

1. Objectifs du TP

- Comprendre le modèle de boîte "classique" et le modèle de boîte "flexible"
- Comprendre le mode de positionnement CSS "flex"

Ce sujet est consultable en version PDF [sur Chamilo](#).

2. Préparation et introduction

À faire

1. Créer un répertoire **TP3** dans le répertoire **r1_02** ;
2. Se positionner dans ce répertoire ;
3. Recopier dans votre répertoire **TP3** le contenu de [l'archive](#).

Il y a plusieurs façons de mettre en page un site et donc de positionner les divisions les unes par rapport aux autres. Les techniques sont variées et ont évolué au cours du temps. Le positionnement "flex" est actuel et il faut savoir le maîtriser, même si d'autres modes de positionnement peuvent s'avérer utiles dans certains cas.

3. Le modèle de boîte (modèle classique)

3.1. Introduction au modèle de boîte classique

Rappel: tout élément HTML est une « boîte » et possède:

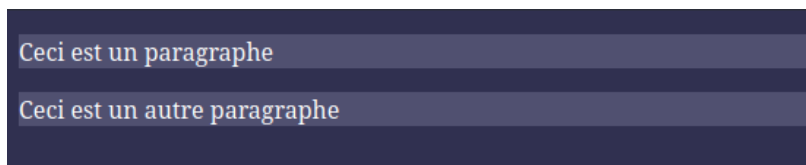
- Un contenu : le texte d'un paragraphe par exemple, ou une autre boîte.
- Une marge interne : padding.
- Une marge externe : margin.
- Une bordure : border.

Dans cette partie nous manipulons le modèle de boîte de CSS avec un exemple très simple. Le fichier **exo-modele-boite/page.html** contient 4 blocs (boîtes) imbriqués: **html**, **body** et deux paragraphes **p**.

Intéressons-nous à l'outil de Développement Web de Firefox. Cet outil est extrêmement utile et dès que vous faites du développement web vous devez l'activer.

À faire

- Visualiser le contenu de **page.html**
- Créer un fichier **page.css** et ajouter une couleur de fond à **body** (#303050), et aux paragraphes **p** (#505070). Définir les caractères de couleur blanche dans le selecteur **body**. Vous devez obtenir la page ci-dessous.



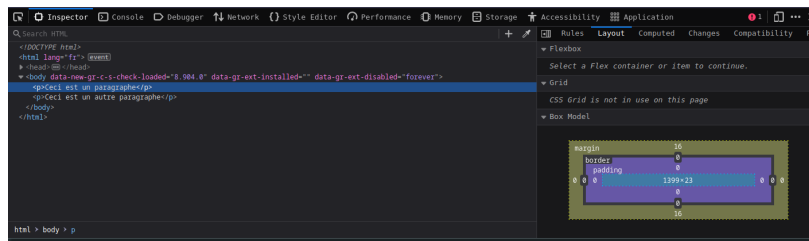
légende : Résultat attendu

- Activer l'outil (menu **Tools / Web Developer / Toggle Tools** ou bien raccourci clavier **Ctrl + Shift + K**) ;
- Ouvrir l'onglet inspecteur (Inspector) et le panneau latéral Règles (sous-onglet **Rule**) ;
- Remarquer que les balises **<p>** hérite de la couleur des caractères définie dans **body**.

On s'intéresse maintenant au modèle de boîtes.

À faire

Dans l'onglet inspecteur (Inspector), ouvrir le panneau latéral **Layout** (modèle de boîtes), comme sur la figure ci-dessous. Quelle est la largeur (width) des paragraphes ? Leur hauteur (height) ? Les valeurs des marges intérieures (padding) ? Des marges extérieures (margin) ?



légende : Modèle de boîte de Firefox

À faire

- Noter que par défaut, la boîte **p** a une marge extérieure (margin) haute et basse de 16 pixels.

Quelle est la taille en pixels de l'espace entre le paragraphe et le haut de la fenêtre ? D'où vient cette valeur ? Quelle est la taille de l'espace entre les deux paragraphes ? Que s'est-il passé ?

Vous devez constater que les marges extérieures (margin) entre les paragraphes ne s'additionnent pas. Ce mécanisme se nomme la **fusion des marges**. Ce mécanisme de fusion concerne **uniquement les marges extérieures hautes et basses** (**margin-top** et **margin-bottom**). Cette fusion s'opère entre éléments frères (qui se suivent, les 2 boîtes **p** dans notre cas), mais également entre un élément et son parent. Pour plus de détails, lire les explications sur le site [Alsacreation](https://alsacreation.com/fr/developpement-web/la-fusion-des-marges/)

3.1. Centrage d'une boîte dans sa boîte parent

Nous avons utilisé des valeurs de type pixels, il est possible d'utiliser une valeur relative de type pourcentage. Le pourcentage fait toujours référence à une autre valeur, celle de sa boîte parent.

À faire

- Ajouter dans le fichier CSS une largeur pour les paragraphes de 50% ;
- Dans l'inspecteur, onglet **Layout**, observer la taille de l'élément **body** et **p**. Vérifier que les paragraphes font bien la moitié de la page Web.
- Modifier le CSS pour placer une marge extérieure gauche aux boîtes **p** de 10 pixels.

Quelle est la largeur totale des boîtes **p** ?

- Supprimer la marge extérieure et ajouter une marge intérieure gauche de 10px ;
- Rajouter une bordure solide, blanche, de 2 pixels ;

Quelle est la largeur totale des boîtes **p** ? Qu'en déduisez-vous quant au calcul de la taille des boîtes ?

Largeur total = + +

⚠ Attention

Notez bien qu'en l'absence d'informations explicites sur la taille, une boîte prend toute la place disponible à l'intérieur de sa boîte parent.

La valeur **auto** sur les marges signifie que les marges sont calculées automatiquement en fonction du contexte des autres boîtes:

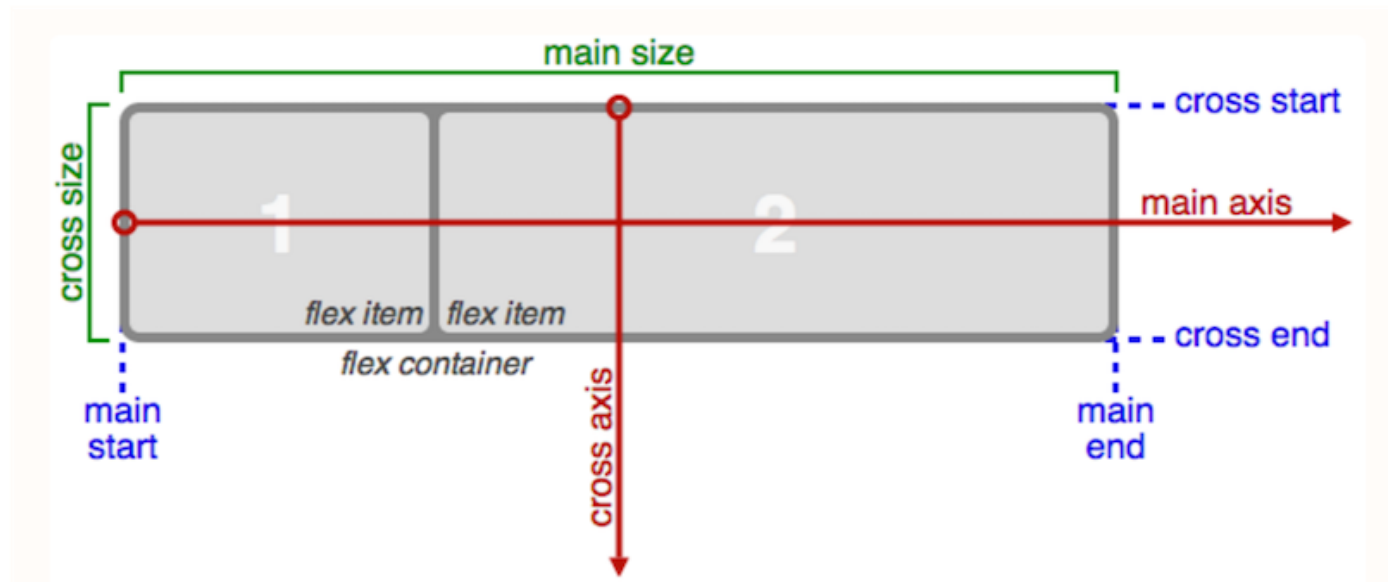
✎ À faire

- Changer les marges extérieures gauches et droites des paragraphes en **auto** ;
- Calculer les marges à l'aide de l'inspecteur.

4. Positionnement Flex

Le positionnement "flex" est actuel et est à privilégier. Il utilise un **modèle de boîte flexible (flexbox)** qui définit 2 axes: l'axe principal (main axis) et l'axe secondaire (cross axis).

Par défaut, l'axe principal est horizontal et l'axe secondaire vertical, mais il est possible d'inverser.



Le principe de base du positionnement flex est le suivant :

- un "flex-container" contient des éléments "flex-items" (ce sont les enfants directs du conteneur)
- les "flex-items" (ou plus simplement les "items") vont pouvoir être positionnés de différentes façons à l'intérieur du "flex-container"

Pour réaliser la suite du TP, n'hésitez pas à vous référer aux documents suivants car les possibilités sont nombreuses:

- le cours (pour une introduction)
- la [CSS3 Flexbox Layout module](#) sur le site Alsacrétions (en français)
- la [La mise en page avec Flexbox](#) sur le site OpenClassrooms (en français)

Axe principal d'affichage des items :

À faire

- Visualiser la page [exo-flex/pageFlex.html](#) dans le navigateur ;
- Observer que 3 items (ici ce sont des section) sont alignés horizontalement (c'est le mode par défaut) ;
- Ajouter dans la première règle du fichier [styleFlex.css](#) la propriété [flex-direction](#) et tester les différentes valeurs. Cette propriété s'applique au "flex-container" et détermine l'axe principal du modèle de boîte flexible.

Nous venons donc de voir que les items peuvent être alignés soit horizontalement (par défaut en mode "display: flex;"), soit verticalement. Cela définit ce qu'on appelle l'axe principal. Il y a également un axe secondaire (cross axis) :

- Si l'axe principal est horizontal, l'axe secondaire est l'axe vertical ;
- et inversement!

Le retour à la ligne des items (wrap) :

Supposons que l'axe principal est horizontal, par défaut les items essaient de rester sur la même ligne. Si l'espace est insuffisant, cela peut provoquer des "bugs" d'affichage. Il est possible de spécifier que les items doivent aller à la ligne lorsqu'ils n'ont plus la place grâce à la propriété [flex-wrap](#).

À faire

- Réduire la fenêtre du navigateur et regarder ce qu'il se passe. Les éléments restent sur la même ligne.
- Ajouter dans la première règle du fichier [styleFlex.css](#) la propriété [flex-wrap](#) et tester les différentes valeurs.

Alignement (répartition) des items sur l'axe principal :

Pour changer l'alignement sur l'axe principal, on utilise la propriété [justify-content](#).

À faire

- Ajouter dans la première règle du fichier [styleFlex.css](#) la propriété [justify-content](#) et tester les différentes valeurs ;
- Tester avec un axe principal horizontal, puis vertical.

Alignement (répartition) des items sur l'axe secondaire :

Pour changer l'alignement cette fois sur l'axe secondaire, on utilise la propriété [align-items](#).

À faire

- Ajouter dans la première règle du fichier [styleFlex.css](#) la propriété [align-items](#) et tester les différentes valeurs ;
- Tester avec un axe principal horizontal, puis vertical.

⚠ Attention

Notez que les différentes propriétés que nous avons vues ci-dessus s'appliquent au conteneur parent et impactent l'affichage des items. C'est une forte valeur ajoutée du positionnement Flex par rapport à d'autres positionnements qui nécessitent d'appliquer le style aux éléments enfants. Dans la suite, nous verrons que nous pouvons

Flexibilité des items :

La flexibilité des items est réalisée grâce à la propriété [flex](#). Cette propriété est un raccourci pour [flex-grow](#), [flex-shrink](#), et [flex-basis](#) :

flex-grow (valeur par défaut : 0) :

Permet de définir par quel facteur un item va s'élargir par rapport aux autres items appartenant au même conteneur. Par défaut, un item ne grossit pas (il conserve sa taille initiale).

flex-shrink (valeur par défaut : 1) :

Permet de définir par quel facteur un item va se rétrécir par rapport aux autres items appartenant au même conteneur. Par défaut, les items peuvent rétrécir du même facteur ;

flex-basis (valeur par défaut : auto):

Permet de définir la taille initiale d'un item. La valeur **auto** indique que la taille sera égale à la largeur (si l'axe principal est horizontal) ou la hauteur (si l'axe principal est vertical).

✎ À faire

- Appliquer la propriété **flex-basis**: 350px ; aux trois items et observer le résultat en redimensionnant la page ;
- Appliquer la propriété **flex-grow**: 1 ; au premier item et observer le résultat en redimensionnant la page ;
- Appliquer la propriété **flex-grow**: 2 ; au dernier item et observer le résultat en redimensionnant la page ;
- Appliquer la propriété **flex-shrink**: 2 ; au premier item et observer le résultat en réduisant la taille de la page de façon à ce que la taille des items diminuent en deça de la valeur spécifiée dans la propriété flex-basis ;
- Appliquer la propriété **flex-shrink**: 0 ; au premier item et observer le résultat en réduisant la taille de la page de façon à ce que la taille des items diminuent en deça de la valeur spécifiée dans la propriété flex-basis ;
- Combiner différentes valeurs au travers de la propriété raccourcie flex.


Pour finir, quelques exemples à réaliser (certains ont été réalisés précédemment):

Essai positionnement flex

item 1

ceci est le premier flex-item

item 2



item 3


ceci est le troisième flex-item

Essai positionnement flex

item 1

ceci est le premier flex-item

item 2



item 3


ceci est le troisième flex-item

Essai positionnement flex

item 1

ceci est le premier flex-item

item 2



item 3


ceci est le troisième flex-item

Essai positionnement flex

item 3

ceci est le troisième flex-item

item 2



item 1

ceci est le premier flex-item

5. Portfolio

Nous allons reprendre l'exercice du portfolio et appliquer du style CSS aux différents éléments de la page.

À faire

- Définir une largeur (flex-basis) à l'élément `<aside>` ;
- Faire en sorte que l'élément `<aside>` apparaisse au dessous de la zone principale de la page lorsque l'espace est insuffisant ;
- Positionner les liens de la barre de navigation selon vos envies : à gauche, droite, au centre, avec des espacement, etc. ;
- Positionner les liens des réseaux sociaux d'une manière différentes : à gauche, droite, au centre, avec des espacement, etc. ;
- Profiter du temps restant pour compléter les informations des différentes pages ;
- Téléverser le portfolio à nouveau sur Chamilo.

6. Pour aller plus loin

Jouez au jeu "Flexbox Froggy" (a game where you help Froggy and friends by writing CSS code!) : <http://flexboxfroggy.com/>

Si vous arrivez jusqu'au bout... c'est que vous maîtrisez parfaitement le positionnement "flex"! bravo!