http://WWW.CABARE.NET©

CSS 1.0 – CSS 2.1 -

Les feuilles de style CSS – Kompozer et CSS – Cascaded Style Sheet – Normes CSS -

Michel Cabaré – Laurent Lallias – Ver 2.1 – Janv 2010-

Les Styles CSS – Css 1.0 & Css 2.0

Michel Cabaré / laurent Lallias – Ver 2.1 – Janv 2010

www.cabare.net ©

www.lallias.com ©

TABLE DES MATIERES

PRESENTATION DES CSS 4
QU'EST-CE UNE FEUILLE DE STYLE EN CASCADE : 4
AVANTAGES DES FEUILLES DE STYLE : 5
LIEUX DE STOCKAGE DES STYLES & CASCADE: 5
CASCADE - HERITAGE: 6
STYLE – REGLES – SELECTEUR: 6
PROPRIETE ET TYPE DE VALEURS: 7
COMMENTAIRES EN CSS 9
LES SELECTEURS SIMPLES 10
SELECTEUR DE TYPE (REDEFINITION DE BALISE): 10
GROUPER PLUSIEURS SELECTEURS 11
KOMPOZER & FEUILLES CSS & SELECTEURS 12
CREATION D'UNE FEUILLE DE STYLE 12
CREATION DE REGLE 14
MODIFIER UNE REGLE EXISTANTE 15
Utilisation du style avec Kompozer : 16
ATTACHER UNE FEUILLE DE STYLES 16
BALISES DIV & SPAN 17
SUPERPOSITION DE STYLES "DIFFERENTS": 17
SUPERPOSITION AVEC DIV OU SPAN : 18
SUPERPOSITION AVEC DIV ET SPAN : 20
LES SELECTEURS CONTEXTUELS 21
QU'EST CE QU'UN CONTEXTE: 21
SELECTEUR CONTEXTUEL SUR BALISE HTML : 21
KOMPOZER & SELECTEURS CONTEXTUELS 22
CREATION AVEC KOMPOZER - EXEMPLE 1 : 22
Conséquence 22
Création - Mode opératoire 22
Utilisation avec Kompozer 22
CREATION AVEC KOMPOZER - EXEMPLE 2 : 23
Utilisation avec Kompozer : 24
LES SELECTEURS PSEUDO-CLASSE 25
SELECTEUR PSEUDO-CLASSE (A:HOVER …): 25
KOMPOZER & PSEUDO-CLASSE 26
PSEUDO CLASSES 26
Utilisation en Kompozer : 26
PSEUDO CLASSES ET AUTRES SELECTEURS 26
Exemple 26
LES SELECTEURS DE CLASSE 27
SELECTEUR DE CLASSE – ("CLASS" A USAGE REPETE): 27
Exemple : 28

KOMPOZER & SELECTEURS DE CLASSE **29**
SELECTEUR DE CLASSE **29**
Utilisation sur une balise précise avec Kompozer 29
Supprimer l'utilisation sur une balise précise d'un style 30
Utilisation sur une partie de texte avec Kompozer 30
Supprimer l'utilisation sur une partie de texte 30
SELECTEURS CONTEXTUEL & CLASSE **31**
SELECTEUR CONTEXTUEL CLASSE + BALISE: **31**
Classe + Balise 31
CREATION AVEC KOMPOZER **33**
SELECTEUR CONTEXTUEL BALISE + CLASSE: **34**
Balise + Classe 34
CREATION AVEC KOMPOZER **36**
Utilisation avec Kompozer 36
ELEMENTS BLOC – INLINE & MARGES **37**
SELECTEURS GENRE BLOCK OU INLINE: **37**
NOTION DE BOITE - BLOC: **38**
NOTION DE MARGIN - PADDING: **39**
ORDRE DE DEFINITION TOP – RIGHT – BOTTOM - LEFT **40**
CROISEMENT DES MARGES VERTICALES **40**
NOTION DE BORDER : **41**
TAILLE DE BOITE – HEIGHT – WIDTH : **42**
POSITIONS D'ELEMENTS BLOC **43**
POSITION DANS LE FLUX COURANT - NORMAL: **43**
ALIGNEMENT DU TEXTE DANS UN BLOC LINE-HEIGHT – TEXT-ALIGN : **43**
POSITIONNEMENT RELATIF (DANS LE FLUX): **44**
POSITIONNEMENT EN ABSOLU : **44**
ALIGNEMENT D'UN BLOC DANS UNE PAGE MARGIN : AUTO : **47**
POSITIONNEMENT EN FLOTTANT : **47**
GABARIT DE PAGES : **48**
BLOC « NON VISIBLE » DISPLAY - VISIBILITY: **49**
GESTION DES IMPRESSIONS **51**
SPECIFIER LE PERIPHERIQUE @MEDIA: **51**
PREVOIR UNE IMPRESSION: **51**

PRESENTATION DES CSS

Qu'est-ce une feuille de style en cascade :

HTML, conçut au départ comme langage universel pour décrire le contenu et la structure d'un document, et non pas leur aspect, s'est vu progressivement rajouter bon nombre de fonctionnalités, notamment par la balise <FONT>

Mais au lieu d'étendre sans cesse les possibilités du langage HTML, il a été pensé plus sage de décrire la présentation d'un document dans un fichier à part.

Ainsi HTML retrouve sa fonctionnalité première, à savoir décrire la structure et le contenu d'un document, indépendamment de sa mise ne forme, qui se retrouve stockée dans un autre document

le **W3C** est pratiquement à l'initiative de cette solution, même si, comme d'habitude, il n'y a pas unanimité sur la méthode pour décrire la présentation d'un document HTML dans une feuille de style, d'autant plus que ces feuilles étant désormais des documents à part, grande est la tentation d'inventer des normes un peu "propriétaires"…

La définition officielle est disponible auprès du W3C à l'adresse **http://www.w3w.org**

CSS1	**Cascading Style Sheets**	**version 1**	**Décembre 1996**
CSS2	**Cascading Style Sheets**	**version 2**	**Mai 1998**
CSS3	**Cascading Style Sheets**	**version 3**	**En cours …**

Les feuilles de style sont supportées par tous les grands navigateurs depuis leur versions 3.x respectives pour Explorer et Netscape, et des le début pour Mozilla, Opera, Firefox, Safari… mais de façon parfois incomplète et/ou différentes …

N.B: AUJOURD'HUI ENCORE AUCUN NAVIGATEUR NE RESPECTE TOTALEMENT CSS1 , ET ENCORE MOINS CSS2 !
A MEDITER LORSQUE ON VERRA LES DIVERSES BALISES

Une feuille de style c'est donc un modèle de mise en forme applicable à un texte, comme par exemple :

- Gras, italique, souligné
- espacement entre les lignes
- largeur de la marge droite

Avantages des feuilles de style :

Puisque une feuille de style sépare les caractéristiques de présentation du contenu, elle peut permettre de regrouper une série de mise en forme commune et donc faciliter l'homogénéisation de présentation d'un site

Une modification de la feuille de style permettra une modification de l'affichage de toutes les pages HTML qui sont fondées dessus !

Puisque la mise en forme est regroupée dans un fichier distinct, il ne devraient plus y avoir de nouvelles balises HTML spécifiques à telle ou telle mise en forme sur tel ou tel navigateur (du moins en ce qui concerne le HTML "pur", c'est à dire le langage structurel de la page)

lieux de stockage des styles & Cascade:

Malheureusement les styles ne sont pas tous gérés dans des fichiers externes, bien individualisés. Mais plusieurs endroits potentiels peuvent contenir des styles.

Par conséquent les feuilles de styles sont dites en cascades (**CSS** pour **Cascading Style Sheet**) car il existe une notion de priorité dans les mises en formes, de manière à ce que les navigateurs puissent décider en cas de problème, quel style utiliser

Il est possible d'intégrer des styles de plusieurs manières dans une page HTML, essentiellement de trois manières :

- Stocker les styles dans un fichier xxx.css à part via une balise <LINK>
  on parle de **style associés** ou **"linking"** ou **feuille de style**

  Syntaxe:

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

- Stocker les styles dans l'entête de la page HTML entre <HEAD> et </HEAD>
  on parle de **style intégrés** ou **"embedding"** ou de **page**

  Syntaxe:

  ```
  <head>
  <style type="text/css">
  Zone_de_définition_des_styles_de_pages
  </style>
  </head>
  ```

- Stocker les styles en ligne dans une balise HTML, et donc ne s'appliquant qu'au contenu géré par cette balise <xxxx>. On parle alors de **style en ligne** ou de **"in-lining"** ou **incorporés** ou **locaux**

  Syntaxe:

  <balise_html style=" Zone_de_styles_locaux">xxxx</fin_balise>

Cascade - heritage:

La notion d'héritage est une notion simple qui peut vite devenir complexe lorsque les styles sont multiples et variés.

Les règles qui permettent de définir l'héritage sont les suivantes :

- Tous les style sont hérités, sauf les propriété padding et margin (cela ne concerne que le problème de positionnement)
- On récupère tous les styles appliqués aux objets parents, que l'on appelle encore conteneur, sauf si il y a re-définition. Auquel cas, la règle du "dernier qui a parlé" est valable.

En résumé donc, si pas de contrordre, les effets se superposent, et sinon c'est le dernier qui a causé qui a raison !

Style – règles – sélecteur:

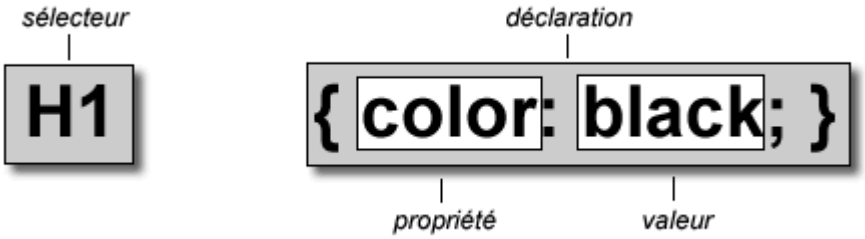
Un style c'est une règle qui s'applique à une partie d'un document HTML

La règle (définition de la règle proprement dite)c'est une liste entre accolades de définitions de propriétés "nom:valeurs" séparées par des points-virgule .

Un style c'est une écriture du genre

```
H1 {font-size: 18pt}
H2 {font-size: 10pt; color:red}
P {font-family : "Arial"; color:#00FF00}
```

Ici ce sont des règles qui redéfinissent des balises HTML mais ce qui compte c'est leur structure



P {font-family : "Arial"; color:#00FF00}

Sélecteur { règle }

Une règle étant un – des couple(s) propriété:valeur séparés par ; et entre { }

{ propriété:valeur }

{ propriété:valeur ; propriété:valeur}

Propriété et type de valeurs:

N.B: noter que dans une règle on utilise le **:** comme symbole d'affectation alors que en HTML pour définir des attributs dans une balise on utilise le **=**

NB: dans les règles les minuscules majuscules sont autorisées, par contre les espacement peuvent avoir de l'importance, ainsi que éventuellement l'ordre dans lequel les propriétés sont énumérés.

h1 {	**font-SIZE: 26pt}**	**correct**
H1 {	**font-size: 26 pt}**	**incorrect (espace entre le 26 et le pt..)**

Il s'agit ici de parler des différentes valeurs possible que l'on peut donner aux propriétés., ces valeurs peuvent être de nature différentes :

Alphabétique:

noter que dans une règle on ne place jamais le texte entre guillemets (sauf très rares exceptions) , alors que en HTML c'est plutôt le cas.

Numérique - unités absolues :

mm	millimètres
cm	centimètres
inch	25.4 mm
pixel	le plus petit éléments affichable sur un écran d'ordinateur

Numérique - unités absolues pour polices de caractères:

pt point qui vaut 1/72 d'inch soit 0.35278 mm

pica vaut 12 points soit 4.233 mm

Numérique - unités relatives pour polices de caractères:

On utilise souvent cette technique pour définir des hauteurs de police.

On peut ainsi définir une hauteur de police par défaut (avec le sélecteur body par exemple) puis exprimer les autres sélecteur (p, h1,…) comme étant une valeur relative de la police utilisée.

Par exemple

```
body{ font-family: arial;
      font-size: 14px;      }
h1  {  font-size: 2.5em;    }
h2  {  font-size: 2em;      }
h3  {  font-size: 1.5em;    }
```

donnerait

Ceci est un texte normal

ceci est un titre 1

Ceci est un titre 2

ceci est un titre 3

Ex est l'équivalent de la hauteur du "x" de la police utilisée

Arial

titre en 1ex

titre en 2ex

titre en 3 ex

Times

titre en 1ex

titre en 2ex

titre en 3 ex

Em est égale à la hauteur du "m" de la casse de la police de caractère utilisée

Arial

titre en 1em

titre en 2em

titre en 3 em

Times

titre en 1em

titre en 2em

titre en 3 em

Les couleurs suivent un régime particulier car elles peuvent être exprimées de manière alphabétique ou numérique, avec les formes suivantes :

Couleurs en alphabétique

Déconseillées formellement, sauf pour des tests rapides de fonctionnement:

aqua – black – blue – fuchsia – gray – green – lime – maroon – navy - olive - purple – red – silver – teal – white – yellow –

Couleurs en triplet Rouge-Vert-Bleu

Existent avec plusieurs notations

décimale	chaque valeur est cotée de 0 à 255
123,191,45	vert- jaune léger
%	chaque valeur est cotée de 0% à 100%
48%, 75%, 18%	vert – jaune –léger
hexadécimale	chaque valeur est cotée de 0 à 255 soit #FF
#7BBF2D	vert – jaune –léger
hexadécimale simple	chaque valeur est cotée de 0 à 15 soit #F
#8C3	la valeur la plus proche du vert – jaune –léger

Commentaires en css

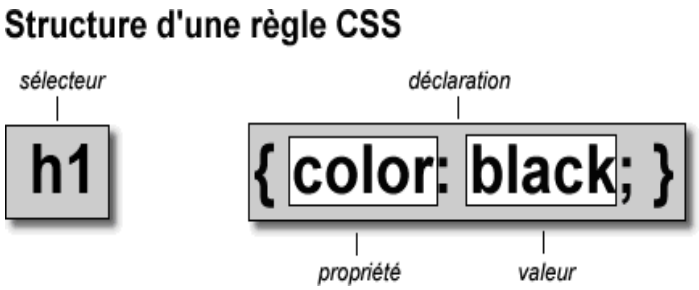
/* Ceci est un commentaire */

LES SELECTEURS SIMPLES

Sélecteur de type (redéfinition de balise):

Lorsque le sélecteur figure tel que devant une déclaration de style, on dit que c'est un **sélecteur de TYPE**, car il s'applique à tous les éléments HTML de ce type…

Syntaxe:




```
<STYLE type="text/css">
        H1 {
            font-family:courier new;
            color:#00FF00
            }
        P {color:#FF0000}
</STYLE>
```

H1 en courier new, vert

tous les paragraphe sont rouge…

et on peut affiner ainsi

```
<STYLE type="text/css">
        Body {margin:0}
</STYLE>
```

Pour toute la page on donne une marge à 0…

A chaque fois que vous utiliserez la balise html **h1**, le texte s'écrira comme il a été défini dans la déclaration. Tous les h1 utilisés dans la page sont naturellement en police courier new, vert (en plus de leurs caractéristiques par défaut)

Grouper plusieurs sélecteurs

Syntaxe:

```
Sélecteur1, Sélecteur2… {déclaration}
```

Mais rien n'empêche de redéfinir des règles supplémentaires pour un des sélecteurs

```
h1, h2{
        font-family: arial;
        color: #ff0000;
}
h2{ color: #00 ff 00}
```

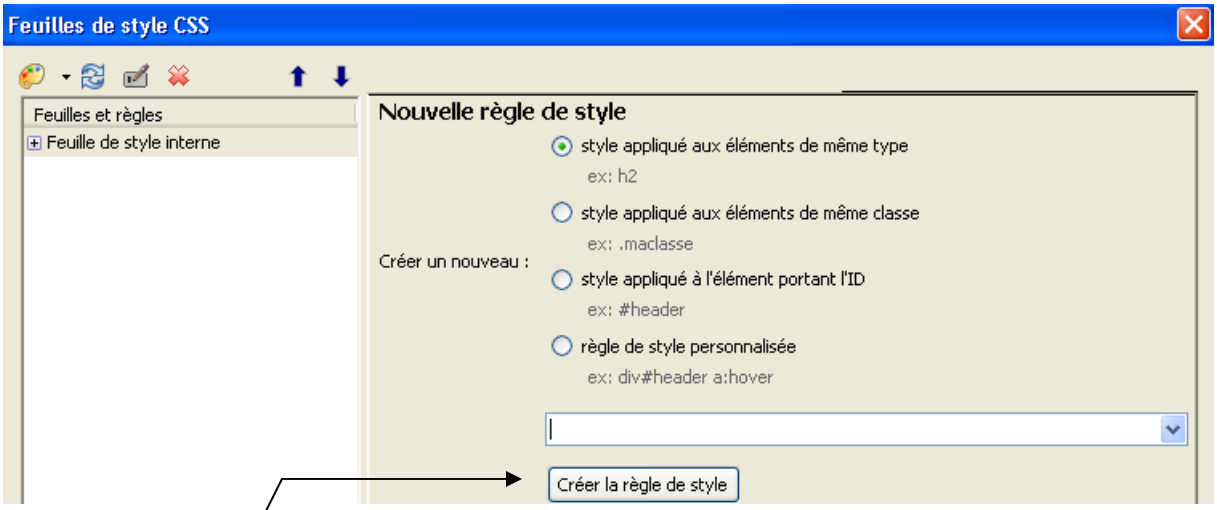
Permet de définir que h1 et h2 ont la même police mais pas la même couleur

KOMPOZER & FEUILLES CSS & SELECTEURS

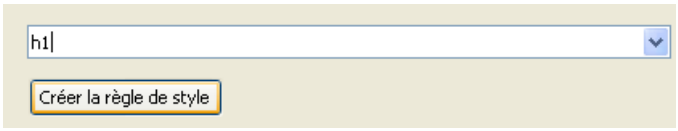
Création d'une feuille de style

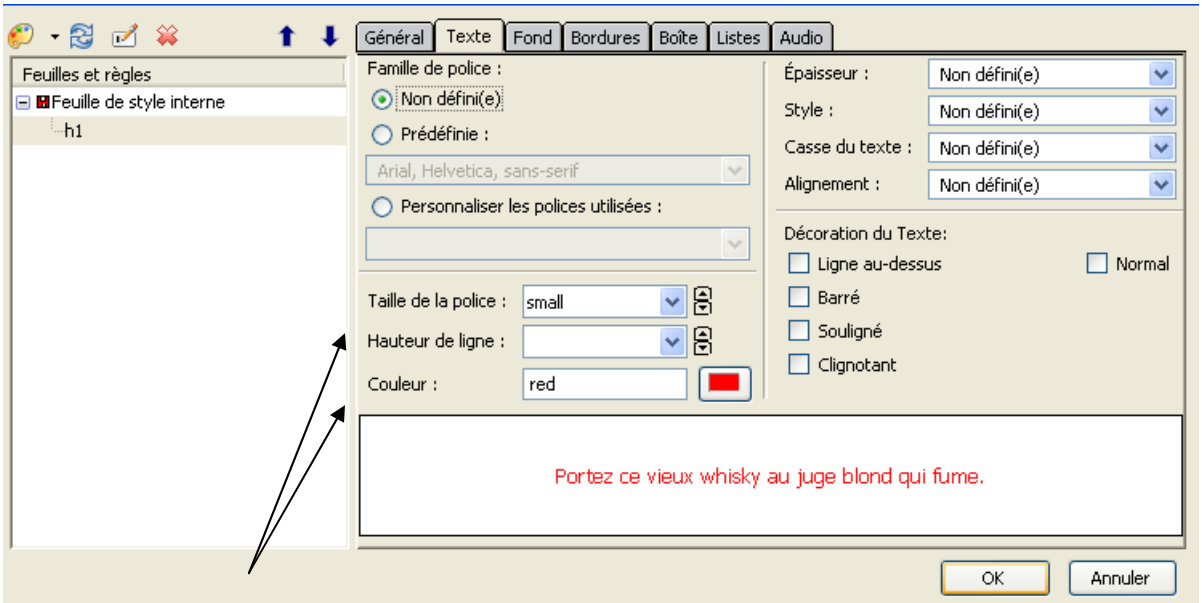
Enregistrez la page web active

Cliquez sur  ou **Outil** / **Editeur Css** ou **F11**



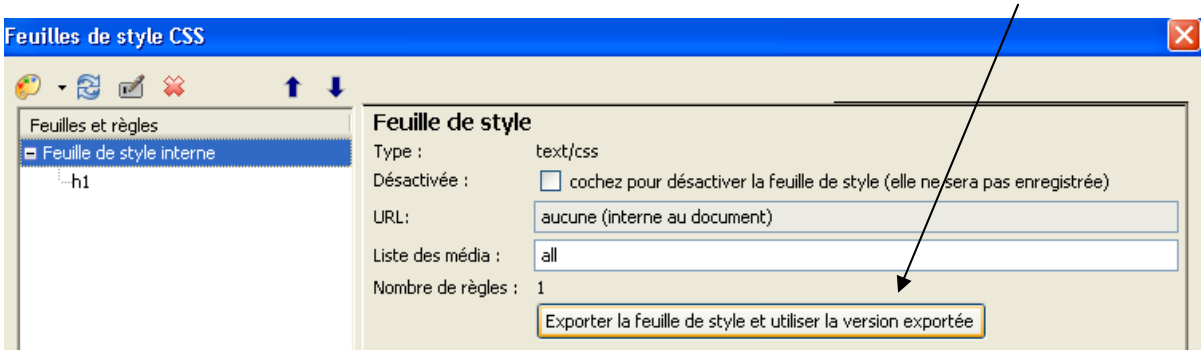
On crée une règle, puis on exportera la règle **…** Par exemple on veut créer la règle **H1-petit-rouge**…

Donc  Donnant

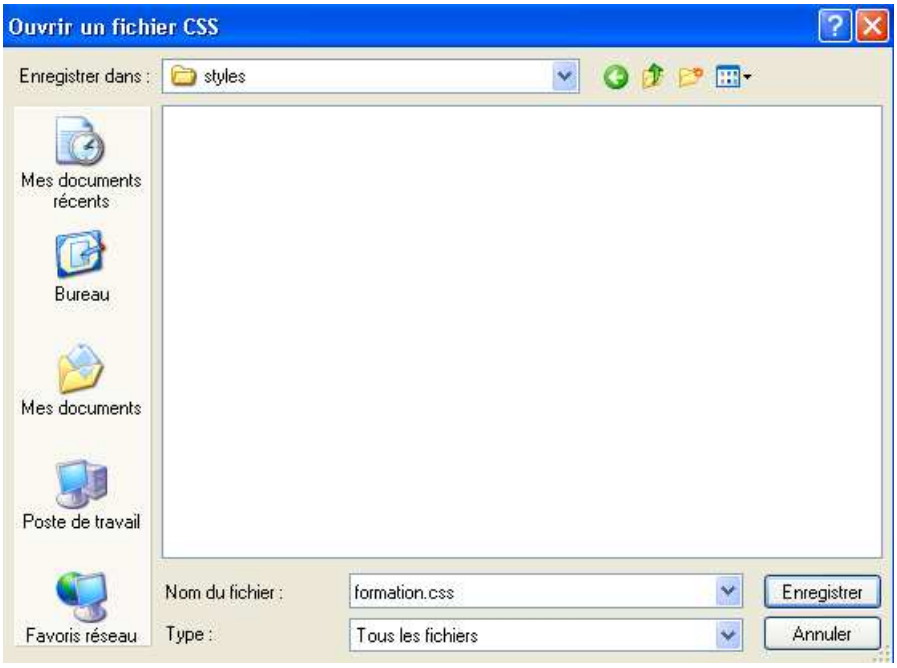


Puis onglet Texte on effectue nos reglages… OK

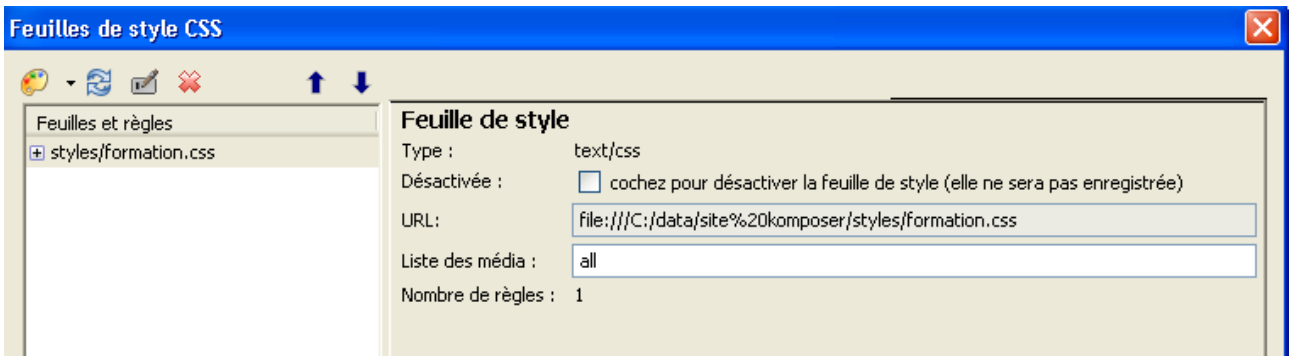
Ensuite on rappelle **F11**, on demande **Exporter le feuille de style et…**



Placez vous dans le dossier qui doit recevoir votre feuille de styles, donnez un nom à votre fichier feuille de styles (N'oubliez pas l'extension .css), puis cliquez sur Enregistrer.



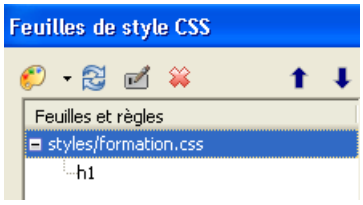
Cela donne



Il n'y a plus qu'a faire OK.

Dans cet exemple la feuille s'appelle *formation.css* et se trouve dans le dossier Styles.

Elle contient le style h1



Création de règle

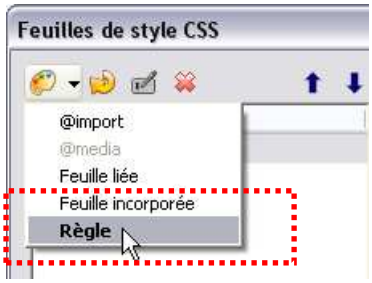
Nous allons redéfinir la balise **body** :

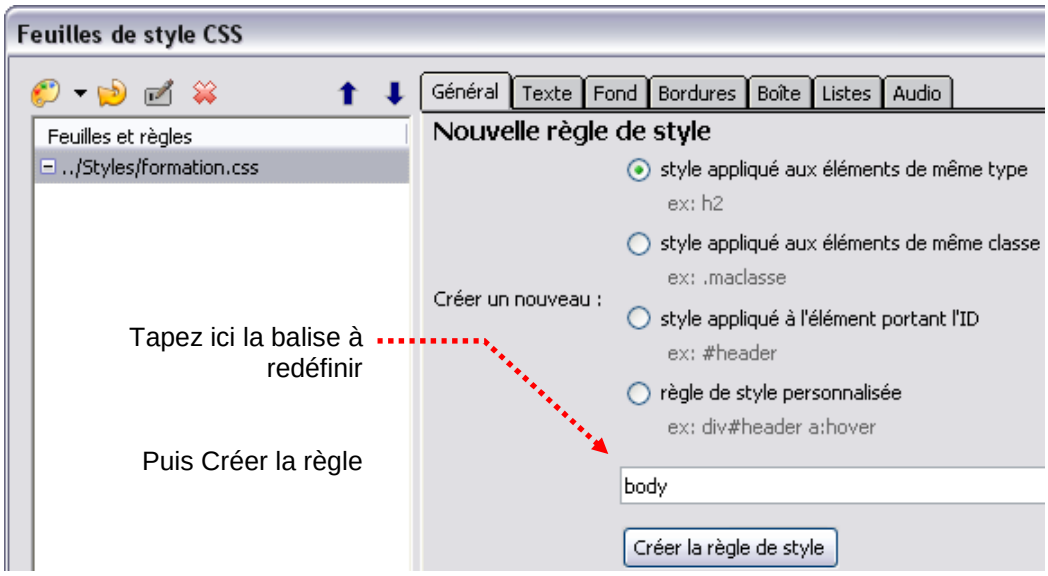
Avec les effets suivants : Police Arial, Taille 18 px, Rouge, Arrière plan Gris

Si besoin cliquez de nouveau sur 

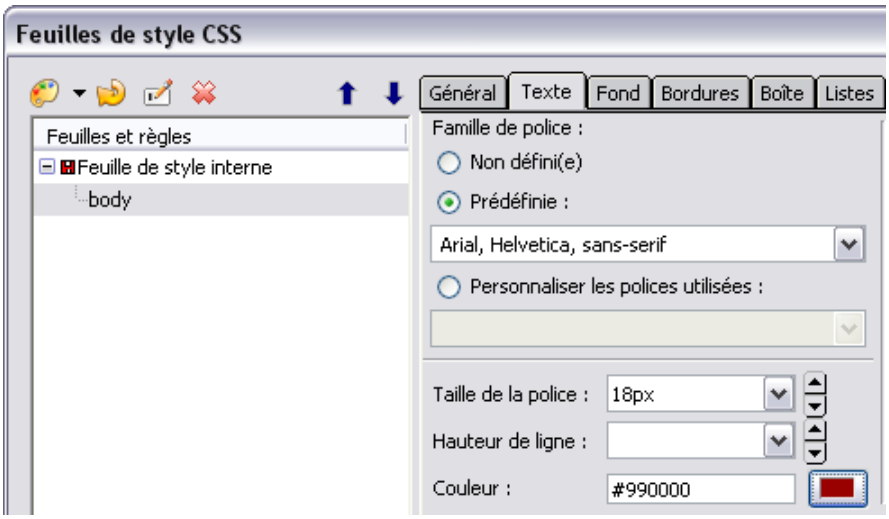
ou **F11**

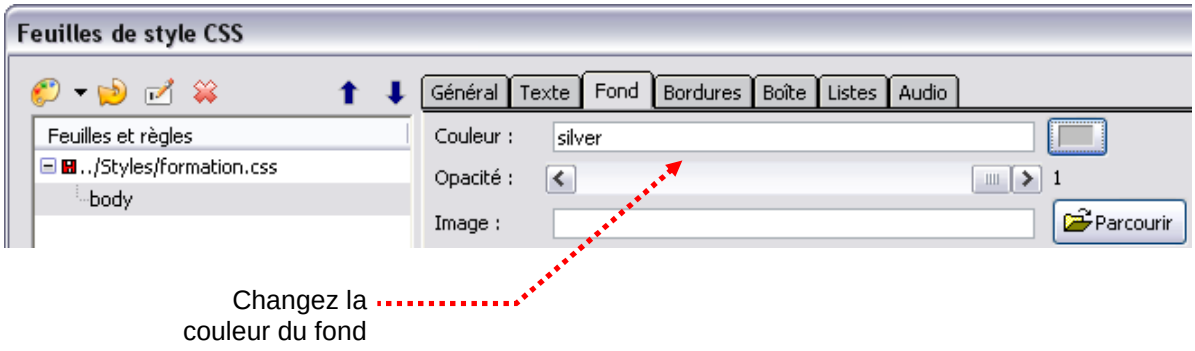
Si besoin cliquez sur **Règle**





Puis réalisez les divers réglages





Fichier HTML

Il n'y a que la relation au fichier feuille de styles

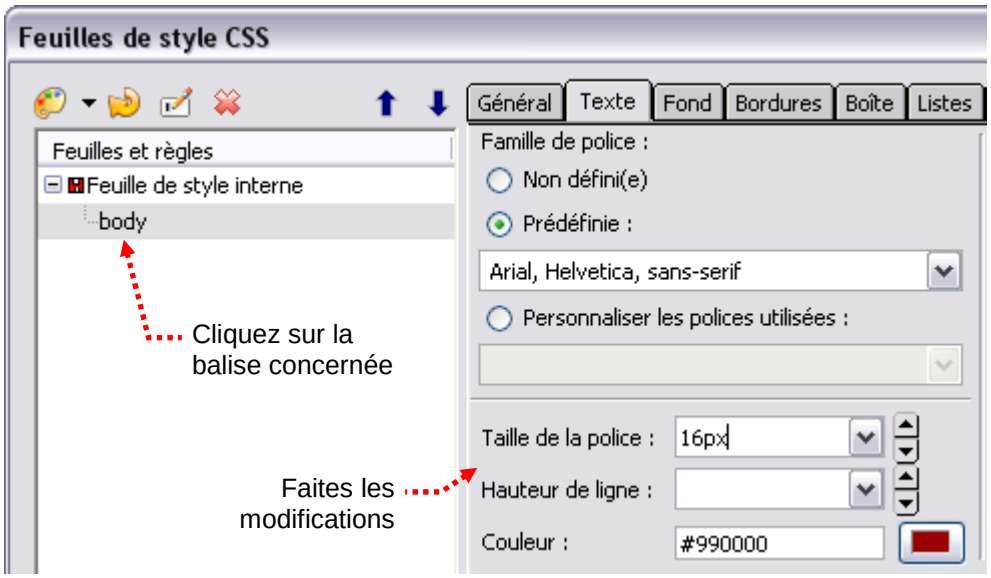
```
<head>
  <link rel="stylesheet" href="../Styles/formation.css" type="text/css" />
</head>
```

Feuille de styles

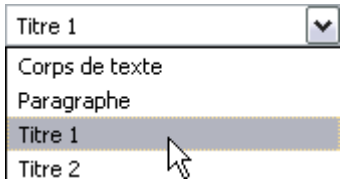
```
body {
        font-family: Arial,Helvetica,sans-serif;
        font-size: 18px;
        color: #990000;
        background-color: silver;
}
```

Modifier une règle existante

Si besoin cliquez de nouveau sur



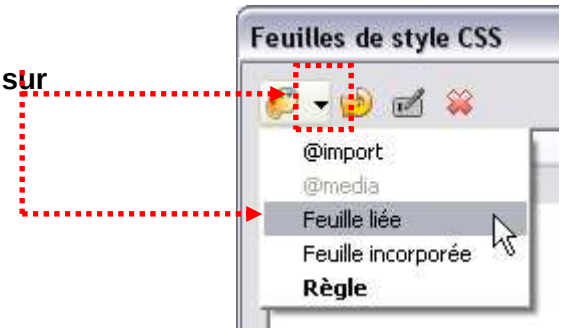
Utilisation du style avec Kompozer :

Pour la balise body	Rien à faire c'est l'aspect de la page par défaut
Pour la balise p	Faire un paragraphe
Pour les balises h1, h2, …,h6	
Pour les balises ul, ol	Créer une liste à puce, une liste numérotée

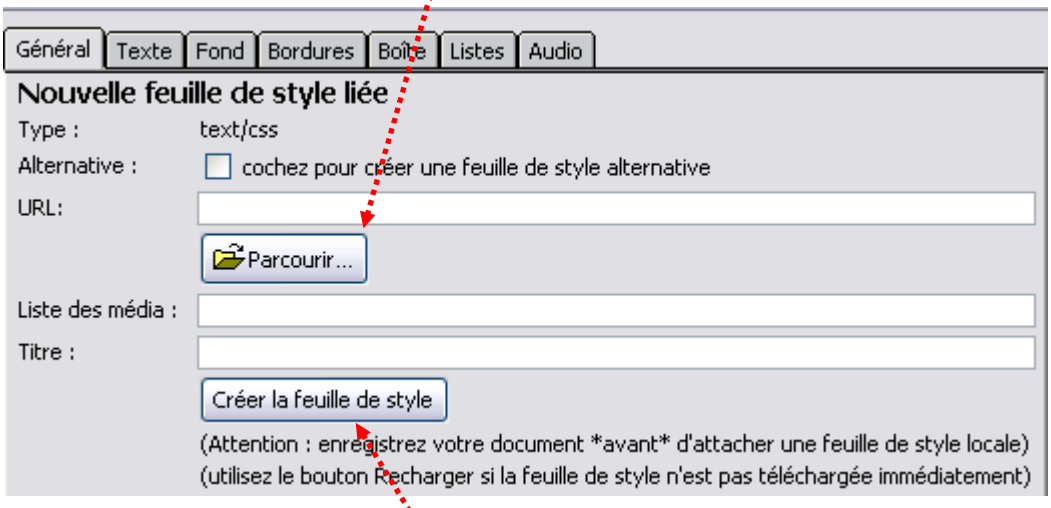
Attacher une feuille de styles

Avant tout enregistrez la page active

Cliquez sur CaScadeS **puis sur**



Parcourez l'arborescence à la recherche de la feuille de styles



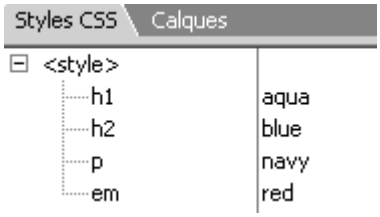
Cliquez sur

Créer la feuille de style

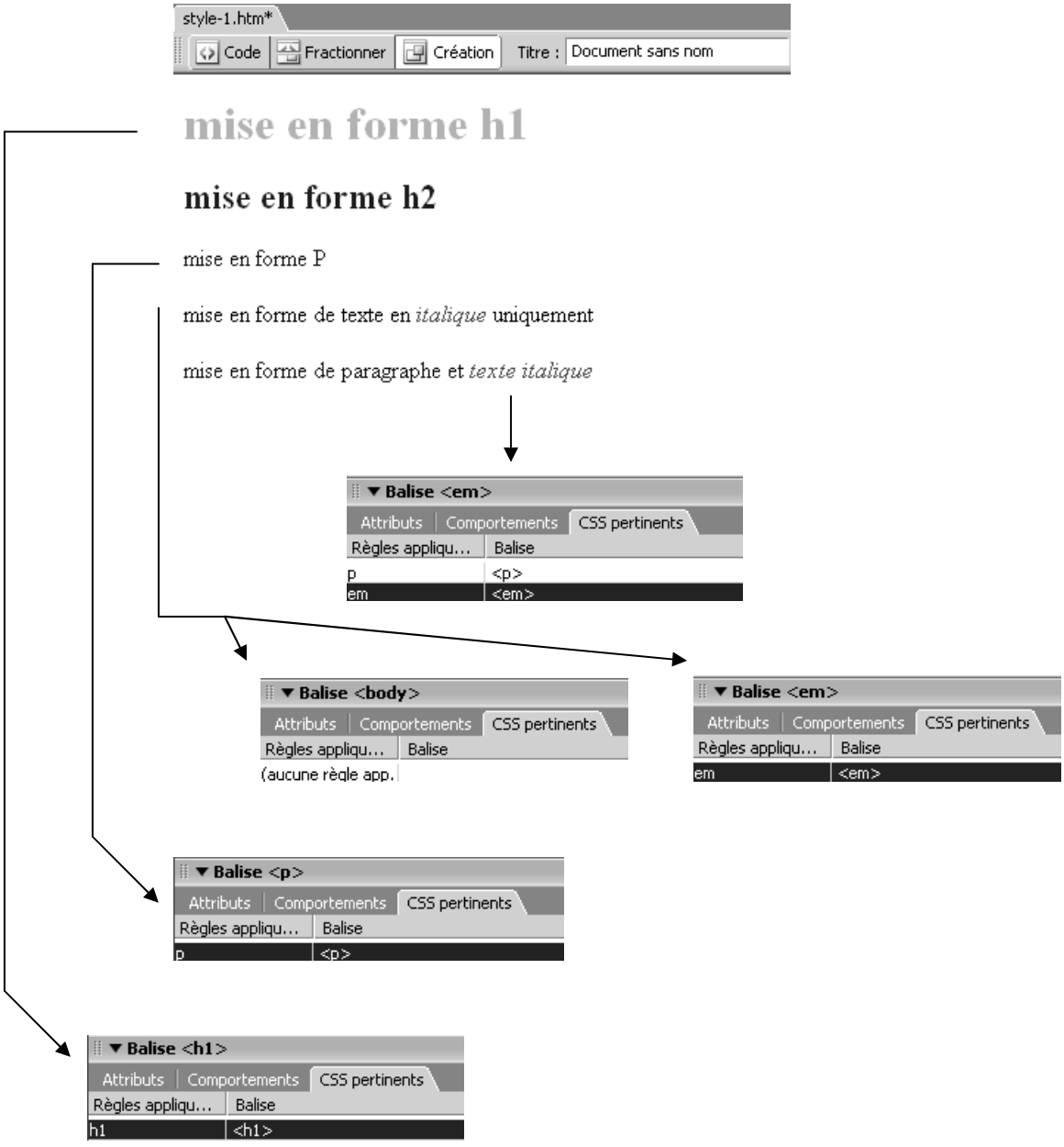
(sans commentaires…)

Superposition de styles "différents":

Travaillons un style pour les balises **h1 - h2 - p – em** :

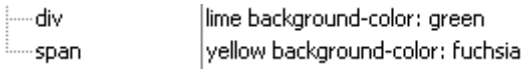


et regardons ces styles dans un fichier **style_01.htm**



Superposition avec DIV ou SPAN :

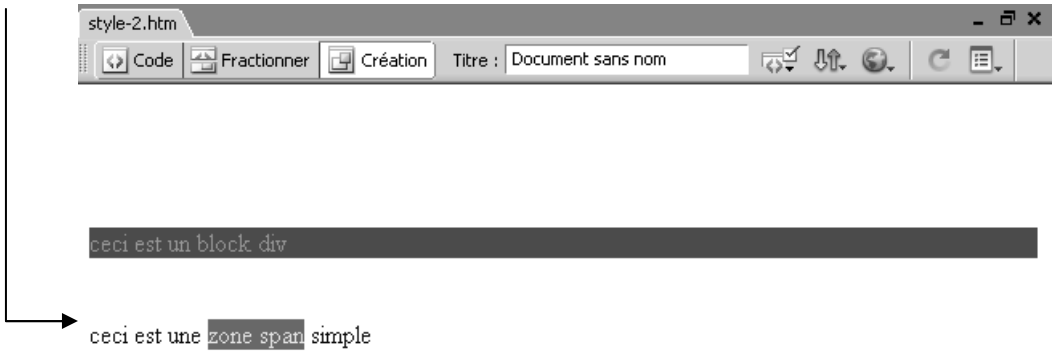
Nous allons créer deux nouveau styles de type **DIV** et **SPAN** :



DIV : balise de type Block (Fond vert foncé, caractère vert clair)

SPAN : balise de type Inline (Fond fuchsia, caractère yellow)

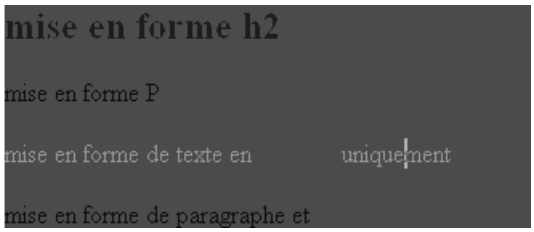
et appliquons ces styles dans **style_01b.htm** suivant

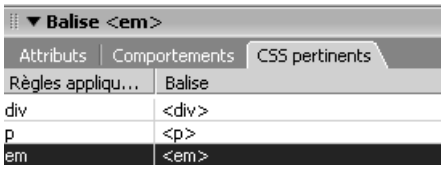


Maintenant essayons de superposer, les style donnés précédemment dans un fichier **style-01c.htm**.

Lorsque l'on applique le style DIV celui-ci doit se "rajouter" sur toute la zone car DIV est une balise de type bloc qui peut donc englober d'autres balises…

```
<h1>mise en forme h1</h1>
<div>
<h2>mise en forme h2</h2>
<p>mise en forme P</p>
mise en forme de texte en <em>italique</em> uniquement
<p>mise en forme de paragraphe et <em>texte italique</em> </p>
</div>
```





et selon ou l'on est, les style se superposent

Lorsque l'on applique le style SPAN celui-ci ne devrait pas se "rajouter" sur toute la zone car SPAN est une balise de type inline qui peut donc ne peut englober d'autres balises bloc…

```
<h1>mise en forme h1</h1>
 <span>
 <h2>mise en forme h2</h2>
 <p>mise en forme P</p>
mise en forme de texte en <em>italique</em> uniquement
<p>mise en forme de paragraphe et <em>texte italique</em> </p>
</span>
```

mise en forme h2

mise en forme P

mise en forme de texte en *italique* uniquement

mise en forme de paragraphe et *texte italique*

Surprise !

Cela à l'air de marcher ! ! mais combien de temps ?

Vérifions quelques navigateurs…

NETSCAPE 4.7

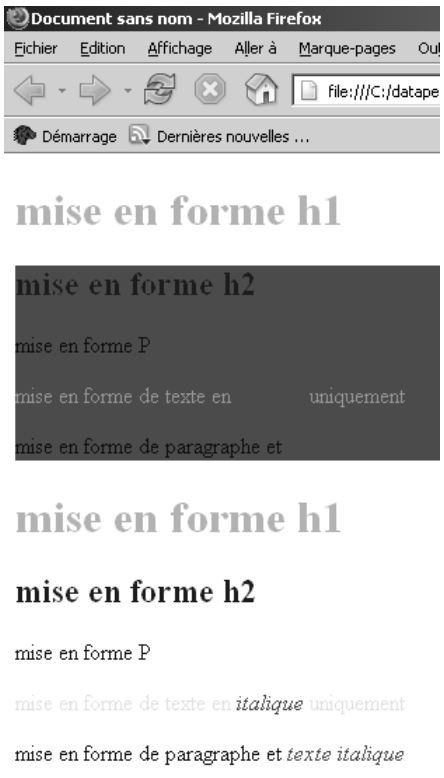
(sans commentaire)

INTERNET EXPLORER 6.0

(très permissif)

FIREFOX - OPERA

(puriste)



Il paraît clair que si on veut avoir une chance de résultat, il faut mettre la barre sur une écriture la plus rigoureuse possible, et tester sur les navigateurs ciblés… Par conséquent si on veut imbriquer un tableau ou des paragraphes à l'intérieur d'une zone stylée, il faut le faire Dans une balise **DIV**, et non pas dans une balise SPAN

Superposition avec DIV et SPAN :

Concluons ce chapitre en disant que une BALISE DIV peut contenir des zones SPAN mais que le contraire ne doit pas marcher… Dans un fichier **style-01d.htm**.

```
<div>zone div
<p>mise en forme P</p>
<span>mise en forme avec balise SPAN</span>
<p>mise en forme de paragraphe et <em>texte italique</em> </p>
fin zone div</div>
```

OK doit marcher

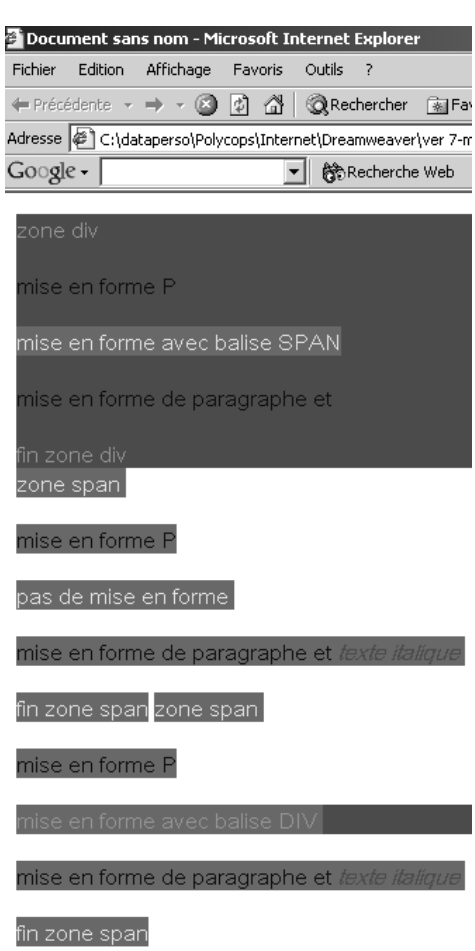
```
 <span>zone span
 <p>mise en forme P</p>
 pas de mise en forme
 <p>mise en forme de paragraphe et <em>texte italique</em> </p>
 fin zone span</span>
```

KO ne doit pas marcher

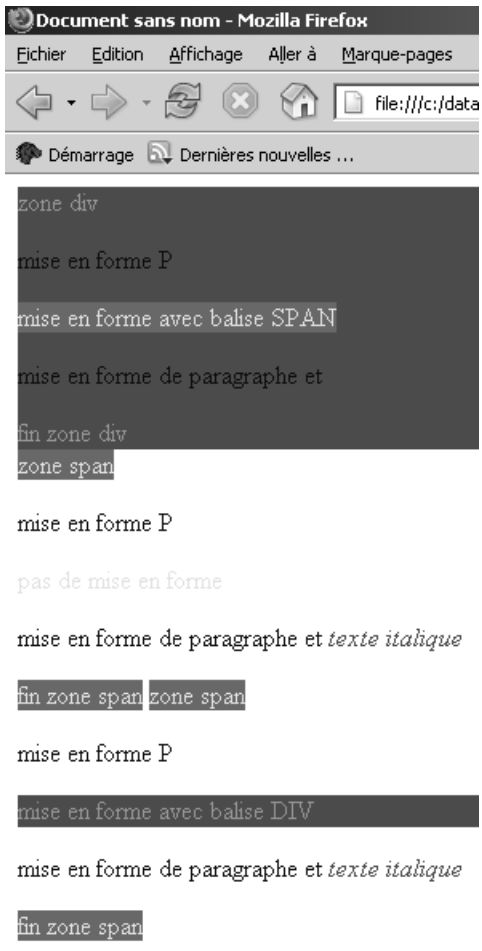
```
 <span>zone span
 <p>mise en forme P</p>
<div>mise en forme avec balise DIV </div>
 <p>mise en forme de paragraphe et <em>texte italique</em> </p>
 fin zone span</span>
```

KO ne doit pas marcher

Avec I.E.



Avec Firefox



Donc seule une écriture correcte donne une chance d'égalité sur le résultat…

LES SELECTEURS CONTEXTUELS

Qu'est ce qu'un contexte:

Les **sélecteurs contextuels** sont composés d'une suite de plusieurs **sélecteurs simples** séparés par un espace (tous les sélecteurs décrits jusqu'ici étaient des sélecteurs simples). Seuls les éléments existants qui correspondent au dernier sélecteur simple de la suite sont concernés, et ce seulement si le motif de recherche correspond.

Cela correspond donc à une suite de balises:

Soit une suite de sélecteurs de type, comme dans "*H1 suivit de em*"

Soit une suite de sélecteur de classe et sélecteur de type, comme dans "*à l'intérieur d'une zone de classe xxxxx, on trouve une balise H3*"

Sélecteur contextuel sur balise HTML :

Lorsque on énumère une série de balises HTML, on parle alors de un **sélecteur contectuel sur balise HTML** car il s'applique uniquement si on rencontre la série de balises, dans cet ordre.

```
<STYLE type="text/css">
        P em {color:blue}
</STYLE>
```

Dans un paragraphe <P>, tout texte en italique sera mis en bleu… il n'y a pas de virgule entre P em)

Création avec Kompozer - Exemple 1 :

Feuille de styles	Page Web
h1 em { color : #00 FF 00}	<h1>Titre italique en h1

Titre *italique* en h1

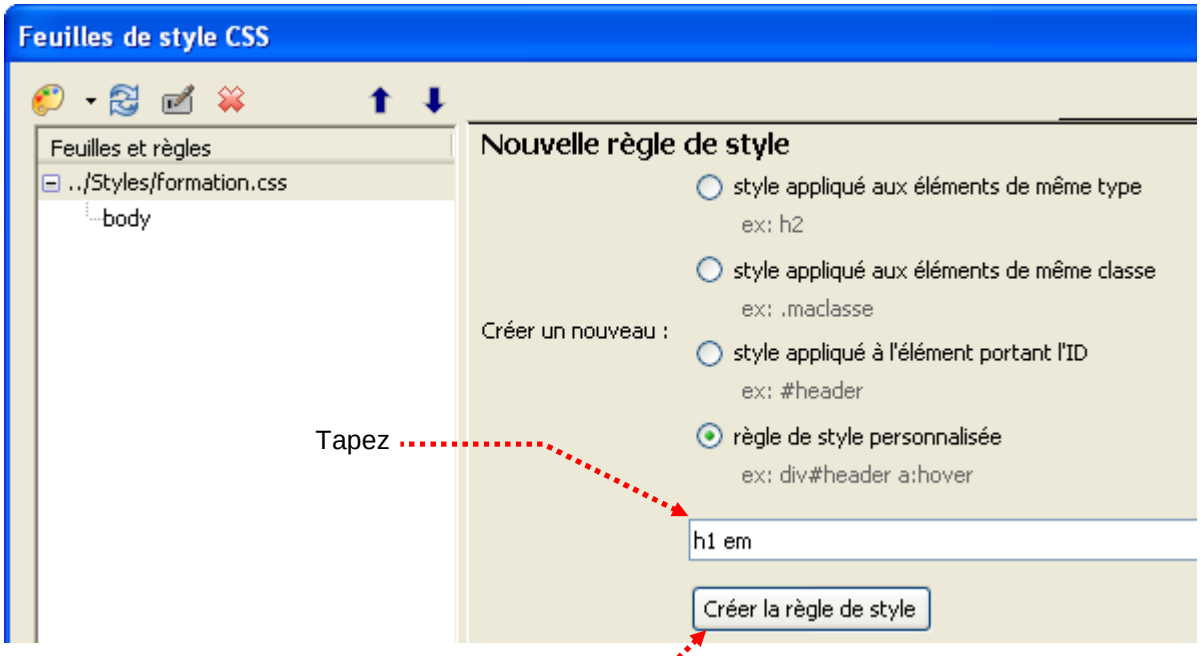
L'italique est vert lorsqu'il est utilisé dans un h1

Conséquence

Seuls les mots en italique dans un titre h1 sont en verts.

Création - Mode opératoire

Si besoin cliquez sur CaScadeS **puis**



Cliquez sur Créer la règle de style

Utilisation avec Kompozer

Ecrire le texte le mettre en h1, sélectionner une partie du texte et cliquez sur ! (ce bouton s'appelle « Mettre en évidence »).

Création avec Kompozer - Exemple 2 :

- Elément 1
- Elément 2
  - Sous élément 1
  - Sous élément 2
    - Sous sous élément 1
    - Sous sous élément 2
- Elément 3

Feuille de styles

```
ul {
        list-style-type: square;
}
ul ul{
        list-style-type: circle;
}
ul ul ul {
        list-style-type: disc;
}
```

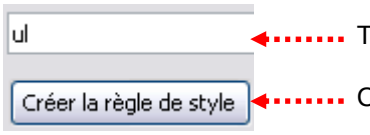
Permet de définir les différents niveaux de puces

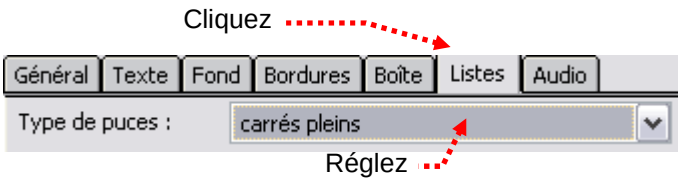
Page Web

```
<ul>
 <li>Elément 1</li>
 <li>Elément 2</li>
 <ul>
  <li>Sous élément 1</li>
  <li>Sous élément 2</li>
  <ul>
   <li>Sous sous élément 1</li>
   <li>Sous sous élément 2</li>
  </ul>
 </ul>
 <li>Elément 3</li>
</ul>
```

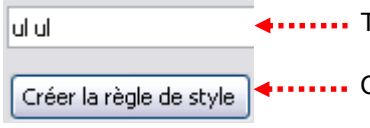
Si besoin cliquez sur CaScadeS **puis**

Il faut ensuite redéfinir ul

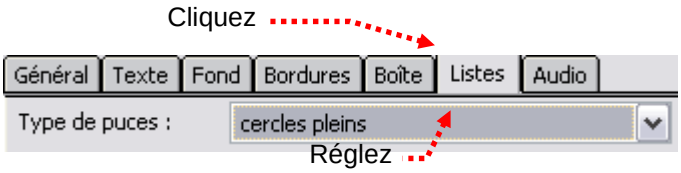




Il faut ensuite redéfinir ul ul



Il faut ensuite redéfinir ul ul ul



Utilisation avec Kompozer :

Créez une liste à puce, puis faire le décalage des niveaux par 

LES SELECTEURS PSEUDO-CLASSE

Sélecteur pseudo-classe (a:hover …):

Le sélecteur permet de spécifier une combinaison "**Balise + Evènement"**

Ce qui permet de définir les 4 situations suivantes à propos des liens hyper-texte :
sélecteur de lien a pseudo classe link - visited - hover - active

```
<STYLE type="text/css">
a:link {
        color: #FF0000;
        text-decoration: none;
}
</STYLE>
```

Essentiellement les liens hypertexte lorsque un des 4 événement pré-définis survient

NB: Si on défini les 4 pseudo liens - link - visited - hover – active, il faut les définir dans cet ordre …

Syntaxe : Ces sélecteurs doivent être définis dans cet ordre.

a {/* déclarations */}	permet de définir l'aspect commun des liens
a:link {/* déclarations */}	permet de définir l'aspect des liens non visités
a:visited {/* déclarations */}	permet de définir l'aspect des liens visités
a:hover {/* déclarations */}	permet de définir l'aspect du lien survolé
a:active {/* déclarations */}	permet de définir l'aspect du lien actif

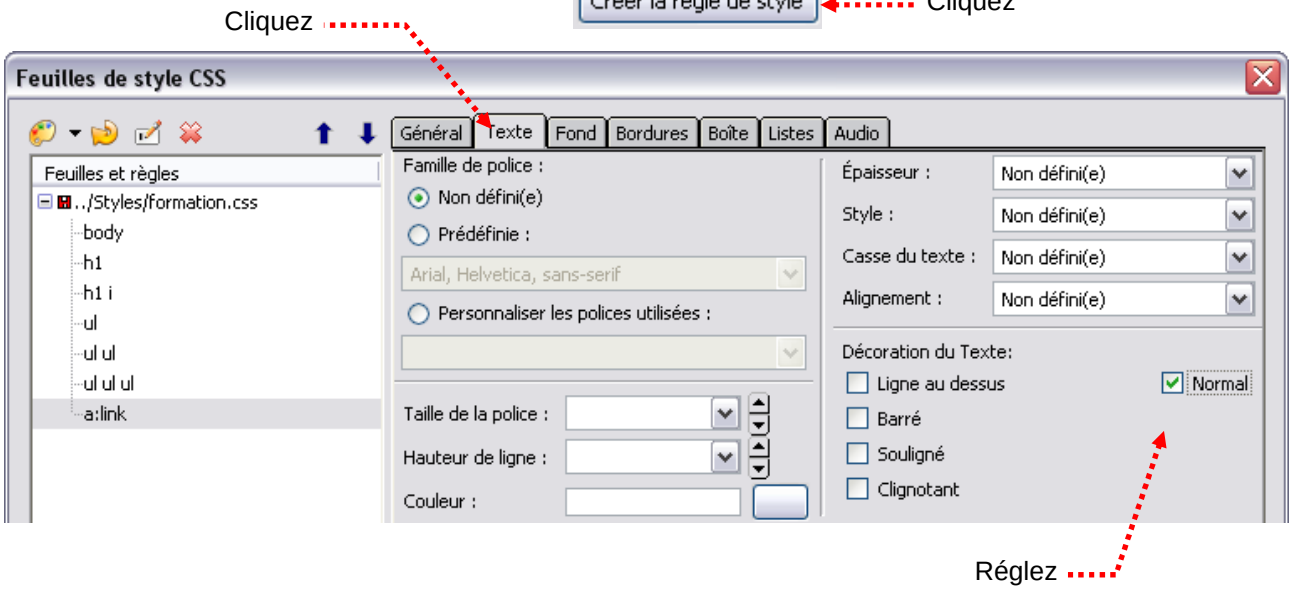
KOMPOZER & PSEUDO-CLASSE

Pseudo classes

On veut supprimer le soulignement des liens non visités

Si besoin cliquez sur CaScadeS **puis**

Il faut ensuite redéfinir a:link



Utilisation en Kompozer :

Réalisez un lien.

Pseudo classes et autres sélecteurs

En théorie, vous pouvez utiliser n'importe quel sélecteur avec ces pseudos classes

```
Balise_html:pseudo_classe {/* déclarations */}
```

Exemple

p:hover{color:red;} Vous constaterez que cela ne fonctionne pas sur IE.

LES SELECTEURS DE CLASSE

Sélecteur de classe – ("Class" à usage répété):

Une Classe permet de regrouper tous les éléments de présentation voulus et de définir la portée de la règle indépendamment de la nature HTML des balises. Dans le document on peut donner plusieurs fois cette règle.

Une Classe est définie par l'attribut **CLASS**, qui peut se donner sur quasiment toutes les balises HTML existantes.

```
<STYLE type="text/css">
        .installation {
                color: #0000FF;
                background-color: #00FFFF;
        }
</STYLE>
```

Toute zone entre des balises HTML marquée de l'attribut **Class=….**

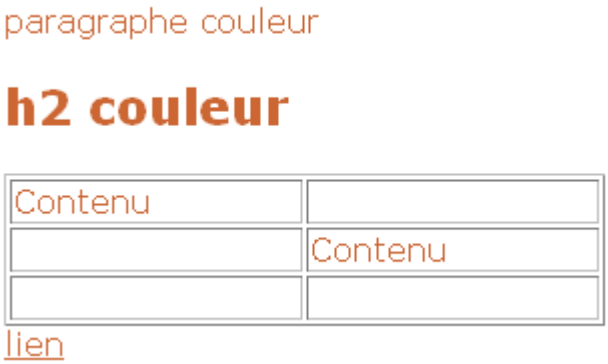
Ce style et alors applicable sur des balises très différentes

```
<h1 class="installation">titre en installation</h1>
<p class="installation">ceci est un paragraphe complet </p>
<span class="installation">d'installation</span>
<table width="80%"  border="1" class="installation">
<a href="www.go.fr" class="installation">suite de l'installation</a>
```

NB: Sont autorisées comme auparavant lettres, chiffres, underscore "_" et le tiret "– " MAIS IL DOIT COMMENCER PAR UN POINT .

NB: Noter que le point dans la définition ne se retrouve pas dans l'utilisation .

Exemple :



Feuille de styles

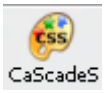
```
.couleur {
        font-family: Verdana;
        color: #CC6633;
}
```

Page Web

```
<p class="couleur">paragraphe couleur</p>
<h2 class="couleur">h2 couleur</h2>
<table width="300" border="1" class="couleur">
<a href="#" class="couleur">lien</a>
```

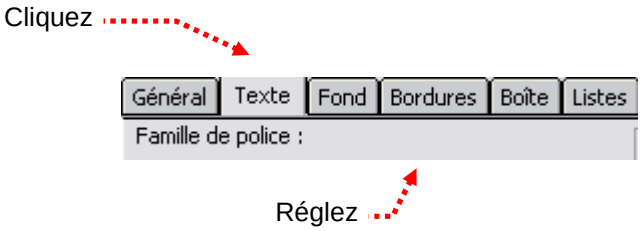
KOMPOZER & SELECTEURS DE CLASSE

Selecteur de classe

Si besoin cliquez sur 

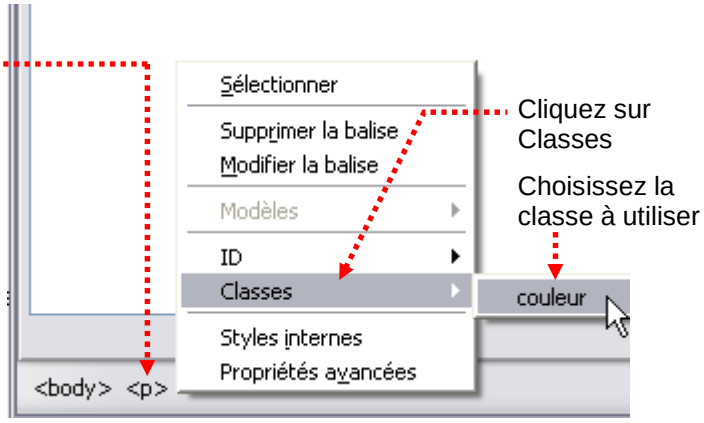
puis

Il faut ensuite créer la classe .couleur



Utilisation sur une balise précise avec Kompozer

Cliquez bouton droit (dans le sélecteur de balises) sur la balise concernée



Vous devez obtenir

```
<nom_de_la_balise class="couleur">….
```

Exemple

```
<p class="couleur">Ceci est un texte</p>
```

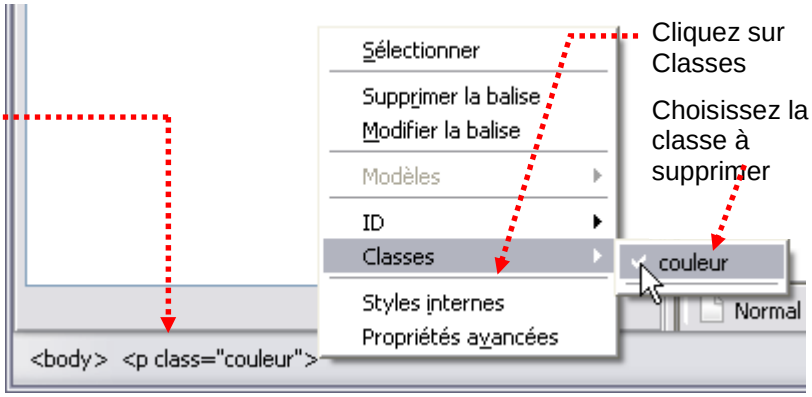
Attention :

Dans le cas d'une sélection, tous les objets sélectionnés subissent le paramètre

```
<span class= "couleur".>objet </span>
```

Supprimer l'utilisation sur une balise précise d'un style

Cliquez bouton droit (dans le sélecteur de balises) sur la balise concernée



Utilisation sur une partie de texte avec Kompozer

Sélectionnez le texte concerné

Cliquez sur

Dans le cas d'une sélection d'une sélection de texte

Vous devez obtenir	Exemple
….xxxx……	Ceci **est un **texte

Supprimer l'utilisation sur une partie de texte

Sélectionnez le texte concerné

Format>Supprimer tous les styles de texte

SELECTEURS CONTEXTUEL & CLASSE

Sélecteur contextuel Classe + Balise:

Lorsque on énumère a l'intérieur d'une classe, une balise HTML, on parle alors de un **sélecteur contectuel sur Classe** car il s'applique uniquement si on rencontre la série de balises à l'intérieur d'une zone présentée par la classe , dans cet ordre.

On peut (sans créer une Classe) créer des sélecteurs contextuels basés sur une classe…

Classe + Balise

```
.suite p {
    font-size: medium;
    color: #0033FF;
}
.suite h1 {
    font-size: medium;
    color: #0000FF;
}
.suite a:hover {
    font-size: medium;
    color: #003300;
    text-decoration: none;
}
```

```
<div class="suite">
  <p>voici du texte</p>
  <h1>voici un titre</h1>
  <p>voici un <a href="go">lien</a></p>
</div>
```

mais il ne faudrait pas être tenté d'écrire ce genre de code :

```
<p class="suite">voici du texte</p>
<h1 class="suite">voici un titre</h1>
<p>voici un <a href="go" class="suite">lien</a></p>
```

En effet l'objectif ici est de définir d'abords la classe sur une zone, puis pour une balise particulière de définir un autre effet.

exemple

Feuille de styles

```
.suite p {
        font-family: Arial;
        font-size: 12px;
        color: #FF0000;
}
.suite h1 {
        font-family: Times;
        font-size: 24px;
        color: #00FF00;
 }
.suite a:link {
        font-size: 14px;
        color: #339999;
        text-decoration: none;
}
```

Page Web

```
<div class="suite">
        <p> Ceci est un paragraphe</p>
        <h1>Ceci est un titre</h1>
        <a href="Lien"> Retour</a>
</div>
```

Ceci est un paragraphe

Ceci est un titre

Retour

Page Web

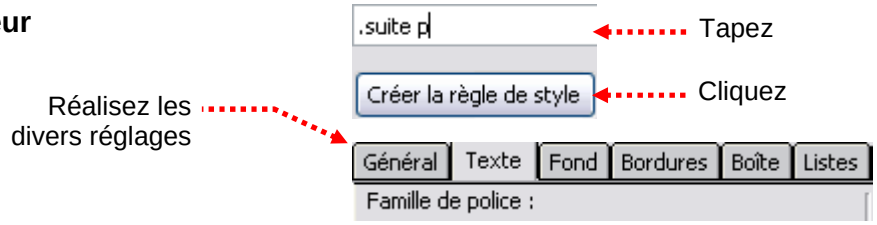
```
<table width="200" border="0" class="suite">
  <tr>
    <td><h1>fsdfdf</h1></td>
    <td><a href="lien.html">Retour</a>
</td>
  </tr>
</table>
```

fsdfdf Retour

Création du sélecteur contextuel de classe

Si besoin cliquez sur CaScadeS **puis**

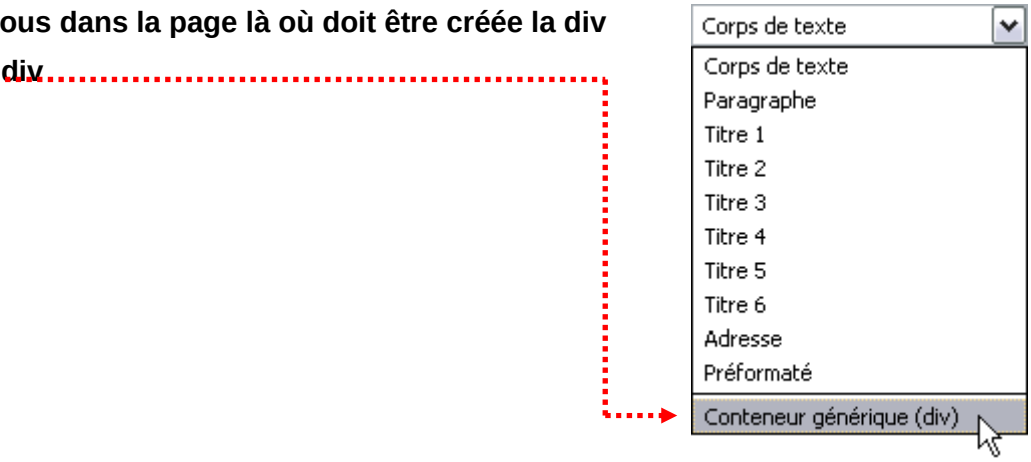
Créer le sélecteur



Utilisation sur une div - Création de la div

Placez vous dans la page là où doit être créée la div

Créer la div

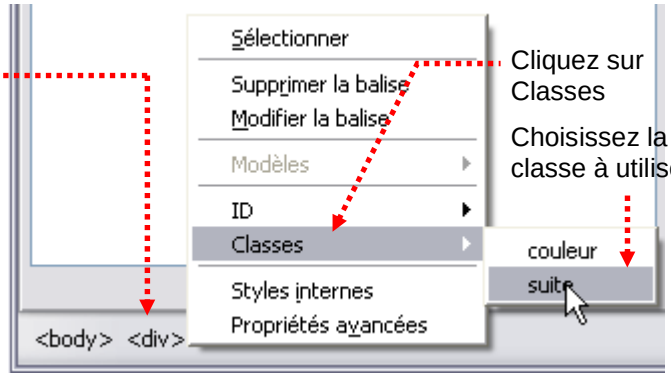


Vous obtenez une zone rouge pointillée qui représente la div créée

Attention : si vous sélectionnez plusieurs objets (plusieurs paragraphes par exemple) cela va faire une div par objet sélectionné.

Il faut donc sélectionner uniquement un élément, créer la div et mettre ensuite les autres éléments dans la div créée.

Cliquez bouton droit (dans le sélecteur de balises) sur la balise concernée

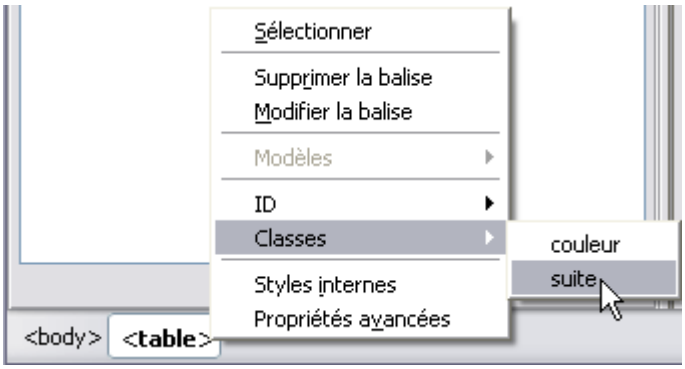


Page Web

```
<div class="suite">
   <p>Ceci est une division</p>
   <p>avec plusieurs paragraphes</p>
</div>
```

Utilisation sur un conteneur

Autre solution vous sélectionnez un objet conteneur **(body, table, tr, td…) et vous utilisez la classe**



Page Web

```
<table class="suite" border="1" cellpadding="2"
 cellspacing="2" width="200">
  <tbody>
    <tr>
      <td>
      <p>Premier</p>….
```

Sélecteur contextuel Balise + Classe:

Dans ce cas on limite l'utilisation de la classe à un sélecteur précis

Balise + Classe

```
p.autre {
    font-size: medium;
    color: #FF3300;
}
h1.autre {
    font-size: medium;
    color: #FF00FF;
}
a:hover.autre {
    font-size: medium;
    color: #00FF00;
    text-decoration: none;
}
```

```
<p class="autre">voici du texte</p>
<h1 class="autre">voici un titre</h1>
<p>voici un <a href="go" class="autre">lien</a></p>
```

mais il ne faudrait pas être tenté d'écrire ce genre de code :

```
<div class="autre">
  <p>voici du texte</p>
  <h1>voici un titre</h1>
  <p>voici un <a href="go">lien</a></p>
</div>
```

En effet l'objectif ici est de définir d'abords la balise, puis un effet particulier pour cette balise.

exemple

Feuille de styles

```
p.perso {
        font-family: Arial;
        font-size: 12px;
        color: #FF0000;
}
h1.perso  {
        font-family: Times;
        font-size: 24px;
        color: #00FF00;
 }
a.perso:link {
        font-size: 14px;
        color: #339999;
        text-decoration: none;
}
```

Page Web

```
<p class="perso">Ceci est un paragraphe</p>
<h1 class="perso">Ceci est un titre</h1>
<a href="lien.html" class="perso">Retour </a>
```

Ceci est un paragraphe

Ceci est un titre

Retour

Si on utilise sur une div (ou sur un objet contenant) ce genre de styles, ce sont les styles "normaux" qui fonctionnent.

Page Web

```
<div class="perso">
<p>Ceci est un paragraphe</p>
<h1>Ceci est un titre </h1>
<a href="lien.html" class="perso">Retour </a>
</div>
```

Ceci est un paragraphe

Ceci est un titre

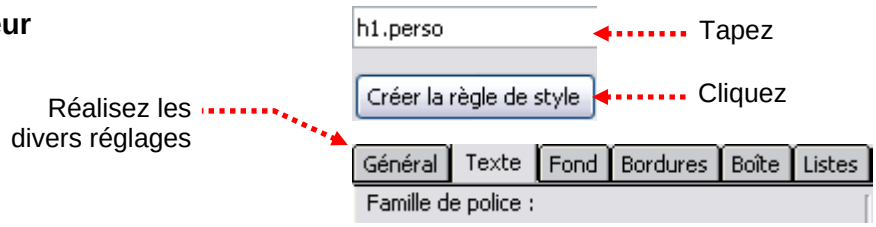
Retour

Création avec Kompozer

Si besoin cliquez sur 

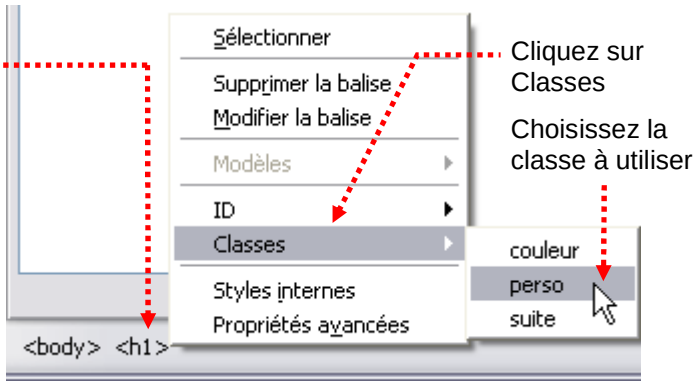
puis

Créer le sélecteur



Utilisation avec Kompozer

Cliquez bouton droit (dans le sélecteur de balises) sur la balise concernée



Eléments Bloc – Inline & marges

Sélecteurs genre Block ou Inline:

Le sélecteur (définissant la portée de la règle) peut appartenir à deux familles, soit une catégorie structure **Block**, soit une catégorie élément simple **Inline**.

Sélecteur de **classe Block** – (Bloc en français - structure) : Contient des ruptures de ligne, et peut contenir d'autres sélecteurs de type Block ou inline

- Balises HTML Concernées:
  - <HTML> <BODY>
  - <H1> <H2> … <H6>
  - <P>
  - <HR>
  - <TABLE> <FORM>
  - <DL> <DT> <DD> <UL> <OL> <LI>
  - <DIV>
  - <PRE> <BLOCKQUOTE>

Sélecteur de **classes éléments Inline** – (en ligne en français – balises simples) : Ne contient pas de ruptures de ligne et peut contenir d'autres sélecteurs uniquement de type inline (mais c'est assez rare)

- Balises HTML Concernées:
  - <A>
  -
  - <IMG>
  - <SPAN>
  - <EM> <STRONG> <I> <B> <BIG> <SMALL> <SCRIPT>

la famille BLOC - "structure" peut être positionnée (et donc à des propriétés de position… de marges…), alors que la famille INLINE - "élément simple » non.

N.B: On peut passer d'une structure BLOC à une structure INLINE (et vice-versa) grâce à la propriété CSS "**display**".

Display :block ;

Display :inline ;

Ainsi l'écriture « H1 Display :inline » évite le retour à la ligne après le titre…

Notion de boite - Bloc:

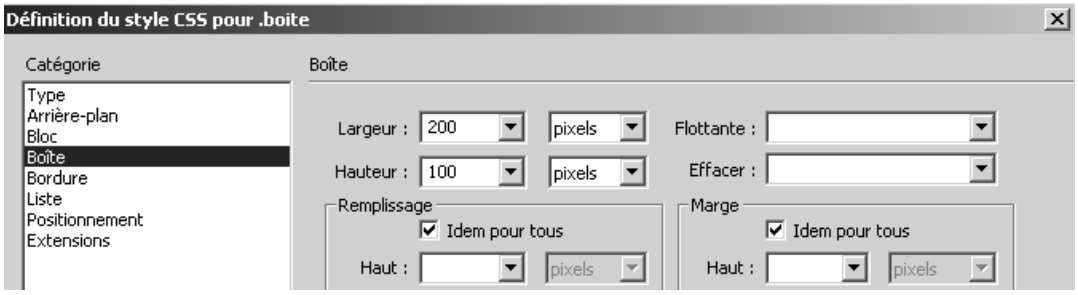
Toute balise HTML peut grâce à une définition de style simple devenir ce que l'on appelle une boite. Il suffit de lui ajouter une dimension **Width** ou **Height** pour que automatiquement l'objet HTML se transforme en Boîte.

Ainsi l'écriture

```
<H1 STYLE="width:200px;height:100px; background-color: #00FFFF"">
    titre boite
</H1>
```

suffirait…

Mais comme nous connaissons les styles, Il suffit de créer une classe et de lui donner une largeur ou une hauteur (par commodité pour mieux la voir un fond est pratique…)



Voyons dans dans **style-09.htm** quelques boîtes de définies

```
<hl class="boite">titre en boite</hl>
<p class="boite">paragraphe en boite</p>
<DIV class="boite">texte en boite</DIV>
<br>
<SPAN class="boite">texte inline en boite</SPAN>
```

N.B: Noter que seuls des style avec des sélecteur de type Bloc (et non inline) peuvent générer des Boites…

Si on tient à cela il faut utiliser dans le style **display:block**

```
.boiteforce {
    height: 100px;
    width: 200px;
    background-color: #00FFFF;
    display:block;
}
```

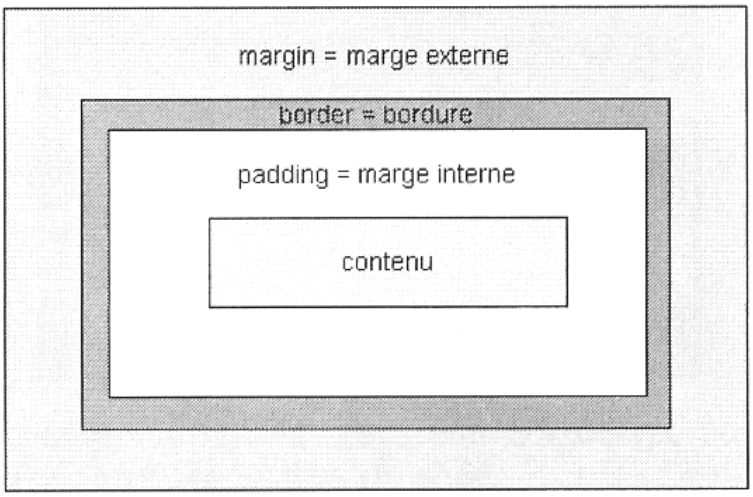
texte inline en boite

texte inline en boite force

Notion de margin - padding:

Une boite crée à forcément des marges, connues sous les propriétés **Margin** et **Padding** et une bordure, connue sous la propriété **Border**.

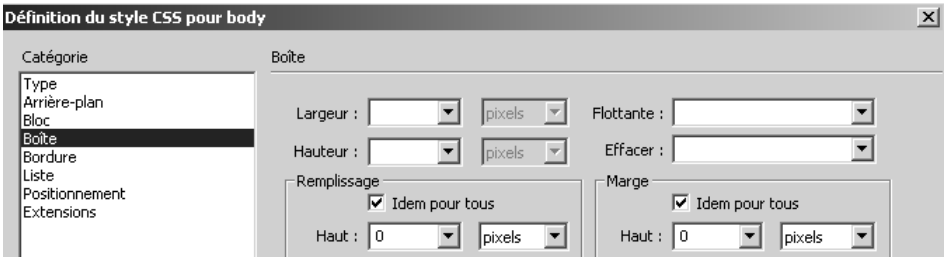
Tout cela encadre son contenu



Les valeurs **Margin** et **Padding** ne sont pas hérités, et de plus les valeurs des marges par défaut varient selon les navigateurs

- pour toutes les boites **Bloc** les marges sont non nulles par défaut (sauf celles construites à partir d'un DIV).
- Pour toutes les boites **Inline** les marges elles sont nulles par défaut.

N.B: Par conséquent il est nécessaire pour pouvoir finement travailler, de réduire les marges à 0. On va donc redéfinir les deux balises **BODY** et **P** par un style du genre ci dessous (on peut aussi traiter pour toutes les balises **h1** – **h2** etc)



Dans **style-09b.htm** on à mis a zéro les marges, et on voit que le bloc s'inscrit bien en haut a gauche du navigateur… (avant il subsistait des marges.)

Ordre de Définition Top – Right – Bottom - Left

N.B: lorsque l'on défini des tailles, le sens est : Haut – Droite – Bas – Gauche

Pour définir les tailles de la marge pour les 4 cotés identiques on peut aussi écrire:

```
p {
        margin: 0px;
}
```

Pour définir les tailles pour les 4 cotés: (haut – droite – bas – gauche)

```
p {
        margin: 0px 10px  15px  10px;
}
```

Définir les tailles pour les cotés 2 à 2: (haut – droite, bas et gauche déduites…)

```
p {
        margin: 0px 10px;
}
```

Croisement des marges verticales

Il existe un problème pour les marges verticales, (et que verticales) on dit que les marges verticales se croisent, dans le sens ou **lorsque les marges hautes et basses se rencontrent, elles se chevauchent jusqu'à ce que l'une d'elles touche le bord de l'autre élément**.

Dans **style-09c.htm**: avec marge haute de 50px

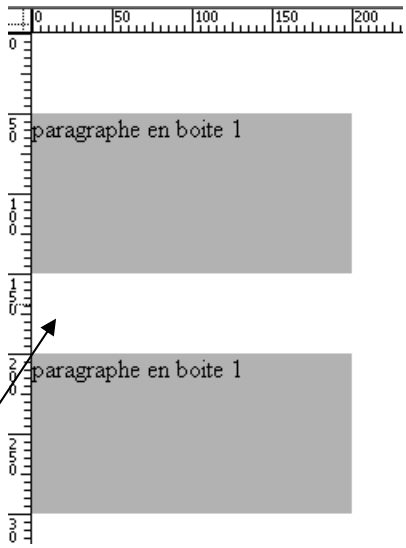
et basse de 30px

```
.boitel {
    height: 100px;
    width: 200px;
    margin: 50px 0px 30px 0px;
    background-color: #00FFFF;
}
```

ces deux paragraphes devraient être distant

de 80=50+30px

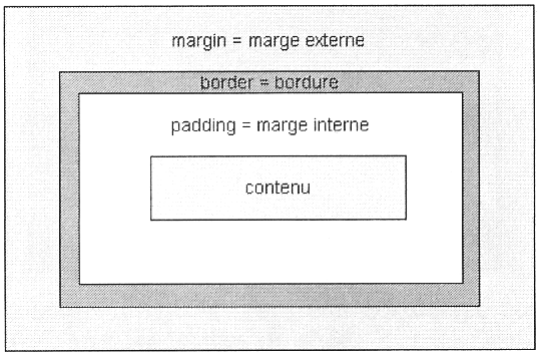
```
<p class="boitel">paragraphe en boite 1</p>
<p class="boitel">paragraphe en boite 1</p>
```



Ce qui n'est pas le cas ! cela vaut ici 50 px !

Notion de Border :

Comme on l'a dit précédemment, il existe une propriété **Border** qui possède trois propriétés :



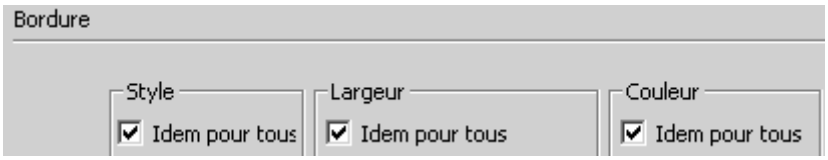
style – largeur – couleur

qui par défaut sont positionnées à **border-style=none;**

border-width = medium ;

border-color = black;

Par conséquent par défaut, il n'y a pas de bordure, or si on veut la rendre visible, ne serait-ce que pour mieux voir le Bloc, il faut savoir que cela peut modifier la mise ne page car alors elle à une épaisseur …. (a ce propos pour la rendre visible il vaut mieux utiliser la couleur de fond **background-color**)



N.B: On peut aussi utiliser une notation dites simplifiées dans le cas ou l'on veut le même effet pour les 4 cotés d'une bordure, dans ce cas les valeurs données sont dans l'ordre **style-width-color**

Dans **style-09d.htm**: on peut trouver deux écritures donnant la même chose, mais l'une sous une forme abrégée et l'autre en étendue:

```
.boitel {
    height: 100px;
    width: 200px;
    margin: 50px 0px 30px 0px;
    background-color: #00FFFF;
    border:1px solid #FF0000;
}
```

```
.boite2 {
    height: 100px;
    width: 200px;
    margin: 50px 0px 30px 0px;
    background-color: #00FFFF;
    border-width:1px;
    border-style: solid;
    border-color: #FF0000;
    }
```

N.B: lorsque l'on définis des bordures pour un Bloc, le sens classiquement est le suivant : Haut – Droite – Bas – Gauche

```
.boite3 {
    height: 100px;
    width: 200px;
    margin: 50px 0px 30px 0px;
    background-color: #FFFF00;
    border-width:1px 3px 3px 1px;
    border-style: solid;
    border-color: #00FFFF #FF00FF #FF00FF #00FFFF;
    }
```

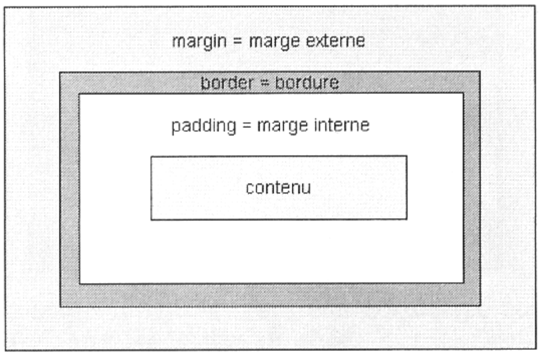
Toujours dans **style-09d.htm**

Taille de Boite – height – width :

Une boite **Bloc** à une taille, les valeurs **Width** et **Height** ne sont pas héritées, et les dimensions peuvent être données de différentes manières, mais surtout en pixels **px**

```
.boite1 {
    height: 100px;
    width: 100px;
    background-color: #00FFFF;
}
```

Le problème c'est de savoir comment est calculée avec la somme des tailles des éléments qui la compose, y compris le contenu, à savoir la propriétés **Width** (pour le contenu) à laquelle s'ajoute la propriété **Padding, Border (et margin**.)

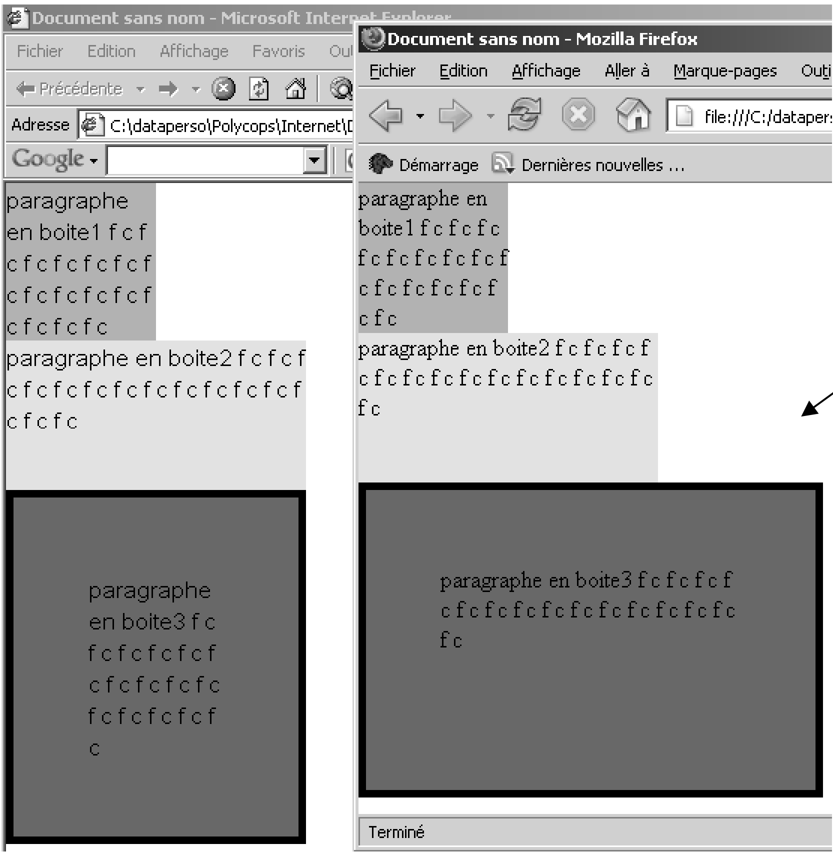


La norme défini que la "***largeur d'un BLOC***" = **Width + Padding + Border (+ margin)**

mais avec IE 5.5 la "***largeur d'un BLOC***" = **Width**

Donc pour IE **Padding** et **Border** prennent place à l'intérieur de la zone contenu

(avec IE6.0 si la page HTML contient un Doctype… alors on retrouve la norme)



style-09e.htm on voit les différences selon la norme, IE 5.5 ou 6.0 (sans doc type)

N.B: Il vaut mieux alors éviter de donner des largeurs si des bordures latérales ou padding sont données également (idem pour les hauteurs)

Position dans le Flux courant - normal:

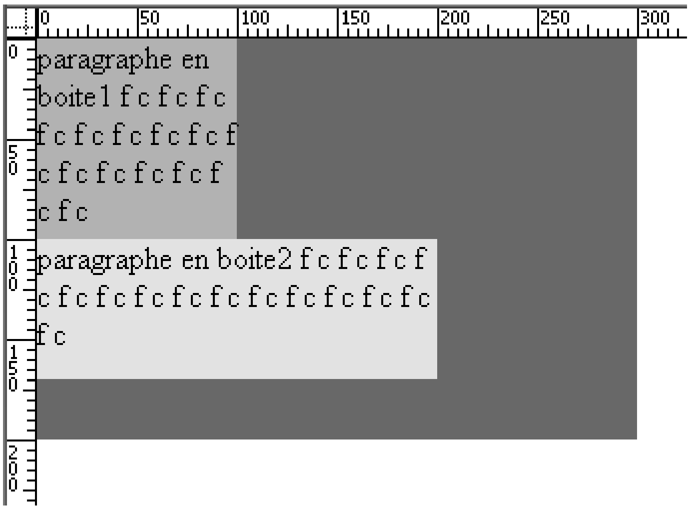
Tous les éléments en HTML se placent dans l'ordre dans lequel ils apparaissent dans la page. (l'ordre des balises HTML dans le fichier source). Par conséquent Il s'agit du positionnement "naturel" de la boite, la boite se place dans le FLUX de son conteneur, (et oui, au minimum une boite fait partie du conteneur BODY, ou plus fréquemment d'un DIV…)

- Tous les éléments de type **Bloc** se succèdent verticalement, chaque nouveau bloc se plaçant sous le précédant.
  Les **Bloc** occupent toute la largeur disponible de leur conteneur, sauf indication spécifique de dimensions.
- Tous les éléments de type **Inline** se succèdent les uns a coté des autres, avec retour à la ligne s'il n'y a plus de place disponible.

Dans **style-10.htm**

```
.cadre {
    height: 200px;
    width: 300px;
    background-color: #FF00FF;<p></p>
    border: solid 1px #000000;
}

.boite1 {
    height: 100px;
    width: 100px;
    background-color: #00FFFF;
}
.boite2 {
    height: 70px;
    width: 200px;
    background-color: #FFFF00;
}
```

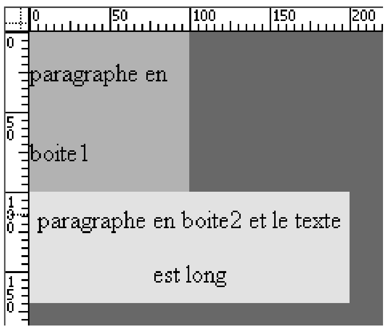


Alignement du texte dans un Bloc line-height – text-align :

Par défaut dans un bloc le texte est positionné sur le haut du bloc, si on veut donner un hauteur à une ligne de texte, alors on peut utiliser

line-height: **..px**; et dans ce cas le texte est centré en hauteur dans la ligne.

text-align:center; Pour que le texte soit centré horizontalement dans le bloc



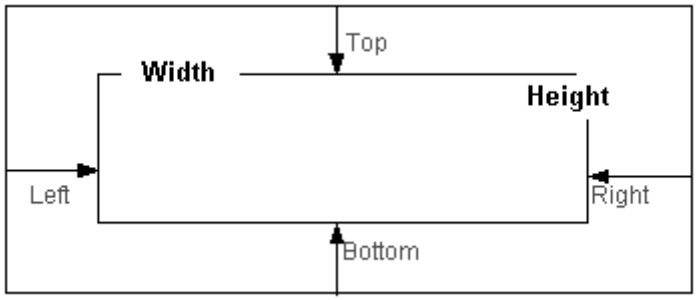
Dans **style-10a.htm**

Positionnement Relatif (dans le flux):

En plus du raisonnement dans le flux, il est possible de spécifier des propriétés telles que **position:relative; (c'est la valeur par defaut !)**

Lorsque l'on positionne une boite en relatif, elle se positionne d'abords dans le Flux courant, puis se décale par rapport à cette position selon les propriétés :

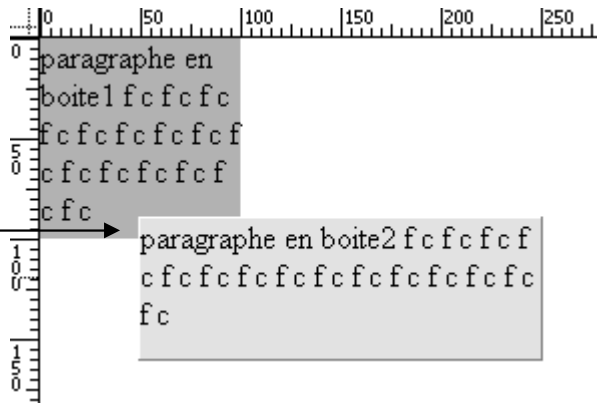
Left Top et éventuellement **Bottom et Right**



Dans **style-11.htm**

```
.boite1 {
    height: 100px;
    width: 100px;
    background-color: #00FFFF;
}
.boite2 {
    position:relative;
    bottom:10px;
    left:50px;

    height: 70px;
    width: 200px;
    background-color: #FFFF00;
}
```



Concrètement cela ne sert que si l'on veut obtenir des superpositions

Positionnement en absolu :

Il s'agit du positionnement au pixel près de la boite, qui se place dans son conteneur, (et oui, comme toujours, au minimum une boite fait partie du conteneur BODY, ou plus fréquemment d'un DIV…) donc TOUJOURS par rapport à la position de son conteneur…

N.B: Dans ce sens le conteneur doit toujours être positionné ,(que ce soit en relatif, ou en absolu, même sans paramètres de positionnement…) sinon il n'est pas pris en compte et on "remonte" sur le conteneur précédant , voire à BODY.

En spécifiant la propriété **position:absolute; l**e Bloc se positionne par rapport à son Conteneur (positionné) le plus proche grâce aux propriétés **Left Top** et éventuellement **Bottom et Right.**

Lorsque l'on positionne une boite en absolu**,** elle est retirée du flux courant **!**

Dans **style-11b.htm** on défini deux classes boite1 et boite2 en absolute :

```
.boite1 {
    height: 100px;
    width: 100px;
    background-color: #00FFFF;

    position: absolute;
    top: 25px;
    left: 25px;
}
```

```
.boite2 {
    height: 100px;
    width: 100px;
    background-color: #FFFF00;

    position: absolute;
    top: 50px;
    left: 50px;
}
```

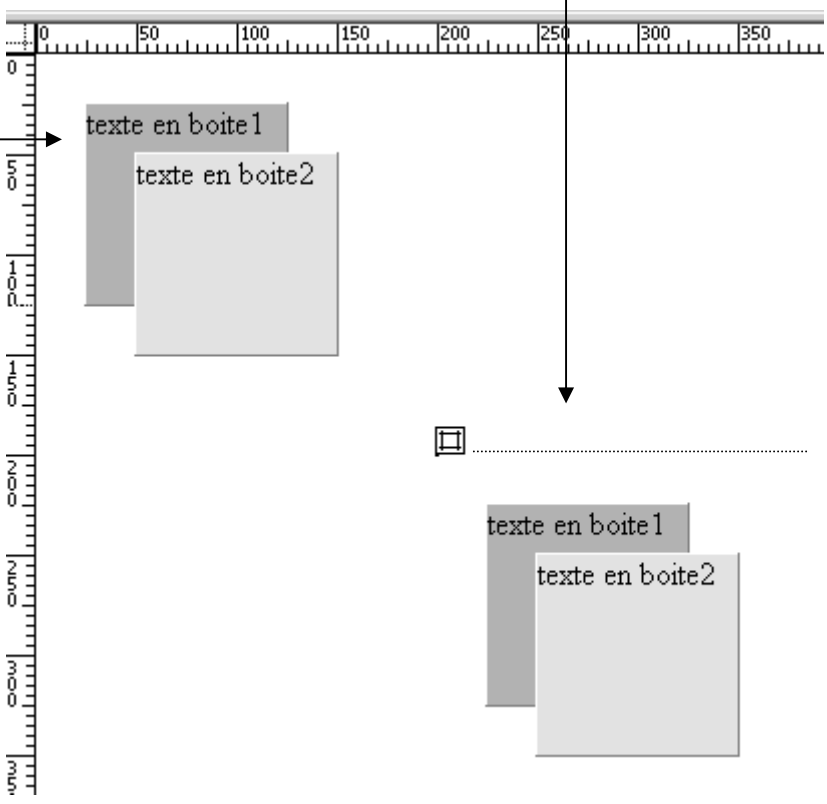
ainsi que un style decale

```
.decale {
    position: absolute;
    left: 200px;
    top: 200px;
}
```

le code suivant :

```
<body>
<p class="boite1">texte en boite1</p>
<p class="boite2">texte en boite2</p>
<DIV class="decale">
<p class="boite1">texte en boite1</p>
<p class="boite2">texte en boite2</p>
</DIV>
```

donne



Se place par rapport a body…

Se place par rapport au DIV qui est en 200px et 200px …

Dans **style-11c.htm** on défini une classe pied en **position:relative** de manière a ce que elle se place dans le flux…

```
.pied {
    position: relative;
    height: 50px;
    background-color: #00FFFF;
}
```

et on définit ensuite deux styles **copyright** et **date** en **position:absolute** destinés à placer des éléments dans la classe pied…

```
.copyright {
    position: absolute;
    bottom: 10px;
    left:10px;
    color:#0000FF;
    font-size:  10px;
}
```

```
.date {
    position: absolute;
    bottom: 10px;
    right:10px;
    color:#0000FF;
    font-size:  10px;
}
```

pour obtenir l'effet

N.B: ici l'idée est que les textes sont placée toujours de la même manière dans le pieds, indépendamment de l'endroit ou le pied se situe…

Les informations dans le pieds ne sont plus tributaires du flux, car elles sont placées en position absolue dans le pieds, par conséquent si le code suivant paraît correct

```
<Div Class=pied>
<div class=copyright>
    reservé formation
</div>
<div class=date>
    décembre 2005
</div>
</Div>
```

celui-ci l'est tout autant !

```
<Div Class=pied>
<div class=date>
    décembre 2005
</div>
<div class=copyright>
    reservé formation
</div>
</Div>
```

Alignement d'un Bloc dans une page margin : auto :

Par défaut un bloc est positionné selon la valeur donnée à margin. Si on veut qu'il se centre dans la page, ou dans le bloc qui le contient)

au lieu de donner des valeurs en pixel, genre

margin: 0px;

on va donner la valeur auto, comme dans

margin: auto;

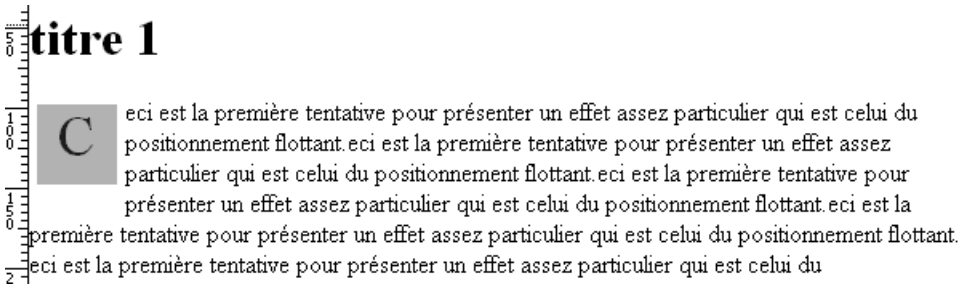
Positionnement en Flottant :

Il est possible de spécifier une propriété telles que **float:left;** ou **float:right;**

Lorsque l'on place un élément en "flottant", il est retiré du flux et se positionne sur la gauche (ou sur la droite), les éléments qui suivent se plaçent alors dans l'espace restant laissé à disposition.

Dans **style-12.htm** on défini un style **lettrine**

```
.lettrine {
    float:left;
    margin:5px;
    background-color:#00FFFF;
    color:#0000FF;
    font-size:36px;
    width:50px;
    height:50px;
    text-align:center;
}
```

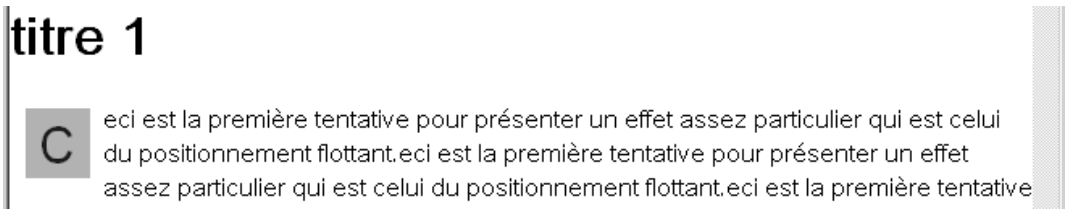


avec comme code simplement

```
<hl> titre 1</hl>
<Div Class=lettrine>
C</div>
eci est la première tentative pour présenter un effet assez particulier
```

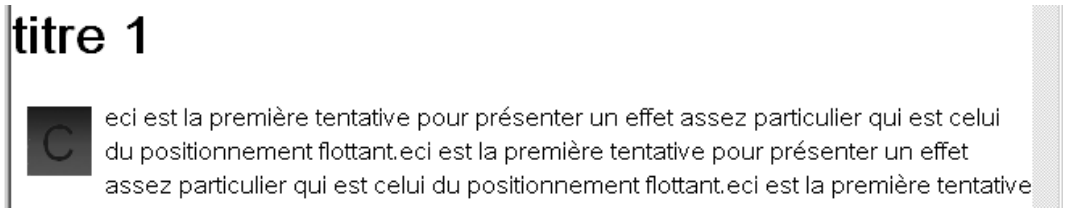
au niveau de la taille, on peut travailler plus proprement, en indiquant une taille non plus en pixel mais en ems `font-size:3em;`

Dans **style-12b.htm** on défini un style **lettrine ayant 2 lignes de hauteur…**



Dans **style-12c.htm** on défini un style **lettrine ayant 2 lignes de hauteur et une image en fond…**

```
background-image: url(style-12c.gif);
```



Gabarit de Pages :

Il est possible de créer des gabarits de pages, par exemple pour créer un site web, composé de une liste de menus sur la gauche de 150px de large, et le contenu du site sur 600 px, soit un total utile de 750 px on peut procéder de la manière suivante:

Dans **style-13.htm** on défini

```
#global {
    width: 750px;
    background-color: #66FFFF;
    margin: auto;
}
#menus {
    float: left;
    width: 150px;
    background-color: #0099FF;
    color: #FFFFFF;
}
#contenu {
    margin-left: 150px;
    background-color: #CCFF99;
}
```

← Ici on défini la taille max totale

Margin : auto permet de centrer

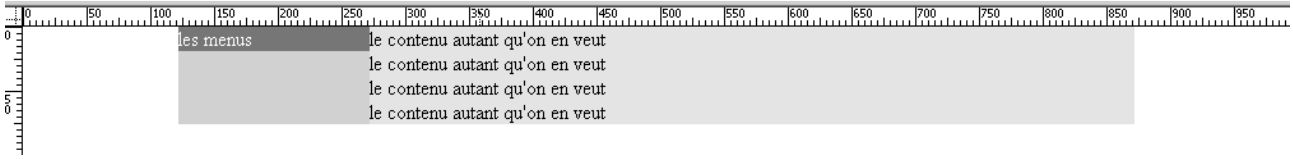
← Ici la taille des menus

← Et on laisse la place sur la gauche avant d'afficher le contenu

Ce qui avec le code suivant

```
<body>
<div id="global">
  <div id="menus">
  les menus
  </div>
  <div id="contenu">
    <p>le contenu autant qu'on en veut </p>
    <p>le contenu autant qu'on en veut </p>
    <p>le contenu autant qu'on en veut </p>
    <p>le contenu autant qu'on en veut </p>
  </div>
</div>
</body>
```

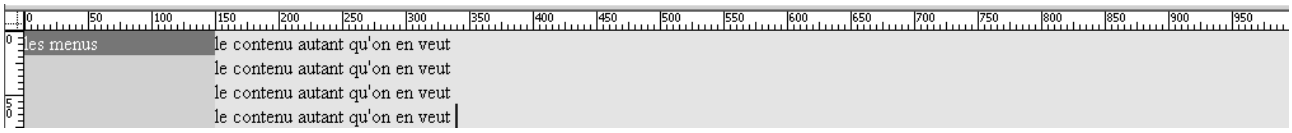
donne le résultat



Dans **style-13b.htm** il suffit de modifier (en enlevant la largeur…à)

```
#global {
    background-color: #66FFFF;
}
```

pour obtenir un contenu qui prends toute la largeur restante …



Bloc « non visible » display - visibility:

La propriété **display** accepte un argument **none** (en plus de inline ou bloc) permettant de spécifier qu'elle ne doit pas être affichée.

Le bloc est retiré du flux et n'est pas affiché !

display:none;

Une autre méthode pour ne pas afficher un bloc consiste à utiliser la propriété **visibility** avec comme argument **hidden**

La différence avec display:none réside dans le fait que le bloc n'est pas retiré du flux, il occupe toute sa place, simplement il n'est pas visible…

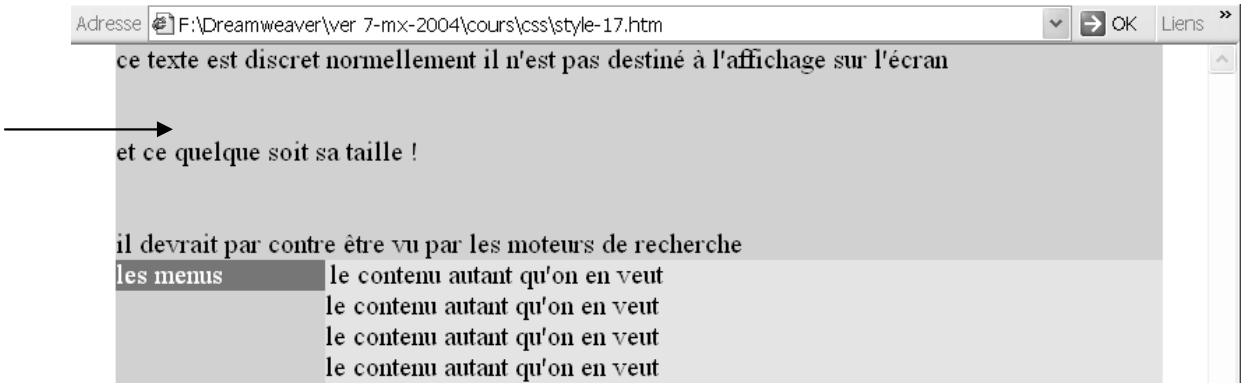
visibility:hidden;

dans le fichier **style-17.htm** un bloc ne s'affiche pas mais prends sa place

Il est donc possible de créer un bloc qui n'apparaisse pas à l' écran… et cela sans que sa place soit prise,

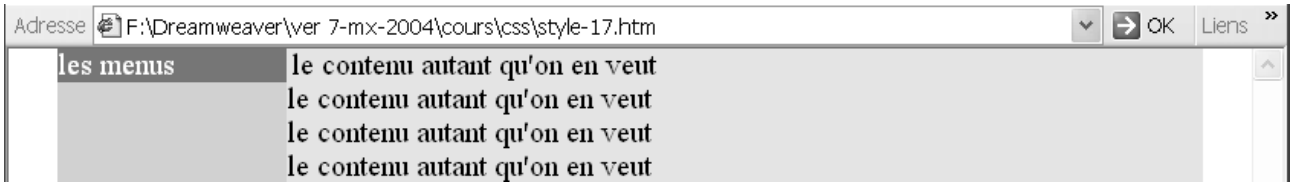
Donc 3 écritures possibles	ou	ou
position : absolute **visibility: hidden;**	**display : none**	**visibility: hidden;**
en absolu, (donc retiré du flux) et caché	retiré du flux et caché	dans le flux mais caché

Comme dans le fichier **style-17b.htm**



Cela peut surprendre, mais cela permet par exemple de rajouter du texte dans la page , à des fins de référencement, sans le faire afficher…

alors on obtient



Gestion des impressions

Spécifier le périphérique @media:

La commande

@media xxx {

}

Permet de définir de quelle manière les blocs doivent être traités lorsqu'ils sont envoyés vers le périphérique xxx.

La liste des périphérique est longue, mais très mal implémentée selon les différents navigateurs… Concrètement, on peut utiliser

media = "all" s'applique à tous les périphériques

équivalent à ne rien spécifier, car c'est la valeur par défaut.

media = "print" s'applique aux sorties papier

équivalent à ne rien spécifier, car c'est la valeur par défaut.

media = "handheld" s'applique aux navigateurs de poche

encore mal implémenté….

Prévoir une impression:

Par conséquent un bloc que l'on ne souhaiterais pas imprimer pourrait se voir affecter du style suivante :

```
#menu{
    Display : none ;
}
```

Et un bloc que l'on souhaiterais imprimer pourrait se voir affecter du style suivante :

```
#contenu{
    Position : relative ;
    Margin :0px ;
    Padding :0px
    Width :16cm
    Height:auto
    Top : 0px; right : 0px; bottom : 0px; left : 0 px;
    Border :none;
}
```

Tout cela peut se mettre dans une feuille se style spécifique, style-imprime.css …

dans le fichier **style-18.htm**

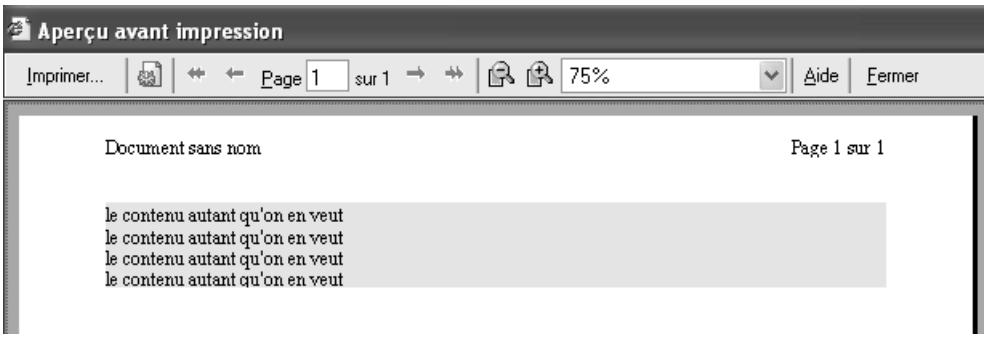
Les styles sont les suivants

```
#menus {
    float: left;
    width: 150px;
    background-color: #0099FF;
    color: #FFFFFF;
}
#contenu {
    margin-left: 150px;
    background-color: #CCFF99;
}
```

On prévoit une redéfinition de style en cas d'impression,

```
@media print
{
 #menus {
    display : none;
 }
 #contenu {
    margin-left: 0px;
}
```

Ce qui peut se vérifier en demandant un **aperçu avant impression…** :



Il est même possible de spécifier la présence d'un saut de page avant (ou après) l'élément ou cette instruction se présente…:

Page-break-before: always

Page-break-after: always

N.B: il s'agit bien sur d'un style embarqué dans une balise HTML comme dans le fichier **style-18b.htm**

<p style="page-break-before:always">