Parfait — voici une **version prête à enseigner**, simple à suivre en classe.

Cours d'1h : apprendre Flexbox

Niveau

Débutant

Pré-requis

Les étudiant·es connaissent déjà les balises : `header`, `nav`, `article`, `footer`

Objectif du cours

Comprendre comment **positionner des éléments avec Flexbox** pour construire une petite mise en page.

---

1. Déroulé minute par minute

0–5 min — Repartir du HTML connu

Tu peux commencer comme ça :

Aujourd'hui, on va voir comment placer des éléments dans une page.
Vous connaissez déjà des balises comme `header`, `nav`, `article`, `footer`.
Le problème, c'est que par défaut, ces blocs se mettent les uns en dessous des autres.
Flexbox permet de contrôler leur organisation.

Code à montrer :

```
<header>Header</header>
<nav>Nav</nav>
<article>Article</article>
<footer>Footer</footer>
```

Question à poser :

- "Comment ces éléments se placent-ils naturellement ?"

Réponse attendue :

- verticalement
- en pile

5–10 min — Montrer le problème

Dire :

Si je veux mettre `nav` à gauche et `article` à droite, le HTML seul ne suffit pas.
Il faut utiliser du CSS.
Et pour ça, Flexbox est très pratique.

10–15 min — Principe fondamental

Phrase importante à répéter :

Flexbox s'applique au parent.

Exemple :

```
<div class="container">
  <div>Bloc 1</div>
  <div>Bloc 2</div>
  <div>Bloc 3</div>
</div>
.container {
  display: flex;
}
```

Dire :

Dès que je mets `display: flex` sur le conteneur, ses enfants se placent sur une ligne.

15–20 min — Vocabulaire minimum

Introduire simplement :

- **conteneur flex** = le parent
- **items flex** = les enfants

Puis :

Flexbox fonctionne avec deux axes :

- l'axe principal
- l'axe secondaire

Sans trop complexifier :

- si `flex-direction: row`, l'axe principal est horizontal

- si `flex-direction: column`, l'axe principal est vertical

20–30 min — Les 4 propriétés essentielles

a. `flex-direction`

```
.container {
  display: flex;
  flex-direction: row;
}
```

Dire :

`row` met les éléments sur une ligne.
`column` les remet en colonne.

Exemple :

```
.container {
  display: flex;
  flex-direction: column;
}
```

---

b. `justify-content`

```
.container {
  display: flex;
  justify-content: center;
}
```

Dire :

`justify-content` aligne les éléments sur l'axe principal.

Valeurs à montrer :

- `flex-start`
- `center`
- `space-between`

---

c. `align-items`

```
.container {
  display: flex;
  align-items: center;
}
```

Dire :

`align-items` aligne les éléments sur l'autre axe.

---

d. `gap`

```
.container {
  display: flex;
  gap: 20px;
}
```

Dire :

`gap` ajoute de l'espace entre les éléments.

30–40 min — Application à une vraie structure HTML

Montrer cette structure :

```
<body>
  <header>Mon site</header>

  <main class="contenu">
    <nav>Menu</nav>
    <article>Texte principal</article>
  </main>

  <footer>Bas de page</footer>
</body>
```

Puis le CSS :

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
}

header,
nav,
article,
footer {
  border: 1px solid black;
  padding: 20px;
}

.contenu {
  display: flex;
  gap: 20px;
}

nav {
```

```
  width: 200px;
}

article {
  flex: 1;
}
```

Explication orale :

Ici, `main` devient le conteneur flex.
Ses deux enfants, `nav` et `article`, se mettent côte à côte.
`nav` garde une largeur fixe.
`article` prend le reste de la place.

40–45 min — Introduire `flex: 1`

Dire :

`flex: 1` veut dire : prends l'espace restant.

Exemple :

```
article {
  flex: 1;
}
```

C'est souvent la propriété la plus utile dans une mise en page simple.

45–55 min — Exercice en autonomie

Consigne à donner :

Créez une page avec :

- un `header` en haut
- un `nav` à gauche
- un `article` à droite
- un `footer` en bas

Contraintes :

- `nav` et `article` doivent être côte à côte
- il faut un espace entre eux
- `article` doit prendre plus de place que `nav`

55–60 min — Correction collective

Faire corriger par la classe, puis montrer une solution.

---

2. Script oral simple

Tu peux presque lire ceci tel quel :

Flexbox est un système de mise en page en CSS.
Il sert à aligner et répartir des éléments dans un conteneur.
Le point essentiel, c'est qu'on active Flexbox sur le parent avec `display: flex`.
Ensuite, les enfants deviennent des éléments flexibles.
Avec `flex-direction`, on choisit la direction.
Avec `justify-content`, on organise les éléments sur l'axe principal.
Avec `align-items`, on les aligne sur l'autre axe.
Avec `gap`, on ajoute un espace entre eux.
Et avec `flex: 1`, un élément peut occuper l'espace restant.

---

3. Fichier complet à projeter

HTML

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Cours Flexbox</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <header class="top">
    <h1>Mon site</h1>
  </header>

  <main class="contenu">
    <nav>
      <h2>Navigation</h2>
      <ul>
        <li>Accueil</li>
        <li>Portfolio</li>
        <li>Contact</li>
      </ul>
    </nav>

    <article>
      <h2>Article</h2>
      <p>
```

```
        Ceci est le contenu principal. Grâce à Flexbox, le menu et
l'article
        peuvent être placés côte à côte facilement.
      </p>
    </article>
  </main>

  <footer>
    <p>Footer</p>
  </footer>
</body>
</html>
```

CSS

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
}

.top {
  background: #d9d9d9;
  padding: 20px;
  text-align: center;
}

.contenu {
  display: flex;
  gap: 20px;
  padding: 20px;
}

nav {
  width: 200px;
  background: #efefef;
  padding: 20px;
}

article {
  flex: 1;
  background: #f7f7f7;
  padding: 20px;
}

footer {
  background: #d9d9d9;
  padding: 20px;
  text-align: center;
}
```

---

4. Ce qu'il faut écrire au tableau

Tu peux résumer ainsi :

```
parent {
  display: flex;
```

```
}
```

Puis :

```
flex-direction
justify-content
align-items
gap
flex
```

Et faire noter :

- `display: flex` → active Flexbox
- `flex-direction` → direction
- `justify-content` → axe principal
- `align-items` → axe secondaire
- `gap` → espace entre éléments
- `flex: 1` → prend l'espace restant

---

5. Erreurs fréquentes à signaler

- mettre `display: flex` sur le mauvais élément
- confondre `justify-content` et `align-items`
- oublier que Flexbox agit sur les enfants directs
- oublier que `article` peut grandir avec `flex: 1`

---

6. Petite phrase de conclusion

Flexbox permet d'organiser facilement des éléments dans une page.
C'est très utile pour faire des menus, des colonnes, centrer du contenu et répartir l'espace.

Je peux maintenant te faire la **version en mode fiche de cours imprimable pour les étudiant·es**, ou la **version diaporama en 8 slides**.