05

Diseñando con el mouse

El siguiente código permite dibujar círculos en la pantalla mientras el botón izquierdo se encuentra presionado, eso significa que mientras no soltemos el botón izquierdo, diseñaremos un círculo a lado de otro, que dará un efecto como si estuviéramos pasando un pincel por la pantalla. Si soltamos el botón izquierdo, la acción de mover el mouse sobre la pantalla no tendría que diseñar nada. Al final tendremos un efecto visto en herramientas como Paint de Windows.

El código es el siguiente:

```
<canvas width="600" height="400"></canvas>

<script>
    var pantalla = document.querySelector('canvas');
    var pincel = pantalla.getContext('2d');

    pincel.fillStyle = 'grey';
    pincel.fillRect(0, 0, 600, 400);

    var puedoDibujar = false;

    function dibujarCirculo(evento) {

        if(puedoDibujar) {
            var x = evento.pageX - pantalla.offsetLeft;
            var y = evento.pageY - pantalla.offsetTop;
            pincel.fillStyle = 'blue';
            pincel.beginPath();
```

```
            pincel.arc(x, y, 5, 0, 2 * 3.14);
            pincel.fill();
        }

    }

    pantalla.onmousemove = dibujarCirculo;

    function habilitarDibujar() {

        puedoDibujar = true;
    }

    function deshabilitarDibujar() {

        puedoDibujar = false;
    }

    pantalla.onmousedown = habilitarDibujar;

    pantalla.onmouseup = deshabilitarDibujar;

</script>
```

COPIA EL CÓDIGO

Para desarrollar este programa reutilizamos todo el código que ya teníamos para dibujar circunferencias, y solo adicionamos la lógica nueva de transformar nuestro mouse en un pincel.

Considera que la parte esencial está en saber si diseñamos o no en la pantalla mientras pasamos el mouse sobre la misma. Y sabemos que la condición está asociada a si el botón izquierdo del mouse está o no siendo presionado.

Siendo así declaramos una variable booleana llamada `puedoDibujar` que comienza como `false`. Esa variable será utilizada por la función `dibujarCirculo` para saber si debe o no diseñar.

En el código habrás visto tres eventos nuevos que estamos usando, ellos son `onmousemove`, `onmousedown` y `onmouseup`, donde el primero permite capturar el movimiento del mouse, el segundo sirve para ejecutar un código cuando el mouse está presionado y el tercero cuando el botón del mouse es soltado. Entonces, asociamos el primer evento a la función para dibujar circulo: `pantalla.onmousemove = dibujarCirculo`. Además, creamos dos funciones `habilitarDibujar` y `deshabilitarDibujar` y asociamos respectivamente cada función con los eventos del mouse `onmousedown` y `onmouseup`. Dentro de las funciones, lo que hacemos es atribuir la variable `puedoDibujar` para `true` cuando queremos `habilitarDibujar` y vuelve para `false` cuando queremos `deshabilitarDibujar`.

Copia y pega este código en tu editor de texto, guarda el archivo como `disenharConMouse.html` y experimenta su funcionalidad, abre la consola de desarrollador en tu navegador para evaluar y analizar el comportamiento de las variables y funciones que creamos.

Ahora vamos al desafío:

El desafío de este ejercicio es poder cambiar el color del pincel, haciendo clic en una paleta de colores que vamos a tener en el extremo izquierdo superior de nuestro `canvas`, vamos a disponibilizar 3 colores para que el usuario pueda escoger el color que quiera en su pincel, los colores que usaremos serán el rojo, verde y azul (`red`, `green` y `blue`). El usuario tiene que visualizar algo así:

Por lo tanto, cuando el usuario haga clic en el cuadrado verde, el pincel diseñará círculos verdes, cuando haga clic en el rojo, diseñará círculos rojos, y así sucesivamente, recordando que el color inicial (default) es el azul. Un punto importante a tomar en cuenta es que debemos restringir el área de la paleta para no poder diseñar nada encima de ella.

Consejos antes del ejercicio

Ya lo dijo el emperador romano Julio Cesar "Divide y reinarás", divide el problema en problemas menores e intenta resolverlo por etapas:

- Crear los cuadrados de la paleta (puedes definir las dimensiones de los cuadrados como 50 x 50)
- Crear la paleta de colores
- Crear la lógica que al momento de hacer clic el color del pincel cambie.

Este es el último ejercicio del curso, por ese motivo, no tiene tantas pistas y eso lo hace más desafiante, pero después de todo lo que vimos con seguridad que estás preparado para vencer este desafío satisfactoriamente, usa todos tus apuntes y anotaciones, concéntrate, tómate tu tiempo y luego compara tu solución con la del instructor.

Opinión del instructor

Vamos a seguir el consejo de nuestro amigo extinto, el emperador Julio César y vamos a dividir nuestro problema:

Crear los cuadrados de la paleta

Lo primero que resolveremos, será crear las variables y la función que usaremos para crear un cuadrado genérico.

```
    pantalla.onmousedown = habilitarDibujar;

    pantalla.onmouseup = deshabilitarDibujar;

</script>
```

COPIA EL CÓDIGO

En esta primera parte, definimos las coordenadas de nuestros cuadrados `var xRojo = 0` , `var xVerde = 50` , `var xAzul = 100` , todos ellos tienen la misma coordenada vertical `var yCuadrados = 0` , y por último, el tamaño de los cuadrados de 50 píxeles `var tamanhoCuadrados = 50` . Luego, creamos la función genérica `dibujarCuadrado()` que recibe como parámetro las coordenadas (x,y), el tamaño y el color del cuadrado. De momento, esta función `dibujarCuadrado()` no realiza nada porque no está siendo llamada aún.

Crear la paleta de colores

En una segunda etapa vamos a crear una función que dibuje la paleta de colores, que se encargue de llamar a la función `dibujarCuadrado` y le pase los parámetros correctos para graficar los 3 cuadrados, nuestro código quedaría así:

```
    function dibujarPaletaColores() {

        dibujarCuadrado(xRojo, yCuadrados, tamanhoCuadrados, '
        dibujarCuadrado(xVerde, yCuadrados, tamanhoCuadrados,
        dibujarCuadrado(xAzul, yCuadrados, tamanhoCuadrados, '

    }

    pantalla.onmousemove = dibujarCirculo;

    function habilitarDibujar() {

        puedoDibujar = true;

    }

    function deshabilitarDibujar() {

        puedoDibujar = false;

    }

    pantalla.onmousedown = habilitarDibujar;

    pantalla.onmouseup = deshabilitarDibujar;

    dibujarPaletaColores();

</script>
```

COPIA EL CÓDIGO

En esta segunda parte de nuestro problema, lo que hicimos fue crear la función `dibujarPaletaColores` que no recibe parámetros, pero que dentro de su bloque llama 3 veces a la función `dibujarCuadrado` y le pasa los parámetros correctos para que sean diseñados los cuadrados que componen la paleta de colores.

Crear la lógica que al momento de hacer clic el color del pincel cambie - Parte 1: Limitar el área de la Paleta

Ahora, en la tercera parte de nuestro problema, nos vamos a enfocar en la lógica para que al momento de hacer clic el color del pincel cambie, esta última parte, la vamos a subdividir en dos partes, la primera nos enfocaremos en no permitir que el usuario diseñe encima de nuestra paleta de colores, y una segunda parte en detectar cual cuadrado fue accionado con el clic del mouse.

Para resolver que el usuario no esté permitido de diseñar encima de nuestra paleta, aplicaremos la siguiente lógica, si el mouse se encuentra posicionado entre las coordenadas `X=0` y `X=3*(tamanhoCuadrados+5)` , y entre `Y=0` y `Y=(tamanhoCuadrados+5)` , no podrá diseñar, incluso cuando el botón izquierdo del mouse esté presionado. Toma en cuenta que sumamos el radio de la circunferencia también, porque en caso de que el usuario grafique en el borde inferior o borde derecho de la paleta corremos el riesgo de que se grafique una parte del círculo dentro de la paleta. Para aplicar la lógica mencionada, vamos a crear una función llamada `puedeDisenharArea` con todas las condiciones, esta función retornará `true` caso sea posible y `false` caso contrario. La función recibirá como parámetro las coordenadas `X` y `Y` del mouse.

Pero para poder accesar a las coordenadas del mouse necesitamos usar el evento `onmousemove` que es parte de nuestro objeto `pantalla` , dicho evento anteriormente estaba asociado directamente a la función `dibujarCirculo` , ahora lo vamos a asociar con una nueva función que crearemos: `capturarMovimientoDelMouse` y que recibirá como parámetro el evento de mover el mouse. En esta función `capturarMovimientoDelMouse` vamos a guardar las variables de las coordenadas `X` y `Y`, y dentro de la misma función vamos a crear un `if` para verificar si el mouse está localizado en el área restringida, si no estuviera vamos a llamar a la función `dibujarCirculo`.

Necesitamos, también, alterar la función `dibujarCirculo`, ya que ahora recibirá como parámetro las coordenadas (x,y) del clic del mouse cuando deseemos graficar.

Nuestro código hasta este punto está de la siguiente manera:

```
    }

    pantalla.onmousemove = capturarMovimientoDelMouse;

    pantalla.onmousedown = habilitarDibujar;

    pantalla.onmouseup = deshabilitarDibujar;

    dibujarPaletaColores();

</script>
```

COPIA EL CÓDIGO

Haz pruebas y verifica que no consigue diseñar por encima de la paleta de colores. Ahora, solo falta la ultima parte donde el usuario podrá escoger el color del pincel.

Crear la lógica que al momento de hacer clic el color del pincel cambie - Parte 2: Seleccionar color de la paleta

Para ello, lo primero que haremos será crear una variable `colorActual` y le atribuimos el color `blue` por defecto, luego vamos a detectar cuál color de la paleta fue seleccionado a través de la función `seleccionarColor` que asociaremos al evento `onclick` de nuestra pantalla. Voy a explicar la lógica para el color rojo, para los otros cuadrados la lógica es la misma. Si el usuario hace clic en la coordenada `X=15` y `Y=20` , necesitamos verificar si `X` es mayor que `xRojo` , que es la posición del eje `X` del cuadrado rojo, y además necesitamos verificar si `X` es menor que `xRojo + tamanhoCuadrados` porque ese es el range del cuadrado rojo en el eje horizontal. Pero también necesitamos verificar el eje vertical, para ello, vamos a comparar si `Y` es mayor que `yCuadrados` y que `Y` sea menor que `yCuadrados + tamanhoCuadrados` . Ojo, también necesitamos cambiar la función `dibujarCirculo` porque tuvimos que adicionar la variable del color seleccionado `colorActual` entre los parámetros recibidos, hicimos esa alteración en la declaración de la función `dibujarCirculo` y en la llamada a la función `dibujarCirculo` que está localizada dentro de la función `capturarMovimientoDelMouse` .

Nuestro código final quedó de la siguiente manera:

```
   pantalla.onmousedown = habilitarDibujar;

   pantalla.onmouseup = deshabilitarDibujar;

   dibujarPaletaColores();

   pantalla.onclick = seleccionarColor;

/script>
```

COPIA EL CÓDIGO

El código final quedó un poco extenso, con seguridad que algunas etapas e instrucciones podrían ser realizadas con un código más optimizado, solo que para poder explicar paso a paso tuvimos que hacerlo así. Existen innúmeras formas de resolver este problema, ésta es solo una de las tantas, espero que tu versión haya sido más innovadora que la del instructor.