Les principaux sélecteurs CSS

1. **Sélecteurs simples**
    - * : Sélecteur universel (s'applique à tous les éléments).
    - element : Sélecteur d'éléments (balises HTML comme div, p, a).
    - .class : Sélecteur de classe (s'applique aux éléments ayant une classe spécifique).
    - #id : Sélecteur d'identifiant (s'applique à l'élément ayant cet identifiant).
    - [attribute] : Sélecteur d'attribut (cible les éléments ayant un attribut spécifique).
2. **Sélecteurs combinés**
    - E1 E2 : Sélecteur descendant (cible les E2 à l'intérieur des E1).
    - E1 > E2 : Sélecteur enfant direct (cible les E2 qui sont enfants directs des E1).
    - E1 + E2 : Sélecteur adjacent (cible le premier E2 immédiatement après E1).
    - E1 ~ E2 : Sélecteur général de frères (cible tous les E2 après E1 au même niveau).
3. **Pseudo-classes**
    - :hover : État survolé.
    - :focus : État actif d'un élément (ex. champ de formulaire actif).
    - :nth-child(n) : Cible le n-ième enfant d'un parent.
    - :first-child : Premier enfant d'un parent.
    - :last-child : Dernier enfant d'un parent.
4. **Pseudo-éléments**
    - ::before et ::after : Insèrent du contenu avant ou après un élément.
    - ::placeholder : Style de l'attribut placeholder dans un champ <input>.
5. **Sélecteurs d'attributs avancés**
    - [attr=value] : Sélecteur d'attribut égal à une valeur spécifique.
    - [attr^=value] : L'attribut commence par une valeur spécifique.
    - [attr$=value] : L'attribut finit par une valeur spécifique.
    - [attr*=value] : L'attribut contient une valeur spécifique.

Calcul de priorité des sélecteurs CSS

La priorité des sélecteurs CSS est calculée en fonction d'un système de **poids**. Voici les règles générales :

Priorité et score

- **Inline styles** : Les styles définis directement dans un élément HTML via l'attribut style (ex.) ont **la plus haute priorité**.
    - Poids : **1000**.
- **Sélecteurs ID** : Les sélecteurs #id ont une priorité très élevée.
    - Poids : **100**.
- **Sélecteurs de classes, pseudo-classes, attributs** : Les sélecteurs .class, [attr=value] et :hover sont moins prioritaires que les ID.
    - Poids : **10**.

- **Sélecteurs d'éléments et pseudo-éléments** : Les sélecteurs de type comme `div`, `h1`, ou `::before` ont la priorité la plus basse.
    - Poids : **1**.
- **Sélecteur universel et héritage** : Le sélecteur `*` et les styles hérités ont une priorité de **0**.

Calcul du score

Prenons un exemple pour comprendre comment calculer la priorité :

- Sélecteur : `div .box #main:hover`
    - `div` → Poids : 1.
    - `.box` → Poids : 10.
    - `#main` → Poids : 100.
    - `:hover` → Poids : 10.
    - **Score total = 1 + 10 + 100 + 10 = 121**.

Méthodes pour gérer les conflits de priorité

1. **Utiliser `!important`** : Forcer un style particulier à s'appliquer en ajoutant `!important`.
    - Exemple

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

2. Utiliser des sélecteurs spécifiques **: Augmentez la spécificité de vos sélecteurs en ajoutant des classes ou des ID.**
3. Respecter l'ordre de déclaration **: Si deux sélecteurs ont la même spécificité, le dernier dans le fichier CSS est appliqué**.

Exercices pratiques

Exercice 1 : Identifier la priorité

Attribuez une couleur différente à ces trois paragraphes et déterminez lequel aura la priorité :

```
<p id="main" class="text">Paragraphe 1</p>
<p class="text">Paragraphe 2</p>
<p>Paragraphe 3</p>
```

Styles CSS :

```
p { color: blue; }
.text { color: green; }
```

```
#main { color: red; }
```

Question : Quelle couleur sera appliquée à chaque paragraphe ? et pourquoi ?

Exercice 2 : Corriger les conflits

Écrivez le CSS nécessaire pour que tous les boutons aient une bordure rouge, sauf ceux ayant la classe .primary qui doivent avoir une bordure bleue.

HTML :

```
<button>Normal</button>
<button class="primary">Primary</button>
```

Exercice 3 : Utilisation des pseudo-classes

Créez une règle CSS qui change la couleur des liens :

- Rouge lorsqu'ils sont survolés (:hover).
- Vert lorsqu'ils ont été visités (:visited).
- Bleu par défaut.

Exercice 4 : Sélecteurs avancés

Créez une règle qui cible uniquement les champs <input> ayant un attribut type commençant par text.

HTML :

```
<input type="text" />
<input type="password" />
<input type="textarea" />
```

Question : Écrivez le CSS correspondant.

Exercice 5 : Priorité et !important

Expliquez pourquoi la couleur rouge sera appliquée dans l'exemple suivant, et proposez une solution pour forcer la couleur bleue.

HTML :

```
<p class="important">Texte important</p>
```

CSS :

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

Exercice 6 : Priorité des sélecteurs multiples

Considérez le HTML suivant :

```
<div id="container" class="box">
  <p class="text">Texte 1</p>
  <p>Texte 2</p>
</div>
```

Et les styles CSS :

```
p { color: green; }
#container p { color: blue; }
.box p { color: red; }
```

Questions :

1. Quelle couleur sera appliquée au **Texte 1** ?
2. Quelle couleur sera appliquée au **Texte 2** ?

Exercice 7 : !important et spécificité

Considérez le HTML suivant :

```
<div id="card" class="highlight">
  <p class="info">Paragraphe important</p>
</div>
```

Et les styles CSS :

```
p { font-size: 16px; }
.info { font-size: 18px !important; }
#card .info { font-size: 20px; }
```

Question : Quelle taille de police sera appliquée au paragraphe ? Pourquoi ?

Exercice 8 : Priorité avec des sélecteurs combinés

Considérez le HTML suivant :

```
<div id="wrapper">
  <div class="content">
    <h1>Bonjour</h1>
  </div>
</div>
```

Et les styles CSS suivants :

```
h1 { font-weight: normal; }
.content h1 { font-weight: bold; }
#wrapper .content h1 { font-weight: lighter; }
```

Questions :

1. Quel style sera appliqué à l'élément <h1> ?
2. Quel est le score de spécificité pour chacun des sélecteurs ci-dessus ?

Exercice 9 : Sélecteurs d'attributs

Considérez le HTML suivant :

```
<input type="text" id="input1" class="field" />
<input type="password" id="input2" class="field" />
```

Et les styles CSS suivants :

```
input { border: 1px solid black; }
input[type="text"] { border: 2px solid blue; }
.field { border: 3px solid green; }
#input1 { border: 4px solid red; }
```

Questions :

1. Quelle bordure sera appliquée à input1 ?
2. Quelle bordure sera appliquée à input2 ?

Exercice 10 : Conflit entre classes et IDs

Considérez le HTML suivant :

```
<div id="header" class="banner">
  <h1>Bienvenue</h1>
</div>
```

Et les styles CSS suivants :

```
.banner { background-color: yellow; }
#header { background-color: orange; }
div { background-color: grey; }
```

Question : Quelle couleur de fond sera appliquée au `div` avec l'ID `header` ? Pourquoi ?

Exercice 11 : Priorité des pseudo-classes

Considérez le HTML suivant :

```
<a href="#" class="link">Lien 1</a>
<a href="#">Lien 2</a>
```

Et les styles CSS suivants :

```
a { text-decoration: none; }
a.link { text-decoration: underline; }
a:hover { text-decoration: line-through; }
```

Questions :

1. Quelle apparence aura le **Lien 1** au survol de la souris ?
2. Quelle apparence aura le **Lien 2** au survol de la souris ?

Exercice 12 : Ordre de déclaration et priorité

Considérez le HTML suivant :

```
<div class="content" id="main">
  <p>Paragraphe test</p>
</div>
```

Et les styles CSS suivants :

```
#main p { color: blue; }
.content p { color: green; }
p { color: red; }
```

Question : Quelle couleur sera appliquée au paragraphe et pourquoi ?

Exercice 13 : Combinaisons complexes

Considérez le HTML suivant :

```
<div id="block1" class="container">
  <div class="child">Texte A</div>
</div>
<div class="container">
  <div class="child">Texte B</div>
</div>
```

Et les styles CSS suivants :

```
div { color: black; }
.container .child { color: red; }
#block1 .child { color: blue; }
```

Questions :

1. Quelle couleur sera appliquée au **Texte A** ?
2. Quelle couleur sera appliquée au **Texte B** ?

Exercice 14 : Priorité avec inline styles

Considérez le HTML suivant :

```
<p style="color: orange;" class="highlighted">Texte principal</p>
```

Et les styles CSS suivants :

```
p { color: black; }
.highlighted { color: green; }
```

Question : Quelle couleur sera appliquée au paragraphe ?

Exercice 15 : Combinaison de pseudo-classes et classes

Considérez le HTML suivant :

```
<ul>
  <li class="active">Élément 1</li>
  <li>Élément 2</li>
  <li>Élément 3</li>
</ul>
```

Et les styles CSS suivants :

```
li { font-size: 14px; }
li.active { font-size: 16px; }
li:hover { font-size: 18px; }
```

Questions :

1. Quelle taille de police sera appliquée à l'**Élément 1** au survol ?
2. Quelle taille de police sera appliquée à l'**Élément 2** au survol ?