

Mise en page Responsive

Développement Web - HTML / CSS



Sommaire

1. Introduction.
2. Media Queries.
3. Flexbox.
4. Grid.

1. Introduction.

1. Introduction.

Qu'est-ce que c'est que le développement responsive :

- 56.8% du trafic internet est fait via des appareils mobiles
- Chaque type d'appareil à un format différent (Ordinateur, tablette, téléphone)
- Chaque appareil peut avoir une taille d'écran différente
- Un site web codé uniquement pour un ordinateur sera lu correctement par moins de 50% des utilisateurs du web

1. Introduction.

Qu'est-ce que c'est que le développement responsive :

- Plusieurs systèmes ont été mis en place pour ne pas avoir à développer des sites différents pour chaque type d'appareil : le design adaptatif ou « responsive design », nécessitant :
 - Des grilles « fluides »
 - Des unités relatives, qui peuvent s'adapter à la taille de l'écran
 - Les Media Queries, une règle CSS qui permet de changer style et le type de mise en page selon la taille de l'écran
- Approche « mobile-first » : On développe d'abord pour mobile, puis pour ordinateurs

1. Introduction.

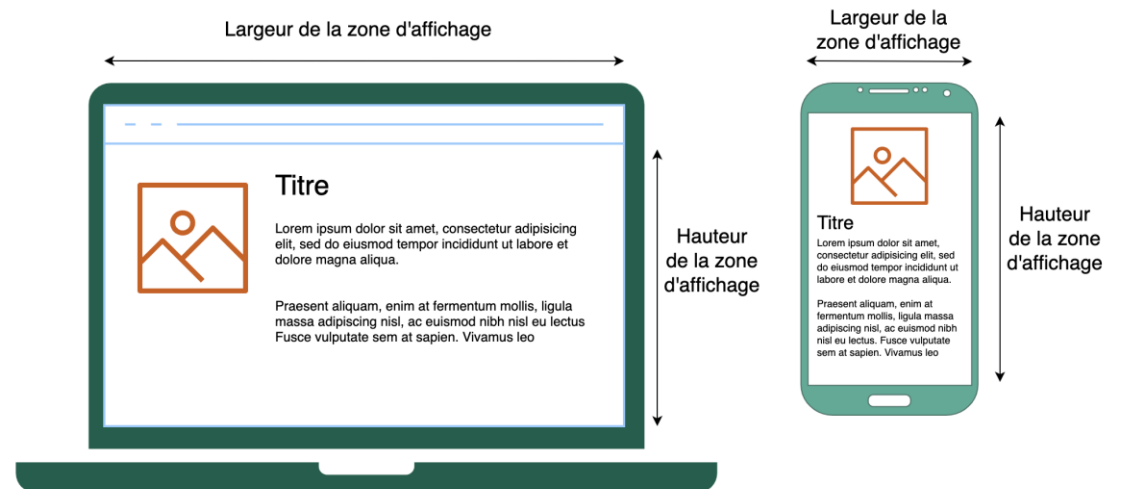


2. Media Queries.

2. Media Queries.

Qu'est-ce que c'est :

- « Requêtes media CSS »
- Règle permettant de créer des styles spécifiques à des taille d'écrans (Réduire la taille d'éléments et agrandir des boutons sur un appareil mobile par exemple)



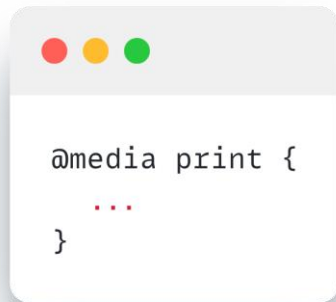
2. Media Queries.

Types de média :

- Un type de média définit une catégorie générale d'appareil :
 - all : Correspond à tous les appareils
 - print : Correspond à l'aperçu avant impression
 - screen : Correspond aux appareils dotés d'un écran
 - speech : Correspond aux outils de synthèse vocale

2. Media Queries.

Cibler des types de média — Exemples :



Les styles contenus dans ce bloc ne seront appliqués que lors de la prévisualisation d'impression et lors de l'impression de la page



Les styles contenus dans ce bloc seront appliqués lors de la prévisualisation d'impression sous tous les appareils dotés d'un écran

2. Media Queries.

Caractéristiques médias :

- Permettent de décrire certaines caractéristiques spécifiques du navigateur de l'utilisateur, de la taille de son écran ou de l'environnement.
- Des expressions de caractéristiques permettent de tester leur présence ou leur valeur. Par exemple :
 - width : Largeur de la zone d'affichage
 - height : Hauteur de la zone d'affichage
 - aspect-ratio : Rapport hauteur/largeur de la zone d'affichage
 - orientation : Orientation de la zone d'affichage (portrait ou paysage)
 - resolution : Densité de pixel de l'appareil
 - color : L'appareil est capable d'afficher des couleurs

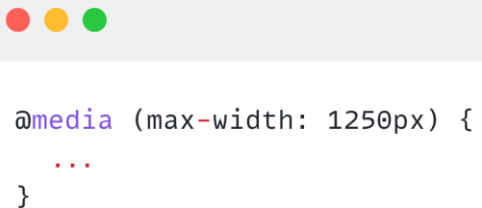
2. Media Queries.

Caractéristiques médias :

- L'ensemble de ces expressions caractéristiques peuvent être combinées avec des opérateurs logiques :
 - and : Permet de combiner plusieurs requêtes média en une seule
 - not : Permet d'utiliser le résultat opposé d'une requête média
 - only : Permet d'appliquer un style uniquement si la requête media est vérifiée
 - « , » : Fonctionne comme le groupement de sélecteurs. Si une des requêtes est valide, l'ensemble sera considéré comme valide.

2. Media Queries.

Cibler des caractéristiques média — Exemples :

A code editor window with a light gray header containing three colored circles (red, yellow, green). The main area is white and contains CSS code.

```
@media (max-width: 1250px) {  
  ...  
}
```

Les styles contenus dans ce bloc seront appliqués uniquement si la zone d’affichage ne dépasse pas 1250px de large (Certaines tablettes et téléphones par exemple)

A code editor window with a light gray header containing three colored circles (red, yellow, green). The main area is white and contains CSS code.

```
@media speech and (aspect-ratio: 11/5) {  
  ...  
}
```

Les styles contenus dans ce bloc ne seront jamais appliqués : Aucun appareil de synthèse vocale ne possède de caractéristiques relatives à la taille de son écran


2. Media Queries.

Combiner les types et caractéristiques de média :

- Il est évidemment de combiner les types et caractéristiques de média afin de pouvoir cibler avec efficacité les appareils auxquels nous souhaitons appliquer des styles.
- « Requêtes média complexes »
- Si aucun type n'est spécifié, le type de media par défaut est « all ».
 - Lorsque les caractéristiques not et only sont utilisées, il est obligatoire de spécifier un type de média

2. Media Queries.

Combiner les types et caractéristiques de média — Exemple :



```
@media (min-height: 680px), screen and (orientation: portrait) {  
  ...  
}
```

Pour appliquer les styles contenus dans ce bloc, il faut :

- Soit avoir un appareil avec une zone d’affichage haute supérieur à 680px (Si la page fait plus de 680px, une imprimante fonctionnera par exemple)
- Soit avoir un écran et être en format portrait (un smartphone par exemple)

2. Media Queries.

Breakpoints :

- « Points d'arrêt »
- Il s'agit des valeurs minimales et maximales permettant de définir le moment auquel la mise en page change via les media queries.
- La largeur est communément utilisée pour définir les breakpoints
- Ces derniers sont également exprimés communément en pixels

2. Media Queries.

Breakpoints standards :

- Il n'y a pas de norme officielle pour couvrir l'ensemble des écrans disponibles sur le marché, mais l'usage a permis de définir des tailles standards efficaces :
 - 320px — 480px : Appareils mobiles
 - 481px — 768px : Tablettes, iPads
 - 769px — 1024px : Petits écrans, ordinateurs portables
 - 1025px — 1200px : Ordinateurs de bureau, grands écrans
 - 1201px et plus : Très grands écrans, télévisions
- Ces tailles peuvent évoluer rapidement (en 2014 par exemple, les tailles d'écrans mobiles étaient radicalement différentes de celles d'aujourd'hui)

Exemples de tailles de zones d'affichages de différents iPhones— yesviz.com/iphones.php

2. Media Queries.

Media Queries et breakpoints — Exemple d'usage :



```
/* Extra small devices (phones, 600px and down) */
@media only screen and (max-width: 600px) {...}

/* Small devices (portrait tablets and large phones, 600px and up) */
@media only screen and (min-width: 600px) {...}

/* Medium devices (landscape tablets, 768px and up) */
@media only screen and (min-width: 768px) {...}

/* Large devices (laptops/desktops, 992px and up) */
@media only screen and (min-width: 992px) {...}

/* Extra large devices (large laptops and desktops, 1200px and up) */
@media only screen and (min-width: 1200px) {...}
```

2. Media Queries.



3. Flexbox.

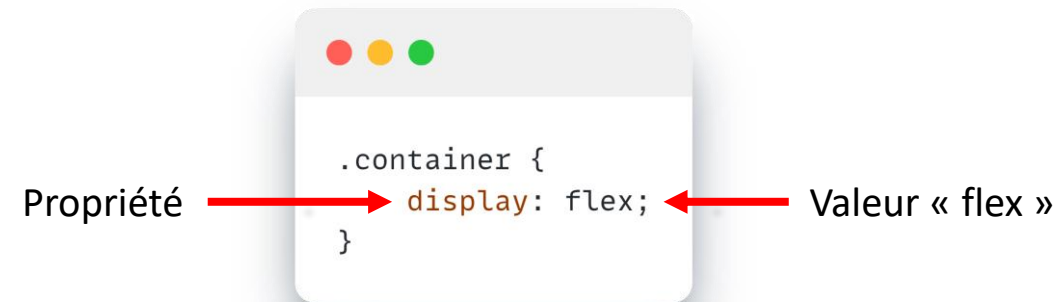
3. Flexbox.

Qu'est-ce que c'est :

- Jusqu'à maintenant, le contenu de vos pages n'a été qu'une succession d'éléments contenus dans une unique grande colonne
- Flexbox permet de mettre en page le contenu en lignes ou en colonnes, tout en s'adaptant aux contraintes de tailles afin d'occuper l'ensemble de l'espace disponible.

3. Flexbox.

Première approche :



Tous les articles :

Article 01.

omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et recusandae. Itaque earum rerum hic aut officiis debitis aut rerum necessitatibus tenetur a sapiente delectus, ut aut saepe eveniet ut et voluptates

Article 02.

reiciendis voluptatibus maiores

```
<body>  
  <h2>Tous les articles :</h2>  
  <div class="container">  
    <article>  
      <h3>Article 01.</h3>  
      <p>  
        omnis voluptas assumenda est, omnis dolor  
        repellendus. Temporibus autem quibusdam et aut  
        officiis debitis aut rerum necessitatibus saepe  
        eveniet ut et voluptates  
      </p>  
    </article>  
    <article>  
      <h3>Article 02.</h3>  
      <p>  
        repudiandae sint et molestiae non recusandae.  
        Itaque earum rerum hic tenetur a sapiente delectus,  
        ut aut reiciendis voluptatibus maiores  
      </p>  
    </article>  
  </div>  
</body>
```

3. Flexbox.

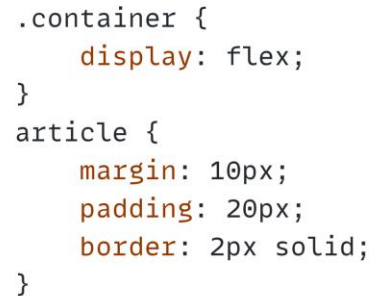
Première approche :

- L’affichage est maintenant en ligne pour l’ensemble des éléments se trouvant dans le conteneur ayant la classe « container »
- Les éléments enfants de ce conteneur s’adaptent à la taille de la zone d’affichage
- On peut ajouter un peu de style pour que les contenus ne se superposent pas

3. Flexbox.

Première approche — Exemple :

Tous les articles :



```
.container {  
  display: flex;  
}  
article {  
  margin: 10px;  
  padding: 20px;  
  border: 2px solid;  
}
```

Article 01.

omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates

Article 02.

repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

3. Flexbox.

Colonnes et lignes :

- Par défaut, Flexbox passe le contenu en ligne, mais il est possible de changer cela grâce à la propriété « flex-direction »
- Cette dernière propose plusieurs directions :
 - row : Par défaut, en rangée
 - column : En colonne
 - row-reverse : En rangée, mais le contenu est inversé
 - column-reverse : En colonne, mais le contenu est inversé

3. Flexbox.

Colonnes et lignes — Exemples :

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

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

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

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

Tous les articles :

Article 01.
omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates

Article 02.
repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

Tous les articles :

Article 01.
omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates

Article 02.
repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

Tous les articles :

Article 02.
repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

Article 01.
omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates

Tous les articles :

Article 02.
repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

Article 01.
omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates

3. Flexbox.

Flex-wrap :

- S'il y a trop de contenu dans le conteneur, ce dernier ne sera pas capable d'afficher le tout et débordera (« overflow »)
- Une propriété permet de prévenir ce cas « flex-wrap » en fonction de la valeur passée:
 - « nowrap » : Déborde de l'écran si trop de contenu et permet de scroller pour voir le reste
 - « wrap » : Disposera les éléments sur une nouvelle ligne si trop de contenu
 - « wrap-reverse » : Disposera les éléments sur une nouvelle ligne et les inversera si trop de contenu

3. Flexbox.

Flex-wrap — Exemples :

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

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

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

Tous les articles :

Article 01.

omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates

Article 02.

repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

Article 03.

repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

Tous les articles :

Article 01.

omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates

Article 02.

repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

Article 03.

repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

Tous les articles :

Article 03.

repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

Article 01.

omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates

Article 02.

repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

3. Flexbox.

Taille adaptative :

- Comme expliqué précédemment, Flexbox permet d'occuper tout l'espace disponible grâce à ses éléments flexibles.
- Il est possible de hiérarchiser l'espace que des éléments peuvent prendre grâce à la propriété « flex » :
 - Par défaut, la valeur flex vaut 1 :
 - Chaque élément occupe autant d'espace
 - Si la valeur flex d'un élément change, un calcul est appliqué pour définir la taille qu'il occupe

3. Flexbox.

Taille adaptative — Exemple :

- Si j'ai 3 éléments et que modifie la valeur flex à 2 pour le premier élément, le calcul suivant est appliqué :
 - Total des valeurs = $1 + 1 + 2 = 4$
 - Mon premier élément vaut 2 parts, il occupe donc $\frac{1}{2}$ de l'espace
 - Mon deuxième élément vaut 1 part, il occupe donc $\frac{1}{4}$ de l'espace
 - Mon troisième élément vaut 1 part, il occupe donc $\frac{1}{4}$ de l'espace

```
article {  
  margin: 10px;  
  padding: 20px;  
  border: 2px solid;  
  width: 200px;  
  flex: 1;  
}  
article:nth-child(1) {  
  flex: 2;  
}
```

Article 01. omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates	Article 02. repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores	Article 03. repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores
---	--	--

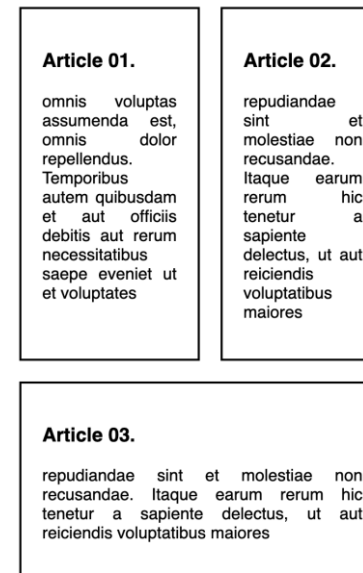
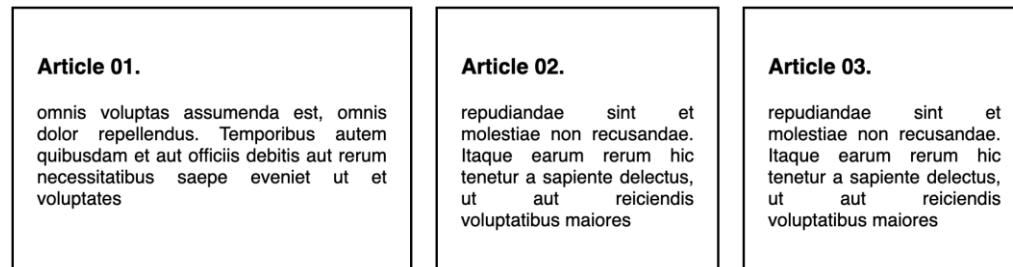
3. Flexbox.

Taille adaptative :

- Il est aussi possible d'ajouter une taille minimale à la propriété flex :
 - La hiérarchie de l'occupation de l'espace restant ne se fera que si les éléments ont leur taille minimum respectée

On constate ici que le premier article est légèrement plus grand que le second mais qu'il ne fait pas le double de sa taille.

Le dernier article occupe l'ensemble de l'espace disponible grâce au flex-wrap du conteneur



3. Flexbox.

Alignement horizontal et vertical :

- Flexbox permet d'aligner les éléments de diverses façons au sein de leur conteneur via deux propriétés :
 - « align-items » : Permet gérer l'alignement horizontal pour une colonne (inversé si rangée)
 - « justify-content » : Permet gérer l'alignement vertical pour une colonne (inversé si rangée)
- Ces deux propriétés fonctionnent aussi bien avec des colonnes que des rangées

3. Flexbox.

align-items :

- « align-items » peut prendre plusieurs valeurs :
 - « stretch » : Valeur par défaut. Les éléments occupent tout l'espace disponible
 - « center » : Centre les éléments
 - « start » : Aligne les éléments au début du conteneur
 - « end » : Aligne les éléments à la fin du conteneur
- Il est possible d'appliquer une propriété « align-self » à un enfant du conteneur pour gérer individuellement son alignement

3. Flexbox.

align-items — Exemples :

```
.container {  
  display: flex;  
  flex-direction: column;  
  align-items: start;  
}  
article {  
  margin: 10px;  
  padding: 20px;  
  border: 2px solid;  
  width: 100px;  
}
```

Tous les articles :

Article 01.

omnis
voluptas
assumenda
est, omnis
dolor
repellendus.
Temporibus
autem
quibusdam et
aut officiis
debitis aut
rerum
necessitatibus
saepe eveniet
ut et
voluptates

```
.container {  
  display: flex;  
  flex-direction: column;  
  align-items: center;  
}  
article {  
  margin: 10px;  
  padding: 20px;  
  border: 2px solid;  
  width: 100px;  
}
```

Tous les articles :

Article 01.

omnis
voluptas
assumenda
est, omnis
dolor
repellendus.
Temporibus
autem
quibusdam et
aut officiis
debitis aut
rerum
necessitatibus
saepe eveniet
ut et
voluptates

```
.container {  
  display: flex;  
  flex-direction: column;  
  align-items: center;  
}  
article {  
  margin: 10px;  
  padding: 20px;  
  border: 2px solid;  
  width: 100px;  
}
```

Tous les articles :

Article 01.

omnis
voluptas
assumenda
est, omnis
dolor
repellendus.
Temporibus
autem
quibusdam et
aut officiis
debitis aut
rerum
necessitatibus
saepe eveniet
ut et
voluptates

3. Flexbox.

align-self — Examples :

```
article {  
  margin: 10px;  
  padding: 20px;  
  border: 2px solid;  
  width: 100px;  
}  
article:nth-child(1) {  
  align-self: start;  
}
```

Article 01.

omnis
voluptas
assumenda
est, omnis
dolor
repellendus.
Temporibus
autem
quibusdam et
aut officiis
debitis aut
rerum
necessitatibus
saepe eveniet
ut et
voluptates

Article 02.

repudiandae
sint et
molestiae non
recusandae.
Itaque earum
rerum hic
tenetur a
sapiente
delectus, ut
aut reiciendis
voluptatibus
maiores

```
article {  
  margin: 10px;  
  padding: 20px;  
  border: 2px solid;  
  width: 100px;  
}  
article:nth-child(1) {  
  align-self: start;  
}  
article:nth-child(2) {  
  align-self: end;  
}
```

Article 01.

omnis
voluptas
assumenda
est, omnis
dolor
repellendus.
Temporibus
autem
quibusdam et
aut officiis
debitis aut
rerum
necessitatibus
saepe eveniet
ut et
voluptates

Article 02.

repudiandae
sint et
molestiae non
recusandae.
Itaque earum
rerum hic
tenetur a
sapiente
delectus, ut
aut reiciendis
voluptatibus
maiores

3. Flexbox.

justify-content :

- « justify-content » peut prendre plusieurs valeurs :
 - « start » : Aligne les éléments au début du conteneur
 - « center » : Centre les éléments
 - « space-between » : Éléments répartis équitablement, aligné sur le début du conteneur
 - « space-around » : Éléments répartis équitablement, avec le même espace autour de chaque élément (entre deux éléments, il y a donc 2 fois plus d'espace qu'entre un élément et l'extrémité du conteneur)
 - « space-evenly » : Éléments répartis équitablement, avec le même espace entre les éléments et l'extrémité du conteneur

3. Flexbox.

justify-content — Exemples :

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

Tous les articles :



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

Tous les articles :



```
.container {  
  display: flex;  
  flex-direction: row;  
  justify-content: space-between;  
}
```

Tous les articles :



```
.container {  
  display: flex;  
  flex-direction: row;  
  justify-content: space-around;  
}
```

Tous les articles :



```
.container {  
  display: flex;  
  flex-direction: row;  
  justify-content: space-around;  
}
```

Tous les articles :



3. Flexbox.

Ordonner les éléments :

- Un autre avantage des éléments flexibles, c'est qu'il est possible de les ordonner sans toucher à l'HTML concerné

```
article:nth-child(1) {  
  order: 1;  
}  
article:nth-child(2) {  
  order: 3;  
}  
article:nth-child(3) {  
  order: 2;  
}
```

Tous les articles :

Article 01.

Article 03.

Article 02.

```
<body>  
  <h2>Tous les articles :</h2>  
  <div class="container">  
    <article>  
      <h3>Article 01.</h3>  
    </article>  
    <article>  
      <h3>Article 02.</h3>  
    </article>  
    <article>  
      <h3>Article 03.</h3>  
    </article>  
  </div>  
</body>
```

3. Flexbox.

Imbrication de Flexbox :

- Pour réaliser une mise en page satisfaisante, il est quasiment nécessaire d'imbriquer des colonnes et rangées

Contenu principal

Article 01.

omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates

Tous les articles :

Article 02.

omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates

Article 03.

repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

Article 04.

repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores

```
<body>
  <div class="col">
    <div class="row">

      <main class="main-content">
        <h2>Contenu principal</h2>
        <article class="flex-1">
          <h3>Article 01.</h3>
          <p>
            omnis voluptas assumenda est, omnis dolor repellendus.
            Temporibus autem quibusdam et aut officiis
            debitis aut rerum necessitatibus saepe eveniet ut
            et voluptates
          </p>
        </article>
      </main>

      <aside class="aside-content">
        <h2>Tous les articles </h2>
        <div class="col">
          <article>
            <h3>Article 02.</h3>
            <p>
              omnis voluptas assumenda est, omnis dolor repellendus.
              Temporibus autem quibusdam et aut officiis
              debitis aut rerum necessitatibus saepe eveniet ut
              et voluptates
            </p>
          </article>
          <article>
            <h3>Article 03.</h3>
            <p>
              repudiandae sint et molestiae non recusandae.
              Itaque earum rerum hic tenetur a sapiente delectus,
              ut aut reiciendis voluptatibus maiores
            </p>
          </article>
          <article>
            <h3>Article 04.</h3>
            <p>
              repudiandae sint et molestiae non recusandae.
              Itaque earum rerum hic tenetur a sapiente delectus,
              ut aut reiciendis voluptatibus maiores
            </p>
          </article>
        </div>
      </aside>
    </div>
  </div>
</body>
```

```
body {
  font-family: Helvetica, sans-serif;
  text-align: justify;
}
.col {
  display: flex;
  flex-direction: column;
}
.row {
  display: flex;
  flex-direction: row;
  flex-wrap: wrap;
  margin: 5px;
}
article {
  margin: 2px;
  padding: 10px;
  border: 2px solid;
}
.main-content {
  display: flex;
  flex-direction: column;
  flex-wrap: wrap;
  align-items: stretch;
  flex: 2;
}
.aside-content {
  display: flex;
  flex-direction: column;
  flex-wrap: wrap;
  flex: 1;
  min-width: 300px;
}
.flex-1 {
  flex: 1;
}
```

3. Flexbox.



4. Grid.

4. Grid.

Qu'est-ce que c'est :

- Les « Grid » (ou grilles) est une façon alternative à Flexbox de disposer le contenu sur une page
- Les grilles permettent de découper une page HTML en lignes et colonnes, sans imposer de structure de contenu :
 - Il est donc possible, par exemple de positionner des éléments d'une grille pour qu'ils puissent se chevaucher

4. Grid.

Comment structurer une page avec les grilles :

1. Créer une grille dans son conteneur principal
 1. Spécifier le nombre de colonnes
 2. Spécifier la fraction d'occupation de l'espace par les colonnes
2. Définir l'espacement entre chaque colonnes et lignes
3. Définir la hauteur le nombre de lignes et leur hauteur
4. Positionner les éléments au sein de la grille

4. Grid.

Créer une grille — Exemple :

Propriété gérant le type de mise en page (flex ou grid par exemple)

Chaque ligne ou colonne a une marge est espacée de 10px avec la suivante

```
.wrapper {  
  display: grid;  
  grid-template-columns: repeat(3, 1fr);  
  grid-gap: 10px;  
  grid-auto-rows: minmax(100px, auto);  
}
```

Valeur « grid »

Génère 3 colonnes d'une fraction (donc 1/3 par colonne)

Les lignes sont créées automatiquement si du contenu est placé dessus

Taille minimale de la ligne

Taille maximale de la ligne

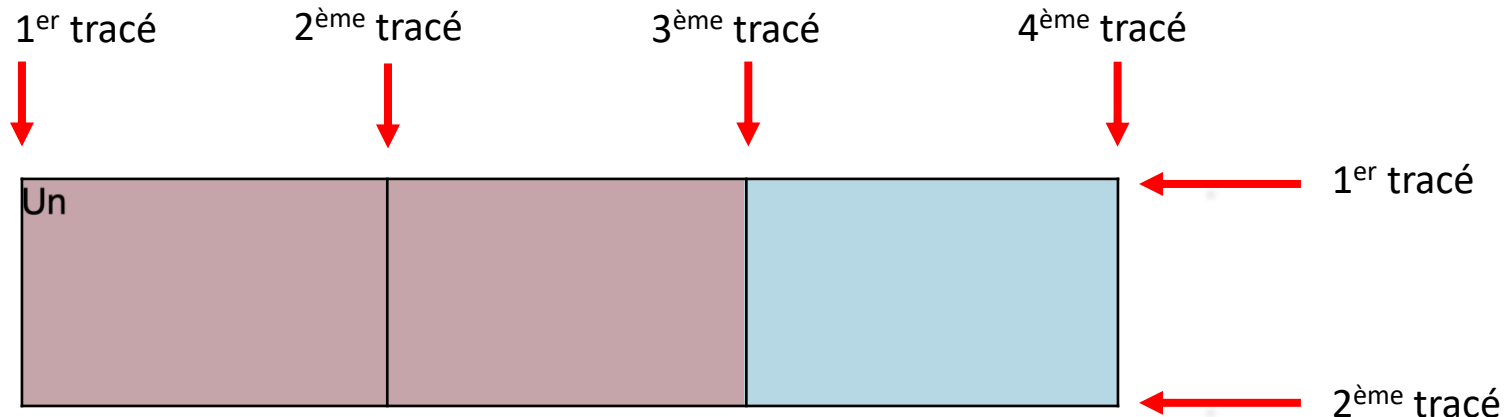
4. Grid.

Disposer des éléments sur une grille — Exemple :

```
.one {  
  grid-column: 1 / 3;  
  grid-row: 1;  
  background-color: rgb(255, 0, 0, 0.3);  
}
```

Commence au premier tracé
« virtuel » et se fini au troisième

Ligne commençant au
premier tracé « virtuel »

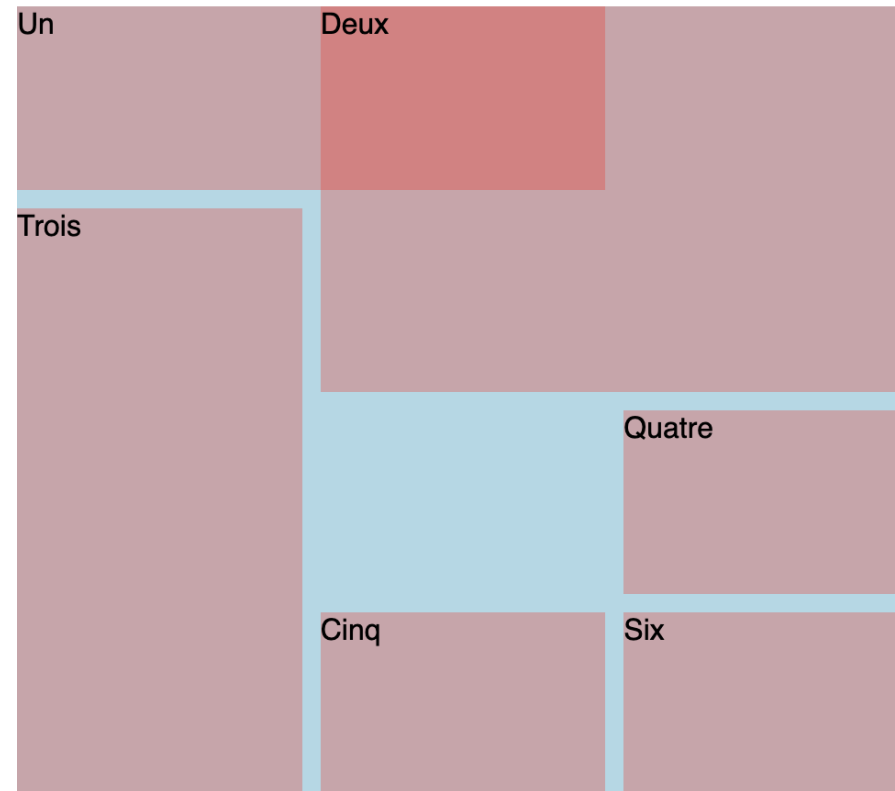


Les tracés sont « virtuels »
car ils ne sont pas affichés
en CSS

4. Grid.

Disposer plusieurs éléments sur une grille — Exemple :

```
<body>
  <div class="wrapper">
    <div class="one">Un</div>
    <div class="two">Deux</div>
    <div class="three">Trois</div>
    <div class="four">Quatre</div>
    <div class="five">Cinq</div>
    <div class="six">Six</div>
  </div>
</body>
```



```
.wrapper {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  grid-gap: 10px;
  grid-auto-rows: minmax(100px, auto);
}

.one {
  grid-column: 1 / 3;
  grid-row: 1;
}

.two {
  grid-column: 2 / 4;
  grid-row: 1 / 3;
}

.three {
  grid-column: 1;
  grid-row: 2 / 5;
}

.four {
  grid-column: 3;
  grid-row: 3;
}

.five {
  grid-column: 2;
  grid-row: 4;
}

.six {
  grid-column: 3;
  grid-row: 4;
}

div {
  background-color: lightblue;
}

div > div {
  background-color: rgb(255, 0, 0, 0.3);
}
```

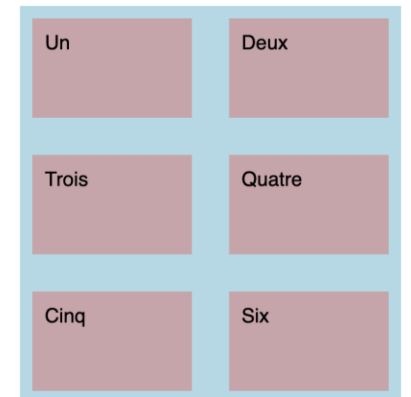
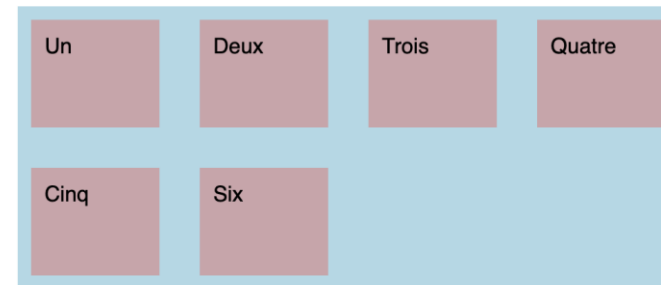
4. Grid.

Positionnement automatique des éléments :

- Le principal avantage des grilles est le positionnement automatique et responsive des éléments enfants :
 - Il suffit d'ajouter un nouveau conteneur en HTML pour qu'il s'ajoute à la grille

```
<body>
  <div class="wrapper">
    <div>Un</div>
    <div>Deux</div>
    <div>Trois</div>
    <div>Quatre</div>
    <div>Cinq</div>
    <div>Six</div>
  </div>
</body>
```

```
.wrapper {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(100px, 1fr));
  grid-gap: 10px;
  grid-auto-rows: minmax(100px, auto);
  background-color: lightblue;
}
div > div {
  background-color: rgb(255, 0, 0, 0.3);
  margin: 10px;
  padding: 10px;
}
```



4. Grid.

Quelques informations supplémentaires :

- Les possibilités offertes par les grilles étant quasiment infinies (imbrications de grilles, définition de zones, etc.), il est difficile de tout évoquer ou de connaître par cœur.
- Des exemples de mise en pages sont disponibles sur le site gridbyexample.com
- La complexité des grilles fait qu'elles ne sont utilisées que pour des mises en pages simples (Voir le site d'[airbnb](https://airbnb.com)), en utilisant le positionnement automatique.

4. Grid.



