J.-Y. Tigli
Polytech'Nice – Sophia
SI5 IFI IAM ELIM
2017-2018

Tutorial : Web responsive design avec CSS3 et HTML5

1 Les bases du Web responsive design en CSS3

Le responsive design permet d'adapter le style de votre page selon l'écran qui le consulte. Cela vous permet de faire un site adapté aux smartphone/tablette/PC.
Aujourd'hui, l'utilisation du responsive design est simple et inévitable via l'utilisation des media queries.

Mise en œuvre :

Pour nos exemples, nous pourrons

- regrouper style et HTML dans une seul fichier ou
- créer 2 fichiers séparés
  - index.html : Où se trouvera le contenu HTML de la page Web
  - style.css : Où nous définirons le style à appliquer sur cette page

Dans ce dernier cas il ne faudra par oublier de rajouter un lien vers le style dans le fichier HTML :

```
<link rel="stylesheet" type="text/css" href="style.css">
```

Outils de Test :

Dans le cadre de ce tutorial nous utiliserons le simulateur du W3C : https://www.w3schools.com/css/tryit.asp?filename=tryresponsive_mediaquery.
Les plus avancés pourront tester les site Web sur le propre smart phone, sachant que pour la restitution de données des pages web sont souvent suffisantes pour peu qu'elles s'adaptent au terminal mobile.

1.1 Les bases du viewport

La plupart des sites optimisés pour les mobiles utilisent la balise suivante :

```
<meta name="viewport" content="width=device-width, user-scalable=no">
```

Elle permet de communiquer de définir la taille de l'écran du client.

La propriété `width` contrôle la taille du *viewport*. Elle peut être réglée à une valeur précise de pixels, comme `width=600`, ou bien à la valeur spéciale `device-width` qui correspond à la largeur de l'écran en pixels CSS à l'échelle 100%.

Il existe les propriétés et valeurs `height` et `device-height` correspondantes, qui peuvent être utiles pour les pages contenant des éléments qui changent de taille ou de position en fonction de la hauteur du *viewport*.

La propriété `initial-scale` contrôle le niveau de zoom lorsque la page est chargée pour la première fois. Les propriétés `maximum-scale`, `minimum-scale`, et `user-scalable` contrôlent la manière dont les utilisateurs sont autorisés à zoomer ou dézoomer une page.

1.2 CSS3 : les Media-Queries et leurs règles

Les `Media Queries` permettent donc de gérer :

- Le type de média
- La taille de l'écran
- La taille de la fenêtre
- La résolution
- Le nombre de couleurs
- L'orientation
- ...

La documentation de référence se trouve sur : https://www.w3.org/TR/css3-mediaqueries/ et un excellent aide-mémoire sur https://developer.mozilla.org/fr/docs/Web/CSS/Requ%C3%AAtes_m%C3%A9dia/Utiliser_les_Media_queries.

Il existe de nombreuses règles permettant de construire des media queries. Voici les principales :

- `color` : gestion de la couleur (en bits/pixel).
- `height` : hauteur de la zone d'affichage (fenêtre).
- `width` : largeur de la zone d'affichage (fenêtre).
- `device-height` : hauteur du périphérique.
- `device-width` : largeur du périphérique.
- `orientation` : orientation du périphérique (portrait ou paysage).
- `media` : type d'écran de sortie. Quelques-unes des valeurs possibles :
  - `screen` : écran « classique » ;
  - `handheld` : périphérique mobile ;
  - `print` : impression ;
  - `tv` : télévision ;
  - `projection` : projecteur ;
  - `all` : tous les types d'écran.

On peut rajouter le préfixe `min-` ou `max-` devant la plupart de ces règles. Ainsi, `min-width` signifie « Largeur minimale », `max-height` « Hauteur maximale », etc.
La différence entre `width` et `device-width` se perçoit surtout sur les navigateurs mobiles des smartphones.

Les règles peuvent être combinées à l'aide des mots suivants :

- `only` : « uniquement » ;
- `and` : « et » ;
- `not` : « non ».

Voici quelques exemples :

```
/* Sur les écrans, quand la largeur de la fenêtre fait au maximum 1280px */
@media screen and (max-width: 1280px)

/* Sur tous types d'écran, quand la largeur de la fenêtre est comprise entre 1024px et 1280px */
@media all and (min-width: 1024px) and (max-width: 1280px)

/* Sur les téléviseurs */
@media tv

/* Sur tous types d'écrans orientés verticalement */
@media all and (orientation: portrait)
```

Question 1: Pour chacune de ces règles décrivez l'effet qu'elle a sur le rendu de la page Web.

1.3 Exemple et Test

Voici un exemple simple du site :
http://www.w3schools.com/css/tryit.asp?filename=tryresponsive_mediaquery

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
body {
   background-color: lightgreen;
}
@media only screen and (max-width: 500px) {
   body {
      background-color: lightblue;
   }
}
</style>
</head>
<body>

<p>Resize the browser window. When the width of this document is less than 500 pixels, the background-color is "lightblue", otherwise it is "lightgreen".</p>
```

Tutorial : Web responsive design avec CSS3 et HTML5

```
</body>
</html>
```

Question 2: A quoi servent les règles de media queries dans cet example ?

Question 3: Rajouter un changement de couleur de fond orange pour une largeur de fenêtre à 300px.

2 Mise en pratique des média queries sur un exemple complet

Bon, changer la couleur du texte, c'est bien joli mais cela n'apporte pas grand-chose. Par contre, cela devient de suite plus intéressant quand on se sert des media queries pour modifier l'apparence de son site en fonction de la résolution. Vous allez voir qu'on peut faire tout ce qu'on veut !

Pour cet exemple, je vous propose de reprendre le design que nous avons créé pour le site web de Zozor (figure suivante).

Le site est bien adapté à la plupart des résolutions d'écran mais, quand l'écran est plus petit que 1024 px, il devient nécessaire de « scroller » vers la droite pour voir toute la page. Le site n'est donc pas très pratique à consulter sur un petit écran.

Question 4: utilisez les media queries pour changer l'apparence du site sur les résolutions inférieures à 1024 px de largeur en opérant les modifications suivantes :

- le menu de navigation en haut à droite sera disposé en hauteur plutôt qu'en largeur, et les liens seront écrits en plus petit ;
- la bannière avec le pont de San Francisco (le *Golden Gate*) sera supprimée, car elle prend beaucoup de place et n'apporte pas beaucoup d'informations ;
- le bloc `<aside>` « À propos de l'auteur » sera placé sous l'article (et non pas à côté), et son contenu sera réorganisé (la photo de Zozor sera positionnée en flottant).

Nous allons travailler directement à l'intérieur du fichier `style.css`.

Vous trouverez les fichiers Web du site « non responsive » sur
https://www.dropbox.com/s/atbto6muvgehcuq/Site%20Web%20Type.zip?dl=0

2.1.1.1 La page

Pour le moment, la largeur de la page est fixée à 900 px et le contenu est centré :

```
#bloc_page
{
   width: 900px;
   margin: auto;
}
```

À la suite de ces lignes, je vous propose d'ajouter la règle media query suivante :

```
@media all and (max-width: 1024px)
{
   #bloc_page
   {
      width: auto;
   }
}
```

La règle signifie : « Pour tous les types d'écrans, si la largeur de la fenêtre ne dépasse pas 1024 px, alors exécuter les règles CSS suivantes ».

Les règles CSS en question sont très simples, il n'y en a en fait qu'une seule : on donne une largeur automatique à la page (plutôt qu'une largeur fixe de 900 px). La page prendra alors tout l'espace disponible dans la fenêtre. Cela évite l'apparition de barres de défilement horizontales sur les petites résolutions.

`auto` est la valeur par défaut de la propriété `width`. Par défaut, les blocs ont une largeur automatique (ils prennent toute la place disponible). Cette valeur « écrase » celle que nous avions forcée à 900px quelques lignes plus haut : nous revenons donc au comportement par défaut du bloc.

2.1.1.2 Le menu de navigation

Nous voulons que le menu de navigation prenne moins de place sur les petites résolutions. Plutôt que de lui donner une dimension fixe, nous allons lui redonner sa dimension automatique flexible d'origine. Chaque élément du menu s'écrira en dessous du précédent : pour cela, nous demandons à ce que les éléments de la Flexbox soit organisés en colonne.

Enfin, le texte sera écrit plus petit et nous retirons la bordure en bas des liens lors du survol, car elle est moins adaptée à cette disposition.

```
@media all and (max-width: 1024px)
{
   nav
   {
      width: auto;
      text-align: left;
   }

   nav ul
   {
      flex-direction: column;
   }

   nav li
   {
      padding-left: 4px;
   }

   nav a
   {
      font-size: 1.1em;
   }
```

```
    nav a:hover
    {
        border-bottom: 0;
    }
}
```

2.1.1.3 La bannière

Pour retirer la bannière, rien de plus simple : nous utilisons la propriété `display` à laquelle nous affectons la valeur `none`. Si la fenêtre est trop petite, nous préférons masquer complètement la bannière :

```
@media all and (max-width: 1024px)
{
    #banniere_image
    {
        display: none;
    }
}
```

2.1.1.4 Le bloc « À propos de l'auteur »

Plutôt que de placer ce bloc à droite de l'article, nous allons le faire passer en-dessous grâce à des Flexbox en colonne. Ce type de disposition « de haut en bas » est plus adapté aux petits écrans.

À l'intérieur du bloc, nous réajustons un peu la position des éléments : la photo de Zozor, notamment, sera placée en flottant à droite.

```
@media all and (max-width: 1024px)
{
    section
    {
        flex-direction: column;
    }

    article, aside
    {
        width: auto;
        margin-bottom: 15px;
    }
```

```
    #fleche_bulle
    {
        display: none;
    }

    #photo_zozor img
    {
        width: 110px;
        float: right;
        margin-left: 15px;
    }

    aside p:last-child
    {
        text-align: center;
    }
}
```

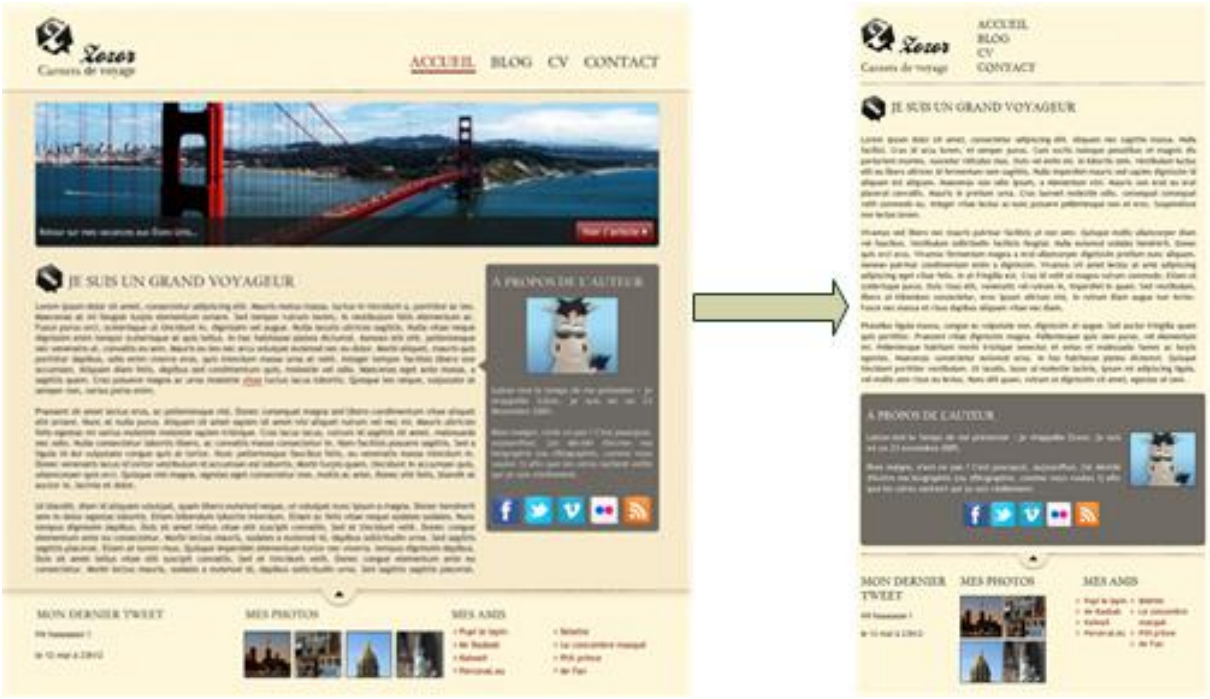
Que signifie `aside p:last-child` ?

C'est un sélecteur avancé que nous n'avons pas utilisé jusqu'ici. `aside p` signifie « Tous les paragraphes à l'intérieur de la balise `<aside>` ». Avec `:last-child`, on cible uniquement le dernier paragraphe dans le bloc `aside` (celui qui contient les liens vers Facebook et Twitter), pour pouvoir centrer les images. Bien entendu, on aurait aussi pu affecter une `class` ou un `id` à ce paragraphe pour le cibler directement, mais je n'ai pas voulu modifier le code HTML.

2.1.1.5 Le résultat

La page est désormais complètement réorganisée lorsque la fenêtre fait 1024 px ou moins de largeur.

Tutorial : Web responsive design avec CSS3 et HTML5



Le même site est présenté différemment en fonction de la largeur de l'écran