

HTML5



CSS Tutorial

LES BASES EN CSS

CSS Tutorial

SELECTEURS ET PROPRIETES CSS

Les sélecteurs CSS simples

- Un sélecteur permet de cibler un ou plusieurs éléments HTML afin de leur appliquer un style particulier.
- Il existe différents types de sélecteurs qu'on appelle **sélecteurs « simples »** ou **« complexes »**.

```
p{  
  color: blue;  
  font-size: 14px;  
}
```

- Pour tester le code : <http://jsbin.com/gopega/27/edit?html,css,output>

CSS Tutorial

OÙ ECRIRE LE CODE CSS ?

Ecrire le CSS dans un élément HTML style

Il est possible d'écrire le CSS à trois endroits différents.

- Tout d'abord: dans un élément HTML style

```
<!DOCTYPE html>
<html>
  <head>
    <title>Où écrire le CSS ?</title>
    <meta charset= "utf-8">

    <style>
      body{
        background-color: orange;
      }
      p{
        color: blue;
        font-size: 16px;
      }
    </style>
  </head>

  <body>
    <h1>Un titre de niveau 1</h1>
    <p>Un paragraphe</p>
    <p>Un deuxième paragraphe</p>
  </body>
</html>
```

Ecrire le CSS dans la balise ouvrante des éléments HTML

- On peut également écrire notre code CSS à l'intérieur de la balise ouvrante de nos éléments HTML. On utilise dans ce cas cette fois-ci un attribut style (à ne pas confondre avec l'élément style).

```
<!DOCTYPE html>
<html>
  <head>
    <title>Où écrire le CSS ?</title>
    <meta charset= "utf-8">
  </head>

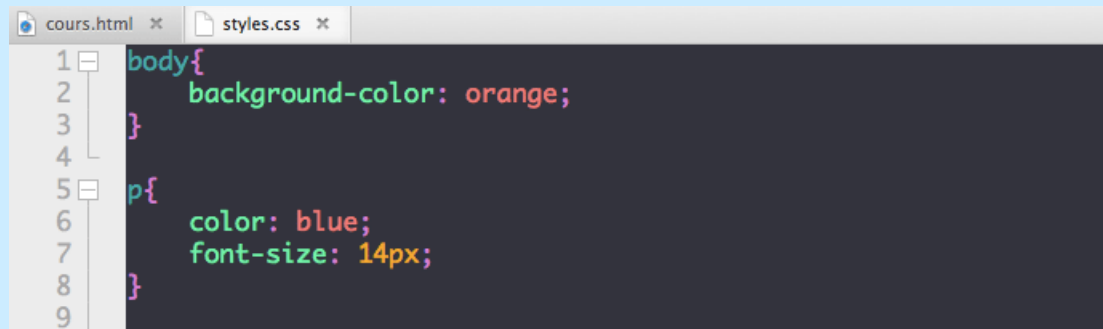
  <body style="background-color:orange;">
    <h1>Un titre de niveau 1</h1>
    <p style="color: blue;font-size: 16px;">Un paragraphe</p>
    <p>Un deuxième paragraphe</p>
  </body>
</html>
```



- Pour tester le code : <http://jsbin.com/gopega/29/edit?html,css,output>

Ecrire le CSS dans un fichier CSS séparé

- Enfin, on peut écrire le code CSS dans un fichier séparé portant l'extension « .css ». C'est la méthode recommandée, qui sera utilisée autant que possible.



The image shows a code editor window with two tabs: 'cours.html' and 'styles.css'. The 'styles.css' tab is active, displaying the following CSS code:

```
1 body{  
2   background-color: orange;  
3 }  
4  
5 p{  
6   color: blue;  
7   font-size: 14px;  
8 }  
9
```


Ecrire le CSS dans un fichier CSS séparé

- Pour lier le fichier CSS au fichier HTML il faut utiliser un nouvel élément HTML : l'élément **link**.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Où écrire le CSS ?</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1>Un titre de niveau 1</h1>
    <p>Un paragraphe</p>
    <p>Un deuxième paragraphe</p>
  </body>
</html>
```



- Pour tester le code : <http://jsbin.com/sipujup/13/edit?html,css,output>

CSS Tutorial

COMMENTAIRES ET INDENTATION EN CSS

Les commentaires

- Commenter le code CSS n'est pas une option : cela va très vite devenir indispensable car on se rendra vite compte que les fichiers CSS s'allongent très rapidement.
- La syntaxe des commentaires en CSS est totalement différente de celle des commentaires en HTML.

```
/*Je suis un commentaire*/  
body{  
    background-color: orange;  
}  
  
/*Un deuxième  
 *commentaire,  
 *multi-lignes*/  
p{  
    /*color: blue;*/  
    font-size: 14px;  
}
```

L'indentation en CSS

- Indenter en CSS est également très important afin de conserver le plus de clarté possible dans son code et de paraître professionnel si un jour vous devez le distribuer.
- En terme de règles, on indentera en général d'une tabulation les différentes déclarations concernant un sélecteur donné.
- Pour plus de lisibilité, on retournera également à la ligne après chaque déclaration.

CSS Tutorial

SELECTEURS CSS SIMPLES

Les sélecteurs de type élément

- Ex : p et h1

```
<!DOCTYPE html>
<html>
  <head>
    <title>Où écrire le CSS ?</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1>Un titre de niveau 1</h1>
    <p>Un paragraphe contenant un <a href="http://wikipedia.org">lien</a></p>
    <p>Un deuxième paragraphe</p>
  </body>
</html>
```

Les sélecteurs de type élément

- Ex : p et h1

```
/*Notre titre h1 va s'afficher en rouge*/  
h1{  
    color: red;  
}  
  
/*Nos paragraphes seront bleus*/  
p{  
    color: blue;  
}  
  
/*Le texte de nos liens sera vert et gras*/  
a{  
    color: green;  
    font-weight: bold;  
}
```



- Pour tester le code : <http://jsbin.com/sipujup/14/edit?html,css,output>

Appliquer un style à différents types d'éléments

- h1 et p groupés

```
<!DOCTYPE html>
<html>
  <head>
    <title>L'héritage en CSS</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1>Un titre de niveau 1</h1>
    <p>Un paragraphe</p>
    <p>Un autre paragraphe</p>
    <p>Un troisième paragraphe</p>
  </body>
</html>
```


Appliquer un style à différents types d'éléments

- h1 et p groupés

```
h1, p{  
  color: blue;  
}
```



- Pour tester le code : <http://jsbin.com/gopega/38/edit?html,css,output>

CSS Tutorial

LES ATTRIBUTS ID ET CLASS

Les attributs id et class et les sélecteurs associés

- Les sélecteurs #id et .class permettent de cibler un élément en particulier plutôt qu'un type d'élément.
- Les valeurs indiquées pour les attributs id et class ne doivent contenir ni caractères spéciaux ni espaces et commencer par une lettre.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Où écrire le CSS ?</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1>Un titre de niveau 1</h1>
    <p id="p1">Un paragraphe contenant</p>
    <p class="p1">Un deuxième paragraphe</p>
    <p>Un troisième paragraphe</p>
  </body>
</html>
```

Les attributs id et class et les sélecteurs associés

- Utilisation:

```
/*L'élément portant l'id "p1" sera en bleu*/  
#p1{  
    color: blue;  
}  
  
/*L'élément portant la class "p1" sera en rouge*/  
.p1{  
    color: red;  
}
```



- Pour tester le code : <http://jsbin.com/sipujup/15/edit?html,css,output>

Class et id : différences et utilisation avancée

- Une différence notable existe entre les deux attributs class et id : chaque id doit avoir une valeur unique tandis que plusieurs attributs class peuvent posséder la même valeur.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Sélecteurs CSS</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <!--On donne la même class à notre titre et à un paragraphe-->
    <h1 class="p1">Un titre de niveau 1</h1>
    <p id="p1">Un paragraphe contenant</p>
    <p class="p1">Un deuxième paragraphe</p>
    <p>Un troisième paragraphe</p>
  </body>
</html>
```

Class et id : différences et utilisation avancée

- Utilisation :

```
/*L'élément portant l'id "p1" sera en bleu*/  
#p1{  
    color: blue;  
}  
  
/*Les éléments portant la class "p1" seront en rouge*/  
.p1{  
    color: red;  
}
```



- Pour tester le code : <http://jsbin.com/sipujup/16/edit?html,css,output>

Class et id : différences et utilisation avancée

- On va également pouvoir attribuer un attribut class et un attribut id à un même élément HTML afin de pouvoir lui appliquer un style « semi-général » et un style particulier.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Sélecteurs CSS</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1 >Un titre de niveau 1</h1>

    <!--Ce paragraphe va être mis en forme par #p1 et .p1-->
    <p id="p1" class="p1">Un paragraphe contenant</p>
    <p class="p1">Un deuxième paragraphe</p>
    <p>Un troisième paragraphe</p>
  </body>
</html>
```

Class et id : différences et utilisation avancée

- Utilisation :



- Pour tester le code : <http://jsbin.com/sipujup/17/edit?html,css,output>

Class et id : différences et utilisation avancée

- Noter qu'en cas de conflit, c'est le style du sélecteur le plus précis qui va être appliqué.
- Par exemple, si vous donnez une class et un id à un même élément HTML et que vous utilisez cette class et cet id pour définir deux couleurs différentes, c'est la couleur de l'id qui va s'imposer car l'attribut id permet de cibler du code HTML plus précisément que l'attribut class.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Sélecteurs CSS</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1 >Un titre de niveau 1</h1>

    <!--En cas de conflit, l'id est prioritaire sur la class-->
    <p id="p1" class="p1">Un premier paragraphe</p>

    <p class="p1">Un deuxième paragraphe</p>
    <p>Un troisième paragraphe</p>
  </body>
</html>
```

Class et id : différences et utilisation avancée

- (Suite)



- Pour tester le code : <http://jsbin.com/sipujup/18/edit?html,css,output>

Class et id : différences et utilisation avancée

- Bien sûr, toutes les propriétés CSS ne s'appliquent pas à tous les éléments HTML. Par exemple, cela n'aurait pas de sens de donner une propriété **color** à un élément **br** puisque celui-ci ne représente pas un contenu visible par l'utilisateur

CSS Tutorial

L'HERITAGE EN CSS

Qu'est-ce que l'héritage ?

- L'héritage est une notion centrale et fondamentale en CSS.
- **La notion d'héritage signifie que tout élément HTML enfant va hériter, « en cascades », des styles de ses parents.**
- C'est par ailleurs de là que vient le nom du CSS : Cascading StyleSheets, ou Feuilles de Style en Cascades.

```
<!DOCTYPE html>
<html>
  <head>
    <title>L'héritage en CSS</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1>Un titre de niveau 1</h1>
    <p>Un paragraphe avec du <strong>texte important</strong></p>
    <p>Un autre paragraphe</p>
  </body>
</html>
```

Qu'est-ce que l'héritage ?

- Par exemple, tous les éléments à l'intérieur de l'élément body sont des enfants de cet élément. Si l'on applique un style à l'élément body, ses enfants en hériteront automatiquement.
- Ici, les éléments h1, p et strong ont hérité des styles de leur parent body. Ainsi, nos différents textes s'affichent en violet.

```
body{  
  color: purple;  
}
```



Priorité et ordre en CSS

- Cette notion d'héritage est très puissante et très pratique en CSS.
- Cependant, il convient de bien comprendre dans quel ordre et selon quelle priorité s'appliquent les différents styles à un élément afin de ne pas faire d'erreur.
- Exemple :

```
<!DOCTYPE html>
<html>
  <head>
    <title>L'héritage en CSS</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1>Un titre de niveau 1</h1>
    <p>Un paragraphe avec du <strong>texte important</strong></p>
    <p>Un autre paragraphe</p>
    <p id="para3">Un troisième paragraphe</p>
  </body>
</html>
```

Priorité et ordre en CSS

- (Suite)



- Pour tester le code : <http://jsbin.com/sipujup/19/edit?html,css,output>

Priorité et ordre en CSS

- Comment interpréter l'exemple ci-dessus ? Tout d'abord, notez que nous appliquons à chaque fois la même propriété color à nos différents éléments avec des valeurs différentes. Il va donc y avoir conflit.
- On applique une couleur violette à notre élément body. Ainsi, tous les éléments contenus dans body vont hériter de cette couleur par défaut sauf si une autre couleur est définie entre temps.
- C'est le cas pour les éléments de type p, auxquels on attribue une couleur bleue. Ici, nous avons donc un premier conflit : nos paragraphes doivent-ils hériter des styles définis avec le sélecteur body ou de ceux définis avec le sélecteur p ?

Priorité et ordre en CSS

- La solution est simple : le sélecteur `p` cible de manière plus précise les paragraphes que le sélecteur `body`, et le style défini dans le sélecteur `p` est donc plus proche de nos paragraphes que celui défini dans `body`; c'est donc bien celui-ci qui sera appliqué.
- Par défaut, tous nos paragraphes seront donc bleus, sauf si un style encore plus proche leur est appliqué. C'est le cas pour notre paragraphe comportant la `class="para3"`.
- Comme le ciblage est plus précis, les éléments possédant cet attribut `class` vont récupérer les styles de l'attribut `class` s'il y a conflit.
- Pour le texte dans l'élément `strong`, celui-ci récupère le style le plus proche de lui, c'est-à-dire celui qui lui est appliqué directement via le sélecteur `strong`.

Quel ordre par rapport à l'emplacement du CSS ?

De même, il va y avoir un ordre de priorité selon où on va écrire notre code CSS, puisque – rappelons-le – on peut l'écrire à trois endroits différents :

- Au sein de la balise ouvrante de chaque élément HTML, dans un attribut style ;
- Dans un élément style placé dans l'élément head d'un fichier HTML;
- Dans un fichier CSS séparé.

```
<!DOCTYPE html>
<html>
  <head>
    <title>L'héritage en CSS</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
    <style>
      h1{
        color: purple;
      }
      p{
        color: red;
      }
    </style>
  </head>

  <body>
    <h1>Un titre de niveau 1</h1>
    <p>Un paragraphe avec du <strong>texte important</strong></p>
    <p>Un autre paragraphe</p>
    <p>Un troisième paragraphe</p>
  </body>
</html>
```

Quel ordre par rapport à l'emplacement du CSS ?

- Si on écrit le CSS à différents endroits, et si il y a un conflit, le style appliqué à l'élément directement (grâce à l'attribut style) sera prioritaire
- Ensuite, ce sera le style écrit dans l'élément HTML style qui sera retenu.
- Finalement, le style écrit dans un fichier CSS séparé sera retenu en dernier.
- Noter qu'on pourra outrepasser cette notion d'ordre en ajoutant la mention "!important" à la fin d'une déclaration CSS qu'on veut imposer.

```
h1{  
    color: green;  
}  
  
/*On force la priorité du style*/  
p{  
    color: blue !important;  
}
```

Quel ordre par rapport à l'emplacement du CSS ?

- (Suite)

```
h1{  
  color: green;  
}  
  
/*On force la priorité du style*/  
p{  
  color: blue !important;  
}
```



CSS Tutorial

LES ELEMENTS HTML DE TYPE
BLOCK ET DE TYPE INLINE

Les éléments de type block

- En HTML, tout élément est soit de type « block » (bloc), soit de type « inline » (en ligne).
- Par exemple, l'élément p est un élément de type block tandis que l'élément strong est un élément de type inline
- Connaître le type d'un élément HTML est essentiel pour créer et mettre en forme des pages web

```
<!DOCTYPE html>
<html>
  <head>
    <title>Eléments block vs inline</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1>Eléments block et inline</h1>
    <p>L'élément p est un élément de type block</p>
    <p>Un <strong>autre</strong> paragraphe</p>
  </body>
</html>
```

Les éléments de type block

- Un élément de type block va toujours commencer sur une nouvelle ligne et prendre toute la largeur disponible dans la page
- Un élément de type block peut contenir d'autres éléments de type block ainsi que des éléments de type inline.
- Exemple avec un élément p :

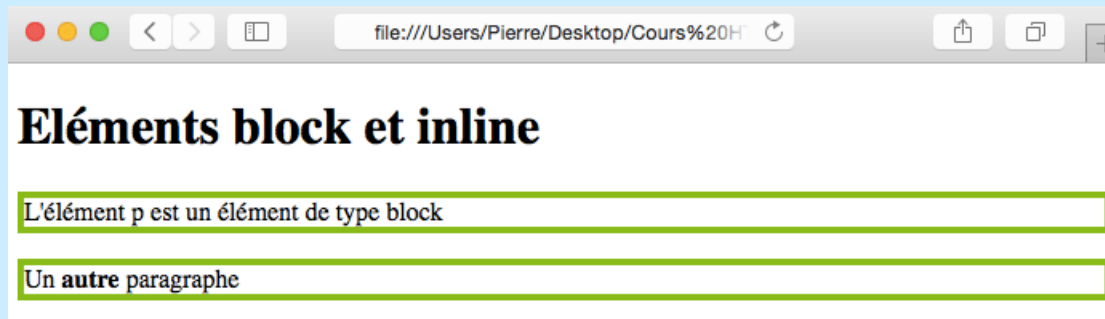
```
<!DOCTYPE html>
<html>
  <head>
    <title>Eléments block vs inline</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1>Eléments block et inline</h1>
    <p>L'élément p est un élément de type block</p>
    <p>Un <strong>autre</strong> paragraphe</p>
  </body>
</html>
```


Les éléments de type block

- (Suite)

```
p{  
  border: 4px solid #88BB11;  
}
```



- Pour tester le code : <http://jsbin.com/sipujup/20/edit?html,css,output>

Les éléments de type block

- Ici, on s'est contenté de créer une bordure autour des éléments p grâce à la propriété CSS border (étudiée en détail plus tard).
- Les éléments HTML de type block les plus communs sont les suivants :
 - L'élément p ;
 - Les éléments h1, h2, etc. ;
 - Les éléments ol et ul ;
 - L'élément form (utilisé pour créer des formulaires) ;
 - L'élément div (que nous étudierons dans la partie suivante).

Les éléments de type inline

- Au contraire des éléments de type block, les éléments de type inline ne vont pas commencer sur une nouvelle ligne mais s'insérer dans la ligne actuelle.
- Les éléments de type inline prennent uniquement la largeur qui leur est nécessaire (c'est-à-dire la largeur de leur contenu).
- Par exemple, l'élément HTML `strong` est un élément de type inline

```
<!DOCTYPE html>
<html>
  <head>
    <title>Éléments block vs inline</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1>Éléments block et inline</h1>
    <p>L'élément p est un élément de type block</p>
    <p>Un <strong>autre</strong> paragraphe</p>
  </body>
</html>
```

Les éléments de type inline

- (suite)

```
strong{  
  border: 4px solid #88BB11;  
}
```



- Pour tester le code : <http://jsbin.com/gopega/42/edit?html,css,output>

Les éléments de type inline

- Les éléments HTML de type inline les plus communs sont les suivants :
 - Les éléments strong et em ;
 - L'élément a ;
 - L'élément img ;
 - L'élément span (que nous allons étudier dans la partie suivante)

CSS Tutorial

LES ELEMENTS HTML DIV ET SPAN

Utilisation de l'élément HTML div

- Les éléments div et span n'ont pas pour but de donner un sens à un contenu.
- Il servent de conteneurs pour faciliter la mise en forme d'une page HTML à des niveaux différents.
- La différence entre ces deux éléments est que le premier est de type block tandis que le second est de type inline.

Utilisation de l'élément HTML div

- **L'élément HTML div est un élément de type block.** Cet élément va souvent être utilisé comme conteneur pour plusieurs autres éléments HTML.
- Soit un exemple pratique, où nous avons créé un **div** autour de deux paragraphes.
- Ensuite, nous appliquons les styles CSS directement à ce div.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Éléments div et span</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

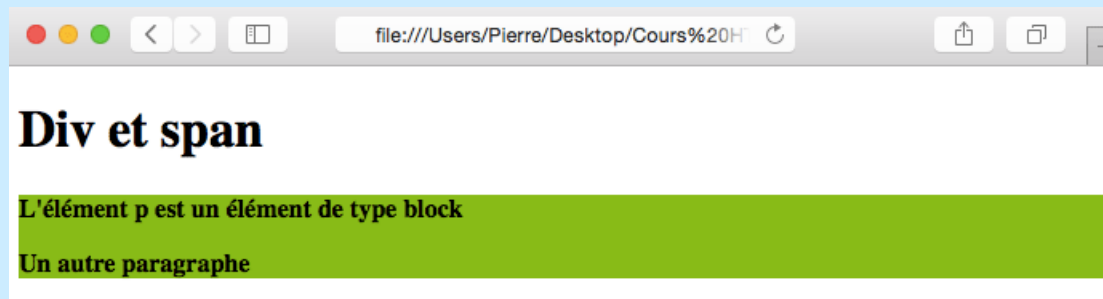
  <body>
    <h1>Div et span</h1>

    <div class="div-para">
      <p>L'élément p est un élément de type block</p>
      <p>Un <strong>autre</strong> paragraphe</p>
    </div>
  </body>
</html>
```


Utilisation de l'élément HTML div

- (Suite)

```
/*Nous utilisons des notations hexadécimales pour la couleur ici.  
*Nous verrons ces notations plus tard dans le cours*/  
.div-para{  
    background-color: #88BB11;  
    font-weight: bold;  
}
```



- Pour tester le code : <http://jsbin.com/gopega/43/edit?html,css,output>

Utilisation de l'élément HTML span

- Cet élément va souvent servir de conteneur pour du contenu textuel. Cet élément va s'avérer pratique dans certains cas où nous aimerions mettre en forme une portion de texte à l'intérieur d'un élément sans pouvoir la cibler de façon simple.
- L'élément span sera souvent utilisé avec un attribut class afin de pouvoir le cibler plus facilement en CSS.

Utilisation de l'élément HTML span

- Ici, nous appliquons une couleur de fond verte à notre élément div portant la valeur « green », puis une couleur de fond bleue plus spécifiquement à notre élément span.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Éléments div et span</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="styles.css">
  </head>

  <body>
    <h1>Div et span</h1>

    <div class="green">
      <p>L'élément p est <span class="blue">un élément de type block</span></p>
      <p>Un <strong>autre</strong> paragraphe</p>
    </div>
  </body>
</html>
```

Utilisation de l'élément HTML span

- (Suite)

```
/*Nous utilisons des notations hexadécimales pour la couleur ici.  
*Nous verrons ces notations plus tard dans le cours*/  
.green{  
    background-color: #88BB11;  
    font-weight: bold;  
}  
  
.blue{  
    background-color: #1188BB;  
}
```



- Pour tester le code : <http://jsbin.com/sipujup/21/edit?html,css,output>

