LYCÉE ARISTIDE BERGÈS
SIO

Chart.JS

Framework de gestion de graphiques

Rédigé par

David ROUMANET
Professeur BTS SIO

Changement

Date	Révision

Sommaire

A Les bibliothèques graphiques

A.1 Les frameworks ou librairies

Il est fréquent qu'un développeur ait besoin d'une fonction existante dans le code d'un autre développeur.

La question qui se pose, est "comment partager ses codes ?"

La création de librairies ou bibliothèques de fonctions ou encore de framework (cadre de fonction) est alors utile pour permettre à d'autres codeurs d'utiliser nos propres codes. Un exemple provient de **Bootstrap** : il s'agit à l'origine d'un moyen de partager le même design pour les développeurs chez Twitter. Aujourd'hui, Bootstrap est largement répandu et ce framework est disponible gratuitement.

A.2 Les graphiques

La possibilité de dessiner en JavaScript est fournie par l'usage d'un canevas : il s'agit d'une zone rectangulaire dans laquelle il est possible de tracer des formes géométriques. Des traits, des ellipses ou des rectangles et des textes graphiques.

Les fonctions sont primitives et demandent beaucoup de codes pour parvenir à une représentation graphique correct. Notamment, pour représenter des statistiques ou créer des graphiques similaires à ceux disponibles dans les tableurs.

A.3 Intérêt

Si l'usage d'un graphique statistique n'est pas utile dans l'immédiat, il est intéressant pour les sites web avec des bases de données : on peut ainsi fournir des statistiques de fréquentation, un stock de produit, le résultat d'un vote, etc.

B Framework chartJS

ChartJS est un framework créé en 2013 sous licence MIT. Il est simple mais complet pour créer dynamiquement des graphiques dans une page web.

Ce projet est maintenu par une communauté, car il est open-source.

Les caractéristiques principales sont :

- Intégration aux canevas HTML5
- Responsive (s'adapte à la taille de la fenêtre qui le contient)
- Animation lors du dessin
- 8 types de graphiques différents

B.1 Documentation

Chart.JS est très bien documenté : https://www.chartjs.org/docs/latest/

La version actuelle est supérieure à 4 et il existe aussi une version 2.x et 3.x : la documentation permet de choisir la version avec laquelle travailler, mais donne aussi les éléments pour migrer à la nouvelle version.

B.2 Démonstration

Il est possible d'utiliser Chart.JS de deux manières :

- Téléchargement local : https://www.chartjs.org/docs/latest/getting-started/installation.html
- Lien CDN : <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

Le premier code de test sera l'exemple donné dans la documentation.

Testons le code suivant :

demo_chartJS_1.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>Demo 01 - Chart.JS</title>
</head>
<body>
    <div>
        <canvas id="myChart"></canvas>
      </div>

      <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

      <script>
        const ctx = document.getElementById('myChart');

        new Chart(ctx, {
          type: 'bar',
          data: {
            labels: ['Red', 'Blue', 'Yellow', 'Green', 'Purple', 'Orange'],
            datasets: [{
              label: '# of Votes',
              data: [12, 19, 3, 5, 2, 3],
              borderWidth: 1
            }]
          },
          options: {
            scales: {
              y: {
                beginAtZero: true
              }
            }
          }
        });
      </script>
</body>
</html>
```

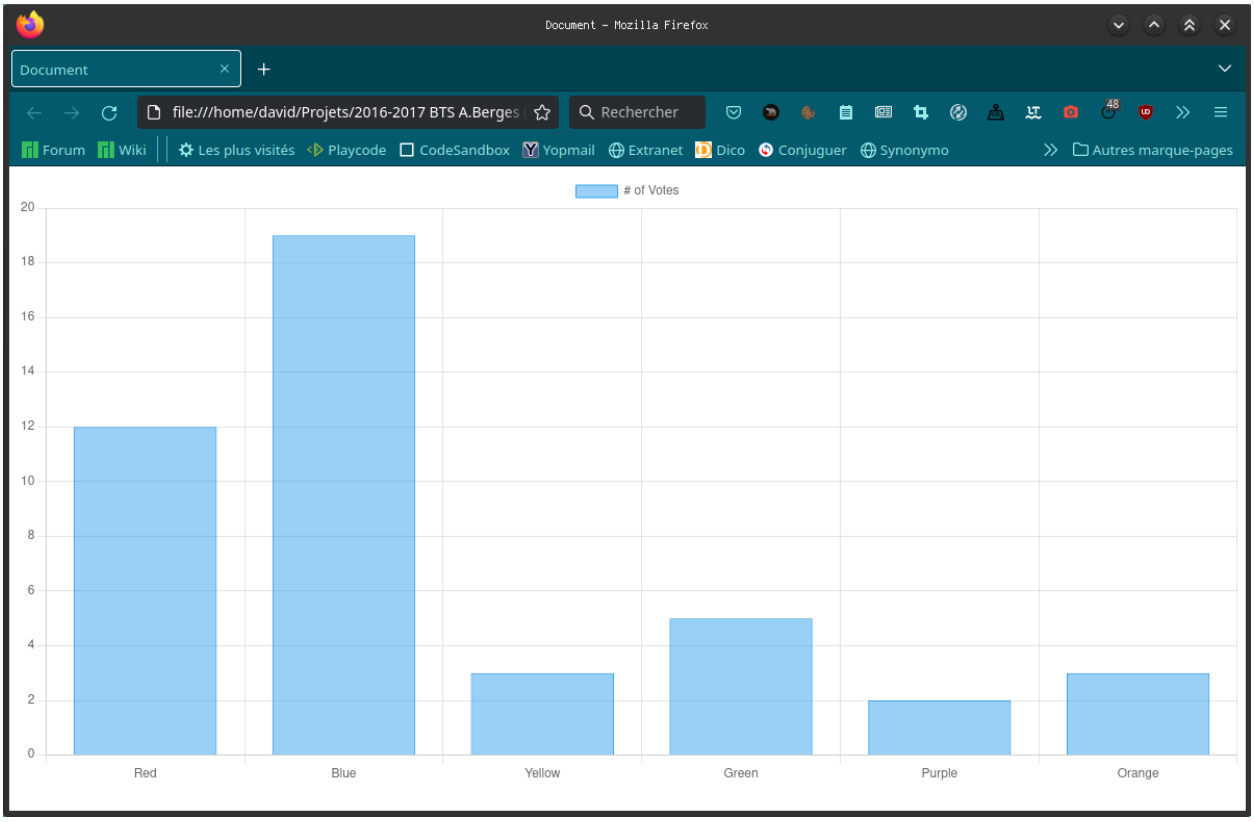
Canvas : zone d'affichage

CDN du framework chart.JS

Labels : tableau d'étiquettes

data: tableau de valeurs

Le résultat est un graphique de type barre. Vous pouvez redimensionner la fenêtre, le graphique s'adapte à la largeur.



La documentation détaille chaque aspect du code fournit.

B.3 Format JSON

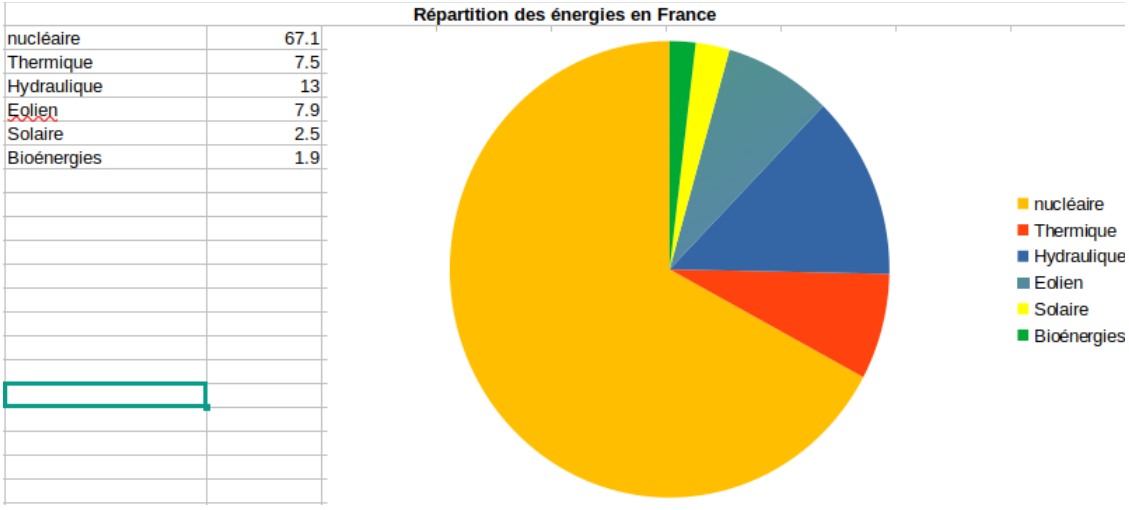
Le code peut paraître complexe, mais dans la réalité, les données utilisent un format appelé JSON (Java Script Object Notator). Un cours spécifique sera donné sur ce format mais voici l'essentiel pour lire comment fonctionne Chart.JS :

- On a toujours un couple mot : objet
  - ex : `data : [1, 2, 3, 4]`
- Les crochets `[` et `]` représentent les cellules d'un tableau
- Les accolades `{` et `}` représentent le contenu d'un objet
  - ex : `{nom :"NORRIS", prenom :"Chuck", surnom:["chuck", "texas ranger", "big chuck"]}`
- On peut faire un retour à la ligne après une virgule

B.4 Création d'un graphique camembert

Enregistrez votre document sous le nom `demo_chartJS_2.html` pour faire le nouveau graphique.

À l'aide de la documentation, essayez de générer l'équivalent du graphique suivant avec Chart.JS



B.5 section complémentaire :

Ici, nous allons voir comment rendre les données accessibles depuis un autre script. Vous pouvez *créer le bouton* dans la page web et utiliser un tableau généré aléatoirement.

Le script pour le bouton doit être placé en premier, avant la section <script> de Chart.JS

```
<script>
    // Dessin dynamique des données sur clic du bouton
    let donnees = []
    let btnRestart = document.getElementById("btnRestart")
    btnRestart.addEventListener("click", () => {
        for (let t=0; t<donnees.length; t++) {
            donnees[t] = Math.floor(Math.random()*1e9)
        }
        barChart.update()
    })
</script>
```

Copier les éléments entre crochet après `data :` et collez les dans le script ci-dessus, après `let donnees =` puis remplacer `data : […]` par

```
data: donnees,
```

Enfin, il faut stocker la création du graphique dans une variable barChart :

```
let barChart = new Chart(ctx, {
```