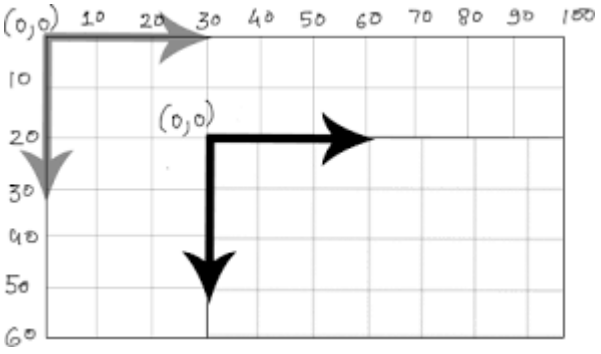
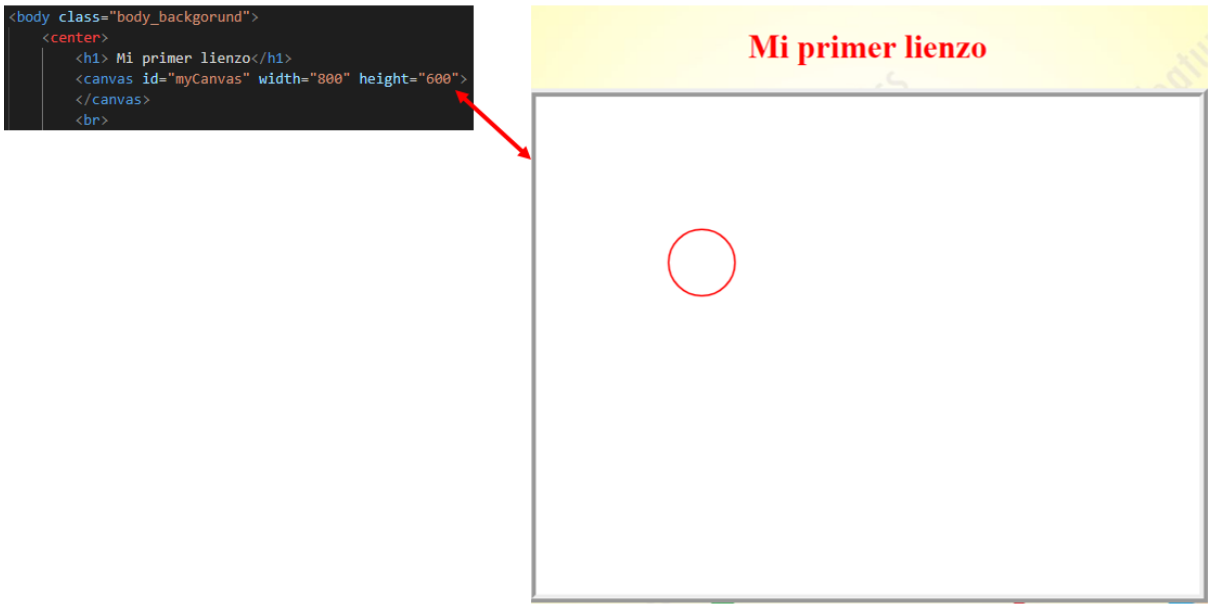
Coordenadas [Imagen -1]



Código HTML para el lienzo -

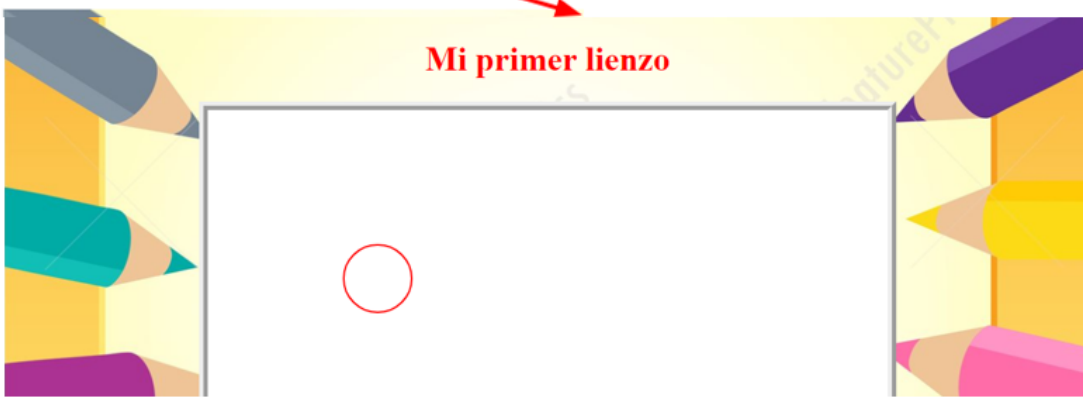
```
<body class="body_backgorund">
    <center>
        <h1> Mi primer lienzo</h1>
        <canvas id="myCanvas" width="800" height="600">
        </canvas>
        <br>
```

Output -



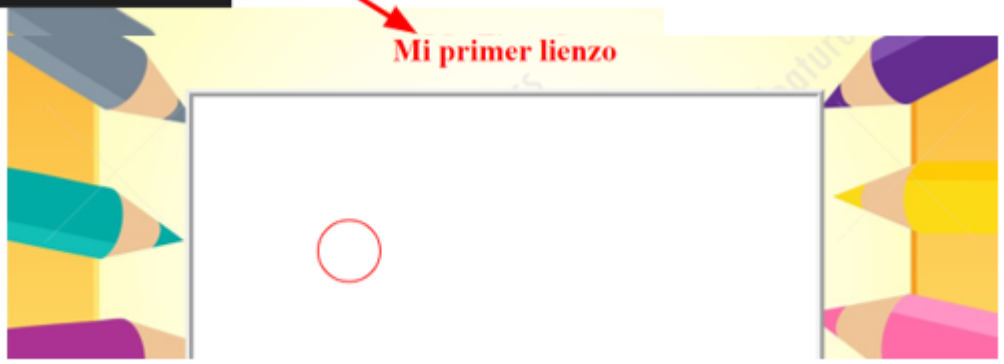
Estilo del fondo para la página web

```
.body_backgorund {
  background-image: url("bg1.jpg");
  background-position: center;
  background-size: cover;
```



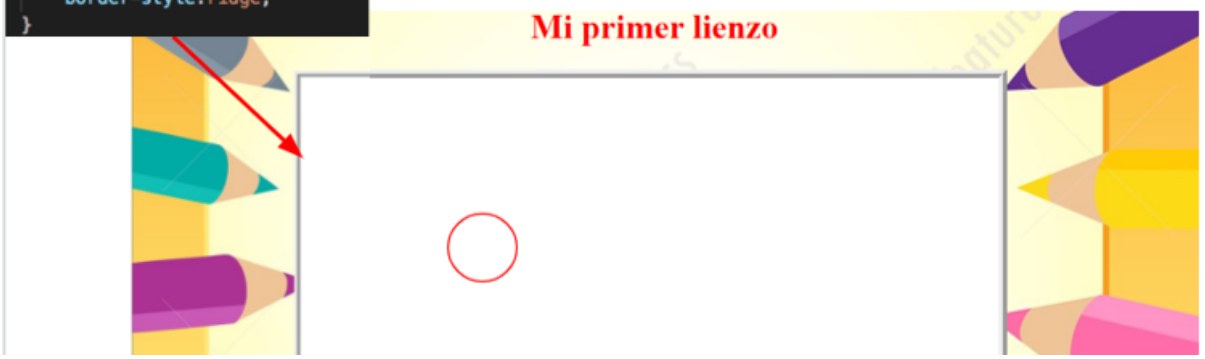
Estilo h1

```
h1
{
    color: red;
    font-size: 40px;
```



Estilo del lienzo

```
#myCanvas
{
    border-width:10px;
    background-color: white;
    border-style:ridge;
}
```



Código JS

Almacena el elemento HTML Canvas en una variable

```
canvas = document.getElementById("myCanvas");
```

```
color = "red";
```

Programación profesional

```
//Workable Code
canvas.getContext("2d").beginPath();
canvas.getContext("2d").strokeStyle =
color;
canvas.getContext("2d").lineWidth = 2;
canvas.getContext("2d").arc(200, 200,
40 ,0 , 2*Math.PI);
canvas.getContext("2d").stroke();
```

El código anterior funcionará, pero debemos usar un código más profesional, así que usaremos el siguiente

```
ctx= canvas.getContext("2d");
```

```
ctx.beginPath();
ctx.strokeStyle = color;
ctx.lineWidth = 2;
ctx.arc(200, 200, 40 ,0 , 2 * Math.PI);
ctx.stroke();
```

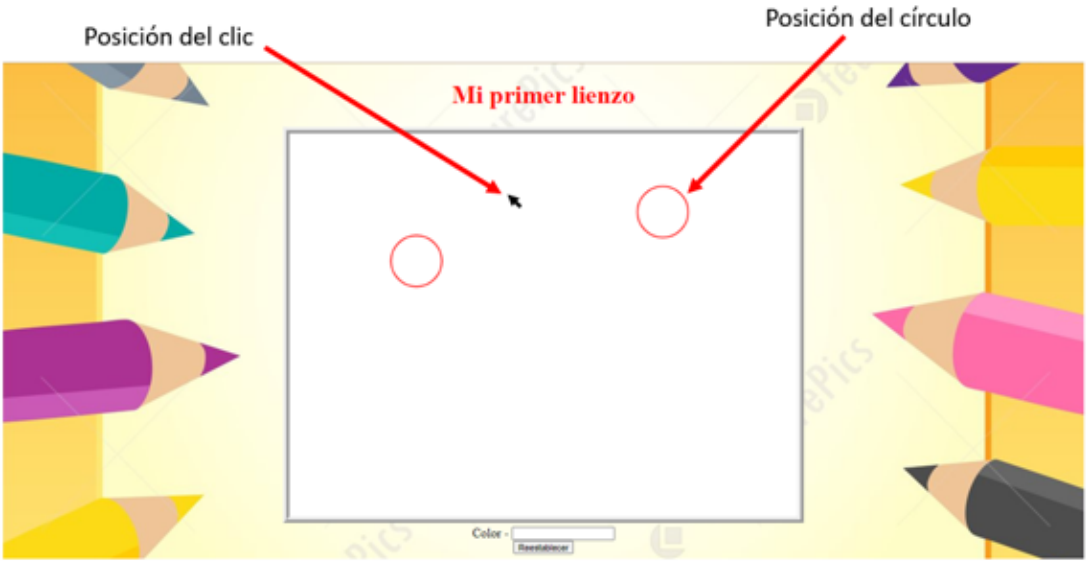
Como resultado de un código profesional, hemos reducido la línea del código.

Explicación de offset

Escribe el código así:

```
mouse_x = e.clientX
mouse_y = e.clientY
```

Ejecuta el código e intenta dibujar un círculo. Verás que, al hacer clic en el lienzo, el círculo no se dibuja en el lugar donde haces clic, sino que aparece en otro lado.
Por ejemplo:

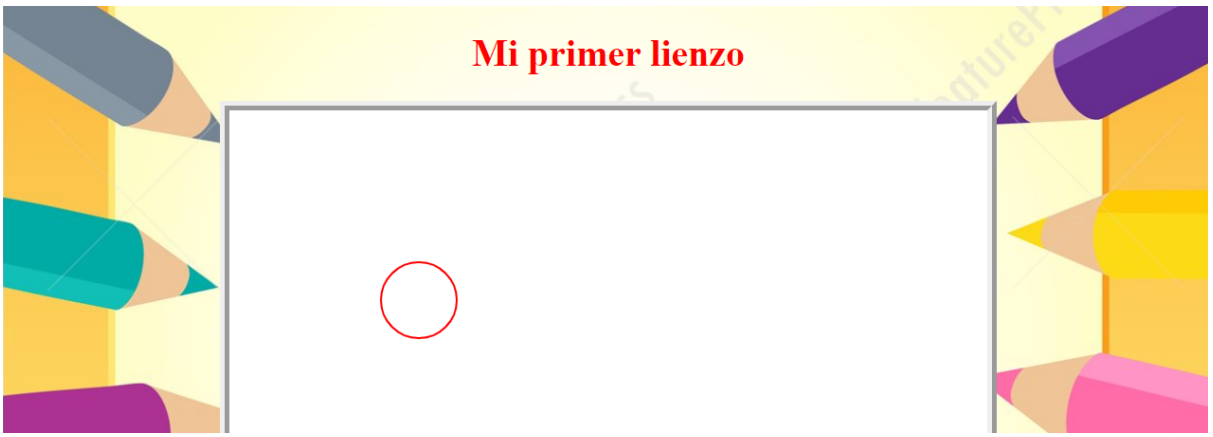


Es por esto que debemos usar las coordenadas para obtener la posición real.

Predefine el código para el círculo [Imagen -2]

```
ctx.beginPath();
ctx.strokeStyle = color;
ctx.lineWidth = 2;
ctx.arc(200, 200, 40 ,0 , 2 * Math.PI);
ctx.stroke();
```

Output



Explicación de addEventListener

Sintaxis:

```
element.addEventListener("event", my_function);
my_function(e)
{
//cualquier código
}
```

- **element** puede ser cualquier elemento HTML.
- **addEventListener**: se encuentra junto al elemento y cuando se activa el evento, corre la función. Es similar al bloque AddListener que utilizamos para la aplicación de chat. Lo utilizamos para bloquear el monitor si el mensaje de chat se envió a la firebase.
- **event -** Puede ser cualquier evento. Por ejemplo: click, mousemove, mousedown (es decir, cada que se haga clic en el mouse).
- **my_function**: Puede ser cualquier función. Cuando se active el evento, queremos que la función que definimos se active.
- **my_function(e):** Es la función que definimos. Esta función realizará las tareas que le hayamos escrito. **e** se refiere al evento de la función. **e** se relaciona con el evento.Por ejemplo, si el evento es **mousedown**, entonces esta **e** se debe relacionar con el evento **mousedown**.

Ejemplo:

```
button = document.getElementById("button");
button.addEventListener('click' , my_function);
function my_function(e) {
    console.log('Soy un botón ');
}
```

Primero tenemos un botón, el cual es un elemento HTML, y lo colocamos dentro de la **variable del botón**.
Después vamos a adjuntar la variable del botón al **addEventListener.**
Vamos a definir el tipo de evento, que será un **evento click.**
Then we will call our function which is **my_function.**
Ahora vamos a definir nuestra función **my_function**, la cual será el código para el texto "**Soy un botón**" en la pantalla de la consola.
Cuando se haga clic en el botón, la función correrá y mostrará "Soy un botón" en la consola.

ARCO

Sintaxis del arco (x, y, r, startAngle, endAngle);

x - La coordenada horizontal del centro del arco es la coordenada X.
y - La coordenada vertical del centro del arco es la coordenada Y.
r - El radio del arco.
startAngle - El ángulo en el que comienza el arco. Este se mide a partir del eje X.
endAngle - El ángulo donde termina el arco. Este se mide a partir del eje X.

Código addEventListener -

```
canvas.addEventListener("mousedown", my_mousedown);

function my_mousedown(e)
{
    //obtener color del cuadro de entrada
    //inicio de la actividad adicional
    color = document.getElementById("color").value;
    console.log(color);
    //final de la actividad adicional

     mouse_x = e.clientX - canvas.offsetLeft;
     mouse_y = e.clientY - canvas.offsetTop;

    console.log("X = " + mouse_x + " ,Y =  " + mouse_y);
    circle(mouse_x , mouse_y);
}
```

Output de -

```
console.log("X = " + mouse_x + " ,Y =  " + mouse_y);
```

```
X = 273 ,Y =  184                                    main.js:26
                                                     main.js:20
X = 631 ,Y =  278                                    main.js:26
```

Código del círculo -

```
function circle(mouse_x , mouse_y)
{
ctx.beginPath();
ctx.strokeStyle = color;
ctx.lineWidth = 2;
ctx.arc(mouse_x, mouse_y, 40 ,0 , 2*Math.PI);
ctx.stroke();
}
```

Experimenta para aprender más sobre el arco-
https://amdavalos.github.io/arco/