TP1

Ibrahim ALAME

2 décembre 2024

Nous allons faire un site avec plusieurs pages de présentation d'un café fictif à Paris.

1 Création du projet

1. Commencez par créer dans WebStorm un projet vide, par exemple cafe .
2. Dans ce dossier créez un dossier images qui contiendra les images de notre site.
3. Créez un dossier css au même niveau que le dossier images.

1.1 Création du fichier styles.css

Créez un fichier `styles.css` dans le dossier css.

Il est très courant dans un projet front-end de créer un fichier `styles.css` pour contenir le CSS commun à toutes les pages.

1.2 Création du fichier index.html

Créez un fichier au premier niveau et appelez le `index.html`.

Dans les applications et les sites Web , `index.html` est le point d'entrée par convention. C'est-à-dire que c'est le premier fichier envoyé au navigateur.

Création des deux autres pages

Notre projet comporte deux autres pages : une pour le formulaire de contact, et l'autre pour la localisation.

Nous allons donc créer un fichier `where.html` et un fichier `contact.html`. Modification d'`index.html`

Dans le fichier `index.html` nous allons mettre :

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Café Florette | Accueil</title>
</head>
<body>

</body>
```

```
</html>
```

Ensuite nous importons notre fichier `styles.css` en donnant un chemin relatif à partir du dossier courant (le dossier où se situe `index.html`).

1.2.1 Modification de styles.css

Dans le fichier `styles.css` nous allons mettre :
Appliquer deux règles :

1. La première règle utilise le sélecteur universel * pour enlever les marges et le remplissage par défaut, notamment appliqués sur body .

```
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
```

2. La deuxième règle sélectionne tous les titres de niveau `h1,h2` et `h3` et définit une marge de 20px vers le bas (`margin-bottom: 20px;`).

1.3 Mise en place du header

1.3.1 Modification de index.html

Nous allons commencer par mettre en place la structure de la partie header de notre page :

- Nous ajoutons un lien pour charger la police Open Sans depuis une API de Google .

```
<link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,400,700&display=swap"
rel="stylesheet"
/>
```

- Nous avons sélectionné trois graisses : 300, 400 et 700.
- Dans la partie body nous créons un élément sémantique header dans lequel nous plaçons un élément sémantique nav car il contiendra les liens pour naviguer sur les pages de notre site.

```
    <header>
      <nav>


      </nav>
    </header>
```

- Dans cet élément nav , nous plaçons le titre de la page h1 qui est cliquable car nous y imbriquons un élément a .

```
    <h1><a href="./index.html">CAFE FLORETTE</a></h1>
```

- Ainsi, les utilisateurs seront redirigés sur la page d'accueil lorsqu'ils cliquerons sur le titre.
- Ensuite, nous créons une liste non ordonnée ul , dans laquelle nous plaçons trois list items . Chaque li contient un lien dans un élément a pour pouvoir naviguer vers les autres pages du site.

```
        <ul>
          <li><a href="./index.html">Accueil</a></li>
          <li><a href="./where.html">Nous trouver</a></li>
          <li><a href="./contact.html">Contact</a></li>
        </ul>
```

1.3.2 Raccourci Emmet

Nous allons voir un nouveau raccourci Emmet qui permet de grouper des éléments pour appliquer par exemple * .
Ici nous voulons créer trois fois des li dans lesquels nous avons des a : `ul>(li>a)*3`

1.3.3 Modification de styles.css

Nous allons ajouter quelques règles CSS dans notre fichier styles.css :

1. Nous commençons par utiliser la police Open Sans chargée depuis Google pour tous nos éléments. Nous définissons une famille de polices de repli dans le cas où la police ne pourrait être chargée.

```
html,
body {
  font-family: "Open Sans", sans-serif;
}
```

2. Ensuite, nous remplaçons les règles CSS par défaut pour les éléments a : à savoir le soulignage et le bleu. Nous enlevons le soulignage en faisant `text-decoration: none;` et nous changeons la couleur avec la propriété color (`color: #222;`).

1.3.4 Modification de index.html

Nous allons maintenant attaquer le style de notre header :
Pour cela, nous ajoutons deux classes qui nous permettront de définir des règles CSS .

1. Une première classe `header` qui nous permettra spécifiquement de cibler ce header (pour rappel, nous pouvons avoir plusieurs éléments sémantiques header dans une même page).
2. Une seconde classe `active` , qui nous permettra de définir un style particulier pour le lien de l'accueil. Cela permettra d'indiquer à l'utilisateur qu'il a sélectionné le lien "Accueil" et qu'il est sur la page correspondante.

1.3.5 Modification de styles.css

Nous allons maintenant passer aux règles de style :

```
* {
  margin: 0;
}

html,
body {
  font-family: "Open Sans", sans-serif;
}

h1,h2,h3 {
  margin-bottom: 20px;
}

a {
  color: #222;
  text-decoration: none;
}

/* Header */
.header {
  background: black;
  overflow: auto;
  line-height: 1.7em;
}

.header nav h1 {
  float: left;
  padding: 20px;
  margin-bottom: 0;
}

.header nav h1 a {
  color: white;
}

.header nav ul {
  float: right;
  list-style: none;
```

```
}

.header nav ul li {
  float: left;
}

.header nav ul li a {
  color: white;
  padding: 20px;
  display: block;
}

.header nav ul li a:hover,
.active {
  background: #444;
}
```

Nous allons reprendre ensemble les éléments essentiels.

- **Pourquoi avons-nous dû utiliser overflow-auto sur le parent des éléments positionnés en float ?**

  Le container header ne contient que des éléments qui sont positionnés en float .

  Or, pour rappel, lorsque nous plaçons un élément en float , il est retiré du flux normal des éléments.

  Du coup, l'élément header fait comme si il n'y avait pas d'éléments et il n'a donc pas de hauteur (pour rappel les éléments de bloc prennent la hauteur de leur contenu).

  L'utilisation de overflow : auto permet la création d'un contexte de formatage de blocs et oblige l'élément à prendre en compte les éléments positionnés en float .

  Ne vous attardez pas trop sur ce point car ce n'est pas l'approche la plus moderne. Nous verrons comment il est possible de réaliser la même chose avec le CSS le plus moderne lorsque nous verrons les boites flexibles et les grilles dans les chapitres suivants.

  Sachez que c'était la meilleure manière de faire sans ces deux modules CSS plus modernes. Il est encore possible que vous la rencontriez sur des sites développés il y a quelques temps.

- **Pourquoi devons nous définir une line-height ?**

  Par défaut, les navigateurs prennent 1.2 (ce qui signifie que la valeur de font-size de l'élément est multiplié par 1.2).

  Donc comme un h1 dans une nav a une taille de 1.5em par défaut sa line-height sera

de 1.2*1.5*16px donc 28.8 px . Car pour rappel, dans un navigateur, 1em vaut 16px par défaut.

Et comme les éléments ont une taille de 1 em soit 16px , les liens auront une line-height de 1.2*1*16 px soit 19.2 px .

Pour donner à nos éléments la même line-height , nous la fixons donc à 1.7em ce qui permet que les clics sur les liens puissent se faire sur toute la hauteur du header.

Nous ne détaillerons pas plus le comportement des éléments en float car ce n'est pas utile pour vous. Nous verrons très en détails les grilles et les boites flexibles à la place.

1.4 Mise en place de la partie centrale

1.4.1 Modification de index.html

Nous mettons un élément sémantique section pour notre partie centrale sur lequel nous plaçons une classe vp .

```
<section class="vp">
        <div class="vp-content">

        </div>
</section>
```

1. A l'intérieur nous plaçons le titre principal dans un h1 .

   ```
   <h1>Toute l'expertise française dans un café</h1>
   ```

2. Ensuite nous créons une div pour créer un séparateur.

   ```
   <div class="separator"></div>
   ```

3. Après, nous mettons un paragraphe de texte qui contiendrait normalement la description du lieu.

   ```
   <p>
      Lorem ipsum dolor sit amet consectetur, adipisicing elit. Aspernatur
      nemo cupiditate earum eum porro unde sint ducimus vero voluptas,
      perferendis quasi quia accusamus omnis rerum. Neque sapiente quidem
      distinctio dolorem?
   </p>
   ```

4. Enfin, nous plaçons un lien que nous allons transformer en bouton avec plusieurs classes.

   ```
   <a href="./where.html" class="btn btn-primary my-10">DECOUVRIR</a>
   ```

1.4.2 Modification de styles.css

Nous allons maintenant créer les règles CSS :

```
/* General */
p {
  margin-bottom: 10px;
}

.my-10 {
  margin-top: 10px;
  margin-bottom: 10px;
}

.separator {
  margin: 20px auto;
  height: 10px;
  width: 100px;
  background: #111;
}

.btn {
  display: inline-block;
  font-size: 14px;
  padding: 15px 25px;
  border-radius: 30px;
  font-weight: 700;
}

.btn-primary {
  background: white;
  color: #222;
}

.btn-primary:hover {
  background: #222;
  color: white;
}

/* Header */

.header {
```

```
  background: black;
  overflow: auto;
  line-height: 1.7em;
}

.header nav h1 {
  float: left;
  padding: 20px;
  margin-bottom: 0;
}

.header nav h1 a {
  color: white;
}

.header nav ul {
  float: right;
  list-style: none;
}

.header nav ul li {
  float: left;
}

.header nav ul li a {
  color: white;
  padding: 20px;
  display: block;
}

.header nav ul li a:hover,
.active {
  background: #444;
}

/* value proposition */
.vp {
  height: 700px;
  background: url("../images/cafe1.jpg") center center;
  background-size: cover;
```

```
  text-align: center;
}

.vp-content {
  max-width: 600px;
  padding-top: 150px;
  color: white;
  margin: auto;
}

.vp-content h1 {
  font-size: 50px;
}

.vp-content p {
  font-size: 18px;
  letter-spacing: 0.5px;
  line-height: 1.9em;
}
```

1.4.3 Description des classes génériques

Nous allons commencer par décrire les classes génériques que nous mettons en place.

1. La classe my-10 signifie margin y , donc verticales (en haut et en bas), de 10 px . Donner un tel nom permet de savoir facilement ce que fait cette classe générique.
2. La classe separator permet de créer un séparateur qui est seulement un rectangle de 100px par 10px avec un arrière-plan noir et des marges verticales (en haut et en bas) de 20px . Nous le centrons horizontalement en passant auto en deuxième valeur de la propriété margin .
3. La classe btn permet de créer un bouton en donnant au texte contenu du padding en haut en en bas de 15px et à droite et à gauche de 25px , elle permet également d'arrondir les angles avec border-radius . Elle dispose l'élément en inline-block pour pouvoir disposer plusieurs boutons sur la même ligne tout en respectant le padding . Enfin, elle donne au texte une taille de 14px et une graisse de 700 .
4. La classe btn-primary s'utilisera toujours avec la classe btn et permet de donner une couleur blanche à l'arrière-plan et une couleur noire au texte.
5. La classe btn-primary :hover permet d'utiliser une pseudo-classe pour inverser la couleur de l'arrière-plan et la couleur du texte lorsque l'utilisateur passe son curseur sur le bouton.

1.4.4 Description des classes pour le contenu central

Nous allons ensuite décrire les classes mises en place pour le contenu central.

1. La classe vp permet de créer un bloc de 700px de hauteur pour positionner une image en arrière-plan en utilisant une url . Nous centrons l'image verticalement et horizontalement avec center center . Nous utilisons cover pour recouvrir l'espace disponible.
2. La classe vp-content permet de mettre le texte en blanc, de le centrer horizontalement avec margin : auto . Elle permet également de mettre 150px de padding au-dessus. Enfin, elle définit une largeur maximale à 600px .

1.5 Suite et fin de la page d'accueil

1.5.1 Modification de index.html

Nous allons terminer en ajoutant une photo, une section reservation et une partie footer .

```
<!-- ... -->
<div class="person"></div>
<section class="reservation">
  <div class="container">
    <h2>Réservez dès maintenant</h2>
    <p>
      Lorem ipsum dolor sit amet consectetur adipisicing elit. Aut similique
      soluta voluptates, placeat iure neque dolores vel? Facere perspiciatis
      vel quasi, deleniti inventore obcaecati animi harum totam. Hic, nisi
      cumque.
    </p>
    <a href="./contact.html" class="btn btn-dark my-20">RESERVEZ</a>
  </div>
</section>
<footer>
    <p>Copyright &#9400; - florette - 2019</p>
</footer>
```

Nous ajoutons une div où nous placerons une photo en utilisant du CSS .
Puis une nouvelle section permettant d'aller sur la page de contact.
Et enfin, un bas de page standard avec un signe Copyright.
Pour faire un signe Copyright vous pouvez utiliser les entités.
Une entité HTML est une chaîne de caractères qui commence par une esperluette & et se termine avec un point virgule ; .
Les entités sont utilisées pour afficher des caractères réservés. De nombreux caractères ont des entités mnémoniques. Par exemple, l'entité pour le copyright est © que nous aurions également pu utiliser.

1.5.2 Modification de styles.css

Il nous reste à définir les sélecteurs et les règles correspondantes côté CSS :

```
/* ... */
.my-20 {
```

```
  margin-top: 20px;
  margin-bottom: 20px;
}
/* ... */
.btn-dark {
  background: #222;
  color: white;
}

.btn-dark:hover {
  background: white;
  color: #222;
  border: 1px solid #222;
}

/* ... */

/* reservation */

.reservation {
  padding: 100px 0px;
  text-align: center;
}

/* footer */

footer {
  background: #222;
  color: white;
  padding: 30px 0;
  text-align: center;
}

footer p {
  margin-bottom: 0px;
}
```

1. my-20 est une classe générique que nous créons pour appliquer des marges verticales (d'où le y) de 20px .
2. btn-dark est une classe permettant de créer un bouton sombre plutôt que blanc. btn-dark :hover est un sélecteur utilisant une pseudo-classe afin de cibler les boutons sombres

lorsque l'utilisateur passe sa souris dessus. Dans ce cas, nous inversons les couleurs et nous ajoutons une bordure noire pour bien distinguer le bouton.

3. reservation permet simplement d'appliquer du remplissage (padding) en haut et en bas de l'élément et d'aligner le texte au centre du bloc.
4. footer permet de créer un arrière-plan sombre et de mettre la couleur du texte à blanc. Il applique également du padding en haut et en bas de 30px et aligne le texte au centre.
5. footer p permet d'enlever la marge de 10px que nous appliquons normalement à nos paragraphes. Il gagne sur le sélecteur p en spécificité ce qui permet donc de supprimer la marge pour le paragraphe du footer .

1.6 Mise en place de la page de localisation

1.6.1 Modification de where.html

Nous allons mettre en place la page de localisation. Une partie sera similaire à la page d'accueil :

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <link rel="stylesheet" href="css/styles.css" />
    <script
      src="https://kit.fontawesome.com/97bb762bba.js"
      crossorigin="anonymous"
    ></script>
    <link
      href="https://fonts.googleapis.com/css?family=Open+Sans:300,400,700&display=swap"
      rel="stylesheet"
    />
    <title>Café florette | Localisation</title>
  </head>
  <body>
    <header class="header">
      <nav>
        <h1>
          <a href="./index.html">CAFE FLORETTE</a>
        </h1>
        <ul>
          <li><a href="./index.html">Accueil</a></li>
          <li><a class="active" href="./where.html">Nous trouver</a></li>
          <li><a href="./contact.html">Contact</a></li>
```

```
        </ul>
      </nav>
    </header>
    <section class="where">
      <div class="container">
        <img
          class="where-img"
          src="images/cafe2.png"
          alt="carte"
        />
        <div class="where-text">
          <h1 class="underline">Où nous trouver</h1>
          <h3>
            Métro : ligne <span class="chip chip-purple">4</span> - Odéon, ligne
            <span class="chip chip-brown">11</span> - Mabillon
          </h3>
          <p>Adresse : 131 Boulevard Saint-Germain, 75006 Paris</p>
        </div>
      </div>
    </section>
    <footer>
      <p>Copyright &#9400; - florette - 2019</p>
    </footer>
  </body>
</html>
```

Sur cette page, nous aurons la même partie nav mais ce sera le lien "Nous trouver" qui aura la classe active .

Nous aurons seulement une section avec l'image de l'emplacement du café, l'adresse et les accès en transport en commun.

Nous avons mis un titre de niveau h1 qui décrit bien le contenu de la page, nous avons également mis un titre h3 avec les lignes de métro.

1.6.2 Modification de styles.css

Il ne reste qu'à faire les sélecteurs et les règles CSS :

```
/* ... */
.underline {
  padding-bottom: 10px;
  border-bottom: 1px solid #ddd;
}
```

```
.chip {
  display: inline-block;
  text-align: center;
  width: 30px;
  border-radius: 100%;
  background: #333;
  color: white;
}

.chip-purple {
  background: #be418c;
}

.chip-brown {
  background: #8c5e24;
}
/* ... */

/* where */

.where {
  padding: 180px 0;
  overflow: auto;
  min-height: 100vh;
}

.where-img {
  float: left;
  width: 45%;
  border: 1px solid #ddd;
  border-radius: 5px;
}

.where-text {
  float: right;
  width: 50%;
}
```

underline permet de mettre une ligne donnant un effet de soulignement. Il s'agit en fait seulement d'une bordure sur un seul côté de 1px grise. chip permet de créer une petite bulle. Nous utilisons une disposition inline-block pour pouvoir lui donner une taille et en mettre plusieurs sur la même ligne.

Les autres sélecteurs permettent simplement de donner la couleur adaptée pour l'arrière plan des bulles.

Pour la section nous utilisons également la propriété float pour l'image et le texte. Nous utilisons overflow : auto pour le container .

1.7 Mise en place de la page de contact

1.7.1 Modification de contact.html

Il ne nous reste plus qu'à mettre en place la page de contact :

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <link rel="stylesheet" href="./css/style.css" />
    <script
      src="https://kit.fontawesome.com/97bb762bba.js"
      crossorigin="anonymous"
    ></script>
    <link
      href="https://fonts.googleapis.com/css?family=Open+Sans:300,400,700&display=swap"
      rel="stylesheet"
    />
    <title>Café florette | Accueil</title>
  </head>
  <body>
    <header class="header">
      <nav>
        <h1>
          <a href="./index.html">CAFE FLORETTE</a>
        </h1>
        <ul>
          <li><a href="./index.html">Accueil</a></li>
          <li><a href="./where.html">Nous trouver</a></li>
          <li><a class="active" href="./contact.html">Contact</a></li>
        </ul>
      </nav>
    </header>
    <section class="contact">
      <div class="contact-form-container">
```

```
        <form action="#">
          <h1 class="underline">Une question ? Contactez nous</h1>
          <label class="my-20" for="name">Nom</label>
          <input type="text" name="name" id="name" />
          <label class="my-20" for="email">Email</label>
          <input type="email" name="email" id="email" />
          <label class="my-20" for="question">Questions et remarques</label>
          <textarea name="question" id="question"></textarea>
          <button class="btn btn-dark my-20">SOUMETTRE</button>
        </form>
      </div>
    </section>
    <footer>
      <p>Copyright &#9400; - florette - 2019</p>
    </footer>
  </body>
</html>
```

Nous gardons également les mêmes liens dans nav mais plaçons cette fois la classe ac tive sur le lien "Contact".

Nous avons une section qui va contenir notre formulaire de contact.

Ce formulaire va comporter un titre de niveau h1 , deux input et un textarea .

Nous plaçons un label pour chaque champ.

Nous avons enfin un bouton sur lequel nous plaçons les classes que nous avions déjà définies pour obtenir un bouton sombre.

A noter que nous ne plaçons rien dans action car nous n'avons pas de serveur où envoyer le formulaire.

1.7.2 Modification de styles.css

Nous passons maintenant au CSS :

```
/* ... */

/* contact */

.contact {
  padding: 180px 0px 100px 0px;
  min-height: 100vh;
}

.contact-form-container {
  max-width: 600px;
```

```
  margin: auto;
}

.contact-form-container form label {
  display: block;
  font-weight: 700;
}

.contact-form-container form input,
textarea {
  display: block;
  width: 100%;
  padding: 15px;
  border: 1px solid #ddd;
  border-radius: 3px;
}

.contact-form-container form textarea {
  min-height: 80px;
}
```

Avec les sélecteurs contact-form-container form input et textarea , nous ciblons tous les champs de notre formulaire. Nous leur donnons une largeur de 100% de leur élément parent qui a une taille maximale de 600px . Nous leur donnons une bordure de 1px en arrondissant les bordes avec border-radius .