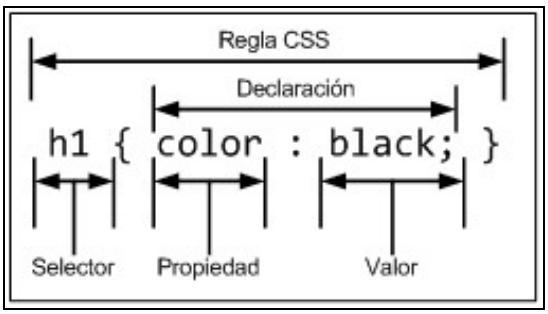
1 Selectores

O selector é a parte da regra de estilo que identifica o elemento ó que se lle aplican as instrucións de presentación. CSS ofrece varios tipos de selectores para mellorar a flexibilidade e eficacia da creación de follas de estilo. Aquí presentaremos os selectores seguintes:

1.1 Sumario

1.2 Compoñentes dun estilo CSS Básico



1.3 Selectores de elemento

Este é o selector mais sinxelo, vexamos uns exemplos:

```
h1 {color: blue;}
h2 {color: blue;}
p {color: blue;}
```

O exemplo seguinte ten o mesmo efecto que o anterior:

```
h1, h2, p {color: blue;}
```

Tamén temos un selector de elementos "universal" **(*),** vexamos un exemplo:

```
* {color: grey;}
```

◊ Esta regra de estilo pon en gris **todos** os elementos do documento dos que non estea especificada outra cor.
◊ É recomendable non empregar o selector universal se a páxina contén formularios.

1.4 Selectores contextuais

◊ Os **selectores de elemento** aplícanse a todos os casos nos que se atope o elemento no documento. Por contra, os **selectores contextuais** permiten aplicar estilos aos elementos baseándose no seu contexto ou que teñan certa relación con outro elemento.
◊ Hai varios tipos de selectores contextuais: **descendente**, **fillo** e **irmán**.
◊ Os selectores contextuais empregan un carácter específico para indicar o tipo de relación entre os elementos dos selectores.

1.4.1 Selector descendente

◊ Os selectores descendentes seleccionan elementos contidos noutro elemento.
◊ Indícanse nunha lista de elementos separados por un espazo, comezando có elemento de nivel mais alto.
◊ O seguinte exemplo especifica que os elementos *em* deben ter cor azul, pero só se son descendentes dun elemento de lista (*li*). Todos os demais elementos *em* non se verán afectados:

```
li em {color: blue;}
```

◊ Os selectores descendentes tamén se poden agrupar, vexamos un exemplo:

```
h1 em, h2 em, h3 em {color: red;}
```

◊ Os selectores descendentes tamén poden estar "aniñados" en varias capas de profundidade. O seguinte exemplo pon de cor amarelo "só" o texto enfatizado (*em*) das áncoras (*a*) que se atopen nas listas ordenadas (*ol*):

```
ol a em {color: yellow;}
```

Si se emprega o selector descendente combinado co selector universal, pódese restrinxir o alcance dun selector descendente. O seguinte exemplo, mostra os dous enlaces de cor vermello:

```
p a { color: red; }

<p><a href="#">Enlace</a></p>
<p><span><a href="#">Enlace</a></span></p>
```

Sen embargo, no seguinte exemplo soamente o segundo enlace amósase en cor vermella:

```
p * a { color: red; }

<p><a href="#">Enlace</a></p>
<p><span><a href="#">Enlace</a></span></p>
```

A razón é que o selector **p * a** tradúcese coma todos os elementos de tipo **<a>** que se atopan dentro de calquera elemento, que a súa vez, se atopa dentro dun elemento de tipo **<p>**. Como o primeiro elemento **<a>** está directamente debaixo dun elemento **<p>**, non se cumpre a condición do selector **p * a**.

1.4.2 Selector fillo

◊ Un selector fillo é parecido a un selector descendente pero **só selecciona fillos directos dun elemento dado**. É dicir, non deben existir "niveis intermedios".
◊ Os elementos fillo están separados polo símbolo "maior que" (>).
◊ No seguinte exemplo ponse en gris o fondo do texto enfatizado pero só se é fillo directo dun parágrafo:

```
p > em {background-color: gray;}
```

Se aplicamos o estilo anterior ó seguinte parágrafo de texto só o primeiro *em* terá unha cor de fondo gris, porque o segundo é fillo dun elemento *strong*.

```
<p>O sistema operativo <em>Windows Vista</em> está sendo substituído,
en moitos equipos, polo <strong><em>Windows XP</em><strong>.
```

Os selectores fillo non teñen soporte en Netscape 4 nin en Explorer 6. Explorer 7 xa ten soporte.

1.4.3 Selector irmán

- Este selector utilízase para seleccionar un elemento que segue inmediatamente a outro có mesmo elemento pai.
- Utilízase o signo mais (+) para seleccionalos.
- Por exemplo, se queremos por en cor azul o primeiro parágrafo que segue a unha cabeceira de primeiro nivel, faríase así:

```
h1 + p {color: blue;}
```

Os selectores irmán non teñen soporte en Netscape 4 nin en Explorer 6. Explorer 7 xa ten soporte.

Outro exemplo:

```
h1 + h2 { color: red; }

<body>
<h1>Titulo1</h1>
<h2>Subtítulo</h2>
...
<h2>Outro subtítulo</h2>
...
```

Os estilos do selector h1 + h2 aplícanse ao primeiro elemento <h2> da páxina, pero non ao segundo <h2>, xa que:

- O elemento pai de <h1> é <body>, o mesmo pai ca o dos dous elementos <h2>. Así, os dous elementos <h2> cumpren a primeira condición do selector adxacente.
- O primeiro elemento <h2> aparece no código HTML xusto despois do elemento <h1>, polo que este elemento <h2> tamén cumpre a segunda condición do selector adxacente.
- Polo contrario, o segundo elemento <h2> non aparece xusto despois do elemento <h1>, polo que non cumpre a segunda condición do selector adxacente e polo tanto non se lle aplican os estilos de h1 + h2.

1.5 Selectores de clase e ID

Para poder facer uso de selectores mais específicos, faise preciso introducir os conceptos de **identificador (*id*)** e **clase (*class*)**. Nos seguintes apartados tamén veremos as **pseudo-clases** e os **pseudo-elementos** que tamén nos axudan para especificar mais polo miúdo a quen se lle aplica certo estilo.

1.5.1 Identificadores

◊ Os elementos HTML dispoñen dun atributo chamado identificador (*id*), que ten como finalidade asignar unha identificación unívoca a un único elemento. Deste xeito, CSS ou outra linguaxe poderá seleccionalo e distinguilo dos demais.

◊ Un *id* debe ser único en cada documento. Se temos varios elementos que precisen un tratamento similar empregaremos *class* tal e como se verá no seguinte apartado.

◊ Así, empregando este atributo *id* asignase ó elemento un nome específico que o distingue dos demais. Vexamos un exemplo.

```
<p id="nova">
```

◊ O uso dos identificadores é opcional polo que non ten sentido definilos se despois non se prevé o seu uso.

Recoméndase que o valor do *id* sexa un nome que clasifique de forma breve e esquemática ós elementos e que sexa facilmente recoñecible polo programador.

◊ Hoxe en día, os atributos *id* utilízanse para identificar seccións principais das páxinas. Algúns valores frecuentes serán: *contido*, *cabeceira*, *pe*,...

◊ Seguindo có mesmo exemplo, se queremos que o tipo de letra do elemento *p* identificado como **nova**, teña un tamaño de 14 píxeles, teríase que escribir a seguinte regra de estilo:

```
p#nova {font-size: 14px;}
```

◊ Có símbolo almofadiña (*#*) e o nome do identificador seleccionarase só o parágrafo en cuestión. Aínda que, como é único en cada documento, a regra de estilo anterior tamén se pode poñer do seguinte xeito:

```
#nova {font-size: 14px;}
```

1.5.2 Clases

◊ Emprégase o atributo *class* para identificar distintos elementos como parte dun grupo conceptual. Así, os elementos dunha clase poden modificarse cunha soa regra de estilo.
◊ No seguinte exemplo vemos que se identifican varios elementos dun documento como "especial":

```
<h1 class="especial">¡Atención!</h1>
<p class="especial">Hoxe temos grandes rebaixas.</p>
```

◊ Tamén se pode facer que un elemento do documento pertenza a mais dunha clase, só hai que separar estas con espazos. No seguinte exemplo facemos que un parágrafo pertenza á clase "novo" e á clase"especial" á vez:

```
<p class="novo especial">Hoxe temos grandes rebaixas.</p>
```

◊ Para identificar os estilos dunha determinada clase faise do seguinte xeito:

```
h1.especial {color: red;}
p.especial {color: blue;}
```

◊ Para aplicar unha propiedade a todos os elementos da mesma clase, débese omitir o nome da etiqueta no selector:

```
.especial {color: green;}
```

◊ Se queremos que todos os elementos dunha clase teñan unha propiedade menos algúns podemos facer así:

```
.especial {color: green;}
h1.especial{color: red;}
```

◊ Ter en conta que os nomes class non poden conter espazos en branco. Ademais, buscádelle un nome con significado para eses elementos (nova, alerta, datos-principais, datos-secundarios,...).
◊ Non excederse có seu uso, é moi fácil perderse, repetilos,... Hai sempre que gardar un guión-recordatorio.

1.6 Pseudoselectores

- Se queremos aplicar regras de estilo a elementos como os vínculos visitados, a primeira liña dun parágrafo ou á súa primeira letra, empregaremos os ***pseudoselectores***.

1.6.1 Pseudoclases

Como o seu nome indica, as pseudoclases funcionan como se existira unha clase aplicada a un grupo de elementos, na maioría dos casos será o elemento áncora (*a*).

- **Pseudoclases áncora**

  Temos as seguintes:

  - Enlaces non visitados:

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

  - Enlaces que xa foron visitados:

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

  - Cambiar a aparencia do enlace cando se pasa o punteiro por enriba:

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

  - Cambiar a aparencia do enlace xusto no intre no que se está premendo nel:

```
a:active {color: maroon;}
```

Algo típico é quitar o subliñado dos hiperenlaces:

```
a:link {color: red; text-decoration: none;}
a:visited {color: blue; text-decoration: none;}
```

Ou esta outra, que fai que o subliñado apareza cando poñemos o punteiro enriba do hiperenlace:

```
a:link {color: red; text-decoration: none;}
a:visited {color: blue; text-decoration: none;}
a:hover {color: red; text-decoration: underline;}
```

As pseudoclases áncora deben aparecen nunha determinada orde: ***:link, :visited, :hover, :active***.
Por se axuda, para recordalo, empréganse as iniciais **LVHA** que son as siglas de: ***love, HA!***.

- **Outras pseudoclases CSS**

Existen outras pseudoclases moito menos empregadas que as áncora:

:focus - Selecciona elementos que teñen o foco, como un elemento dun formulario. Un exemplo:

```
input:focus {background-color: yellow;}
```

:first-child - Selecciona o primeiro fillo dun elemento pai. No exemplo seguinte seleccionamos o primeiro *li* dunha *ul*:

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

¡Ollo! Moitos navegadores non teñen soporte para :focus e :first-chlid, por exemplo: Explorer 6 e anteriores, IE5 para Macintosh, Netscape 6+ Opera 7+.

1.6.2 Pseudoelementos

Os selectores pseudoelemento funcionan coma se estiveran inserindo elementos ficticios na estrutura do documento para dar estilo.
Estes pseudoelementos adoitan ser partes dun elemento xa existente en función do contexto, como pode ser a súa primeira liña ou a súa primeira letra.
Vexamos algúns:

:first-line - Aplica unha regra de estilo á primeira liña do elemento especificado. O seguinte código engade espazo extra á primeira liña do texto de cada parágrafo:

```
p:first-line {letter-spacing: 6pt;}
```

:first-letter - Aplica un estilo á primeira letra dun elemento. O seguinte exemplo modifica o estilo da primeira letra dos parágrafos clasificados como "definicion":

```
p.definicion:first-letter {font-size: 300%; color: red; }
```

:before e **:after** - Dar estilo ó contido existente antes e despois dun elemento.

1.7 Laboratorio de Simulación de Selectores

1.8  Sobre a páxina de Simulación

Nos enlaces máis abaixo, poderás probar os selectores e ver o resultado da selección na mesma páxina.
Para **accesos dende Internet** recoméndase escoller o primeiro hiperenlace **Servidor Externo**.
Para **accesos dende o IES San Clemente**, escoller o enlace equivalente **Mirror Instituto**.

Servidor Externo: Laboratorio de selectores.

Mirror Instituto: Laboratorio de selectores.

1.9 A palabra clave !important en CSS

!important funciona como unha palabra clave para ignorar as regras en cascada. Calquera definición que vaia acompañada dun **!important** terá maior importancia que calquera outra.

Agora vexamos exemplos para poder entende-lo ben:

Cando temos unha propiedade aplicada dúas veces, o navegador fará caso á última:

```
#main {
    width:600px;
    width:800px;
}
```

Polo tanto neste exemplo o navegador asignará 800 píxeles.

A declaración **!important** pode ser usada para dar prioridade a diferentes parámetros:

```
#main {
    width:600px !important;
    width:800px;
}
```

Agora neste exemplo aplicarase un ancho de 600 píxeles.

Internet Explorer 6 e versións anteriores ignoraban esta palabra clave (!important) mentres que IE7 si a soporta polo que se pode empregar para realizar pequenos hacks CSS.

Ou o que é o mesmo, para poder aplicar diferentes deseños nunha mesma folla de estilo teremos que usar os Hacks, que non é outra cousa que empregar regras que unicamente as entenda un dos navegadores.

Polo tanto poderémolo empregar de tal xeito que distingamos entre diferentes regras dependendo do navegador ou sen necesidade de usar Javascript, é un hack moi básico pero sen dúbida axudará a entender para que serve a palabra clave "**!important**":

```
#main {
    margin: 0 auto 0;
    max-width: 900px;
    min-width: 770px;
    width:auto !important;
    width:800px;
}
```

Neste exemplo aplícase un ancho dinámico (auto) a todos aqueles navegadores que soportan "!important" mentres que aplica un ancho de 800 píxeles a aqueles que non o soportan como por exemplo IE6.

Este pequeno truco non é realmente de todo útil pero sen dúbida axudará a comprender o significado desta palabra clave CSS.

Pero non hai que esquecer que **!important** é moi potente porque anteponse ó resto de regras polo que hai que usala con coidado como por exemplo cando queremos que unha definición non sexa substituída por ningunha outra ou para restablecer o valor dunha propiedade que non nos lembramos onde lle asignamos un valor que agora non queremos (sae algo en cor vermella e non nos lembramos onde o puxemos, pois faremos color: green !important; e solucionado).

--Manuel Vieites e -- Rafael Veiga feb 2008