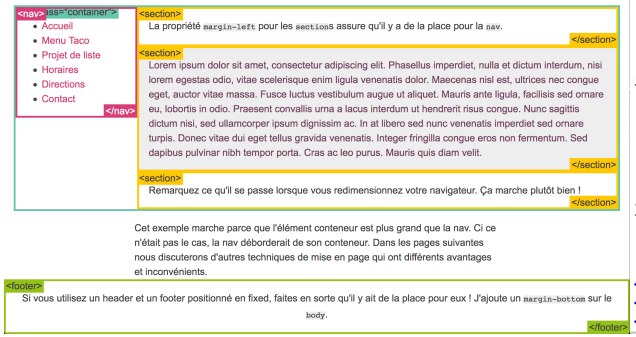
Technologies du Web
CSS, les feuilles de style
Notion 3

Florent Carlier, Valérie Renault

prenom.nom@univ-lemans.fr
Université du Maine

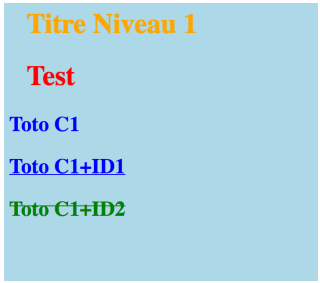
Positionnement des blocs : exemple - Approfondissement

```css
.container {
  position: relative;
}
nav {
  position: absolute;
  left: 0px;
  width: 200px;
}
section {
  /* position is static by default */
  margin-left: 200px;
}
footer {
  position: fixed;
  bottom: 0;
  left: 0;
  height: 70px;
  background-color: white;
  width: 100%;
}
body {
  margin-bottom: 120px;
}
```



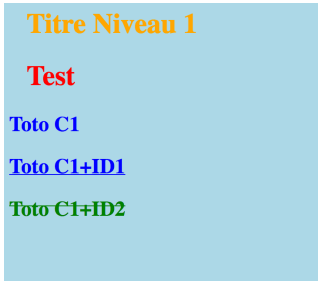

```html
<html>
<head>
<style>
  .container {
    position: relative;
  }
  nav.elem {
    position: absolute;
    left: 0px;
    width: 200px;
  }
  section {
    /* position is static by default */
    margin-left: 200px;
  }
  footer.elem {
    position: fixed;
    bottom: 0;
    left: 0;
    height: 90px;
    background-color: white;
    width: 100%;
    margin: 0;
    z-index: 1;
  }
  body {
    margin-bottom: 120px;
  }
</style>
</head>
<body>
<div class="container elem">
  <span class="label">&lt;div class="container"></span>

  <nav class="elem elem-red">
    <span class="label">&lt;nav></span>
    <ul>
      <li><a href="position-example.html">Accueil</a></li>
      <li><a href="position-example.html">Contact</a></li>
    </ul>
    <span class="endlabel">&lt;/nav></span>
  </nav>

  <section class="elem elem-green">
    <span class="label">&lt;section></span>
    <p>La propriété <code>margin-left</code> pour les <code>section</code>s assure qu'il y a de la
place pour la <code>nav</code>.</p>
    <span class="endlabel">&lt;/section></span>
  </section>

  <section class="elem elem-green ipsum">
    <span class="label">&lt;section></span>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum
interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. </p>
    <span class="endlabel">&lt;/section></span>
  </section>

  <footer class="elem elem-orange">
    <span class="label">&lt;footer></span>
    <p>Si vous utilisez un header et un footer positionné en fixed, faites en sorte qu'il y ait de
la place pour eux ! J'ajoute un <code>margin-bottom</code> sur le <code>body</code>.</p>
    <span class="endlabel">&lt;/footer></span>
  </footer>

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

Compléments Cascading - Exemples




```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
<style>
h1 {
    color: orange;
}
div h1 {
    color: red;
}
.C1 {
    color: blue;
}
#ID1 {
    text-decoration : underline;
}
#ID2 {
    text-decoration : overline;
    color: green;
}
</style>
</head>
```

Compléments Cascading - Exemples




```
<body>
<h1>Titre Niveau 1</h1>
<div>
  <h1>Test</h1>
</div>
<div class="C1">
 <h2>Toto C1</h2>
</div>
<div class="C1" id="ID1">
  <h2>Toto C1+ID1</h2>
</div>
<div class="C1" id="ID2">
  <h2>Toto C1+ID2</h2>
</div>
</body>
</html>
```

CSS Layout - La propriété d'affichage (1/2)

La propriété display est la propriété CSS la plus importante pour le contrôle de la mise en page. La propriété display spécifie si / comment un élément est affiché. Chaque élément HTML a une valeur d'affichage par défaut en fonction du type d'élément.

Eléments au niveau du bloc

Un élément de niveau bloc commence toujours sur une nouvelle ligne et occupe toute la largeur disponible. Exemples d'éléments de niveau bloc : <h1> - <h6> <p> <form> <header> <footer> <section>

Éléments en ligne

Un élément en ligne ne démarre pas sur une nouvelle ligne et ne prend que la largeur nécessaire.
Exemples d'éléments en ligne : <a> <img>

CSS Layout - La propriété d'affichage (2/2)

```
li {
    display: inline;
}
a {
    display: block;
}
```

Vous pouvez masquer un élément en définissant la propriété display sur none.
display : none ; est couramment utilisé avec JavaScript pour masquer et afficher les éléments sans les supprimer et les recréer.
L'élément sera masqué et la page sera affichée comme si l'élément n'était pas là :

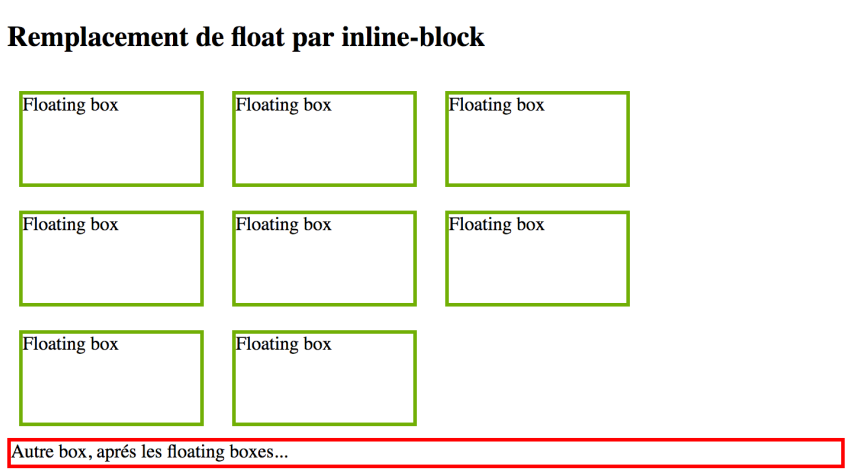
```
h1.hidden {
    display: none;
}
```

visibility :hidden ; cache également un élément.
Cependant, l'élément occupera toujours le même espace qu'avant. L'élément sera caché, mais affecte toujours la mise en page :

```
h1.hidden {
    visibility: hidden;
}
```

Valeur float / inline-block

```
<!DOCTYPE html>
<html>
<head>
<style>
.floating-box {
    display: inline-block; /* float: left; */
    width: 150px;
    height: 75px;
    margin: 10px;
    border: 3px solid #73AD21;
}

.after-box {
    /* clear: left; */
    border: 3px solid red;
}
</style>
</head>
<body>

<h2>Remplacement de float par inline-block</h2>

<!-- Float Box -->
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>

<div class="after-box">Autre box, aprés les floating boxes...</div>

</body>
</html>
```



Formats Audio et Vidéo

Format ?

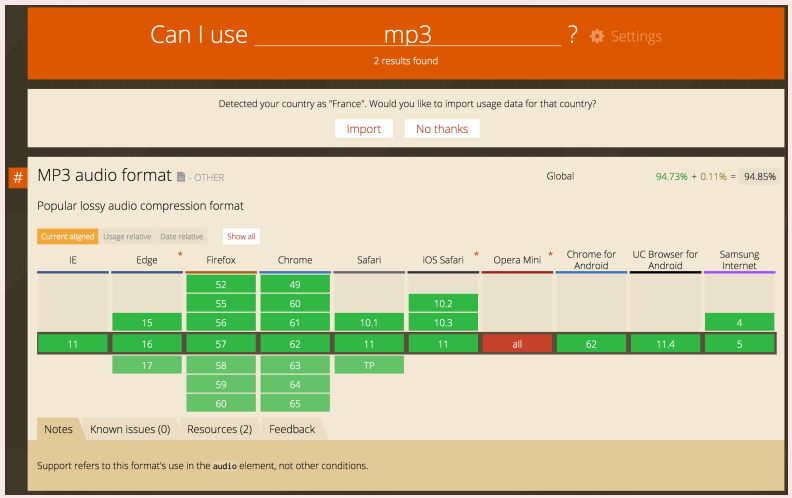
Images et Balise <img /> : différents formats d'images (JPEG, PNG, GIF, etc.).
Pour la vidéo et l'audio, Attention particulière sur les formats reconnus.

Formats Audio

Pour diffuser de la musique / son

- **MP3 : L'un des plus compatibles (tous les appareils savent lire des MP3).**
- **AAC : Utilisé majoritairement par Apple sur iTunes.**
- **OGG : Format Ogg Vorbis (logiciel libre / Linux). Aucun brevet.**
- **WAV (format non compressé) : Très volumineux.**

Compatibilité : Caniuse.com



Formats Vidéo

3 éléments :

- Un format conteneur : Extension du fichier : AVI, MP4, MKV...
- Un codec audio : c'est le format du son de la vidéo, généralement compressé.
- Un codec vidéo : c'est le format qui va compresser les images.

Formats :

- H.264 : l'un des plus puissants et des plus utilisés aujourd'hui.
- Ogg Theora : un codec gratuit et libre de droits, mais moins puissant que H.264.
- WebM : un autre codec gratuit et libre de droits, plus récent. Proposé par Google, c'est le concurrent le plus sérieux de H.264.

Compatibilité : Caniuse.com

Insertion d'un élément audio

En théorie, il suffit d'une simple balise pour jouer un son sur notre page :

```
<audio src="musique.mp3"></audio>
```

En pratique, le navigateur va seulement télécharger les informations générales sur le fichier (Métadonnées) mais il ne se passera rien de particulier.

Attributs de la balise <audio>

- **controls : pour ajouter les boutons « Lecture », « Pause » et la barre de défilement.**
- **width : pour modifier la largeur de l'outil de lecture audio.**
- **loop : la musique sera jouée en boucle.**
- **autoplay : la musique sera jouée dès le chargement de la page.**
- **preload : indique si la musique peut être préchargée dès le chargement de la page ou non. Cet attribut peut prendre les valeurs :**
  - **auto(par défaut) : le navigateur décide s'il doit précharger toute la musique, uniquement les métadonnées ou rien du tout.**
  - **metadata : charge uniquement les métadonnées (durée, etc.).**
  - **none : aucun pré-chargement**

On ne peut pas forcer le pré-chargement de la musique, c'est toujours le navigateur qui décide.

syntaxe <audio>

```
<audio src="hype_home.mp3" controls></audio>
```

Pourquoi ouvrir la balise pour la refermer immédiatement après ? Afficher un message ou Proposer une solution de secours pour les navigateurs qui ne gèrent pas cette nouvelle balise.

```
<audio src="hype_home.mp3" controls>Veuillez mettre à jour votre navigateur !</audio>
```

Et si le navigateur ne gère pas le MP3, comment faire ?
Il faut proposer plusieurs versions du fichier audio. Dans ce cas, on va construire notre balise comme ceci :

```
<audio controls>
  <source src="hype_home.mp3">
  <source src="hype_home.ogg">
</audio>
```

Le navigateur prendra automatiquement le format qu'il reconnaît.

Insertion d'une vidéo

Balise <video>

pour insérer une vidéo dans la page :

```
<video src="sintel.webm"></video>
```

Aucun contrôle ne permet de lancer la vidéo !

Attributs de la balise <video>

- **poster : image à afficher à la place de la vidéo tant que celle-ci n'est pas lancée. Par défaut, le navigateur prend la première image de la vidéo.**
- **controls : pour ajouter les boutons « Lecture », « Pause » et la barre de défilement.**
- **width : pour modifier la largeur de la vidéo.**
- **height : pour modifier la hauteur de la vidéo.**
- **loop : la vidéo sera jouée en boucle.**
- **autoplay : la vidéo sera jouée dès le chargement de la page.**
- **preload : indique si la vidéo peut être préchargée dès le chargement de la page ou non. Cet attribut peut prendre les valeurs :**
  - **auto(par défaut) : le navigateur décide s'il doit précharger toute la vidéo, uniquement les métadonnées ou rien du tout.**
  - **metadata : charge uniquement les métadonnées (durée, dimensions, etc.).**
  - **none : pas de préchargement.**

On ne peut pas forcer le préchargement de la vidéo, c'est toujours le navigateur qui décide.

Syntaxe <video>

Les proportions de la vidéo sont toujours conservées.

```
<video src="sintel.webm" controls poster="sintel.jpg" width="600"></video>
```

Pourquoi ouvrir et refermer immédiatement après la balise ?

```
<video src="sintel.webm" controls poster="sintel.jpg" width="600">
    Il est temps de mettre à jour votre navigateur !
</video>
```

Reconnaître différents formats vidéo ?

```
<video controls poster="sintel.jpg" width="600">
    <source src="sintel.mp4">
    <source src="sintel.webm">
    <source src="sintel.ogv">
</video>
```

Framework CSS (1/2)

Framework ?

Utiliser un framework CSS est une bonne idée uniquement si il fait vraiment ce dont votre site à besoin.

http ://www.blueprintcss.org/

http ://bluetrip.org/

http ://twitter.github.com/bootstrap/

http ://foundation.zurb.com/

http ://groundworkcss.github.com/

Framework CSS (2/2)

http ://imperavi.com/kube/

http ://semantic-ui.com/

http ://susy.oddbird.net/

http ://purecss.io/

http ://www.unsemantic.com/

CSS3 Introduction

CSS3 est le dernier standard du CSS.
Il est compatible avec les versions précédentes.
Les "modules" CSS3 les plus importants sont :

- Selectors
- Box Model
- Backgrounds and Borders
- Image Values and Replaced Content
- Text Effects
- 2D/3D Transformations
- Animations
- Multiple Column Layout

La plupart des nouvelles propriétés CSS3 sont implémentés dans les derniers navigateurs.

CSS et Mise en page

Block flottants pour faire des mises en page.
La mise en page de sites web moderne se fait avec deux nouvelles techniques :

- flexbox : pour des mises en page fluides.
- grid layout : plus récent et moins bien supporté, créer des sites sur des principes de grilles.

Mise en page avancée : Flexbox

Pour créer des mises en page fluides :

```
https://css-tricks.com/snippets/css/a-guide-to-flexbox/
https://openclassrooms.com/courses/apprenez-a-creer-votre-site-web-
                              avec-html5-et-css3/la-mise-en-page-avec-flexbox
https://developer.mozilla.org/fr/docs/Web/CSS/
                              Disposition_des_bo%C3%AEtes_flexibles_CSS/
                              Utilisation_des_flexbox_en_CSS
https://www.alsacreations.com/tuto/lire/1493-css3-flexbox-layout-module.html
```

Démonstration Flexbox

```
https://www.w3schools.com/css/css3_flexbox.asp
```

Mise en page avancée : Grid Layout

Grid layout pour créer une grille CSS (attention au support)

```
https://la-cascade.io/css-grid-layout-guide-complet/
https://developer.mozilla.org/fr/docs/Web/CSS/CSS_Grid_Layout
https://css-tricks.com/snippets/css/complete-guide-grid/
```

Responsive Web Design - Grid-View

Propriétés

https://www.w3schools.com/css/css_rwd_grid.asp

Démonstration Grid Layout

https://www.w3schools.com/css/tryit.asp?filename=tryresponsive_col-s

CSS3 Media Queries

CSS3 introduit des requêtes média et étendent l'idée des types de médias CSS2 : au lieu de chercher un type d'appareil, ils regardent les capacités de l'appareil.

Les requêtes média peuvent être utilisées :

- largeur et hauteur de la fenêtre
- largeur et hauteur de l'appareil
- orientation (la tablette / téléphone est-elle en mode paysage ou portrait ?)
- résolution

L'utilisation de requêtes multimédias est une technique couramment utilisée pour fournir une feuille de style personnalisée aux tablettes, à l'iPhone et aux Androïdes.

Syntaxe de requête de média

Une requête multimédia se compose d'un type de média et peut contenir une ou plusieurs expressions, qui se résolvent en vrai ou faux.

```
@media not|only mediatype and ( expressions ) {
    CSS-Code;
}
```

Lorsqu'une requête de média est vraie, la feuille de style correspondante ou les règles de style sont appliquées, en suivant les règles de cascade normales.
Le type de support est facultatif et le type 'all' sera implicite.

Vous pouvez également avoir différentes feuilles de style pour différents médias :

```
<link rel="stylesheet" media=" mediatype and|not|only ( expressions )" href=" print.css ">
```

CSS3 Types Media

Description des valeurs :

- all : utiliser pour tous les types d'appareil média,
- print : utiliser pour les imprimantes,
- screen : utiliser pour les écrans de PC, tablettes, smartphone,
- speech : utiliser par les lecteurs vocaux.

Media Queries - Exemple 1 :

media + Taille visualisation

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
    background-color: pink;
}

@media screen and (min-width: 480px) {
    body {
        background-color: lightgreen;
    }
}
</style>
</head>
<body>

<h1>Retailler la page pour voir l'effet</h1>
<p>Le media queries ne s'applique uniquement si le type est un 'screen' et
la page de visualisation est supérieur à 480px.</p>

</body>
</html>
```

Media Queries - Exemple 2 :

Menu qui flottera à gauche de la page si la fenêtre a une largeur de 480 pixels ou plus (si la fenêtre est inférieure à 480 pixels, le menu sera au-dessus du contenu) :

```
<html>
<head><meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
.wrapper {overflow: auto;}
#main {margin-left: 4px;}
#leftsidebar { float: none; width: auto; }
#menulist { margin: 0; padding: 0; }
.menuitem {
    background: #CDF0F6; border: 1px solid #d4d4d4; border-radius: 4px;
    list-style-type: none; margin: 4px; padding: 2px;
}

@media screen and (min-width: 480px) {
    #leftsidebar {width: 200px; float: left;}
    #main {margin-left: 216px;}
}
</style>
</head>
<body>
<div class="wrapper">
  <div id="leftsidebar">
    <ul id="menulist">
      <li class="menuitem">Menu-item 1</li><li class="menuitem">Menu-item 2</li>
      <li class="menuitem">Menu-item 3</li>
   </ul>
  </div>
  <div id="main"><h1>Retailler la page pour voir l'effet</h1></div></div>
</body>
</html>
```

Web Responsive Design - Introduction

Qu'est-ce que Responsive Web Design ?

Le design web réactif rend votre page Web belle sur tous les appareils.
Le design web responsive n'utilise que du HTML et du CSS.
Le design web réactif n'est pas un programme ou un JavaScript.

Concevoir pour la meilleure expérience pour tous les utilisateurs

Les pages Web peuvent être consultées à l'aide de nombreux périphériques différents : ordinateurs de bureau, tablettes et téléphones. Votre page Web devrait être belle et facile à utiliser, quel que soit l'appareil.

Web Responsive Design - Introduction

Les pages Web ne doivent pas laisser de l'information pour s'adapter à des appareils plus petits, mais plutôt adapter son contenu à n'importe quel appareil :

La conception web réactive est l'utilisation de CSS et HTML pour redimensionner, masquer, rétrécir, agrandir ou déplacer le contenu pour le rendre bien sur n'importe quel écran.

Qu'est-ce que Viewport ?

La fenêtre est la zone visible de l'utilisateur d'une page Web.
La fenêtre d'affichage varie en fonction de l'appareil et sera plus petite sur un téléphone mobile que sur un écran d'ordinateur.
Avant les tablettes et les téléphones mobiles, les pages Web étaient conçues uniquement pour les écrans d'ordinateur, et il était courant que les pages Web soient de conception statique et de taille fixe.
Avec le surf sur Internet en utilisant des tablettes et des téléphones mobiles, les pages Web de taille fixe étaient trop grandes pour s'adapter à la fenêtre. Pour résoudre ce problème, les navigateurs de ces appareils ont réduit l'ensemble de la page Web pour l'adapter à l'écran.

Syntaxe Viewport

HTML5 a introduit une méthode permettant aux concepteurs Web de prendre le contrôle de la fenêtre d'affichage via la <meta>.
Vous devez inclure l'élément <meta> viewport suivant dans toutes vos pages Web :

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

Un élément <meta> viewport donne au navigateur des instructions sur la façon de contrôler les dimensions et la mise à l'échelle de la page.
La partie width=device-width définit la largeur de la page en fonction de la largeur d'écran de l'appareill.

La partie initial-scale=1.0 définit le niveau de zoom initial lorsque la page est chargée.