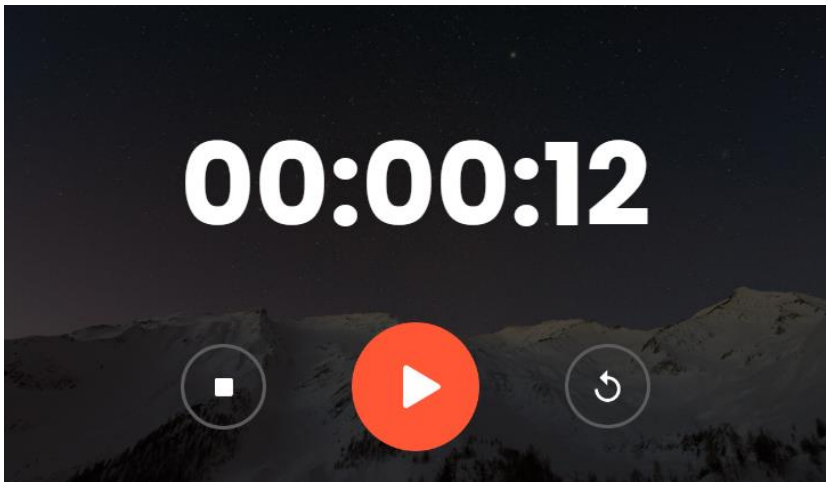
Cronometro



Aprendices a continuación realizaremos un pequeño cronometro que nos permitirá jugar con los controles y manipulación del DOM, vamos con nuestro código HTML:

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Cronometro</title>
    <link rel="stylesheet" href="style.css">
</head>

<body>
    <div class="stopWatch">
        <h1 class="time">00:00:00</h1>
        <div class="controls">
            <img src="images/stop.png" onclick="stopTimer()" alt="stop">
            <img src="images/start.png" onclick="startTimer()" alt="start">
            <img src="images/reset.png" onclick="resetTimer()" alt="reset">
        </div>
    </div>

    <script src="index.js"></script>
</body>

</html>
```

Este código HTML muestra tres imágenes dentro de la página web, cada una de ellas con un atributo **onclick** que hace referencia a una función JavaScript diferente cuando se hace clic en ellas. A continuación, te explico cada parte del código:

1. **<img src="images/stop.png" onclick="stopTimer()" alt="stop">** Esta línea muestra una imagen con la ruta de origen (**src**) establecida en "images/stop.png". Cuando se hace clic en la imagen, se llama a la función **stopTimer()**. El atributo **alt** se utiliza para proporcionar un texto alternativo para la imagen en caso de que no se pueda cargar o se utilicen lectores de pantalla.

2. **<img src="images/start.png" onclick="startTimer()" alt="start">** Esta línea muestra otra imagen con la ruta de origen (**src**) establecida en "images/start.png". Cuando se hace clic en la imagen, se llama a la función **startTimer()**. Al igual que en el ejemplo anterior, el atributo **alt** proporciona un texto alternativo para la imagen.

3. **<img src="images/reset.png" onclick="resetTimer()" alt="reset">** Esta línea muestra una tercera imagen con la ruta de origen (**src**) establecida en "images/reset.png". Cuando se hace clic en la imagen, se llama a la función **resetTimer()**. El atributo **alt** proporciona un texto alternativo para la imagen.

En resumen, este código HTML muestra tres imágenes que, cuando se hace clic en ellas, activan diferentes funciones de JavaScript (**stopTimer()**, **startTimer()**, **resetTimer()**) según la imagen en particular.

Las imágenes que se están incluyendo en el ejemplo son para que ustedes busquen unas similares y las agreguen a su gusto.

Vamos con el CSS:

```css
@import url("https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;300;400;500;600;700;800;900&display=swap");

* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "Poppins", sans-serif;
}

body {
  background-color: white;
}

.stopWatch {
  background-image: linear-gradient(rgba(0, 0, 0, 0.7), rgba(0, 0, 0, 0.7)),
    url(images/background.png);
  max-width: 600px;
  background-size: cover;
  background-position: center;
  padding: 40px 0;
  margin: 200px auto 0;
  text-align: center;
  color: white;
}

.stopWatch h1 {
  font-size: 80px;
  margin: 50px 0;
}
```

```css
.controls {
  display: flex;
  align-items: center;
  justify-content: center;
}

.stopWatch img {
  width: 60px;
  margin: 0 30px;
  cursor: pointer;
}

.stopWatch img:nth-child(2) {
  width: 90px;
}
```

1. **@import url("https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;300;400;500;600;700;800;900&display=swap");** Esta línea importa la fuente de Google Fonts llamada "Poppins" con diferentes pesos (100 a 900) y la hace disponible para su uso en el resto del código.

2. *** { ... }** Este bloque de código establece que se apliquen las siguientes propiedades a todos los elementos de la página:

    - **margin: 0;** establece los márgenes en 0.
    - **padding: 0;** establece el relleno en 0.
    - **box-sizing: border-box;** establece el modelo de caja como "border-box", lo que significa que el tamaño total de un elemento incluye el contenido, el relleno y el borde, pero no el margen exterior.
    - **font-family: "Poppins", sans-serif;** establece la fuente de la familia "Poppins" como la fuente utilizada para el texto en toda la página.

3. **body { ... }** Este bloque de código define las propiedades para el elemento **body**, que es el cuerpo principal de la página:

    - **background-color: white;** establece el color de fondo del cuerpo en blanco.

4. **.stopWatch { ... }** Este bloque de código define las propiedades para un elemento con la clase "stopWatch", que representa el área del cronómetro:

    - **background-image: linear-gradient(...), url(images/background.png);** establece una imagen de fondo que es una superposición de un degradado y una imagen llamada "background.png" ubicada en la carpeta "images". El degradado crea un fondo semitransparente.
    - **max-width: 600px;** establece el ancho máximo del elemento en 600 píxeles.
    - **background-size: cover;** ajusta el tamaño de la imagen de fondo para cubrir completamente el área del elemento.
    - **background-position: center;** centra la imagen de fondo horizontalmente y verticalmente.
    - **padding: 40px 0;** establece el relleno superior e inferior del elemento en 40 píxeles y el relleno izquierdo y derecho en 0.
    - **margin: 200px auto 0;** establece los márgenes superior, izquierdo y derecho del elemento.
    - **text-align: center;** centra el texto en el elemento.
    - **color: white;** establece el color del texto en blanco.

5. **.stopWatch h1 { ... }** Este bloque de código define las propiedades para el elemento **h1** dentro del elemento con la clase "stopWatch":

    - **font-size: 80px;** establece el tamaño de fuente en 80 píxeles.

- **margin: 50px 0;** establece los márgenes superior e inferior del elemento en 50 píxeles y el margen izquierdo y derecho en 0.

6. **.controls { ... }** Este bloque de código define las propiedades para un elemento con la clase "controls", que contiene los controles del cronómetro:
    - **display: flex;** establece el contenedor como un contenedor flexible.
    - **align-items: center;** alinea los elementos secundarios verticalmente en el centro.
    - **justify-content: center;** centra los elementos secundarios horizontalmente.
7. **.stopWatch img { ... }** Este bloque de código define las propiedades para las imágenes dentro del elemento con la clase "stopWatch":
    - **width: 60px;** establece el ancho de las imágenes en 60 píxeles.
    - **margin: 0 30px;** establece los márgenes izquierdo y derecho de las imágenes en 30 píxeles y el margen superior e inferior en 0.
8. **.stopWatch img:nth-child(2) { ... }** Este bloque de código establece propiedades adicionales para la segunda imagen dentro del elemento con la clase "stopWatch":
    - **width: 90px;** establece el ancho de la segunda imagen en 90 píxeles.

Ahora vamos con nuestro JS:

```
let timeEl = document.querySelector(".time");
let timer = null;
let [hours,minutes,seconds] = [0,0,0];

function stopWatch(){
    seconds++;
    if(seconds==60){
        seconds=0;
        minutes++;
        if(minutes==60){
            minutes=0;
            hours++;
        }
    }

    let h = hours<10 ? ("0"+hours) : hours;
    let m = minutes<10 ? ("0"+minutes) : minutes;
    let s = seconds<10 ? ("0"+seconds) : seconds;

    timeEl.innerHTML = h+":"+m+":"+s;
}

function startTimer(){
    if(timer≠null){
        clearInterval(timer);
    }
    timer = setInterval(stopWatch,1000);
    console.log(timer)
}

function stopTimer(){
    clearInterval(timer);
    console.log(timer)
}

function resetTimer(){
    clearInterval(timer);
    [hours,minutes,seconds] = [0,0,0];
    timeEl.innerHTML = "00:00:00";
    console.log(timer)
}
```

@hdtoledo

Este código JavaScript controla el funcionamiento de un cronómetro en una página web. A continuación, te explico cada parte del código:

1. **let timeEl = document.querySelector(".time");** Esta línea selecciona el elemento del DOM con la clase "time" y lo almacena en la variable **timeEl**. Este elemento se utiliza para mostrar el tiempo transcurrido en el cronómetro.

2. **let timer = null;** Esta línea declara una variable **timer** y la inicializa con el valor **null**. Esta variable se utiliza para almacenar el identificador del intervalo del temporizador.

3. **[hours, minutes, seconds] = [0, 0, 0];** Esta línea utiliza la destructuración de matrices para declarar e inicializar tres variables: **hours**, **minutes** y **seconds**. Cada una de ellas se inicializa con el valor **0**.

4. **function stopWatch() { ... }** Esta función se ejecuta cada vez que se actualiza el cronómetro. Incrementa los segundos y, si los segundos llegan a 60, incrementa los minutos y si los minutos llegan a 60, incrementa las horas. Luego, formatea las variables **hours**, **minutes** y **seconds** con ceros a la izquierda cuando sea necesario y actualiza el contenido del elemento **timeEl** con el tiempo actualizado.

5. **function startTimer() { ... }** Esta función se ejecuta cuando se hace clic en el botón de inicio del cronómetro. Verifica si el temporizador ya está en funcionamiento (es decir, **timer** no es **null**), y si es así, lo detiene utilizando **clearInterval(timer)**. Luego, inicia un nuevo intervalo del temporizador llamando a **setInterval(stopWatch, 1000)** y almacena el identificador del intervalo en la variable **timer**. El intervalo del temporizador se establece en 1000 milisegundos (1 segundo), y se llama a la función **stopWatch** cada segundo para actualizar el cronómetro.

6. **function stopTimer() { ... }** Esta función se ejecuta cuando se hace clic en el botón de detener del cronómetro. Detiene el temporizador llamando a **clearInterval(timer)**, lo que cancela el intervalo del temporizador y detiene la ejecución de **stopWatch**.

7. **function resetTimer() { ... }** Esta función se ejecuta cuando se hace clic en el botón de reinicio del cronómetro. Detiene el temporizador llamando a **clearInterval(timer)**, restablece las variables **hours**, **minutes** y **seconds** a **0**, actualiza el contenido del elemento **timeEl** a "00:00:00" y muestra **null** en la consola.

En resumen, este código controla el funcionamiento de un cronómetro en una página web. La función **stopWatch** se ejecuta cada segundo para actualizar el tiempo transcurrido, mientras que las funciones **startTimer**, **stopTimer** y **resetTimer** controlan el inicio, detención y reinicio del cronómetro respectivamente.