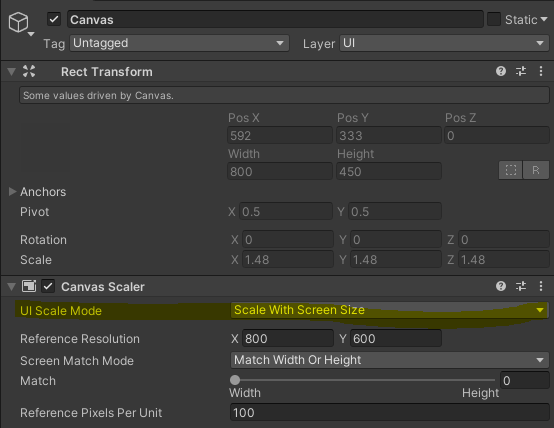
En este último reto veremos como ejecutar el fin de la partida, en este caso cuando uno de los jugadores llegue a 5 puntos, se mostrará un mensaje diciendo cuál ha ganado, además se reseteará el contador por 5 segundos y la partida se reanudará.

Primero que nada, seguro has notado que cuando tu pantalla cambia de tamaño, el texto cambia de lugar, vamos a solucionar eso de primero, vamos a ir a “Canvas” -> “Canvas Scaler”, y donde dice “UI Scale Mode” vamos a seleccionar “scale with screen size”.



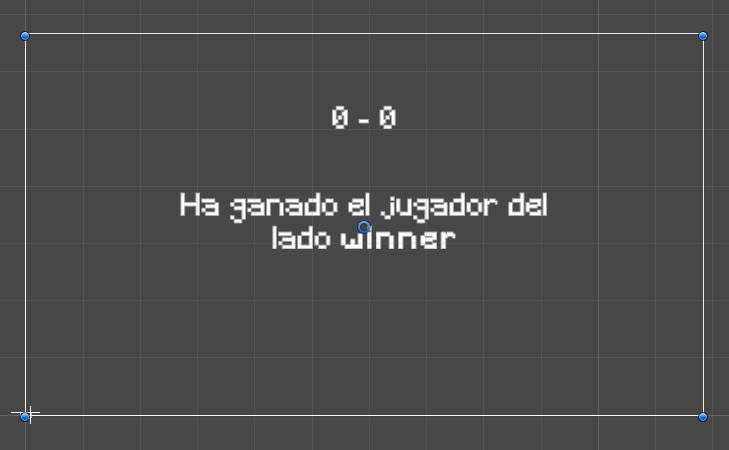
Ahora simplemente tenemos que organizar nuestro texto nuevamente y veremos que al cambiar el tamaño de la pantalla la posición no cambia.

Aprovecharemos para cambiar el nombre del texto del puntaje por “ScoreText” lo usaremos luego.

Vamos a crear un nuevo texto y le pondremos de nombre “WinnerPlayerText”, le pondremos el siguiente texto únicamente para probar “Ha ganado el jugador del lado <b>winner</b>”.

Aunque el curso no va de eso, quiero que te fijes que “<b>” representa negrillas, aunque no es HTML completo, es algo llamado rich text que puede hacer nuestro juego mucho más vistoso.

Centremos el texto que se vea agradable a la vista, debería quedarte algo así:



Y luego de eso borra el texto, ya que no lo usaremos.

En nuestro script Ball.cs vamos a cambiar nuestro start, vamos a cambiar la forma en la que obtenemos el textScore y obtendremos el winnerScore.

**void** **Start**()

{

textScore = GameObject.Find("ScoreText").GetComponent<TMP\_Text>();

winnerScore = GameObject.Find("WinnerPlayerText").GetComponent<TMP\_Text>();

winnerScore.gameObject.SetActive(**false**);

}

Lo pondremos como inactivo para evitar errores más adelante.

Ahora vamos a crear dos Rpc para controlar ese texto, uno para activarlo y mostrar el nombre del jugador ganador y otro para desactivarlo.

[ClientRpc]

**public** **void** **RpcWinState**(**string** winner)

{

winnerScore.gameObject.SetActive(**true**);

winnerScore.text = $"Ha ganado el jugador del lado <b>{winner}</b>";

}

[ClientRpc]

**public** **void** **RpcDisableWinState**()

{

winnerScore.gameObject.SetActive(**false**);

}

Vamos a cambiar nuestra función de StartBall, vamos a añadirle el parámetro delay, es de tipo float.

IEnumerator StartBall(float delay) …

Este delay lo usaremos para mostrar durante un tiempo y luego desactivar el texto de winnerScore, lo pondremos de esta manera.

IEnumerator **StartBall**(**float** delay)

{

rb.simulated = **false**;

rb.velocity = Vector2.zero;

transform.position = Vector2.zero;

// Nuevo codigo

**yield** return new **WaitForSeconds**(delay);

RpcDisableWinState();

**yield** return new **WaitForSeconds**(2);

rb.simulated = **true**;

**float** direction = Random.Range(0f, 1f) > 0.5f ? 1 : -1;

rb.velocity = Vector2.right \* speed \* direction;

}

Como notarás ahora tenemos todo el código para desactivar nuestro winnerScore, pero todavía no tenemos nada para activarlo, vamos a solucionar eso yendo a la función Update:

Vamos a añadir una comprobación para saber si ya superamos los 5 puntos, podemos hacerlo de esta manera:

**int** delay = 0;

**if** (transform.position.x > networkManager.rightRacketSpawn.position.x)

{

leftScore++;

// Codigo nuevo

**if** (leftScore >= 5)

{

RpcWinState("Izquierdo");

delay = 5;

leftScore = 0;

rightScore = 0;

}

StartCoroutine(StartBall(delay));

RpcUpdateTextScore(leftScore, rightScore);

}

De esta manera activamos el winnerScore, los puntajes se resetean y se espera 5 segundos. Nota que este código es **solo para el lado izquierdo** toca hacer lo mismo para el lado derecho, *te lo dejo a ti* .

Ya con eso tendremos nuestro fin del juego completo, recuerda crear un nuevo build y probar.

