

Développement Web Côté client Module 4 - CSS



Objectifs



Découvrir le CSS



Savoir mettre en place du CSS



Comprendre les sélecteurs



Découvrir les propriétés CSS



Comprendre la cascade, la spécificité et l'héritage

Introduction

CSS



- Définition :
 - CSS : Cascading Style Sheets
 - Traduction : Feuilles de style en cascade
- Rôles :
 - Sert à décrire la présentation des pages HTML, XML
- Standards :
 - Définis par le W3C
- Version actuelle :
 - CSS 3

Mise en place CSS

Mise en place

- Il existe trois façons de mettre en place du CSS :
 - Sur une feuille de **style externe** au HTML (externe)
 - Dans un **bloc dédié** dans la page HTML (interne)
 - Directement **dans une balise HTML** (inline)
- Il y a une priorité pour chacune de ces façons de procéder:
 - Inline > Interne > externe
 - Ce qui signifie que si un style est appliqué sur un même élément HTML sur ces trois niveaux, c'est le style inline qui sera prioritaire car il est plus spécifique.

Mise en place

Exemple :

- Mettre le titre h1 de la page en rouge :

- Code CSS :

- `color: red;`

- Code HTML :

```
<!DOCTYPE html>
<html lang="fr">

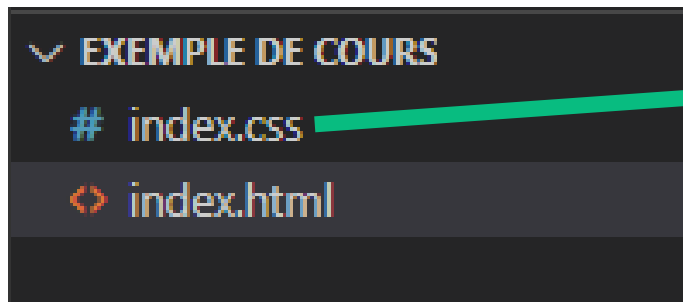
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>CSS</title>
</head>

<body>
  <h1>Hello World!</h1>
</body>

</html>
```

Externe

- Il faut créer une feuille de style dans le projet et y ajouter le code CSS :



```
h1 {  
  color: red;  
}
```

- Il faut lier la page CSS à la page HTML en ajoutant une balise `link` dans le `head` :

```
<link rel="stylesheet" href="index.css">
```


DÉMONSTRATION

Style externe

Interne

Il faut ajouter un bloc de code CSS dans le head de la page HTML grâce à la balise style :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS - Style interne</title>
  <style>
    h1 {
      color: red;
    }
  </style>
</head>
<body>
  <h1>Hello World!</h1>
</body>
</html>
```

DÉMONSTRATION

Interne

Inline

Il faut ajouter un attribut style directement sur l'élément HTML :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS - Style interne</title>
</head>
<body>
  <h1 style="color: red;">Hello World!</h1>
</body>
</html>
```

DÉMONSTRATION

UPDATE
Inline



Recommandations

- **Il est préférable d'utiliser le style externe**
 - Maintenabilité
 - Centralisation du code CSS
 - Code CSS réutilisable
 - Architecture Web plus solide
 - Recommandation du W3C
 - Possibilité d'importer une page CSS dans une autre (`@import`)
- Les styles interne et inline doivent être évités

Syntaxe CSS

CSS

- Pour appliquer un style à un élément, il convient de respecter la syntaxe suivante :

```
selecteur {  
    property1: value;  
    property2: value;  
}
```

- Il est possible d'insérer des commentaires :
 - `/* mon commentaire */`

Sélecteurs

CSS

Sélecteurs

- Pour appliquer du CSS à un élément d'une page Web, il doit être sélectionné. Il faut donc un **sélecteur**.
- Les sélecteurs sont nombreux et variés en CSS.
- Les trois sélecteurs principaux :
 - Sélecteur de **balise**
 - Sélecteur de **classe**
 - Sélecteur **d'identifiant**


Sélecteurs de balise

- Le sélecteur de balise (ou sélecteur de type) fonctionne directement en appelant **l'élément HTML** sans les chevrons
- Un style sur un élément HTML s'applique à tous les éléments du même type de la page
- Exemple :
 - `h1 { ... }`
 - `p { ... }`
 - `div { ... }`
 - `a { ... }`
 - `span { ... }`
 - ...

Sélecteurs de classe

- Le sélecteur de classe est défini sur un élément HTML via l'attribut **class**
- Cette classe est utilisée dans la feuille de style en **précédant son nom d'un point**
- Un style sur une classe s'applique à tous les éléments porteurs de la même classe
- Plusieurs classes peuvent être ajoutées à un même élément HTML
- Exemple :

```
<p class="vert">Un texte en vert</p>
```

```
.vert {  
  color:  green;  
}
```

Sélecteurs d'identifiant

- Le sélecteur d'identifiant est défini sur un élément HTML via l'attribut **id**
- Cet identifiant est utilisé dans la feuille de style en **précédant son nom d'un dièse**
- L'identifiant a pour objectif d'être **unique**. Il ne doit pas se retrouver sur un autre élément HTML
- Dans l'idéal, l'utilisation des identifiants comme sélecteurs n'est pas une pratique très recommandée. **L'identifiant sera plutôt réservé au JavaScript.**
- Exemple :

```
<p id="bleu">Un texte en bleu</p>
```

```
#bleu {  
  color:  blue;  
}
```

Sélecteurs et spécificité

- La **spécificité** s'applique également sur les sélecteurs. Plus ce dernier est précis, plus il est spécifique et donc prioritaire
- Par exemple, le sélecteur d'identifiant est plus spécifique que celui de la classe. Sur un même élément HTML possédant un id et une class qui comporterait un style sur une même propriété, celui de l'identifiant serait prioritaire
- id > class > élément

Sélecteurs

- Il existe également d'autres sélecteurs :
 - Le **sélecteur universel**, `*`, qui permet de sélectionner tous les éléments de la page
 - Les **sélecteurs d'attributs** `[attribut]`
 - Les **pseudo-classes** qui permettent d'appliquer un style selon un état de l'élément (par exemple `a:hover` permet d'appliquer un style si la souris survole l'élément)
 - Les **pseudo-éléments** permettent de mettre en forme certaines parties d'un élément (par exemple `p::first-line` permet de travailler la première ligne du paragraphe)
 - Il est aussi possible de combiner des critères pour plus de précisions avec les combinateurs (*sélecteurs de voisins directs, sélecteurs de voisins, sélecteurs d'éléments enfants, sélecteurs d'éléments descendants, combinateurs de colonne*)

DÉMONSTRATION

Sélecteurs

Convention de nommage

Afin de nommer ses identifiants (id, class) correctement, il faut respecter certains principes :

- Le nom choisi ne peut contenir que :
 - des lettres
 - des caractères ISO 10646 U+00A0 et supérieurs
 - des traits d'union
 - des tirets bas
- Il ne peut pas débiter par :
 - Un chiffre
 - Deux traits d'union
 - Un trait d'union suivi d'un chiffre

Convention de nommage

- Lorsque le projet grandit et que le ou les fichiers CSS deviennent volumineux, il peut être bon d'adopter une convention de nommage précise.
- Deux conventions sont beaucoup utilisées :
 - **OOCSS** (Object Oriented CSS)
 - **BEM** (Block, Element, Modifier)
- Pour en savoir plus :
 - [Page GitHub du OOCSS](#)
 - [Documentation BEM](#)

TRAVAUX PRATIQUES

CSS DINER

Propriétés CSS

Propriétés

- Les propriétés CSS sont nombreuses (plus de 200)
- Elles concernent tous les sujets :
 - Textes
 - Images
 - Ancres
 - Couleurs
 - Formulaires
 - Validation de données
 - Animations
 - ...

Propriétés

Par exemple, pour le texte, il existe :

- `font-size` : Permet de modifier la taille d'un texte
- `font-weight` : Permet de modifier le 'poids' du texte (mise en gras)
- `font-style` : Permet de modifier le style du texte (italique)
- `text-decoration` : Permet de choisir le soulignement du texte
- `text-shadow` : Permet d'ajouter une ombre à un texte
- `font-family` : Permet de choisir la police (font)
- `text-align` : Permet de déterminer le placement horizontal du texte (aligné à gauche, centré, justifié, ...)
- `color` : Permet de choisir la couleur du texte
- ...

Propriétés

Il n'est pas possible de découvrir l'ensemble des propriétés CSS pendant ce cours.

- Elles sont trop nombreuses
- Certaines sont très rarement utilisées
- Il n'est pas possible de connaître par cœur l'ensemble de ces éléments
- Le travail d'un développeur est avant tout de savoir chercher de l'information face à un problème donné

Propriétés

Voici malgré tout quelques propriétés couramment utilisées

color

margin

background-color

padding

font-family

background-image

border

display

font-weight

border-radius

width

height

font-size

font-style

**TRAVAUX
PRATIQUES**

Quizz

Cascade, spécificité et héritage

CSS

Cascade, spécificité

- Parfois, si deux règles s'appliquent au même élément, il y a un conflit
- La **cascade** est le mécanisme qui contrôle quelle règle s'applique lors de ce conflit
- Ce concept est étroitement lié à celui de la **spécificité**
- Elle permet de « scorer » un sélecteur
- Ce score est couramment porté sous cette forme : x.x.x (ex: 0.1.0)

Spécificité

- Calcul de la spécificité :
 - La spécificité se calcule sur le **sélecteur** choisi. Plus celui-ci est spécifique et plus le score est élevé
 - Le chiffre des **unités** est lié au sélecteurs de balise et pseudo-éléments
 - Le chiffre des **dizaines** est lié au sélecteur de classes, attributs et pseudo-classes
 - Le chiffre des **centaines** est lié au sélecteurs d'identifiant
 - En cas d'égalité, c'est la dernière règle déclarée qui est prioritaire
 - Il est aussi possible de « casser » la spécificité avec **!important**
- Par exemple :

h1 { }

0.0.1

.titre { }

0.1.0

#titre { }

1.0.0

section p { }

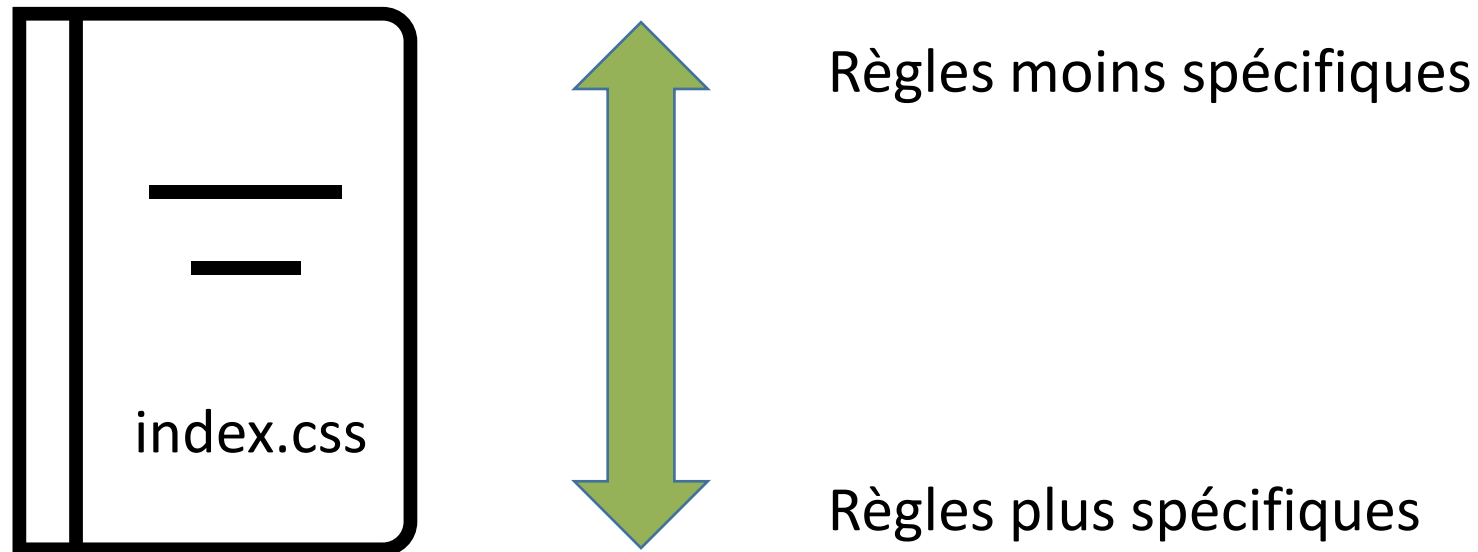
0.0.2

section.titre { }

0.1.1

Spécificité

Afin de rendre sa feuille de style plus lisible et maintenable dans le temps, il convient de « ranger » les règles de la moins spécifique à la plus spécifique



TRAVAUX PRATIQUES

Spécificité

Héritage

- En CSS, certaines propriétés peuvent être **héritées**.
- Par exemple, si une couleur de texte est définie sur un élément parent alors l'élément enfant prendra par défaut cette couleur.
- **Attention ! Toutes les propriétés ne s'héritent pas.**
- Il est possible d'utiliser la valeur **inherit** pour définir un héritage explicite y compris sur les propriétés qui ne s'héritent pas.

Héritage

Voici quelques propriétés héritables :

- Color
- Cursor
- Font-family
- Font-size
- Font-style
- Font-weight
- Letter-spacing
- Line-height
- List-style-image
- List-style-position
- List-style
- Text-align
- Text-indent
- Text-justify
- visibility






DÉMONSTRATION

Héritage

**TRAVAUX
PRATIQUES**

TP Fil rouge

CONCLUSION

-  Vous avez découvert le CSS
-  Vous savez mettre en place du CSS
-  Vous comprenez les sélecteurs
-  Vous avez découvert les propriétés CSS
-  Vous comprenez la cascade, la spécificité et l'héritage