CIR1

Web statique – travaux pratiques 4

Nathalie HELAL

1 Préambule

- Créez le répertoire TP4 (sur votre ordinateur portable) dans lequel vous mettrez l'ensemble de vos fichiers (html, css, …), répertoire d'images (si nécessaire) et autre documents (si besoin) pour ce quatrième TP.
- A la fin de ce TP, il faut déposer (sur campus.isen) l'ensemble de vos fichiers html, css, répertoire d'images, … **mais sous forme d'un seul dossier compressé .zip nommé comme indiqué dans le dépôt de ce TP4 sur** campus.isen**.** Attention, n'oubliez pas de numéroter (et de nommer) chaque exercice comme indiqué.

2 Objectif de ce TP

1. Réaliser un site web adaptatif (Responsive Web Design) (viewport, media queries);
2. Utiliser une grille "à la main" en flottant ;
3. Utiliser le système de grille CSS3 CSS Grid Layout.

3 Travail demandé

Vous vous rappelez les différents positionnements CSS revus au TP3

3.1 Site web adaptatif / Responsive Web Design

De nos jours un site web peut se consulter sur de nombreux appareils : écrans d'ordinateurs fixes ou portables, tablettes, téléphones mobiles, liseuses, interfaces mode texte / consoles, ...

Définition à retenir : un site web responsive est un site permettant une bonne expérience de lecture et de navigation sur n'importe quel appareil.

Regardez la documentation w3school sur le responsive web design disponible sur ce lien https://www.w3schools.com/css/css_rwd_intro.asp et celle de MDN sur le lien suivant https://developer.mozilla.org/fr/docs/D%C3%A9veloppement_Web/Design_web_Responsive

Vous allez voir dans ce TP les principaux moyens de réaliser de tels sites avec HTML 5 et CSS 3.

3.2 HTML viewport

La documentation w3school est ici.

Le "**viewport**" (qu'on pourrait traduire en français par "fenêtre de visualisation") correspond à la région visible de l'utilisateur. Elle sera par exemple plus petite sur un smartphone que sur un écran d'ordinateur.

Une balise **meta viewport**, placée dans l'en-tête, donne au navigateur des instructions de contrôle des dimensions et de l'échelle d'une page web.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

La propriété **width=device-width** demande à ce que la page s'adapte à la taille de l'écran de l'appareil de l'utilisateur.

La propriété **initial-scale=1.0** fixe le niveau de zoom initial à l'ouverture de la page.

- Testez les pages **sansviewport.html** et **avecviewport.html** situées dans les données de ce TP avec l'image placée dans le dossier images, en redimensionnant le navigateur pour voir le résultat de l'utilisation de cette balise meta.

Remarquez que dans le fichier **avecviewport.html**, les propriétés CSS de l'image ont été modifiée aussi pour occuper tout l'espace disponible sans déformer l'image (ne pas l'agrandir plus que sa taille initiale).

```
img {
   max-width: 100%;
   height: auto;
}
```

3.3 Positionnement float et taille en pourcentage avec une grille de 12 colonnes.

Vous pouvez ensuite utiliser des éléments **positionnés en float avec des tailles exprimées par des pourcentages** pour vous adapter automatiquement à la taille de l'écran. C'est ce que vous avez déjà fait au TP précédent.

Nous nous inspirons ici de ce tutoriel w3school.

On choisit de découper la largeur de l'écran en 12 colonnes, aussi chaque colonne a une largeur de 100/12=8,33%.

Chaque ligne du site sera constituée de blocs occupant les 12 colonnes. Exemples :

- Une ligne avec un bloc occupant 4 colonnes et un bloc de 8 colonnes.
- Une ligne composée de 4 blocs de 3 colonnes.
- Une ligne composée d'un bloc de 12 colonnes.

Chaque ligne devra occuper les 12 colonnes. On ne peut pas faire une ligne de 2 blocs de 4 colonnes par exemple.

- Vous allez réaliser le site illustré sur l'image suivante.

Ce site comporte 3 lignes :
- La première ligne (bloc titre en-tête "header" contenant "Mon site") est composée d'un bloc couvrant les 12 colonnes.
- La deuxième ligne est composée de 3 blocs :
  - Un premier bloc de 3 colonnes pour le menu.
  - Un deuxième bloc de 6 colonnes pour le contenu de la page.
  - Un troisième bloc de 3 colonnes pour le "blabla annexe".
- La troisième et dernière ligne (bloc bas de page "footer" contenant "Blabla bas de page") est composée d'un bloc couvrant les 12 colonnes.

- Nommez les pages html et css nécessaires pour réaliser ce site **siteGrillePourcentages.html** et **siteGrillePourcentages.css**

Voici comment réaliser ce site.

1. Pour commencer, il faut assurer que le rembourrage (padding, marge interne) et la bordure soient inclus dans la largeur et la hauteur des éléments (sinon les bordures des blocs se chevauchent).

```
* {
   box-sizing: border-box;
}
```

2. Pour réaliser ces colonnes, vous allez vous servir des propriétés css suivantes précisant des classes de bloc de largeurs de 1 à 12 colonnes.

```
.col-1 {width: 8.33%;}
.col-2 {width: 16.66%;}
.col-3 {width: 25%;}
.col-4 {width: 33.33%;}
.col-5 {width: 41.66%;}
.col-6 {width: 50%;}
.col-7 {width: 58.33%;}
.col-8 {width: 66.66%;}
.col-9 {width: 75%;}
.col-10 {width: 83.33%;}
.col-11 {width: 91.66%;}
.col-12 {width: 100%;}
```

3. La propriété **[class*="col-"]** ("toutes les classes contenant "col-", revoir pour rappel les habillages de texte du premier exercice du TP2) permet alors de configurer toutes les propriétés communes à ces blocs. Dans notre cas, ce sera un positionnement flottant à gauche pour les avoir les unes à la suite des autres, et un peu de marge interne :

```
[class*="col-"] {
    float: left;
    padding: 15px;
}
```

4. Dans le fichier html nous pourrons alors créer une ligne (row) par exemple ainsi (la somme devant faire 12 colonnes et 100% de la largeur de la page) :

```
<div class="row">
  <div class="col-3">...</div> <!-- 25% de largeur de page -->
  <div class="col-9">...</div> <!-- 75% de largeur de page -->
</div>
```

5. Pour que chaque ligne se finissent bien par un retour à la ligne (arrêt du positionnement flottant) et l'espace soit bien occupé (display:block) voici les propriétés à mettre pour la classe row :

```
.row::after {
    content: "";
    clear: both;
    display: block;
}
```

6. Pour le site vous devez donc créer le fichier html **siteGrillePourcentages.html** contenant le code suivant qui a permis de créer ce site :

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="siteGrillePourcentages.css">
    <title> Mon Site en grille avec 12 colonnes </title>
</head>
<body>

<div class="header">
   <h1> Mon Site </h1>
</div>

<div class="row">
          <div class="col-3 menu">
            <ul>
              <li><a href="#">Page 1</a></li>
              <li><a href="#">Page 2</a></li>
              <li><a href="#">Page 3</a></li>
            </ul>
          </div>
          <div class="col-6">
             <h1>Titre contenu page 1</h1>
             <p>Contenu de la page 1.</p>
          </div>

          <div class="col-3 right">
            <div class="aside">
               <p> Blabla annexe </p>
            </div>
          </div>
</div>

<div class="footer">
   <p> Blabla bas de page </p>
</div>

</body>
</html>
```

7. Il ne vous reste plus qu'à compléter l'habillage du site (**siteGrillePourcentages.css**) avec les aides ci-dessous.

```
* {
   box-sizing: border-box;
}
.row::after {
   content: "";
   clear: both;
   display: block;
}
[class*="col-"] {
   float: left;
   padding: 15px;
}
.col-1 {width: 8.33%;}
.col-2 {width: 16.66%;}
.col-3 {width: 25%;}
.col-4 {width: 33.33%;}
.col-5 {width: 41.66%;}
.col-6 {width: 50%;}
.col-7 {width: 58.33%;}
.col-8 {width: 66.66%;}
.col-9 {width: 75%;}
.col-10 {width: 83.33%;}
.col-11 {width: 91.66%;}
.col-12 {width: 100%;}

html {
   /*Mettre la police Lucida Sans, sans-serif sinon.*/
}
.header {
   /*Couleur de fond #003466 */
   /*Texte en blanc */
   /*Marge interne 15 px*/
}
.menu ul {
   /* Ce qu'il faut pour un menu vertical (cf TP précédent)*/
   /* Marges à 0*/
}
.menu li a{
   /*Marge interne 8 px*/
   /*Marge externe en bas de 7 px*/
   /*Couleur de fond #28d0d0 */
   /*Texte en blanc */
   /* Ajouter ce qu'il faut pour un menu vertical (cf TP précédent)*/
}
```

```
.menu li a:hover {
   /*Couleur de fond #F23000 */
}

.aside {
   /*Couleur de fond #28d0d0 */
   /*Texte en blanc */
   /*Texte centré */
   /*Taille de police 12 px*/
   /*Marge interne 15 px*/
}

.footer {
   /*Couleur de fond #003466 */
            /*Texte en blanc */
   /*Texte centré */
   /*Taille de police 12 px*/
   /*Marge interne 15 px*/
}
```

3.4 CSS Media Queries

Pour mieux encore vous adapter aux différents écrans, vous pouvez vouloir modifier l'agencement des blocs, par exemple passer d'un menu horizontal sur grand écran à un menu vertical sur un petit écran, voire enlever du contenu. Ceci est possible grâce aux **media queries** CSS 3 avec la règle **@media**.

Les documentations w3school et MDN sur les media queries sont respectivement ici et là.

La règle **@media** permet d'exécuter un bloc de propriétés CSS (et donc définir un habillage) seulement si une certaine condition est vraie.

Voici la règle pour les tailles de fenêtre de navigateur **plus petites que 600px** (vous pourrez facilement généraliser pour n'importe quelle taille)

```
@media only screen and (max-width: 600px) {
   ...
   /* Règles CSS pour ce cas*/
   ...
}
```

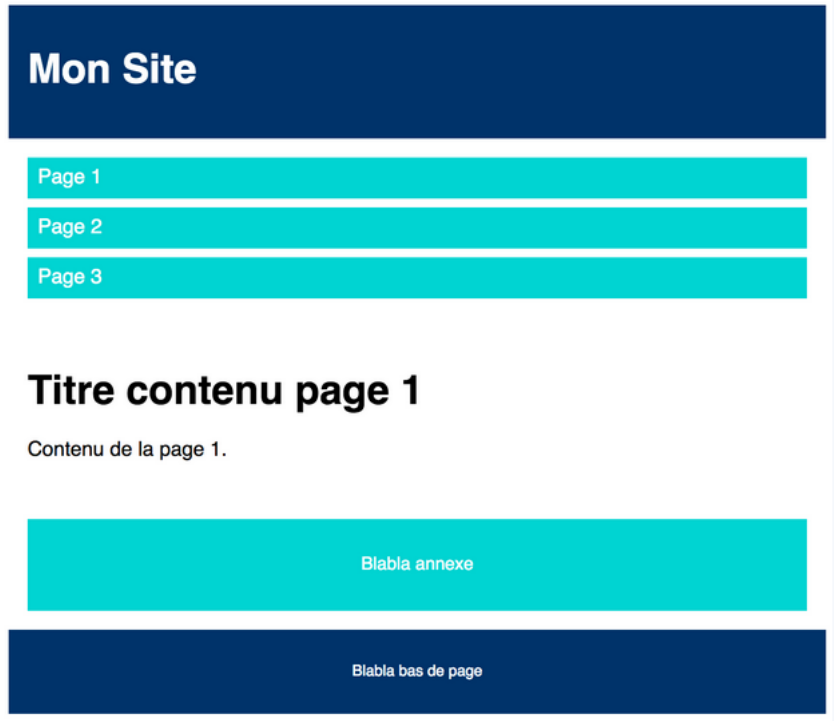
Voici la règle pour les tailles de fenêtre de navigateur **plus grandes que 600px** (vous pourrez facilement généraliser pour n'importe quelle taille)

```
@media only screen and (min-width: 600px) {...}
```

Il est conseillé (approche dîtes "mobile first") de démarrer le design pour un petit écran puis d'ajouter des media queries avec l'augmentation de la taille de l'écran. Par exemple, pour faire une adaptation aux petits écrans (exemple d'un téléphone), écrans moyens (exemple d'une tablette), et grand (exemple de l'écran d'un ordinateur) on fera :

```
/* Design écran plus petit que 600px; */
@media only screen and (min-width: 600px) {
   /* Modification de design pour les plus de 600px*/
}
@media only screen and (min-width: 768px) {
   /* Modification de design pour les plus de 768px*/
}
```

- Modifier le CSS du précédent site tel que si la taille de fenêtre est plus petite que 768px alors l'affichage soit celui illustré ci-dessous.



Dans ce cas très particulier, il suffit juste d'ajouter ceci (méthode non "mobile first")

```
@media only screen and (max-width: 768px) {
  /* For mobile phones: */
  [class*="col-"] {
    /*Largeur de toutes les colonnes est de 100%*/
  }
```

3.4.1 Menu horizontal Web responsive

Reprenez maintenant votre menu horizontal réalisé au TP précédent

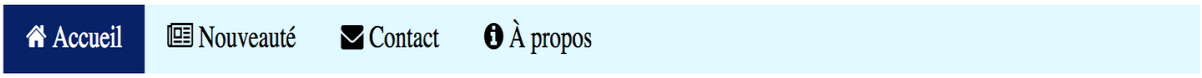


Illustration du menu horizontal avec des icônes quand la taille de fenêtre est plus grande que 500px.

- Rendez ce site Web responsive pour les écrans plus petits que 500px.

Illustration du menu horizontal avec des icônes quand la taille de fenêtre est plus petite que 500px.

N'oubliez pas d'ajouter à votre fichier HTML la balise meta viewport

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Pour le CSS

```
@media screen and (max-width: 500px) {
  <à compléter> {
    float: none;
  }
}
```

3.5 CSS Grid Layout

Vous avez vu en début de TP comment faire "à la main" une grille de 12 colonnes avec un positionnement flottant et des tailles en pourcentage.

Depuis CSS a évolué et a été créé le concept de grille CSS **CSS grid layout**.

Cette grille est facile à utiliser, il suffit de mettre la propriété **display: grid;** dans un conteneur parent en définissant le nombre de colonnes et de grilles ainsi que leurs dimensions avec par exemple **grid-template-columns: 100px 100px 100px;** (la grille est composée de 3 colonnes de 100px de large) et **grid-template-rows: 50px 50px;** (la grille est composée de 2 lignes de 50px de hauteur)

```
.conteneur{
    display: grid;
    grid-template-columns: 100px 100px 100px;
    grid-template-rows: 50px 50px;
}
```

Les enfants (bloc contenus dans le parent) occuperont alors des cases de cette grille.

3.5.1 CSS Grid Garden Tuto

Pour étudier le fonctionnement et l'apprendre ! ;) faites ce petit tuto de 28 niveaux très bien fait pour découvrir les propriétés de base: http://cssgridgarden.com/#fr.

3.5.2 CSS Grid Layout de l'exercice 3.3

Pour la suite de ce TP (et pour pouvoir faire l'exercice suivant), il faut étudier le guide sur Grid Layout fait par Alsacréations disponible sur le lien suivant **https://www.alsacreations.com/article/lire/1388-css3-grid-layout.html .** De plus un mémento CSS Grid Layout se trouve ici**.**

- Réalisez le même site (similaire) en utilisant le système de grille CSS3, que celui fait avec une grille "à la main" au début du TP (exercice 3.3).

Aide : pour votre Grid vous pouvez définir 3 colonnes 1fr 2fr 1fr et 3 lignes 2fr 6fr 1fr

Aide : vous pouvez ajouter de l'espace entre les cases avec ***grid-gap : 10px*** *(propriété à spécifier dans le bloc « conteneur » ou « wrapper » avec le* ***display : grid*** *et les* ***grid-template****).*

Aide : Pour développer ce site et votre futur site beaucoup d'exemples se trouvent ici

3.6 Quizz CSS Alsacréations

Complétez vos connaissances CSS avec les Quizz CSS Alsacréations