CSS - Style des listes

Les listes ordonnées

Par défaut, la numérotation des éléments inclus dans un élément <ol> s'effectue en chiffres. Nous pouvons faire varier cette numérotation en attribuant un style à l'élément à l'aide de la propriété **list-style-type** dont la syntaxe est :list-style-type :<type>

NB : Plus généralement, cette propriété ne s'applique qu'aux éléments de listes (<ol>, <ul>)

Pour les listes ordonnées, le paramètre <type> peut prendre une des valeurs suivantes :

- **decimal** : numérotation en chiffres : 1, 2, 3… C'est la valeur par défaut.
- **decimal-leading-zero** : idem, mais les nombres inférieurs à 10 sont précédés d'un zéro : 01, 02, 03… Cette valeur n'est pas prise en compte par Internet Explorer.
- **upper-latin** ou **upper-alpha** : numérotation alphabétique en majuscules : A, B, C…
- **lower-latin** ou **lower-alpha** : numérotation alphabétique en minuscule : a, b, c…
- **upper-roman** : numérotation en chiffres romains majuscules I, II, III, IV…
- **lower-roman** : numérotation en chiffres romains minuscules i, ii, iii, iv…
- **lower-greek** : numérotation en caractères grecs minuscules, a, b, y… Cette valeur n'est pas prise en compte par Internet Explorer.
- **georgian** : numérotation en caractères géorgiens. Cette valeur n'est pas prise en compte par Internet Explorer.
- **armenian** : numérotation en caractères arméniens. Cette valeur n'est pas prise en compte par Internet Explorer.

La propriété **list-style-position** permet de définir la position du marqueur de liste par rapport à la boite principale dont la syntaxe est la suivante : list-style-position :inside|outside|inherit.

- **inside**, les caractères de la numérotation sont placés dans la marge de retrait du texte de l'item.
- **outside**, qui est la valeur par défaut, ces caractères sont intégrés au texte de l'item.
- **inherit** permet de définir explicitement le style de l'élément parent mais la propriété est de toute façon héritée par défaut.

Les listes à puces

Pour les listes à puces, la syntaxe de la propriété **list-style-type** est simplifiée et se résume à ceci :

list-style-type:disc | circle | square | none | inherit

- **disc** : la puce est un disque plein (c'est la valeur par défaut) ;
- **circle** : la puce est un cercle ;
- **square** : la puce est un carré plein ;
- **none** : pas de puce.

Les puces graphiques

La propriété **list-style-image** définit l'image utilisée comme puce devant les éléments de listes.

Affichage des listes en ligne

La propriété **display** permet de modifier le comportement des listes en lui attribuant la valeur inline. Dans ce cas, les items d'une liste s'affichent sur la même ligne.

CSS - Bordures, Marges, Espacements et les Ombres

Les bordures

La création de bordures autour d'un élément permet d'obtenir des effets visuels qui attirentl'attention et permettent également par exemple de mettre davantage en évidence lastructure de la page. En CSS, nous pouvons attribuer à une bordure des caractéristiquesvariées, telles qu'un style, largeur et couleur.

Lespropriétés CSS qui permettent d'indiquer la bordure sont **border, border-width, border-color, border-style…**

Pour **border** on peut utiliser jusqu'à trois valeurs pour modifier l'apparence de la bordure :

- **border-width** : indiquez la largeur de votre bordure. (valeur en pixels)
- **border-color** : couleur de votre bordure.
- **border-style**: là, vous avez le choix. Votre bordure peut être un simple trait, ou des pointillés, ou encore des tirets, voici les différentes valeurs disponibles :
  - none : pas de bordure (par défaut) ;
  - solid : un trait simple ;
  - dotted : pointillés ;
  - dashed : tirets ;
  - double : bordure double ;
  - groove : en relief ;
  - ridge : autre effet relief ;
  - inset : effet 3D global enfoncé ;
  - outset : effet 3D global surélevé.

Exemple : h1 { border: 3px bluedashed; } (bordure bleue, en tirets, épaisseur de 3 pixels)

Pour mettre des bordures différentes en fonction du côté (haut, bas, gauche ou droite), vous devrez utiliser ces quatre propriétés :

- border-top : bordure du haut
- border-bottom : bordure du bas
- border-left : bordure de gauche
- border-right : bordure de droite

Il existe aussi des équivalents pour paramétrer chaque détail de la bordure :

- border-top-width pour modifier l'épaisseur de la bordure du haut,
- border-top-color pour la couleur du haut, etc.

Exemple :

p {border-left: 2px solid black; border-right: 2px solid black; }

Bordures arrondies

La propriété **border-radius**permet d'arrondir facilement les angles de n'importe quel élément. Il suffit d'indiquer la taille de l'arrondi en pixels :

Exemple : p{border-radius: 10px;}

Il est possible de préciser la forme de l'arrondi pour chaque coin grâce à la syntaxe suivante :

border-radius: 10px 5px 10px 5px; dont les valeurs correspondent aux angles dans cet ordre :

- en haut à gauche ;
- en haut à droite ;
- en bas à droite ;
- en bas à gauche.

Les ombres

Les ombres des boîtes : box-shadow

La propriété box-shadow s'applique à tout le bloc et prend quatre valeurs dans l'ordre suivant :

- le décalage horizontal de l'ombre ;
- le décalage vertical de l'ombre ;
- l'adoucissement du dégradé ;
- la couleur de l'ombre.

Exemple : p { box-shadow: 6px 6px6px black;}

L'ombre du texte : text-shadow

La propriété text-shadow permet d' ajouter l'ombre directement sur les lettres de votre texte. Les valeurs fonctionnent exactement de la même façon que box-shadow.

Exemple : p{ text-shadow: 2px 2px 4px black;}

Les marges

Permettent d'aérer le contenu d'une page et en particulier l'espace entre un élément et ses voisins dans la page.

La propriété CSS qui permet de définir la largeur des marges d'un élément, est **margin (dispose de quatre valeurs: haut, droit, bas, gauche).**
Exemple : p {margin: 100px 150px 100px 80px;}

Pour mettre des marges différentes en fonction du côté (haut, bas, gauche ou droite), vous devrez utiliser ces quatre propriétés :

- margin-top : définit la marge haute
- margin-right : définit la marge droite
- margin-bottom : définit la marge basse
- margin-left : définit la marge haute

Exemple :

```
p {
   margin-top: 100px;
   margin-bottom: 100px;
   margin-right: 150px;
   margin-left: 80px;}
```

Les espacements

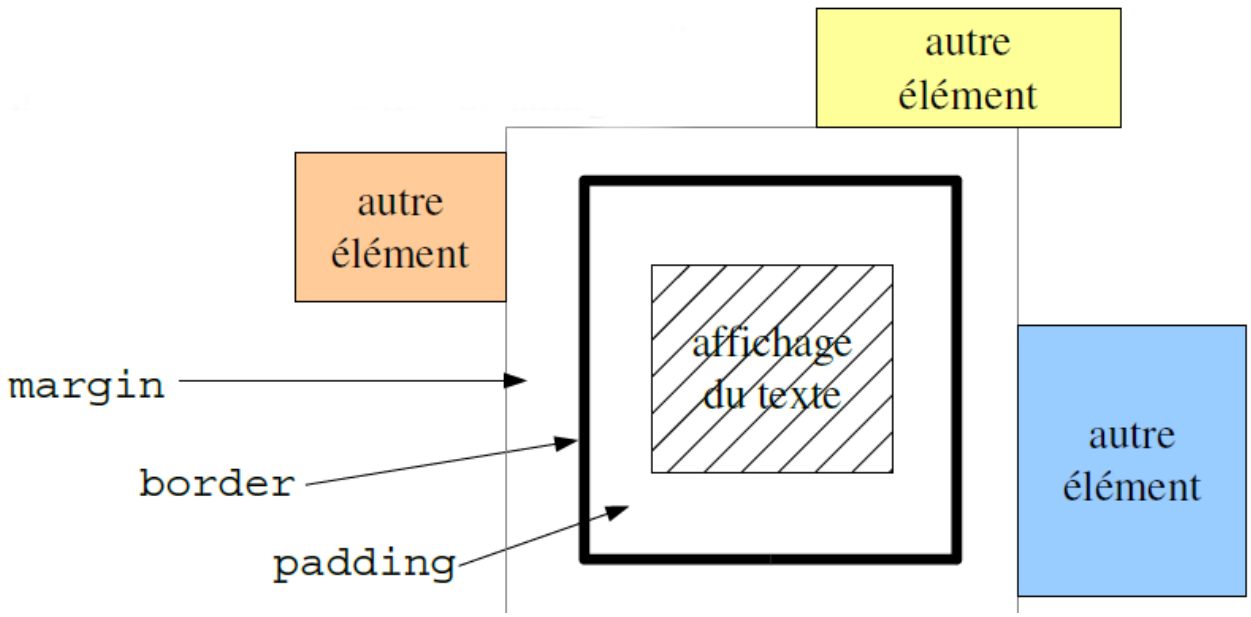
La propriété **padding** permet de générer de l'espace autour du contenu **dispose de quatre valeurs: haut, droit, bas, gauche).**
Exemple : p {padding: 50px 30px 50px 80px;}

Il est encore possible de définir individuellement chacun des espacements d'un élément au moyen despropriétés suivantes :

- padding-top : définit l'espacement haut ;
- padding-right : définit l'espacement droit ;
- padding-bottom : définit l'espacement bas ;
- padding-left : définit l'espacement gauche.

Exemlpe :

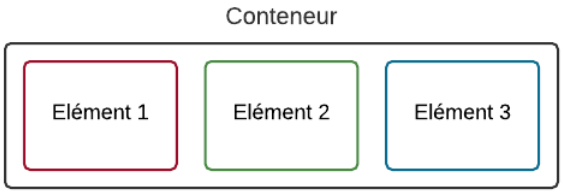
p {padding-top: 50px; padding-right: 30px;padding-bottom: 50px; padding-left: 80px;}



CSS - La mise en page avec Flexbox

Un conteneur des éléments

Le principe de la mise en page avec **Flexbox** est simple : vous définissez un conteneur, et à l'intérieur vous placez plusieurs éléments.



Le conteneur est une balise HTML, et les éléments sont des balises HTML à l'intérieur :

```
<div id="conteneur">

<div class="element">Elément 1</div>

<div class="element">Elément 2</div>

<div class="element">Elément 3</div>

</div>
```

La propriété qui permet d'afficher les blocs du conteneur côte à côte est :**display: flex**;

Exemple :#conteneur{display: flex;}

La direction

La propriété CSS **flex-direction**, permet de positionner les éléments du conteneur dans le sens que l'on veut. Il peut prendre les valeurs suivantes :

- **row** : organisés sur une ligne (par défaut)
- **column**: organisés sur une colonne
- **row-reverse** : organisés sur une ligne, mais en ordre inversé
- **column-reverse** : organisés sur une colonne, mais en ordre inversé

Exemple : #conteneur{display: flex; flex-direction: column;}

Le retour à la ligne

La propriété CSS **flex-wrap**, permet de renvoyer les éléments à la ligne lorsqu'il n'y a plus de place. Il peut prendre les valeurs suivantes :

- **nowrap**: pas de retour à la ligne (par défaut)
- **wrap** : les éléments vont à la ligne lorsqu'il n'y a plus la place
- **wrap-reverse** : les éléments vont à la ligne lorsqu'il n'y a plus la place en sens inverse

Exemple : #conteneur{display: flex; flex-wrap: wrap;}

Alignement des éléments du conteneur

Pour changer l'alignement, on utiliser la propriété CSS **justify-content**, qui peut prendre ces valeurs :

- **flex-start** : alignés au début (par défaut)
- **flex-end** : alignés à la fin
- **center** : alignés au centre
- **space-between** : les éléments sont étirés sur tout l'axe (il y a de l'espace entre eux)
- **space-around** : les éléments sont étirés sur tout l'axe, mais ils laissent aussi de l'espace sur les extrémités

Exemple :#conteneur{display: flex;justify-content: space-around;}

Agrandir les éléments du conteneur

La propriété **flex** permet d'agrandir un élément du conteneur pour occuper tout l'espace restant.

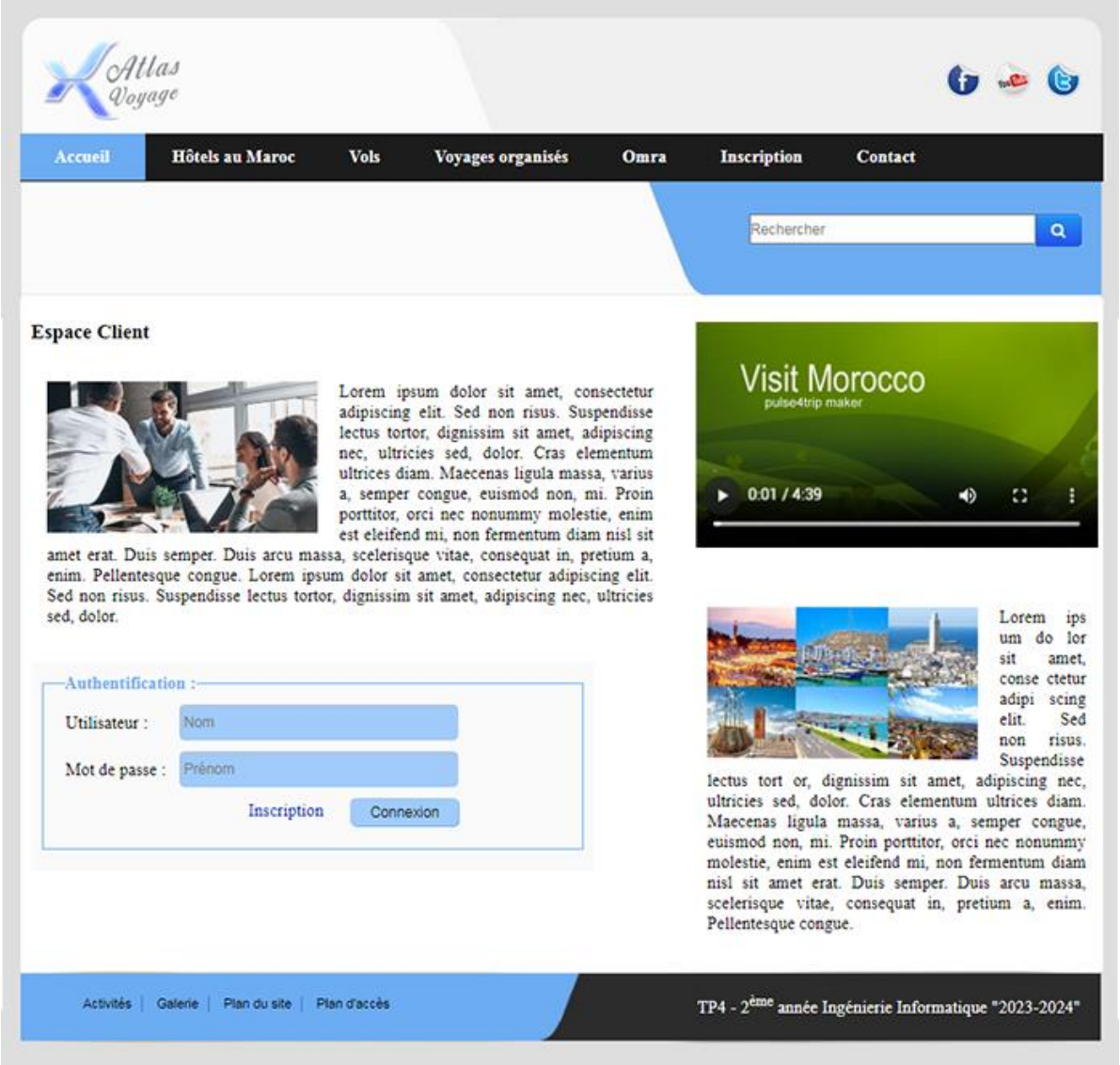
Syntaxe : .element:nth-child(2){flex: 1;}

TP5 HTML & CSS

Créer les menus ci-dessous :

TP6 HTML & CSS

Le but est de mettre en forme la page d'accueil "**index.html**" de la société "**Atlas Voyage**"(**figure ci-dessous**) à l'aide d'une feuille de style externe nommée **style.css** destinée à l'affichage sur écran.



Structure de la page web "index.html" :