La cascade en CSS

1. Qu'est-ce que la cascade ?

La cascade est un concept clé en CSS qui détermine comment les styles sont appliqués lorsqu'ils proviennent de plusieurs sources. Elle permet de résoudre les conflits en établissant un ordre de priorité basé sur :

1. **L'origine des styles** (navigateur, utilisateur ou auteur).
2. **La spécificité des sélecteurs**.
3. **La règle d'importance (** `!important` **)**.

Grâce à la cascade, CSS peut combiner des règles provenant de fichiers différents tout en appliquant les bonnes priorités.

2. Comment les styles sont appliqués ?

2.1. L'origine des styles

Il existe trois origines des styles CSS :

1. **Les styles du navigateur** : Ce sont les styles par défaut appliqués par chaque navigateur (par exemple, la taille des titres `<h1>` ou le fond blanc d'un document).
2. **Les styles de l'utilisateur** : Certains utilisateurs peuvent définir leurs propres styles pour personnaliser leur navigation.
3. **Les styles de l'auteur** : Ce sont ceux écrits par vous, le développeur.

Priorité : Auteur > Utilisateur > Navigateur

2.2. La spécificité des sélecteurs

Lorsque plusieurs règles ciblent le même élément, la spécificité décide laquelle sera appliquée.

Règle de calcul de spécificité :

- **Sélecteurs d'éléments/tag** : 1 point (ex. `h1` , `p`).
- **Classes, pseudo-classes et attributs** : 10 points (ex. `.class` , `a:hover` , `[type="text"]`).
- **ID** : 100 points (ex. `#id`).
- **Styles en ligne (attribut** `style` **)** : 1000 points.

Exemple :

```
 <p id="important" class="highlight">Texte de test</p>
p {
  color: blue; /* 1 point */
}
.highlight {
  color: green; /* 10 points */
}
#important {
  color: red; /* 100 points */
}
```

La couleur rouge (style ID) sera appliquée car elle a la spécificité la plus élevée.

2.3. La règle `!important`

La déclaration `!important` permet d'outrepasser toutes les autres règles, même celles ayant une spécificité plus élevée. Cependant, elle doit être utilisée avec parcimonie pour éviter de rendre le code difficile à maintenir.

Exemple :

```
 p {
  color: blue !important;
}
#important {
  color: red;
}
```

Malgré la spécificité plus élevée de `#important` *, la couleur bleue sera appliquée grâce à* `!important` *.*

2.4. L'ordre de déclaration

En cas d'égalité de spécificité, la dernière règle déclarée dans le fichier prendra le dessus.

Exemple :

```
 p {
  color: green;
}
p {
  color: purple;
}
```

La couleur purple sera appliquée car c'est la dernière déclaration.

3. Exemple pratique

HTML :

```
<div id="main" class="container">
 <h1>Bienvenue</h1>
 <p>Ce paragraphe est stylisé.</p>
 <p class="highlight">Ce texte est très visible.</p>
</div>
```

CSS :

```
p {
 color: blue; /* Spécificité : 1 */
}
.highlight {
 color: green; /* Spécificité : 10 */
}
#main p {
 color: red; /* Spécificité : 101 */
}
.highlight {
 color: yellow !important;
}
```

Explication :

- Le premier paragraphe `<p>` hérite de la spécificité de `#main`, donc il sera rouge.
- Le second paragraphe avec la classe `highlight` sera jaune à cause de `!important`.

4. À RETENIR

- **Origine des styles (priorité décroissante)** :
  - Auteur > Utilisateur > Navigateur.
- **Spécificité (poids des sélecteurs)** :
  - Tag (1) < Classe (10) < ID (100) < Style en ligne (1000).
- **Règles importantes :**
  - `!important` > toutes les autres déclarations.
- Lorsque deux règles ont la même spécificité :
  - La dernière dans le fichier prend priorité.

Résumé des priorités :

Critère	Priorité
Règle !important	Très élevé
Spécificité	Moyen
Ordre de déclaration	Faible