CSS: Les Sélécteurs avancés en CSS

Par Robert DIASSÉ

Sélecteurs Avancés en CSS

Les sélecteurs avancés en CSS offrent une précision accrue pour cibler des éléments spécifiques dans une page web. Ils permettent de définir des styles en fonction de la relation entre les éléments, de leurs attributs ou de leur état. Voici un aperçu des sélecteurs avancés les plus utilisés :

1. **Sélecteur d'éléments enfants (>)** : Cible les éléments qui sont des enfants directs d'un autre élément.

   Exemple :

   ```
   ul > li {
       /* Styles pour les éléments li qui sont des enfants directs d'ul */
   }
   ```

2. **Sélecteur d'éléments adjacents (+)** : Cible les éléments qui sont adjacents immédiats d'un autre élément.

   Exemple :

   ```
   h2 + p {
       /* Styles pour le paragraphe qui suit immédiatement un titre h2 */
   }
   ```

3. **Sélecteur d'éléments généraux (~)** : Cible les éléments qui sont des frères suivants d'un autre élément, sans être nécessairement adjacents immédiats.

   Exemple :

   ```
   h2 ~ p {
       /* Styles pour tous les paragraphes qui suivent un titre h2 */
   }
   ```

4. **Sélecteur d'éléments d'attribut (`[attribut=valeur]`)** : Cible les éléments ayant un attribut spécifique avec une valeur spécifique.

   Exemple :

```
input[type="text"] {
    /* Styles pour les éléments input de type texte */
}
```

Pseudo-classes CSS(`:`)

Les pseudo-classes en CSS permettent de cibler des éléments dans des états spécifiques ou des relations avec d'autres éléments.

1. **:hover** : Cible un élément lorsqu'il est survolé par la souris.

   Exemple :

```
button:hover {
    background-color: #f0f0f0;
}
```

2. **:active** : Cible un élément lorsque l'utilisateur clique dessus.

   Exemple :

```
button:active {
    transform: translateY(1px);
}
```

3. **:focus** : Cible un élément lorsqu'il a le focus, généralement utilisé pour les éléments de formulaire.

   Exemple :

```
input:focus {
    border-color: #007bff;
    box-shadow: 0 0 5px #007bff;
}
```

4. **:first-child** : Cible le premier enfant d'un élément parent.

   Exemple :

```

```

```
li:first-child {
    font-weight: bold;
}
```

5. **:last-child** : Cible le dernier enfant d'un élément parent.

   Exemple :

```
li:last-child {
    color: red;
}
```

6. **:nth-child(n)** : Cible l'élément spécifié par sa position dans la liste d'enfants de son parent.

   Exemple :

```
tr:nth-child(even) {
    background-color: #f2f2f2;
}
```

7. **:nth-of-type(n)** : Cible l'élément spécifié par sa position dans la liste d'éléments du même type.

   Exemple :

```
p:nth-of-type(odd) {
    color: blue;
}
```

8. **:not(selector)** : Cible les éléments qui ne correspondent pas au sélecteur spécifié.

   Exemple :

```
input:not([type="submit"]) {
    border: 1px solid #ccc;
}
```

9. **:checked** : Cible les éléments de formulaire qui sont cochés.

   Exemple :

```
input[type="checkbox"]:checked + label {
    font-weight: bold;
}
```

10. **:visited** : Cible les liens déjà visités par l'utilisateur.

    Exemple :

```
a:visited {
    color: purple;
}
```

Pseudo-éléments CSS(::)

Les pseudo-éléments CSS permettent de styliser certaines parties spécifiques des éléments HTML.

1. **::before** : Insère du contenu avant le contenu de l'élément sélectionné.

   Exemple :

```
p::before {
    content: "Avant ";
}
```

2. **::after** : Insère du contenu après le contenu de l'élément sélectionné.

   Exemple :

```
p::after {
    content: " Après";
}
```

3. **::first-line** : Stylise la première ligne d'un élément de bloc.

   Exemple :

```
p::first-line {
    font-weight: bold;
}
```

4. **::first-letter** : Stylise la première lettre d'un élément de bloc.

   Exemple :

```
p::first-letter {
    font-size: 150%;
```

```
}
```

5. **::selection** : Stylise le texte sélectionné par l'utilisateur.

   Exemple :

```
::selection {
    background-color: yellow;
    color: black;
}
```

6. **::placeholder** : Stylise le texte de l'attribut placeholder d'un élément input.

   Exemple :

```
input::placeholder {
    color: gray;
}
```

7. **::marker** : Stylise le marqueur d'une liste.

   Exemple :

```
li::marker {
    color: red;
}
```

8. **::backdrop** : Stylise le fond derrière un élément avec un contenu superposé, comme une modale.

   Exemple :

```
dialog::backdrop {
    background-color: rgba(0, 0, 0, 0.5);
}
```

9. **::before et ::after avec contenu généré** : Insère du contenu généré avant ou après l'élément sélectionné.

   Exemple :

```
button::before {
    content: "\2713"; /* Insertion d'un symbole de coche */
}
```

10. **::nth-letter(n)** : Stylise la n-ième lettre d'un élément de bloc.

    Exemple :

```
p::nth-letter(3) {
    color: blue;
}
```

Ces pseudo-classes et pseudo-éléments sont largement utilisés pour styliser des éléments HTML de manière précise et sophistiquée.

Les sélecteurs avancés en CSS permettent un contrôle plus précis sur le style des éléments dans une page web, en fonction de divers critères tels que la structure du document, les interactions de l'utilisateur ou les attributs des éléments. Ils sont essentiels pour créer des mises en page complexes et des designs sophistiqués.

EXERCICES

Exercice 1 : Utilisation des Pseudo-classes CSS

Vous disposez d'une liste d'étudiants répartis en deux groupes, A et B. Chaque groupe est représenté par une section avec un titre et une liste d'étudiants. Chaque étudiant est contenu dans un élément `<li>` avec la classe `.etudiant`; contenant son nom,prenom et age.

Tâche :

1. Utilisez les pseudo-classes CSS appropriées pour styliser les groupes A et B différemment.
2. Appliquez un style différent au deuxième étudiant de chaque groupe.
3. Appliquez un style de survol lorsque l'utilisateur survole chaque étudiant.
4. Appliquez un style différent au groupe B s'il contient des étudiants ayant une certaine classe, par exemple `.excellent`.
5. Excluez certains étudiants de l'application d'un style particulier, par exemple, n'appliquez pas de style au dernier enfant de chaque groupe.

Exercice 2 : Utilisation des Pseudo-elements CSS

Vous travaillez sur une application de blog où les utilisateurs peuvent consulter des articles sur différents sujets; chaque sujet est contenu dans une balise `article` et chaque article est dans une liste `li` avec un titre . Pour améliorer l'apparence et la convivialité de votre application, vous avez décidé d'ajouter des styles décoratifs aux titres et aux descriptions des articles dans la liste.

1. personnaliser les marqueurs de liste des articles en ajoutant des icônes ou des couleurs en fonction de la catégorie de l'article.

2. ajouter un symbole d'attention avant les titres des articles qui ont une priorité élevée ou qui sont mis en évidence.