
- AssetStore Photon Fusion Free: https://assetstore.unity.com/packages/tools/network/photon-fusion-267958
- Tutorial COMPLETO Unity 2D desde Cero: https://www.youtube.com/watch?v=GbmRt0wydQU
- 100 TIPS para aprender a usar UNITY: https://www.youtube.com/watch?v=1W2jsoqLVEc&t=43s
- Juego de Plataformas 2D/Unity Tutorial/1-Capitulo/Escenario/Programaciom videojuegos: https://www.youtube.com/watch?v=-m7ZaHhkDAc
- Unity UI Tutorial | An Introduction: https://www.youtube.com/watch?v=IuuKUaZQiSU
- Create 2D lighting in Unity 2022 in 5MINS: https://www.youtube.com/watch?v=ACyqpLh4jrs
- https://store.steampowered.com/app/3153620/Descenso/
- Unity Essentials: https://learn.unity.com/pathway/unity-essentials
- Unity Tutorial en Español para Principiantes: https://www.youtube.com/watch?v=x3FbFa843Pw
- Como crear un Juego Multijugador en Unity: https://www.youtube.com/watch?v=byRoYfGfXJw
- AssetsStore Pun 2 – Free: https://assetstore.unity.com/packages/tools/network/pun-2-free-119922
- Your Photon Cloud Apps: https://dashboard.photonengine.com