

APRENDIENDO CHART JS

1. Diseñe un nuevo archivo HTML (**grafico-demo.html**), para este ejemplo, cree una fila y divida en dos columnas iguales (Bootstrap 5.x)

```
grafico-demo.html U x
grafico-demo.html > html > head > link
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Document</title>
8   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet">
9 </head>
10 <body>
11
12   <div class="container">
13     <div class="row">
14
15       <!-- Aquí renderizamos el gráfico -->
16       <div class="col-md-6">
17
18       </div>
19
20       <div class="col-md-6">
21
22       </div>
23     </div>
24   </div>
25
```

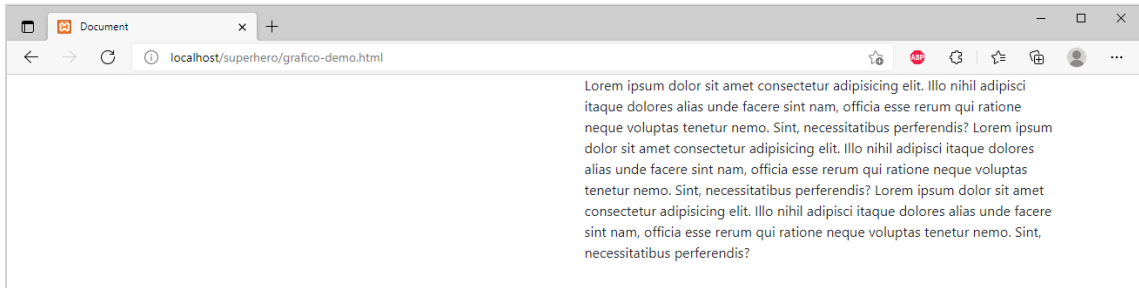
2. En la primera columna agregue una etiqueta **canvas** y asígnele un ID

```
11
12 <div class="container">
13   <div class="row">
14
15     <!-- Aquí renderizamos el gráfico -->
16     <div class="col-md-6">
17       <canvas id="grafico"></canvas>
18     </div>
19
```

3. La tercera columna es solo un espacio que puede ser utilizado para cualquier otro propósito, así que construya un párrafo con muchas líneas de prueba:

```
12 <div class="container">
13   <div class="row">
14
15     <!-- Aquí renderizamos el gráfico -->
16     <div class="col-md-6">
17       <canvas id="grafico"></canvas>
18     </div>
19
20     <div class="col-md-6">
21       <p>
22         Lorem ipsum dolor sit amet consectetur adipisicing
23         Lorem ipsum dolor sit amet consectetur adipisicing
24         Lorem ipsum dolor sit amet consectetur adipisicing
25       </p>
26     </div>
27   </div>
28 </div>
29
```

Hasta el momento debería tener



4. Incluya un bloque JavaScript, haga uso del CDN e ingrese las siguientes instrucciones:

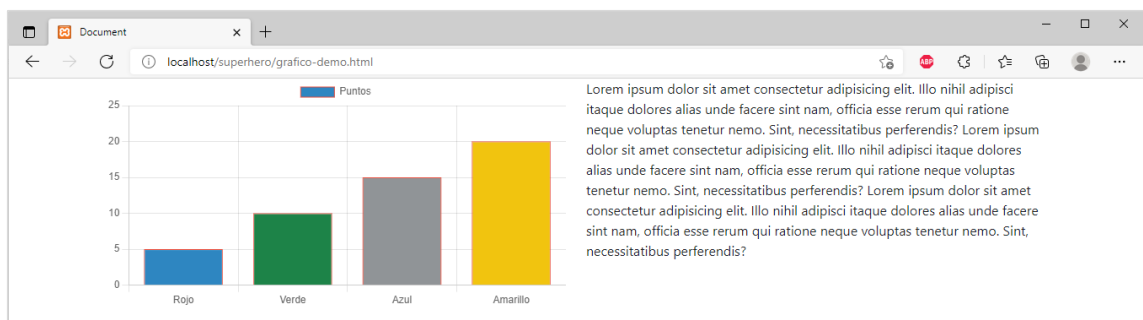
CDN:

<https://cdn.jsdelivr.net/npm/chart.js>

```
31 <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
32
33 <script>
34   document.addEventListener("DOMContentLoaded", () => {
35
36     //Objeto para referencias el canvas
37     const lienzo = document.getElementById("grafico");
38
39     const graficoBarras = new Chart(lienzo, {
40       type: 'bar',
41       data: {
42         labels: ['Rojo', 'Verde', 'Azul', 'Amarillo'],
43         datasets: [
44           {
45             borderColor: '#E74C3C',
46             backgroundColor: ['#2E86C1', '#1D8348', '#909497', '#F1C40F'],
47             label: 'Puntos',
48             data: [5, 10, 15, 20, 25],
49             borderWidth: 1
50           }
51         ]
52       }
53     });
54
```

Los atributos **borderColor** y **backgroundColor** NO son obligatorios. Si los incluye, puede elegir la gama de colores que desee: <https://htmlcolorcodes.com/es/>

5. En el navegador



IMPORTANTE:

- Los atributos definidos con:
 - Atributo: `''` Cadena
 - Atributo: `1` Entero
 - Atributo: `{}` Objetos
 - Atributo: `[]` Colecciones
 - Atributo: `[{}]` Colección de objetos
 - Atributo: `['']` Colección de cadenas
- Analizando código anterior:

The diagram shows a code snippet for creating a bar chart. Annotations explain the following parts:

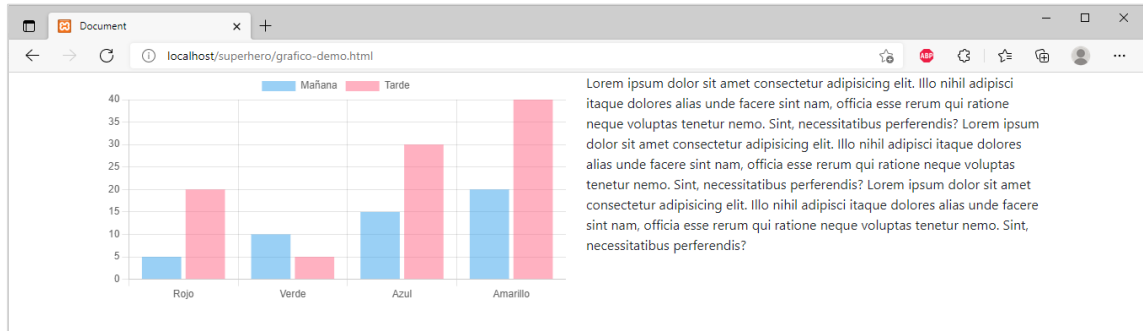
- Tipo de gráfico:** Points to the `type: 'bar'` property.
- Etiquetas (eje X):** Points to the `labels` array: `['Rojo', 'Verde', 'Azul', 'Amarillo']`.
- Datos (eje Y):** Points to the `data` array: `[5, 10, 15, 20, 25]`. The annotation states: "Pueden ser N conjunto de datos, cada uno deberá tener label y data".
- borderColor – backgroundColor son atributos opcionales:** Points to the `borderColor` and `backgroundColor` properties.

```
const graficoBarras = new Chart(lienzo, {
  type: 'bar',
  data: {
    labels: ['Rojo', 'Verde', 'Azul', 'Amarillo'],
    datasets: [
      {
        borderColor: '#E74C3C',
        backgroundColor: ['#2E86C1', '#1D8348', '#909497', '#F1C40F'],
        label: 'Puntos',
        data: [5, 10, 15, 20, 25],
        borderWidth: 1
      }
    ]
  }
});
```

6. Antes de continuar y a fin de mantener el código lo más limpio posible, eliminaré `borderColor`, `backgroundColor` y `borderWidth` (si Ud. Desea lo puede mantener). Para comprender mejor el concepto de datasets, agregue un nuevo set de datos:

```
33 <script>
34 document.addEventListener("DOMContentLoaded", () => {
35
36   //Objeto para referencias el canvas
37   const lienzo = document.getElementById("grafico");
38
39   const graficoBarras = new Chart(lienzo, {
40     type: 'bar',
41     data: {
42       labels: ['Rojo', 'Verde', 'Azul', 'Amarillo'],
43       datasets: [
44         {
45           label: 'Mañana',
46           data: [5, 10, 15, 20, 25]
47         },
48         {
49           label: 'Tarde',
50           data: [20, 5, 30, 40, 10]
51         }
52       ]
53     }
54   });
55 }
```

7. El resultado en el navegador



ACTUALIZANDO DATOS

Vamos a copiar el archivo anterior (grafico-demo.html) y crearemos el archivo grafico-update.html, a partir de este momento, trabajaremos con este último

1. Empezaremos agregando un formulario y 4 cajas de texto, además de un botón actualizar, estos elementos se ubicarán donde anteriormente estaba el párrafo con el texto de prueba:

```
12 <div class="container">
13   <div class="row">
14     <!-- Aquí renderizamos el gráfico -->
15     <div class="col-md-6">
16       <canvas id="grafico"></canvas>
17     </div>
18
19     <!-- Ahora construiremos un formulario -->
20     <div class="col-md-6">
21       <form action="" autocomplete="off" class="mt-2">
22         <div class="form-floating mb-3">
23           <input type="number" class="form-control" placeholder="Valor rojo" id="rojo">
24           <label for="rojo">Valor rojo</label>
25         </div>
26         <div class="form-floating mb-3">
27           <input type="number" class="form-control" placeholder="Valor verde" id="verde">
28           <label for="verde">Valor verde</label>
29         </div>
30         <div class="form-floating mb-3">
31           <input type="text" class="form-control" placeholder="Valor azul" id="azul">
32           <label for="azul">Valor azul</label>
33         </div>
34         <div class="form-floating mb-3">
35           <input type="text" class="form-control" placeholder="Valor amarillo" id="amarillo">
36           <label for="amarillo">Valor amarillo</label>
37         </div>
38       </form>
39       <button type="button" id="actualizar" class="btn btn-info">Actualizar</button>
40     </div>
41   </div> <!-- Fin row -->
42 </div> <!-- Fin container -->
```

2. El siguiente paso será crear un objeto para manejar al botón

```
document.addEventListener("DOMContentLoaded", () => {
  //Objeto para referencias el canvas
  const btActualizar = document.querySelector("#actualizar");
  const lienzo = document.getElementById("grafico");
```

3. Agregue una nueva función que permita leer el valor de cada caja de texto, convirtiendo el valor a entero, estos constituirán un nuevo arreglo que reemplazará al existente.

```
function actualizarData(){
  const nuevoArreglo = [
    parseInt(document.querySelector("#rojo").value),
    parseInt(document.querySelector("#verde").value),
    parseInt(document.querySelector("#azul").value),
    parseInt(document.querySelector("#amarillo").value)
  ];

  //Asignamos el nuevo datos
  graficoBarras.data.datasets[0].data = nuevoArreglo;
  graficoBarras.update();
}

//Evento que dispara al método
btActualizar.addEventListener("click", actualizarData);
```

Fuera del método actualizarData, utilice el evento click del botón para desencadenar esta acción

4. Resultado:

