

04/01/2022

CSS 3 Partie 1

CSS



MODULE B1.PROG WEB

Le Principe des feuilles de style	2
Historique	2
Intégrer les styles CSS	2
STYLE EN LIGNE : le plus prioritaire	3
STYLE INTERNE.....	3
style externe	4
La déclaration de style CSS.....	5
Les selecteurs	5
Les sélecteurs de type	5
Les sélecteurs de classe	6
Les sélecteurs d'identification	7
Sélecteur multiple.....	9
Notions de cascade.....	9
La notion d'heritage.....	10
Les unites de mesures.....	12
Les couleurs	12

LE PRINCIPE DES FEUILLES DE STYLE

Les feuilles de style sont des ajouts de code au langage HTML qui vont prendre en charge la présentation du document.

Le concept de feuilles de style repose sur le **concept** de la séparation du **contenu** et de la **présentation** dans l'élaboration d'applications HTML.

Le rôle du HTML se limite ainsi à la structure de la page. Pour aller au bout du **concept** presque tous les éléments de présentation des balises ou des attributs présents dans le HTML 4.0 ont disparu.

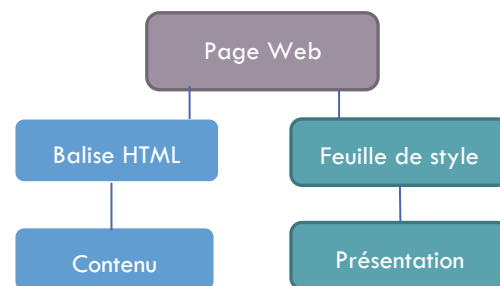
Citons, par exemple, l'absence en HTML5 de l'attribut `border` qui fixe une bordure à la balise de tableau `<table>`

Citons encore la balise `` qui est supprimée. Ainsi la ligne suivante :

```
<h1><font color="green">TODO write content</font></h1>
```

Est remplacé par la ligne suivante avec l'attribut **style**.

```
<h1 style="color:red"> TODO write content</h1>
```



C'est pourquoi les styles CSS seront définis dans des feuilles.

La modification d'un style dans cette feuille, et ce sont des centaines de pages qui voient leurs présentations modifiées.

HISTORIQUE

Les premières pages Web étaient pour le moins austères avec leur fond de page gris, un texte de couleur noire et structuré uniquement avec des titres et des listes.

Assez rapidement cependant, dès 1994, devant les relatives faiblesses du HTML en matière de présentation, l'idée d'adjoindre au HTML des styles, comme dans les logiciels de traitement de texte, fit son apparition.

Lorsqu'en 1995 le W3C (pour World Wide Web Consortium) commença à fonctionner, une de ses premières réalisations concrètes fut la publication en novembre 1995 d'une première ébauche de travail (working draft) sur les feuilles de style. Ce document devint une recommandation officielle le 17 décembre 1996 sous le nom de "Feuilles de style en cascade niveau 1" (Cascading Style Sheets, level 1) ou de manière abrégée CSS1.

Dans la foulée, un groupe de travail spécifique fut créé avec comme mission d'étendre le concept des feuilles de style. Une nouvelle recommandation fut publiée le 12 mai 1998 sous le vocable CSS2.

Quelques corrections et des extensions mineures sont proposées courant 2006 sous la dénomination CSS2.1.

Depuis 2009, il est question du CSS3 qui est un ensemble de module. Pour connaître les modules des CSS3, voici l'URL du W3C indispensable à connaître et à consulter régulièrement : <http://www.w3.org/Style/CSS/current-work#CSS3>.

Pour chaque module le W3C donne une indication de statuts avec un code couleur. Ici REC : le module est officiellement terminé et est proposé en recommandation.

Cela signifie que CSS 3 est un standard du Web qui évolue encore. Rien n'est figé. Ce qui est vrai aujourd'hui peut être modifié demain. Il convient de savoir si telle ou telle propriété est exploitable ou non, si elle est reconnue par tel ou tel navigateur et aussi selon la version des navigateurs. Pour ce faire, la solution la plus utilisée actuellement est de tester les propriétés sur le site **Can I use** : <http://caniuse.com>.

Abbreviation	Full name
WD	Working Draft
LC	Last Call
CR	Candidate Recommendation
PR	Proposed Recommendation
REC	Recommendation

INTEGRER LES STYLES CSS

Il faut donc lier la structure HTML à la mise en forme et mise en page CSS. Pour ce faire, nous avons plusieurs possibilités qui vont répondre chacune à une fonctionnalité précise.

STYLE EN LIGNE : LE PLUS PRIORITAIRE

Déclarer les styles CSS directement dans l'élément HTML concerné ou **style en ligne**, avec l'attribut global `style`. Cette possibilité permet d'appliquer un style CSS de manière unique.

Cette façon de procéder est cependant à éviter pour 2 raisons principales :

1. Pour respecter le principe de la séparation du contenu et de la présentation.
2. Il n'est pas rare de rencontrer des sites de plusieurs centaines de pages. Une mise à jour graphique peut ainsi constituer un travail gigantesque si le style est appliqué de façon individuelle à chaque balise

Exemple :

HTML	VISUEL
<code><h1 style="color: red;">titre en rouge</h1></code>	titre en rouge
<code><h1 style="font-style: italic;">titre en italic</h1></code>	<i>titre en italic</i>
<code><h1> titre en style normal </h1></code> <code><!--style par défaut --></code>	style normal

Remarque : chaque élément HTML possède un style par défaut. Dans le dernier exemple, pour la balise **h1**, après inspection de l'élément (F12 dans le navigateur) nous avons le style suivant :

Une balise **h1** s'affiche par défaut avec une taille de 2em, en gras, et avec des marges de 0,67em.

```
h1 {
  display: block;
  font-size: 2em;
  margin-block-start: 0.67em;
  margin-block-end: 0.67em;
  margin-inline-start: 0px;
  margin-inline-end: 0px;
  font-weight: bold;
}
```

L'application d'un style en ligne vient à **ajouter** et/ou **remplacer** au style par défaut une propriété CSS.

Dans l'exemple 1 : nous **ajoutons** au style par défaut, la propriété **color**. Notre titre **h1** aura sa couleur en rouge.

Dans l'exemple 2 : les propriétés CSS `display` et `font-size` sont **remplacées** par de nouvelles.

```
element.style {
  color: red;
}
h1 {
  display: block;
  font-size: 2em;
  margin-block-start: 0.67em;
  margin-block-end: 0.67em;
  margin-inline-start: 0px;
  margin-inline-end: 0px;
  font-weight: bold;
}
```

Figure 1: exemple 1

```
element.style {
  color: red;
  display: inline;
  font-size: 1em;
}
h1 {
  display: block;
  font-size: 2em;
  margin-block-start: 0.67em;
  margin-block-end: 0.67em;
  margin-inline-start: 0px;
  margin-inline-end: 0px;
  font-weight: bold;
}
```

Figure 2: Exemple 2

STYLE INTERNE

Permet d'intégrer les styles CSS dans la page HTML, avec la balise `<style>`, dans la balise `<head>`. Tous les styles CSS définis dans cette page pourront être appliqués à tous les éléments HTML de cette page.

On note qu'il est toujours possible d'intégrer du style en ligne. Dans ce cas là, c'est ce dernier le plus prioritaire avec toujours le même principe d'ajout ou de remplacement de propriété CSS.

Exemple : Toutes les balise <h1> sont en couleur rouge par défaut. Le style en ligne est plus prioritaire.

HTML	VISUEL
<pre><!-- dans la balise <HEAD> --> <style> h1 { color: red; } <!-- dans la balise <BODY> --> <h1> titre niveau 1 en rouge </h1> <h1 style="font-style: italic;"> titre niveau 1 en rouge et en italique </h1> <h1 style="color: blue;"> titre niveau 1 en bleu </h1></pre>	<p>titre niveau 1 en rouge</p> <p><i>titre niveau 1 en rouge et en italique</i></p> <p>titre niveau 1 en bleu</p>

Dans le dernier exemple l'inspection de l'élément donne le résultat suivant :

```
element.style {
  color: blue;
}

h1 {
  color: red;
}

h1 {
  display: block;
  font-size: 2em;
  margin-block-start: 0.67em;
  margin-block-end: 0.67em;
  margin-inline-start: 0px;
  margin-inline-end: 0px;
  font-weight: bold;
}
```

Le style en ligne remplace la couleur définie dans le style interne et applique une couleur bleue à une balise en particulier

Le style interne applique une couleur rouge à toutes les balise h1

Style par défaut d'une balise h1 : taille 2em, en gras, avec marge

STYLE EXTERNE

Créer un fichier .css (ou plusieurs) séparée qui contiendra toutes les règles CSS. Cette fois la portée est maximale, puisque tous les styles définis dans le fichier .css pourront être appliqués à tous les éléments HTML de toutes les pages Web. Cette façon de procéder respecte au mieux la séparation du contenu et de la présentation.

Un fichier .css contient uniquement des déclarations de style. Il ne contient pas de balises HTML

Pour lier la page HTML avec sa feuille de style, on ajoutera entre les balises <head> et </head>, un ou plusieurs liens vers la feuille de style en question :

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

Dans le code suivant on note les 4 niveaux de style qui s'ajoutent les uns aux autres du plus global au plus prioritaire :

STYLE FINAL = STYLE PAR DEFAUT + STYLE EXTERNE + STYLE INTERNE + STYLE EN LIGNE :

HTML	CSS externe
<pre><link href="style.css" rel="stylesheet" type="text/css"/> <!-- dans la balise <HEAD> --> <style> h1 { color: red; } </style> ... <!-- dans la balise <BODY> --> <h1> titre niveau 1 en rouge </h1> <h1 style="font-style: italic;"> titre niveau 1 en rouge et en italique </h1> <h1 style="color: blue;"> titre niveau 1 en bleu </h1></pre>	<pre>/*commentaire : fichier style.css*/ /* ajout d'une ligne au-dessus du titre */ h1 { text-decoration: overline; }</pre>
VISUEL	
<p>titre niveau 1 en rouge</p> <p><i>titre niveau 1 en rouge et en italique</i></p> <p>titre niveau 1 en bleu</p>	<pre>element.style { font-style: italic; } h1 { color: red; } h1 { text-decoration: overline; } h1 { display: block; font-size: 2em; margin-block-start: 0.67em; margin-block-end: 0.67em; margin-inline-start: 0px; margin-inline-end: 0px; font-weight: bold; }</pre> <p>Style en ligne</p> <p>+ Style Interne</p> <p>+ Style Externe</p> <p>+ Style par défaut</p> <p>= STYLE FINAL</p>


LA DECLARATION DE STYLE CSS

La déclaration d'un style s'effectue par le binôme

propriété : valeur ;

Exemple :

L'exemple suivant pourrait se lire : appliquer sur les balises h1 une couleur d'arrière-plan jaune sur texte rouge, et une décoration de texte souligné

CSS	VISUEL
<pre>h1 { color: red; text-decoration: underline; background-color: yellow; }</pre>	

Détaillons cette déclaration :

- La **propriété** identifie ce qui sera défini dans le style adopté. Ces propriétés sont nombreuses et sont énumérées dans les spécifications CSS :
 - Police de caractère (font)
 - Sur le texte (text)
 - Sur l'arrière-plan (background)
 - Sur la bordure (border), etc.
- La **valeur** identifie la nature de l'effet de style souhaité. La valeur s'exprime par
 - un mot-clé
 - un pourcentage
 - une grandeur en fonction de la propriété à laquelle elle est assignée.
 - Et même par une fonction (par exemple rgb())
- Une déclaration de style se termine toujours par un point-virgule.
- Il est possible de définir plusieurs déclarations de style pour un sélecteur. Un par ligne pour plus de lisibilité.

Par exemple :

```
propriété1:valeur;
propriété2:valeur;
propriété3:valeur;
```

LES SELECTEURS

Ce sont les sélecteurs qui indiquent où s'appliquent les styles CSS. Les sélecteurs et les styles CSS constituent les règles CSS.

Quelques règles de syntaxe sur le nommage des sélecteurs :

- Le nom des sélecteurs n'est pas sensible à la casse. Par contre, les éléments inclus, comme les polices de caractères ou les URL y sont sensibles. Donc, pour plus de facilité, il est d'usage de n'utiliser que des minuscules.
- Le nom des sélecteurs ne doit pas comporter d'espaces, de caractères accentués, ni de caractères spéciaux (+, \$, @, #...).
- Enfin, le nom des sélecteurs ne doit pas commencer par un chiffre.

Nous allons détailler 3 types de sélecteurs pour commencer : TYPE, CLASSE, D'IDENTIFICATION

LES SELECTEURS DE TYPE



<https://codepen.io/samuel-gibert/pen/WNZJzNy>

La syntaxe d'un sélecteur de type est la dénomination de la balise suivie de la déclaration de style comprise entre des accolades ouvrantes et fermantes. Ce type de sélecteur cible tous les éléments HTML du type du sélecteur

```
NomDeLaBalise { déclaration de style }
```

Exemple:

```
/* Cibler tous les éléments <p> et appliquer le style sur toutes ces balises*/
p { background-color: red;}
```

Remarquez que l'on reprend simplement la dénomination de l'élément HTML sans ses signes inférieur (<) et supérieur (>). Donc uniquement le texte de la balise.

Ce qui pourrait se lire : "appliquer à toutes les balises <p> l'effet de style décrit entre les accolades".

LES SELECTEURS DE CLASSE

<https://codepen.io/samuel-gibert/pen/WNZJzXL>

Les sélecteurs de classe CSS permettent de cibler des éléments d'un document en fonction du contenu de l'attribut "class" de chaque élément. Ce sélecteur se distingue par le point "." devant le sélecteur.

```
.nom_de_la_classe { déclaration(s) de style}
```

Soit un point, suivi du nom que vous voulez attribuer à la classe, suivi de la déclaration de style entre des accolades.

Cette définition de classe pourra être utilisée pour n'importe quelle balise du document ou de l'application qui possède l'attribut HTML class="nom_de_la_classe".

Exemple :

HTML	CSS externe
<p><p class="rouge"> Bienvenue </p></p> <p>On note dans l'exemple que le '.' disparaît lorsqu'il est utilisé comme style pour la balise.</p>	<pre>.rouge{ background-color: red; color:white; font-size: 1.5em; }</pre>
VISUEL	

La même classe peut être appelée plusieurs fois quel que soit la balise HTML sur laquelle elle s'applique. Le sélecteur de class est prioritaire par rapport au sélecteur de type

Exemple :

HTML	CSS externe
<pre><h1 class="bleue">Lorem ipsum dolor sit amet</div> <h1>Lorem ipsum dolor sit amet</h1> <p class="bleue">Lorem ipsum dolor sit amet</p></pre> <p>On note ici que la class "bleue" est attribué à une balise <h1> et <p> ce qui entraîne un visuel légèrement différent. Cela s'explique du fait que le style externe s'ajoute au style par défaut de chacune des balises</p>	<pre>h1 { /* sélecteur de type */ background-color: red; color:white; } .bleue { /* sélecteur de class */ background-color: blue; color:white; font-size: 1.5em; }</pre>
VISUEL	

Il existe une autre façon de définir une classe :

```
nom_balise.nom_de_la_classe {déclaration(s) de style}
```

Soit un nom de balise, suivi d'un point, suivi du nom que vous voulez donner à la classe, suivi de la déclaration de style entre des accolades.

Exemple :

HTML	CSS externe
<pre><div class="rouge"> item 1 <blockquote class="rouge"> coucou </blockquote> </div></pre> <p>On note que le style s'est appliqué qu'à la balise <blockquote></p>	<pre>blockquote.rouge { background-color: red; color:white; }</pre>
VISUEL	
<div>item 1</div> <div>coucou</div>	

Il est possible d'appliquer plusieurs "class" sur une même balise. Il suffit de séparer le nom par un espace.

Exemple :

HTML	CSS externe
<pre><p class="rouge souligne"> coucou </p> <h1 class="rouge souligne"> Bienvenue </h1></pre> <p>La balise <h1> récupère le style de la class ".souligne" et ".rouge"</p> <p>La balise <p> récupère le style de la class ".souligne" et ".rouge" et comme il s'agit d'une balise <p> le style "p.rouge.souligne" s'applique aussi avec un soulignement jaune.</p>	<pre>.souligne{ text-decoration: underline; } .rouge { background-color: red; color:white; } p.rouge.souligne{ text-decoration-color: yellow; }</pre>
VISUEL	
<p>The diagram illustrates the combination of CSS rules. On the left, a block of CSS for 'p' (display: block, margins: 1em) is combined with a rule for '.rouge' (background-color: red, color: white) and a rule for '.souligne' (text-decoration: underline). These are combined to create the final visual result on the right: a red box containing the text 'coucou' with a yellow underline, and a red box containing the text 'Bienvenue' with a yellow underline.</p>	

LES SELECTEURS D'IDENTIFICATION



<https://codepen.io/samuel-gibert/pen/PoJeRdd>

L'**attribut** id permet d'identifier un élément HTML. Il ne peut y avoir 2 balises HTML avec le même id. Ce point est particulièrement important lorsque nous aborderons le JavaScript.

Le **sélecteur id**, aussi appelé identifiant id, permet d'appliquer une feuille de style sur l'élément HTML portant cet "id".

La définition d'un sélecteur id est :

```
#nom_de_l'identifiant {déclaration(s) de style}
```

Soit un dièse (#), suivi du nom que vous voulez donner à l'identifiant id (ou déjà donné à votre balise HTML), suivi de la déclaration de style entre des accolades.

Exemple :

HTML	CSS externe
<pre><p id="rouge"> coucou </p> <p id="rouge"> coucou </p> <p> Bienvenue </p></pre> <p>Un identifiant id ne peut figurer qu'une fois dans le document HTML. Ainsi, ce qui suit serait incorrect !</p>	<pre>#rouge { background-color: red; color: white; }</pre>
VISUEL	
<div> <div> Duplicate ID "rouge". From line 22, column 9; to line 22, column 24 (Rule Category: Element IDs) ----- (Alt-Enter shows hints) </div> <div> <div>coucou</div> <div>Bienvenue</div> </div> </div>	

On peut mélanger les déclarations class et id.

HTML	CSS externe
<pre><p id="rouge" class="bleu"> coucou </p> <p class="bleu"> coucou </p></pre>	<pre>#rouge { background-color: red; color: white; } .bleu { background-color: blue; color: white; }</pre>
VISUEL	
<div> <div> <pre>#rouge { background-color: red; color: white; } .bleu { background-color: blue; color: white; }</pre> </div> <div> <p>Dans la 1^{ère} balise <p> de l'exemple nous avons un sélecteur de class et un sélecteur d'identification qui agissent tous 2 sur la propriété background-color. Dans ce cas-là le sélecteur d'identification est prioritaire, comme illustré avec F12, et annule les propriétés du sélecteur de class.</p> </div> </div>	<div> <div>coucou</div> <div>coucou</div> </div>

Dans ce nouvel exemple la dernière balise combine 2 types de sélecteurs :


HTML	CSS externe
<pre><p id="rouge" class="taille"> coucou </p> <p class="bleu"> coucou </p></pre> <p>Dans le 1^{er} <p> nous avons 2 sélecteurs (de class et d'identifications) qui ne sont pas en conflit au niveau des propriétés. Le style qui est appliqué est la somme du style par défaut, du style de l'identificateur, et le style de la class.</p>	<pre>#rouge { background-color: red; color: white; } .bleu { background-color: blue; color: white; } .taille {font-size: 25px; }</pre>
VISUEL	
<div> <div> <pre>p { display: block; margin-block-start: 1em; margin-block-end: 1em; margin-inline-start: 0px; margin-inline-end: 0px; }</pre> </div> <div>+</div> <div> <pre>#rouge { background-color: red; color: white; } .taille { font-size: 30px; }</pre> </div> <div>=</div> <div> <div>coucou</div> <div>coucou</div> </div> </div>	

Comme pour le sélecteur de class, il existe une autre façon de définir un sélecteur d'identification

nom_balise#nom_de_l'identifiant {déclaration(s) de style}

Soit un nom de balise, suivi d'un dièse (#), suivi du nom que vous voulez donner à l'identifiant, suivi de la déclaration de style entre des accolades.

Exemple :

HTML	CSS externe
<pre><p id="rouge"> coucou </p> <h1 id="rouge">essai</h1></pre> <p>Attention comme expliqué précédemment un identifiant id doit être unique. Cet exemple est donc incorrect !</p>	<pre>p#rouge { background-color: red; color: white; }</pre>
VISUEL	
 <p>Le style c'est correctement appliqué qu'à la seule balise <p> possédant l'identifiant rouge.</p>	

SELECTEUR MULTIPLE



<https://codepen.io/samuel-gibert/pen/eYGrrgg>

On peut désigner plusieurs objets pour y appliquer le même style. Dans ce cas, chacune des balises concernées devra être séparée par une virgule.

Exemple 1 :

```
h1,h2,h3 { déclaration de style }
```

Ce qui signifie que les balises <h1>, <h2> et <h3> auront le même effet de style décrit entre les accolades.

Exemple 2 : On applique ici le même style aux éléments h1 et aux balises div de classe "contenu"


```
h1, div.contenu{
    color:blue;
    background-color:white;
}
```

NOTIONS DE CASCADE

Comme vu en début de ce cours, l'intégration du CSS peut se faire à 3 niveaux différents :

- En ligne : directement dans une balise HTML
- En interne : Dans une balise <style> contenu dans le <head>
- En externe : dans un fichier .css utilisé avec <link>


Nous l'avons déjà vu, il peut se produire des conflits entre les différentes spécifications de style. Supposons le cas suivant :

HTML	CSS externe
<pre><style> h1 { text-decoration: underline; color: blueviolet; } </style> ... <h1> coucou </h1></pre>	<pre>h1 { text-decoration: overline; color: green; }</pre> <p>2 sélecteurs de type h1 sont définis en interne et dans un fichier css externe</p>
VISUEL	
 <p>En cas de conflit entre une spécification de style externe et interne, c'est la spécification de la feuille de style interne qui sera retenue par le navigateur</p>	

Il en est de même, en cas de conflit entre une feuille de style interne et en ligne, c'est cette dernière qui l'emportera. L'ordre de priorité de la plus basse à la plus haute :

- Les feuilles de style externes.
- Les feuilles de style internes.
- Les feuilles de style en ligne.

Une directive permet de modifier cette priorité : **!important** et **!veryimportant**

HTML	CSS externe
<pre><style> h1 { text-decoration: underline; color: blueviolet; } </style> ... <h1> coucou </h1></pre>	<pre>h1 { text-decoration: overline !important; color: green !important; }</pre>
VISUEL	
	
Maintenant que la directive !important est ajouté dans le CSS externe nous avons le visuel suivant. C'est le style externe qui vient d'être appliqué	

LA NOTION D'HERITAGE



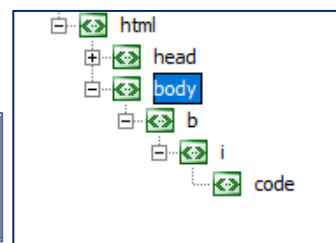
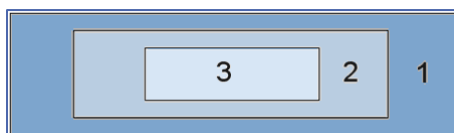
<https://codepen.io/samuel-gibert/pen/MWEGXZz>

Soit le code HTML suivant et son visuel

```
<b>Une phrase en gras, (1)
  <i>avec des parties en italique et (2)
    <code>du code</code> (3)
  </i>
</b>
```

Une phrase en gras, avec des parties en italique et du code

Le HTML est basé sur le modèle PARENT-ENFANTS. Toutes balises à forcément un parent hormis la balise <html> qui est la balise de plus haut niveau. Une balise peut contenir des ENFANTS. C'est le cas dans cette exemple où la balise <code> à un parent <i> qui a lui-même un parent , ayant lui-même un parent <body>, ayant un parent <html> comme illustré.

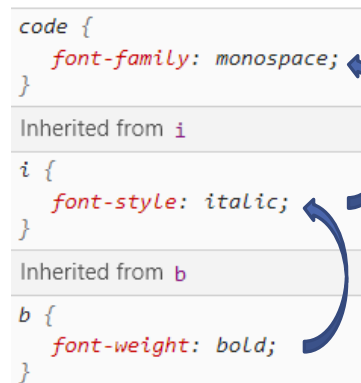


Ce modèle PARENT-ENFANTS entraine le principe d'héritage en CSS : **chaque élément Enfant reçoit (DANS LA PLUPART DES CAS*) en héritage tous les styles (définies ou par défaut) de son élément Parent.**

D'où le nom CSS = **C**ascading **S**tyle **S**heets

En inspectant la balise <code> de l'exemple précédent nous avons le style comme illustré : Le texte de la balise <code> sera avec une police de caractère "monospace" (son style par défaut), et hérite (**Inherited**) aussi de la balise <i>, donc en italique (italic) , et hérite aussi de la balise donc en gras (bold).

On peut aussi le lire : le style de se propage en **cascade** sur ses enfants. Donc le style de s'applique aussi à son enfant <i> qui se propage en cascade sur son enfant <code>



***DANS PLUPART DES CAS !!** : certaines propriétés CSS ne sont pas héritable et ne se transmettent pas du PARENT vers l'ENFANT.

Prenons le cas de la propriété width :

<https://developer.mozilla.org/fr/docs/Web/CSS/width>.

Celle-ci n'est pas héritable.

Héritée

non

Si l'on applique le style suivant à la balise html nous obtenons le visuel suivant

Une phrase en gras, avec des parties en italique et du code

```
<style>
  html {
    color: brown;
  }
</style>
```

Et à l'inspection de la balise <code>

Conclusion : les balises enfants héritent des propriétés de style du parent.

```
code {
  font-family: monospace;
}
Inherited from i
i {
  font-style: italic;
}
Inherited from b
b {
  font-weight: bold;
}
Inherited from html
html {
  color: brown;
}
```

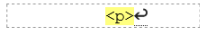
Remarque : toute balise ne peut-être ENFANT de n'importe quel PARENT. Cela est spécifié dans le langage HTML. Par exemple il n'est pas possible de mettre une balise <p> dans une balise <h1>.

Par exemple la balise <h1> appartient à la catégorie de contenu FLUX (voir copie d'écran). Cette balise accepte d'autres balises de type phrasé uniquement.

<u>Catégories de contenu</u>	Contenu de flux, contenu de titre, contenu tangible.
Contenu autorisé	Contenu phrasé.
Omission de balise	Aucune, la balise d'ouverture et la balise de fermeture sont obligatoires
Parents autorisés	Tout élément qui accepte le contenu de flux ; n'utilisez pas de titre comme enfant d'un élément <hgroup> , c'est à présent obsolète.
Rôles ARIA autorisés	tab , presentation
Interface DOM	HTMLHeadingElement (en-US)

Comme la balise <p> est une balise de type FLUX il n'est pas possible de l'imbriquer dans une balise <h1> et l'inverse est également vrai. Pour plus d'information ce site permet de savoir ce qui est autorisé voir le lien ci-dessous https://developer.mozilla.org/fr/docs/Web/Guide/HTML/Content_categories

Il existe un outil permettant de contrôler votre code : W3C validator : <https://validator.w3.org>. Ici l'erreur est expliquée :

2. **Error** Element `p` not allowed as child of element `h1` in this context.
 From line 16, column 13; to line 16, column 15

 Contexts in which element `p` may be used:
 Where flow content is expected.

LES UNITES DE MESURES

De nombreuses unités de mesure sont utilisés. Elles s'expriment en pouces (inches), en centimètres, en millimètres, en points, en picas, en pixels et en pourcentage.

On distingue les valeurs relatives qui peuvent varier selon l'ordinateur utilisé et les valeurs absolues qui restent constantes

Les valeurs absolues : <https://www.w3.org/Style/Examples/007/units.fr.html>

Unité	Nom	Description	Valeur	Exemple
pt	point	72 pt = 1 inch	entier	48pt
pc	pica	1 pc = 12 pt	réel	4.5pc
mm	millimètre	1 mm = .24 pc	entier	60mm
cm	centimètre	1 cm = 10 mm	entier	6cm
in	inch	1 in = 2.54 cm	réel	0.1 in

Les valeurs relatives :

Unité	Description	Valeur	Exemple
em	Unité relative se basant sur la taille de police par défaut de la page.	réel	1.8em
ex	Unité relative à la hauteur de la minuscule de l'élément sélectionné.	réel	1.3ex
px	Le pixel est la plus petite partie d'une image. Dépend de la résolution d'écran.	entier	220px
%	Pourcentage	entier	80%

LES COULEURS

Les feuilles de style CSS proposent de multiples notations pour déclarer une couleur. Soit :

- La notation hexadécimale classique soit #rrggbb avec trois composantes (RGB): rouge, vert et bleu
- La notation hexadécimale abrégée soit #rgb. Chaque chiffre est dupliqué. Par exemple, #fd3 correspond à la notation classique #ffdd33. On ne pourra pas abréger une couleur comme #cfe4f5.
- La notation décimale. Par exemple, color: rgb(0, 0, 255). 255 correspond ici à la valeur hexa FF
- La notation en pourcentage. Par exemple, color: rgb(25%, 50%, 0%). La valeur 0% signifie l'absence de la composante, 100% qu'elle est à son maximum.
- Les mots-clés soit, par exemple, color: red.

À ces notations, le CSS3 a ajouté :

- La notation RGBA. Une composante en plus. Cela devient donc rgba(0,0,0,0). La dernière valeur indiquant le degré d'opacité ou de transparence entre 0 et 1.