Visualizar Estadísticas De Forma Elegante Con Chart.js

La creación de **gráficos estadísticos** que se vean de forma elegante puede ser un dolor de cabeza en algunas oportunidades. Muy a menudo, los **diseñadores web ó programadores** utilizan imágenes para mostrar los datos utilizando Photoshop o **Illustrator**, con lo cual puede tardar mucho mas tiempo de lo estimado.

Lo bueno es que, hay muchas librerías gratuitas de JavaScript disponibles en línea, para que los diseñadores web ó programadores pueden utilizar para mostrar datos rápidamente. El diseñador web ó programador puede entonces presentar datos sin la necesidad de crear imágenes con algún editor de imágenes como Photoshop.

La libreria Chart.js es una de las grandes bibliotecas de JavaScript en línea, la cual ayuda a plasmar datos usando el elemento canvas de HTML5 para dibujar gráficos y tablas.

Para ver Chart.js en acción, vamos a usar datos ficticios para construir algunos gráficos estadísticos:

- Gráfico de líneas
- Gráfico de barras
- Gráfico de Radar
- Gráfico Área Polar
- Gráfico de sectores
- Gráfico de torta

Lo Que Se Necesita Para Este Tutorial

- Librería Chart.js
- Algo de tiempo y paciencia ;)

Primeros Pasos

En primer lugar, tenemos que copiar el archivo **chart.min.js** que esta en la carpeta descomprimida en nuestra carpeta js o directorio donde tengamos nuestro proyecto.

Añadiendo Al Proyecto

Lo siguiente que tenemos que hacer es crear un nuevo archivo HTML e incluir la librería Chart.js:

[html]

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Visualize Data Beautifully Using JS Charts</title>
<link href="css/style.css" media="screen" rel="stylesheet">
<script type="text/javascript" src="js/Chart.min.js"></script>
</head>
<body>
</body>
</html>
```

[/html]

Dibujando Un Gráfico De Líneas

Para dibujar un gráfico de líneas, tenemos que crear un elemento canvas de HTML5 en nuestra sección del **body** de nuestro archivo HTML.

[html]<canvas id="canvas" height="450" width="600"></canvas>[/html]

A continuación, tenemos que crear un script para inicializar la clase gráfica (en este ejemplo, he utilizado myLineChart) y luego se recupera el contexto 2D del lienzo donde queremos dibujar el gráfico. Copie el código antes de la etiqueta del cuerpo de cierre.

[js]

var myLineChart = new Chart(document.getElementById("canvas").getContext("2d")).Line(LineChart, {scaleFontSize : 13, scaleFontColor : "#ffa45e"});

[/js]

Nota: También puede agregar algunos estilos que quiera en el gráfico, como el color del texto y el tamaño de la fuente mediante las opciones de gráfico.

En el ejemplo anterior, se ha utilizado **scaleFontSize** y **scaleFontColor** para cambiar el tamaño de fuente y el color del texto de los datos. Puede consultar la documentación de Chart.js para ver las opciones de diseño disponibles para cada gráfico.

[js]

```
var LineChart = {
labels: ["Ruby", "jQuery", "Java", "ASP.Net", "PHP"],
datasets: [{
fillColor: "rgba(151,249,190,0.5)",
```

```
strokeColor: "rgba(255,255,255,1)",
pointColor: "rgba(220,220,220,1)",
pointStrokeColor: "#fff",
data: [10, 20, 30, 40, 50]
}, {
fillColor: "rgba(252,147,65,0.5)",
strokeColor: "rgba(255,255,255,1)",
pointColor: "rgba(173,173,173,1)",
pointStrokeColor: "#fff",
data: [28, 68, 40, 19, 96]
}]
}
```

[/js]

Luego de ejecutar el anterior código, veremos lo siguiente:

Dibujando Un Gráfico De Barras

Para dibujar un gráfico de barras , tenemos que crear un elemento canvas de HTML5 en primer lugar en nuestra sección de cuerpo de nuestro archivo HTML.

[html]<canvas id="canvas" height="450" width="600"></canvas>[/html]

A continuación, vamos a crear un script para representar a la clase de gráfico de barras (en este ejemplo, se ha utilizado **myBarChart**) y luego recuperar el contexto 2D del lienzo. Copie el código antes de la etiqueta de cierre de body.

[js]

```
var myBarChart = new Chart(document.getElementById("canvas").getContext("2d")).Bar(BarChart, {scaleFontSize : 13, scaleFontColor : "#ffa45e"});
```

[/js]

El gráfico de barras tiene casi la misma estructura de datos que el gráfico de líneas. Vamos a usar un objeto (en este ejemplo , se utilizó BarChart) para mantener en las etiquetas y los valores de gráfico de barras. Como puede ver , estamos mostrando los mismos datos que el ejemplo gráfico de líneas.

[js]

```
var BarChart = {
labels: ["Ruby", "jQuery", "Java", "ASP.Net", "PHP"],
datasets: [{
fillColor: "rgba(151,249,190,0.5)",
strokeColor: "rgba(255,255,255,1)",
data: [13, 20, 30, 40, 50]
}, {
fillColor: "rgba(252,147,65,0.5)",
strokeColor: "rgba(255,255,255,1)",
data: [28, 68, 40, 19, 96]
}]
}
```

[/js]

Una vez se ejecute el ejemplo, veremos lo siguiente:

Mostrar Gráfico De Radar

Al igual que los anteriores ejemplos , comenzamos con el canvas:

[html]<canvas id="canvas" height="450" width="610"></canvas>[/html]

A continuación , vamos a crear un script para inicializar la clase gráfico radial (en este ejemplo, se ha utilizado

myRadarChart) y luego se recupera el contexto 2D del lienzo. Ponga el código de abajo antes de la etiqueta de cierre de **body**.

[js]

```
var myRadarChart = new Chart(document.getElementById("canvas").getContext("2d")).Radar(RadarChart, {pointLabelFontSize : 13, pointLabelFontColor : "#ffa45e"});
```

[/js]

En el gráfico de radar, tenemos que mostrar una etiqueta a cada punto de la gráfica. Esto incluirá una matriz de cadenas y luego mostrarlo alrededor del gráfico. Para ello, vamos a crear un objeto nuevo (para este ejemplo se ha usado RadarChart) como contenedor de las etiquetas y los valores de nuestro gráfico radial.

[js]

```
var RadarChart = {
labels: ["Ruby", "jQuery", "Java", "ASP.Net", "PHP"],
datasets: [{
fillColor: "rgba(151,249,190,0.5)",
strokeColor: "rgba(255,255,255,1)",
pointColor: "rgba(220,220,220,1)",
pointStrokeColor: "#fff",
data: [10, 20, 30, 40, 50]
}, {
fillColor: "rgba(252,147,65,0.5)",
strokeColor: "rgba(255,255,255,1)",
pointColor: "rgba(173,173,173,1)",
pointStrokeColor: "#fff",
data: [28, 48, 40, 19, 96]
}]
}
```

[/js]

La siguiente imagen es como quedara el gráfico de este tipo:

Las anteriores gráficas, no son las únicas que se pueden realizar. Solo mostramos lo fácil que es realizar este tipo de gráficas con es librería, en la documentación podrás encontrar mucha mas información sobre las gráficas que no mostramos.

Finalmente

Chart.js es una gran librería de JavaScript para crear gráficos de una manera en que pueda ser creativo. Este tutorial te ha llevado a través de los pasos a seguir para usar esta nueva librería. Sin embargo, hay algunos inconvenientes en el uso de **Chart.js**, como, que no tiene ninguna información sobre herramientas y la interactividad en él. Esperamos que esta información haya sido de gran utilidad si en algún momento deseas realizar un gráfico determinado de tus datos.