CSS Selector là gì? 8 Loại CSS Selector CẦN BIẾT

Biết cách sử dụng các **Bộ chọn CSS** (hay còn gọi là **CSS Selector**) là một kỹ năng bắt buộc của **lập trình viên front end**.

CSS Selector là gì?

CSS Selector là gì?

Hiểu đơn giản **CSS Selector** là thứ cho phép bạn nhắm mục tiêu tới các phần tử HTML để áp dụng các thuộc tính CSS cho chúng.

CSS Selector giống như là đường dẫn, chỉ định để cho CSS biết bạn đang muốn điều chỉnh, tạo kiểu cho phần tử HTML nào vậy.

Đây là file HTML và file CSS. Và bạn sẽ viết CSS để điều chỉnh, tạo kiểu cho phần tử HTML.

CSS Selector trong file CSS

Qua bài viết này, bạn sẽ được tìm hiểu về **8 Loại CSS Selector phổ biến nhất**.

Chỉ với **8 Bộ chọn CSS** này mình tin rằng bạn còn biết nhiều hơn một lập trình viên Frontend chuyên nghiệp.

Tại sao bạn cần biết 8 CSS Selector này?

8 Loại CSS Selector Phổ biến nhất

Mặc dù có rất nhiều CSS Selector khác, nhưng mình thấy chúng thiếu tổ chức, thiếu ví dụ hoặc có quá nhiều thông tin.

Đó là lý do tại sao mình nhóm CSS Selector này thành 8 loại chung.

Khi bắt đầu tìm hiểu về mỗi nhóm, bạn sẽ thấy có một đoạn mã HTML.

Hãy sử dụng đoạn mã HTML đấy để thử nghiệm loại CSS Selector được giới thiệu trong nhóm.

Lưu ý: Danh sách này không đầy đủ và nó còn tiếp tục được update.

Nhưng về cơ bản, nó sẽ đáp ứng hầu hết các nhu cầu sử dụng CSS Selector của bạn.

Mục lục:

1. Basic CSS Selector

Bộ chọn CSS cơ bản (**Basic CSS Selector**) sử dụng chọn element / class / id.

Chúng được sử dụng thường xuyên nhất và dễ nhớ nhất.

... Như đã nói từ trước, đây là code HTML sử dụng để cho bạn chơi với Basic CSS Selector.

Code HTML:

```
<div id="app">
  <div class="container">
    <p class="hello">Hello</p>
    <p class="hola">Hola</p>
  </div>
  <div class="other">
    I am left behind...
  </div>
</div>
```

- Element Selector: `Element`. Nó chọn bất kỳ phần tử nào đó.

```
p { color: blue; }
div { color: magenta; }
```

- Class Selector: `.class`. Nó chọn tất cả các phần tử có `class` đã cho.

```
.hello {
  color: red;
}
```

- ID Selector: `#id`. Nó chọn tất cả các phần tử có `id` đã cho.

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

- Universal selector: `*`. Nó chọn tất cả các phần tử.

```
* {
  color: yellow;
}
```

> Lưu ý: Trước khi chuyển sang phần tiếp theo. Hãy nhớ là thực hành sử dụng các CSS Selector đã được giới thiệu để thử nghiệm với mã HTML ở trên xem có thành công không bạn nhé.

Và **CSS Dinner** là một trò rất thú vị để bạn luyện tập sử dụng CSS Selector. Vừa học lý thuyết về các Selctor (bên dưới đây) và dùng nó để vượt qua 32 Level này nhé.

Học CSS Selector qua Game CSS Dinner

> Tham khảo ngay các khóa học dưới đây để nhanh chóng làm chủ bộ kỹ năng lập trình Web. Học với giảng viên doanh nghiệp. Hỗ trợ việc làm cuối khóa.

- ***KHÓA HỌC LẬP TRÌNH PHP***
- ***KHÓA HỌC LẬP TRÌNH JAVA***

2. Descendant CSS Selector

Đây là các **CSS Selector để chọn hậu duệ** của bất kỳ phần tử nào.

Code HTML:

```
<div class="container">
  <div class="paragraph-container">
    <p id="hola-id" class="hola-class">Hola World</p>
    <p class="hello-class">Hello World</p>
    <p class="hello-class again-class">Hello Again World</p>
  </div>
</div>
```

- Any descendant selector: `A B`. Chọn bất kỳ phần tử B nào là hậu duệ của A. Hậu duệ có thể được lồng rất sâu.

```
.container .hello-class {
  color: red;
}
```

Chúng ta có thể kết hợp với `*` để chọn tất cả các phần tử hậu duệ:

```
.paragraph-container * {
  color: blue;
}
```

CSS Selector trên chọn mọi phần tử bên trong `.paragraph-container`

Child Selector: `A > B`. Không giống như *Any Descendant CSS Selector*, CSS Selector này chỉ chọn hậu duệ trực tiếp.

Thử nghiệm với CSS như sau để xem kết quả nhé:

```
.paragraph-container > .hello-class {
  color: blue;
}
```

Trong khi đó, CSS Selector bên dưới sẽ không hoạt động vì class `.hello-class` trong HTML này không phải là hậu duệ trực tiếp.

```
.container > .hello-class {
  color: blue;
}
```

3. Multiple CSS Selector

Multiple CSS Selector cho phép chúng ta chọn nhiều phần tử không liên quan với nhau.

Code HTML:

```
<div class="container">
  <div class="paragraph-container">
    <p id="hola-id" class="hola-class">Hola world</p>
    <p class="hello-class">Hello world</p>
    <p class="hello-class again-class">Hello again world</p>
  </div>
  <p class="outside-class">I'm outside</p>
</div>
```

- Multiple CSS Selector: `A, B, C, D ...` . Để chọn nhiều phần tử / class / id.

```
.outside-class, .again-class, .hola-class {
  color: purple;
}
```

4. Combination CSS Selector

Conbination CSS Selector cho phép bạn chọn thành phần rất cụ thể bằng nhiều tham chiếu.

Trường hợp sử dụng phổ biến nhất là hiệu ứng làm nổi bật button khi được di chuột (hovered) / nhấp (clicked) bằng cách cung cấp cho chúng một class `.active`.

Code HTML:

```
<div class="container">
  <div class="paragraph-container">
    <p id="hola-id" class="hola-class">Hola world</p>
    <p class="hello-class">Hello world</p>
    <p class="hello-class active">Hello again world</p>
  </div>
  <p class="outside-class">I'm outside</p>
</div>
```

- Combination CSS Selector: `AB`. Cho phép chọn phần tử chứa cả A và B. Cú pháp trông giống như Descendant CSS Selector, ngoại trừ phần này không có khoảng trắng.

```
p.active {
  color: yellow;
}
```

Chúng ta có thể kết hợp nhiều thứ, không phải chỉ có class. (Nhớ là để chúng sát với nhau)

```
p#hola-id {
  color: blue;
}
.hello-class.active {
  color: red;
}
```

5. Sibling CSS Selector

Sibling CSS Selector nhắm chọn các phần tử anh chị em.

Code HTML:

```
<div class="container">
  <div class="paragraph-container">
    <p id="hola-id" class="hola-class">Hola world</p>
    <p class="hello-class">Hello world</p>
    <p class="hello-class again-class">Hello again world</p>
  </div>
  <p class="outside-class">I'm outside</p>
</div>
```

- Bộ chọn Anh / Chị / Em liền kề (Nghiêm ngặt): `A + B`. Nhắm mục tiêu một phần tử anh chị em được đặt ngay sau phần tử đó.

```
#hola-id + .hello-class {
  color: blue;
}
```

Lưu ý rằng CSS bên dưới sẽ không hoạt động vì class `.again`, mặc dù đó là anh chị em của `#hola-id`, nhưng nó cách `#hola-id` 2 phần tử.

```
#hola-id + .again-class {
  color: blue;
}
```

- Bộ chọn Anh / Chị / Em liền kề (Không nghiêm ngặt): `A ~ B`. Giống như bộ chọn bên trên nhưng không giới hạn 1 phần tử đầu tiên.

```
#hola-id ~ .hello-class {
  color: purple;
}
```

Bạn thấy không, nó chọn cả thẻ `p` bên dưới vì chúng là phần tử anh chị em với nhau và có class là `.hello-class`

> *Lưu ý: Bộ chọn này không hoạt động ngược lại.*

Thế nên CSS bên dưới không hoạt động.

```
.again-class ~ #hola-id {
  color: red
}
```

6. Pesudo CSS Selector

Code HTML:

```
<div class="container">
  <div class="paragraph-container">
    <p id="hola-id" class="hola-class">Hola world</p>
```

```
    <p class="hello-class">Hello world</p>
    <p class="hello-class again-class">Hello again world</p>
  </div>

  <p class="outside-class">I'm outside</p>

  <ul id="list-id" class="list-class">
    <li class="list-item-class">First</li>
    <li class="list-item-class">Second</li>
    <li class="list-item-class">Third</li>
    <li class="list-item-class">Fourth</li>
    <li class="list-item-class">Fifth</li>
  </ul>

  <div class="single-paragraph-container">
    <p>I'm the only child of this span</p>
  </div>
</div>
```

- Chọn phần tử đầu tiên: `A:first-child`. Nó chọn đến phần tử con đầu tiên. (`A` phải có cha mẹ).

```
li:first-child {
  color: blue;
}
```

Bạn đoán thử xem. Nhắm mục tiêu `ul:first-child` có thành công không?

```
ul:first-child {
  color: red;
}
```

Chẳng có gì xảy ra.

Điều này là do mặc dù `ul` có cha mẹ (div với class `.container`), nhưng `ul` không phải là con đầu tiên (nó là con thứ 3).

CSS dưới đây sẽ hoạt động vì `.paragraph-container` là con đầu tiên của `.container`.

```
.paragraph-container:first-child {
  color: red;
}
```

- Chọn phần tử con cuối cùng: A:last-child. Hoạt động như `A:first-child`, ngoại trừ thay vì chọn phần tử con đầu tiên, nó chọn phần tử con cuối cùng.

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

- Chỉ chọn phần tử con: A:only-child. Chọn tất cả A là con duy nhất của cha mẹ nó. Tương tự như bộ chọn phần tử con đầu tiên (`A:first-child`) và phần tử con cuối cùng (`A:last-child`). Ngoại trừ mục tiêu không được có anh chị em nào.

```
p:only-child {
  color: red;
}
```

Lưu ý mặc dù chúng ta có một số phần tử `p`, chỉ phần tử cuối cùng được áp dụng vì các phần tử `p` khác không phải là phần tử con duy nhất của cha mẹ chúng.

Nói cách khác, phần tử con có anh chị em thì không được áp dụng.

- Bộ chọn phần tử con thứ N: `A:nth-child(n)`. Nó chọn từng mục tiêu là con thứ `n` của cha mẹ nó.

```
li:nth-child(2) {
  color: red;
}
```

Hãy thử với phần tử `p`:

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

Bạn có thể thấy tại sao hai phần tử `p` thay đổi màu sắc?

- Bộ chọn con thứ N cuối cùng: `A:nth-last-child(n)`. Nó tương tự như bộ chọn con thứ N, ngoại trừ nó được tính từ cuối cùng.

```
li:nth-last-child(2) {
  color: red;
}
```

- Không chọn: `A:not(B)`. Chọn tất cả các phần tử A mà không phải B.

```
p:not(.outside-class) {
  color: blue;
}
```

- Bộ chọn loại đầu tiên: `A:first-of-type`. Nó chọn phần tử đầu tiên thuộc loại này trong số các phần tử anh chị em của nó.

Nghe có vẻ tương tự `:first-child` nhỉ?

Nhưng không phải.

Để cảm nhận sự khác biệt của nó, hãy thử nghịch nó xem sao:

```
li:first-of-type {
  color: blue;
}
```

Ok, đó không phải là một ví dụ hay vì nó cho thấy kết quả tương tự như `li:first-child` 😅.

Hãy làm một ví dụ khác:

```
p:first-of-type {
  color: red;
}
```

Cái này thì tốt hơn.

So sánh với `p:first-child`. Bạn sẽ nhận thấy rằng `p` ở giữa chuyển sang màu đỏ khi sử dụng `:first-of-type.`

Điều này là do `p` ở giữa là loại phần tử `p` đầu tiên trong số các anh chị em của nó.

- Bộ chọn loại cuối cùng: `A:last-of-type`. Giống như trên, nhưng ngược lại.

```
li:last-of-type {
  color: blue;
}
```

- Bộ chọn loại thứ N: `A:nth-of-type(n)`. Cũng giống như hai bộ chọn trên, nhưng nó chọn phần tử thứ N.

```
li:nth-of-type(2) {
  color: blue;
}
```

Hãy thử với phần tử khác.

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

Nếu chúng ta thay đổi nó thành `p:nth-of-type(1)`, thì nó hoạt động giống như `first-of-type`.

Ngoài ra nếu chúng ta thay đổi nó thành `p:nth-of-type(4)`, không có thay đổi gì. Đây là do không có phần tử `p` thứ 4 trong số các anh chị em.

- Bộ chọn Only type: `A:only-of-type`. Chọn phần tử chỉ có loại đó, không có Anh / Chị / Em.

```
p:only-of-type {
  color: purple;
}
```

Chú ý, chúng ta có hai phần tử `p` đổi màu.

Điều này là do có hai phần tử `p` này không có Anh / Chị / Em cùng loại.

7. Pesudo-Selector (links và input)

Dưới đây là danh sách các **pesudo selector** khác. Chúng thường được liên kết với các liên kết (`a`) (mặc dù chúng có thể hoạt động với các phần tử không liên kết `non-link`).

Code HTML:

```
<div id="app">
  <a class="cheesyLink" href="#">I like cheese</a>
  <a class="sweetLink" href="#">I like donut</a>
  <div class="burger">I like cheezburger</div>
  <div class="container">
    <form onsubmit="event.preventDefault()">
      <input class="myinput"type="text" />
      <input class="mysubmit" type="submit" />
    </form>
  </div>
</div>
```

- Hover Selector: `A:hover`. Chọn phần tử được hover. Thường được sử dụng để làm nổi bật các liên kết.

```
a:hover {
  color: red;
}
```

Không chỉ hoạt động với thẻ `a`. Bộ chọn này hoạt động với mọi thứ.

```
.burger:hover {
  color: red;
}
```

- Focus Selector: `A:focus`. Chọn phần tử bạn đang tập trung vào nó. Thường được sử dụng với `input`.

```
input:focus {
  background: red;
}
```

- Active Selector: `A:active`. Chọn phần tử đang có trạng thái active.

```
.cheesyLink:active {
  background: red;
}
```

Lưu ý khi bạn click vào nó, nền (background) sẽ thay đổi.

- Link Selector: `A:link`. Nó chọn tất cả các link chưa được click.

```
a:link {
  background: blue;
}
```

8. Attribute CSS Selector

Bộ chọn thuộc tính CSS (**Attribute CSS Selector**) sử dụng để chọn thuộc tính HTML (**HTML Attributes**).

Code HTML:

```
<div for="chocolate">Chocolate</div>
<div for="peanut">Peanut</div>
<div for="butter">Butter</div>
<div>Jelly</div>
```

- Bộ chọn thuộc tính cơ bản: `A[B]`. Chọn tất cả các phần tử `A` có thuộc tính `B`.

```
div[for] {
  color: red;
}
```

- Bộ chọn thuộc tính cụ thể: `A[B="C"]`. Chọn các phần tử `A` có thuộc tính `B` với giá trị là `C`.

```
div[for="chocolate"]{
  color: blue;
}
```

- Bộ chọn thuộc tính cụ thể (Bắt đầu với ...): `A[B^="C"]`. Chọn tất cả các phần tử `A` có thuộc tính `B` với giá trị bắt đầu là `C`. Ký tự `^` là ký tự thể hiện chuỗi bắt đầu (trong **Biểu thức chính quy** (Regex)).

```
div[for^="cho"] {
```

```
  color: magenta;
}
```

- Bộ chọn thuộc tính cụ thể (Kết thúc với ...): `A[B$="C"]`. Chọn tất cả các phần tử `A` có thuộc tính `B` với giá trị kết thúc là `C`. Ký tự `$` là ký tự thể hiện chuỗi kết thúc trong (Biểu thức chính quy (Regex)).

```
div[for$="er"] {
  color: yellow;
}
```

- Bộ chọn thuộc tính cụ thể (Chứ ký tự ...): `A[B*="C"]`. Chọn tất cả phần tử `A` với thuộc tính `B` chứa giá trị `C`.

```
div[for*="ut"]{
  color: cyan;
}
```

Ok, đến đây, mình rât tiếc phải xin lỗi rằng:

Bạn đã biết quá nhiều về CSS Selector...

Mi đã biết quá nhiều về CSS Selector

Với 8 Bộ **CSS Selector** này mình tin rằng đáp ứng đầy đủ cho bạn trong quá trình lập trình web / app.