Web Programming 1.2 CSS appliquées aux Eléments Sémantiques de Contenu
Initiation CSS

Nous avons utilisé HTML5 pour un **contenu sémantique**,

… **CSS** est maintenant utilisé pour appliquer un **style** au **contenu**

Sommaire

Anatomie CSS
Duo propriété / valeur

1. Comment appliquer les CSS ?
   - L'attribut style d'un élément
   - L'élément style dans le head du document
   - Lien vers un fichier CSS externe
2. Sélecteurs primaires : l'élément HTML, class, id, héritage
3. Style du Texte: texte et polices de caractères, *web font*, listes, liens
4. Styles des Eléments *Block* : *box model* (*margin* & *padding*), bordure, arrière-plans, style des tableaux
5. Sélecteurs complexes: *Child selectors*, *Attribute selectors*, *pseudo-classes*, *pseudo-elements*, …

Anatomie d'une règle CSS

: séparateur

propriété : valeur ⇐ déclaration CSS

Nom de propriété

Valeur, avec unité le cas échéant

Comment appliquer les CSS ?

a. Style déclaré au niveau de l'élément: **l'attribut *style*** pour appliquer simplement, essentiellement pour test.

Jeux de paires *propriété*:*valeur*, séparés par **;**

<element style="property:value; property:value;**…"> </element>**

b. L'élément **style** déclaré dans le head

Paires de *propriété* : *valeur* entre accolades

Le **sélecteur** définit les éléments HTML concernés par ces règles CSS

```
<style>
selector {property: value; property:value;… }
</style>
```

c. CSS déclarées dans un fichier externe, lié au document web actuel

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

myCSS_file.css ne contient que des règles CSS

Pourquoi un fichier externe?
Etude de cas: un site web

Sélecteurs primaires : élement, class, id, hétitage

déclaration CSS élement HTML

- Type de sélecteur ≡ simplement un élement HTML

element {property : value; … }

<element attribute="value"> … </element>

- Sélecteur de classe ≡ tous les éléments HTML5 ayant un attribut *class* avec le nom de classe défini

Notez le . Devant le nom de la classe

.className {property : value; … }

<element class="className"> … </element>

- Sélecteur *id* ≡ élément HTML5 ayant la valeur id

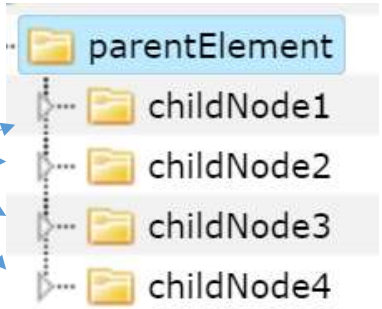
Notez le **#** Devant le nom de la classe

#idValue {property : value; … }

<element id="idValue"> … </element>

- Héritage CSS ≡ éléments HTML5 enfants selon le D.O.M

parentElement {property : value;… }



… mais avec des exceptions…
Certaines propriétés ne s'héritent pas !

Style du Texte: *text* et *font*, *web font*

Détails sur le style du texte et de la police, comme la police appliquée au texte, sa taille, en gras ou en italique, ..., et la mise en forme du texte (alignement du texte dans son conteneur), …

Unités : px, pt, em, rem, …

```
8  <p style="font-family:Arial;font-size:14pt;text-align:justify;">
9  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris s
   velit convallis molestie. Vestibulum bibendum ipsum libero. Fusce
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris suscipit mauris sit amet velit convallis molestie. Vestibulum bibendum ipsum libero. Fusce feugiat orci quis massa sollicitudin accumsan. Fusce pretium tempor lacus quis vestibulum. Suspendisse nulla lorem, vulputate a malesuada sit amet, ultricies vel ipsum.

Ces propriétés CSS remplacent les éléments HTML5 obsolètes (*font*) et les attributs dépréciés (*align*).

CSS font : https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Fonts ⇒ font-….. : value
CSS text : https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Text ⇒ text-….. : value

Polices Web utilisant un service en ligne, ex. Google Fonts (https://fonts.google.com/)

```
5     <title>Google Font link</title>
6  <link href="https://fonts.googleapis.com/css?family=Charmonman" rel="stylesheet">
7  <style>
8  p {font-family: 'Charmonman', cursive;}
9  </style>
```

```
5     <title>Google Font @import</title>
6  <style>
7  @import url('https://fonts.googleapis.com/css?family=Charmonman');
8  p {font-family: 'Charmonman', cursive;}
9  </style>
```

```
5     <title>Google Font @font-face local</title>
6  <style>
7  @font-face {
8    font-family: "Charmonman";
9    src: url("./charmonman/Charmonman-Regular.ttf");
10 }
11 p {font-family: 'Charmonman';}
12 </style>
```

Style du Texte : listes, liens

Listes L'attribut ***type*** des listes ordonnées et non ordonnées (*ol*, *ul*) spécifiant l'apparence (puces) d'un élément de liste est désormais **déprécié**!
⇒ **list-style-type** doit être dorénavant utilisée.

Valeurs possibles : *disc*, *square*, *circle*, une valeur '*string*' (Firefox), ou *none* (listes sans puces, cas du menu de navigation basé sur des listes stylisées)

Liens Pour chaque **état** du lien, correspond une *pseudo-classe*

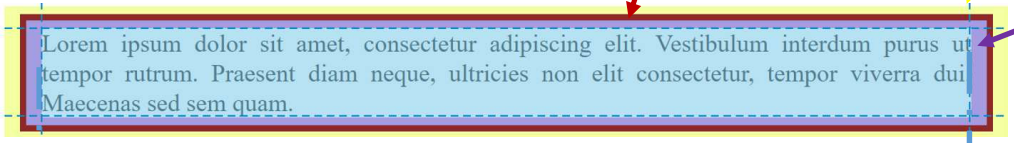
Link (non visité)	a:link {text-decoration:none; }
Visited	a:visited { }
Hover	a:hover {background-color:#CDFEAA; }
Focus	a:focus { }
Active	a:active { }

Cela vous permet de modifier les styles par défaut : *text-decoration*, *cursor* style, *colors*, …

Style des élément *Blocks* : *box model* ou modèle de boite (*margin* & *padding*)

Propriétés : **border** **margin** **padding**

Chaque élément HTML peut avoir une **bordure** ,une marge **externe**, une marge **interne**, dont les valeurs sont définies en px



HTML Code

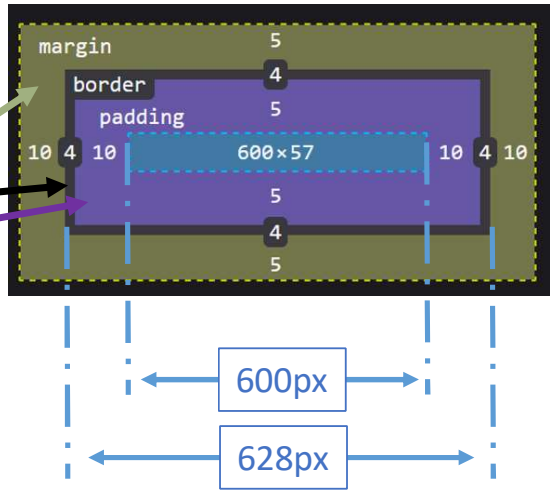
```
<p class="p0">Lorem ipsum dolor
purus ut tempor rutrum. Praesent
dui. Maecenas sed sem quam.
</p>
```

Le contenu à une largeur de 600px

CSS

```
p {text-align: justify;}
.p0 {
    width:600px;
    margin: 5px 10px 5px 10px;
    border:4px solid red;
    padding:5px 10px 5px 10px;
}
```

Box model



La largeur totale de la *boite* est de 628px (4+10+600+10+4), la marge externe qui entoure la *boite* est eclue !

The *Box Model* – Etude de cas … et la propriété CSS *box-sizing*

Taille de la boîte et espace utilisé, qui dependent des épaisseurs *border* et *padding*, margin est fixé à 0 ici, parce que la marge externe n'est pas prise en compte dans le modèle *box size*.

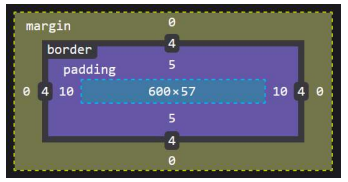
Largeur fixée à 600px, texte justifié

Total Width :

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum interdum purus ut tempor rutrum. Praesent diam neque, ultricies non elit consectetur, tempor viverra dui. Maecenas sed sem quam.

border + padding

```
border:4px solid red;
padding:5px 10px 5px 10px;margin: 0;
```

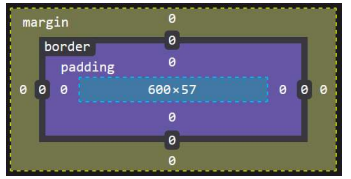


628px

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum interdum purus ut tempor rutrum. Praesent diam neque, ultricies non elit consectetur, tempor viverra dui. Maecenas sed sem quam.

ni *border*, ni *padding*

```
padding:0;
```

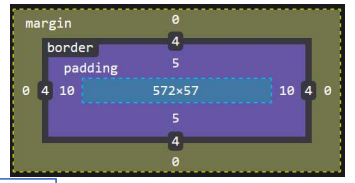


600px

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum interdum purus ut tempor rutrum. Praesent diam neque, ultricies non elit consectetur, tempor viverra dui. Maecenas sed sem quam.

border + padding

```
border:4px solid red;
padding:5px 10px 5px 10px;
box-sizing: border-box;
```



600px

4+10+572+10+4

+box-sizing de valeur *border-box*

➡ Il peut donc être difficile de disposer des éléments qui devraient avoir la même largeur malgré une marge ou une bordure différente.

box-sizing peut résoudre cette difficulté

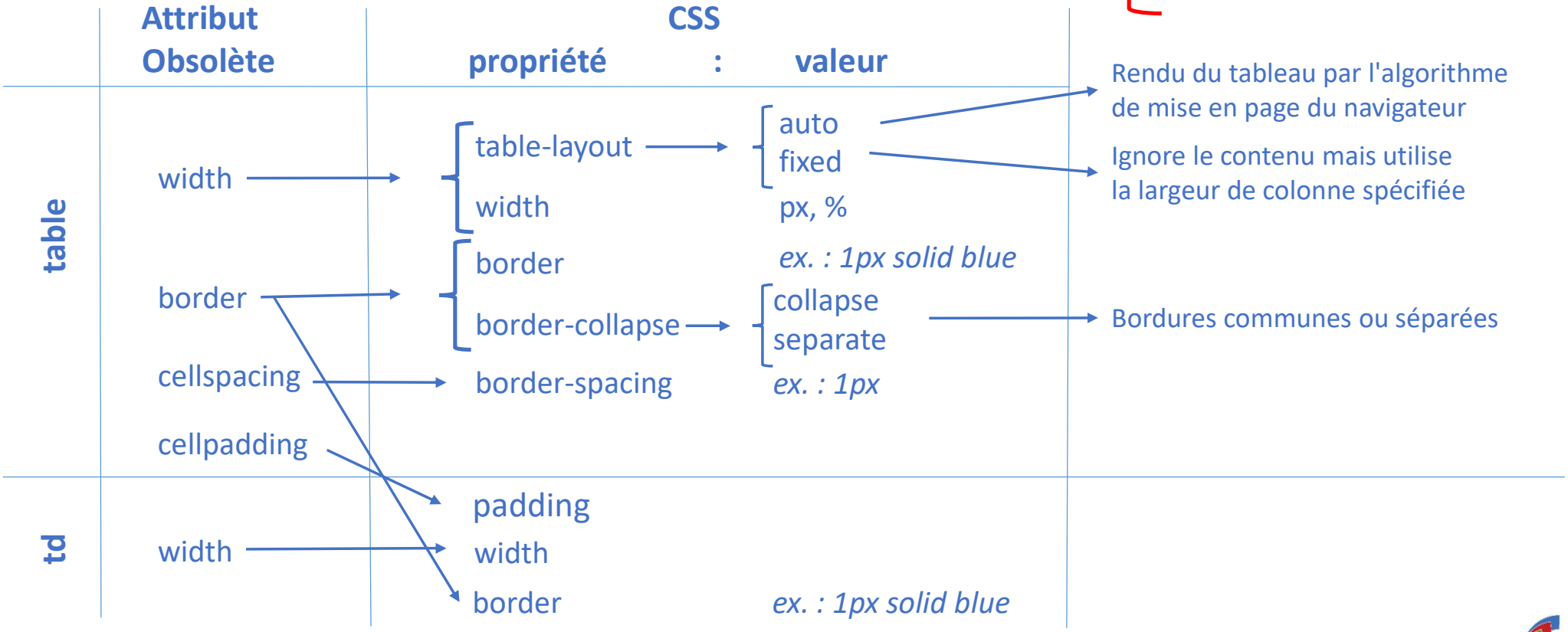
Style des Tableaux HTML

Données Tabulaires => tableau à 2 dimensions avec colonnes et lignes

Mais certains attributs sont obsolètes!

~~border~~
~~cellpadding~~
~~cellspacing~~
~~width~~

…mais



Sélecteurs complexes : *Child selectors, Attribute selectors, pseudo-classes, pseudo-elements, …*

Sélecteur primaire

```
<h1>Title Level 1</h1>
<p id="para1">Phrasing content</p>
<h2 class="title2">Title Level 2</h2>
```

h1 {property:value;} Sélectionne tout élément h1
#para1 { } Sélectionne *l'unique* élément dont l'attribut id est égal à para1
.title2 { } Sélectionne tout élément ayant un attribut de classe qui contient le mot title2

Child selector - Descendant

```
<p>
    <i>italic word</i>
    <a>
        <i>italic link</i>
    </a>
</p>
```

p i { } Sélectionne tout élément i qui est un descendant d'un élément p

Child selector – Direct Child

```
<p>
    <i>italic word</i>
    <a>
        <i>italic link</i>
    </a>
</p>
```

p > i { } Sélectionne tout élément i qui est un enfant *direct* d'un élément *p*

Guide : https://learn.shayhowe.com/advanced-html-css/complex-selectors/
Description CSS : http://gallery.theopalgroup.com/selectoracle/

Sélecteurs complexes : *Child selectors, Attribute selectors, pseudo-classes, pseudo-elements, …*

Sélecteur **frère (sibling)** et **immédiatement adjacent frère** (**adjacent sibling**)

```
<body>
    <h1>Tile 1</h1>
    <p>...</p>
    <h2>Title 2</h2>
    <p>Text</p>
    <blockquote>
        <p>...</p>
    </blockquote>
    <p>...</p>
```

h2 ~ p { } — Sélectionne tout élément p qui suit un élément h2 avec un parent commun

h2 + p { } — Sélectionne tout élément p qui suit immédiatement un élément h2

Attribute selectors

```
<a href="#" target="_blank">...</a>
<a href="https://www.qwant.com/">...</a>
```

a[target] { } — Sélectionne tout élément ayant un attribut **target**

a[href="https://www.qwant.com/"] { } — Sélectionne tout élément ayant un attribut **href** qui est égal à: *https://www.qwant.com/*

a[href*="qwant"] {...} — Sélectionne tout élément ayant un attribut **href** qui contient la chaîne *qwant*

a[href^="https://"] {...} — Sélectionne tout élément ayant un attribut **href** qui commence avec *https://*

a[href$=".com"] {...} — Sélectionne tout élément ayant un attribut href qui se termine par *.com*

Sélecteurs complexes : *Child selectors, Attribute selectors, pseudo-classes, pseudo-elements*, …

Pseudo-classes :(first| last | only)-child, nth-child, (first| last | only)-of-type

```
<ul>
    <li>First item selected</li>
    <li></li>
    <li>Item, multiple of 3, selected</li>
    <li></li>
    <li></li>
    <li>Item, multiple of 3, selected</li>
    <li>Last Item selected</li>
</ul>
```

li:first-child { }

Sélectionne tout élément li qui est un premier enfant

li:nth-child(3n) { }

Sélectionne tout élément li qui est un enfant sur trois à partir du troisième enfant

li:last-child { }

Sélectionne tout élément li qui est un dernier enfant

Pseudo-classes :***hover***

```
<p>Element selected when a user moves
their cursor over the element</p>
```

p:hover { }

Sélectionne tout élément p qui est dans un état de survol (événement *over*)

Pseudo-elements + génération de contenu

```
<strong>Text between CSS
double quotes</strong>
```

strong:before {content: "«";}
strong:after {content: "»";}

Sélectionne tout contenu placé devant un élément *strong* (et ajoute du contenu)
Sélectionne tout contenu placé après un élément *strong* (et ajoute du contenu)

… to be continued