Support de cours CSS3

Jérôme AMBROISE

Next Formation
Semaine 27/02/2017 - 01/03/2017

Sommaire (1/2)

Sommaire (2/2)

1.

Introduction

Présentation

Historique

Intégration

Commentaires

Introduction
Présentation

- ➔ CSS, pour “Cascading Style Sheets”
- ➔ Langage de mise en forme du HTML

Introduction
Historique

- CSS1 : 1996
- CSS2 : 1998 - 2003
- CSS3 : 2005

Introduction
Présentation

Le CSS s’appuie sur le DOM (Document Object Model) de l’HTML pour attribuer des propriétés à des balises.

Introduction
Intégration

➔ Comment intégrer du CSS au HTML ?

- Style inline
- Style en-tête
- Feuille de style externe

Introduction
Intégration - Style inline

Pour intégrer du CSS au sein d'une page HTML, il est possible de définir un attribut "style" sur une balise et de modifier ainsi sa mise en forme.
Exemple :

 Bienvenue

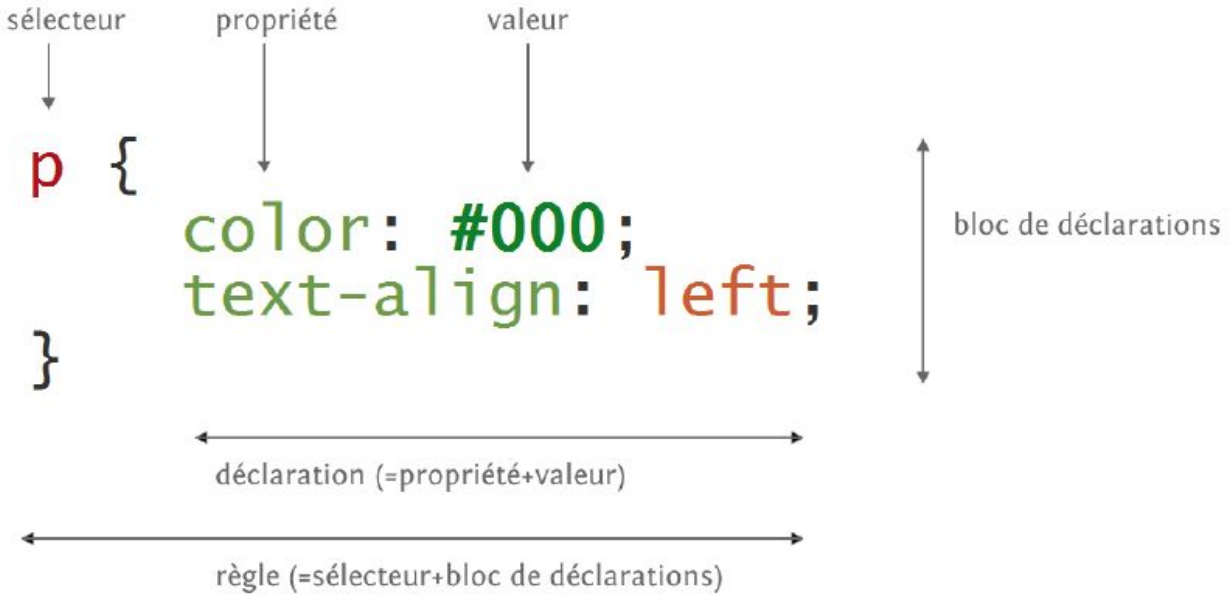
Introduction
Intégration - Style en-tête

La deuxième façon d'intégrer du CSS est de le définir dans une balise <style> située dans la balise <head> du fichier HTML.

Exemple :

```
…
<head>
        <style>
                p { color:blue; }
        </style>
</head>
<body>
        <p style="color:blue;"> Bienvenue </p>
</body>
…
```

Introduction
Intégration - Définitions de propriétés



Introduction
Intégration - Feuille de style externe

Cette méthode est la méthode conseillée à utiliser en priorité.

- ➔ Elle permet de créer une feuille de style externe, c'est en fait un fichier d'extension ".css".
- ➔ Nous pouvons alors définir des règles de mises en formes réutilisables dans plusieurs fichiers HTML.
- ➔ De plus cela permet de séparer la couche de structure (HTML) et la couche de mise en forme (CSS) : clarté du code, séparation des métiers.

Introduction
Intégration - Feuille de style externe

Exemple :

Côté HTML :

```
<head>
        <link rel="stylesheet" href="style.css"> <!-- Chemin relatif par rapport à la page HTML -->
</head>
<body>
        <p> Bonjour ! </p>
</body>
```

Côté CSS :

```
p
{
        color : yellow;
        background-color: red;
}
```

Introduction
Intégration - Priorité des styles

Il arrive qu'il explique plusieurs définitions d'une propriété pour le même élément HTML, dans ce cas quelle méthode CSS a la priorité?

Priorité 1 :

Le style inline (par exemple pour les images)

Priorité 2 :

Le style en-tête et style externe : la dernière définition proposée est celle qui sera prise en compte (se rappeler que c'est un programme qui parse le code de haut en bas)

➔ Dans le <head>, c'est donc la dernière déclaration d'un style qui sera priorisée.

Introduction
Intégration - Priorité des styles

Dans le cas où vous voulez prioriser un style externe peu importe l'ordre d'appel et être davantage prioritaire qu'un style externe, il existe une exception :

➔ Utiliser "!important" juste après la valeur de votre propriété.

Exemple :

```
p
{
    color: chartreuse !important;
}
```

Introduction
Intégration - Type de média

Par défaut l'appel externe d'une feuille de style s'applique à tous les médias (all), il est possible de définir un style pour un média particulier :

Media	Description
all	Utilisé pour tous les périphériques de type média
screen	Utilisé pour les écrans d'ordinateur
print	Utilisé pour les imprimantes
tv	Utilisé pour les appareils de type télévision
projection	Utilisé pour les présentations projetées, comme les diapositives

Introduction
Intégration - Type de média

Il existe aussi des types de média d'accessibilité :

Type	Description
aural	Utilisé pour synthétiseurs vocaux et sonores
braille	Utilisé pour les dispositifs tactile de braille
embossed	Utilisé pour les imprimantes en braille paginées
tty	Utilisé pour les supports utilisant une grille de caractères à pas fixe, comme les télétypes et les terminaux (console)

Introduction
Les commentaires

Les commentaires en CSS sont des instructions facultatives commençant par les caractères /* et se terminant par */

Leur usage est principalement informationnel :

- ➔ Justifier tel ou tel choix de déclaration CSS
- ➔ Délimiter les différentes parties de la feuille de styles.

Introduction
Les commentaires

Exemple :

```
/* Définitions structurelles */
header { ... }
main {  ... }

/* Définitions de la section 1 */
section#presentation { ...  }

/* Définitions de la section 2 */
section#commentaires {  ... }
```

2.

Les sélecteurs

Cibler une balise
Notion de classe CSS
Notion d'id CSS
Les balises non-sémantiques
Les pseudo-classes
Les pseudo-éléments
Sélection multiple
Sélection avec attributs

Les sélecteurs
Cibler une balise

La première façon d'appliquer d'appliquer des propriétés CSS est de cibler le nom d'une balise :

Exemple :

```
header { background-color : blue;}
main { background-color: red; }
p { color : green; }
a { color: black; }
```

Les sélecteurs
Notion de classe CSS

Les classes CSS permettent de définir plusieurs propriétés de style réutilisables.

- ➔ Définition de la classe dans le CSS (attribution d'un nom arbitraire)
- ➔ Application de ces propriétés sur des éléments HTML avec la même classe

On peut donc définir des classes qui ne vont pas s'appliquer à toutes les balises comme nous l'avions fait avant.

Les sélecteurs
Notion de classe CSS

Côté CSS :

- ➔ La classe est préfixée d'un point " . "
- ➔ Le nom de classe est arbitraire mais doit commencer par une lettre

Côté HTML :

- ➔ Ajout d'un attribut " class "
- ➔ Valeur de l'attribute " class " : le nom de votre class (sans le point)

Les sélecteurs
Notion de classe CSS

Exemple :

Côté CSS :

```
.error { color: red; }
.warning { color: orange; }
```

Côté HTML :

```
<p class="error"> Ceci est une erreur </p>
<p class="warning"> Ceci est un avertissement</p>
<p class="error"> Ceci est une autre erreur </p>
<p> Ceci est un paragraphe sans classe </p>
```

Les sélecteurs
Notion d' id CSS

L'id CSS est très proche de la classe (attribution + utilisation). La seule différence est qu'un id ne peut être présent seulement une fois dans la page.

- ➔ Permet de créer une ancre HTML pour les liens inter-page
  - ◆ Exemple d'une cible de lien avec ancre (index.html#portfolio)
  - ◆ Lors du clic sur le lien la page HTML s'affiche directement sur l'id défini
- ➔ Récupération de l'élément avec JavaScript (getElementById())

Les sélecteurs
Notion d' id CSS

Côté CSS :

- ➔ La classe est préfixée d'un dièse " # "
- ➔ Le nom de l'id est arbitraire mais doit commencer par une lettre

Côté HTML :

- ➔ Ajout d'un attribut " id"
- ➔ Valeur de l'attribute " id" : le nom de votre id (sans le dièse)

Les sélecteurs
Notion d' id CSS

Exemple :

Côté CSS :

```
#presentation { background-color: grey; }
#commentaires { background-color: silver; }
```

Côté HTML :

```
<section id="presentation"> … </section>
<section id="commentaires"> … </section>
<section> … </section>
```

Les sélecteurs
Les balises non-sémantiques

Cas particulier : Je possède dans mon HTML un paragraphe, et je souhaite seulement modifier une partie .

Côté HTML :

 Je souhaite attribuer un style à cette partie : il fait chaud

Du fait que le CSS s'appliquer sur une balise, il faut que j'en ajoute une :

➔ Utilisation d'une balise sémantique à privilégier :

 Je souhaite attribuer un style à cette partie : il fait chaud

➔ Si aucune balise sémantique n'a de sens dans le contexte, je peux utiliser la balise "span" (pas de sémantique, utilisée seulement pour le style)

 Je souhaite attribuer un style à cette partie : il fait chaud

Les sélecteurs
Les balises non-sémantiques

Je peux attribuer une classe et/ou un id à cette balise pour appliquer une style :

Côté HTML :

 Je souhaite attribuer un style à cette partie : il fait chaud

Côté CSS :

#chaleur { background-color: red; }

Les sélecteurs
Les balises non-sémantiques

La balise “span” est de type inline, elle est affichée par défaut à l’intérieur d’une ligne:

Son équivalent de type block existe : c’est la balise “div” bien connue. Si vous voulez attribuer un style particulier à des blocs entiers, utilisez cette balise.

Remarque :
Toujours vérifier si une balise sémantique n’existe pas avant d’utiliser “div” et qui est compatible avec le contexte. Les balises “div” ne sont pas mauvaises en soit, le but est de limiter au maximum leur utilisation.

Les sélecteurs

Sélecteurs : balise + non-sémantique

Les sélecteurs de balise et de classe/id sont cumulables. Je peux cibler une certaine balise ayant une certaine classe ou un certain id pour gagner en précision :

Exemple :

section#presention
section.commentaire
div.error
span.info

Prenons le dernier exemple : “span.warning”. Si j’applique une classe “warning” sur une balise autre (Texte), le style ne s’appliquera pas.

Les sélecteurs

Sélecteurs : balise + non-sémantique

Exemple :

Côté CSS :

```
span.info {color:chocolate; }
```

Côté HTML :

```
<span class="info">Texte</span>
<div class="info">Texte</div>
<div>Texte</div>
```

La propriété s'applique seulement sur la balise "span"

Les sélecteurs
Les pseudo-classes

- ➔ Ce sont comme des classes de style, mais un élément acquiert une pseudo-classe par action de l'utilisateur par exemple.
- ➔ Dans la feuille de style, un nom de pseudo-classe est précédé par deux points.
- ➔ Les pseudo-classes s’appliquent sur un sélecteur
  - ◆ Selecteur + “ : ” + Pseudo-classe

Les sélecteurs

Les pseudo-classes

Les pseudos-classes sont notamment utilisées avec les liens :

Pseudo classe	Description
:link	Cible un élément qui est la source d'un lien non encore visité.
:hover	Utilisé pour les imprimantes
:focus	Lorsque le visiteur passe la souris sur le lien
:active	Lorsque le lien a un focus actif
:visited	Cible un élément qui est la source d'un lien ayant été visité.

Les sélecteurs

Les pseudo-classes

Exemple :

Côté HTML :

```
<p>
        Voici un lien : <a href="test.html">Lien vers un test</a>
</p>
```

Côté CSS :

```
a:link { color:yellow; }
a:visited { color:chocolate; }
a:hover { color:orange; }
a:focus { color:red; }
a:active { color:green; }
```

Les sélecteurs

Les pseudo-classes

D'autres pseudos classes :

Pseudo classe	Description
p:first-child	Sélectionne tous les éléments <p> qui sont le premier enfant de son parent
p:first-of-type	Sélectionne chaque <p> élément qui est le premier <p> élément de son parent
p:last-child	Sélectionne tous les éléments <p> qui sont le dernier enfant de son parent
p:last-of-type	Sélectionne chaque <p> élément qui est le dernier <p> élément de son parent

Les sélecteurs
Les pseudo-classes

D’autres pseudos classes :

Pseudo classe	Description
:not(p)	Sélectionne chaque élément qui n'est pas un élément <p>
p:nth-child(2)	Selects every <p> element that is the second child of its parent
p:nth-of-type(2)	Sélectionne chaque <p> élément qui est le second <p> élément de son parent

Les sélecteurs

Les pseudo-classes

D’autres pseudos classes :

Pseudo classe	Description
input:checked	Sélectionne chaque élément <entrée> coché
input:disabled	Sélectionne chaque élément <input> désactivé
input:enabled	Sélectionne chaque élément <input> activé
input:focus	Sélectionne l'élément <input> qui a le focus
input:invalid	Sélectionne tous les éléments <input> avec une valeur non valide

Les sélecteurs
Les pseudo-classes

D’autres pseudos classes :

Pseudo classe	Description
input:optional	Sélectionne les éléments <input> sans attributs "required"
input:read-only	Sélectionne les éléments <input> avec un attribut "readonly" spécifié
input:read-write	Sélectionne les éléments <input> sans attribut "readonly"
input:required	Sélectionne les éléments <input> avec un attribut "required" spécifié
input:valid	Sélectionne tous les éléments <input> avec une valeur valide

Les sélecteurs
Les pseudo-éléments

Les pseudos-élements sont similaires aux pseudos-classes mais ne dépendent pas d'actions utilisateurs.

➔ Les pseudo-éléments s'appliquent sur un sélecteur
- Selecteur + " :: " + Pseudo-classe

Les sélecteurs

Les pseudo-classes

D’autres pseudos classes :

Pseudo élément	Description
p::first-letter	Sélectionne la première lettre de chaque élément <p>
p::first-line	Sélectionne la première ligne de chaque élément <p>
p::selection	Sélectionne la partie d'un élément qui est sélectionnée par un utilisateur
input:focus	Sélectionne l'élément <input> qui a le focus
input:invalid	Sélectionne tous les éléments <input> avec une valeur non valide

Les sélecteurs
Sélection multiple

CSS offre des sélecteurs permettant de sélectionner divers balises à travers le DOM (hiérarchie, sélection multiple, …).

Sélecteur	Description
*	Permet de sélectionner tous les éléments
E, F	Sélection des éléments E et F
E F	Sélection des éléments F présent dans les éléments E
E > F	Sélection des éléments F directement enfant des éléments E
E ~ F	Sélection des éléments F étant des frères de E (même niveau dans le DOM)

Les sélecteurs
Sélection avec attributs

Sélecteur	Description
E[src]	Sélection des éléments E possédant l'attribut "src"
E[src="val"]	Sélection des éléments E possédant l'attribut "src" ayant pour valeur "val"
E[src^="val"]	Sélection des éléments E possédant l'attribut "src" et dont la valeur de l'attribut commence par "val"
E[src^="val"]	Sélection des éléments E possédant l'attribut "src" et dont la valeur de l'attribut termine par "val"
E[src*="val"]	Sélection des éléments E possédant l'attribut "src" et dont l'une de ses valeur est "val"

3.

Les couleurs

Les couleurs

Propriétés de couleurs

Formatage du texte
Les couleurs

Les couleurs s'utilisent avec la propriété : "color"
Une couleur peut être définies de plusieurs façons :

➔ Un mot-clé réservé : blue, red, black, chocolate, …

Vous pouvez trouver ses couleurs sur différentes sources (http://colours.neilorangepeel.com/)

➔ Une valeur hexadécimale : #00ff00, #a9a9a9

Des palettes sont intégrées dans les navigateurs, des logiciels de manipulation d'images ou sur internet (http://www.code-couleur.com/)

➔ Une valeur RGB (teinte de rouge, vert et bleu) : rgb(24,125,255), rgb(100,17,45)

Formatage du texte
Les couleurs

CSS3 introduit de nouvelles façons d'appeler des couleurs :

- ➔ RGBA : Même principe que RGB avec l'ajout de l'opacité (entre 0 et 1) : rgba(0, 0, 255, 0.5)

Cela introduit la transparence.

- ➔ HSL (Teinte, Saturation, Luminosité) : hsl(195, 49%, 50%), hsl(345, 14%, 85%)
- ➔ HSLA : même principe que HSL avec l'introduction de l'opacité comme pour RGBA : hsla(170, 50%, 45%, 1), hsla(170, 50%, 45%, 0.5)

Formatage du texte
Les couleurs

Exemple :

Côté HTML :
 Contenu du texte

Côté CSS :
```
.text-blue-dark
{
	color : #011283;
}
```

Formatage du texte
Propriétés de couleurs

La couleur peut-être utilisé avec différentes propriétés :

color : couleur du texte
background-color : fond de l'élément
border-color : couleur de la bordure

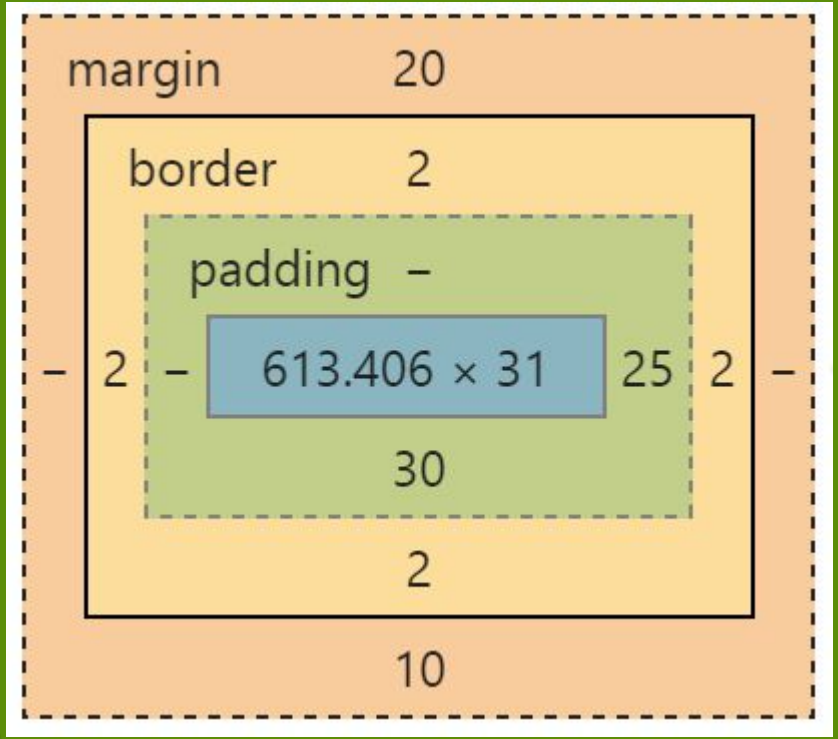
4.

Le modèle de boite

Présentation
Les unités
Largeur et hauteur
Extremums de dimensions
Les bordures
Les bordures arrondies
Les marges extérieures
Les marges intérieures

Le modèle de boite

Présentation



Le modèle de boite
Les unités

Les unités écran de pc :

Unité	Description
px	Définition d'une taille en pixel
%	Définition d'une taille en pourcentages par rapport au parent direct
em	Définition d'une taille proportionnelle à la taille du texte de son parent.
rem	Définition d'une taille proportionnelle à la taille spécifiée pour la balise html.

Le modèle de boite
Les unités

Prérequis pour utiliser l'unité "rem" :

```
html { font-size: 62.5%; } /* 1rem vaut 10px */
body { font-size: 1.4rem; } /* 1.4 rem vaut 14px */
h1   { font-size: 2.4rem; } /* 2.4 rem vaudra 24px */
```

Le modèle de boite
Les unités

Les unités imprimantes :

Unité	Description
em	Définition d'une taille proportionnelle à la taille du texte de son parent.
cm, mm	Définition d'une taille en multiple de mètres
pt	Définition d'une taille en points (1pt = 1/72 of 1in)
in	Définition d'une taille en inches (1in = 96px = 2.54cm)

Le modèle de boite
Largeur et hauteur

Il n’est pas possible de dimensionner des éléments de type “inline”. Pour les autres :

Propriétés de largeur : “width”

Propriétés de hauteur : “height”

Le modèle de boite

Largeur et hauteur

Exemple :

Côté HTML :

```
<div id="test">
        <img src="..." alt=".." />
        ...
</div>
```

Côté CSS :

```
#test
{
        width: 100%;
        height: 300px;
}
```

Le modèle de boite
Extremums de dimensions

Il est possible de définir un maximum et un minimum aux dimensions.

➔ La dernière valeur l'emporte

Exemple :

```
article
{
        width: 55%;
        max-width: 700px;
        min-width: 300px;
}
```

Le modèle de boite
Les bordures

Il existe 3 propriétés pour définir une bordure :

border-width : épaisseur de la bordure

border-color : couleur de la bordure

border-style : style de la bordure

Le modèle de boite

Les bordures

Il existe 3 propriétés pour définir une bordure :

- ➔ border-width : épaisseur de la bordure
- ➔ border-color : couleur de la bordure
- ➔ border-style : style de la bordure (solid, double, inset, outset, dashed, dotted, groove, ridge, none, hidden)

Le modèle de boite

Les bordures

➔ Il est possible de définir de manière différentes les 4 bordures (haut, droit, bas, gauche)

Exemple :

```
p {
   border-top-style: dotted;
   border-right-style: solid;
   border-bottom-style: dotted;
   border-left-style: solid;
   border-top-width: 2px;
   border-right-width: 4px;
   border-bottom-width: 6px;
   border-left-width: 8px;
   border-top-color: blue;
   border-right-color: cornsilk;
   border-bottom-color: #a95c5c;
   border-left-color: rgba(36,14,171,0.5);
}
```

Le modèle de boite
Les bordures

➔ Propriétés raccourcies

Exemple :

```
p {
        border-color: grey;
        border-width: 8px 6px; /* haut et bas */
        border-style: dotted solid ridge double ; /* haut, droite, bas, gauche (horloge) */
}
```

Le modèle de boite
Les bordures arrondies

➔ La propriété “border-radius” permet d’appliquer des arrondies aux bordures :

```
p {
    border: 2px solid chocolate;
    border-radius: 6px;
    border-top-left-radius: 10px;
}
```

Remarque : les raccourcis sont aussi disponibles.

Le modèle de boite
Les bordures arrondies

➔ Les raccourcis sont aussi disponibles :

```
p {
    border: 2px solid chocolate;
    border-radius: 6px 2px;
    /* (haut-gauche/bas-droite) (haut-droit/bas-gauche) */
}
```

Le modèle de boite
Les marges extérieures

- Les marges extérieurs sont définies grâce à la propriété “margin” :
  - La valeur est une unité de mesure
  - Les raccourcis sont disponibles
  - La valeur “auto” centre horizontalement la balise de type block (si la largeur est définie)
  - La valeur “inherit” définit la même largeur que le parent

Le modèle de boite
Les marges intérieures

➔ Les marges intérieures sont similaires aux marges extérieures sauf que l'espace définit se trouve entre la bordure et le contenu. La propriété à utiliser est : "padding"

5.

Formatage du texte

Présentation
Polices de caractères
Taille de police de caractères
Alignement
D'autres propriétés
Débordements

Formatage du texte
Présentation

CSS offre des possibilités pour mettre en forme le texte.

Le formatage simple :

➔ Mise en italique : “font-style:italic;”
➔ Mise en gras : “font-weight: bold;”

Remarque : “font-weight” accède des nombres :

Formatage du texte

Police de caractères

Les polices de caractères sont définies grâce à la propriété “font-family” :

➔ Ces valeurs sont une liste de police de caractères classées par ordre de priorité :

Exemple :

font-family: Verdana, Arial, sans-serif, serif;

Remarque : on sert plusieurs polices de caractères au cas où le navigateur/utilisateur ne possède pas la police.

Web safe-fonts : http://www.cssfontstack.com/

Formatage du texte
Taille de police de caractères

Les tailles de polices de caractères sont définies grâce à la propriété “font-size” :

Exemple :

```
P
{
        font-family: Verdana, Arial, sans-serif, serif;
        font-size: 1.5rem;
}
```

Formatage du texte
Taille de police de caractères

Les décorations de texte s'utilisent avec la propriété "text-decoration" :

➔ Valeurs : underline (soulignement), overline (ligne au dessus), line-through (barré),

Exemple :

```
P
{
        text-decoration: underline;
}
```

Formatage du texte
Taille de police de caractères

Il est possible d'obtenir des lettres capitales grâce à la propriété "font-variant" prenant la valeur "small-caps" :

Exemple :

```
P
{
        font-variant: small-caps;
}
```

Formatage du texte
Taille de police de caractères

La super-propriété “font” permet de définir les différentes propriétés de formatage du texte :

➔ font-weight, font-style, font-size, font-variant, font-family

Exemple :

```
p{
        font: 800 italic 1.5rem small-caps Verdana,Arial,serif;
}
```

Formatage du texte
Alignement

Pour aligner le texte :

➔ Propriété : text-align

➔ Valeurs : left, right, center, justify

Exemple :

```
p{
        text-align: center;
}
```

Formatage du texte
D'autres propriétés

- ➔ Taille d'une ligne : "line-height: 2;" (par défaut 1)
- ➔ Espace entre chaque lettre : "letter-spacing: 2px;"

Exemple:

```
p {
   font-style: oblique;
   font-weight: bold;
   line-height: 2;
   letter-spacing: 2px;
}
```

Formatage du texte
D'autres propriétés

- ➔ white-space : espace entre les paragraphes (pre, pre-line, …)
- ➔ text-indent : alinéa (unité de mesures)

Exemple :

```
p{
        white-space: pre-wrap;
        text-indent: 3rem;
}
```

Formatage du texte
D'autres propriétés

➔ Word-wrap : forcer le retour à la ligne d'un mot trop long qui excéderait les dimensions définies

Exemple :

```
p{
        word-wrap: break-word;
}
```

Formatage du texte
Débordements de contenu

Lorsque le contenu d'un bloc excède ses dimensions, nous pouvons gérer le débordement.

- ➔ Propriété : overflow
- ➔ Valeurs : visible (par défaut), hidden, scroll, auto

Exemple :

```
p{
        overflow: auto;
}
```

Formatage du texte
Débordements de contenu

Le débordement peut être géré selon l'axe X et Y

➔ Propriété : overflow-x, overflow-y

Exemple :

```
p{
        overflow-x: hidden; /* dépassement mot trop long */
        overflow-x: scroll; /* dépassement contenu trop long */
}
```

6.

Les fonds d’écrans

Super-propriété

La couleur de fond

L’image

Répétition de l’image de fond

Comportement de l’image au scroll

Les fonds d'écrans

Super-propriété

La super-propriété "background" permet de définir les propriétés d'un fond d'écran avec une image :

```
section {
   background: #ffffff url("img/banner.png") no-repeat right top fixed ;
}
```

Propriétés :

- ➔ background-color (si transparence de l'image)
- ➔ background-image
- ➔ background-repeat
- ➔ background-attachment
- ➔ background-position

Les fonds d'écrans

La couleur de fond

Pour définir une couleur de fond :

- ➔ Propriété : "background-color"
- ➔ Valeur : valeur de la couleur

Exemple :

```
p {
   background-color: darksalmon;
}
```

Remarque : lors de l'utilisation d'une image de fond, la couleur apparait "en-dessous" de l'image, il faut donc une transparence de l'image pour voir la couleur.

Les fonds d'écrans

L'image

Pour définir une image :

➔ Propriété : "background-image"
➔ Valeur : url de l'image

Exemple :

```
p {
   background-image: url("img/banner.gif");
}
```

Remarque : lors de l'utilisation d'une image de fond, la couleur apparait "en-dessous" de l'image, il faut donc une transparence de l'image pour voir la couleur.

Les fonds d'écrans

Répétition de l'image de fond

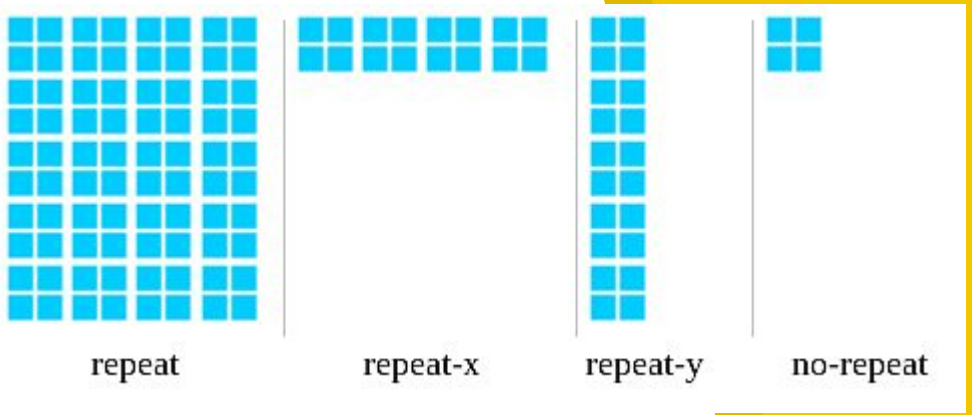
Définit la répétition de l'image :

➔ Propriété :

"background-repeat"

➔ Valeur :

Repeat (par défaut), repeat-x, repeat-y, no-repeat



Exemple :

```
p {
   background-repeat: no-repeat;
}
```

Les fonds d'écrans

Position de l'image

Pour définir la position de l'image si l'image est plus petite que son contenant.

Propriété : "background-position"

➔ Valeur : 2 mesures (axe x, axe y) soit en mot-clé soit en unité de mesure

Exemple :

```
p {
        background-image: url("img/banner.gif");
        background-position : 50% 50%; /* centré */
}
```

Mot clé : top, right, bottom, left, center

Les fonds d'écrans

Comportement de l'image au scroll

Pour définir le comportement de l'image lors du parcours vertical de la page :

Propriété : "background-attachment"

➔ Valeurs : scroll, fixed, …

Exemple :

```
p {
        background-image: url("img/banner.gif");
        background-attachment: fixed
}
```

7.

Positionnement

Centrer un bloc horizontalement

L’affichage “inline-block”

Position avec “float”

La propriété “position”

La propriété “position” - absolute

La propriété “position” - relative

La propriété “position” - fixed

Positionnement

Centrer un bloc horizontalement

Pour centrer automatiquement une balise de type block, utiliser une marge automatique suffit :

```
article
{
        width: 30rem;
        margin: auto;
}
```

Remarque : il faut définir une largeur car l'élément vaut 100% par défaut.

Positionnement

Centrer un bloc horizontalement

Si vous voulez centrer un élément de type inline, il est possible de redéfinir son affichage par défaut.

Exemple :

```
#presentation > a
{
        display: block;
        width: 200px;
        margin: auto;
}
```

Remarque : il faut définir une largeur car l'élément vaut 100% par défaut.

Positionnement
L'affichage "inline-block"

Nous connaissons actuellement 2 modes d'affichages : "inline" et "block". D'autres existent, notamment "inline-block". Ce mode d'affichage permet :

- ➔ D'attribuer des dimensions comme un élément "block"
- ➔ D'avoir plusieurs éléments "inline-block" sur la même "ligne" comme les élément "inline".

Exemple :

```
section article
{
        display: inline-block;
}
```

Positionnement
L'affichage "inline-block"

Il est possible de gérer l'alignement des éléments "inline-block" :

➔ Propriété : vertical-align

➔ Valeurs : top, middle, bottom, …

Exemple :

```
section > article {
   display: inline-block;
   width: 33%;
   vertical-align: middle;
}
```

Positionnement
Position avec “float”

Il est possible de positionner les éléments avec la propriété “float”

➔ Valeurs : left, right

- Cela nous permet par exemple de créer une lettrine (lettre stylisé en début de paragraphe).
- En outre, cela peut aussi nous servir à placer des blocs entre eux pour créer un layout.

Mettre en place ses deux possibilités.

Positionnement

Position avec “float”

Dans le cas où 2 éléments flottent l’un à côté de l’autre, il est possible de forcer le retour à la ligne entre les deux éléments :

- ➔ Propriété : clear
- ➔ Valeurs : left, right

La propriété “clear” s’appliquer au second élément : il empêche la flottaison avant lui.

Positionnement

Position avec “float”

Si un élément est plus grand que l'élément qui le contient et qu'il utilise “float”, il débordera à l'extérieur de son conteneur.
Pour ça, il est possible d’utiliser un hack sur le conteneur :

```
.clearfix::after {
    content: "";
    clear: both;
    display: table;
}
```

Positionnement
La propriété “position”

Une des méthodes pour placer des éléments entre eux est la propriété “position”. Elle possède différente valeur :

- ➔ absolute
- ➔ fixed
- ➔ relative

Positionnement

La propriété “position” - absolute

La valeur “absolute” de la propriété “position” permet de placer un élément par rapport à la page entière.

Pour préciser l’emplacement voulu, il faut indiquer les valeurs de décalage par rapport à la page :

- ➔ top : distance entre l’élément et le haut de la page
- ➔ right: distance entre l’élément et le bord droit de la page
- ➔ bottom: distance entre l’élément et le bas de la page
- ➔ Left : distance entre l’élément et le bord gauche de la page

Positionnement

La propriété “position” - absolute

Exemple :

Côté HTML :

```
<aside id="position-absolute">
        <h2>Un article</h2>
        <p>e dividendo ac partiendo docet, non quo modo efficiatur concludaturque ratio tradit, non qua via captiosa solvantur ambigua distinguantur ostendit; iudicia rerum in sensibus ponit, quibus si semel aliquid falsi pro vero probatum sit, sublatum esse omne iudicium veri et falsi putat.</p>
</aside>
```

Côté CSS :

```
#position-absolute
{
        width: 15rem;
        position: absolute;
        right:3rem;
        top: 3rem;
}
```

Positionnement

La propriété “position” - fixed

Similaire à la valeur “absolute”, la valeur “fixed” permet de positionner un élément dans la page en définissant sa distance entre les bords de la page.

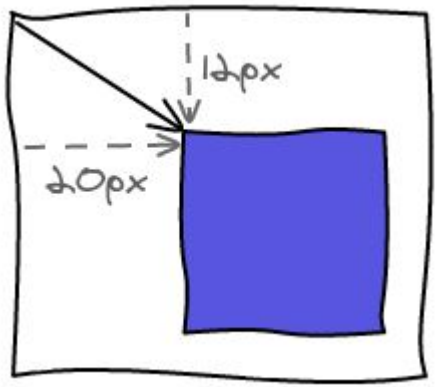
La différence est que le placement est “fixe”, c’est à dire que lorsque l’utilisateur parcours la page (scroll), l’ élément reste à sa place (à la même place sur l’écran de l’utilisateur).

Exemple d’application : barre de navigation et/ou footer.

Positionnement

La propriété “position” - relative

La valeur “relative” permet de déplacer un élément par rapport à sa position initiale.



Les valeurs de décalage (top, right, bottom, left) prennent comme origine le coin supérieur gauche de l’élément à sa position initiale.

Positionnement
La propriété “position”

Lorsque qu’il y a des imbrications de propriétés “position”, prenez garde !

L’origine, à la base la page entière, est remplacée par le parent ayant une position à “absolute” le plus proche.

Merci de votre attention !

=)