

Desenvolupament Digital

CSS. Introducció i Selectors

CSS (Cascading Style Sheets)

És un llenguatge que ens serveix per especificar l'**aparença** dels elements que formen part d'un **document HTML**.

Preferiblement, escribim el codi CSS **en un arxiu a part** i l'enllacem mitjançant una etiqueta **<link>** dins la secció **<head>**:

```
<head>
  ...
  <link rel="stylesheet" href="/style.css">
</head>
```

D'aquesta manera, podem fer servir **un mateix arxiu CSS** per a **diversos documents HTML**.

** També podem escriure codi CSS dins un element <style> o dins l'atribut "style" d'un element.*

Selectors

CSS es basa en l'ús de **selectors** per aplicar **propietats** a determinats elements.

```
h1 {  
    color: #333;  
    font-weight: bold;  
    font-size: 25px;  
    line-height: 32px;  
}
```

El **selector** determina **a quins elements s'aplicarà** un conjunt de propietats. Hi ha múltiples **propietats** que ens permeten **ajustar l'aparença** dels elements.

Selectors

CSS es basa en l'ús de **selectors** per aplicar **propietats** a determinats elements.

```
selector — h1 {  
    color: #333; — declaració  
    propietat — font-weight: bold; — valor  
    font-size: 25px;  
    line-height: 32px;  
}
```

El **selector** determina **a quins elements s'aplicarà** un conjunt de propietats. Hi ha múltiples **propietats** que ens permeten **ajustar l'aparença** dels elements.

Podem aplicar les mateixes regles a diferents selectors separant-los amb comes:

```
h1, h2, h3 {  
    font-weight: bold;  
}
```

Selectors per nom d'etiqueta

Existeixen diversos tipus de selectors. El més bàsic és un **nom d'etiqueta** HTML.

Per exemple, podem aplicar propietats a tots els elements **<a>**:

```
a {  
    /* Propietats */  
}
```

** A CSS podem escriure comentaris entre els caràcters “/*” i “*/”.*

Selectors per classe

Una altra manera d'identificar elements és mitjançant **noms de classe**. Per fer-ho, precedim el nom de la classe d'un punt.

```
.nom-de-classe {  
    /* Propietats */  
}
```

En aquest cas, seleccionariem qualsevol element amb la classe “**nom-de-classe**”.

Podem especificar **múltiples classes**, de manera que **totes elles hagin de ser presents** a l'element al qual s'apliquen les regles.

```
.classe1.classe2 {  
    /* Propietats */  
}
```

Quan seleccionem d'aquesta manera, no posem cap espai entre les diferents classes.

També podem **combinar classes amb un nom d'etiqueta**.

Per exemple, per seleccionar qualsevol element **<h1>** que tingui la classe "**main-title**":

```
h1.main-title {  
    /* Propietats */  
}
```

El nom d'etiqueta sempre s'escriu primer.

Selectors per id

Podem seleccionar un element pel seu atribut “**id**”. Per fer-ho, el precedim del caràcter “**#**”.

```
#about {  
    /* Propietats */  
}
```

En aquest cas, seleccionariem l’element amb l’id “**about**”.

Selector universal

Existeix un caràcter especial que ens serveix per seleccionar **qualsevol element**:

```
* {  
    /* Propietats */  
}
```

En aquest cas, seleccionariem **tots els elements** del document.

Selectors combinats

Podem **combinar** selectors senzills per crear **selectors més complexes**. Això ens permet seleccionar elements **en funció dels elements que els continguin**.

Per exemple, si volem seleccionar qualsevol **paràgraf** que estigui **dins un element <aside>**, podem fer-ho de la següent manera:

```
aside p {  
    /* Propietats */  
}
```

Un **espai** entre dos selectors indica que **l'element posterior ha d'estar dins de l'element anterior**. És **indiferent a quants nivells de profunditat**.

En l'exemple següent, doncs, estariem seleccionant **tots tres paràgrafs**:

```
aside p {  
    /* Propietats */  
}
```

```
<aside>  
→ <p> ... </p>  
→ <p> ... </p>  
  <footer>  
    → <p> ... </p>  
  </footer>  
</aside>
```

Podem especificar que un element ha d'estar **immediatament a dins de l'anterior** (al primer nivell de profunditat) amb el símbol “>”.

Modificant l'exemple anterior de la següent manera, **no seleccionariem el paràgraf dins de <footer>**:

```
aside > p {  
    /* Propietats */  
}
```

```
<aside>  
→ <p> ... </p>  
→ <p> ... </p>  
    <footer>  
        ✗ <p> ... </p>  
    </footer>  
</aside>
```

Podem **encadenar tants nivells de selectors com vulguem**, de manera que podem obtenir selectors compostos com els següents:

```
aside > footer p {}
```

```
.shop-item h3.item-title a {}
```

```
.site-header nav > * {}
```

*És preferible intentar **reduir**, en la mesura del possible, el **nombre de selectors encadenats**. Fer servir un ús extensiu de les classes ens permetrà reduir la complexitat dels selectors.*

Especificitat

Per defecte les declaracions CSS **s'apliquen en l'ordre en què apareixen**. Si una declaració contradiu una altra, preval la que sigui posterior.

En aquest exemple, el text dels paràgrafs seria de color vermell:

```
p {  
    color: black;  
}  
  
p {  
    color: red;  
}
```

Diferents tipus de selectors, però, tenen **diferent especificitat**:

- Una **classe** és més específica que un **nom d'etiqueta**.
- Un **id** és més específic que una **classe**.

Un selector més específic sempre prevaldrà sobre un menys específic.

Per exemple, amb els següents HTML i CSS, el títol seria vermell:

```
<h1 id="site-title" class="title">Títol</h1>
```

```
#site-title { color: red; }  
.title { color: blue; }  
h1 { color: black; }
```

Un **major nombre d'elements** en un selector compost impliquen **major especificitat**, però **mai superaran un únic element que tingui major especificitat que cadascun d'ells**.

Per exemple, aquí tenim una sèrie de selectors ordenats de menys a més específic:

Menys específic

p {}

aside > **footer** **p** {}

.highlight {}

.main-content **p.highlight** {}

Més específic

#tagline {}

Per evitar problemes d'especificitat i confusions, és recomanable:

- ***Simplificar el tipus de selectors*** que fem servir.
- ***Evitar selectors molt específics*** com selectors per id.
- ***Ordenar els selectors*** de menys a més específics.

