

# Développement front

MMI 2 – CM#4 S3



Danielo **JEAN-LOUIS**  
Michele **LINARDI**

# CSS — Rappels

- Langage de description
  - Complémentaire du HTML
- Permet de définir la présentation d'une page web
- Extension de fichier : .css
- Fonctionne avec un système de sélecteurs

# CSS — Sélecteurs (vus jusqu'à présent)

- Sélecteur d'id : #mon-id
- Sélecteur de classe : .ma-classe
- Sélecteur d'attribut : [mon-attribut]
- Sélecteur universel : \*
  - **A utiliser avec parcimonie**
- Sélecteur de groupe : .classe1, .classe2

# CSS — Sélecteurs (vus jusqu'à présent)

- Sélecteur de descendants :
  - .classe1 [espace] .classe2
- Sélecteur de pseudo-classe : :pseudo-classe
- Sélecteur de pseudo-element : ::pseudo-element

**CSS 3 est venu ajouter  
(progressivement) d'autres  
sélecteurs (et pseudo-classes) très  
utiles pour le développement**

**Des sélecteurs dits “avancés”**

# Pourquoi les sélecteurs CSS avancés ?

- Réduire l'interdépendance entre le développement front et back pour appliquer certaines déclarations CSS
  - Ex : Appliquer une classe une fois sur deux
- Pallier aux limites de certains sélecteurs CSS
  - Ex : sélecteur de descendants



# Pourquoi les sélecteurs CSS avancés ?

- Améliorer le référencement (SEO) du site web en réordonnant le site pour que le HTML soit “SEO compatible” mais la mise en page du site conforme au design

# Sélecteur d'enfants directs

- Permet de cibler l'enfant direct d'une balise



```
div > span {  
    background-color: blue;  
}
```

Le caractère ">" permet de cibler tout <span> contenu directement dans une balise <div>

Note : il est possible de mettre plusieurs ">" dans le sélecteur

## Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/Child\\_combinator](https://developer.mozilla.org/fr/docs/Web/CSS/Child_combinator)

# Sélecteur d'enfants directs



```
<div>
  <span>
    Ciblé
    <span>Non ciblé</span>
  </span>
</div>
```

Dans le cas du code CSS précédent, seul le premier `<span>` sera ciblé

## Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/Child\\_combinator](https://developer.mozilla.org/fr/docs/Web/CSS/Child_combinator)

# Sélecteur de voisin direct

- Cible toute balise située juste à côté d'une autre



```
.paragraphe+ .liste {  
    margin-block: 1.5rem;  
}
```



```
<div class="paragraphe"></div>  
<ul class="liste"></ul>
```

Le caractère “+” permet de cibler tout `.liste` précédé **directement** par `.paragraphe`


Note : il est possible de mettre plusieurs “+” dans le sélecteur

Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/Next-sibling\\_combinator](https://developer.mozilla.org/fr/docs/Web/CSS/Next-sibling_combinator)

# Sélecteur de voisin

- Alternative au sélecteur de voisin direct
- Cible tout élément suivant un autre (immédiatement ou non)



```
.paragraphe~.liste {  
    margin-block: 1.5rem;  
}
```



```
<div class="paragraphe"></div>  
<figure>❗ [ ... ] ❗</figure>  
<ul class="liste"></ul>
```

Le caractère “~” permet de cibler tout .liste précédé par .paragraphe~, et ce, même s’il y a un élément (ou plus) entre les deux

Note : il est possible de mettre plusieurs “~” dans le sélecteur (non consécutifs)

Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/Subsequent-sibling\\_combinator](https://developer.mozilla.org/fr/docs/Web/CSS/Subsequent-sibling_combinator)

# Pseudo-classe - :is(<selecteurs>)

- Alternative “saine” du sélecteur de groupe
  - Ex : .ma-classe1, .ma-classe2
- Permet de raccourcir des sélecteurs

## Source(s) :

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

# Pseudo-classe - :is(<selecteurs>)



```
:valid,  
:unsupported {  
    /* ... */  
}
```

La pseudo-classe “unsupported” n’existant pas, l’ensemble du sélecteur ne sera pas appliqué



```
:is(:valid, :unsupported) {  
    /* ... */  
}
```

Avec la pseudo-classe is(), même si “:unsupported” n’existe pas, les déclarations CSS s’appliqueront pour “:valid”

Source(s) :

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

# Pseudo-classe - :is(<selecteurs>)



```
h1.newclass, p.newclass, li.newclass {  
  background-color: yellow;  
}
```



```
.newclass:is(h1, p, li) {  
  background-color: yellow;  
}
```

Les deux codes font la même chose, mais avec la pseudo-classe is(),  
le code est plus court

Source(s) :

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



# Pseudo-classe - `selecteur:has(<condition>)`

- Cible le sélecteur en fonction de la condition passée en paramètre
- Sorte de sélecteur de parent
- Peut être combiné avec la pseudo-classe `:is()`
  - Voir documentation mdn

Source(s) :

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

# Pseudo-classe - selecteur:has(<condition>)



```
.patisserie:has( > img ) {  
    grid-column: 1/-1;  
}
```

On cible tout élément avec la classe `.patisserie` qui possède une balise `img` comme enfant direct

Source(s) :

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

## Pseudo-classe - selecteur:not(<condition>)

- Sélectionne tout élément qui ne correspond pas aux conditions passées en paramètres
- Inverse des pseudo-classes :has() et :is()

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/:not>

# Pseudo-classe - selecteur:not(<condition>)



```
.article :not(p):not(.important) {  
  background-color: darkred;  
}
```

Cible tout élément n'ayant pas la classe "important" ou n'étant pas la balise <p> contenu dans la classe .article



```
.article:not(.important, .archive) {  
  color: rebeccapurple;  
  font-style: italic;  
}
```

Cible tout élément avec la classe .article et ne possédant pas la classe .important ni .archive

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/:not>

# Pseudo-classe - :nth-child()

- Cible des éléments en fonction de leur indice dans leur parent (premier élément, indice = 1)
- Accepte en condition un mot-clé ou une fonction sous la forme :  $An+B$ 
  - A : Incrément entier
  - B : Décalage entier
  - n : Entier positif commençant à 0

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/:nth-child>

# Pseudo-classe - :nth-child()



```
li:nth-child(even) {  
  background-color: pink;  
}
```

On cible un <li> sur deux en commençant par le deuxième

Note : Le mot-clé “odd” fait l’inverse



```
li:nth-child(-n + 2) {  
  color: darkblue;  
}
```

On cible les deux premiers <li>

$-0 + 2 = 2$

$-1 + 2 = 1$

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/:nth-child>

# Pseudo-classe - :nth-child()

- Possibilité de faire une plage de sélection



```
li:nth-child(n+4):nth-child(-n+10) {  
    /* [ ... ] */  
}
```

On cible le quatrième <li> jusqu'au dixième <li>

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/:nth-child>

## Pseudo-classe - :nth-child(of <selector>)

- Alternative à la pseudo-classe :nth-child()
- Sélectionne les n éléments trouvés (quelque soit leur position dans leur parent)

Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/:nth-child#la\\_syntaxe\\_of\\_selector](https://developer.mozilla.org/fr/docs/Web/CSS/:nth-child#la_syntaxe_of_selector)



# Pseudo-classe - :nth-child(of <selector>)



```
:nth-child(-n + 3 of li.important) {  
  color: red;  
}
```

On cible les trois premiers <li> avec la classe .important trouvés



```
li.important:nth-child(-n + 3) {  
  color: red;  
}
```

On cible les trois premiers <li> avec la classe .important si et seulement s'ils sont les premiers de leur parent

Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/:nth-child#la\\_syntaxe\\_of\\_selector](https://developer.mozilla.org/fr/docs/Web/CSS/:nth-child#la_syntaxe_of_selector)

# Pseudo-classe - :nth-child() - Raccourcis

- Pseudo-classes permettant de cibler directement un index précis
  - Premier élément : :first-child
  - Dernier élément : :last-child
- Calculateur de nth-child()
  - <https://css-tricks.com/examples/nth-child-tester/>

Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/:nth-child#la\\_syntaxe\\_of\\_selector](https://developer.mozilla.org/fr/docs/Web/CSS/:nth-child#la_syntaxe_of_selector)

# Pseudo-classe - :nth-last-child()

- Fonctionne comme :nth-child() mais on compte à partir de la fin
- Fonction de forme :  $An+B-1$



```
li:nth-last-child(-n + 3) {  
  color: red;  
}
```

On cible les trois derniers <li>

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/:nth-last-child>

# Pseudo-classe - :nth-of-type()

- Fonctionne comme :nth-child() mais permet de cibler un sélecteur
- Existe également sous la forme :nth-last-child()

## Source(s) :

- <https://webdocs.dev/fr/docs/web/css/:nth-of-type>
- <https://webdocs.dev/fr/docs/web/css/:nth-last-of-type>
- <https://la-cascade.io/articles/la-difference-entre-nth-child-et-nth-of-type>

# Pseudo-classe - :nth-of-type()



```
p:nth-of-type(3n) {  
  color: lime;  
}
```

On cible une balise <p> sur trois

## Source(s) :

- <https://webdocs.dev/fr/docs/web/css/:nth-of-type>
- <https://webdocs.dev/fr/docs/web/css/:nth-last-of-type>
- <https://la-cascade.io/articles/la-difference-entre-nth-child-et-nth-of-type>

# Pseudo-classe - :nth-of-type()

Fonctionne

```
<div>
  <h1 class="titre">Formations</h1>
  <p>MMI</p>
  <p>GE2I</p>
</div>
```

Ne fonctionne pas

```
p:nth-of-type(2) {
  color: lime;
}
```

```
p:nth-child(2) {
  color: lime;
}
```

Dans cette configuration, seul :nth-of-type(2) changera la couleur des deux balises <p>

Source(s) :

- <https://webdocs.dev/fr/docs/web/css/:nth-of-type>
- <https://webdocs.dev/fr/docs/web/css/:nth-last-of-type>
- <https://la-cascade.io/articles/la-difference-entre-nth-child-et-nth-of-type>

# Pratiquons ! - Découvrons les sélecteurs CSS avancés

Pré-requis :

- Avoir la ressource ressources/selecteurs-css-avances

A télécharger ici :

[https://github.com/DanYellow/cours/raw/refs/heads/main/s3-integration-web/cours-magistraux/numero-4/s3-integration-web\\_cours-magistraux\\_numero-4.ressources.zip](https://github.com/DanYellow/cours/raw/refs/heads/main/s3-integration-web/cours-magistraux/numero-4/s3-integration-web_cours-magistraux_numero-4.ressources.zip)

# Sélecteurs CSS avancés

- Fonctionnent avec tailwindcss avec des modifieurs dédiés :
  - :last, :even, :has()...
- Ou classes :
  - peer, group-\*

Source(s) :

- <https://tailwindcss.com/docs/hover-focus-and-other-states#pseudo-classes>
- <https://tailwindcss.com/docs/hover-focus-and-other-states#styling-based-on-the-descendants-of-a-peer>



**Questions ?**

