

Chapitre 5 : Des exemples de composants flexibles

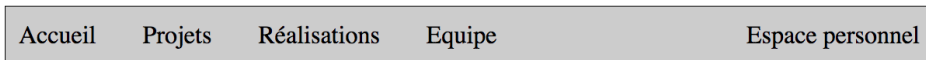
A. Créer des barres de navigation	78
B. Exploiter les alignements des blocs	94
C. Créer des formulaires	104

A. Créer des barres de navigation

1. Création d'une barre de menu simple

Le fichier d'exemple à télécharger se nomme : **01-Exemples Flexbox/C05-A-1.html**.

Pour ce premier exemple de réalisation avec Flexbox, nous allons concevoir une barre de menu simple, mais efficace. Voici le résultat final que nous allons obtenir :



Voici la structure HTML initiale :

```
<nav id="nav-bar">
  <a href="#">Accueil</a>
  <a href="#">Projets</a>
  <a href="#">Réalisations</a>
  <a href="#">Equipe</a>
  <a href="#">Espace personnel</a>
</nav>
```

Nous utilisons l'élément HTML5 `<nav>` pour accueillir la barre de navigation. Cet élément possède l'identifiant `nav-bar`.

Voici la règle CSS pour cet élément :

```
#nav-bar {
  display: flex;
  flex-flow: row nowrap;
  justify-content: flex-start;
  width: 600px;
  background-color: #ccc;
}
```

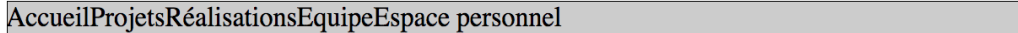
Analysons les propriétés CSS utilisées :

- `display: flex` : nous appliquons un affichage flexible.
- `flex-flow: row nowrap` : nous utilisons les valeurs par défaut pour le flux du conteneur. Nous aurions parfaitement pu ne pas les indiquer. L'axe principal est horizontal (`flex-direction: row`) et nous interdisons le passage à la ligne (`flex-wrap: nowrap`).
- `justify-content: flex-start` : à nouveau, nous indiquons la valeur par défaut (donc nous aurions pu aussi l'omettre) pour l'alignement sur l'axe principal. Les enfants sont alignés au début de l'axe horizontal, donc à partir du bord gauche de cet élément parent.
- `width: 600px` : la barre de menu à une largeur fixe de 600 pixels dans cet exemple. Vous pouvez bien sûr indiquer la valeur que vous souhaitez.

- `background-color: #ccc` : indique que l'arrière-plan de la barre de navigateur est gris clair.

Dans cet élément `<nav>`, nous imbriquons des éléments `<a>` pour créer les liens de la barre de menu. Il n'est pas nécessaire d'ajouter une liste `<ul>`, comme vous pourriez le faire usuellement, car tous les éléments enfants d'un conteneur flexible, seront automatiquement flexibles à leur tour.

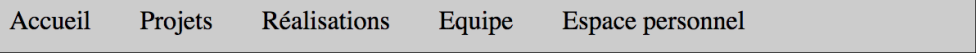
Voici l'affichage initial obtenu :



Pour espacer les liens, nous appliquons une marge à droite (`margin-right: 10px`) de chaque élément `<a>` et un remplissage interne (`padding: 10px`). Les liens ne sont pas soulignés (`text-decoration: none`) et leur couleur est noire (`color: #000`).

```
#nav-bar a {  
    margin-right: 10px;  
    padding: 10px;  
    text-decoration: none;  
    color: #000;  
}
```

Voici l'affichage obtenu :



Maintenant, nous souhaitons avoir un effet de survol au-dessus de chaque lien dans la barre de navigation. Nous utilisons cette nouvelle règle CSS :

```
#nav-bar a:hover {  
    background-color: #000;  
    color: #FFF;  
}
```

- `background-color: #000` applique une couleur d'arrière-plan noire aux liens.
- `color: #FFF` indique que la couleur du texte des liens est blanche.

Voici l'affichage obtenu au survol d'un lien :



Dernière amélioration, nous souhaitons que le dernier lien, **Espace personnel**, soit placé tout à droite de la barre de navigation. Pour cela nous créons cette nouvelle règle :

```
#nav-bar a.perso {  
    margin-left: auto;  
    margin-right: 0;  
}
```

- La propriété `margin-left: auto` permet de positionner l'élément avec le maximum d'espace sur sa gauche. Ainsi ce lien est calé sur la droite de son élément parent.
- `margin-right: 0` permet de supprimer la marge droite initiale qui était appliquée à tous les éléments `<a>` enfants pour les espacer entre eux.

Voici le code complet de cet exemple :

```
<!doctype html>  
<html lang="fr">  
  <head>  
    <meta charset="UTF-8">  
    <title>Ma page web</title>  
    <style>  
      #nav-bar {  
        display: flex;  
        flex-flow: row nowrap;  
        justify-content: flex-start;  
        width: 600px;  
        background-color: #ccc;  
      }  
      #nav-bar a {  
        margin-right: 10px;  
        padding: 10px;  
        text-decoration: none;  
        color: #000;  
      }  
      #nav-bar a:hover {  
        background-color: #000;  
        color: #FFF;  
      }  
      #nav-bar a.perso {  
        margin-left: auto;  
        margin-right: 0;  
      }  
    </style>  
</head>
```

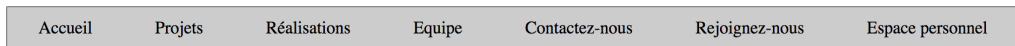
```
<body>
  <nav id="nav-bar">
    <a href="#">Accueil</a>
    <a href="#">Projets</a>
    <a href="#">Réalisations</a>
    <a href="#">Equipe</a>
    <a href="#" class="perso">Espace personnel</a>
  </nav>
</body>
</html>
```

2. Création d'une barre de menu responsive

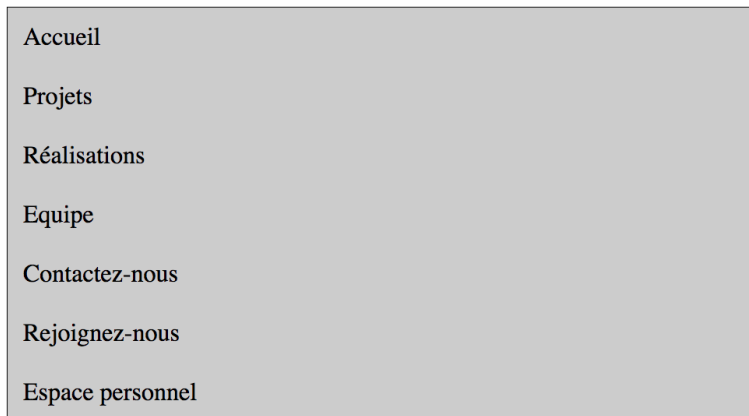
Le fichier d'exemple à télécharger se nomme : **01-Exemples Flexbox/C05-A-2.html**.

Pour ce deuxième exemple de menu, nous allons concevoir une barre de menu responsive, avec des liens répartis horizontalement et centrés.

Voici le résultat final obtenu sur un écran large :



Voici le résultat final obtenu sur un petit écran :



Voilà la structure HTML initiale qui est très simple :

```
<nav id="nav-bar">
  <a href="#">Accueil</a>
  <a href="#">Projets</a>
  <a href="#">Réalisations</a>
  <a href="#">Equipe</a>
```

```
<a href="#">Contactez-nous</a>
<a href="#">Rejoignez-nous</a>
<a href="#">Espace personnel</a>
</nav>
```

Nous avons une structure similaire à celle du premier exemple, nous n'y revenons pas.

Nous allons adopter une démarche *responsive first*. Cela veut dire que nous allons d'abord créer toutes les règles CSS pour avoir une mise en forme optimisée pour les petits écrans, pour les smartphones. Ces règles CSS contiendront aussi les propriétés communes pour les smartphones et pour les écrans plus larges.

Commençons par la barre de navigation en elle-même, par la boîte `<nav id="nav-bar">`. Voici la règle CSS dédiée :

```
/* Propriétés communes pour tous les écrans */
/* et pour les smartphones */
#nav-bar {
    display: flex;
    flex-direction: column;
    background-color: #ccc;
}
```

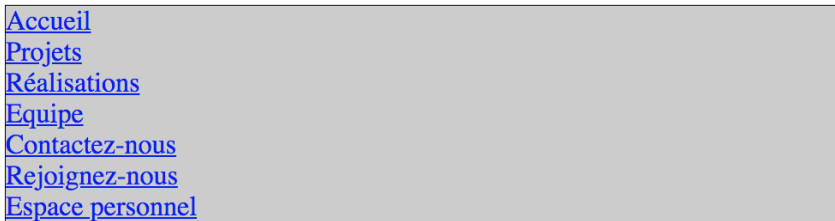
Étudions les propriétés utilisées :

- `display: flex` : nous souhaitons bien sûr avoir un affichage flexible.
- `flex-direction: column` : permet d'indiquer que nous voulons un affichage vertical pour l'axe principal. C'est logique puisque nous sommes dans un affichage sur smartphone, donc les liens doivent bien s'afficher les uns au-dessous des autres.

Vous remarquez que nous ne précisons pas la propriété `flex-wrap: nowrap`, puisque c'est la valeur par défaut.

- `background-color: #ccc` indique que la couleur d'arrière-plan est gris clair.

Voici l'affichage obtenu :



```
Accueil
Projets
Réalisations
Equipe
Contactez-nous
Rejoignez-nous
Espace personnel
```