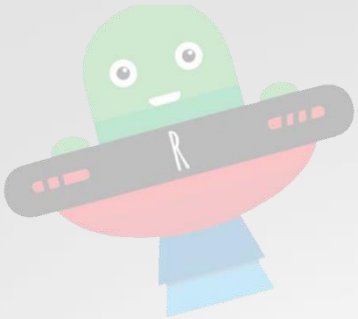


ROCKBOTIC

Soñamos a lo grande

inventamos el mañana

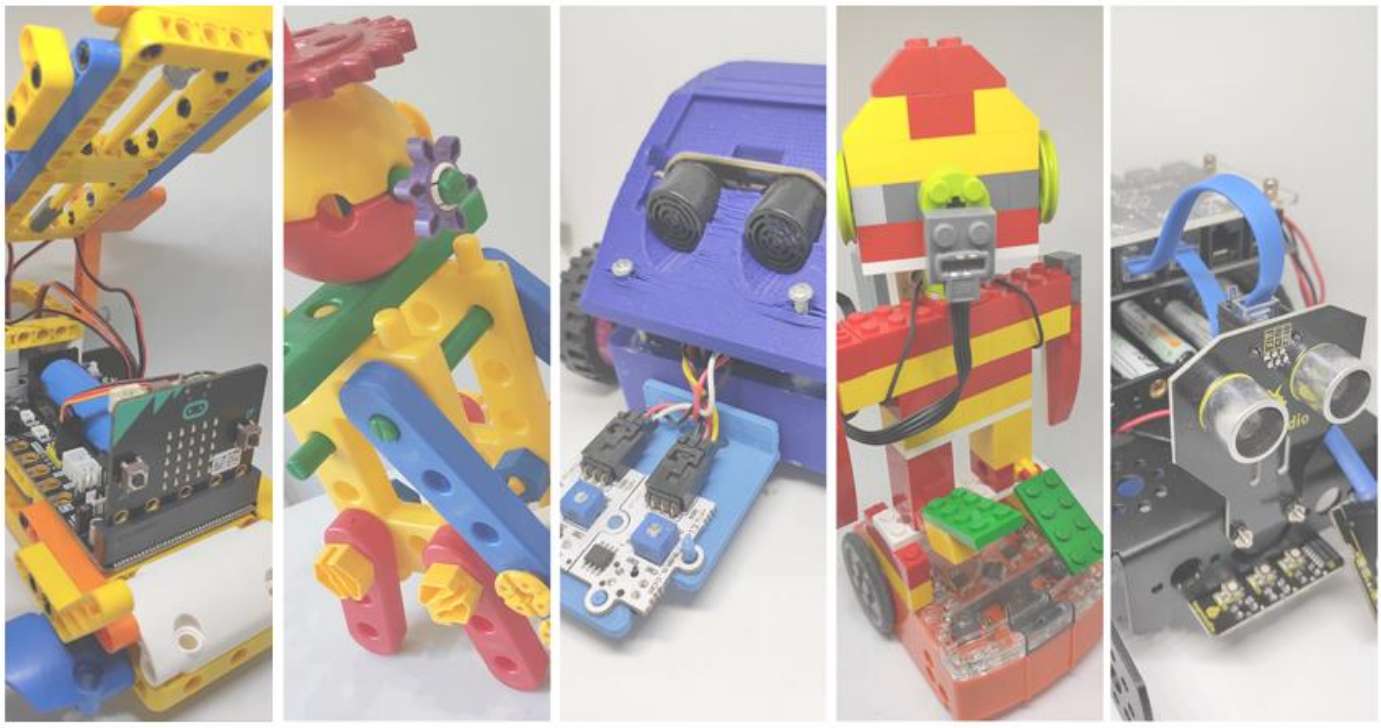
CUADERNO DE ACTIVIDADES DE SECUNDARIA

¿QUÉ ES ESTE CUADERNO?

Este cuaderno de actividades es un material para aprender, imaginar, crear y pasarlo muy bien estos días en casa.

En él encontrarás:

- ¿QUÉ HEMOS APRENDIDO? Actividades de repaso de lo que hemos aprendido durante las clases de robótica, programación y diseño 3D.
- ¡EMPEZAMOS LA AVENTURA! Aquí te explicamos cómo trabajar con las diferentes tecnologías que te proponemos en el cuaderno, con pasos a pasos para trabajar con Unity y hacer circuitos electrónicos con 1,2,3 circuits.
- ¿TE ATREVES CON ESTOS RETOS? Si aún te has quedado con ganas de aprender más, en esta sección vas a encontrar diferentes retos tecnológicos... ¿serás capaz de superarlos?
- PARA MÁS DIVERSIÓN... En este apartado encontrarás actividades divertidas para pasarlo muy bien estos días.
- SOLUCIONES ACTIVIDADES DE REPASO En esta sección encontrarás las soluciones a las actividades de repaso. ¡No hagas trampas y no las mires hasta el final!



¿QUÉ HEMOS APRENDIDO?

Durante estos meses en la clase de tecnología, robótica y programación hemos aprendido cosas muy interesantes...

¿Te atreves a probar si las recuerdas?

1. SOPA DE LETRAS DE ROBOTS

Intenta encontrar todas las palabras que aparecen en la parte inferior en esta sopa de letras.

C	S	E	N	S	O	R	S	B	H	C	U	P	R	O	Y	E	C	T	O
O	I	M	C	O	M	P	U	T	A	D	O	R	A	N	V	I	P	T	Z
D	B	L	N	C	V	A	X	C	B	T	T	B	X	I	N	N	R	E	G
I	E	C	U	A	E	N	A	I	D	P	E	W	N	K	U	S	O	L	P
G	K	Y	B	B	H	X	R	E	H	Q	F	R	Z	Y	L	T	G	E	Y
O	X	I	C	L	T	E	C	N	O	L	O	G	I	A	R	R	R	C	V
U	Z	G	D	E	Z	M	Q	C	O	X	M	F	H	A	O	U	A	T	G
E	T	W	G	S	H	F	G	I	W	H	O	A	Q	T	B	C	M	R	T
R	O	B	O	T	I	C	A	A	W	T	T	J	T	L	O	C	A	O	O
S	O	L	D	A	D	U	R	A	N	Q	O	M	Z	G	T	I	Y	N	R
N	E	N	G	R	A	N	A	J	E	S	R	K	M	Y	T	O	Y	I	N
F	V	Z	E	J	K	O	O	V	C	M	E	T	A	L	A	N	S	C	I
K	M	E	M	O	R	I	A	Q	F	H	U	O	L	B	R	E	Z	O	L
D	I	V	E	R	S	I	O	N	D	J	I	W	C	Z	N	S	K	S	L
O	O	G	C	O	N	S	T	R	U	I	R	P	G	I	N	S	W	W	O
A	R	U	E	D	A	S	A	Y	U	C	O	N	E	C	T	O	R	E	S

COMPUTADORA
METAL
CONECTORES
DIVERSION
BATERIA
CABLES

PROYECTO
RUEDAS
CHIP
CODIGO
MEMORIA
MOTOR

ROBOTICA
CONSTRUIR
TECNOLOGIA
PROGRAMA
ENGRANAJES
SOLDADURA

CIENCIA
ROBOT
SENSOR
INSTRUCCIONES
TORNILLOS
ELECTRONICOS

2. SENSORES Y ACTUADORES

En clase hemos visto que los robots tienen diferentes sensores y actuadores...

¿Recuerdas que es un sensor? Escríbelo a continuación:

¿Qué tipos de sensores y actuadores podemos añadir a una placa Arduino?

¿Qué tipo de sensores y actuadores tiene la tarjeta Micro:bit?

Dibuja un robot de tu imaginación con diferentes sensores y actuadores. ¿Crees que cuando volvamos a clase serás capaz de construirlo y programarlo?

3. ¿CUÁNTO SABES DE ROBÓTICA?

Hemos preparado un test para ver cuánto saber de robótica...

¿Te atreves a ponerte a prueba?

También puedes hacerlo online a través de este enlace: https://es.educaplay.com/recursos-educativos/5204844-cuanto_sabes_de_robotica.html

1. ¿Qué necesita un robot para interactuar de forma autónoma con el entorno?

- ☐ Brazos
- ☐ Sensores
- ☐ Actuadores
- ☐ Ruedas

2. La robótica es una disciplina basada en...

- ☐ La informática
- ☐ La mecánica
- ☐ Las matemáticas
- ☐ Multidisciplinar

3. ¿Qué es Arduino?

- ☐ Una aplicación
- ☐ Una placa
- ☐ Un juego
- ☐ Un tipo de pizza

4. ¿Quién formuló las leyes de la robótica?

- ☐ Isaac Asimov
- ☐ Niel deGrasse Tyson
- ☐ Carl Sagan
- ☐ Eduard Punset

5. Para que un robot tenga algún grado de inteligencia artificial necesita de:

- ☐ Partes móviles
- ☐ Algoritmo adaptativo

- ☐ Cámaras de visión
- ☐ Capacidad de hablar

6. ¿Qué tipo de robot puede no tener inteligencia artificial (IA)?

- ☐ Chat Bot
- ☐ Robot de montaje
- ☐ Rover de exploración espacial
- ☐ Furby

7. ¿Qué es Blender?

- ☐ Un programa de diseño 3D
- ☐ Un teléfono de última generación
- ☐ Un videojuego
- ☐ El nombre de mi mascota

8. ¿Qué es Unity?

- ☐ Algo
- ☐ Un programa de diseño 3D
- ☐ Un entorno de desarrollo de videojuegos en 3D y 2D
- ☐ Una nueva marca de ropa

9. De que hablamos cuando nos referimos a "Grados de libertad" en un brazo robótico.

- ☐ La apertura de agarre que tiene
- ☐ El número de botones que tiene
- ☐ Nº de direcciones en las que se puede mover
- ☐ Libertad de ejecutar una acción

10. Un robot debe ser:

- ☐ Educado
- ☐ Ser móvil
- ☐ Llevar cámara
- ☐ Ser programable

¡EMPEZAMOS LA AVENTURA!

1 2 3 CIRCUIT

1. EMPEZAMOS CON 1 2 3 CIRCUIT

123 Circuit, es una herramienta online gratuita de Autodesk que permite simular la conexión y montaje de circuitos electrónicos usando el protoboard. Además, permite simulación de circuitos, e incluso podemos realizar la "programación virtual" de las placas Arduino y comprobar el funcionamiento. Esta herramienta es gratuita e intuitiva, fácil de arrastrar elementos y soltar en la construcción y diseño de circuitos.

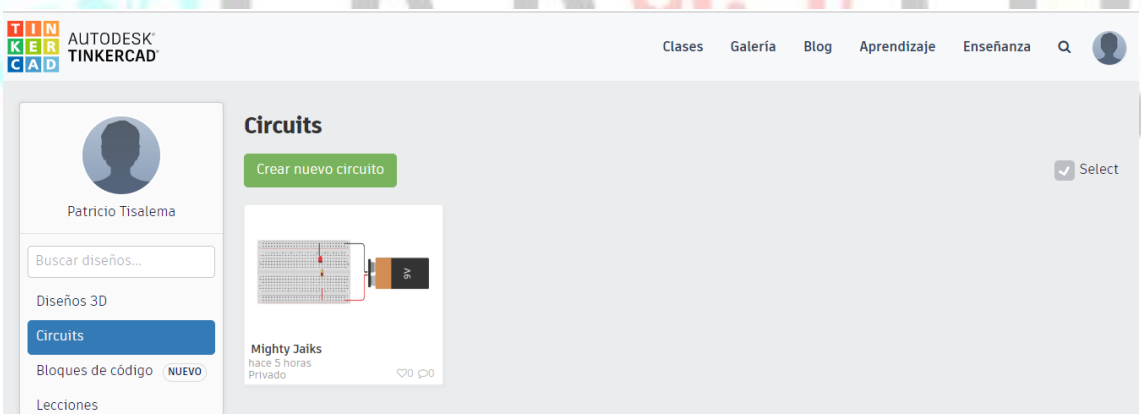
¿Cómo entrar a 123 circuit?

Para entrar a la clase que hemos creado para estas actividades, has click en este enlace:

https://www.tinkercad.com/joinclass/UIJTFM8MQGTS

Si te pide un alias escribe: Bombilla1

Para empezar a diseñar el circuito, hacer click sobre la opción **CIRCUIT** y CREAR NUEVO CIRCUITO. Pon tu nombre personal como nombre del proyecto.

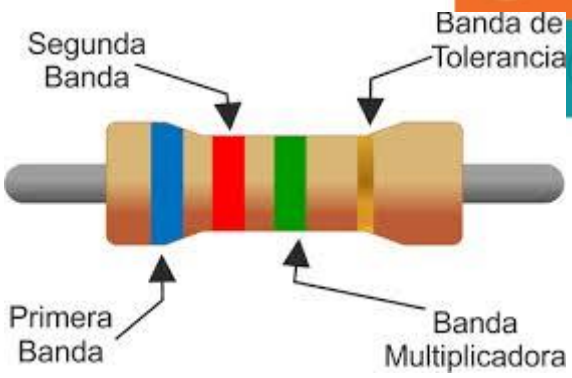


Conociendo la electrónica básica

Antes de hacer el montaje y la programación de circuitos, vamos a conocer rápidamente algunos conceptos básicos de elementos electrónicos que usaremos en estos ejercicios.

¿Qué es una Resistencia?

La Resistencia Eléctrica es la oposición o dificultad al paso de la corriente eléctrica. Cuanto más se opone un elemento de un circuito a que pase por el la corriente, más resistencia tendrá. La resistencia eléctrica se mide en Ohmios (Ω) y se representa con la letra R.



¿Qué es una batería?

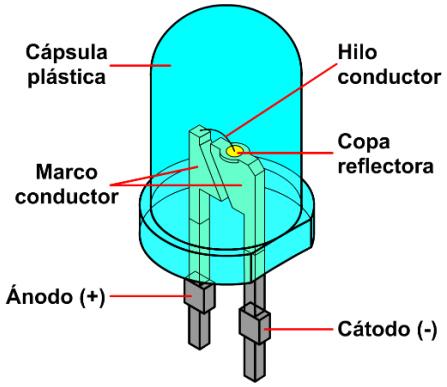
Una batería es un dispositivo que almacena electricidad en forma de energía química. ... Para recargar la batería, debe conectarse una fuente de alimentación externa, como un cargador de baterías, un alternador o un panel solar, con una tensión de aproximadamente 2,4 V por célula.

¿Qué es potenciómetro?

Un potenciómetro es un resistor eléctrico con un valor de resistencia variable y generalmente ajustable manualmente. El valor de un potenciómetro viene expresado en ohmios (símbolo Ω) como las resistencias, y el valor del potenciómetro siempre es la resistencia máxima que puede llegar a tener.

¿Qué es un LED?

Un LED (acrónimo del concepto inglés light-emitting diode) es un diodo emisor de luz. En su interior hay un semiconductor que, al ser atravesado por una tensión continua, emite luz, lo que se conoce como electroluminiscencia. Existen distintos tipos de led en función de las tecnologías usadas para su fabricación y montaje sobre circuitos electrónicos.

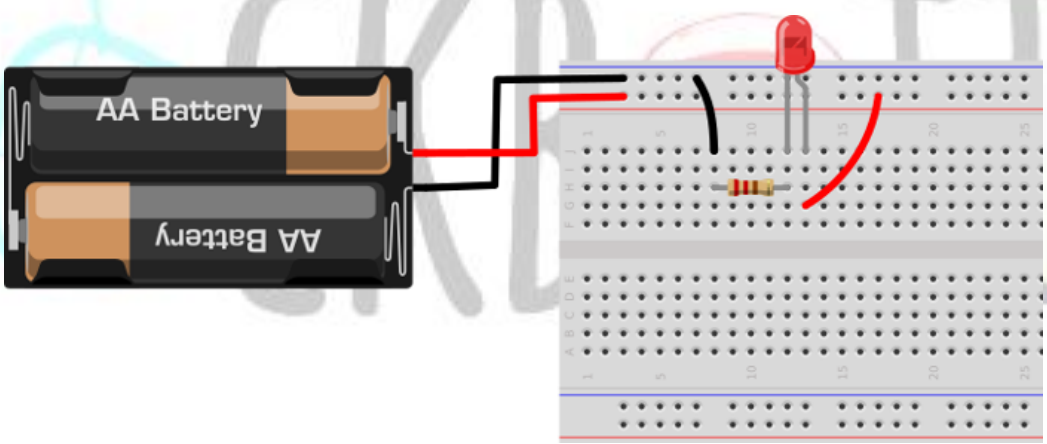


¿Qué es un zumbador?

Un zumbador (en inglés: buzzer) es un transductor electroacústico que produce un sonido o zumbido continuo o intermitente de un mismo tono (generalmente agudo).

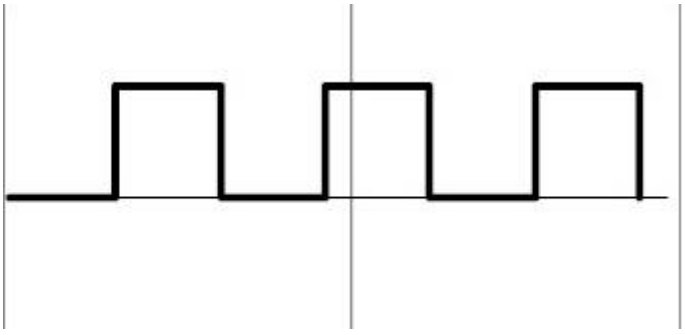
¿Qué es un protoboard?

La Protoboard, llamada en inglés breadboard, es una placa de pruebas en los que se pueden insertar elementos electrónicos y cables con los que se arman circuitos sin la necesidad de soldar ninguno de los componentes. Las Protoboards tienen orificios conectados entre si por medio de pequeñas laminas metálicas. Usualmente, estas placas siguen un arreglo en el que los orificios de una misma fila están conectados entre sí y los orificios en filas diferentes no. Una Protoboard es un instrumento que permite probar el diseño de un circuito sin la necesidad de soldar o desoldar componentes.



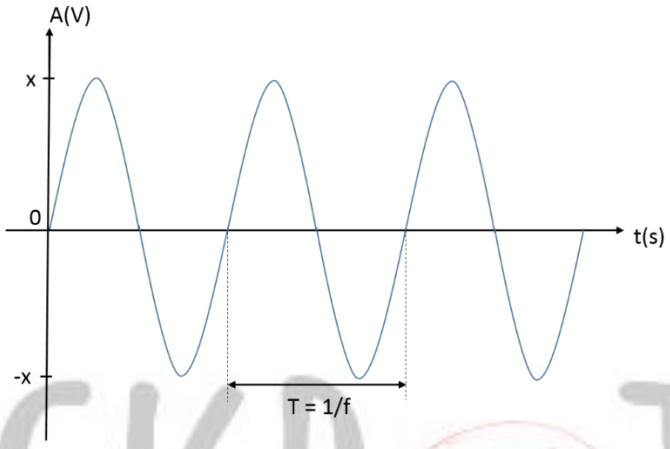
¿Qué es señal digital?

Una señal digital es aquella que presenta una variación discontinua con el tiempo y que sólo puede tomar ciertos valores discretos (1,0). Su forma característica es ampliamente conocida: la señal básica es una onda cuadrada (pulsos) y las representaciones se realizan en el dominio del tiempo.



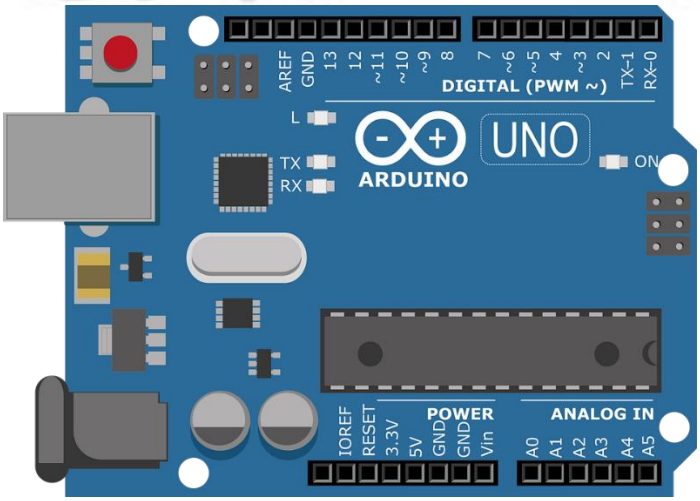
¿Qué es señal analógica?

La señal analógica es aquella que presenta una variación continua con el tiempo, es decir, que a una variación suficientemente significativa del tiempo le corresponderá una variación igualmente significativa del valor de la señal (la señal es continua). Los valores de una señal analógica pueden estar entre 0 a 1023.



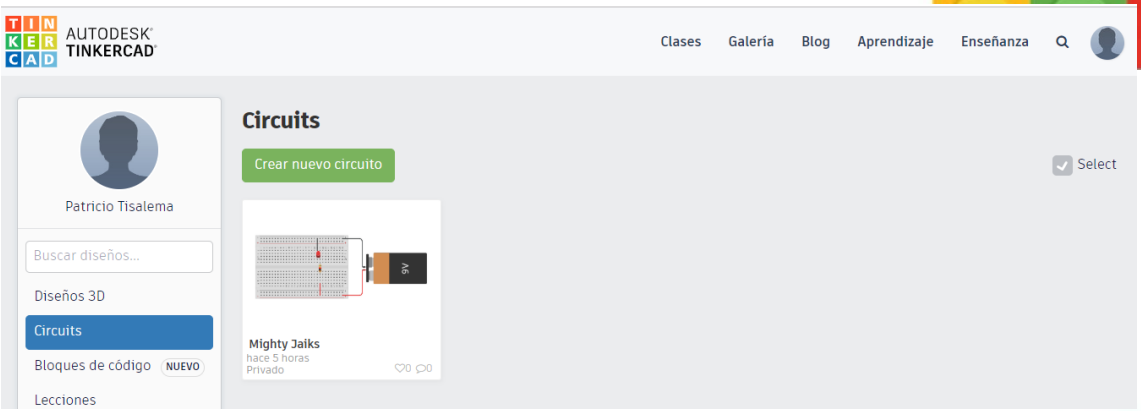
¿Qué es arduino?

Arduino es una plataforma de desarrollo basada en una placa electrónica de hardware libre que incorpora un microcontrolador re-programable y una serie de pines hembra. Estos permiten establecer conexiones entre el microcontrolador y los diferentes sensores y actuadores de una manera muy sencilla



Actividades

Entramos a la página e iniciamos un nuevo proyecto.



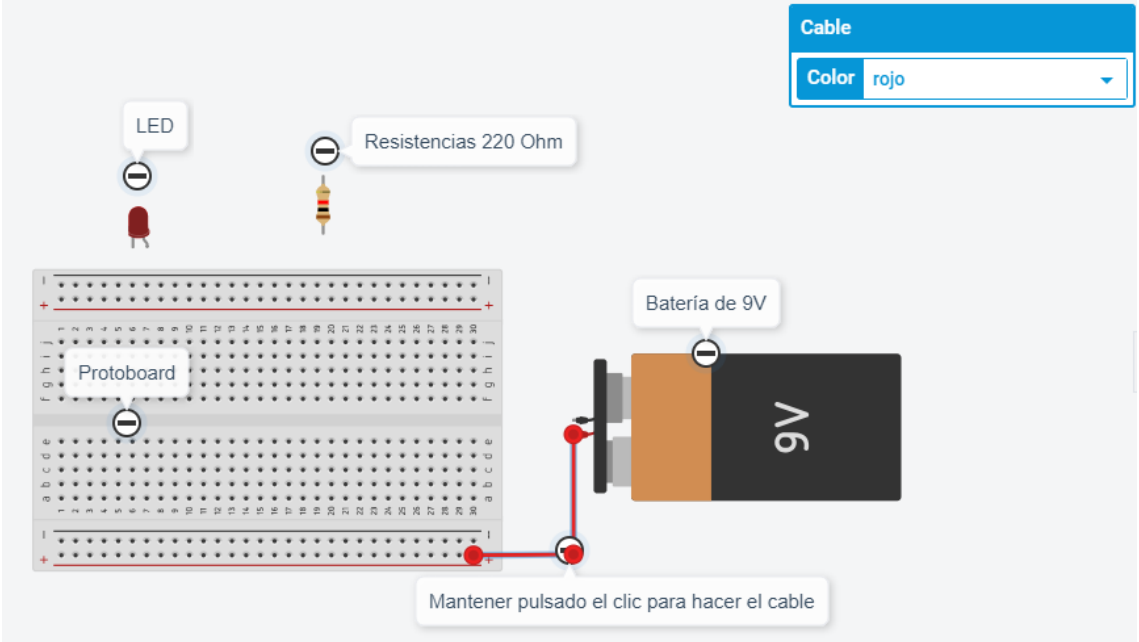
El primer ejercicio es hacer un montaje de un circuito para encendido de un LED.

Materiales de diseño:

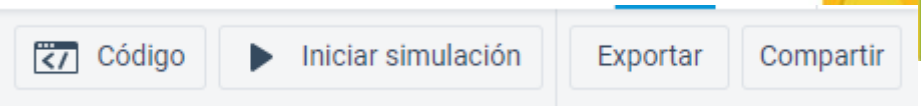
- Protoboard
- 1 LED
- 1 Resistencia de 220 Ω
- 1 bateria de 9V

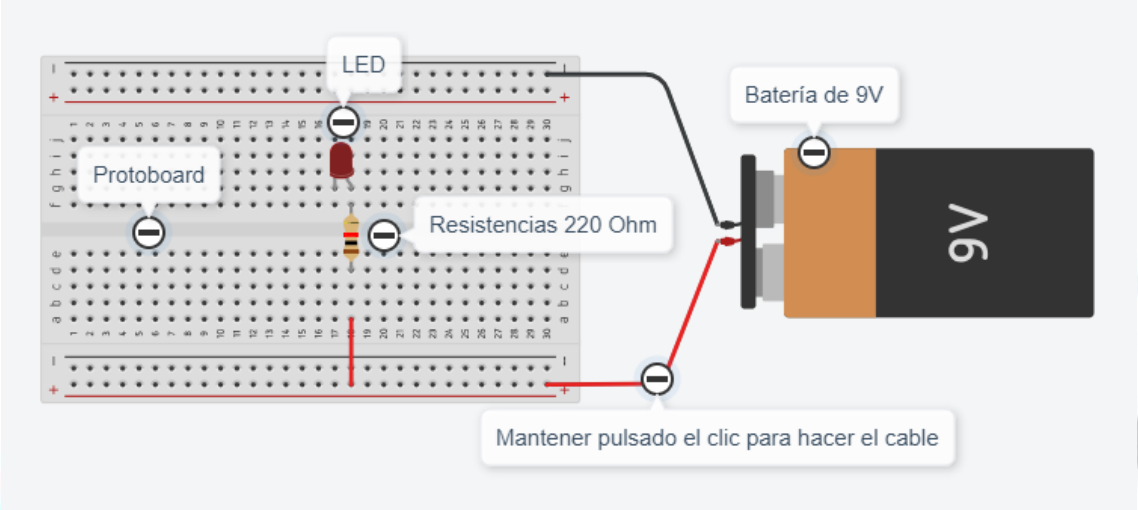
Pasos para seguir

1. Buscar los elementos electrónicos en el panel derecho del programa



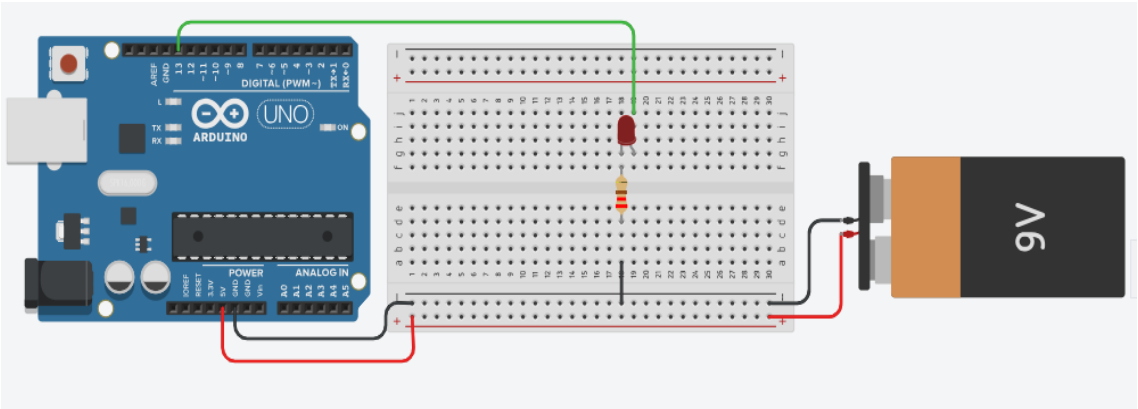
2. Hacer el montaje como se muestra en la imagen y pulsar en el botón iniciar simulación, para ver el resultado de encender y apagar el led de la placa protoboard.





Aprendiendo a programar

Ahora pondremos una placa Arduino a nuestro montaje y programaremos para que el LED que estará conectado al pin 13 de Arduino encienda y apague.



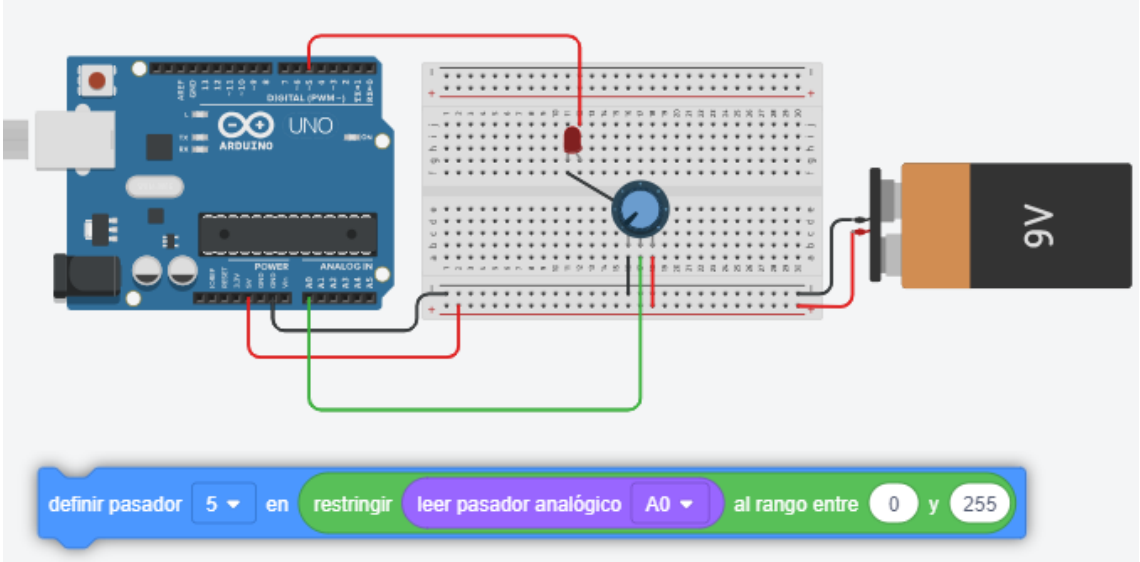
Programación

Ahora haremos click en el botón código y programaremos nuestro led para encender y apagar.

Ejercicio

En este ejercicio haremos el montaje de un circuito sobre el protoboard, donde el LED esté conectado al potenciómetro, para controlar la intensidad de luz mientras se gira el potenciómetro. El LED estará conectado al pin digital 5 de la tarjeta Arduino y el potenciómetro al pin analógico A0 de Arduino.

Ejemplo de montaje y programación.



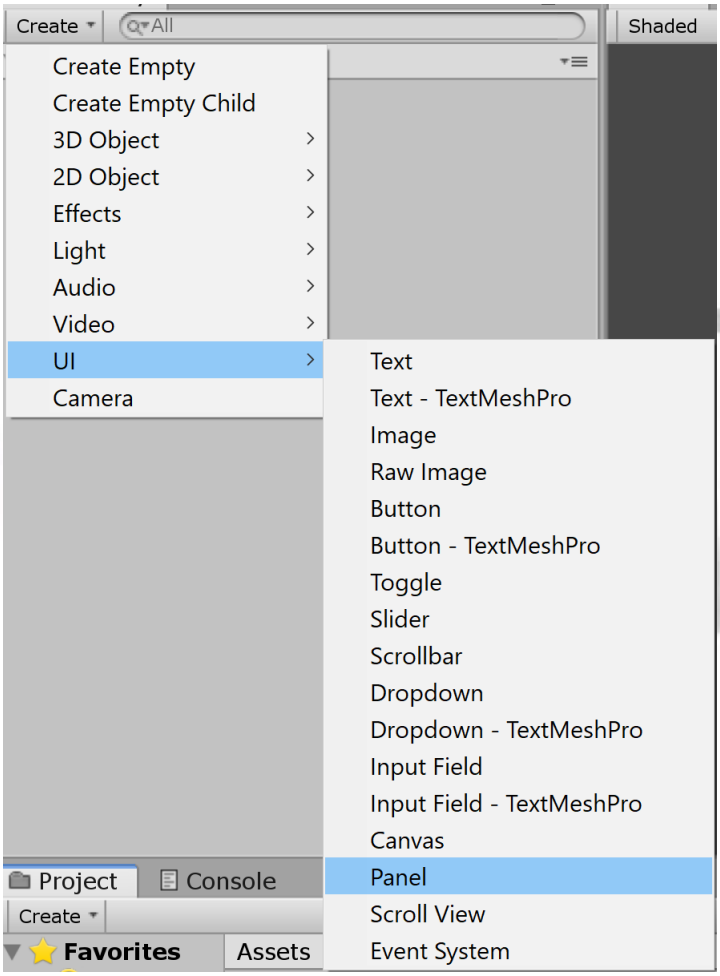
En este ejercicio el bloque azul define el pin digital 5 de Arduino, el bloque verde define que los valores del potenciómetro no deben pasar de un valor de 255 y el bloque morado define el pin analógico A0 donde está conectado al potenciómetro.

¡PROGRAMAMOS CON UNITY!

1. CREAMOS EL TABLERO DEL TICTACTOE O TRES EN RAYA

Creamos un nuevo proyecto con estilo 2D, una vez estemos en la pantalla inicial de Unity.

Para que nuestro juego tenga un sitio donde "estar", debemos crear un tablero, para ello en el panel créate, buscaremos la opción **UI/Panel**.

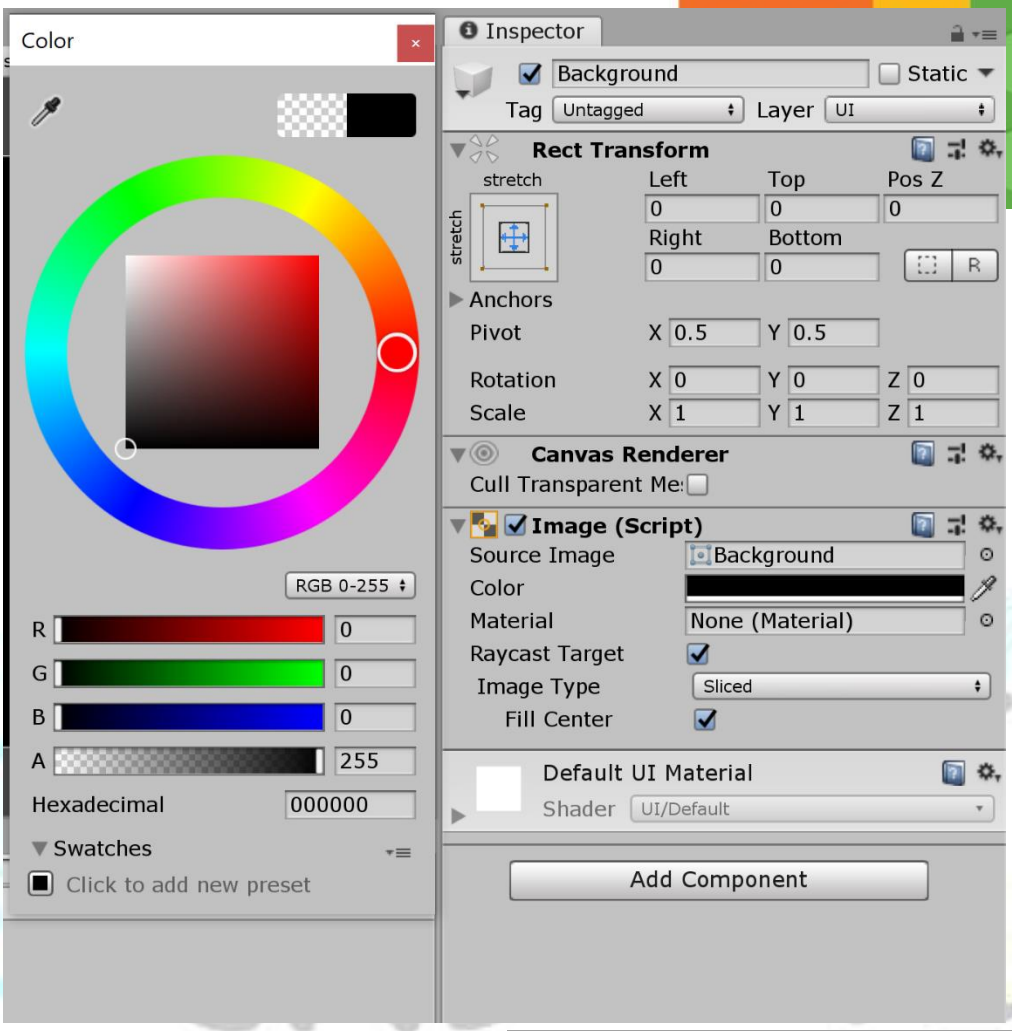


Esto nos crea un nuevo objeto llamado panel que como observamos es hijo de un Canvas, también nos aparece un EventSystem en la jerarquía.

Nos colocaremos de modo que veamos nuestro panel en nuestra pantalla, para ello usaremos la rueda del ratón para quitar el zoom y pulsándola para posicionarnos en el centro del panel.

Seleccionamos el panel y observamos el inspector.

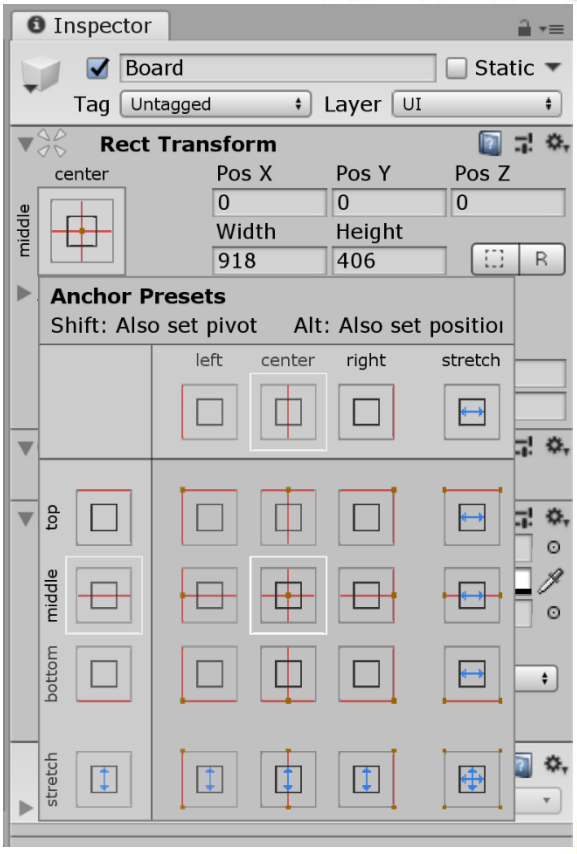
- Cambiamos el nombre por "Background".
- Cambiamos las propiedades del color del fondo poniéndole en lugar de negro, 100% opacidad.



Ahora, tenemos un panel negro donde trabajar, a continuación, comenzaremos a crear las líneas de nuestro tablero de TicTacToe.

Para hacer esto, debemos crear otro panel y poner las propiedades correspondientes.

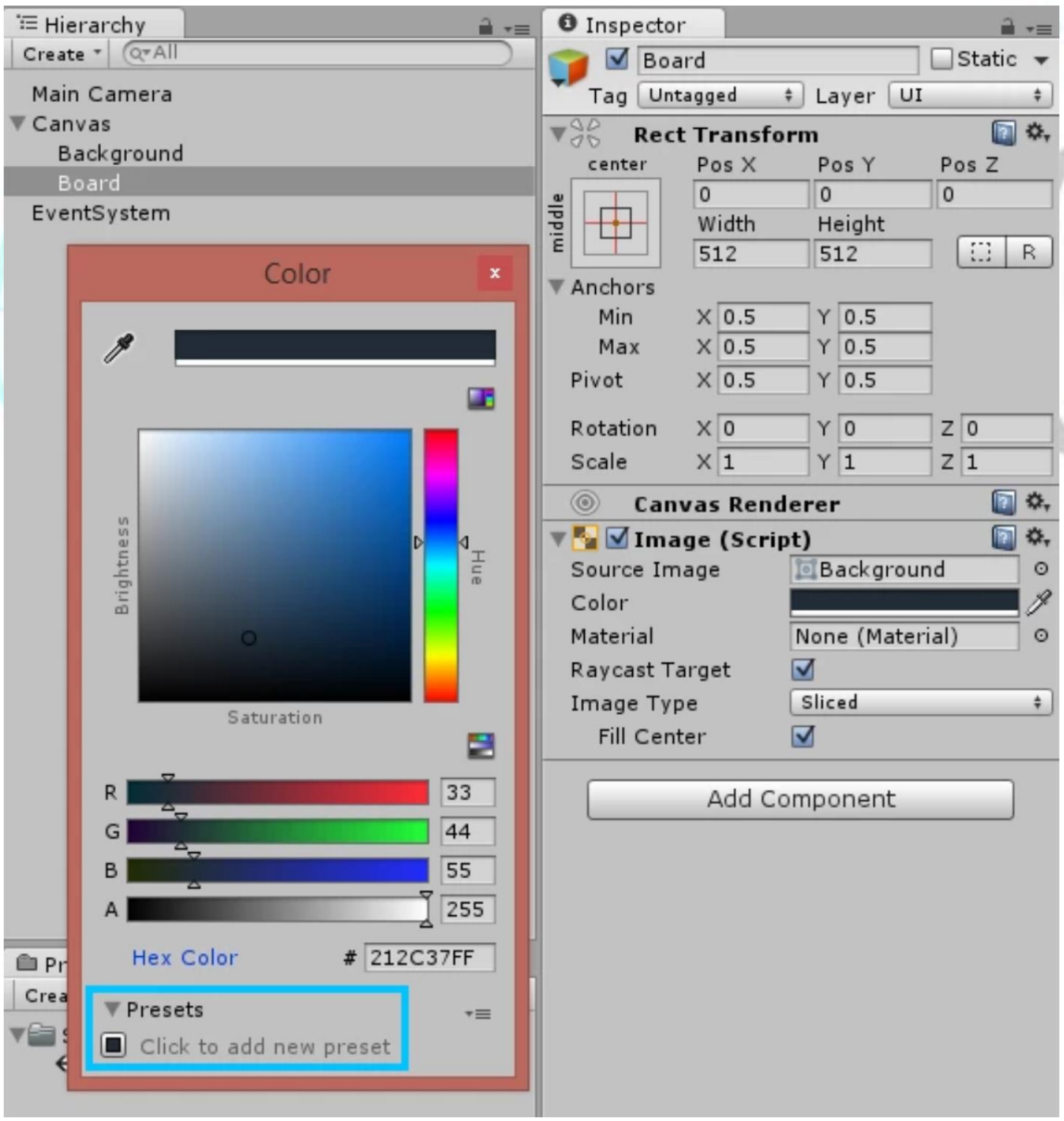
- Crearemos un nuevo UI panel. Create/UI/Panel
- Con el nuevo panel seleccionado cambiaremos su nombre por "Board"
- Pulsaremos en la opción de "Anchor" y dejando pulsadas las teclas "Shift" y "Alt", seleccionaremos la opción del medio.



Esto nos permite cambiar al centro de nuestro GameObject, como podemos observar, ahora nos aparecen 4 pequeños triángulos en el centro de nuestro canvas. Ahora haremos que parezca más un tablero, selecciona el objeto del tablero.

- Establece el parámetro **RectTransform's Width** en 512.
- Establece el parámetro **RectTransform's Height** en 406.
- En las opciones de la imagen seleccionaremos las propiedades del color, en esta ocasión vamos a ponerlo de un color azul muy oscuro (33, 44, 55, 255) con 100% opacidad.

Una vez puesto el color lo guardaremos como un preset para así asegurarnos que tenemos este color en el futuro.



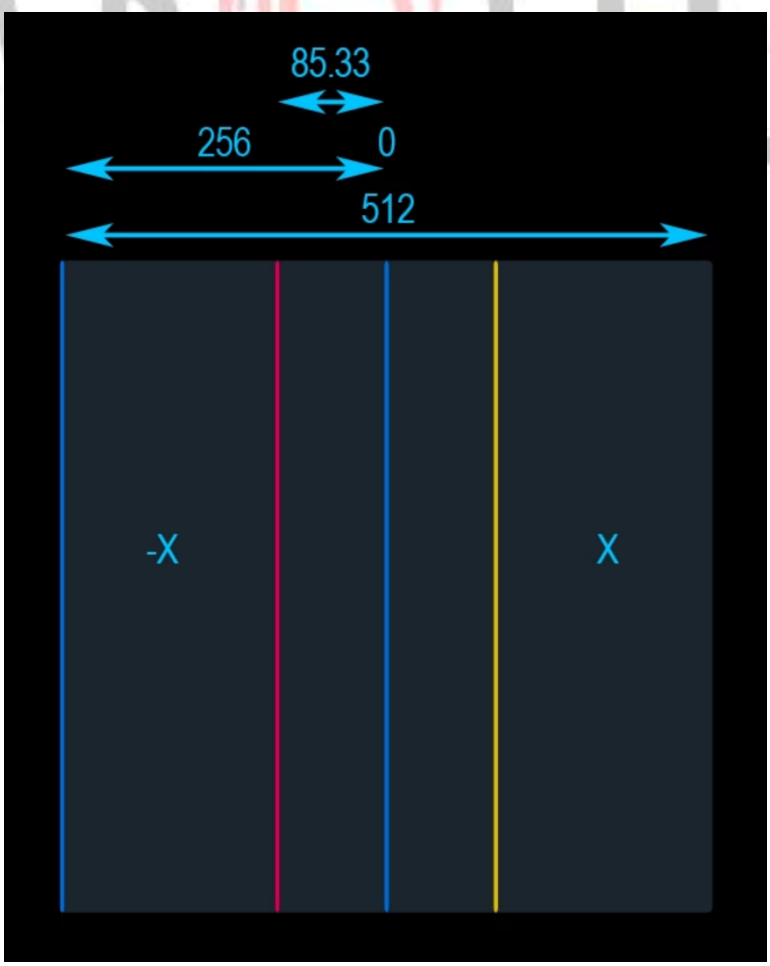
Llegados a este punto y con el panel en su sitio y colocado, debemos empezar a construir las separaciones de nuestro panel. Para ello separaremos nuestro tablero en 9 espacios, lo haremos usando el panel de UI.

Para crear nuestra cuadricula:

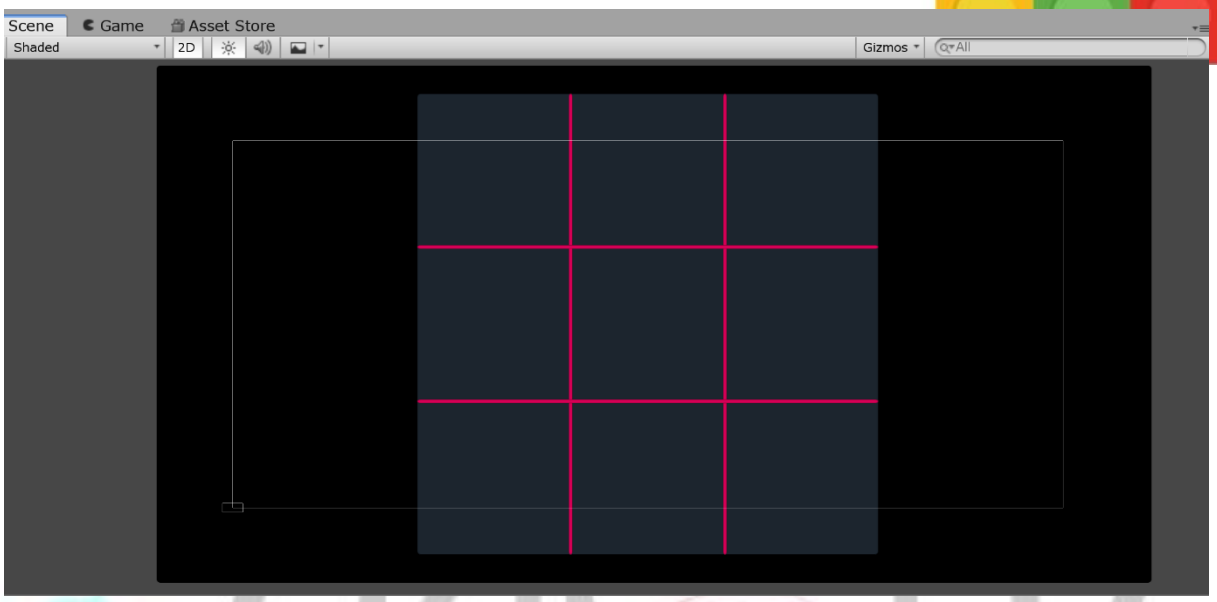
- Creamos un nuevo panel como hemos aprendido a hacer hasta ahora.
- Renombramos nuestro objeto nuevo como “Grid”, y colocamos el Anchor, pivot y Position en el centro.
- El parámetro Width lo establecemos en 5
- El parámetro Height lo establecemos en 512
- Ponemos la Position X en -85.33
- Pondremos un color rosa (255, 0, 102, 255).
- Colocamos el color en los preset para poder usarlo en las siguientes líneas de separación.

¿Cómo obtuvimos el número mágico -85.33 para el valor de Posición X? Esto se hace simplemente con las matemáticas básicas.

El tablero tiene 512 píxeles por 512 píxeles. Si simplemente dividiéramos 512 por 3 obtendríamos un número alrededor de 170 (pronto veremos este número nuevamente). Sin embargo, cuando se trata de posicionamiento en la escena o en el espacio de la pantalla, el mundo comienza en un punto de origen y el valor de las posiciones se aleja de ese punto, positivo en una dirección y negativo en la otra. El tablero del juego está en el centro de la pantalla con una posición X de 0 en su centro y la posición X de -256 y 256 en cada una de las esquinas. ¡Queremos que la línea de la cuadrícula sea 1/3 del camino de 0 a 256, o 256 dividido por 3, lo que equivale a 85.33!



Ahora haremos el resto de las líneas de la cuadricula. Recordad que las posiciones en X cambiarán entre 85.33 y -85.33 tanto en la posición X como en la posición Y.



Al final, debe quedarnos algo parecido a esto.

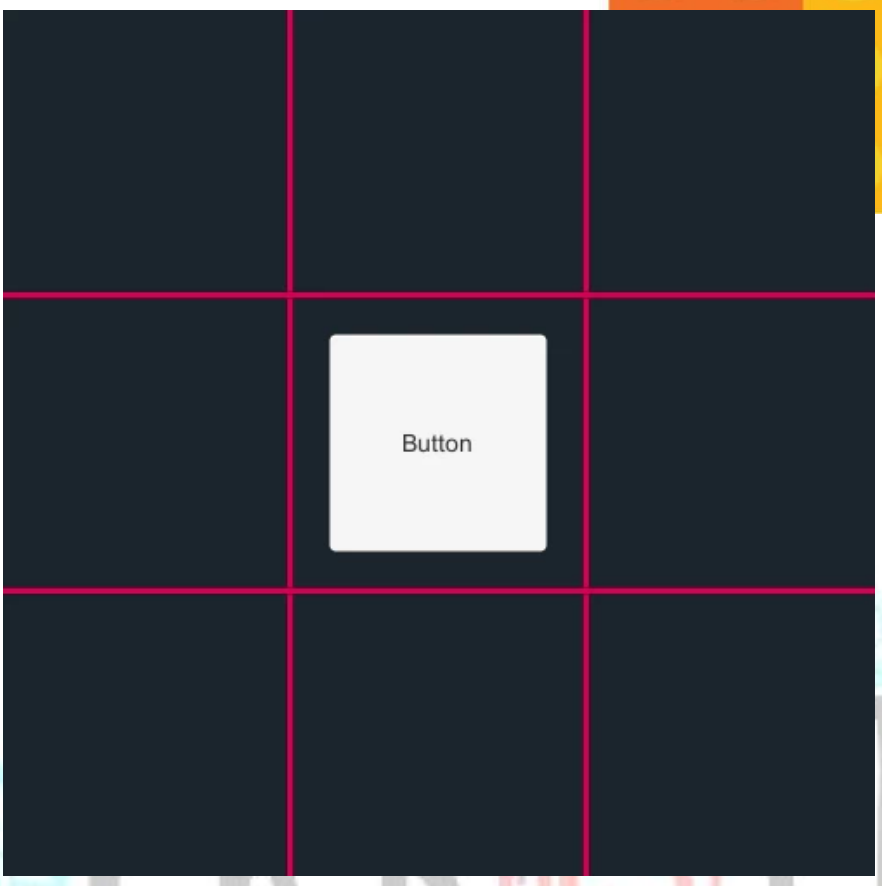
2. CREAMOS LAS PIEZAS DEL TICTACTOE O TRES EN RAYA

Con nuestro tablero de juego básico configurado, ahora necesitamos encontrar una manera para que nuestros jugadores interactúen con el juego. El jugador debe poder hacer clic en un cuadrado de la cuadrícula y asignar su valor, ya sea "X" u "O", a ese espacio.

Una de las mejores maneras de obtener la interacción de "clic" de un usuario es usar un botón de UI. Los botones pueden detectar un clic y realizar una acción o un conjunto de acciones en respuesta a la entrada. Convenientemente, los botones UI vienen con un elemento de texto adjunto como elemento secundario. Una manera fácil y conveniente de mostrar el valor del jugador, ya sea "X" u "O", es simplemente agregar el movimiento del jugador directamente al elemento de texto UI adjunto como un carácter de texto.

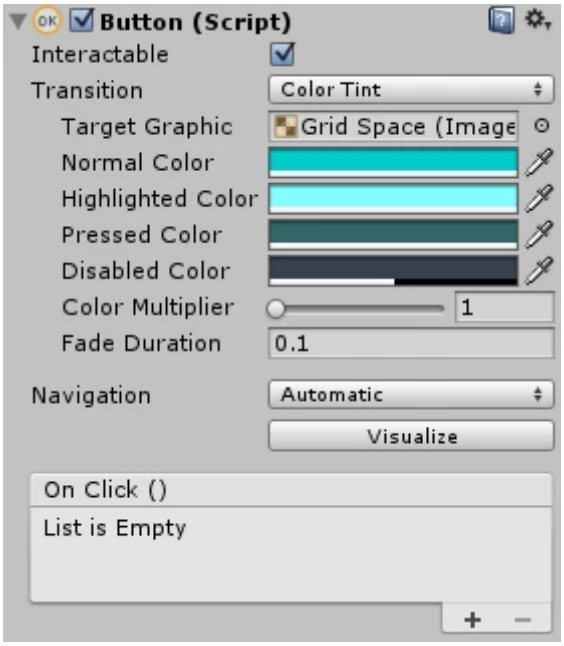
- Creamos un Botón Create > UI > Button
- Con el botón GameObject seleccionado,
    - Renombramos el objeto como “Grd Space”
    - Pondremos en “Width” en 128
    - Pondremos en “Height” en 128
    - Reseteamos el “RectTransform”

Esto, nos creará un Botón en medio de nuestro tablero.



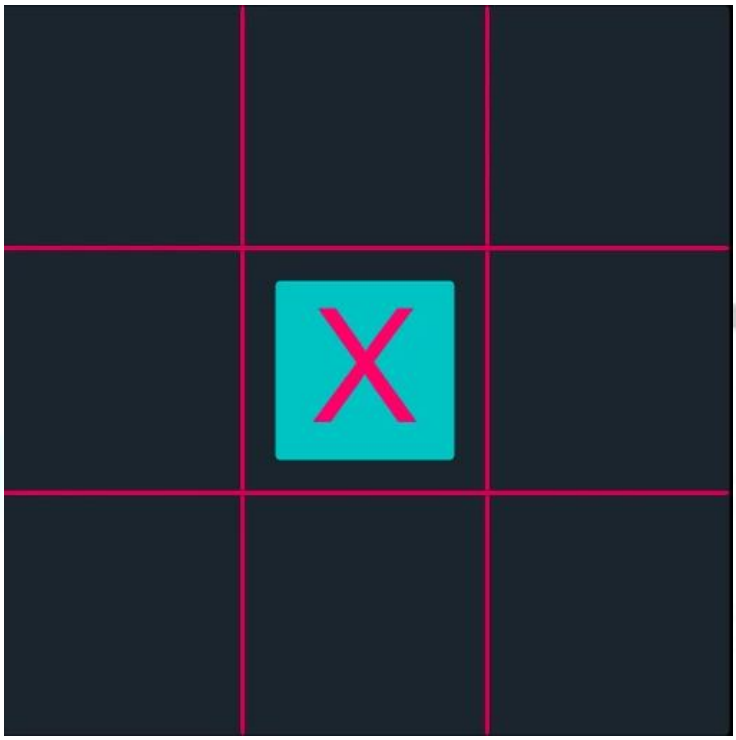
El siguiente paso es establecer el estilo del elemento del botón UI. Haremos esto configurando el color de los elementos de transición en el componente Button y el componente de color Text en el elemento de texto de la interfaz de usuario secundaria.

- Selecciona el objeto Grid Space en la jerarquía.
- Con Grid Space seleccionado
    - Cambiamos el Normal Color a azul (0, 204, 204, 255).
    - El "Highlighted Color" lo cambiamos a un azul más claro (128, 255, 255, 255)
    - La opción de "pressed Color" la pondremos con un azul verdoso oscuro (51, 102, 102, 255)
    - Por último la parte de "Disabled Color" tendrá un azul oscuro (55, 66, 77, 255)



Ahora en la ventana de “Hierachy” abriremos nuestro botón y haremos doble clic en la opción de texto.

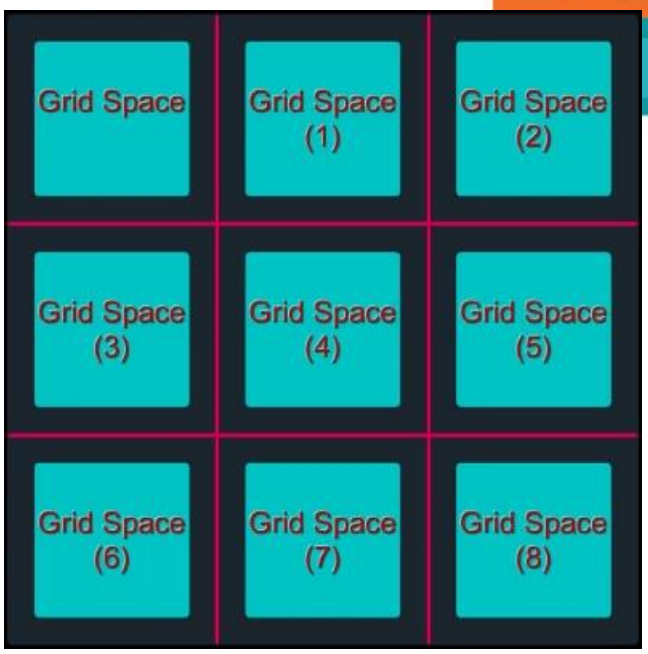
- Cambiaremos el texto por la letra “X”
- El tamaño debe ser 111.
- El color del texto, siguiendo la estética de nuestro panel lo pondremos del mismo color (en mi caso de color rosa (255, 0, 102, 255)) si lo hemos guardado, con dar clic al color es suficiente.



Ahora podemos guardar todo esto como un “Prefab”

- Creamos una nueva carpeta en el “Project Window”
- La llamamos “Prefabs”
- Seleccionamos el “Grid Space” en la ventana de jerarquía y la arrastramos hasta la carpeta que acabamos de crear.

Ahora tenemos nuestro botón de cuadrícula prefabricado. Necesitamos duplicar este botón ocho veces, para un total de nueve botones. Estos se colocarán alrededor del tablero de juego en forma de cuadrícula. Unity creará un nombre único agregando un número al final del nombre de cada GameObject. En la Ventana de Jerarquía, seleccione el Objeto de Juego de Espacio de Cuadrícula y duplíquelo 8 veces.



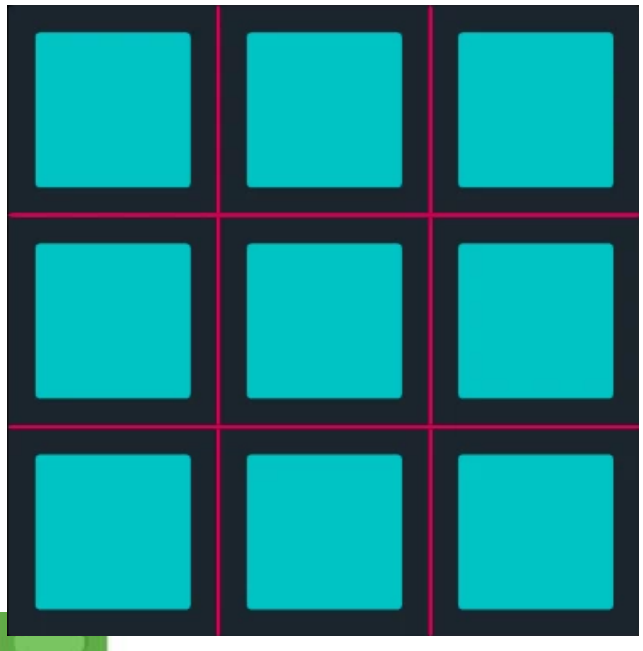
Hay dos formas fáciles de mover estos botones a su lugar. Una es establecer los valores directamente en RectTransform del botón de la interfaz de usuario. El otro es arrastrar los botones en la vista de escena y usar las funciones de colocación y ajuste para alinear los botones.

Para mostrar, por ejemplo, cómo establecer la posición de un botón utilizando directamente RectTransform del botón:

- Seleccionamos el "Grid Space" que queramos editar
    - Vamos cambiando las posiciones de "X" o "Y" a -170 o 170

Ahora borraremos la "X" del texto de nuestros botones, para que quede en blanco (la "X" nos ayuda a colocar mejor los botones si lo vamos a colocar arrastrando los elementos).

Al final nuestro panel de juego tiene que quedar algo así:



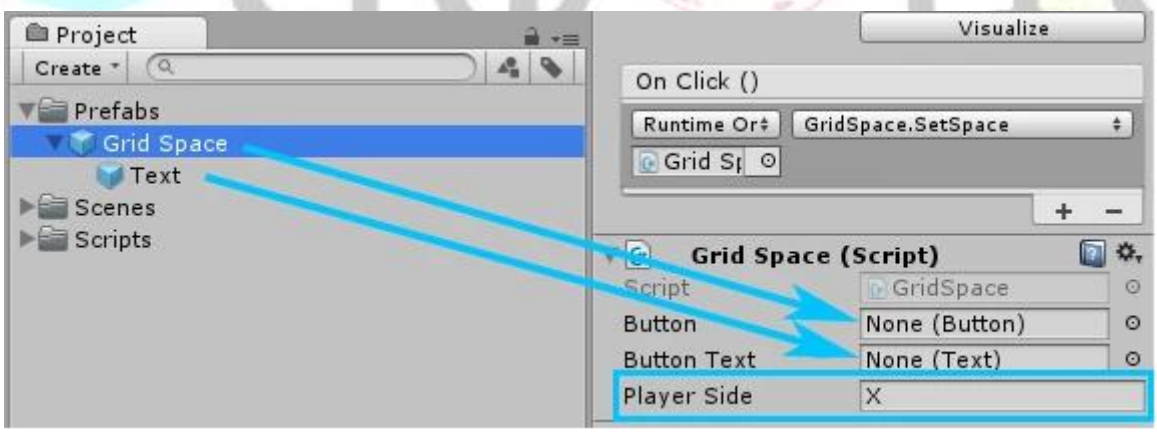
Guardaremos nuestro proyecto. Con esto ya tendríamos todo lo que es la interface creada, en la siguiente parte, comenzaremos a crear nuestra programación y una IA para poder jugar.

3. CONTINUAMOS TICTACTOE

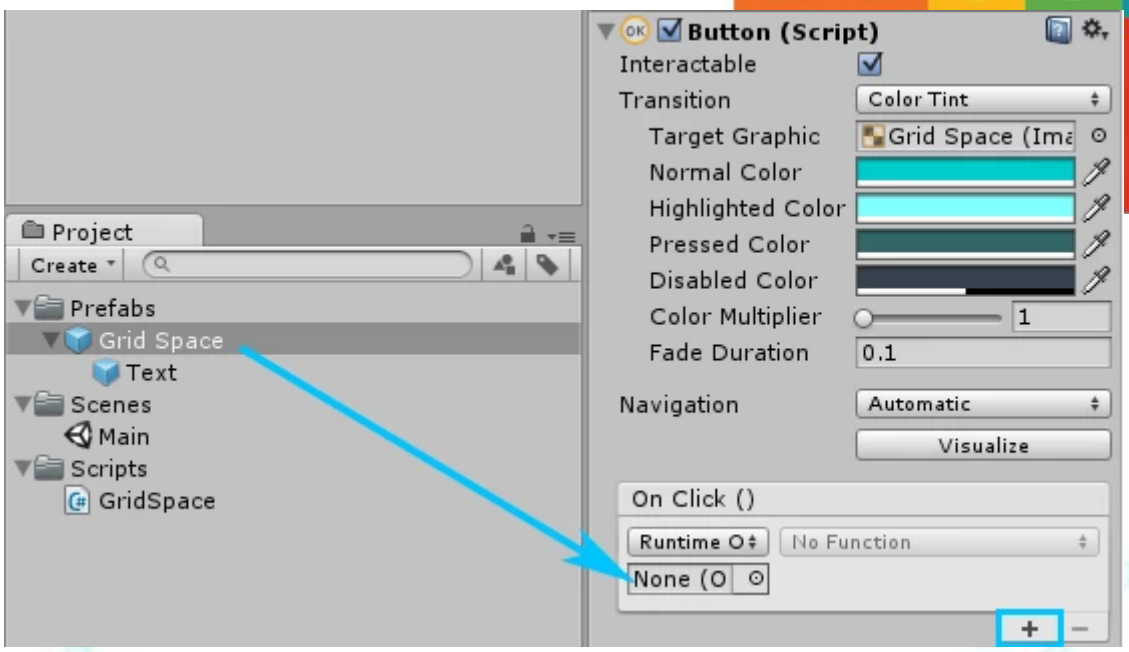
Para empezar con el apartado de programación comenzaremos creando una carpeta en nuestra "Project Window", a la cual llamaremos "Scripts". Dentro de esta carpeta crearemos un nuevo Script que llamaremos "GridSpace", haremos doble clic en el Script para abrir el editor de código.

```
using UnityEngine;
using System.Collections;
using UnityEngine.UI;
public class GridSpace : MonoBehaviour
{
    public Button button;
    public Text buttonText;
    public string playerSide;
    public void SetSpace()
    {
        buttonText.text = playerSide;
        button.interactable = false;
    }
}
```

Una vez creado nuestro Script seleccionamos nuestro "Grid Space" en nuestra Project window, arrastraremos nuestro prefab hasta la opción "Button". Acto seguido arrastramos el hijo "Text" hasta la propiedad de "Button text". En el apartado de "Player Side", debemos poner "X".

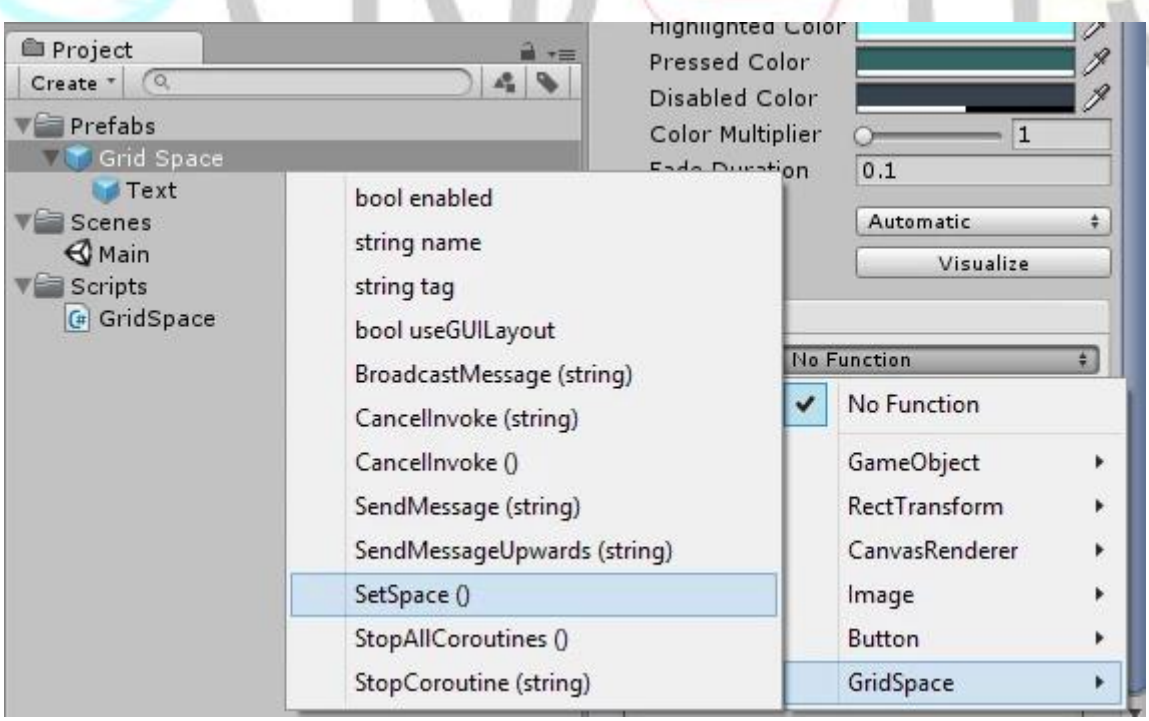


Ahora volveremos a seleccionar nuestro "Grid Space". añadimos una nueva opción pulsando sobre el símbolo "+", donde nos aparece la opción "None" arrastramos nuestro prefab..



Con el prefabricado Grid Space seleccionado,

- Con el componente Botón, en la lista desplegable de funciones, seleccione GridSpace> SetSpace.
- Guarda la escena
- Dale al modo reproducción y pulsa en los espacios de la cuadricula



Ahora necesitamos convertir este comportamiento básico en un juego.

Para hacer esto, necesitaremos mover el control a un lugar central creando un Game Controller. El controlador de juego establecerá el lado del jugador inicial, ya sea "X" u "O", hará un seguimiento de quién es el turno y cuando se hace clic en un botón, envía el lado del jugador actual, busca un ganador y cambia de lado o finaliza el juego.

Necesitaremos un nuevo script para hacer esto, así que creemos uno adjunto a su propio GameObject.

- Clic derecho en la ventana de "Hierachy" Create>Create Empty
- Le pondremos como nombre "GameController"
- Creamos un nuevo Script que se llame "GameController" también, haremos doble clic para editar el código

El script de GameController necesitará conocer todos los botones del juego para que pueda verificar su estado y determinar si ha sido una victoria. Para esto, el script debe contener una colección de todos los botones de Grid Space. Queremos verificar el componente de texto del botón para el lado del jugador para ver si hay 3 valores coincidentes en una fila, así que hagamos una matriz del tipo Texto.

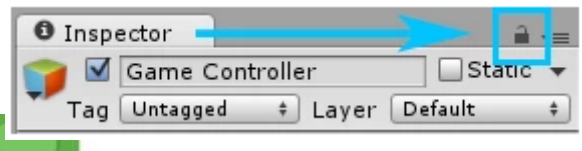
El guion final debería verse así:

```
using UnityEngine;
using UnityEngine.UI;
using System.Collections;
public class GameController : MonoBehaviour
{
    public Text[] buttonList;
}
```

Ahora que hemos creado esta matriz de Lista de botones, debemos completarla con los componentes de Texto de nuestros botones de Espacio de cuadrícula. Recuerde, ordenar aquí es importante. También es importante que vinculemos los GameObjects de texto secundarios con la matriz Lista de botones, no con los GameObjects de Grid Space primarios.

Una forma de llenar la matriz de la Lista de botones en el Controlador de juegos en el Inspector es establecer el tamaño de la matriz en 9 y luego arrastrar cada GameObject de texto, uno a la vez, a la matriz. Esto es un poco tedioso y consume mucho tiempo. Sin embargo, hay una manera más fácil de hacer esto, e implica "bloquear" la ventana del inspector y arrastrar todos los GameObjects a la vez a la matriz.

- Selecciona el botón del candado en Inspector del GameObject

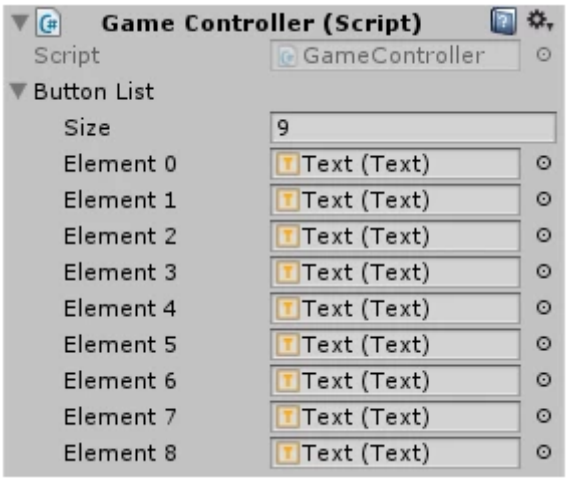


Lo que esto hace es evitar que la ventana del inspector cambie el foco de la selección actual, en este caso, el controlador del juego. Si no hiciéramos esto, cuando, en el siguiente paso, seleccionáramos los Objetos de Juego de Texto secundarios, el foco del Inspector cambiaría a estos Objetos de Juego de Texto secundarios y no podríamos arrastrarlos a la matriz en el Juego Controlador.

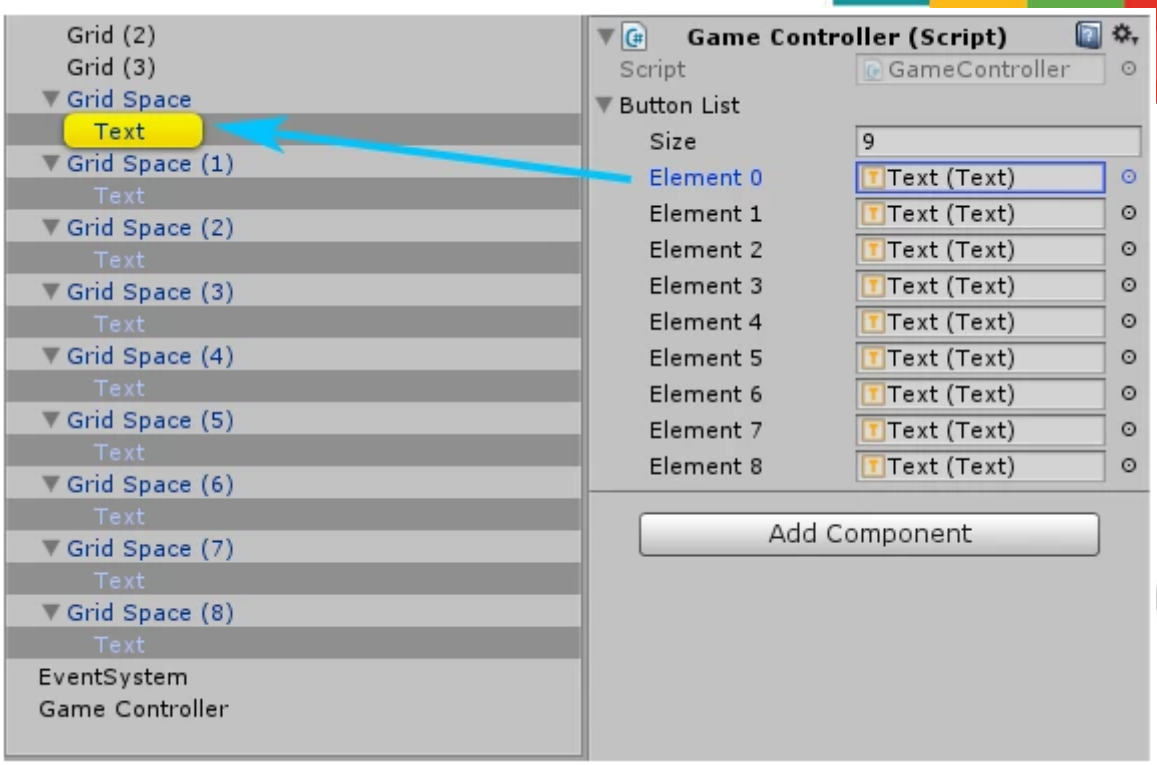
También vale la pena señalar que podemos abrir múltiples ventanas de Inspector, bloqueando solo las que queremos. Esto permite vistas complejas de varios GameObjects en la Jerarquía y Proyecto Windows.

- Abre 9 espacios para los 9 botones en el Inspector haciendo clic en el símbolo "+"
- Arrastra al bloque que le hemos puesto el candado la lista de botones.

Cuando el cursor de arrastre está en la posición correcta, aparecerá un pequeño icono "+" al lado del cursor.



Debemos asegurarnos de que los elementos están colocados en el orden correcto.



Ahora tenemos que avisar al juego de que el jugador ha efectuado un movimiento, para ello debemos informar al controlador de juego que se ha realizado un movimiento y se deben verificar las condiciones de victoria.

Abrimos nuestro Script y colocamos el código.

```
using UnityEngine;
using System.Collections;
using UnityEngine.UI;

public class GridSpace : MonoBehaviour
{

    public Button button;
    public Text buttonText;

    private GameController gameController;

    public void SetGameControllerReference(GameController controller)
    {
        gameController = controller;
    }

    public void SetSpace()
    {
        buttonText.text = gameController.GetPlayerSide();
        button.interactable = false;
        gameController.EndTurn();
    }
}
```

Probamos que todo funcione de forma correcta.

4. CREANDO TURNOS

Necesitamos probar los espacios de la cuadrícula en el tablero y ver si ha habido una victoria. No tenemos los lados del jugador correctamente configurados, pero necesitaremos verificar la propiedad Text de los espacios de la cuadrícula para ver si los valores de la cadena coinciden con "tres en una línea" y si estos valores coinciden con el lado que se está jugando actualmente. Para hacer esto, tendremos que poner el siguiente código.

```
using UnityEngine;
using UnityEngine.UI;
using System.Collections;

public class GameController : MonoBehaviour
{

    public Text[] buttonList;

    private string playerSide;

    void Awake()
    {
        SetGameControllerReferenceOnButtons();
        playerSide = "X";
    }

    void SetGameControllerReferenceOnButtons()
    {
        for (int i = 0; i < buttonList.Length; i++)
        {

buttonList[i].GetComponentInParent<GridSpace>().SetGameControllerReference(this);
        }
    }

    public string GetPlayerSide()
    {
        return playerSide;
    }

    public void EndTurn()
    {
        if (buttonList[0].text == playerSide && buttonList[1].text == playerSide
&& buttonList[2].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[3].text == playerSide && buttonList[4].text == playerSide
&& buttonList[5].text == playerSide)
        {
            GameOver();
```

```
        }

        if (buttonList[6].text == playerSide && buttonList[7].text == playerSide
&& buttonList[8].text == playerSide)

        {
            GameOver();
        }

        if (buttonList[0].text == playerSide && buttonList[3].text == playerSide
&& buttonList[6].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[1].text == playerSide && buttonList[4].text == playerSide
&& buttonList[7].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[2].text == playerSide && buttonList[5].text == playerSide
&& buttonList[8].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[0].text == playerSide && buttonList[4].text == playerSide
&& buttonList[8].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[2].text == playerSide && buttonList[4].text == playerSide
&& buttonList[6].text == playerSide)
        {
            GameOver();
        }
    }

    void GameOver()
    {
        for (int i = 0; i < buttonList.Length; i++)
        {
            buttonList[i].GetComponentInParent<Button>().interactable = false;
        }
    }
}
```

Ahora, todas las condiciones ganadoras posibles deberían funcionar y cuando se hayan cumplido las condiciones ganadoras, el tablero debería estar desactivado.

Luego, necesitamos cambiar de bando cuando se realiza un turno y poder jugar contra otra persona.

En la actualidad no tenemos realmente un juego. Podemos verificar las condiciones de victoria y cuando obtenemos tres "X" seguidas, el juego termina. Ahora necesitamos poder cambiar de bando. Para cambiar de lado, necesitaremos verificar de qué lado estamos y alternar los valores del jugador; intercambiando "X" por "O" o viceversa.

Para hacer esto, creemos una función que cambie los lados del jugador.

```
using UnityEngine;
using UnityEngine.UI;
using System.Collections;

public class GameController : MonoBehaviour
{

    public Text[] buttonList;

    private string playerSide;

    void Awake()
    {
        SetGameControllerReferenceOnButtons();
        playerSide = "X";
    }

    void SetGameControllerReferenceOnButtons()
    {
        for (int i = 0; i < buttonList.Length; i++)
        {

buttonList[i].GetComponentInParent<GridSpace>().SetGameControllerReference(this);
        }
    }

    public string GetPlayerSide()
    {
        return playerSide;
    }

    public void EndTurn()
    {

        if (buttonList[0].text == playerSide && buttonList[1].text == playerSide
&& buttonList[2].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[3].text == playerSide && buttonList[4].text == playerSide
&& buttonList[5].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[6].text == playerSide && buttonList[7].text == playerSide
&& buttonList[8].text == playerSide)
        {
            GameOver();
        }
```

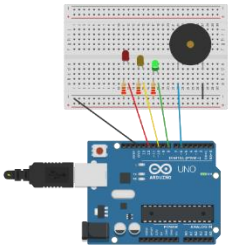
```
        if (buttonList[0].text == playerSide && buttonList[3].text == playerSide
&& buttonList[6].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[1].text == playerSide && buttonList[4].text == playerSide
&& buttonList[7].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[2].text == playerSide && buttonList[5].text == playerSide
&& buttonList[8].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[0].text == playerSide && buttonList[4].text == playerSide
&& buttonList[8].text == playerSide)
        {
            GameOver();
        }

        if (buttonList[2].text == playerSide && buttonList[4].text == playerSide
&& buttonList[6].text == playerSide)
        {
            GameOver();
        }

        ChangeSides();

    }

    void ChangeSides()
    {
        playerSide = (playerSide == "X") ? "O" : "X";
    }

    void GameOver()
    {
        for (int i = 0; i < buttonList.Length; i++)
        {
            buttonList[i].GetComponentInParent<Button>().interactable = false;
        }
    }
}
```

Ahora, cuando hacemos clic en cualquier espacio, los turnos se alternan entre "X" y "O", y si obtenemos tres "X" o tres "O" seguidas, cumplimos las condiciones de victoria y el juego ha terminado.

¡Nuestro juego está esencialmente terminado! En esta etapa, podemos jugar al Tic-Tac-Toe.

¿TE ATREVES CON ESTOS RETOS?

1. RETO 1 2 3 CIRCUIT: ENCENDER EL ZUMBADOR



30 MINUTOS	12-14 AÑOS	DIFICULTAD BAJA

¿Te atreves a hacer tu sólo tu propio circuito?

MATERIALES

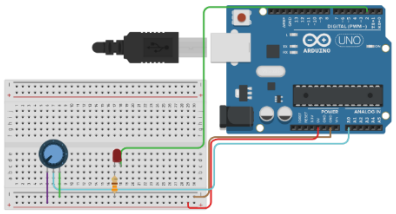
- Ordenador

RETO

Ahora que ya sabes cómo hacer circuitos con el programa, te proponemos un reto: hacer el montaje de un circuito igual al que hemos hecho en el ejercicio donde sólo encendíamos el LED sin programación, pero, ahora, sustituyendo el LED por el zumbador.

¿Serás capaz? ¡Enséñanos como funciona mandándonos un vídeo al WhatsApp que tienen a disposición en tu casa para ponerse en contacto con nosotros!

2. RETO 1 2 3 CIRCUIT: EL POTENCIÓMETRO



40 MINUTOS	12-14 AÑOS	DIFICULTAD MEDIA

¿Te atreves a hacer tu sólo tu propio circuito?

MATERIALES

- Ordenador

RETO

Ahora que ya sabes cómo hacer circuitos con el programa, te proponemos un reto: toma como referencia el ejercicio donde hemos montado el LED con el potenciómetro y ahora sustituye el LED por el zumbador y haz que suba y baje el sonido del zumbador con el movimiento del potenciómetro.

¿Lo has conseguido? ¡Enséñanos como funciona mandándonos un vídeo al WhatsApp que tienen a disposición en tu casa para ponerse en contacto con nosotros!

3. RETO UNITY: CAMBIO DE FORMA DEL TABLERO

40 MINUTOS	12-14 AÑOS	DIFICULTAD MEDIA

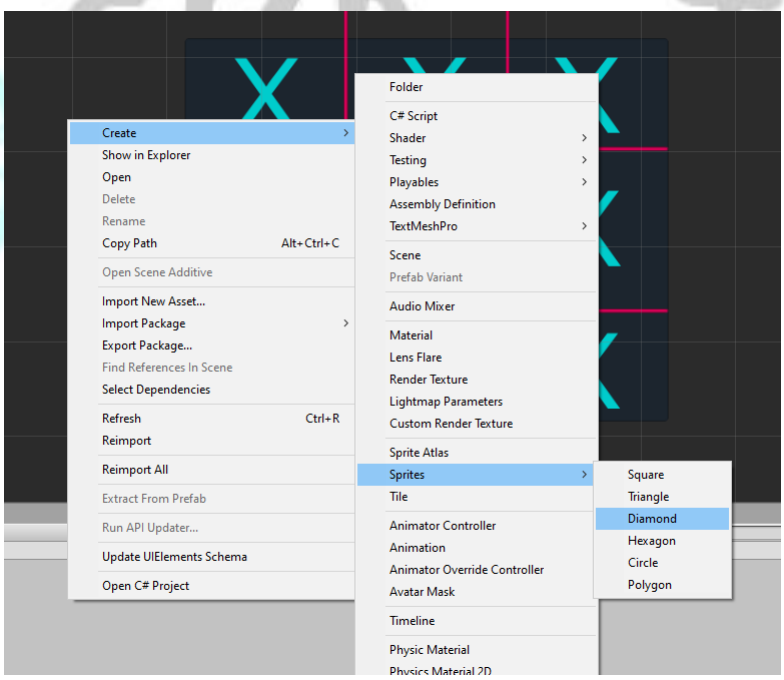
Vamos a intentar cambiar la forma de nuestro tablero en Unity.

MATERIALES

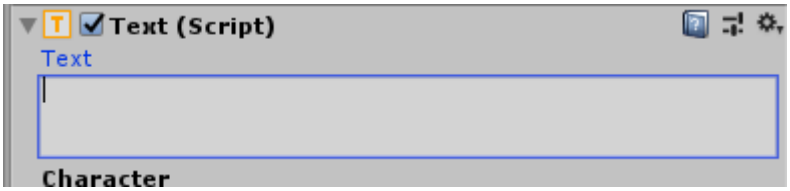
- Ordenador

RETO

Para poder cambiar el tablero y personalizarlo con las formas que más nos gusten debemos poner una forma geométrica que nos guste. Para ello, añadiremos un nuevo Asset, haremos clic derecho en la zona de los Assets en los proyectos y seleccionaremos la siguiente opción.



En las opciones de los botones, debemos quitar el texto que nos aparece para que pueda aparecer nuestra nueva forma geométrica.



A continuación, debemos colocar las piezas y los "grid" posicionados de forma correcta para que todo funcione de la mejor forma posible.

4. RETO UNITY: AÑADIR SONIDO

40 MINUTOS	12-14 AÑOS	DIFICULTAD MEDIA

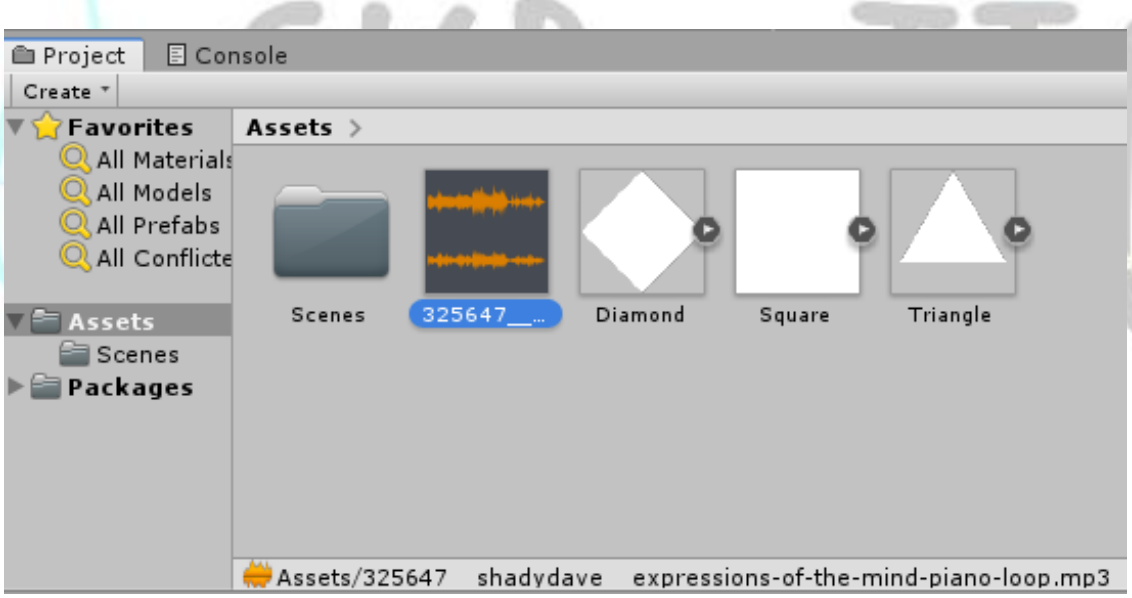
Vamos a intentar añadir sonido a nuestro juego.

MATERIALES

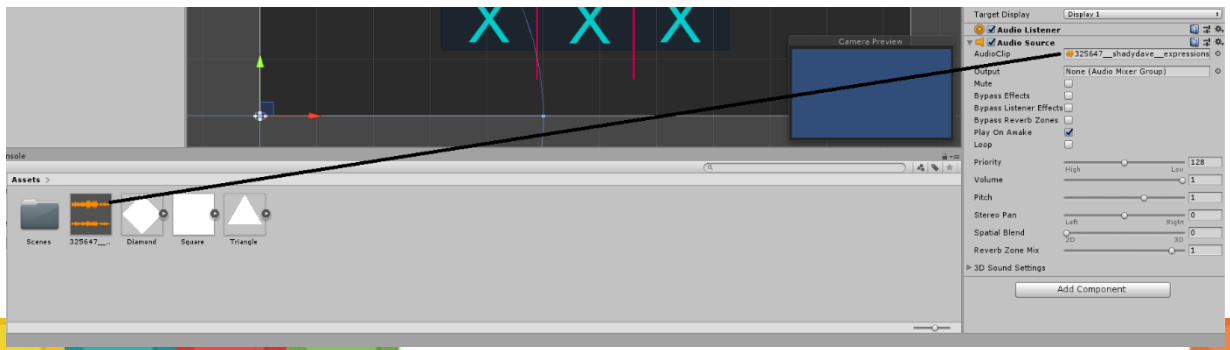
- Ordenador

RETO

Descargaremos una canción que nos guste para nuestro juego y la importaremos a Unity, para hacerlo únicamente tenemos que arrastras nuestro archivo a parte de abajo de Unity.



Una vez importada nuestra canción lo que debemos hacer es añadir un par de componentes a nuestra "Main Camera", pulsaremos sobre el botón de "ADD COMPONENT" y acto seguido añadiremos las opciones de "Audio Listener" y "Audio Source", ujna vez añadidos arrastraremos nuestra canción a la opción de "Audio Clip" y una vez que iniciemos nuestro juego dándole al botón play, la música y el juego se empezaran a reproducir.



PARA MÁS DIVERSIÓN…

TALLER: HACER UN VEHÍCULO PROPULSADO POR AIRE

120 MINUTOS	12-14 AÑOS	DIFICULTAD ALTA

Utilizando la tercera ley de Newton, el principio de acción-reacción, vamos a hacer un coche propulsado con el aire que sale al desinflarse un globo. Esta ley dice que siempre que un cuerpo ejerce una fuerza sobre otro, éste a su vez ejerce una fuerza igual, pero de sentido contrario al primero.

PREPARACIÓN

- Preparar el material que necesitamos la para la actividad
- Podemos ver un vídeo en youtube con esta manualidad aquí: https://www.youtube.com/watch?v=qRgUJfQ4OL0

MATERIALES

- Cuatro tapones de botella de plástico para las ruedas.
- Barrena, punzón o clavo caliente para hacer agujeros en los tapones.
- Dos palitos de barbacoa. Serán los ejes de las ruedas.
- Dos pajitas para albergar los ejes de las ruedas.
- Un globo.
- Una pajita flexible que irá unida al globo.
- Un trozo de cartón o plástico como el de las bandejas de alimentos.
- Cinta adhesiva o de carrocero.
- Tijeras o cúter.
- Material para decorar el coche. Nosotros usamos goma eva, pegamento y grapadora.

¡EMPEZAMOS LA ACTIVIDAD!

1. Haz un agujero en el centro del tapón que sea lo suficientemente grande como para que el palito de barbacoa pase a través de él. Para que las ruedas giren correctamente es muy importante que los agujeros se hagan en el centro. Coloca una rueda en cada pincho.

2. Recorta un rectángulo de cartón o plástico para hacer la base del coche. No lo hagas muy ancho, funcionará mejor.

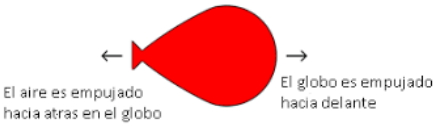
3. Pega con cinta adhesiva las pajitas a la base del coche y recorta a medida. Dentro irán los ejes de las ruedas, por lo que es muy importante que ambas queden paralelas y muy bien fijadas a la base. Introduce el palito de barbacoa en la pajita, corta según la medida que necesites y coloca la otra rueda. Repite el proceso con el otro eje.

4. Prueba el coche para asegúrate de que las ruedas giran bien y que no se tuerce demasiado. Realiza las mejoras necesarias.

5. Conecta la pajita, por el extremo flexible, a la boca del globo de forma que éste se pueda inflar soplando a través de la pajita. Puedes usar cinta adhesiva o una goma elástica para la conexión, lo importante que no haya fugas de aire. Más adelante, el conjunto pajita globo irá pegado a la base del coche de manera que el aire que vaya saliendo de la pajita al deshincharse el globo sea capaz de impulsar al coche.

6. Diseña la carrocería con los materiales que prefieras teniendo en cuenta que debes dejar espacio para colocar la pajita con el globo.

7. Cuando ya lo tengas todo, infla el globo, tapa la pajita para que el aire no escape, coloca el coche en el suelo y ¡a rodar!



¿Puedes hacer que vaya más deprisa?, ¿Más recto?, ¿Más lejos? Introduce las mejoras necesarias o cambia el diseño por completo.

EXPLICACIÓN

Las fuerzas resultan de la interacción entre dos cuerpos y siempre se presentan en pares. Cuando un cuerpo ejerce una fuerza sobre otro, éste a su vez ejerce una fuerza sobre el primero de igual magnitud y dirección, pero de sentido contrario. Si un cuerpo estuviera aislado no podría realizar ninguna fuerza ni tampoco recibirla. Se trata de la tercera ley del movimiento propuesta por Isaac Newton a finales del siglo XVII.

En nuestro coche, las paredes del globo empujan al aire de su interior, que acaba saliendo por la pajita, y el aire empuja al globo en sentido contrario. Como el globo está unido al coche, éste acaba moviéndose en sentido opuesto al aire.

HACIENDO ANIMACIONES CON PISKEL

60 MINUTOS | 12-14 AÑOS | DIFICULTAD ALTA

Vamos a hacer nuestras propias animaciones con sprites y pixel a través de esta herramienta tan interesante.

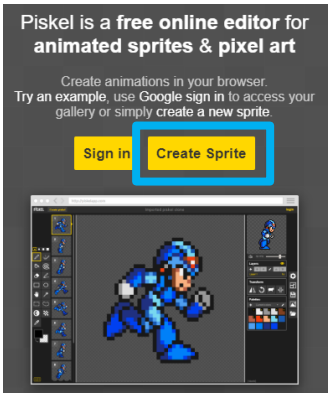
PREPARACIÓN

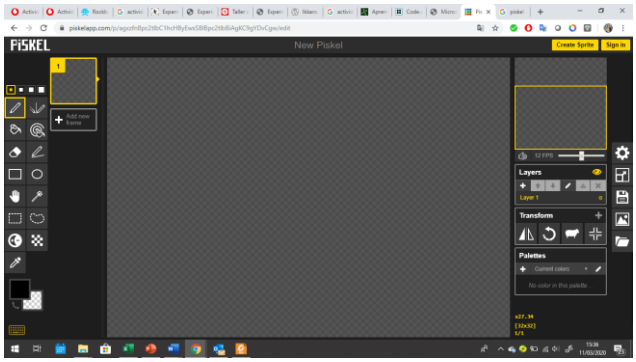
- Necesitaremos acceso a internet
- Para guardar nuestras creaciones online y seguir editando necesitaremos crearnos una cuenta.

MATERIALES

- Ordenador

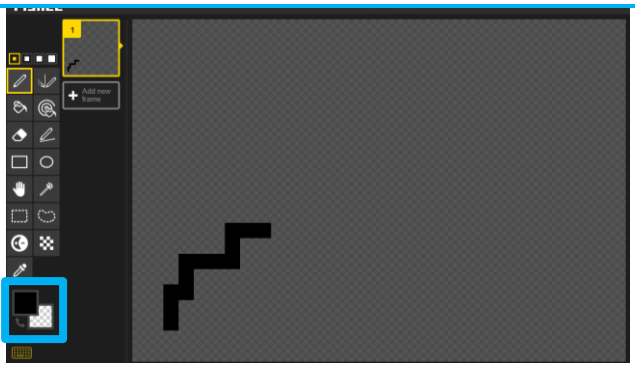
¡EMPEZAMOS LA ACTIVIDAD!



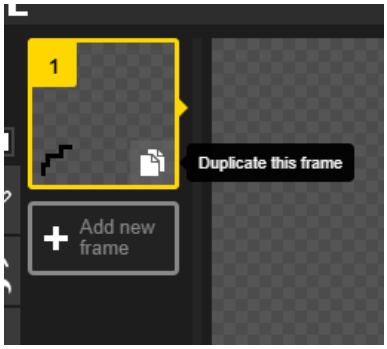


1. Entra en https://piskelapp.com Y pulsa sobre CREATE SPRITE

2. Nos aparecerá la pantalla principal donde vamos a crear nuestras animaciones. Para pintar, debemos seleccionar el lápiz y el grosor que deseemos.



3 Creamos un dibujo con los pixeles y el lápiz, escogiendo el color y el grosor deseado.

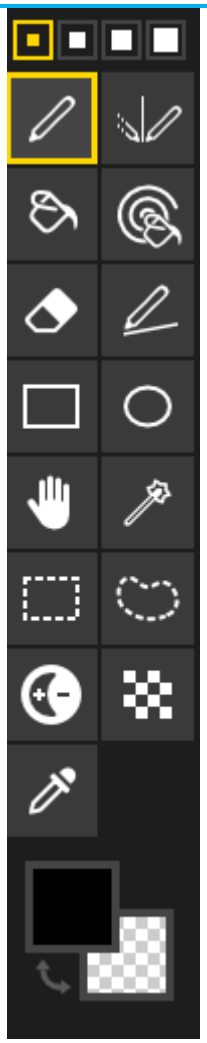


4 Añadimos un nuevo FRAME para crear el fotograma siguiente de nuestra animación.

5 Si queremos trabajar sobre el mismo dibujo que hemos hecho, pulsaremos sobre DUPLICATE THIS FRAME para que duplique el dibujo.

5 En la pantalla superior derecha podemos previsualizar como está quedando nuestra animación, así como modificar la velocidad (frames por segundo-FPS).

6 Una vez tengamos nuestra animación, la podemos guardar en formato GID y compartirla con nuestros amigos y amigas.



HERRAMIENTAS DE TRABAJO

A la izquierda tenemos el menú con las herramientas:

De izquierda a derecha:

1. Lápiz para dibujar
2. Lápiz para dibujar en espejo
3. Cubo de pintura
4. Cubo para pintar todos los píxeles de una capa de un frame del mismo color
5. Goma de borrar
6. Herramienta para hacer líneas rectas
7. Herramienta para hacer rectángulos
8. Herramienta para hacer círculos
9. Herramienta para mover objetos
10. Varita mágica de selección
11. Herramienta de selección rectangular
12. Lazo de selección
13. Herramienta de luminosidad
14. Herramienta para pintar píxeles
15. Cuentagotas para seleccionar colores
16. Selección de color
17. Atajos de teclado

¡Ahora te toca investigar a ti! ¿Qué tal ha quedado tu animación? ¡Enséñanosla mandándonos el gif al WhatsApp que tienen a disposición en tu casa para ponerse en contacto con nosotros!

APRENDE A PROGRAMAR EN PHYTON CON CODE COMBAT

120 MINUTOS | 10-14 AÑOS | DIFICULTAD ALTA

Aprender a programar con Phyton no es fácil…pero con este juego es mucho más divertido. ¡Pruébalo!

DESCRIPCIÓN

Python es uno de los lenguajes de programación más robusto y utilizado en el mundo, pero su principal ventaja gira en torno a su sencilla sintaxis que permite que aprender a programar en python sea bastante sencillo. Incluso, existe una herramienta llamada CodeCombat que nos permite conocer a profundidad las maravillas de este lenguaje mientras jugamos en una aventura bastante divertida.

PREPARACIÓN

- Necesitaremos internet para poder acceder a la siguiente página web: https://codecombat.com/

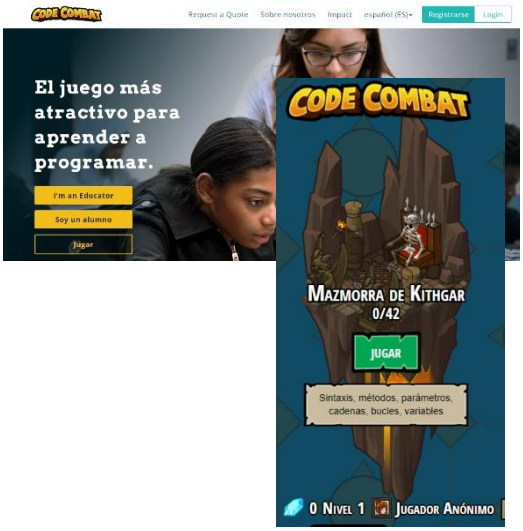
MATERIALES

- Ordenador

¡EMPEZAMOS LA ACTIVIDAD!

1. Entra en la página web de cocecombat y haz click sobre jugar.

   Una vez dentro, haz click en el primer mundo desbloqueado “mazmorra de kihgar”



2 Al empezar el juego debes seleccionar a tu héroe o heroína

3 Y también tendremos que seleccionar el equipamiento.

Una vez dentro nos encontraremos con una pantalla con los siguientes elementos:

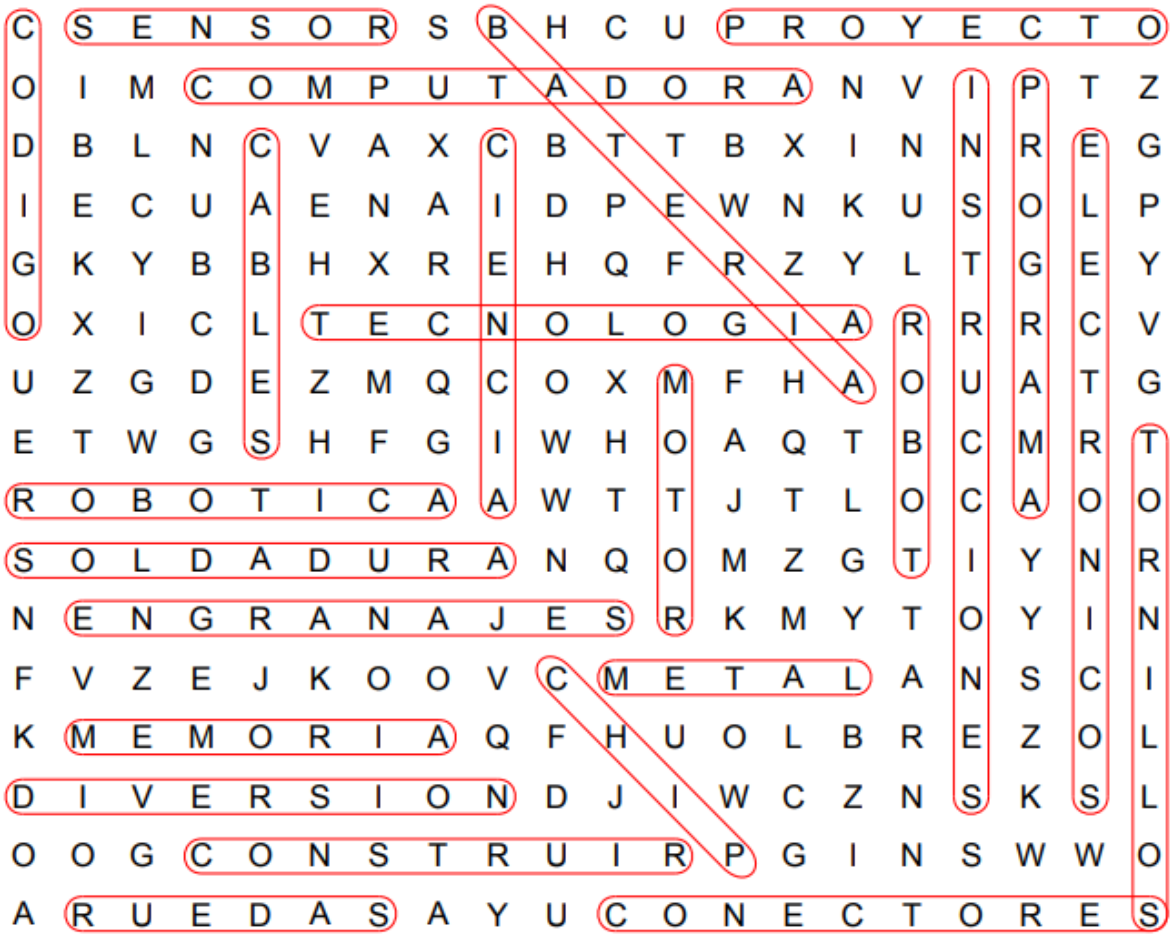
1. Escribe el código en el editor de código
2. Pulsa sobre RUN o CORRER para ver cómo funciona tu código línea por línea.
3. Mira tu código en acción.
4. Despliega esta lista para ver ejemplos de cómo usar el código.
5. Si estás bloqueado en HINTS puedes ver algunos consejos y ayudas.
6. Si te has equivocado con el código, puedes dar a RELOAD para volver a empezar.
7. Asegúrate de completar todos los objetivos de cada nivel.

El programa de dará pistas sobre cómo escribir el código y poco a poco irás completando distintos niveles.

¡PÁSALO GENIAL Y APRENDE MUCHO!

SOLUCIONES ACTIVIDADES DE REPASO

ACTIVIDAD 1. SOPA DE LETRAS DE ROBOTS



ACTIVIDAD 3. ¿CUÁNTO SABES DE ROBÓTICA?

1. Sensores
2. Multidisciplinar
3. Una placa
4. Isaac Asimov
5. Algoritmo adaptativo
6. Robo de montaje
7. Un programa de diseño 3D
8. Un entorno de desarrollo de videojuegos en 3D y 2D
9. Nº de direcciones en las que se puede mover
10. Ser programable