UT3. CSS

UT3. CSS

3.1 Introducció
3.2. La sintaxis
    3.2.1 Les regles arrova (@)
    3.2.2 Els comentaris
    3.2.3 Ubicació del CSS
3.3 Atributs globals: class i id
3.4 Selectors / Pseudoclasses/ Pseudoelements

3.1 Introducció

Per tal d'evitar que l'HTML fos el responsable de la part estètica i visual de la web, es van idear els que s'anomenen, **fulls d'estil**, és a dir, es va dissenyar el llenguatge **CSS** (**Cascading Style Sheets**).

- Com ja hem comentat, al codi **HTML** col·loquem el contingut, és a dir, què ha de visualitzar-se.
- Mentre que amb **CSS** definim la presentació i l'estil, és a dir, com ha de visualitzar-se.

Recomanació: *L'adequat quan treballem amb CSS, és escriure el codi en fitxers independents, que tindran extensió* ***.css****. No convé col·locar codi CSS per tant dins d'arxius HTML, és a dir, s'ha d'evitar col·locar estils en etiquetes <style> en el propi codi HTML.*

3.1 Introducció

Les regles CSS, per poder definir el disseny de la pàgina, utilitzen:

- **Selectors:** Mitjançant ells, podem especificar a quins elements de la pàgina ens estem referint. Són les referències a les etiquetes (per exemple, id's i/o classes d'HTML).

- **Declaracions:** Són les unitat bàsiques de CSS, el que significa que no es pot emprar res més petit. Consisteix bàsicament en l'assignació d'un valor a una propietat. Es podria dir, de forma colloquial que és la resposta a una pregunta: De quin color hauria de ser els fons? per exemple: `color:red`

3.1 Introducció

Com és una regla CSS?

En primer lloc, si volem mitjançant CSS modificar l'estil d'una o diverses etiquetes HTML en una pàgina web, haurem de indicar a quina etiqueta o etiquetes afecta la modificació.Aquesta part del codi són els **selectors**. Per a indicar-los, excepte casos especials, s'escriu el nom de l'etiqueta o una referència a ella. per exemple, si en aquesta part escrivim **h1**, l'estil que apliquem afectarà a totes les etiquetes **<h1>** que hi hagi en la pàgina.

Després, s'haurà d'escriure què és el que volem canviar d'aquesta etiqueta i com. Tot això formarà un **bloc**, que anomenarem **declaració**. En CSS s'empren els signes "**{**" i "**}**" per a delimitar aquest bloc; Dins de les claus escriurem què és el que volem canviar i el nou valor que li donarem. Al que volem canviar li diem **propietat**, i ho indiquem mitjançant una paraula clau. Cada propietat té un **valor** (o varis). El valor indica la variació de la propietat, és a dir, quan o en quina ha canviat la propietat.

3.1 Introducció

Exemple:
Si volem canviar la grandària dels títols amb etiqueta h1: h1 { font- size : 12px } En aquest exemple h1 indica les etiquetes a les quals hem d'aplicar aquest estil; font- size indica la propietat (la grandària de lletra), mentre que 12px és el valor, és a dir la variació de la propietat (el canvi de grandària). Com pot observar-se la propietat i e valor estan separats pel signe de dos punts.

3.1 Introducció

Dins d'una mateixa declaració (signes de claus), pot haver-hi més d'una propietat amb el seu valor, aquestes han d'anar **separades** pel signe de **punt i coma.**

Exemple:

h1 { font- size : 12px; color : blue; }

En aquest exemple les etiquetes h1 reben dues propietats, amb la primera canviem la grandària de lletra, i amb la segona canviem el color de lletra.

El signe de punt i coma, es pot posar al final de l'última propietat, encara que no és obligatori resulta convenient, per si volem afegir alguna propietat més.

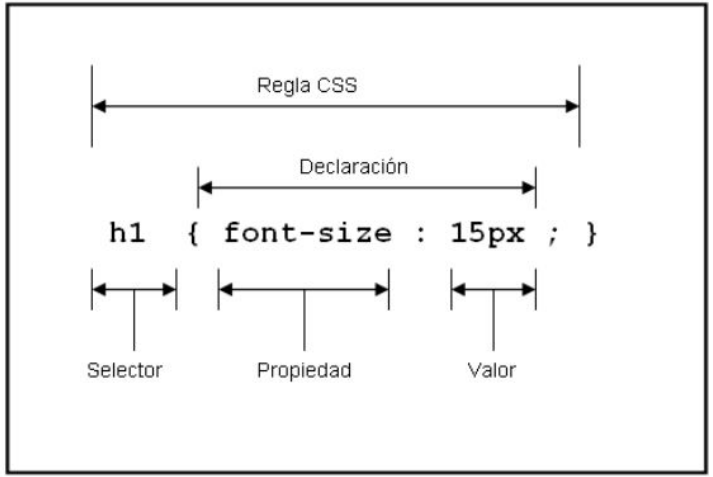
El llenguatge CSS es basa en una d'estructura, que podem resumir de la següent forma:

selector { propietat : valor; propietat : valor; ... }

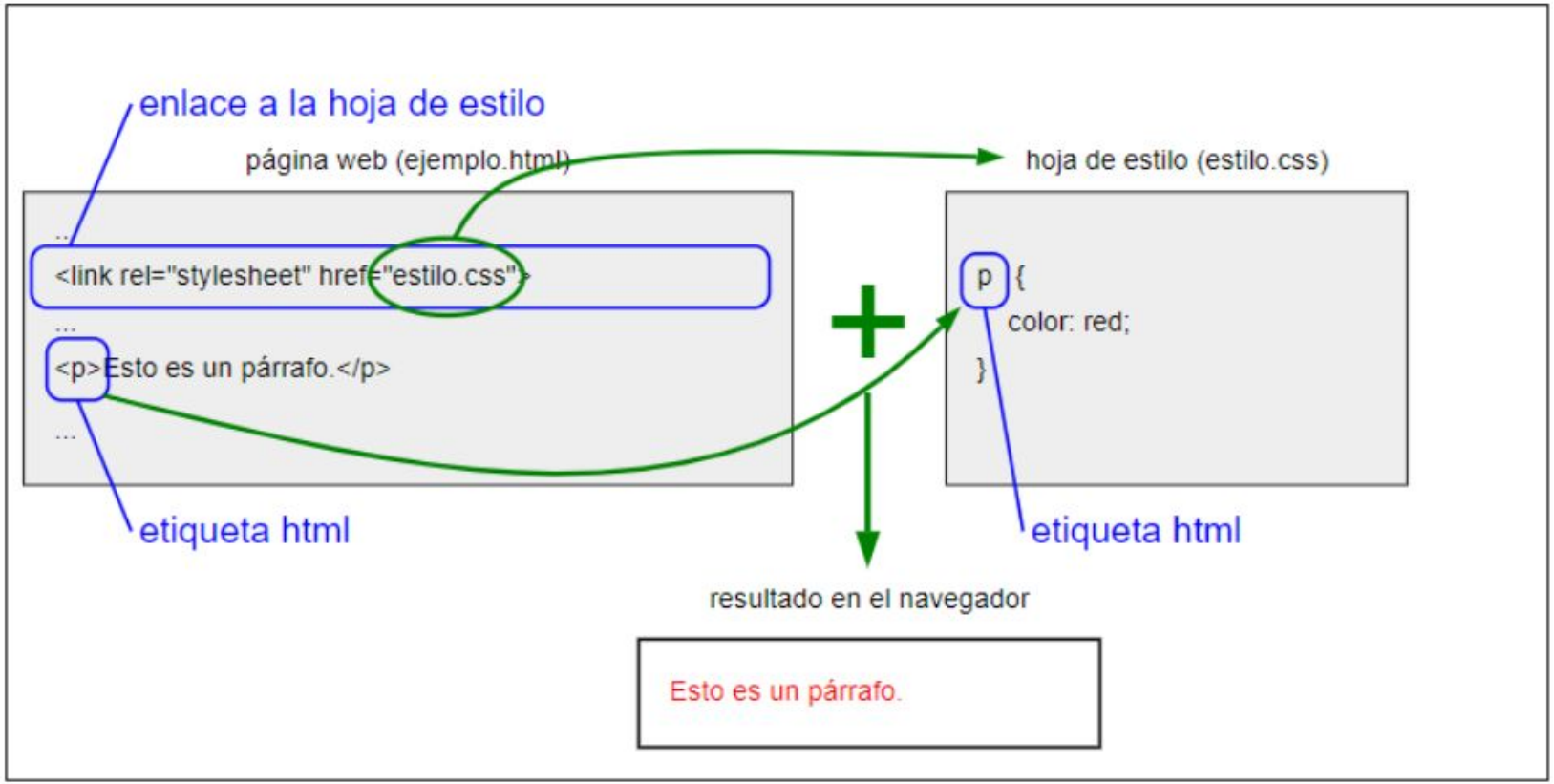
3.1 Introducció

Conceptes importants:

- Cada **regla** consisteix en un o més **selectors** i un **bloc** de declaració (o blocs d'estil).
- Cada **bloc d'estils** es defineix entre **claus**, i està format per una o diverses **declaracions d'estil**.
- Les **declaracions d'estil** tenen **propietats**, els quals tenen un **valor** determinat.
- Els **estils** s'apliquen als **elements** del document que compleixin amb el **selector** que els precedeix.



3.1 Introducció

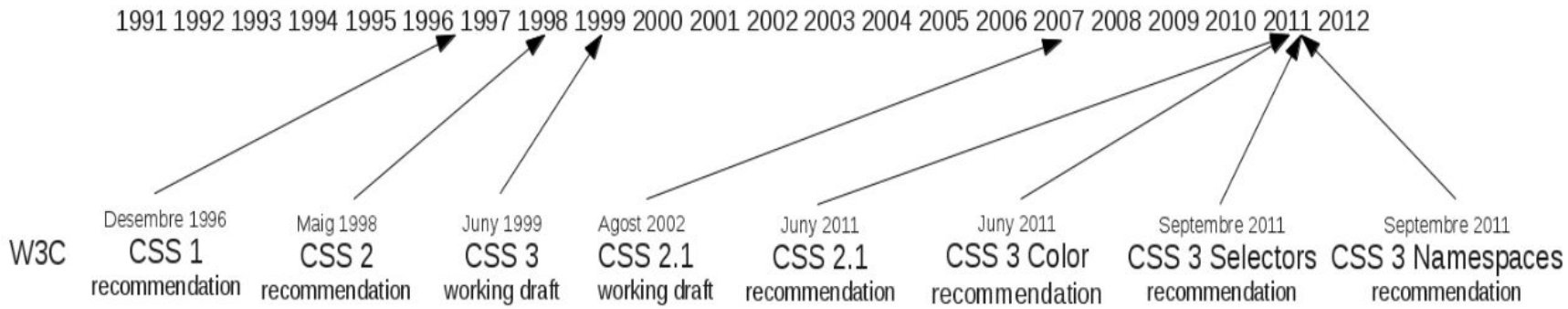


3.1 Introducció

Evolució històrica del CSS

- La primera versió del **CSS** o **Cascading Style Sheets**, va aparèixer en **1996.**
- Després el maig de l'any **1998** es va publicar l'estàndard **CSS versió 2.**
- El 2008, se'n va fer una revisió i es va publicar el CSS versió 2, revisió 1 i es va conèixer com CSS2.1.
- Actualment està en vigor la **versió 3**, el **CSS3**. En aquest cas l'especificació està dividida en **mòduls**, alguns dels quals ja han aparegut estàndards i en d'altres encara s'hi està treballant, és a dir, per una part tenim estàndards i per una altra una tecnologia en **continu desenvolupament.**

3.1 Introducció



Evolució de les versions de CSS

Com podeu observar a partir de la versió CSS3, els estàndards es tracten en mòduls per separat, com els **selectors** o **namespaces.**

20 years of CSS

Estat actual de CSS

3.1 Introducció

Avantatges de l'ús de fulls d'estil

Emprar fulls d’estil ens proporciona els avantatges següents:

- En el camp de **disseny**, el CSS és més **potent** que les marques de disseny que ofereix L'HTML.
- El CSS és un llenguatge **senzill**.
- Es poden especificar **diferents fulls d’estil per a un sol document HTML**. Per exemple, podem tenir l’estil per a la pàgina web quan es visita amb el navegador i l’estil per a quan volem imprimir aquesta pàgina.
- Els fulls d’estil es poden **reutilitzar** des de diferents documents HTML.

3.1 Introducció

Inconvenient de l'ús de fulls d'estil

El gran inconvenient dels fulls d'estil és que **no tots els navegadors es comporten de la mateixa manera davant del mateix full d'estil**.

Això es deu al fet que alguns navegadors no compleixen els estàndards establerts i, amb això, obliguen el programador a codificar diferents fulls d'estil (un per a cada navegador).

Tot i això, en els últims anys els navegadors cada vegada més, s'acosten més a complir els estàndards proposats.

3.2 La sintaxi

Un full d'estil és un conjunt de regles que defineixen l'estètica final dels documents HTML. Com ja hem dit abans aquestes regles estan formades per un selector i per un conjunt de declaracions.

- **Un selector:** Ens serveix per definir a quin o a quins elements volem aplicar les declaracions de la regla.
- **Una declaració:** Està formada per una **propietat** amb el seu **valor** associat. Les declaracions són les diverses característiques que han de complir els elements que concorden amb el selector. A cada propietat de les declaracions, s'hi posa un valor.

```
selector{
     declaració_1
         ...
     declaració_n
    }

on: la sintaxi de cada declaració_i és: propietat_i:valor_i;
```

3.2 La sintaxi

Exemple: si volem que tots els paràgrafs tenguin lletra de mida 10pt i un fons de color gris, definirem:

```
p {
   font-size: 10pt;
   background-color: gray;
    }
```

En aquesta regla, p és el selector i té dues declaracions: font-size:0pt; i background-color:gray.

En la primera declaració font-size: 10pt, font-size és la propietat i 10pt és el valor.

En la segona declaració background-color:gray, background-color és la propietat i gray és el valor.

3.2.1 Les regles arrova (@)

Les regles arrova @

Hi ha un conjunt de regles especials que s'anomenen regles arrova (at-rules en anglès o regles-at). Aquestes regles es caracteritzen perquè comencen pel caràcter **@**. Algunes d'aquestes regles (**@import** o **@namespace**) han d'aparèixer al principi de la fulla d'estil, i les altres (**@font-face**, **@media**, etc.), poden posar-se a qualsevol part del full d'estil.

Vegem quines són aquestes regles i com s'utilitzen:

- **@import:** La regla @import ens serveix per incloure, en el nostre full d'estil, fulls d'estil externs. Això és útil en determinats casos, per exemple, quan es treballa amb una web gran i complexa, l'arxiu CSS és torna molt gran i mal de manejar; en aquest cas seria interessant emprar @import.

Exemple:
Si, per exemple, volem incloure en el nostre full d'estil totes les propietats que hi ha en un document anomenat "nousestils.css", hem d'escriure la línia següent: @import "nousestils.css";

3.2.1 Les regles arrova (@)

Vegem quines són aquestes regles i com s'utilitzen (continuació):

- **@media:** La regla @media serveix per diferenciar per quin mitjà s'ofereixen les propietats que conté aquesta regla. Són les **Media Queries**) i són una de les grans avantatges de CSS3, ja que permeten saber quin sistema està visualitzant la pàgina web, i en funció d'això s'aplicaran unes regles d'estil o unes altres. És un dels recursos de què disposen els dissenyadors per fer els seus llocs responsives.

  La sintaxi genèrica és la següent:

```
@media mitjà {
   propietats
}

on mitjà pot ser print (per imprimir) o screen (per mostrar per pantalla), entre d'altres.
```

3.2.1 Les regles arrova (@)

Exemple: Volem que, quan imprimim el document HTML, la lletra tengui una mida de 12pt, però que, quan es vegi per pantalla, tengui una mida de 14pt. També volem que en els dos casos, l'alçada de la línia sigui d'1.4. En el nostre full d'estil hem d'introduir el codi següent:

```
@media print {
  body{ font-size: 12pt }
}

@media screen {
  body { font-size: 14pt }
}

@media screen, print {
  body { line-height: 1.4 }
}
```

Avui en dia, amb l'augment dels tipus de dispositius, fa que tenguem moltes mides de pantalles diferents (smartsphones, tablets, ordinadors, etc.), i aquesta regla és molt útil per aplicar diferents regles CSS segons el tipus de pantalla o la seva orientació. Aquest tipus de programació se l'ha anomenat Disseny Adaptatiu (en anglès *Responsive Web Design* o RWD), i a la combinació de la regla @media amb les condicions que podem afegir, se l'ha anomenat *media-queries*.

3.2.1 Les regles arrova (@)

Vegem quines són aquestes regles i com s'utilitzen (continuació):

- **@font-face:** Especifica una font no inclosa en el navegador, és a dir, permet definir una tipografia i importar-la pel seu ús a una pàgina web. Abans d'existir aquesta regla @, només es podia definir una llista de famílies en ordre descendent de prioritat amb la regla "font-family".

3.2.1 Les regles arrova (@)

Exemple: En aquest exemple, en la regla @font-face estem definit una font. Per tal de definir-la li posem un nom. Aquest nom es posa amb la propietat font-family i en aquest cas hem triat el nom DeliciousRoman. A més, és imprescindible especificar on es troba aquesta font, per tal que el navegador la pugui descarregar i emprar (això es fa amb la propietat src)

```
@font-face {
  font-family: DeliciousRoman;
  src: url("Delicious-Roman.otf");
              }

  p {
       font-family: DeliciousRoman, Helvetica, Arial, sans-serif
    }
```

Si hem definida la font, després la podrem emprar en qualsevol regla.

La seqüència de font dins l'atribut font-family, vol dir s'intentarà primer aplicar la primera font, si no es pot aplicar, se intentarà amb la segona, i així successivament.

3.2.1 Les regles arrova (@)

Vegem quines són aquestes regles i com s'utilitzen (continuació):

- **@charset:** Especifica quin és el joc de caràcters que farem servir dins del fitxer CSS:
  - @charset "UTF-8": Activa el joc de caràcters pel full d'estil a Unicode UTF-8.
  - @charset 'ISO-8859-15': Activa el joc de caràcters pel full d'estil Latin-9 (Llengües de l'oest d'Europa, amb el símbol de l'euro).

- **@supports:** Ens permet detectar si el navegador de l'usuari suporta o no les noves funcionalitats del CSS. El seu funcionament seria:

Exemple: El CSS aplica les regles només si el navegador suporta display: flex. En cas contrari ho ignora per complet.

```
/* si el navegador suporta display: flex */
@supports (display: flex) {
/* Llavors aplica les regles: */
    .element {
        display: flex;
        ... Més regles
    }
}
```

3.2.1 Les regles arrova (@)

Vegem quines són aquestes regles i com s'utilitzen (continuació):

- **@page:** La fem servir per modificar propietats CSS a l'hora d'imprimir un document, però hem de tenir en compte que només pot actuar sobre un nombre de propietats: sobre els marges, les línies vídues, línies orfes i els salts de pàgina.

Exemple:

```
@page {
  margin-left: 3cm;
}
```

3.2.1 Les regles arrova (@)

Vegem quines són aquestes regles i com s'utilitzen (continuació):

- **@page:** La fem servir per modificar propietats CSS a l'hora d'imprimir un document, però hem de tenir en compte que només pot actuar sobre un nombre de propietats: sobre els marges, les línies vídues, línies orfes i els salts de pàgina.

Exemple:

```
@page {
  margin-left: 3cm;
}
```

3.2.2 Els comentaris

Com en tot llenguatge, la llegibilitat és imprescindible si treballen en grup. És per això que CSS ofereix una manera de comentar els fulls d'estil. Si en un full d'estil volem posar comentaris destinats a l'aclariment del codi CSS, hem de fer servir la sintaxi següent:

```
/* comentaris */
```

Exemple:

```
/* Estil per a totes les capçaleres del document */

h1 {
  text-align: center; /* Text centrat per destacar la importància de la capçalera*/
  color: red;         /* Color de lletra vermella per emfatitzar el text */
}
```

3.2.3 Ubicació del CSS

Veurem 3 maneres diferents d'integrar fulls d'estil en documents HTML:

- La primera consisteix a utilitzar el atribut **style** a la pròpia línia.

```
<body>
<p style="font-family: Verdana; font-size: medium;">HOLA MON</p>
</body>
```

S'utilitza per a indicar l'estil particular de la línia (no utilitza claus).

3.2.3 Ubicació del CSS

- La segona consisteix a incloure el codi css a la capçalera del document, dins de l'etiqueta <style>. Es diu **CSS intern**.

```
<html>
     <head>
          <style type="text/css">
               body {font-family: Courier New;}
               hl {font-family: Arial; font-size: x-large;}
               p {font-family: Verdana; font-size: medium;}
          </style>
     </head>
     <body>
          <hl>Tipo de fuente Arial y tamaño grande</hl>
          <p>Tipo de fuente Verdana y tamaño medio</p>
     </body>
</html>
```

Es fa servir per indicar els estils propis d'aquesta pàgina, que es complementen amb els estils globals del lloc web, i solen definir-se en un fitxer css extern.

3.2.3 Ubicació del CSS

- La tercera consisteix a crear un fitxer css extern i vincular-ho al document mitjançant un enllaç **<link>** o **@import**. Es diu **CSS extern**.

A través de **<link>**

```
<head>
    <link rel="stylesheet" type="text/css" href="estilos.css">
</head>
```

O a través de **@import**

```
<head>
    <style type="text/css">
        @import "estilos.css";
    </style>
</head>
```

estilos.css

```
body {margin: 0px;}
td {color: #000000;
font-size: 12px;}
a {color: #FF6600;
font-weight: bold;}
```

3.2.3 Ubicació del CSS

Prioritat

La prioritat de les regles augmenta com més particular sigui la regla, és a dir:

1. css en línia (+ prioritat)
2. css intern
3. css extern (- prioritat)

Dins de cadascuna d'aquestes formes, en cas de repetició és la darrera regla la que s'aplica, per exemple en aquest cas, s'aplicaria tipus de font Verdana:

```
p {font-family: Arial;
   font-family: Verdana;}
```

Podem augmentar la prioritat utilitzant la paraula reservada **!important** , a l'exemple anterior permetria aplicar la font Arial, tot i no ser la darrera regla. És important conèixer què existeix aquesta regla però no l'empris si no en tens necessitat. (Explicació !important)

3.3 Atributs globals: class i id

Els atributs "**id**" i "**class**" serveixen com a identificadors a HTML, són molt importants perquè els elements HTML puguin ser apuntats i personalitzats amb estils mitjançant CSS.

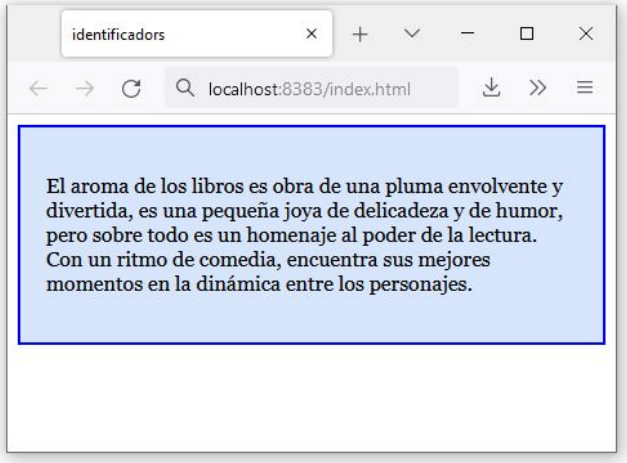
- **Atribut id**

  L'atribut global id, defineix un identificador únic, és a dir, **ha de ser únic en tot el document**. El seu propòsit és identificar l'element quan es vincula.

  El valor d'aquest atribut no ha de contenir espais en blanc (espais, tabulacions, etc.).

3.3 Atributs globals: class i id

```
<html>
  <head>
      <meta charset="utf-8" />
      <title>identificadors</title>
      <style>
         #principal {
            border: 1px solid blue;
            padding: 20px;
            background-color: #d8e6fd;
            font-family: Georgia, 'Times New Roman', Times, serif;
         }
      </style>
  </head>
  <body>
      <div id="principal">
        <p>El aroma de los libros es obra de una pluma envolvente y
          divertida, es una pequeña joya de delicadeza y de humor,
          pero sobre todo es un homenaje al poder de la lectura. Con
          un ritmo de comedia, encuentra sus mejores momentos en la
          dinámica entre los personajes.</p>
      </div>
  </body>
</html>
```



3.3 Atributs globals: class i id

- **Atribut class**

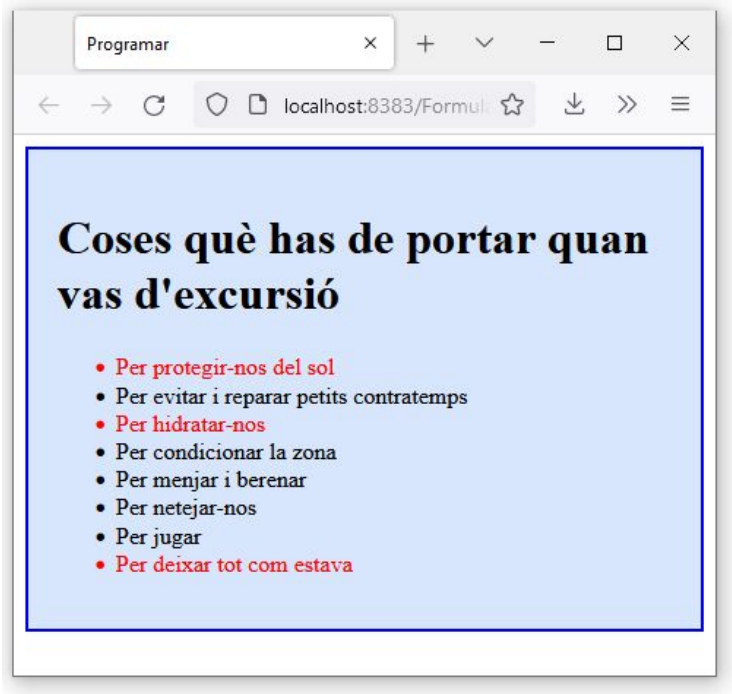
  Si es vol donar o definir un estil distint a elements amb la mateixa etiqueta, es pot fer emprant l'atribut "class". Aquest atribut es pot assignar a qualsevol element d'una pàgina web.

  Encara que l'especificació no posa requisits en el nom de les classes, s'anima als desenvolupadors web a **utilitzar noms que descriguin el propòsit semàntic de l'element**, en lloc de la presentació de l'element.

  Els noms semàntics segueixen sent lògics encara que la presentació de la pàgina canviï.

3.3 Atributs globals: class i id

```
<html>
  <head>
       <meta charset="utf-8" />
       <title> Programar </title>
       <style>
       #principal {
              border: 2px solid blue;
              padding: 20px;
              background-color: #d8e6fd;
       }
       .important {
              color: red;
       }
       </style>
  </head>
  <body>
       <div id="principal">
       <h1> Coses què has de portar quan vas d'excursió</h1>
       <ul>
              <li class="important">Per protegir-nos del sol</li>
              <li>Per evitar i reparar petits contratemps</li>
              <li class="important">Per hidratar-nos</li>
              <li>Per condicionar la zona</li>
              <li>Per menjar i berenar</li>
              <li>Per jugar</li>
              <li class="important">Per deixar tot com estava</li>
       </ul>
       </div>
  </body>
</html>
```



3.4 Selectors / Pseudoclasses/ Pseudoelements

Al fulls d'estil (CSS), els **selectors** són la part de les regles que indiquen al navegador a quins elements s'ha d'aplicar les propietats incloses dins de les declaracions.

Tractarem els següents:

- Selectors bàsics
  - Selector universal
  - Selectors de tipus o etiqueta
  - Selectors de classe
  - Selectors de Id
  - Selector descendent
- Selectors combinadors
  - selector de fills
  - selector de germans adjacents
  - selector general de germans
- Selectors d'atributs
- pseudoclasses
- pseudoelements

3.4 Selectors / Pseudoclasses/ Pseudoelements

Selector universal

S'utilitza per a seleccionar tots els elements de la pàgina.

Se indica mitjançant un asterisc (*).

No s'empra habitualment, ja que és difícil que un mateix estil es pugui aplicar a tots els elements d'una pàgina.

Exemple:

```
* { margin: 0px }
```

3.4 Selectors / Pseudoclasses/ Pseudoelements

Selectors de tipus o etiqueta

Selecciona tots els elements de la pàgina on l'etiqueta indicada, coincideix amb el valor del selector.

Per utilitzar aquest selector, només és necessari indicar el nom d'una etiqueta HTML (sense els caràcters **<** i **>**) corresponent als elements que es volen seleccionar.

Exemple: `p {font-family: Verdana; color: red;}`

Si es volen aplicar els mateixos estils a dues o més etiquetes diferents, es poden encadenar els selectors, separats per comes.

Exemple: `h1, p {font-family: Verdana; color: red;}`

3.4 Selectors / Pseudoclasses/ Pseudoelements

Selectors de classe

S'utilitzen quan volem aplicar el mateix estil a tots els elements què tenen definida una classe.

Exemple: `.important { color: red;}`

També serveix per aplicar un estil concret a elements què ja tenen estil definit per al seu tipus.

Exemple:

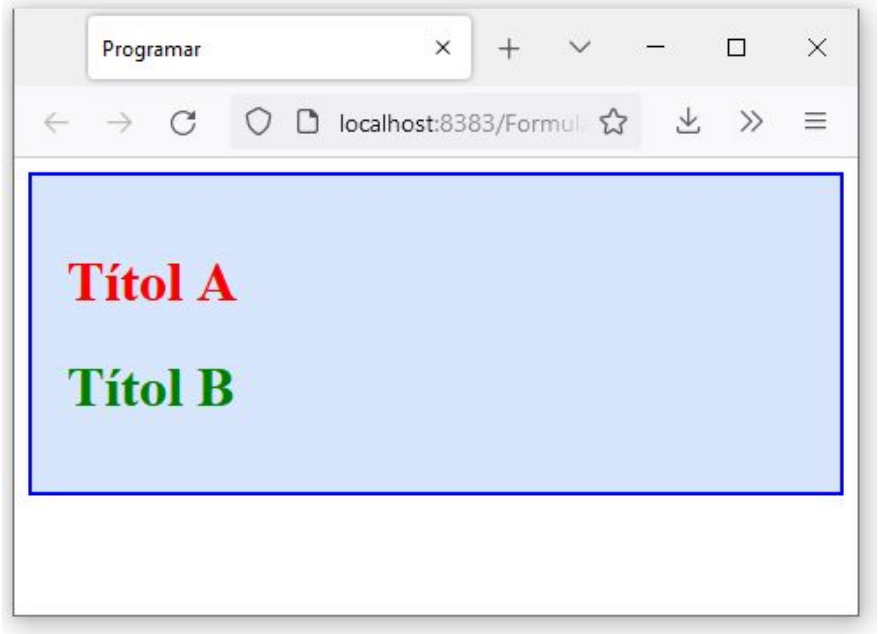
```
h1 { color: red;}
.important { color: green;}

<h1> Títol A </h1>
<h1 class=”important”> Títol B </h1>
```

També es pot fer què s’apliquin a elements què tinguin dues classes o més

3.4 Selectors / Pseudoclasses/ Pseudoelements

```
<html>
  <head>
     <meta charset="utf-8" />
     <title> Programar </title>
     <style>
     #principal {
          border: 2px solid blue;
          padding: 20px;
          background-color: #d8e6fd;
     }
     h1 { color: red; }
     .important { color: green; }
     </style>
  </head>
  <body>
     <div id="principal">
     <h1> Títol A </h1>
     <h1 class="important"> Títol B </h1>
     </div>
  </body>
</html>
```



3.4 Selectors / Pseudoclasses/ Pseudoelements

Selectors de classe

Si indiquem un element abans de la classe, afecta sols als elements d'aquest tipus que tinguin la classe definida. Exemple:

```
p.important{ color: red}

<p class="important"> Això és un text important i gran </p>
```

També es pot fer què s'apliquin a elements què tinguin dues classes o més indicant el nom de les classes juntes. Exemple:

```
.important.gran{ color: red; font-size: 30px;}

<p class="important gran"> Això és un text important i gran </p>
```

3.4 Selectors / Pseudoclasses/ Pseudoelements

Selectors de Id

A vegades, serà necessari aplicar estils **a un únic element** de la pàgina. Encara que es podria emprar un selector de classe per aplicar estils a un únic element, els selectors id són més eficients en aquest cas.

Aquests selectors permeten seleccionar un element de la pàgina a través del valor del seu atribut **id**. Aquests tipus de selectors només seleccionen un element de la pàgina perquè el valor de l'atribut id no es pot repetir dins dos elements diferents de la una mateix pàgina.

La sintaxi és molt similar a la dels selectors de classe, amb la diferència que utilitzen el símbol **#** en lloc del punt.

Exemple:

```
#titolB { color: red;}

<h1> Títol A </h1>
<h1 id=”titolB”> Títol B </h1>
```

3.4 Selectors / Pseudoclasses/ Pseudoelements

Selector descendent

Selecciona els elements que es troben dins d'altres elements.

Els selectors descendents sempre estan formats per dos o més selectors **separats entre sí per espais en blanc**, on el darrer selector indica l'element on s'apliquen els estils i tots els selectors anteriors indiquen el lloc on s'ha de trobar aquest element.

Podem aplicar el selectors descendents a elements, ids, classes.

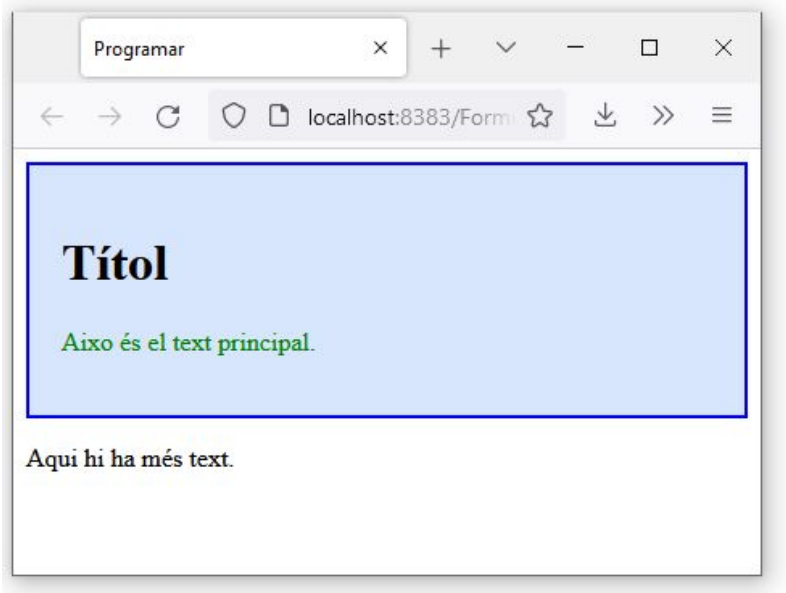
Exemple:

```
div p {...}

.class1 .class2 {...}
```

3.4 Selectors / Pseudoclasses/ Pseudoelements

```
<html>
  <head>
      <meta charset="utf-8" />
      <title> Programar </title>
      <style>
      #principal {
            border: 2px solid blue;
            padding: 20px;
            background-color: #d8e6fd;
      }
      div p { color: green; }
      </style>
  </head>
  <body>
      <div id="principal">
            <h1> Títol </h1>
            <p> Aixo és el text principal. </p>
      </div>
      <p> Aqui hi ha més text. <p>
  </body>
</html>
```



3.4 Selectors / Pseudoclasses/ Pseudoelements

Selector de fills

Serveix per seleccionar el primer descendent d'un element, és a dir, el "fill", excloent de la selecció a la resta de descendents.

Un selector fill es construeix amb el signe >, com ara:

```
p > a {font-size: 50px}
```

Aplicada al codi:

```
<p>Textl</p>
<p><a>Text2</a></p>
<p><span class="net"><a>Text3</a></span></p>
```

Visualitzaria només el Text2 a mida 50px, ja que Text3 sería descendent de 2n nivell,però no seria fill.

3.4 Selectors / Pseudoclasses/ Pseudoelements

Selector de germans adjacents

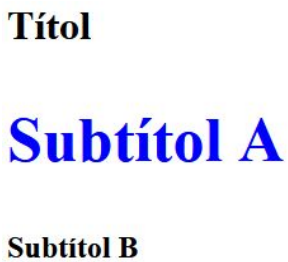
S'empra per seleccionar elements que són "germans" (tenen el mateix pare) i són adjacents (consecutius) al codi.

Es construeix amb el signe **+**, de manera que una regla com a E + M {declaració;}, selecciona a tots els elements de tipus M, tals que E i M tenen el mateix pare i E precedeix immediatament a M al codi (tret de comentaris).

```
h1 + h2 {font-size: 50px; color:blue}
```

Aplicada al codi:

```
<body>
    <h1>Títol</h1>
    <h2>Subtítol A</h2>
    <h2>Subtítol B</h2>
</body>
```



3.4 Selectors / Pseudoclasses/ Pseudoelements

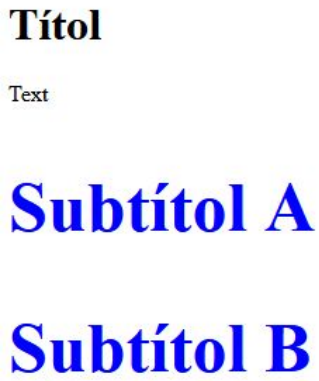
Selector general de germans

El combinador "~", selecciona germans, és a dir, que el segon element segueix al primer (no necessàriament de forma immediata), i els dos tenen el mateix pare.

```
h1 ~ h2 {font-size: 50px; color:blue}
```

Aplicada al codi:

```
<body>
    <h1>Títol</h1>
    <p> Text </p>
    <h2>Subtítol A</h2>
    <h2>Subtítol B</h2>
</body>
```



3.4 Selectors / Pseudoclasses/ Pseudoelements

Selector d’atributs

El selector d'atributs pot seleccionar de 4 maneres diferents:

- **[atribut]**: selecciona els elements que tinguin aquest *atribut* independentment del valor que prengui.
- **[atribut = valor]**: selecciona els elements que tinguin l'*atribut* amb el valor especificat.
- **[atribut ~= valor]**: selecciona els elements que tinguin l'atribut que té la paraula *valor*.
- **[atribut |= valor]**: selecciona els elements que tinguin l'*atribut* i que la paraula comenci amb el *valor* especificat.
- **[atribut ^= valor]**: selecciona els elements què tenen un *atribut* que comença amb la subcadena *valor*.
- **[atribut $= valor]**: selecciona els elements què tenen un *atribut* que acaba amb la subcadena *valor*.
- **[atribut *= valor]**: selecciona els elements què tenen un *atribut* que conté la subcadena *valor*.

3.4 Selectors / Pseudoclasses/ Pseudoelements

Selector d’atributs

Exemple:

[target]	Selecciona tots els elements què tenen definit un attribut target.
[target=_blank]	Selecciona tots els elements amb target="_blank"
[title~=floer]	Selecciona tots els elements amb un títol que tingui la paraula "flor"
[lang|=es]	Selecciona tots els elements amb l’atribut lang que començ per "es"
a[href^="https"]	Selecciona els elements a amb atribut href que comenci per "https"
a[href$=".pdf"]	Selecciona els elements a amb atribut href que acabi en ".pdf"
a[href*="marca"]	Selecciona els elements a amb atribut href que contingui la subcadena “marca”

3.4 Selectors / Pseudoclasses/ Pseudoelements

pseudoclasses

S'utilitzen per seleccionar elements en funció, no del codi com fins ara, sinó del comportament posterior de l'usuari (**esdeveniments**), per exemple, en funció dels enllaços que visita, el moviment del ratolí, etc.

Les pseudoclasses són:

- **:active** S'aplica als elements que estan sent activats, per exemple, mentre el usuari prem el botó esquerre del ratolí.
- **:checked** S'aplica als elements què estan checked. Com els <input>.
- **:disabled** S'aplica als elements deshabilitats.
- **:empty** S'aplica als elements què no tenen fills.
- **:enabled** S'aplica als elements habilitat (not disabled).
- **:first-child** Selecciona elements què són el primer fill del seu pare.
- **:first-of-type** Selecciona elements què són el primer element d'aquest tipus fill del seu pare.
- **:focus** S'aplica a un element que té el focus, per exemple, un quadre d'entrada en un formulari.

3.4 Selectors / Pseudoclasses/ Pseudoelements

pseudoclasses

Les pseudoclasses són (continuació):

- **:hover** S'aplica a un element seleccionat per l'usuari però sense activar-lo, com per exemple en situar el ratolí sobre un objecte sense fer clic.
- **:in-range** Selecciona els elements què tenen un valor dins un rang.
- **:invalid** Selecciona tots els elements amb un valor invalid.
- **:lang(*language*)** Selecciona tots els elements amb un atribut lang que comenci per *language*.
- **:last-child** Selecciona tots els elements què són el darrer fill del seu pare.
- **:last-of-type** Selecciona tots els elements què són el darrer element del seu tipus del mateix pare.
- **:link** (s'aplica als enllaços mai visitats)
- **:not(selector)** Selecciona tots els elements què no són un element indicat amb *selector*.
- **:nth-child(n)** Selecciona tots els elements què són el nth fill del seu pare.

3.4 Selectors / Pseudoclasses/ Pseudoelements

pseudoclasses

Les pseudoclasses són (continuació):

- **:nth-last-child(n)** Selecciona tots els elements què són el nth fill del seu pare contant des del darrer fill.
- **:nth-last-of-type(n)** Selecciona tots els elements què són nth element del seu pare contant des del nth fill.
- **:only-of-type** Selecciona tots els elements què són l'únic element d'aquest del seu pare.
- **:only-child** Selecciona tots els elements què són l'únic fill del seu pare.
- **:optional** Selecciona els elements què no tenen l'attribut required.
- **:out-of-range** Selecciona els elements què tenen un valor fora del rang especificat.
- **:read-only** Selecciona els elements amb l'atribut readonly especificat.
- **:read-write** Selecciona els elements què no tenen especificat l'atribut readonly.
- **:required** Selecciona els elements què no tenen especificat l'atribut required.

3.4 Selectors / Pseudoclasses/ Pseudoelements

pseudoclasses

Les pseudoclasses són (continuació):

- **:root** Selecciona l'element arrel del document.
- **:target** Selecciona els elements amb un ID equivalent al fragment la URL (Ex: pag.html#id).
- **:valid** Selecciona els elements amb un valor valid.
- **:visited** Selecciona els enllaços visitats.

Exemples: https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-classes

3.4 Selectors / Pseudoclasses/ Pseudoelements

pseudoelements

Les pseudoclasses fan referència a la posició de l'element complet en el document, mentre que els pseudoelements fan referència a la posició de determinades parts d'un element en el document.

Els pseudoelements són:

- **::after** Inserta contingut després de cada element. Exemple
- **::before** Insert contingut davant de cada element. Exemple
- **::first-letter** Selecciona la primera lletra de cada element. Exemple
- **::first-line** Selecciona la primera línia de cada element. Exemple
- **::selection** Selecciona la part d'un element què és seleccionada per l'usuari. Exemple
- **::marker** Selecciona el marcador d'un element d'una llista. Exemple
- **::placeholder** Selecciona el placeholder d'un element. Exemple