

# Programació web estàtica

Mònica Ramírez Arceda

**Xarxes d'àrea local**



# Índex

<b>Introducció</b>	7
<b>Objectius</b>	8
<b>1. Introducció a la programació web estàtica</b>	9
1.1. Funcionament del servei web	9
1.1.1. Classificacions de la programació web	10
1.2. Llenguatges de marques	11
1.2.1. L'HTML	11
1.2.2. L'XML	11
1.2.3. La DTD	14
1.2.4. L'XHTML	16
1.3. El CSS	17
1.4. Validació	17
<b>2. Els llenguatges de marques: (X)HTML</b>	18
2.1. Estructura d'un document (X)HTML	18
2.1.1. La declaració XML i la DTD	18
2.1.2. L'element arrel <code>&lt;html&gt;</code>	19
2.1.3. La capçalera i el cos	19
2.1.4. Comentaris	20
2.2. Marques del cos del document	21
2.2.1. Tipus d'elements	21
2.2.2. Atributs genèrics	21
2.3. Marques de text	22
2.3.1. Paràgrafs	23
2.3.2. Els encapçalaments	23
2.3.3. Èmfasi	24
2.3.4. Citacions	25
2.3.5. Salts de línia	26
2.3.6. Separadors	26
2.3.7. Acrònims i abreviatures	27
2.3.8. Text preformatat	27
2.3.9. Codi de programació	28
2.3.10. Divisions sense atributs semàntics	28
2.3.11. Marques “antiquades”	29
2.4. Enllaços	29
2.4.1. Enllaços a pàgines externes	30
2.4.2. Enllaços a pàgines locals	30
2.4.3. Àncores	31
2.5. Imatges	32

2.6. Llistes .....	33
2.6.1. Llistes ordenades .....	33
2.6.2. Llistes desordenades .....	33
2.6.3. Llistes de definició .....	34
2.6.4. Ennuament de llistes .....	35
2.7. Taules .....	35
2.7.1. Expansió de files i columnes .....	37
2.8. Formularis .....	39
2.8.1. Propietats d'un formulari .....	41
2.8.2. Etiquetes d'un formulari .....	42
2.8.3. Atributs genèrics dels controls d'un formulari .....	43
2.8.4. Camps de text .....	43
2.8.5. Camps de contrasenya .....	43
2.8.6. Botons d'opció .....	43
2.8.7. Caselles de verificació .....	44
2.8.8. Llistes de selecció .....	44
2.8.9. Àrees de text .....	46
2.8.10. Botons .....	47
2.8.11. Controls ocults .....	48
2.9. Marcs .....	48
2.9.1. Estructura d'un web fet amb marcs .....	49
2.9.2. Ennuament de marcs .....	51
2.9.3. Modificació de pàgines d'altres marcs .....	51
2.9.4. El problema dels marcs .....	52
2.10. Mapes .....	53
2.10.1. Mapes gestionats des del client .....	53
2.10.2. Mapes gestionats des del servidor .....	55
2.11. Objectes .....	55
2.11.1. Tipus MIME .....	57
2.11.2. Connectors .....	58
2.11.3. Exemples d'incorporació d'objectes genèrics en una pàgina web .....	59
<b>3. Els fulls d'estils: CSS .....</b>	<b>61</b>
3.1. Ubicació dels estils .....	62
3.2. Sintaxi bàsica de CSS .....	63
3.2.1. Les regles arrova .....	64
3.2.2. Comentaris .....	65
3.3. Cascada i herència .....	65
3.3.1. Cascada .....	65
3.3.2. Herència .....	66
3.4. Selectors .....	66
3.4.1. Selectors de tipus .....	67
3.4.2. Selectors de classes .....	67
3.4.3. Selectors d'identificador .....	68
3.4.4. Selectors d'atribut .....	69

3.4.5. Selector universal .....	70
3.4.6. Selectors de descendents .....	70
3.4.7. Selectors de fills .....	71
3.4.8. Selectors de germans adjacents .....	72
3.4.9. Agrupament de selectors .....	72
3.5. Pseudoclasses i pseudoelements .....	73
3.5.1. Pseudoclasses .....	73
3.5.2. Pseudoelements .....	75
3.6. Propietats bàsiques de format .....	76
3.6.1. Mesures i colors .....	76
3.6.2. Fonts .....	76
3.6.3. Aspecte del text .....	78
3.6.4. Color i fons .....	78
3.7. El model de caixa .....	79
3.8. El format visual dels elements d'un document .....	82
3.9. Exemples pràctics amb CSS .....	84
3.9.1. Fer menús amb llistes .....	84
3.9.2. Maquetació de webs sense ús de taules .....	86



## Introducció

Un dels serveis més utilitzats a Internet és el servei web. Aquest servei ens ofereix la possibilitat de navegar per multitud de pàgines web, que estan formades per un conjunt de documents que poden tenir contingut textual, gràfic, àudio, animacions, etc.

Les tecnologies per confeccionar pàgines web són diverses, però en aquesta unitat didàctica ens centrarem en la més bàsica: la programació web estàtica. És a dir, aprendrem les tecnologies bàsiques per tal de confeccionar pàgines web senzilles, sense gaire interacció amb l'usuari, però atractives i accessibles per a aquest.

La unitat didàctica “Programació web estàtica” pretén iniciar l'alumne en el desenvolupament d'aplicacions web: s'aprèn a dissenyar aplicacions web estàtiques i se n'estudia tant l'estructura com l'estètica.

En el nucli d'activitat “Introducció a la programació web estàtica” es donen les nocions bàsiques d'allò que es coneix com a programació web. S'ofereixen els conceptes genèrics de com funcionen les aplicacions web i es fa una descripció genèrica breu dels llenguatges de marques més usats.

En el nucli d'activitat “Els llenguatges de marques: (X)HTML” s'analitza amb profunditat la família de llenguatges (X)HTML. Aquests llenguatges són els que ha d'utilitzar el dissenyador de la pàgina web per estructurar la informació que finalment es visualitza en el navegador. Els llenguatges (X)HTML permeten inserir, en els continguts, marques que els navegadors saben interpretar.

En el nucli d'activitat “Els fulls d'estils: CSS” es presenta el llenguatge CSS, que ens permet fer el disseny estètic de la interfície web.

Per aconseguir aquests objectius, heu de reproduir en el vostre ordinador tots els exemples que s'incorporen en el text, per a la qual cosa, en la secció “Recursos de contingut”, trobareu tots els arxius necessaris, a més de les activitats i els exercicis d'autoavaluació. També és aconsellable, per descomptat, llegir els annexos.

## **Objectius**

En acabar la unitat didàctica, heu de ser capaços del següent:

1. Identificar quan una pàgina web compleix els estàndards vigents.
2. Elaborar pàgines web estàtiques a partir del llenguatge de marques (X)HTML i dels fulls d'estils CSS, amb l'ajut d'eines específiques per crear pàgines web.



## 1. Introducció a la programació web estàtica

La programació web, l'objectiu de la qual és crear aplicacions web, engloba un ampli ventall de tecnologies que inclou l'edició de l'estructura de les pàgines, el disseny d'aquestes pàgines, la interacció amb l'usuari...

Tot i que podem tenir aplicacions web locals al nostre ordinador, l'estructura d'aquest tipus d'aplicacions està pensada perquè sigui una aplicació remota a la qual podem accedir gràcies a un programa (el navegador web).

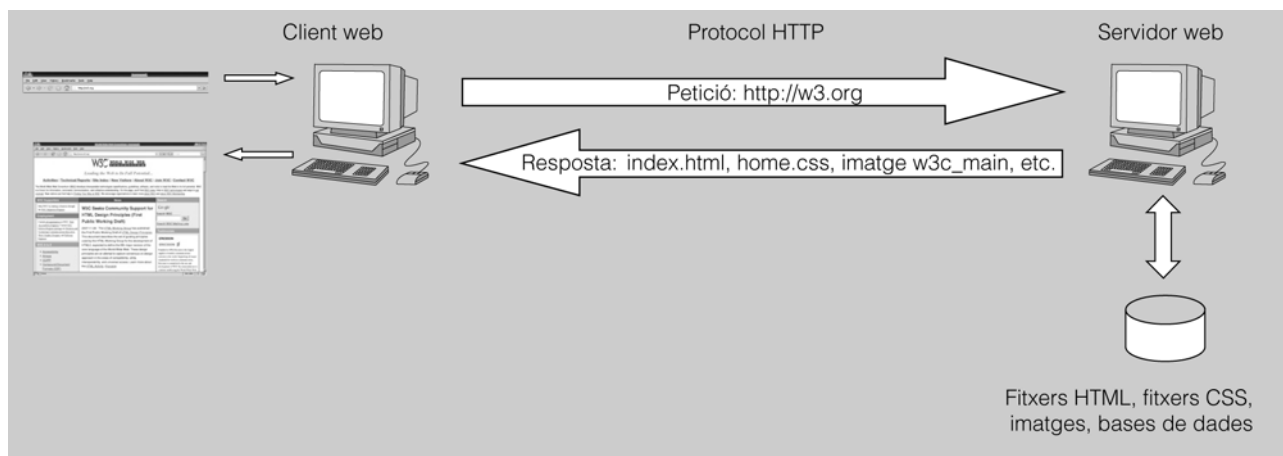
### 1.1. Funcionament del servei web

Quan teclegem una adreça web en el nostre programa navegador, obtenim un seguit d'informació que correspon a la petició feta: text, imatges, vídeos... Però què està passant realment a la xarxa? Quins elements han intervingut perquè haguem pogut obtenir aquesta informació?

L'esquema bàsic d'una connexió web consta de dos actors principals: el client i el servidor. Els programes navegadors fan la funció de client: és el navegador qui sol·licita fer la connexió web i això es produeix en el moment en què teclegem l'adreça **URL** (*uniform resource locator*). Els servidors són els ordinadors que tenen l'aplicació web i que atenen les peticions dels clients.



Figura 1. Esquema bàsic del funcionament del servei web



Quan s'estableix una comunicació entre el client (el navegador) i el servidor se segueixen els passos següents:

- 1) El client fa una petició en què demana veure una pàgina web.
- 2) El servidor processa la petició i envia al client els fitxers corresponents a aquesta petició.
- 3) El client rep la informació, la interpreta i la mostra de manera adequada perquè l'usuari pugui veure-la.

#### Navegadors

En l'actualitat disposem de diversos programes navegadors, com per exemple Firefox, Internet Explorer, Opera i Safari, entre d'altres.

En la figura 1 podeu veure simbolitzat aquest procés.

Ara bé, sempre que dos ordinadors es comuniquen entre si han d'usar un protocol de comunicacions per tal de poder entendre's. Aquest protocol ha de ser conegut per les dues màquines.

Un **protocol de comunicacions** determina com dues màquines o més es comuniquen entre elles i el format de la informació que es transmeten.

Els protocols que es fan servir per demanar i rebre pàgines web són l'**HTTP** (*hypertext transfer protocol*) i l'**HTTPS** (*secure hypertext transfer protocol*), depenent de si enviem la informació en clar o encriptada. Així doncs, la funció principal d'un servidor web és rebre peticions HTTP o HTTPS i servir les pàgines web corresponents.

Els fitxers que rep el client en la comunicació són de diversos tipus: fitxers HTML, fulls d'estils, imatges, fitxers amb codi JavaScript... Els navegadors, doncs, han de saber interpretar tots aquests tipus de fitxers.

### 1.1.1. Classificacions de la programació web

La programació web està en evolució constant i s'entén com a aplicació web tant una senzilla web estàtica on no hi ha interacció amb l'usuari com aplicacions complexes en què intervenen diverses tecnologies.

Una possible classificació de les aplicacions web que hi ha és la següent:

**1) Programació web estàtica.** És aquella en la qual no intervé cap llenguatge de programació, tot i que paradoxalment l'anomenem programació web estàtica. Per confeccionar-la només s'usen llenguatges de marques. Sempre que fem una petició a aquest tipus d'aplicació obtenim els mateixos resultats. Les tecnologies usades en la programació web estàtica són (X)HTML i CSS.

**2) Programació web dinàmica.** És aquella en què el contingut de la pàgina varia segons diversos factors (interactuació amb l'usuari que visita la pàgina, interacció amb una base de dades del servidor...). Segons si la programació es fa en el client o en el servidor, podem distingir entre:

**a) Programació web en el client.** Són les aplicacions web en què tot el codi de programació s'executa en el client. Les tecnologies usades són (X)HTML, CSS i JavaScript.

**b) Programació web en el servidor.** Són les pàgines web en què el servidor web executa codi de programació per obtenir la pàgina HTML resultant. Les tecnologies usades són (X)HTML, CSS, PHP, JSP...

#### AJAX

La tecnologia AJAX (*asynchronous JavaScript and XML*) és una barreja intel·ligent de la programació web en el client i de la programació web en el servidor. Aquesta barreja proporciona pàgines web força espectaculars.

Totes aquestes tècniques de programació web no són incompatibles i es poden usar conjuntament.

## 1.2. Llenguatges de marques

Un llenguatge de marques és una manera de codificar un document en què el text està acompanyat d'etiquetes (marques) que contenen informació addicional sobre l'estructura del text o la seva presentació.

### 1.2.1. L'HTML

El llenguatge de marques **HTML** (*hyper text markup language*) és el llenguatge de marques predominant de les aplicacions web actuals. Permet descriure tant l'estructura com la presentació dels documents que conformen l'aplicació web.

El llenguatge HTML ens permet marcar la nostra informació de manera que podem incorporar-hi encapçalaments, paràgrafs, llistes, enllaços, taules, formularis, imatges...

El llenguatge de marques HTML té, però, dos grans desavantatges:

- 1) El fet que es disposi de marques que serveixen per a la presentació del document fa que es barregi l'estructura i la presentació de la web. Això resta modularitat a la nostra aplicació.
- 2) No té normes molt estrictes de sintaxi.

Per solucionar aquests dos problemes va sorgir el llenguatge de marques XHTML (*extensible hyper text markup language*), que no és altra cosa que aplicar les normes del metallenguatge XML (*extensible markup language*) al ja existent llenguatge HTML.

### 1.2.2. L'XML

L'**XML** (*extensible markup language*) és un metallenguatge de marques que ens serveix per estructurar la informació semàntica, sense tenir en compte la seva aparença final.

L'XML no ens defineix quines marques podem usar, sinó com s'han d'usar. És a dir, ens diu quina sintaxi hem d'usar, siguin quines siguin les marques que usem.

Per exemple, si volem fer un símil amb les llengües usades normalment, podríem definir el metallenguatge de les llengües romàniques donant com a norma que hem d'escriure d'esquerra a dreta, que les frases han de tenir subjecte i predicat, i que podem usar certs signes de puntuació; no direm, però, quines són les paraules que podem usar, això ho definirà cadascuna de les llengües.

Un document XML està format per nodes. Aquests nodes poden ser de diferents tipus, tal com es pot veure en la taula 1, on es mostren alguns dels tipus de nodes més importants.

Taula 1. Tipus de nodes XML

Tipus d'element	Descripció	Sintaxi
Elements	Són les marques que permeten identificar i diferenciar les diferents parts de les dades.	<code>&lt;element&gt;&lt;/element&gt;</code> <code>&lt;element /&gt;</code>
Atributs	Serveixen per donar informació addicional a un element.	<code>&lt;element atribut="valor"&gt;&lt;/element&gt;</code>
Nodes de text	Són les dades del document.	<code>&lt;element&gt;text&lt;/element&gt;</code>
Entitats predefinides	Serveixen per representar caràcters especials.	<code>&amp;entitat;</code>
Comentaris	Serveixen per introduir comentaris per al programador	<code>&lt;!-- comentari --&gt;</code>

En la figura 2 es pot veure un exemple de document XML en què es diferencien els diferents tipus de nodes. En la figura 3 veiem una representació del mateix document en forma d'arbre.

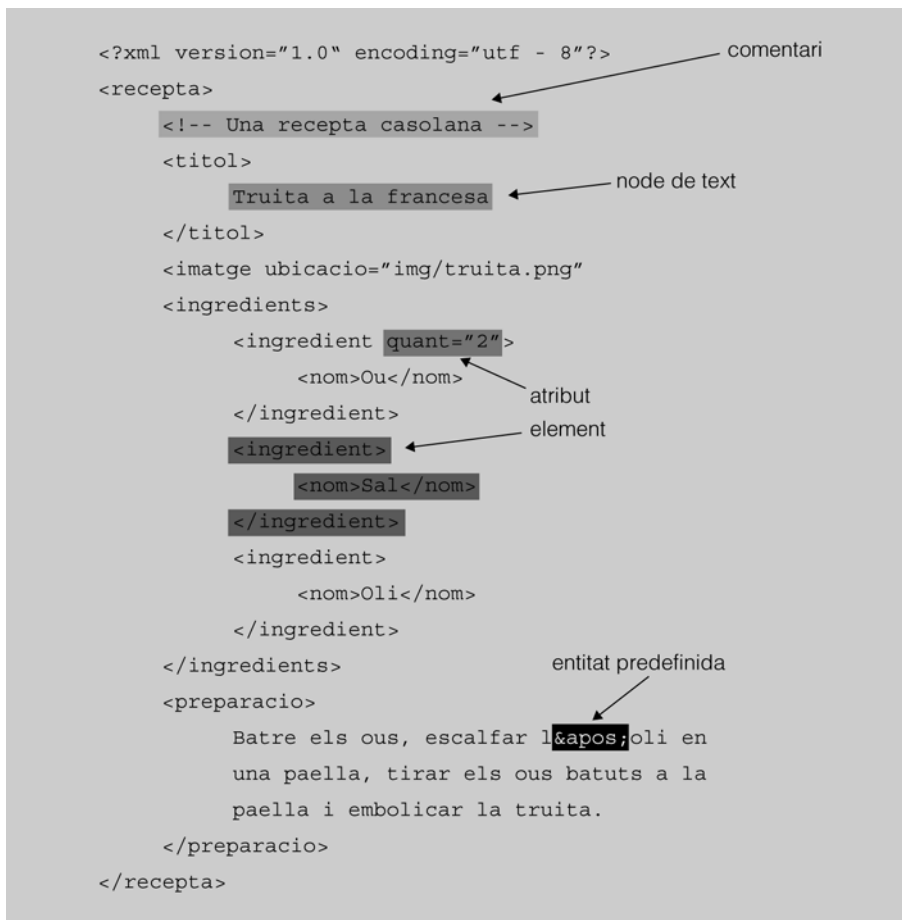
Els nodes d'un document XML estan relacionats entre si. Podem establir diferents relacions:

**Element arrel:** és l'element que engloba tots els altres. En l'exemple de la figura 2 l'element arrel és `<recepta>`.

**Pares i fills:** un element és fill d'un altre quan està immediatament dins seu. En l'exemple de la figura 2 l'element `<nom>` és fill de l'element `<ingredient>` i, per tant, l'element `<ingredient>` és el pare de l'element `<nom>`.

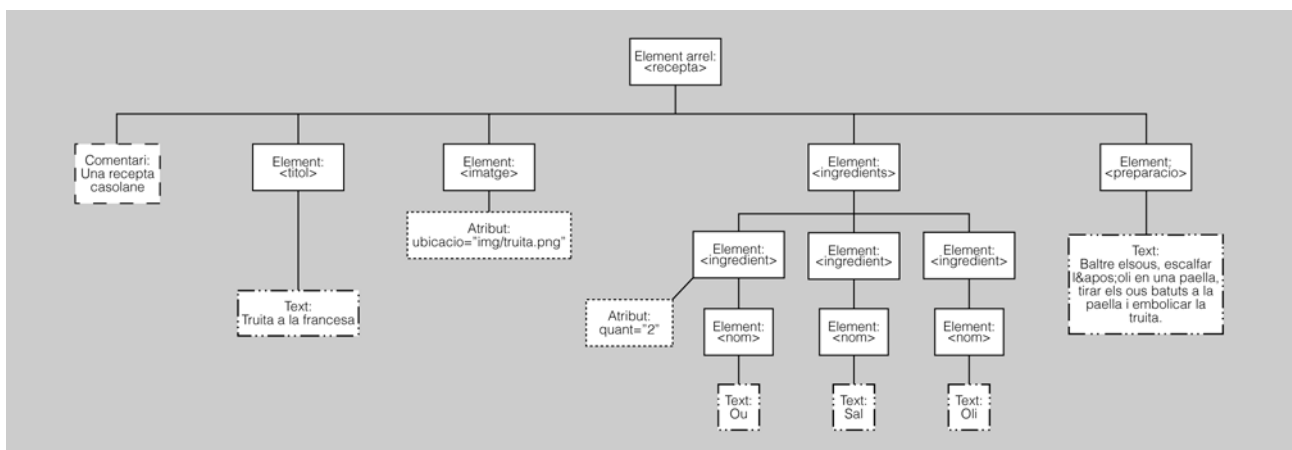
**Germans:** dos elements són germans quan tenen el mateix pare. En l'exemple de la figura 2 els elements `<titol>`, `<imatge>`, `<ingredients>` i `<preparacio>` són germans.

Figura 2. Document XML en què es mostren els diferents tipus de nodes



D'aquesta manera podem entendre un document XML com una estructura en forma d'arbre, tal com es mostra en la figura 3.

Figura 3. Representació d'un document XML en forma d'arbre



Els documents XML han de complir les normes de sintaxi següents:

- Els elements s'han de tancar (o amb la marca de tancament `<marca></marca>`, o tancant-se ella mateixa `<marca />`):
  - És correcte: `<titol>Truita a la francesa</titol>`
  - És incorrecte: `<titol>Truita a la francesa`

- Els elements han d'estar correctament ennuats.
  - És correcte: `<ingredient><nom>Oli</nom></ingredient>`
  - És incorrecte: `<ingredient><nom>Oli</ingredient></nom>`
- Tots els documents tenen un element arrel.

- És correcte:

```
<?xml version="1.0" encoding="utf-8"?>
<recepta>
  ...
</recepta>
```

- És incorrecte:

```
<?xml version="1.0" encoding="utf-8"?>
<recepta>
  ...
</recepta>
<observacions>
  ...
</observacions>
```

- Tots els atributs han d'anar entre cometes.

- És correcte: `<nom quant="2">Ou</nom>`
- És incorrecte: `<nom quant=2>Ou</nom>`

Diem que un document està **ben format** si respecta totes les regles de sintaxi de l'XML.

### 1.2.3. La DTD

Mentre que l'XML ens defineix la sintaxi d'un document, la DTD (*document type definition*) ens defineix quins són els elements i atributs permesos d'aquest document, l'ordre en què poden anar aquests elements i quins nodes poden anar dins de quins altres.

La **DTD** defineix els elements, atributs i entitats permesos en el document, i a més indica com podem combinar aquests nodes.

#### XML Schema

Tot i que l'ús de la DTD està molt estès, hi ha una altra manera de definir els nodes permesos i les seves característiques. L'alternativa és usar XML Schema. Aquesta alternativa ens proporciona més opcions i més flexibilitat.

Si ho comparem amb les llengües parlades, en el cas de les llengües romàniques, l'XML ens definiria que hem d'escriure d'esquerra a dreta. La DTD, en canvi, ens definiria quines paraules podem usar, és a dir, ens definiria el vocabulari. En la taula 2 s'exposen algunes característiques bàsiques de la sintaxi d'una DTD.

Taula 2. Sintaxi bàsica d'una DTD

Sintaxi	Descripció
<code>&lt;!ELEMENT nom-element (EMPTY)&gt;</code>	Defineix un element que no té contingut. En el document XML apareixerà amb una marca que es tancarà en ella mateixa ( <code>&lt;nom-element /&gt;</code> ).
<code>&lt;!ELEMENT nom-element (#PCDATA)&gt;</code>	Defineix un element que conté text.
<code>&lt;!ELEMENT nom-element (elem1, elem2,...)&gt;</code>	Defineix un element que conté altres elements; aquests elements fills poden portar un modificador que ens indica els cops que poden aparèixer: <i>elem</i> : apareix un cop <i>elem*</i> : apareix 0 cops o més <i>elem+</i> : apareix 1 cop o més <i>elem?</i> : apareix 0 cops o 1
<code>&lt;!ATTLIST nom-element nom-atribut tipus-atribut valor-per-defecte&gt;</code>	Defineix un atribut especificant el nom de l'atribut ( <i>nom-atribut</i> ) i el nom de l'element al qual pertany l'atribut ( <i>nom-element</i> ). Els <i>tipus-atribut</i> més usats són els següents: CDATA: text ( <i>valor1 valor2 ...</i> ): els valors de l'atribut només poden ser alguns dels valors enumerats Els <i>valor-per-defecte</i> més usats són: <i>valor</i> : especifica el valor que té l'atribut per defecte #REQUIRED: especifica que l'atribut és obligatori #IMPLIED: especifica que l'atribut no és obligatori

Per exemple, per al document XML definit en la figura 2 creariem un document `recepta.dtd` en què introduiríem la DTD següent:

```
<!ELEMENT recepta ( titol, imatge, ingredients, preparacio ) >

<!ELEMENT titol ( #PCDATA ) >

<!ELEMENT imatge EMPTY >
<!ATTLIST imatge ubicacio CDATA #REQUIRED >

<!ELEMENT ingredient ( nom ) >
<!ATTLIST ingredient quant CDATA #IMPLIED >

<!ELEMENT ingredients ( ingredient+ ) >

<!ELEMENT nom ( #PCDATA ) >

<!ELEMENT preparacio ( #PCDATA ) >
```

Però com li podem dir a un document XML quina DTD ha d'usar? Podem trobar diversos casos, com figuren en la taula 3.

Taula 3. Com es pot enllaçar el document XML amb la DTD que es vol usar

On	Sintaxi	Exemple
La DTD està definida en el mateix document XML.	<code>&lt;!DOCTYPE element-arrel [declaracions]&gt;</code>	<pre>&lt;!DOCTYPE recepta [ &lt;!ELEMENT recepta (titol,plat, ingredients,preparacio)&gt; ... &lt;!ELEMENT preparacio (#PCDATA)&gt; ]&gt; &lt;recepta&gt; ... &lt;/recepta&gt;</pre>

On	Sintaxi	Exemple
La DTD està definida en un fitxer extern.	<code>&lt;!DOCTYPE element-arrel SYSTEM "path/nomFitxer.dtd"&gt;</code>	<code>&lt;!DOCTYPE recepta SYSTEM "dtds/recepta.dtd"&gt;</code>
La DTD està definida en un fitxer extern públic.	<code>&lt;!DOCTYPE element-arrel PUBLIC "identificador" "url"&gt;</code>	<code>&lt;!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd"&gt; &lt;html&gt; ... &lt;/html&gt;</code>

Per exemple, si volguéssim usar una DTD que està descrita en un fitxer de nom `recepta.dtd`, hauríem d'incloure una línia amb la definició de la DTD, després de la declaració de la versió i codificació del document:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE recepta SYSTEM "recepta.dtd">
...
```



Per veure la sintaxi d'XML i DTD completa podeu consultar la secció "Annexos" del web d'aquest crèdit.

Diem que un document **valida** si està ben format i respecta totes les regles que especifica la DTD que té associada.

#### 1.2.4. L'XHTML

Una de les definicions que podem trobar al W3C (*World Wide Web Consortium*) del llenguatge de marques XHTML és la següent: l'**XHTML** (*extensible hyper text markup language*) és una versió més estricta i neta d'HTML, que neix precisament amb l'objectiu de reemplaçar l'HTML davant la seva limitació d'ús amb les cada cop més abundants eines basades en XML. XHTML estén HTML 4.0 combinant la sintaxi d'HTML, dissenyat per mostrar dades, amb la d'XML, dissenyat per descriure les dades.



##### W3C

El W3C (World Wide Web Consortium) és l'organització internacional més important que s'encarrega de desenvolupar i promoció els estàndards en el món web.

Hem de tenir en compte que quan confeccionem un document XHTML, tot i que algunes de les marques permeses ens permeten definir l'aspecte del document, hem d'intentar seguir la filosofia XML, és a dir, definir no més la semàntica del document, no l'aparença.

Per exemple, si volem que el text "Fauna i flora a la Garrotxa" sigui un encapçalament, usarem l'XHTML per fer-ho. Però dir que aquest text és un encapçalament no vol dir, per exemple, que el text "Fauna i flora de la Garrotxa" ha d'aparèixer en negreta i mida de lletra 20 (això seria la seva aparença i per definir-la usarem CSS, *cascade style sheets*).

Quan parlem de la família de llenguatges (**X**)HTML, ens estem referint a dues famílies de llenguatges: la família XHTML i la família HTML, que es distingeixen per complir o no les normes de sintaxi XML. Cadascuna



d'aquestes dues grans famílies tenen diversos llenguatges de marques definit cadascun d'ells per la seva DTD (*strict*, *transitional* o *frameset*). Aquests llenguatges són els següents:

- HTML 4.01 *Strict*
- HTML 4.01 *Transitional*
- HTML 4.01 *Frameset*
- XHTML 1.0 *Strict*
- XHTML 1.0 *Transitional*
- XHTML 1.0 *Frameset*
- XHTML 1.1 DTD

Si estem fent una web que manté l'estructura separada del disseny podem usar les DTD *Strict*, ja que aquestes DTD no permeten usar gairebé cap marca o atribut que faci referència a l'estètica de la web. En canvi, si la nostra web usa marques o atributs que fan referència a l'estètica de la pàgina, ens veiem forçats a usar les DTD *Transitional*. Finalment, si la nostra web està feta amb marcs hem d'usar les DTD *Frameset*.

### 1.3. El CSS

El **CSS** (*cascading style sheets*) és un mecanisme simple que descriu com es mostra un document en la pantalla o com s'imprimeix, o fins i tot, com es pronuncia la informació present en el document a través d'un dispositiu de lectura.

El CSS s'utilitza per donar estil a documents (X)HTML separant el contingut de la presentació. Els estils defineixen la forma de mostrar els elements HTML i XML.

#### Exemple d'aplicació de CSS

Si en un document (X)HTML hem definit que un text és un encapçalament, en el document CSS associat podem definir que els tipus de lletra dels encapçalaments és verdana de mida 20, que es visualitzen en negreta i que són de color vermell.

Si en el document (X)HTML hem definit que un text és un paràgraf, en el document CSS associat podem definir que els paràgrafs tindran fons gris i una vora.

### 1.4. Validació

Quan escrivim un document amb llenguatge (X)HTML ens hem d'assegurar que hem complert els estàndards. Per poder-ho fer W3C ens ofereix un servei de validació de pàgines, en el qual ens diuen si la nostra pàgina compleix els estàndards i, si no els compleix, quins errors hi hem comès.

També existeixen altres eines de validació, com el programa Tidy, que ens permeten validar el nostre document sense necessitat d'estar en línia.



#### Imatge de validació

Quan una web valida, acostuma a posar logos semblants als de la imatge per informar els seus visitants que compleix els estàndards.

## 2. Els llenguatges de marques: (X)HTML

Per confeccionar una pàgina web estàtica és necessari saber com es pot estructurar tota la informació que ha de contenir l'aplicació. Hem de saber, doncs, quins elements podem usar en la família de llenguatges (X)HTML, quina sintaxi utilitzen i en quins casos s'han d'usar.

### 2.1. Estructura d'un document (X)HTML

Qualsevol pàgina (X)HTML té una estructura bàsica. El primer que hem de fer és escollir quin llenguatge de la família (X)HTML volem usar, aquesta elecció ens limita els elements, els atributs i la sintaxi que hem d'usar en la confecció de la web, tot i que les marques més usades són comunes a tots els llenguatges de la família.



Trobareu el codi de tots els exemples en la secció "Recursos de contingut" del web d'aquest crèdit.

En l'exemple que mostrem a continuació s'ha escollit la *DTD (X)HTML 1.0 Strict*. La plantilla base que ha de tenir qualsevol document XHTML amb aquesta DTD és la següent:

```
<?xml version="1.0" encoding="utf-8"?  
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3c.org/TR/xhtml1/DTD/xhtml1-strict.dtd" >  
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ca" lang="ca">  
<head>  
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  
  <title>Títol de la web</title>  
</head>  
<body >  
  <!-- Aquí va el contingut de la web -->  
</body>  
</html>
```

#### 2.1.1. La declaració XML i la DTD

Si usem XHTML, hem d'especificar que el document és XML. Això es fa a la primera línia del document, indicant-ne també la versió i la codificació de caràcters que estem fent servir:

```
<?xml version="1.0" encoding="utf-8"?>
```

Si estiguéssim dissenyant un document HTML, aquesta línia l'hauríem d'ometre, ja que els documents HTML no són documents XML.

A tota pàgina web s'hauria d'especificar la DTD per tal de saber amb quin llenguatge de la família (X)HTML estem treballant. Aquesta especificació

ens permet validar la pàgina. En l'actualitat, però, trobem moltes pàgines web que no especifiquen quin DTD usen. Podem entendre que, o bé no n'usen cap o bé s'han oblidat d'especificar-la. En qualsevol dels dos casos, aquesta pàgina incompleix els estàndards proclamats pel W3C.

Si usem la *DTD XHTML Strict 1.0*, la primera línia del nostre fitxer *.html* és la següent:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3c.org/TR/xhtml1/DTD/xhtml1-strict.dtd" >
```



Per veure una llista de les DTD de la família (X)HTML podeu consultar en la web del crèdit la secció "Annexos" del web d'aquest crèdit.

### 2.1.2. L'element arrel <html>

Tot document XML ha de tenir un element arrel. En el cas de la família de llenguatges (X)HTML aquest element arrel és <html>. En el cas particular d'XHTML hem d'especificar, a més, que estem usant XHTML i la llengua del document:

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ca" lang="ca">
```

### 2.1.3. La capçalera i el cos

Tot document (X)HTML té dues seccions principals: la capçalera, marcada amb la marca <head> i el cos del document, marcat amb la marca <body>.

La capçalera conté altres elements que faciliten informació diversa de la pàgina, com el títol, l'enllaç amb altres fitxers relacionats amb el document (fulls d'estils, codi JavaScript...), metainformació del document...

Una capçalera mínima pot ser la següent:

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>La meua pàgina web</title>
</head>
```

La marca <meta> té diferents funcions, en aquest cas diem que el nostre document s'enviarà com a `text/html` codificat amb UTF-8.

#### Perquè està permès enviar documents XHTML 1.0 com a text/html?

L'XHTML és un format XML; això vol dir que parlant de manera estricta s'hauria d'enviar amb un tipus de mitjà relacionat amb l'XML (`application/xhtml+xml`, `application/xml`, o `text/xml`). Malgrat tot, l'XHTML 1.0 es va dissenyar amb molta cura, de manera que si el document està fet amb cura hauria de funcionar també en agents d'usuari pensats per a l'HTML llegat. Si se segueixen algunes pautes senzilles es pot fer que molts documents XHTML 1.0 funcionin en navegadors antics. Malgrat tot, els navegadors antics només interpreten el tipus de



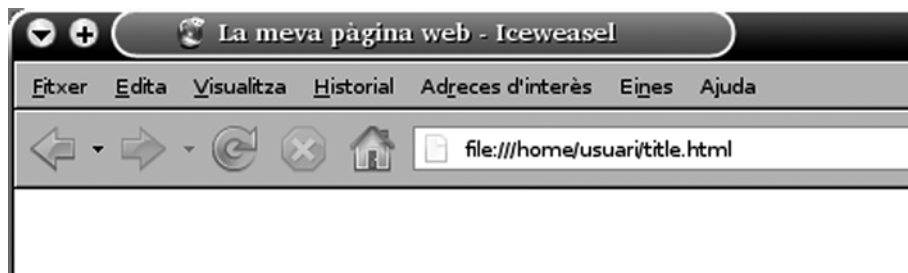
Per veure més usos de la marca <meta>, podeu consultar en la web del crèdit la secció "Annexos" del web d'aquest crèdit.

mitjà text/html, de manera que s'ha d'emprar aquest tipus de mitjà si se'ls envia documents XHTML 1.0. S'ha de tenir present que enviar documents XHTML als navegadors com a text/html vol dir que aquests navegadors veuran el document com a HTML i no com a XHTML.

S. Pemberton (ed.). *Preguntes més freqüents d'HTML i XHTML*.

La marca <title> ens serveix per posar el títol de la nostra web. El navegador mostra el contingut d'aquesta marca com a títol de la finestra o de la pestanya on es mostra el document, com podem veure en la figura 4.

Figura 4. Visualització del contingut de l'element <title>



Després de la capçalera, tenim l'element <body>, que conté tot el contingut de la web.

```
<body>
  <!-- Contingut de la web -->
</body>
```

La taula 4 ens mostra un resum de les marques que utilitzem per crear l'estructura mínima d'un document (X)HTML.

Taula 4. Elements de l'estructura d'un document (X)HTML

Marques	Descripció
<html> </html>	Element arrel de la pàgina web.
<head> </head>	Capçalera del document: espai destinat a contenir informació sobre el mateix document.
<title> </title>	Títol descriptiu de la pàgina web: normalment apareix a la barra del navegador, també és el text que s'emmagatzema en els marcadors del navegador (llista de preferits).
<meta />	Metainformació de la pàgina: podem posar diverses marques <meta>, que serviran per donar informació no visible del document.
<body> </body>	Cos de la web.

#### 2.1.4. Comentaris

En un document (X)HTML podem posar anotacions que no es veuran reflectides quan es miri la web amb el navegador, però que són útils per al desenvolupador web.

La sintaxi dels comentaris amb (X)HTML és la següent:

```
<!-- comentari -->
```

## 2.2. Marques del cos del document

Dins del cos del document (marcat amb la marca `<body>`) hem d'incloure tot el contingut de la web, és a dir, la informació que volem que vegi l'usuari final de l'aplicació web. Disposarem de diferents elements que, amb els seus atributs, ens permetran aconseguir aquest objectiu.

### 2.2.1. Tipus d'elements

Els elements (X)HTML fills de l'element `<body>` es poden classificar en dos tipus: els elements de bloc (*block elements*) i els elements de línia (*in-line elements*).

Els **elements de bloc** –per exemple, els títols, els paràgrafs, les llistes o les taules– són grans estructures que contenen altres elements de bloc, elements de línia, o text. Normalment, el navegador els mostra com a blocs independents i separa un bloc d'un altre amb una línia en blanc.

Els **elements de línia** –per exemple, els hiperenllaços, les citacions o les imatges– són petites estructures que representen o descriuen petits trossos de text o dades. Poden contenir només text o altres elements de línia. Normalment, el navegador mostra els elements de línia un darrere l'altre, en línia, dins de l'element de bloc que els conté.

### 2.2.2. Atributs genèrics

Les marques poden tenir atributs que acaben de definir l'element. Hi ha una sèrie d'atributs comuns a la majoria d'elements i que detallem a continuació:

- **Atributs *core*:** són atributs genèrics.
  - `id`: identifica l'element. No hi poden haver dos elements en una mateixa pàgina amb el mateix `id`.
  - `class`: serveix per associar un element a una classe. Més d'un element pot estar en una mateixa classe. Ens és molt útil a l'hora de definir els estils d'una pàgina, ja que podem especificar característiques comuns a tots els elements que pertanyin a la mateixa classe.
  - `style`: s'usa per definir l'estil de l'element.
  - `title`: s'usa per posar un text amb una informació breu de l'element. Molts navegadors mostren aquest text quan es passa per damunt de l'element.

- **Atributs de llenguatge:** ens especifiquen diferents característiques del llenguatge de l'element.
  - `lang`: especifica el llenguatge de l'element.
  - `dir`: especifica la direcció de lectura.
- **Atributs d'event:** són atributs que ens serveixen per associar una acció per fer quan es produeix alguna interacció de l'usuari cap a l'element que té aquest atribut. Aquests atributs són molt útils quan s'usa el llenguatge de programació JavaScript:
  - `onclick`: l'acció es produeix quan fem un clic amb el ratolí damunt de l'element.
  - `ondblclick`: es fa doble clic amb el ratolí.
  - `onmousedown`: es prem el botó del ratolí.
  - `onmouseup`: es deixa de prémer el botó del ratolí.
  - `onmouseover`: estem amb el ratolí fora d'un element i el movem de manera que passem a estar damunt de l'element.
  - `onmousemove`: es mou el ratolí per damunt de l'element.
  - `onmouseout`: quan estem amb el ratolí damunt d'un element i el movem de manera que deixem d'estar al damunt.
  - `onkeypress`: es prem una tecla i es deixa anar damunt de l'element.
  - `onkeydown`: es prem una tecla damunt de l'element.
  - `onkeyup`: es deixa anar una tecla damunt de l'element.

### 2.3. Marques de text

Quan la informació que volem marcar és text, volem diferenciar les parts d'aquest text per tal de denotar-ne la major o menor importància, aclarir si una part de text és una citació, si hi volem donar més o menys èmfasi...

En la taula 5 es pot veure un resum de les marques de text més importants.

Taula 5. Marques per a text

Marca	Tipus	Descripció
<code>&lt;h1&gt;</code> <code>&lt;/h1&gt;</code> <code>&lt;h2&gt;</code> <code>&lt;/h2&gt;</code> <code>&lt;h3&gt;</code> <code>&lt;/h3&gt;</code> <code>&lt;h4&gt;</code> <code>&lt;/h4&gt;</code> <code>&lt;h5&gt;</code> <code>&lt;/h5&gt;</code> <code>&lt;h5&gt;</code> <code>&lt;/h6&gt;</code>	Bloc	Encapçalaments
<code>&lt;p&gt;</code> <code>&lt;/p&gt;</code>	Bloc	Paràgrafs

Marca	Tipus	Descripció
<code>&lt;br /&gt;</code>	Bloc	Salt de línia
<code>&lt;hr /&gt;</code>	Bloc	Línia separadora
<code>&lt;em&gt; &lt;/em&gt;</code>	Línia	Text amb èmfasi
<code>&lt;strong&gt; &lt;/strong&gt;</code>	Línia	Text amb molt èmfasi
<code>&lt;blockquote&gt;&lt;/blockquote&gt;</code>	Bloc	Citació de tipus bloc
<code>&lt;q&gt; &lt;/q&gt;</code>	Línia	Citació de tipus línia
<code>&lt;cite&gt; &lt;/cite&gt;</code>	Línia	Font de la citació
<code>&lt;acronym&gt; &lt;/acronym&gt;</code>	Línia	Acrònim
<code>&lt;abbr&gt; &lt;/abbr&gt;</code>	Línia	Abreviació
<code>&lt;pre&gt; &lt;/pre&gt;</code>	Bloc	Text preformatat
<code>&lt;code&gt; &lt;/code&gt;</code>	Línia	Codi de programació
<code>&lt;samp&gt; &lt;/samp&gt;</code>	Línia	Sortida d'un programa
<code>&lt;kbd&gt; &lt;/kbd&gt;</code>	Línia	Text per introduir per un usuari
<code>&lt;var&gt; &lt;/var&gt;</code>	Línia	Variable o paràmetre d'un programa
<code>&lt;div&gt; &lt;/div&gt;</code>	Bloc	Marca de tipus bloc sense significat semàntic
<code>&lt;span&gt; &lt;/span&gt;</code>	Línia	Marca de tipus línia sense significat semàntic

### 2.3.1. Paràgrafs

Entenem per paràgraf un conjunt de frases que estan relacionades entre si. Els paràgrafs són elements de tipus bloc i la marca que s'usa és la marca `<p>`.

Si, per exemple, tenim un paràgraf que ens explica què és el llenguatge XHTML, el marquem de la manera següent:

```
<p>
  L'XHTML (acrònim de eXtensible Hyper Text Markup Language) és el llenguatge de marques
  pensat per substituir l'HTML. L'XHTML té, bàsicament, les mateixes funcionalitats que
  l'HTML només que compleix les especificacions més estrictes de l'XML. Per exemple totes
  les marques s'han de tancar apropiadament i han d'estar escrites en minúscules.
</p>
```

### 2.3.2. Els encapçalaments

Els llenguatges de marques (X)HTML ens permeten crear sis nivells d'encapçalament amb les marques `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>` i `<h6>`. Tots aquests element són de tipus bloc.

A continuació veiem un fragment de document XHTML en què ens expliquen diverses característiques d'aquest metallenguatge. Fixem-nos en les marques d'encapçalament:

```
<h1>XML</h1>

<p>XML, de l'anglès eXtensible Markup Language (llenguatge de marques extensible), és un metallenguatge extensible, d'etiquetes,...</p>

<h2>Història</h2>

<p>XML prové d'un llenguatge inventat per IBM als anys setanta, anomenat GML (General Markup Language), i que va ser creat per la necessitat que tenia l'empresa d'emmagatzemar grans quantitats d'informació. Aquest llenguatge...</p>

<h2>Parts d'un document XML</h2>

<h3>Pròleg</h3>

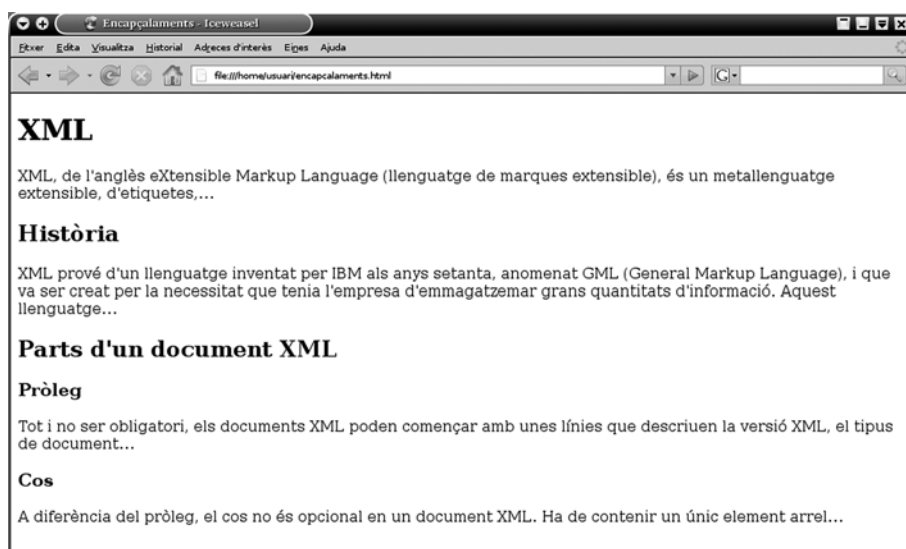
<p>Tot i no ser obligatori, els documents XML poden començar amb unes línies que descriuen la versió XML, el tipus de document...</p>

<h3>Cos</h3>

<p>A diferència del pròleg, el cos no és opcional en un document XML. Ha de contenir un únic element arrel... </p>
```

En la figura 5 podeu veure com visualitzaria un navegador aquesta pàgina XHTML. Fixem-nos que, normalment, els navegadors posen un estil per defecte a cadascuna de les marques. Així doncs, el text marcat amb `<h1>` és més gran que el marcat amb `<h2>`, aquest és més gran que el marcat amb `<h3>` i així successivament.

Figura 5. Exemples de marques d'encapçalament



### 2.3.3. Èmfasi

Quan volem donar èmfasi a una part de text perquè volem destacar-ne la importància, tenim a la nostra disposició dues marques. La marca



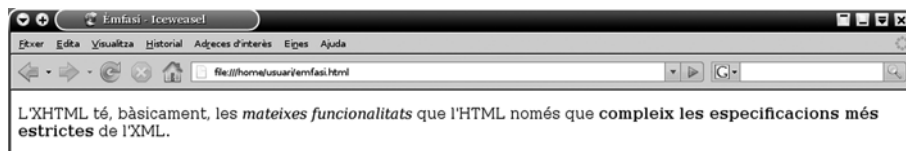
`<em>` serveix per donar èmfasi i la marca `<strong>` per donar molt èmfasi.

Imaginem que tenim un paràgraf en el qual volem destacar algunes de les paraules que conté:

```
<p>L'XHTML té, bàsicament, les <em>mateixes funcionalitats</em>
que l'HTML només que <strong>compleix les especificacions més
estrictes</strong> de l'XML.</p>
```

Per defecte, la majoria de navegadors mostren el text marcat amb `<em>` en lletra cursiva i el text marcat amb `<strong>` amb lletra negreta, com es veu en la figura 6.

Figura 6. Exemple de marques d'èmfasi



### 2.3.4. Citacions

Quan volem fer referència a un text que no hem escrit nosaltres hem de fer servir les citacions. Tenim tres marques diferents per descriure-les: `<blockquote>`, `<q>` i `<cite>`.

Si la citació és tot un paràgraf la marquem amb la marca `<blockquote>`. En canvi, si la citació està enmig d'un element de bloc (per exemple, dins d'un paràgraf) usem la marca `<q>`, que és un element de línia. Finalment, la marca `<cite>` ens serveix per marcar l'origen de la citació (autor, títol...).

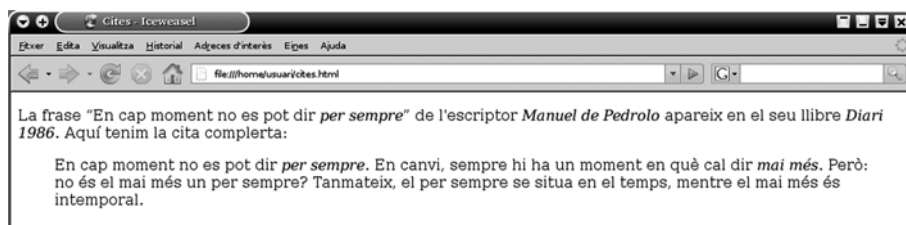
Posem un exemple d'ús d'aquestes marques: imaginem que estem fent una web on volem posar una citació de l'escriptor Manuel de Pedrolo:

```
<p>
  La frase <q>En cap moment no es pot dir <em>per
  sempre</em></q> de l'escriptor <cite>Manuel de Pedrolo</cite>
  apareix en el seu llibre <cite>Diari 1986</cite>. Aquí tenim
  la cita sencera:
</p>
<blockquote>
  <p>
    En cap moment no es pot dir <em>per sempre</em>. En canvi,
    sempre hi ha un moment en què cal dir <em>mai més</em>.
    Però: no és el mai més un per sempre? Tanmateix, el per
    sempre se situa en el temps, mentre el mai més és
    intemporal.
  </p>
</blockquote>
```

Cal destacar que dins la marca `<blockquote>` no podem escriure text directament sinó que hem d'afegir una altra marca de bloc (`<p>`, `<div>...`).

La figura 7 ens mostra el text anterior interpretat per un navegador.

Figura 7. Exemple de marques per a citacions



### 2.3.5. Salts de línia

Si volem forçar un salt de línia podem usar la marca `<br />`. No hem d'abusar d'aquesta marca i només l'hem d'usar en els casos necessaris. Per exemple, s'ha de forçar un salt de línia en cada vers d'un poema o bé en les línies d'una cançó. En canvi, no s'ha d'usar per deixar més espai entre paràgrafs ni per simular paràgrafs (això ho farem amb CSS).

Posem, per exemple, un fragment d'una poesia de Salvador Espriu:

```
<p>
  Darrera aquesta porta visc, <br />
  però no sé <br />
  si en puc dir vida.
</p>
```

El navegador ens fa un salt de línia cada cop que troba la marca `<br />` tal com podem veure en la figura 8.

Figura 8. Exemple amb marca per al salts de línia



### 2.3.6. Separadors

Si volem marcar una separació podem usar la marca `<hr />`, que ens crea una barra horitzontal. Tot i això, com que ens és molt fàcil posar línies separadores amb CSS, aquesta marca està caient en desús.

### 2.3.7. Acrònims i abreviatures

Si en un text trobem un acrònim o una abreviatura també tenim les marques adequades per senyalar-ho. Per marcar que un text és un acrònim fem servir la marca `<acronym>`. Per a les abreviatures usem la marca `<abbr>`. Posem un exemple d'ús:

---

```
<p> El meu ordinador té 512 <abbr title="MegaBytes">MB</abbr> de memòria <acronym title="Random Access Memory">RAM</acronym>. </p>
```

---

En la figura 9 veiem com ens mostraria aquest text un navegador.

Figura 9. Exemple de marques per a acrònims i abreviatures



### 2.3.8. Text preformatat

Si volem posar un fragment de text tal com està escrit en el fitxer de text (X)HTML, hem d'usar la marca `<pre>` que ens indica que aquell fragment de text està preformatat. Aquesta marca fa que tot allò que hi estigui contingut aparegui exactament igual en el navegador. És a dir, respecta els espais, salts de línia...

Veiem, com a exemple, que volem dibuixar amb caràcters una estructura de directoris. El codi és el següent:

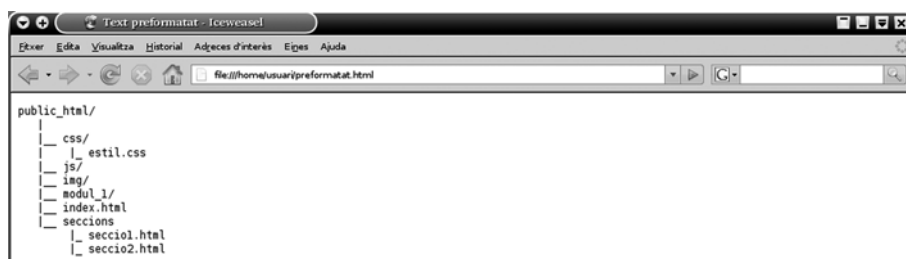
---

```
<pre>
public_html/
|
|_ css/
|_|_ estil.css
|_ js/
|_ img/
|_ modul_1/
|_ index.html
|_ seccions
|_|_ seccio1.html
|_|_ seccio2.html
</pre>
```

---

Aquest codi es visualitza com veiem en la figura 10.

Figura 10. Exemple de text preformatat



### 2.3.9. Codi de programació

Si en una pàgina web volem incloure codi de programació, tenim diverses marques que ens ajuden a formatar aquest codi. A continuació s'exposen aquestes marques i l'ús que tenen:

- La marca `<code>` ens serveix per marcar codi de programació en si.
- La marca `<samp>` serveix per posar el text obtingut en la sortida d'un programa.
- La marca `<kbd>` s'usa per dir a l'usuari quin text ha de teclejar.
- La marca `<var>` serveix per marcar variables o paràmetres de programes.

Tots aquests elements són elements de línia.

En les línies de codi següents podem veure com es poden usar les marques descrites:

---

`<p>En prémer el botó ens apareixerà el text <samp>Introdueix el teu nom:</samp></p>`

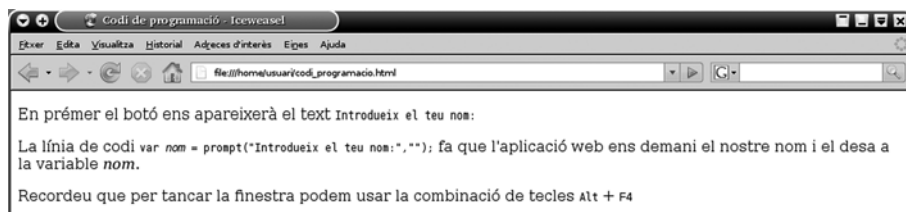
`<p>La línia de codi <code>var <var>nom</var> = prompt("Introdueix el teu nom:", "");</code> fa que l'aplicació web ens demani el nostre nom i el desa a la variable <var>nom</var>.</p>`

`<p>Recordeu que per tancar la finestra podem usar la combinació de tecles <kbd>Alt</kbd> + <kbd>F4</kbd></p>`

---

Aquest codi el podem veure visualitzat en un navegador en la figura 11.

Figura 11. Exemple amb marques per representar codi de programació



### 2.3.10. Divisions sense atributs semàntics

Hi ha dues marques que no tenen un significat semàntic concret. Això ens serveix per delimitar un tros de text o dades que no s'adeqüen a cap de les marques existents.

Per crear un element de bloc genèric usem la marca `<div>` i per crear un element de línia genèric usem la marca `<span>`.

Si, per exemple, volem maquetar la nostra pàgina web, hem de marcar les diverses capes de què disposem: la capçalera, la barra lateral, el cos de la

web i el peu de pàgina. Per definir cadascuna d'aquestes capes, podem usar la marca `<div>`:

```
<div id="capçalera">
  <!-- contingut de la capçalera (logo, títol, navegació interna,...)-->
</div>

<div id="lateral">
  <!-- contingut del menú lateral (enllaços d'interès, informació,...) -->
</div>

<div id="contingut">
  <!-- contingut (el contingut de la pàgina) -->
</div>

<div id="peu">
  <!-- contingut del peu de pàgina (informació sobre drets d'autoria, crèdits, aspectes
legals)-->
</div>
```

### 2.3.11. Marques “antiquades”

En documents (X)HTML podem trobar marques que, tot i estar permeses, ens hem d'acostumar a no usar, ja que fan referència a la presentació del document i no a l'estructura. Aquestes marques són, entre altres: `<b>` (negreta), `<i>` (cursiva), `<u>` (subratllat), `<font>` (tipus de font), `<center>` (centrar text)...

També hi ha diversos atributs de marques correctes que hauríem d'evitar, ja que també es poden reemplaçar amb l'ús de CSS. Per exemple, els atributs `align`, `border`...

## 2.4. Enllaços

L'èxit de la WWW (*World Wide Web*) és el fet de poder “navegar”, és a dir, de poder saltar d'un document a un altre a partir d'enllaços. El sistema que ens permet aquesta navegació s'anomena hipertext.

### L'hipertext

“L'hipertext és un sistema d'organització i presentació de dades que permet a l'usuari de moure's amb gran facilitat entre ítems relacionats.”

*Gran Diccionari de la llengua catalana, Gran Enciclopèdia Catalana*

La marca que té l'(X)HTML per crear hiperenllaços és la marca `<a>`. Els elements `<a>` són elements de línia, ja que podem posar enllaços envoltats de text.

Per indicar quin és el destí del nostre enllaç usem l'atribut `href`. També és convenient usar l'atribut genèric de tipus *core* `title` per posar una breu descripció de l'enllaç.

### Destins dels enllaços

El destí d'un enllaç no ha de ser sempre un altre document (X)HTML. Pot ser una imatge, un fitxer de so, un fitxer de vídeo...

### 2.4.1. Enllaços a pàgines externes

Si volem fer un enllaç a una pàgina externa a la nostra aplicació web hem de saber la URL completa del document al qual volem fer l'enllaç.

Imaginem que volem fer un enllaç a un article que es pot trobar a la URL *http://alistapart.com/articles/previewofhtml5*, però que el text que volem que aparegui en el navegador és simplement "HTML 5". El codi que hem d'introduir és el següent:

```
<p>  
  <a href="http://alistapart.com/articles/previewofhtml5" title="Article HTML 5">HTML  
5</a>  
</p>
```

### 2.4.2. Enllaços a pàgines locals

Si l'enllaç que volem fer està dirigit a un document (X)HTML local, és a dir, a un document que pertany a la mateixa aplicació web, posem, com a valor de l'atribut `href`, el camí relatiu al fitxer al qual volem fer l'enllaç.

Posem com a exemple diferents enllaços, dins d'una mateixa aplicació, que van a les diferents seccions del lloc web. Posem també un últim enllaç que va a una pàgina externa a la nostra aplicació:

```
<div id="menu">  
  <a href="index.html" title="Tornar a la pàgina d'inici">Inici</a>  
  <a href="fotos.html" title="Fotos de la web">Veure les fotos</a>  
  <a href="seccions/contacta.html" title="Formulari de contacte">Contacte</a>  
  <a href="http://ca.wikipedia.org" title="Enllaç a la Viquipèdia">Viquipèdia</a>  
</div>
```

La gran majoria de navegadors visualitzarien el codi anterior com es reflecteix en la figura 12. En la majoria de navegadors podem veure la URL destí a la barra d'estat del navegador, quan ens situem damunt de l'enllaç.

Figura 12. Exemple d'enllaç a pàgines locals



### 2.4.3. Àncores

Alguns documents (X)HTML són molt extensos i ens pot interessar navegar en la mateixa pàgina. Per poder fer això hem de posar unes marques que denotin els punts on volem anar (àncores) i crear enllaços cap a aquests punts.

Els punts on volem anar s'anomenen *àncores* i per tal de crear-les s'usa l'atribut `id` de l'element. Si, per exemple, volem fer una àncora a un element `<h2>` hem d'escriure el codi següent:

```
<h2 id="seccio1">Segona secció</h2>
```

Un cop tenim feta l'àncora hi podem enllaçar des de qualsevol punt del text fent un enllaç a l'àncora. Els enllaços a àncores es caracteritzen perquè el destí es denota amb el valor de l'atribut `id` de l'element al qual volem enllaçar precedit del caràcter `#`. Si volem enllaçar a l'àncora creada en l'exemple anterior escriurem, doncs, el codi següent:

```
<a href="#seccio1" title="Secció 1">Anar a la secció 1</a>
```

Imaginem un text molt llarg amb diferents seccions. Per tenir una bona navegabilitat podem crear un petit índex a l'inici del document que tingui enllaços cap a les diferents parts del document. A més, també podem fer enllaços que ens condueixin a l'inici del document:

```
<h1 id="dalt">Seccions</h1>

<div id="menu">
  <a href="#seccio1" title="Secció 1">Anar a la secció 1</a>
  <a href="#seccio2" title="Secció 2">Anar a la secció 2</a>
  <a href="#seccio3" title="Secció 3">Anar a la secció 3</a>
</div>

<h2 id="seccio1">Secció 1</h2>
<p>...</p>
<p><a href="#dalt" title="A dalt">Anar a dalt</a></p>

<h2 id="seccio2">Secció 2</h2>
<p>...</p>
<p><a href="#dalt" title="A dalt">Anar a dalt</a></p>

<h2 id="seccio3">Secció 3</h2>
<p>...</p>
<p><a href="#dalt" title="A dalt">Anar a dalt</a></p>
```

Si el que volem és enllaçar a una àncora des d'un altre document podem fer el següent:

```
<a href="seccions.html#seccio1" title="Secció 1">Anar a la secció 1</a>
```

#### Àncores amb atribut `name`

Les àncores també es poden fer usant la marca `<a>` amb l'atribut `name`:

```
<a name="seccio1"/>
```

Aquest mètode, però, està obsolet.

## 2.5. Imatges

Les *imatges* són un recurs molt utilitzat en el desenvolupament web. A part de fer més atractiva la nostra aplicació, ens permeten donar informació d'una manera visual.

Tot i això, hem de vigilar, no fos cas que hi hagués un excés d'imatges: hem d'anar amb cura tant amb el nombre d'imatges, com amb la mida d'aquestes imatges. L'excés d'imatges farà que la pàgina sigui massa carregada i desagradable estèticament, i el pes d'aquestes imatges pot fer que la navegació per la nostra aplicació sigui massa lenta.

La marca que ens serveix per inserir una imatge és la marca `<img>`. Es tracta d'un element de línia que no té marca de tancament: es tanca en ella mateixa.

L'atribut `src` ens servirà per dir on està ubicada la imatge (podem posar una URL externa o el camí del fitxer, si disposem de la imatge localment). També usarem l'atribut `alt`, que ens servirà per posar un text alternatiu, en el cas que el navegador no pugués accedir a la nostra imatge.

Si, per exemple, volem inserir una imatge que tenim al directori `imatges` que es diu `logo.png` i, si no es pot visualitzar aquesta imatge, volem que surti el text "Logo de la web", hem d'escriure el codi següent:

---

```

```

---

En el cas que vulguem que una imatge sigui un enllaç haurem d'enniuar correctament les marques `<a>` i `<img>`. Per exemple, si volem incorporar el logotip del W3C per destacar que la nostra pàgina compleix els estàndards i, a més, volem enllaçar aquesta imatge cap al validador en línia que ens ofereix el W3C (i així mostrar que realment el nostre document és vàlid), posarem el codi següent:

---

```
<div id="validacio">
  <a href="http://validator.w3.org/check?uri=referer">
    
  </a>
</div>
```

---

En la figura 13 podem veure com es representa el codi anterior en el navegador.

Figura 13. Exemple d'imatge amb enllaç





## 2.6. Llistes

Les *llistes* són el recurs que ens serveix per fer una enumeració. Hi ha tres tipus de llista: les llistes ordenades, les llistes desordenades i les llistes de definició.

### 2.6.1. Llistes ordenades

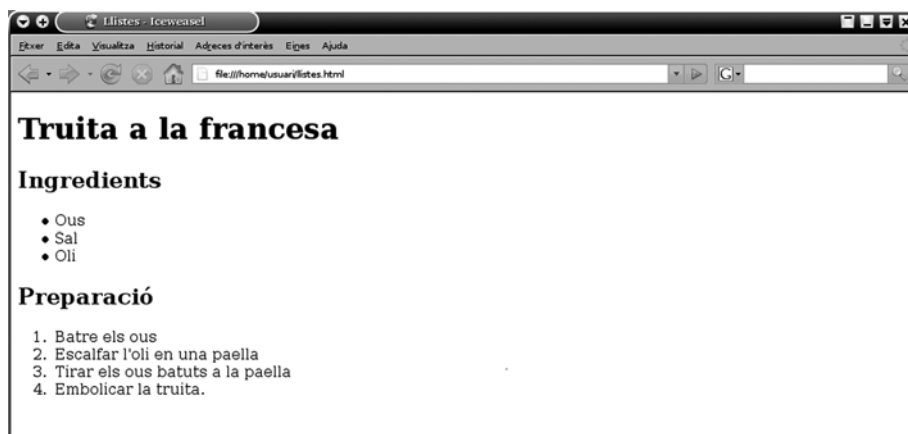
Les *llistes ordenades* són les llistes en què necessitem que els elements estiguin numerats. Per marcar l'inici i el final de la llista usem la marca `<ol>` (*ordered list*) i per marcar cada element de la llista usem la marca `<li>` (*list item*).

Imaginem que estem fent una recepta de cuina i volem enumerar els passos que hem de seguir. El codi seria el següent:

```
<h2>Preparació</h2>
<ol>
  <li>Batre els ous</li>
  <li>Escalfar l'oli en una paella</li>
  <li>Tirar els ous batuts a la paella</li>
  <li>Embolicar la truita</li>
</ol>
```

En la figura 14 podem veure la visualització de l'exemple en un navegador, juntament amb una llista desordenada.

Figura 14. Exemple de llistes



### 2.6.2. Llistes desordenades

Les *llistes desordenades* són les llistes en què no necessitem que els elements estiguin numerats, és a dir, no cal tenir en compte l'ordre de l'enumeració. Per marcar l'inici i el final de la llista usem la marca `<ul>` (*unordered list*) i per marcar cada element de la llista usem la marca `<li>` (*list item*).

Si, per exemple, volem llistar tots els ingredients necessaris per fer una truita a la francesa, ho codifiquem de la manera següent:

```
<h2>Ingredients</h2>
<ul>
  <li>Ous</li>
  <li>Sal</li>
  <li>Oli</li>
</ul>
```

En la figura 14 podem veure la visualització d'aquesta llista no numerada, juntament amb la preparació de la truita, que és una llista numerada.

### 2.6.3. Llistes de definició

Les llistes de definició es caracteritzen per ser una enumeració de definicions de termes. Per tant, els ítems de les llistes de definició estan formats per dos elements: el terme i la definició d'aquest terme. Per marcar l'inici i el final de la llista usem la marca `<dl>` (*definition list*), per a cadascun dels termes s'usa la marca `<dt>` (*definition term*) i per a cadascuna de les definicions s'usa la marca `<dd>` (*definition description*).

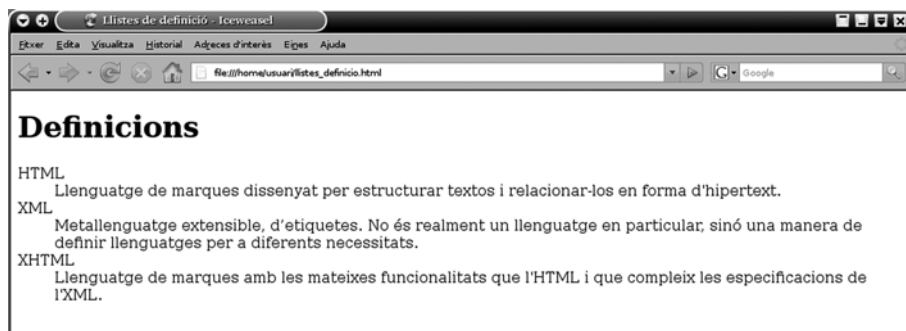
Si volem definir els termes HTML, XML i XHTML podem usar aquest tipus de llistes, que es visualitzen com mostra la figura 15:

```
<dl>
  <dt>HTML</dt>
  <dd>Llenguatge de marques dissenyat per estructurar textos i relacionar-los en forma d'hipertext.</dd>

  <dt>XML</dt>
  <dd>Metallenguatge extensible, d'etiquetes. No és realment un llenguatge en particular, sinó una manera de definir llenguatges per a diferents necessitats.</dd>

  <dt>XHTML</dt>
  <dd>Llenguatge de marques amb les mateixes funcionalitats que l'HTML i que compleix les especificacions de l'XML.</dd>
</dl>
```

Figura 15. Llista de definició



### 2.6.4. Enniament de llistes

Si volem fer llistes complexes, és a dir, inserir llistes dins d'altres llistes, també podem fer-ho. Només hem de vigilar d'obrir i tancar les marques correctament: si volem encapsular una llista, ja sigui numerada o no numerada, aquesta llista ha d'estar dins d'un ítem (<li>) de la llista mare.

Com a exemple, podem pensar en unes instruccions bàsiques de com s'ha d'instal·lar una impressora:

```
<h1>Instruccions d'instal·lació d'una impressora.</h1>
<ol>
  <li>Obrir l'embalatge.
    <ol>
      <li>Mirar el contingut del material:
        <ul>
          <li>una impressora</li>
          <li>tinta negra</li>
          <li>tinta color</li>
        </ul>
      </li>
      <li>Treure el precinte de les tintes.</li>
    </ol>
  </li>
  <li>Endollar la impressora.</li>
  <li>Posar les tintes:
    <ul>
      <li>Posar la tinta negra</li>
      <li>Posar la tinta de color</li>
    </ul>
  </li>
  <li>Imprimir.</li>
</ol>
```

Aquest exemple es veuria com es mostra en la figura 16.

Figura 16. Exemple d'enniament de llistes



## 2.7. Taules

Les *taules* són un recurs que ens permet mostrar informació tabulada en files i columnes. Molts llocs web usen les taules per maquetar la seva es-

tractura, és a dir, si, per exemple, la web té un menú lateral i un cos, es codifica una taula de dues columnes i es posa el contingut de la web en les cel·les d'aquesta taula. Hem d'evitar aquesta pràctica, ja que complica el codi innecessàriament. Per maquetar fem ús dels elements `<div>` conjuntament amb fulls d'estils CSS. Les marques bàsiques per fer taules són les que trobareu en la taula 6.

Taula 6. Marques per a taules

Marca	Descripció	Atributs destacats
<code>&lt;table&gt;</code>	Taula	summary: petit resum del propòsit de la taula width: amplada de la taula border: indica la mida de la vora de la taula cellpadding: indica l'espai entre les vores de la cel·la fins al contingut de la cel·la cellspacing: indica l'espai entre cel·les
<code>&lt;caption&gt;</code>	Títol associat a la taula	
<code>&lt;tr&gt;</code>	Fila	align: alineació horitzontal valign: alineació vertical
<code>&lt;td&gt;</code>	Cel·la	rowspan: nombre de files d'expansió
<code>&lt;th&gt;</code>	Cel·la capçalera	colspan: nombre de columnes d'expansió
<code>&lt;thead&gt;</code>	Ens permet agrupar les diferents files de la taula que formen la capçalera d'aquesta taula.	
<code>&lt;tfoot&gt;</code>	Ens permet agrupar les diferents files de la taula que formen el peu d'aquesta taula.	align: alineació horitzontal valign: alineació vertical
<code>&lt;tbody&gt;</code>	Ens permet fer diferents agrupacions de files de la taula.	

Per realitzar una taula, hem de marcar-ne l'inici i el final amb la marca `<table>`. Dins d'aquest element hem de posar tants elements `<tr>` com files tingui la taula. Finalment, dins de cada element `<tr>` hem de posar tants elements `<td>` com columnes tingui la taula. Si les cel·les són cel·les de capçalera, és a dir, tenen certa rellevància, enlloc de posar la marca `<td>` posem la marca `<th>`.

A vegades, voldrem agrupar les files en grups, per denotar quines files formen la capçalera de la taula, quines el peu i si hi diferents agrupacions en les files que conformen el cos de la taula. Per tal de fer això, usem les marques `<thead>`, `<tfoot>` i `<tbody>`, respectivament.

Imaginem que volem fer una taula de 7 files i 2 columnes. La primera fila és la capçalera de la taula, les cinc següents conformen el cos de la taula i l'última és el peu. Tindrem el codi següent, el resultat del qual el podem veure en la figura 17:

```
<table border="1">
  <thead>
    <tr>
      <th>Nom</th>
```

#### Grups de columnes

Igual que podem fer agrupacions de files d'una taula, també es poden agrupar les columnes d'una taula usant les marques `<colgroup>` i `<col>`.

```

        <th>Descripció</th>
    </tr>
</thead>
<tfoot>
    <tr>
        <th>Número de programadors</th>
        <th>5</th>
    </tr>
</tfoot>
<tbody>
    <tr>
        <td>Richard Stallman</td>
        <td>Emacs, gcc, gdb, fundador del projecte GNU,...</td>
    </tr>
    <tr>
        <td>Linus Torvalds</td>
        <td>Autor del nucli de Linux</td>
    </tr>
    <tr>
        <td>Donald Becker</td>
        <td>Drivers d'Ethernet per a Linux</td>
    </tr>
    <tr>
        <td>Tim Berners-Lee</td>
        <td>Inventor del World Wide Web</td>
    </tr>
    <tr>
        <td>Brian Behlendorf</td>
        <td>Apache</td>
    </tr>
</tbody>
</table>

```

Figura 17. Taula simple



The screenshot shows a web browser window with the title 'Iceweasel'. The address bar shows 'file:///home/usuario/taula.html'. The page content features a table with the title 'Programadors famosos'. The table has two columns: 'Nom' and 'Descripció'. The rows contain the following data:

Nom	Descripció
Richard Stallman	Emacs, gcc, gdb, fundador del projecte GNU,...
Linus Torvalds	Autor del nucli de Linux
Donald Becker	Drivers d'Ethernet per a Linux
Tim Berners-Lee	Inventor del World Wide Web
Brian Behlendorf	Apache
Número de programadors	5

Observem que, en el codi, la fila que fa de peu de pàgina la podem escriure abans que les que fan de cos, ja que el navegador ja sap que és peu de taula, atès que està marcat amb la marca `<tfoot>`.

### 2.7.1. Expansió de files i columnes

Les taules simples ens permeten fer una quadrícula de  $x$  files i  $y$  columnes, però ens podem trobar amb la necessitat de voler fusionar cel·les tant horitzontalment com verticalment. Per aconseguir aquest objectiu la marca `<td>` disposa de dos atributs: `colspan` i `rowspan`.

L'atribut `colspan` serveix per fusionar dues cel·les o més horitzontalment. L'atribut es posa a la cel·la que ha d'ocupar més d'una columna i se li dóna com a valor el nombre de columnes que ha d'ocupar.

L'atribut `rowspan` serveix per fusionar dues cel·les o més verticalment. L'atribut es posa a la cel·la que ha d'ocupar més d'una fila i se li dóna com a valor el nombre de files que ha d'ocupar.

Imaginem que volem posar l'horari escolar que es veu en la figura 18. Per aconseguir-ho hem d'escriure el codi següent:

---

```
<table summary="Horari" border="1">
  <caption>Horari</caption>
  <thead>
    <tr>
      <th>&nbsp;</th>
      <th>Dilluns</th>
      <th>Dimarts</th>
      <th>Dimecres</th>
      <th>Dijous</th>
      <th>Divendres</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>16.00-16.55</td>
      <td rowspan="2">Bases de dades</td>
      <td rowspan="2">Xarxes</td>
      <td rowspan="2">Programació Estructurada</td>
      <td>Xarxes</td>
      <td>Bases de dades</td>
    </tr>
    <tr>
      <td>16.55-17.50</td>
      <td rowspan="2">Sistemes Operatius</td>
      <td rowspan="2">Xarxes</td>
    </tr>
    <tr>
      <td>17.50-18.45</td>
      <td>Programació Estructurada</td>
      <td>Programació Estructurada</td>
      <td>Bases de dades</td>
    </tr>
  </tbody>
  <tbody>
    <tr>
      <td>18.45-19.15</td>
      <td colspan="5">Pati</td>
    </tr>
  </tbody>
  <tbody>
    <tr>
      <td>19.15-20.10</td>
      <td>Programació Estructurada</td>
      <td>Programació Estructurada</td>
      <td rowspan="2">FOL</td>
      <td rowspan="2">Programació Estructurada</td>
      <td rowspan="2">RAT</td>
    </tr>
    <tr>
      <td>20.10-21.05</td>
      <td rowspan="2">Sistemes Operatius</td>
    </tr>
```

```

        <td>Sistemes Operatius</td>
    </tr>
    <tr>
        <td>21.05-21.45</td>
        <td>&nbsp;</td>
        <td>&nbsp;</td>
        <td>Tutoria</td>
        <td>&nbsp;</td>
    </tr>
</tbody>
</table>

```

Figura 18. Taula amb expansions de files i columnes

	Dilluns	Dimarts	Dimecres	Dijous	Divendres
16.00-16.55	Bases de dades	Xarxes	Programació Estructurada	Xarxes	Bases de dades
16.55-17.50					
17.50-18.45	Programació Estructurada	Programació Estructurada	Bases de dades	Sistemes Operatius	Xarxes
18.45-19.15	Pati				
19.15-20.10	Programació Estructurada	Programació Estructurada	FOL	Programació Estructurada	RAT
20.10-21.05		Sistemes Operatius			
21.05-21.45	Sistemes Operatius			Tutoria	

## 2.8. Formularis

Els formularis web ens serveixen per interactuar amb l'usuari i perquè aquest usuari ens pugui transmetre informació. Aquesta informació es pot processar de diferents maneres, segons les necessitats de l'aplicació web.

Tècnicament, un formulari no és altra cosa que un fragment de codi (X)HTML que conté uns elements característics anomenats *controls*. Tenim diversos tipus de control: camps de text, camps de contrasenya, botons d'opció (*radio buttons*), caselles de verificació (*checkbox*), camps per introduir fitxers, llistes de selecció, àrees de text i botons. En la figura 19 podeu veure un formulari amb tots aquests controls.

Figura 19. Controls d'un formulari

**Contacta amb nosaltres**

**Dades personals**

Escriu el teu nom:  ← Camp de text

Escriu la teva contrasenya:  ← Camp de contrasenya

Edat: ☐ Menys de 18 anys ☐ De 18 a 65 anys ☐ Més de 65 anys

On vius?  ← Botons d'opció

Envia'm la teva foto:   ← Camp de fitxer

**Aficions**

Escull les teves aficions: ☐ Llegir ☐ Anar al cine ☐ Trekking

**Comentaris**

Estic molt interessada en la vostra web

← Caselles de verificació

← Àrea de text

Botó de restabliment Botó d'enviament

Cadascun dels controls d'un formulari ha de tenir l'atribut `name`, amb el qual s'identifica la dada que es vol enviar. Posem un exemple: tenim un camp de text l'objectiu del qual és que l'usuari introdueixi el seu nom i suposem que l'usuari introdueix "Pau". Si el valor de l'atribut `name` del camp de text és `nom` (`name="nom"`), la informació que s'enviarà és `nom=Pau`. Fixem-nos en el codi d'aquest formulari:

```
<h1>Contacta amb nosaltres</h1>

<form action="processa.html" method="get">

<fieldset>
<legend>Dades personals</legend>
<div>
  <label for="nom">Escriu el teu nom: </label>
  <input type="text" name="nom" size="30" maxlength="100" id="nom"/>
</div>

<div>
  <label for="contrasenya">Escriu la teva contrasenya: </label>
  <input type="password" name="contrasenya" size="10" maxlength="15" id="contrasenya" />
</div>

<div>
  Edat:
  <input type="radio" name="edat" id="edat_menor" value="menor" />
  <label for="edat_menor">Menys de 18 anys </label>
  <input type="radio" name="edat" id="edat_major" value="major" checked="checked" />
  <label for="edat_major">De 18 a 65 anys </label>
  <input type="radio" name="edat" id="edat_jubilat" value="jubilat" />
  <label for="edat_jubilat">Més de 65 anys</label>
</div>

<div>
<label for="municipi">On vius? </label>
<select name="municipi" id="municipi">
  <option value="Abella de la Conca">Abella de la Conca</option>
  <option value="Abrera">Abrera</option>
  <option value="Àger">Àger</option>
  <option value="Agramunt">Agramunt</option>
  <option value="Aguilar de Segarra">Aguilar de Segarra</option>
  <option value="Agullana">Agullana</option>
  <option value="Aiguafreda">Aiguafreda</option>
  ...
</select>
</div>
<div>
  <label for="foto">Envia'ns la teva foto: </label><input type="file" name="foto" id="foto" />
</div>
</fieldset>

<fieldset>
<legend>Aficions</legend>
<div>
  Escull les teves aficions:
  <input type="checkbox" name="llegir" id="llegir" value="si" />
  <label for="llegir">Llegir</label>
  <input type="checkbox" name="cine" id="cine" value="si" />
  <label for="cine">Anar al cine</label>
  <input type="checkbox" name="trekking" id="trekking" value="si" />
  <label for="trekking">Trekking</label>

```



```
</div>

</fieldset>

<fieldset>
<legend><label for="comentari">Comentaris</label></legend>
<div>
    <textarea name="comentari" id="comentari" cols="30" rows="5">Escriu aquí els teus
comentaris</textarea>
</div>
</fieldset>

<div>
    <input type="reset" value="Esborrar formulari" />
    <input type="submit" value="Enviar formulari" />
</div>

</form>
```

---

### 2.8.1. Propietats d'un formulari

Per marcar el codi (X)HTML que forma part d'un formulari s'usa la marca `<form>`. Aquesta marca té diferents atributs, que ens serveixen per dir com volem trametre la informació que omple l'usuari. A continuació s'enumeren cadascun d'aquests atributs:

- **Atribut `action`:** el valor d'aquest atribut és el destí on ha d'anar a parar la informació que l'usuari ha posat en el formulari. Normalment és una URL amb l'script que ha de processar les dades del formulari, però també podem fer que la informació es dirigeixi a una adreça de correu electrònic.
- **Atribut `method`:** ens permet dir de quina manera enviem la informació. Tenim dues possibilitats: si el valor de l'atribut és "get" enviem la informació juntament amb la URL de destí. Si el valor de l'atribut `method` és "post", la informació s'envia dins de la capçalera HTTP que fa la petició.

A continuació es mostra un formulari amb dos camps de text que demanen a l'usuari el seu nom i contrasenya, i un botó per enviar les dades. Suposem que al servidor hi ha un script fet amb llenguatge PHP anomenat *processa.php*, que és qui rep les dades i les gestiona. Suposem primer que el mètode d'enviament és de tipus `get`:

---

```
<form action="processa.php" method="get">
  <div>
    <label for="nom">Escriu el teu nom: </label>
    <input type="text" name="nom" id="nom"/>
  </div>
  <div>
    <label for="contrasenya">Escriu la teva contrasenya: </label>
    <input type="password" name="contrasenya" id="contrasenya" />
  </div>
  <div>
    <input type="submit" value="Envia les dades" />
  </div>
</form>
```

---

Si l'usuari entra de nom “Anna” i de contrasenya “secreta”, quan premi el botó per enviar el formulari anirem a parar a la URL següent: “http://elquesigui.org/accio.php?nom=Anna&edat=secreta”.

Suposem que canviem el mètode d'enviar les dades per un mètode de tipus post:

---

```
<form action="processa.php" method="post">
...
</form>
```

---

La URL que veuríem seria http://elquesigui.org/accio.php, però les dades s'haurien enviat igualment en la capçalera HTTP.

### Capçalera HTTP

Per tal de veure les capçaleres HTTP quan s'usa el mètode POST, podríem usar un analitzador de xarxes, com pot ser el programa Wireshark. Un exemple de capçalera HTTP amb POST fent la captura amb aquest programa ens retornaria el següent:

```
POST /processa.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; U; Linux i686; ca; rv:1.8.1.12) Gecko/20080129 Icedove/2.0.0.12
(Debian-2.0.0.12-1)
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/
png,*/*;q=0.5
Accept-Language: ca,es;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: http://localhost/formulari_complet.html
Content-Type: application/x-www-form-urlencoded
Content-Length: 28
nom=Anna&contrasenya=secreta
```

- **Atribut enctype:** aquest atribut ens indica el tipus de contingut que enviem amb el formulari (quan el mètode és de tipus post). Per defecte, el valor és application/x-www-form-urlencoded. Si el formulari ens permet enviar un fitxer hem de canviar aquest valor per multipart/form-data.

### 2.8.2. Etiquetes d'un formulari

La marca <label> ens serveix per associar els elements de text explicatius dels controls amb els controls corresponents. L'atribut for de la marca <label> ens permet posar l'identificador (atribut id) del control a què volem associar l'etiqueta.

Per dir que el text “Escriu el teu nom” està associat al control amb id “nom” escriurem el següent:

---

```
<label for="nom">Escriu el teu nom:</label>
<input type="text" name="nom" id="nom"/>
```

---

### 2.8.3. Atributs genèrics dels controls d'un formulari

Hi ha dos atributs que podem assignar a cadascun dels controls d'un formulari. Són els següents:

- `tabindex`: posició en l'ordre de tabulació.
- `disabled`: si l'establim (`disabled="disabled"`), el control queda deshabilitat.

### 2.8.4. Camps de text

Els camps de text són els camps en què l'usuari pot introduir un text relativament curt. Per crear un camp de text usem la marca `<input type="text" />`. A més, tenim els atributs següents:

- `name`: nom del control.
- `value`: valor per defecte del camp de text.
- `readonly`: si l'establim (`readonly="readonly"`), el control és de només lectura, no s'hi pot escriure.
- `maxlength`: nombre màxim de caràcters que permetrem escriure en el control.
- `size`: mida del control.

Un camp de text que mesuri l'espai que ocuparien trenta caràcters però en què permetem que se n'escriguin fins a cent seria:

```
<input type="text" name="nom" size="30" maxlength="100" id="nom" />
```

### 2.8.5. Camps de contrasenya

Els camps de contrasenya són els camps en què l'usuari pot introduir una contrasenya, ja que el text que s'introdueix queda “camuflat” amb asteriscos. Per crear un camp de text usem la marca `<input type="password" />`, i els mateixos atributs que utilitzem per als camps de text. Un exemple de camp de contrasenya seria el següent:

```
<input type="password" name="contrasenya" size="10" maxlength="15" id="contrasenya" />
```

#### Seguretat de contrasenyes

No hem de pensar que la informació que posem en els controls de tipus `password` està securitzada ja que si no s'usa el protocol `https` la informació circula per la xarxa sense xifrar.

### 2.8.6. Botons d'opció

Els botons d'opció ens permeten triar una i només una de les opcions que se'ns ofereixen. Per crear un botó d'opció usem la marca `<input type="radio" />`. A més, tenim els atributs següents:

- `name`: nom del control.
- `checked`: determina si el control està seleccionat.

- `value`: si està seleccionat, estableix quin valor es tramet.
- `size`: determina la mida del control en píxels.

Habitualment, no només posem un botó d'opció sinó que en posem diversos d'associats entre si. Per exemple, si demanem el sexe de la persona, tenim dos botons d'opció, un per senyalar que s'és dona i l'altre per senyalar que s'és home. Per tal que el formulari sàpiga que aquests botons d'opció estan relacionats, donem el mateix valor a l'atribut `name`.

Fixem-nos en l'exemple següent, en què apareixen tres botons d'opció:

---

```
<input type="radio" name="edat" id="edat_menor" value="menor"/>
<label for="edat_menor">Menys de 18 anys </label>
<input type="radio" name="edat" id="edat_major" value="major" checked="checked"/>
<label for="edat_major">De 18 a 65 anys </label>
<input type="radio" name="edat" id="edat_jubilat" value="jubilat"/>
<label for="edat_jubilat">Més de 65 anys</label>
```

---

Observem que els tres controls tenen el mateix valor en el camp `name` (`edat`). Així, quan trametem el formulari ens arriba una de les tres possibilitats següents: *edat=menor* o bé *edat=major* o bé *edat=jubilat*.

### 2.8.7. Caselles de verificació

Les caselles de verificació ens permeten triar una opció o més de diverses possibilitats. Per crear una casella de verificació usem la marca `<input type="checkbox" />`. A més, tenim els mateixos atributs que per als botons d'opció.

En aquest cas l'atribut `name` no ha de ser el mateix ja que es poden prémer una casella o més de les caselles de verificació. Fixem-nos en l'exemple:

---

```
<input type="checkbox" name="llegir" id="llegir" value="si"/>
<label for="llegir">Llegir</label>
<input type="checkbox" name="cine" id="cine" value="si"/>
<label for="cine">Anar al cine</label>
<input type="checkbox" name="trekking" id="trekking" value="si"/>
<label for="trekking">Trekking</label>
```

---

En aquest cas, si preméssim les caselles “Llegir” i “Anar al cine” s'enviaria: *llegir=si&cine=si*.

### 2.8.8. Llistes de selecció

Les llistes de selecció tenen dos formats: les llistes de selecció simple, que ens deixen escollir una única opció d'un llistat de possibilitats i les llistes

de selecció múltiple, que ens deixen escollir més d'una opció (prement la tecla `Ctrl` mentre seleccionem les diverses opcions).

La funció de la llista de selecció simple és similar als botons d'opció (només es pot triar una opció), mentre que la funció de la llista de selecció múltiple és similar a les caselles de verificació (podem triar més d'una opció). Malgrat tot, el format de presentació i el funcionament del control és diferent.

Per crear una llista de selecció usem dues marques:

- La marca `<select>` ens serveix per dir on comença i s'acaba la llista de selecció. Tenim els atributs següents:
  - `name`: nom del control.
  - `size`: mida del control en nombre d'opcions visibles.
  - `multiple`: si l'establim (`<select multiple="multiple">`) farà que aquesta llista sigui una llista de selecció múltiple.
- Per a cadascuna de les opcions usem la marca `<option>`. Aquesta marca està continguda dins de la marca `<select>` i a la seva vegada conté el text de l'opció que s'ha de seleccionar. Té els atributs següents:
  - `value`: determina el valor que s'envia si se selecciona aquesta opció.
  - `selected`: si l'establim (`selected="selected"`) aquesta opció està seleccionada per defecte. Hem de tenir en compte que si la llista és de selecció simple, només haurem de tenir una opció amb aquest atribut, mentre que si és de selecció múltiple podem posar-lo en més d'una opció.

Un exemple de llista d'opció és el següent:

---

```
<label for="municipi">Marca els municipis que has visitat:</label>
<select name="municipi" id="municipi" multiple="multiple">
  <option value="Abella de la Conca">Abella de la Conca</option>
  <option value="Abrera">Abrera</option>
  <option value="Àger">Àger</option>
  <option value="Agramunt">Agramunt</option>
  <option value="Aguilar de Segarra">Aguilar de Segarra</option>
  <option value="Agullana">Agullana</option>
  <option value="Aiguafreda">Aiguafreda</option>
  ...
</select>
```

---

En aquest cas, es permet que l'usuari marqui diverses opcions. Si per exemple, marca que ha visitat les poblacions d'Abrera i d'Aiguafreda, s'envia `municipi="Abrera"&municipi="Aiguafreda"`.

Si volem classificar les diferents opcions de la llista en grups posant-los un títol descriptiu, podem usar la marca `<optgroup>`, com es veu en l'exemple següent:

```
<label for="municipi">On vius? </label>
<select name="municipi" id="municipi" size="15">
  <optgroup label="Capitals">
    <option value="bcn">Barcelona</option>
    <option value="gir" selected="selected">Girona</option>
    <option value="lle">Lleida</option>
    <option value="tar">Tarragona</option>
  </optgroup>
  <optgroup label="Altres municipis">
    <option value="Abella de la Conca">Abella de la Conca</option>
    <option value="Abrera">Abrera</option>
    <option value="Àger">Àger</option>
    <option value="Agramunt">Agramunt</option>
    <option value="Aguilar de Segarra">Aguilar de Segarra</option>
    <option value="Agullana">Agullana</option>
    <option value="Aiguafreda">Aiguafreda</option>
    ...
  </optgroup>
</select>
```

El contingut de l'atribut `label` de l'element `<optgroup>` es mostra com un títol descriptiu, però no es pot seleccionar. Podem veure aquesta llista en la figura 20.

Figura 20. Exemple de llista de selecció



### 2.8.9. Àrees de text

Les àrees de text permeten que l'usuari de l'aplicació web envii una porció de text més gran que amb el control de camp de text. La marca que ens permet inserir una àrea de text és `<textarea>`. Els atributs que podem usar amb aquesta marca són els següents:

- `name`: nom del control.
- `cols`: amplada, en caràcters.
- `rows`: alçada, en nombre de línies.
- `readonly`: si l'establim (`readonly="readonly"`), el control és de només lectura, no s'hi pot escriure.

Veiem un exemple de control d'àrea de text en què deixem a l'usuari cinc files i trenta columnes visibles per escriure. El text que està entre les marques d'inici i fi és el text que surt per defecte a l'àrea de text:

---

```
<textarea name="comentari" id="comentari" cols="30" rows="5">Escriu aquí els teus comentaris</textarea>
```

---

### 2.8.10. Botons

Els botons d'un formulari ens serveixen per fer accions quan els premem. Tenim tres tipus de botons:

- **Botó d'enviament de formulari:** quan es prem aquest botó el formulari es trameta. És a dir, les dades són enviades al destí que hem posat en l'atribut `action` de l'element `form`. Per inserir un botó d'aquest tipus usem la marca `<input type="submit" />`. Disposem a més de l'atribut `value`, que ens indica el text que apareix en el botó.
- **Botó de reestabliment del formulari:** quan es prem aquest botó el formulari torna a posar-se en l'estat inicial. És a dir, s'esborra tot allò que ha escrit l'usuari i es posen les dades per defecte de l'usuari. Per inserir un botó d'aquest tipus usem la marca `<input type="reset" />`. Amb l'atribut `value` podem indicar el text que apareix al botó.
- **Botó genèric:** podem inserir botons en els quals podem afegir noves funcions mitjançant codi de programació. Per poder fer això usem la marca `<button></button>`. El valor que ens apareix al botó és allò que posem entre l'inici i el fi de la marca. Podem posar tant text com imatges. Aquest control té els atributs següents:
  - `type`: defineix quin tipus de botó és. Pot prendre els valors següents: `submit` (amb la mateixa funció que els botons d'enviament de formulari), `reset` (amb la mateixa funció que els botons de reestabliment de formulari), `button` (botó genèric).
  - `name`: nom del control
  - `value`: valor que es trameta quan premem el botó.

Posem com a exemple els botons més usats, que són el de reestabliment de formulari i el d'enviament de formulari:

---

```
<input type="reset" value="Esborrar formulari" />
<input type="submit" value="Enviar formulari" />
```

---

### 2.8.11. Controls ocults

En alguns casos ens pot interessar tenir en un formulari un control amb un valor concret, però que no és visible ni directament modificable per l'usuari final. Per inserir un control d'aquest tipus s'usa la marca `<input type="hidden" />`. Els atributs que s'usen amb aquest control són els següents:

- `name`: nom del control.
- `value`: valor que es tramet.

Per exemple, si volem enviar *seccio=segona*, hem d'escriure el codi següent:

```
<input type="hidden" name="seccio" value="segona" />
```

## 2.9. Marcs

Molt sovint trobem llocs web en els quals, per a totes les pàgines del lloc, hi ha parts amb el mateix contingut (capçalera, bàners, menú...). Una solució barroera per fer aquest tipus de lloc és tenir un document (X)HTML per a cada pàgina on molta part del contingut és el mateix.

El problema que ens pot comportar aquesta solució és que si volem modificar alguna cosa d'aquestes parts fixes, ho haurem de fer a cadascun dels documents (X)HTML. Això, a part de ser una feina repetitiva, pot conduir a múltiples errors.

Els marcs són una solució alternativa a aquest problema. La idea és que dividim el navegador en diverses seccions i a cadascuna de les seccions se'ns carrega una pàgina (X)HTML diferent. Per tal d'aconseguir aquesta solució hi ha un document (X)HTML extra que ens serveix per definir les seccions en què volem dividir la finestra del navegador.

Imaginem la situació següent: volem un lloc web amb una capçalera fixa, un menú lateral fix, un peu de pàgina fix i un cos variant. Per aconseguir aquest lloc web necessitarem diversos fitxers: un fitxer *marcs.html* (o *index.html*, si fos la pàgina principal) on es defineix l'estructura de la pàgina; un fitxer *cap.html*, on posem la capçalera; un fitxer *menu.html*, on posem el menú; un fitxer *peu.html*, i diversos fitxers *.html* que aniran mostrant les diferents seccions de la web (*inici.html*, *seccio1.html*, *seccio2.html*...). La situació descrita es pot veure en la figura 21.

#### Marcs flotants

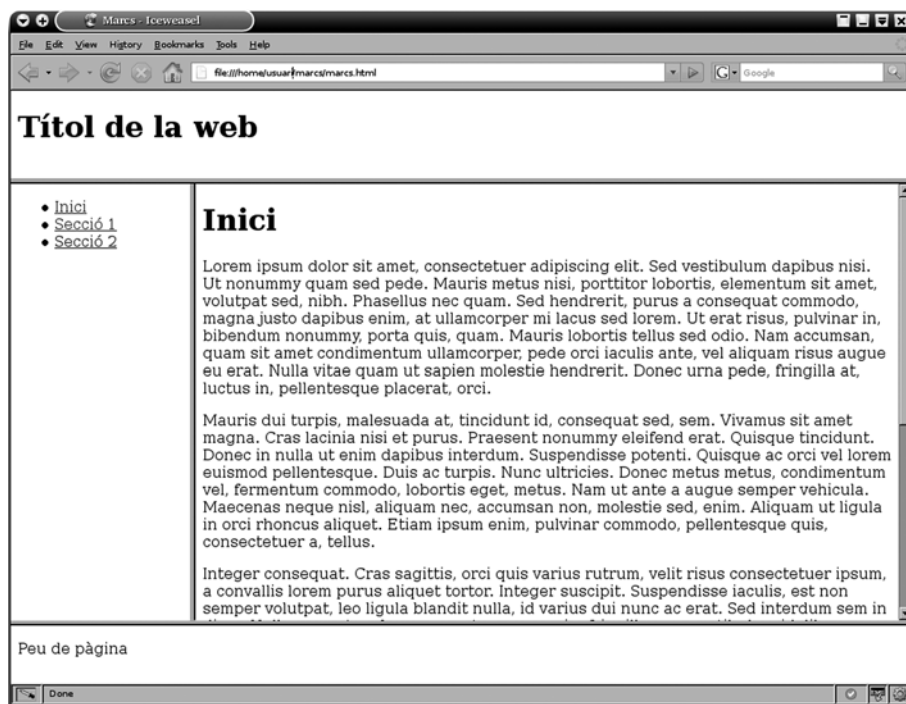
Hi ha un altre tipus de marcs que podem incrustar directament en el document (X)HTML. Funcionen igual que si inseríssim una imatge però el que inserim és un marc que té com a contingut un altre document (X)HTML.

Per fer aquest tipus de marcs s'usa la marca `<iframe>`:

```
<iframe src="pagina.html" />
```



Figura 21. Exemple de marc



### 2.9.1. Estructura d'un web fet amb marcs

L'estructura d'un web fet amb marcs es defineix en un fitxer especial destinat a aquest objectiu. En aquest fitxer especifiquem quantes files i columnes volem i quant espai ocupa cadascuna d'elles. També especifiquem quina és la pàgina inicial que apareix en cadascun dels marcs.

Aquest fitxer de definició dels marcs no té la marca `<body>` i usa dues marques especials: `<frameset>` i `<frame>`. A més, hem d'usar la DTD *(X)HTML 1.0 Frameset*.

La marca `<frameset>` ens serveix per definir en quins subespais dividirem la pantalla. Amb l'atribut `rows` s'especifica el nombre de subespais horitzontals i quant ocupen, i amb l'atribut `cols` s'especifica el nombre de subespais verticals i quant ocupen.

Si no s'estableix l'atribut `rows`, cada columna s'estén per tota la longitud de la pàgina. Si no s'estableix l'atribut `cols`, cada fila s'estén al llarg de tota l'amplada de la pàgina. Si no s'estableix cap dels dos atributs, el marc té exactament la mateixa mida que la pàgina.

Els marcs es creen d'esquerra a dreta per a les columnes, i de dalt a baix per a les files. Quan s'especifiquen tots dos atributs, les vistes es creen d'esquerra a dreta en la fila superior, en la segona fila, etc.

La marca `<frame>` ens serveix per definir com ha de ser cadascun d'aquests subespais. Dins l'element `<frameset>`, tenim un element

`<frame>` per a cada subespai. Disposem dels atributs següents per tal de definir cadascun dels subespais:

- `src`: indica l'adreça del document (X)HTML que hi haurà al marc.
- `name`: assigna un nom al marc per poder-s'hi referir posteriorment.
- `scrolling`: decideix si es col·loquen o no barres de desplaçament en el marc per poder-nos moure pel seu contingut. El seu valor per defecte és `auto`, que deixa la decisió al navegador. Les altres opcions són `yes` i `no`.
- `noresize`: atribut booleà mitjançant el qual, si s'especifica, l'usuari no pot canviar la mida del marc. Per especificar-lo hem d'escriure `noresize="noresize"`.
- `frameborder`: si l'igualem a zero s'elimina la vora amb tots els marcs que conté que també tinguin aquest valor a zero.
- `marginwidth`: permet canviar els marges horitzontals dins un marc. Es representa en píxels.
- `marginheight`: permet canviar els marges verticals dins un marc. Es representa en píxels.

Vegem ara un seguit d'exemples que ens generarien diverses estructures:

### 1) Divisió de la pantalla en dos seccions horitzontals de mida igual.

```
<frameset rows="50%, 50%">
...la resta de la definició...
</frameset>
```

2) Divisió de la pantalla en tres columnes: la segona columna amb una amplada fixa de 250 píxels (la qual cosa és útil, per exemple, per incloure una imatge de mida coneguda), de l'espai restant a la primera columna s'hi assigna el 25% i a la tercera el 75%.

```
<frameset cols="1*,250,3*">
  <frame name="superior" src="superior.html" />
  <frame name="inferior" src="inferior.html" />
</frameset>
```

### 3) Divisió de la pantalla en una quadrícula de 2 × 3 subespais.

```
<frameset rows="30%,70%" cols="33%,34%,33%" />
  <frame name="supesq" src="supesq.html" />
  <frame name="supcent" src="supcent.html" />
  <frame name="supesq" src="supesq.html" />
  <frame name="infesq" src="infesq.html" />
  <frame name="infcent" src="infcent.html" />
  <frame name="infesq" src="infesq.html" />
</frameset>
```

4) Divisió de la pantalla en tres files: la segona fila té una alçada total fixa de 300 píxels. De l'espai restant la primera fila n'ocupa un 30%. Finalment,

l'alçada del quart marc s'ha especificat com a 2\*, de manera que és el doble d'alt que el tercer marc, que té alçada \* (equivalent a 1\*).

---

```
<frameset rows="30%,400,*,2*" />
  <frame name="superior" src="superior.html" />
  <frame name="central" src="central.html" />
  <frame name="inferior" src="inferior.html" />
</frameset>
```

---

Si suposem que la nostra pantalla té una alçada de 1.000 píxels, tindrem que la primera fila té una alçada de 300 píxels (que és el 30% de 1.000), la segona fila té una alçada de 400 píxels. Ens queden 300 píxels disponibles per als altres dos marcs. Per tant, el tercer marc té 100 píxels d'alçada i el quart, 200 píxels.

Els navegadors hauran d'ajustar les longituds absolutes que no sumin el 100% de l'espai real disponible. Si sobra espai, s'haurà de repartir proporcionalment entre cada vista. Si falta espai, s'haurà de reduir cada vista segons la relació entre l'espai indicat i l'espai total.

### 2.9.2. Enniuament de marcs

Els grups de marcs es poden ennuiar fins a qualsevol nivell. En l'exemple següent, el `<frameset>` exterior divideix l'espai disponible en tres columnes iguals.

L'estructura típica d'una capçalera, menú, cos de pàgina i peu de pàgina que hem vist en la figura 21 l'hauríem de pensar com a tres files, en què la segona fila no és un `<frame>` sinó que és un altre `<frameset>` compost per dues columnes:

---

```
<frameset rows="15%, 75%, *">
  <frame name="cap" src="cap.html" />
  <frameset cols="30%, 70%">
    <frame name="menu" src="menu.html" />
    <frame name="cos" src="seccio1.html" />
  </frameset>
  <frame name="peu" src="peu.html" />
</frameset>
```

---

### 2.9.3. Modificació de pàgines d'altres marcs

Un cas habitual en l'ús de marcs és que volem prémer un enllaç i que l'objectiu de l'enllaç se'ns obri en un marc diferent del marc en què estem. Imaginem la situació típica d'un estructura amb una capçalera superior un menú esquerre i el cos de la pàgina a la dreta. Volem que quan premem els enllaços del menú no se'ns obrin al mateix marc on està el menú (marc

inferior esquerre) sinó que se'ns obrin al marc inferior dret, on hi ha el cos de la web.

Per tal de fer això hem d'usar l'atribut `target` de la marca `<a>`. Com a valor de `target` hem d'escriure el nom del marc on volem que se'ns obri la pàgina destí de l'enllaç.

El fitxer *marcs.html* amb l'estructura del lloc web és:

---

```
<frameset rows="15%, 75%, *">
  <frame name="cap" src="cap.html" />
  <frameset cols="30%, 70%">
    <frame name="menu" src="menu.html" />
    <frame name="cos" src="seccio1.html" />
  </frameset>
  <frame name="peu" src="peu.html" />
</frameset>
```

---

Un fragment de codi del document *menu.html*:

---

```
<ul>
  <li><a href="inici.html" target="cos">Inici</a></li>
  <li><a href="seccio1.html" target="cos">Secció 1</a></li>
  <li><a href="seccio2.html" target="cos">Secció 2</a></li>
</ul>
```

---

Això fa que en prémer el segon enllaç se'ns obri el document *seccio1.html* en el marc que té com a valor de `name` el text `cos`. Com hem vist, l'atribut `name` s'especifica en la marca `<frame>` del document de definició de marcs.

#### 2.9.4. El problema dels marcs

Tot i que hi ha múltiples llocs web que usen marcs, el seu ús no és molt recomanat per diversos motius: alguns navegadors no els suporten, els cercadors no saben indexar bé els llocs web fets amb marcs, no compleixen les normes d'accessibilitat...

Hi ha altres maneres d'aconseguir l'efecte dels marcs sense usar-los: usant programació web en el servidor o usant fulls d'estils. Malgrat tot, si no tenim més remei que usar-los i volem donar una alternativa als navegadors que no suporten els marcs podem usar la marca `<noframes>`. Un exemple d'ús seria el següent:

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/
DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title></title>
<meta name="author" content="Monica Ramirez" />
<meta name="date" content="2006-12-19T17:26:20+0100" />
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
```

```
</head>

<frameset rows="15%, 75%, *">
  <frame name="cap" src="cap.html" />
  <frameset cols="20%, 80%">
    <frame name="menu" src="menu.html" />
    <frame name="cos" src="secciol.html" />
  </frameset>
  <frame name="peu" src="peu.html" />
  <noframes>
    <body>
      <h1>El navegador que estàs usant no suporta marcs. Ho sento però no podràs veure el
contingut d'aquesta pàgina...</h1>
    </body>
  </noframes>
</frameset>
</html>
```

## 2.10. Mapes

La combinació de la marca `<a>` amb la marca `<img>` ens permet fer que una imatge sigui alhora un enllaç. Però si allò que volem és que segons on premem de la imatge vagi a un enllaç o a un altre haurem de recórrer a un altre recurs: els mapes sensibles.

Els **mapes sensibles** són imatges que presenten múltiples enllaços segons la zona en què se situï el cursor.

Hi ha dos tipus de mapes, els mapes gestionats des del client i els mapes gestionats des del servidor.

### 2.10.1. Mapes gestionats des del client

La primera tasca que cal fer quan confeccionem un mapa sensible gestionat des del client és dividir la imatge en zones. Per tal de fer això hem de delimitar aquestes zones establint-ne les coordenades.

Hi ha diversos programes de dibuix que ens permeten veure les coordenades de cada punt de la imatge i també alguns editors d'(X)HTML que ofereixen aquesta possibilitat.

Si volem utilitzar mapes sensibles, hem de fer dues actuacions: declarar el mapa i assignar-lo a una imatge. Veiem com podem fer aquests dos passos:

**1)** La declaració d'un mapa es fa amb la utilització de la marca `<map>` seguint la sintaxi següent:

```
<map name="nom_mapa">
  <area shape=... coords=... href=... alt=... />
  ...
</map>
```

#### Eines de confecció de mapes

Un exemple d'editor d'(X)HTML que ens permet trossejar la imatge i donar-nos les coordenades és l'editor *Quanta* amb el connector *kimagemapeditor*.

A la marca `<map>` només hi podem definir el nom del mapa. És dins de cada marca `<area>` que podem definir més dades, els atributs per fer-ho són els següents:

- `shape`: defineix la forma de la zona, que pot ser rectangular (`rect`), circular (`circ`) o poligonal (`poly`).
- `coords`: defineix les coordenades de la zona, seguint la normativa següent:
  - Si es tracta d'un rectangle (`rect`), cal indicar les coordenades  $x_1, y_1, x_2, y_2$ , on  $x_1$  i  $y_1$  són les coordenades de l'extrem superior esquerre, i  $x_2$  i  $y_2$ , les de l'extrem inferior dret.
  - Si es tracta d'un cercle (`circ`), cal indicar les coordenades  $x, y, r$ , on  $x$  i  $y$  són les coordenades del centre del cercle, i  $r$  el radi d'aquest cercle.
  - Si es tracta d'un polígon (`poly`), cal indicar les coordenades  $x_1, y_1, x_2, y_2, x_3, y_3, \dots$ , on cada parella  $x_i, y_i$  defineix un vèrtex del polígon. La darrera parella s'uneix a la primera per tancar el polígon.
- `href`: defineix l'enllaç cap a la destinació de l'usuari si prem sobre la zona.
- `nohref`: si l'establim (`nohref="nohref"`) indiquem que la zona no té cap enllaç definit.
- `alt`: incorpora el text que es presenta en lloc de la imatge en navegadors no visuals per accedir a l'enllaç.

2) L'assignació d'un mapa a una imatge es fa amb l'atribut `usemap` dins la marca `<img>`, que declara la imatge segons la sintaxi següent:

---

```

```

---

El nom del mapa sempre ha d'anar precedit del símbol `#` i ha de coincidir amb l'atribut `name` de la marca `<map>`.

Imaginem que tenim la imatge que es veu en la figura 22, i volem que en prémer sobre el cap anem a la URL <http://fsf.org>, en prémer sobre la panxa anem a <http://gnu.org> i en prémer els peus anem a <http://debian.org>, el codi seria el següent:

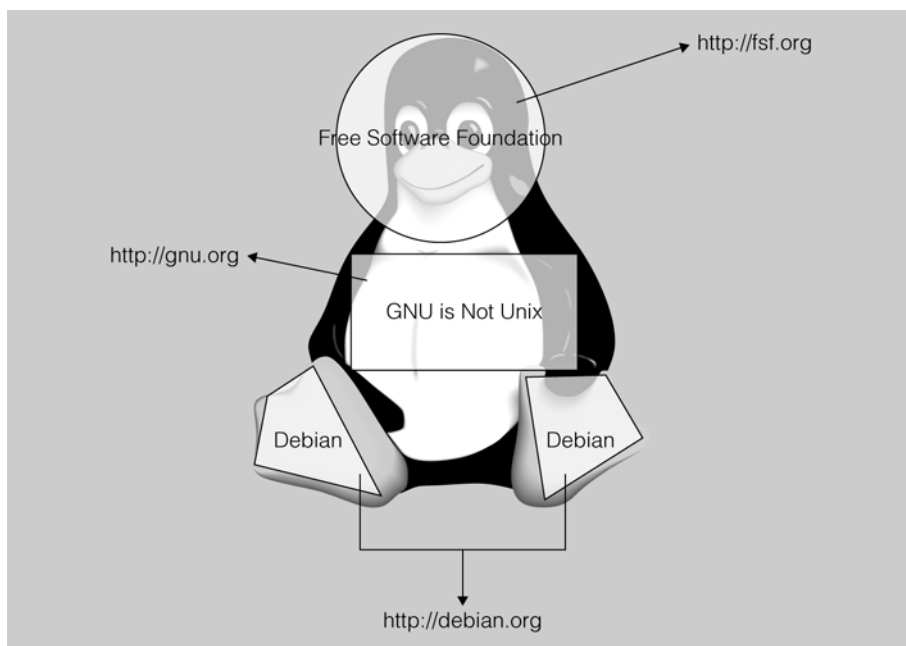
---

```

<map name="linuxmapa" id="mapa">
  <area shape="circle" coords="74,43,40" href="http://fsf.org" alt="Free Software
Foundation" title="FSF" />
  <area shape="rect" coords="42,89,126,133" href="http://gnu.org" alt="GNU is Not Unix"
title="GNU" />
  <area shape="poly" coords="27,130,10,141,5,167,53,179" href="http://debian.org"
alt="Debian" title="Debian" />
  <area shape="poly" coords="108,133,138,134,152,158,114,182" href="http://debian.org"
alt="Debian" title="Debian" />
  <area shape="default" nohref="nohref" alt="resta" />
</map>
```

---

Figura 22. Mapa de tipus client



### 2.10.2. Mapes gestionats des del servidor

L'objectiu dels mapes gestionats des del servidor és enviar al servidor les coordenades del punt on s'ha premut. En el servidor hi ha un programa que sap com s'han de gestionar aquestes coordenades (per exemple, enviar a una altra pàgina segons on s'hagi premut). Per tal de fer això hem d'usar l'atribut `ismap="ismap"` de la marca `<img>`:

---

```
<a href="paginadesti"><img src="" alt="" ismap="ismap" /></a>
```

---

Per exemple:

---

```
<a href="mapaservcoord.html"></a>
```

---

Això fa que si premem les coordenades 5,15 de la imatge anem a la pàgina *mapaservcoord.html?5,15*. La pàgina de destinació pot agafar la part de la URL que conté les coordenades i gestionar-les com convingui.

## 2.11. Objectes

La marca `<object>` neix per oferir una solució universal per a la inclusió de qualsevol tipus de fitxers en els documents (X)HTML, de manera que els navegadors rebin la informació necessària per tractar l'objecte (fitxer incrustat en el document).

Com a regla general, la marca `<object>` serveix per definir un objecte o component extern al navegador que s'encarrega de reproduir l'objec-

te (so, animació, vídeo...). Per aconseguir una reproducció correcta, la marca `<object>` ha de permetre declarar l'objecte, la seva ubicació i el seu tipus, i unes marques optatives especials, `<params>`, permeten acabar de definir els valors que necessiti.

La sintaxi genèrica és la següent:

```
<object atribut1="valor1" atribut2="valor2" ... atributN="valorN">
  <param name="nom" value="valor">
  <param name="nom" value="valor">
  ...
</object>
```

Un atribut bàsic en la declaració d'un objecte és l'atribut `type`, que permet indicar al navegador el tipus d'objecte que ha de carregar. El navegador ha d'utilitzar aquesta informació per esbrinar si disposa d'alguna aplicació adequada per reproduir l'objecte, i n'ha d'avortar la càrrega si no en disposa.

De tot això es dedueix que hi ha dos punts importants perquè un objecte es pugui reproduir correctament:

- 1) El tipus de l'objecte ha d'estar ben identificat en el document (X)HTML. Per a això, cal utilitzar els tipus MIME.
- 2) El navegador ha de tenir instal·lada alguna aplicació adequada per tal de reproduir l'objecte. Les aplicacions que possibiliten aquesta funció s'anomenen connectors (*plugins*).

La taula 7 mostra una llista dels atributs més utilitzats en la marca `<object>`.

Taula 7. Atributs d'`<object>`

Atribut	Descripció
<code>data</code>	Nom del fitxer per reproduir amb la seva ubicació.
<code>type</code>	Cadena amb el tipus MIME adequat a l'objecte.
<code>width</code>	Amplada en píxels del control.
<code>height</code>	Alçada en píxels del control.
<code>standby</code>	Missatge que apareix en pantalla mentre l'objecte es carrega.

Per exemple, si volem reproduir un arxiu de música, hem de fer el següent:

```
<object data="musica.mp3" type="audio/mpeg">
  <p>Text alternatiu</p>
</object>
```



Quant als paràmetres, depenent de quin objecte vulguem encastar, en podem posar uns o uns altres.

### 2.11.1. Tipus MIME

**MIME** és l'acrònim anglès de *Multipurpose Internet Mail Extensions* (extensions multipropòsit de correu d'Internet). Es tracta d'un estàndard que especifica com un programa (inicialment, de correu o navegador web) ha de transferir arxius multimèdia (vídeo, so i, per extensió, qualsevol arxiu que no estigui codificat en US-ASCII). Abans dels tipus MIME, qualsevol arxiu que no es limités a text ASCII s'havia de codificar en US-ASCII.

L'estàndard MIME adjunta un fitxer de capçalera a cada fitxer, que especifica el tipus i el subtipus del contingut de l'arxiu principal. Gràcies a aquesta informació, tant el servidor com el navegador poden gestionar i presentar correctament les dades.

En la utilització diària d'Internet, ens beneficiem dels tipus MIME. Cada vegada que sol·licitem una pàgina d'Internet, s'obre un diàleg entre el nostre navegador i el servidor que proporciona la pàgina. El nostre navegador demana la pàgina i el servidor, abans d'enviar-la, confirma que existeix i comprova el tipus de dades que conté. Això darrer es fa mitjançant el tipus MIME que correspongui.

La gestió de tipus MIME en el web té lloc en tres punts ben diferenciats:

- 1) En el servidor, que ha de ser capaç de gestionar diversos tipus MIME i tenir-los activats.
- 2) En la pàgina web, en què l'autor constantment fa referència a tipus MIME. Així, els enllaços a arxius externs (fulls d'estils, scripts de JavaScript, objectes incrustats...) han d'especificar el tipus de l'arxiu enllaçat. A continuació posem alguns exemples:

**a) Enllaços a fulls d'estils:**

---

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

---

**b) Crida a scripts JavaScript:**

---

```
<script type="text/javascript" src="codi.js"></script>
```

---

**c) Definició d'objectes:**

---

```
<object data="musica.mp3" type="audio/mpeg">  
  <p>Text alternatiu</p>  
</object>
```

---

3) En el navegador del client, que ha d'estar capacitat per interpretar els tipus MIME que el servidor li envia i, fins i tot, ha de poder informar el servidor dels tipus MIME que pot acceptar.

Per permetre aquesta funció, el navegador ha de tenir instal·lades les aplicacions adequades als diferents tipus MIME que interressi gestionar. Són els connectors (*plugins*).

Actualment, els tipus MIME s'agrupen en vuit categories, i cada tipus MIME s'identifica pel nom compost: categoria/tipus. De vegades, es parla de tipus/subtipus. Les vuit categories són les següents: *application*, *audio*, *image*, *message*, *model*, *multipart*, *text* i *video*. Cadascuna està formada per un conjunt més o menys gran de tipus MIME, i cadascun dels tipus acostuma a dur associades una extensió d'arxius o més d'una.

La taula 8 recull alguns dels tipus MIME amb algunes de les extensions d'arxius normalment associades.

Taula 8. Recull de tipus MIME i extensions associades d'arxius

Tipus/Subtipus MIME	Extensions de fitxers	Tipus/Subtipus MIME	Extensions de fitxers
image/png	.png	video/x-msvideo	.avi
image/gif	.gif	application/pdf	.pdf
image/jpeg	.jpg, .jpeg, .jpe	application/postscript	.ai, .eps, .ps
image/tiff	.tif, .tiff	application/rtf	.rtf
audio/x-wav	.wav	application/gzip	.gz
audio/x-midi	.mid	application/x-tar	.tar
text/plain	.txt	application/zip	.zip
text/richtext	.rtf, .rtx	application/x-java-vm	.class
video/mpeg	.mpeg, .mpg, .mpe	application/x-java-archive	.jar

## 2.11.2. Connectors

Per reproduir arxius definits dins les pàgines web com a objectes genèrics, els navegadors necessiten tenir instal·lat el connector adequat.

Els **connectors** (*plugins*) són aplicacions informàtiques que interactuen amb una altra aplicació per afegir-hi una funció o utilitat específica.

És a dir, si en un document web s'hi ha incrustat un objecte (<object>) que fa referència a un document d'un cert tipus MIME, el navegador que



Internet Assigned Numbers Authority

Logotip d'IANA

### IANA

L'organisme que s'encarrega de registrar els tipus MIME és l'IANA, acrònim anglès d'*Internet Assigned Numbers Authority* (Agència d'Assignació de Números d'Internet). A la seva pàgina web podem trobar la llista completa de tipus/subtipus MIME.

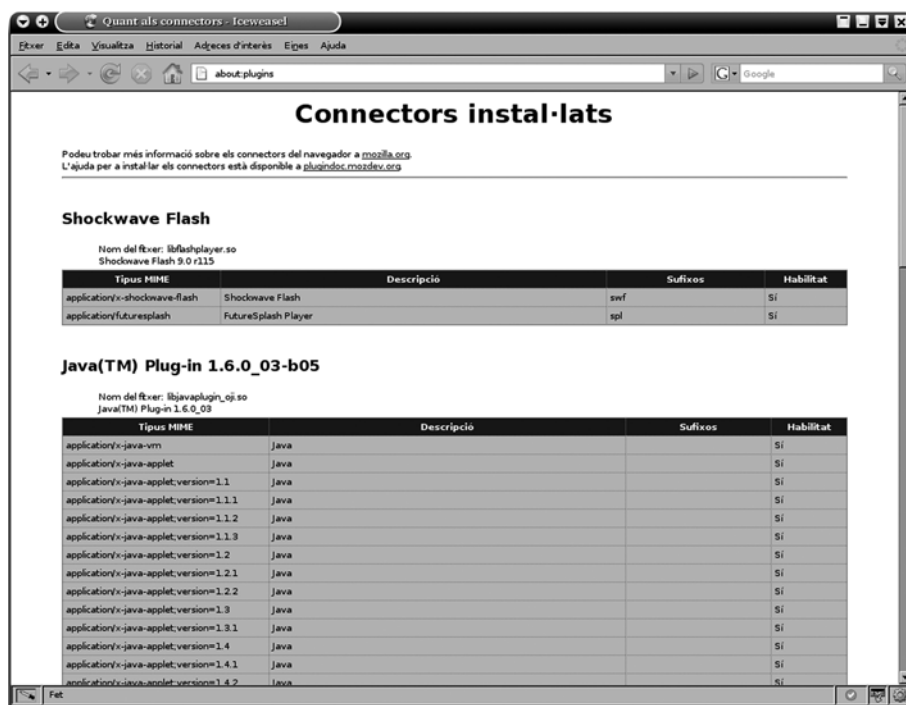
l'hagi de reproduir ha de disposar d'un connector adequat per a aquell tipus MIME.

El fet que el sistema operatiu disposi d'una aplicació que pugui reproduir un determinat arxiu (corresponent a un tipus MIME) no és garantia que els navegadors instal·lats en aquell sistema tinguin instal·lat el connector corresponent.

Els navegadors, en rebre pàgines web que contenen tipus MIME no suportats, haurien d'oferir a l'usuari (recomanació del W3C) la possibilitat d'instal·lar-les d'una manera senzilla.

Si es vol saber els connectors que es tenen instal·lats al navegador Firefox s'ha d'escriure *about:plugins* en la barra de navegació. El navegador ens mostra un document HTML amb la informació corresponent a tots els connectors instal·lats, com es veu en la figura 23. La informació que dona és molt interessant: connectors instal·lats (versió i nom de l'arxiu corresponent) i, per a cada connector, la relació de tipus MIME mitjançant els quals pot donar servei.

Figura 23. Connectors instal·lats al navegador Firefox



### 2.11.3. Exemples d'incorporació d'objectes genèrics en una pàgina web

A continuació es mostren diversos exemples de com podem incrustar alguns tipus de mitjans multimèdia a la nostra web. En tots els exemples, si no es pot accedir al mitjà, es mostra un text que ofereix la possibilitat de baixar-se el fitxer directament.

- Inserció d'un vídeo en format *.mpg*:

---

```
<object data="media/Miraquiensequeja.mpg" type="application/x-mplayer2" width="500"
height="500">
  No es pot accedir al video. Baixa-te'l d'<a href="media/Miraquiensequeja.mpg">aqui</a>
</object>
```

---

- Inserció d'un fitxer àudio en format *.mp3*:

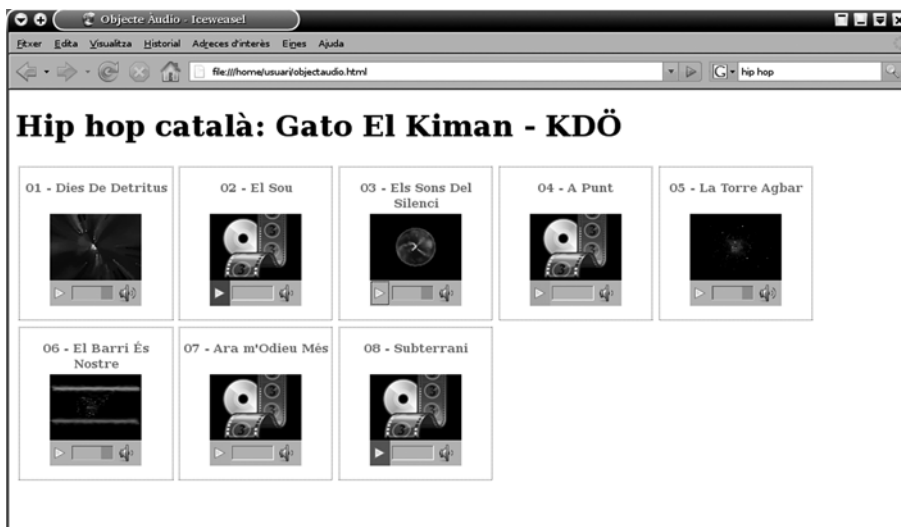
---

```
<div class="canco">
  <h2>01 - Bubamara (main version)</h2>
  <object data="media/01DiesDeDetritus.mp3" type="audio/mpeg" width="100" height="100">
    <param name="autoplay" value="false" />
    <param name="autoStart" value="0" />
    No es pot accedir a l'audio. Baixa-te'l d'<a href="media/01DiesDeDetritus.mp3">aqui</a>
  </object>
</div>
```

---

En la figura 24 podem veure totes les cançons d'un àlbum, on cada cançó està incrustada com un objecte. En aquest cas s'està usant el connector de Totem (programa de visionat de vídeo i àudio en sistemes GNU/Linux).

Figura 24. Objectes d'àudio inserits en una pàgina web



- Inserció d'un format *.swf* (aplicació Flash):

---

```
<object type="application/x-shockwave-flash" data="media/calculator.swf" width="400"
height="400">
  <param name="src" value="media/calculator.swf" />
</object>
```

---

### 3. Els fulls d'estils: CSS

En els orígens del web i en les seves primeres versions, el llenguatge de marques HTML era un llenguatge fàcil d'aprendre i molt reduït quant a les marques i l'estructura. Tot va canviar quan van aparèixer els primers navegadors que eren capaços de representar recursos gràfics per afegir a la informació textual.

D'aquesta manera, el nombre de llocs web va començar a créixer i, amb ell, el nombre de marques que l'especificació HTML preveia. L'objectiu era construir llocs web cada vegada més atractius i, per a això, l'HTML havia d'incloure noves marques destinades a aconseguir efectes visuals.

Amb tots aquests canvis que el web va patir, ens trobem amb un llenguatge que, si bé al principi estava "orientat a l'estructura", ara està totalment "orientat a la presentació".

Així trobem que l'HTML ofereix marques com `<b>`, `<u>`, `<i>` o `<font>`, que són marques de visualització i no aporten res a l'estructura del document representat.

El naixement d'XHTML vol posar fi a aquest problema: les marques permeses en XHTML són marques per marcar l'estructura, no la presentació. Ens falta, però, alguna manera de definir la presentació del document i això és el que ens oferirà l'ús dels fulls d'estils. La finalitat de l'ús de fulls d'estils és, doncs, a més d'incorporar el disseny estètic a la nostra aplicació web, mantenir l'estructura i el disseny separats.

Usar fulls d'estils ens proporciona els avantatges següents:

- Possibilitat de mantenir el codi.
- En el camp de disseny, el CSS és més potent que les marques de disseny que oferia l'HTML.
- El CSS és un llenguatge senzill.
- Es poden especificar diferents fulls d'estils per a un sol document (X)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.

El gran inconvenient dels fulls d'estils és que no tots els navegadors es comporten de la mateixa manera davant del mateix full d'estils. Això es deu al fet que alguns navegadors no compleixen els estàndards establerts i, amb això, obliguen el programador a codificar diferents fulls d'estils (un per a cada navegador).



#### Validació de fulls d'estils

Igual que podem validar documents (X)HTML, també podem validar els nostres fulls d'estils per comprovar que segueixen els estàndards proposats pel W3C.

A més, hem de tenir en compte que hi ha dues especificacions oficials de CSS: CSS1 i CSS2.

### Especificacions CSS

“Els fulls d'estil estan formalment descrits en dues especificacions del W3C: CSS1 i CSS2. CSS1 es va emetre el desembre de 1996 i descriu un model de formatació simple, principalment per a presentacions basades en la pantalla. CSS1 té al voltant de 50 propietats (per exemple color i mida de font). CSS2 s'enllestia el maig de 1998 i està basada en CSS1. CSS2 inclou totes les propietats del CSS1 i n'afegeix al voltant de 70 més, com per exemple propietats per descriure presentacions auditives i salts de pàgina[...]. Si vol llegir les especificacions de CSS, les pot trobar a:

- <http://www.w3.org/TR/REC-CSS1>
- <http://www.w3.org/TR/REC-CSS2>

Håkon Wium Lie; Bert Bos (1999). *Cascading Style Sheets: Designing for the Web* (capítol 2).

Disponible a: <http://www.w3.org/Style/LieBos2e/enter>

Hem de tenir en compte que la majoria de navegadors suporten CSS1, però no tots suporten totes les propietats que ofereix CSS2.

### 3.1. Ubicació dels estils

El CSS ens defineix quin aspecte han de tenir els elements (X)HTML. És per això que hem d'associar d'alguna manera aquestes propietats als elements.

Podem ubicar les propietats CSS en diferents localitzacions:

- **En la marca:** afegint les propietats CSS directament a l'element usant l'atribut `style`. Suposem que volem que un determinat paràgraf estigui centrat amb lletra vermella. El codi és el següent:

---

```
<p style="text-align:center; color:red">Paràgraf centrat vermell</p>
```

---

- **En la capçalera del document (X)HTML:** podem posar les diferents propietats CSS dins de l'element `<style>` que està ubicat dins l'element `<head>`. Suposem que volem que tots els paràgrafs estiguin centrats i amb lletra vermella. El codi és el següent:

---

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3c.org/TR/xhtml1/DTD/xhtml1-strict.dtd" >
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ca" lang="ca">
<head>
  ...
  <style>
    p {
      text-align:center;
      color:red
    }
  </style>
</head>
<body>
  <p>Paràgraf centrat vermell</p>
</body>
</html>
```

---

- **En un document extern:** posem totes les propietats dins d'un document amb extensió `.css` i des del document (X)HTML enllacem aquest full d'estils amb l'ajut de la marca `<link>`, filla de l'element `<head>`. Suposem que volem tenir tots els paràgrafs centrats amb lletra vermella. El document (X)HTML és:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3c.org/TR/xhtml1/DTD/xhtml1-strict.dtd" >
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ca" lang="ca">
<head>
  <link rel="stylesheet" href="estils.css" type="text/css" />
</head>
<body>
  <p>Paràgraf centrat vermell</p>
</body>
</html>
```

El fitxer `estils.css` té el contingut següent:

```
p {
  text-align:center;
  color:red
}
```

L'avantatge d'usar documents externs per establir les propietats CSS és que podem reutilitzar un mateix full d'estils per a diferents documents (X)HTML.

### 3.2. Sintaxi bàsica de CSS

Un full d'estils és un conjunt de **regles** que defineixen l'estètica final dels documents (X)HTML que l'usen. Cadascuna d'aquestes regles està formada per un **selector** i per un conjunt de **declaracions**.

Una declaració està formada per una **propietat** amb el seu **valor** associat.

El **selector** ens serveix per definir a quin o a quins elements volem aplicar les declaracions de la regla. Les **declaracions** són les diverses característiques que ha de complir el selector. A cada **propietat** de cada declaració hi posem el **valor** que s'escaigui.

La sintaxi genèrica d'una regla és, doncs, la següent:

```
selector{
  declaració_1
  ...
  declaració_n
}
```

on la sintaxi de cada *declaracio\_i* és:

---

```
propietat_i: valor_i;
```

---

Per exemple, si volem que tots els paràgrafs tinguin lletra de mida 10 pt i un fons de color gris hem de definir la regla següent:

---

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

---

### 3.2.1. Les regles arrova

Hi ha un conjunt de regles especials que s'anomenen **regles arrova**. Aquestes regles es caracteritzen perquè comencen pel caràcter “@”. Les regles arrova més importants són `@import` i `@media`:

- **Regla `@import`.** La regla `@import` ens serveix per incloure, en el nostre full d'estils, fulls d'estils externs. Si, per exemple, volem incloure en el nostre full d'estils totes les propietats que hi ha en el document *mesestils.css*, hem d'escriure la línia següent:

---

```
@import "mesestils.css";
```

---

- **Regla `@media`.** La regla `@media` serveix per diferenciar per quin mitjà s'ofereixen les propietats que conté aquesta regla. 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.

Imaginem, per exemple, que volem que, quan imprimim el document (X)HTML, la lletra tingui una mida de 10 pt, però que, quan es miri per pantalla, tingui una mida de 12 pt. També volem que en ambdós casos l'alçada de la línia sigui d'1.2. En el nostre full d'estils hem d'introduir el codi següent:

---

```
@media print {  
  body{ font-size: 10pt }  
}  
@media screen {  
  body { font-size: 12pt }  
}  
@media screen, print {  
  body { line-height: 1.2 }  
}
```

---



### 3.2.2. Comentaris

Si en un full d'estils volem posar comentaris destinats a l'aclariment del codi CSS hem de fer servir la sintaxi següent:

---

```
/* comentaris */
```

---

### 3.3. Cascada i herència

Dues de les característiques que fan que els fulls d'estils tinguin una gran potència a l'hora d'implementar-los és la cascada i l'herència.

La cascada es refereix a la possible combinació de diferents fulls d'estils. L'herència fa referència a la capacitat que tenen els elements del document (X)HTML d'heretar propietats dels seus elements antecessors.

#### 3.3.1. Cascada

Les sigles CSS volen dir *Cascading Style Sheets* (Fulls d'estils en cascada). Però, què vol dir cascada en aquest àmbit? Doncs vol dir que podem combinar diferents fulls d'estils i que les propietats de tots ells es van acumulant. Això ens és molt útil quan pensem en llocs web grans, on podem tenir un full d'estils bàsic i anar incorporant-hi altres fulls d'estils, segons les nostres necessitats.

Ara caldria preguntar-nos si podem combinar diferents fulls d'estils i si hi ha propietats que es contradiuen, com se soluciona. Quina propietat té prioritat? Per solucionar aquest problema hi ha establert la jerarquia següent:

**1) Propietats establertes per l'autor de la web.** Les propietats que estableix l'autor es poden ubicar en diferents llocs i, segons aquesta ubicació, la propietat té més o menys prioritat. La jerarquia és la següent, per ordre de quina preval sobre les altres:

- a) Propietats establertes en l'atribut `style` d'un element.
- b) Propietats establertes en l'element `<style>` del document (X)HTML.
- c) Propietats establertes en un full d'estils extern.

**2) Propietats establertes per l'usuari.** Els navegadors permeten a l'usuari establir diferents propietats d'estil: mida de la lletra, colors...

**3) Propietats establertes pel navegador.** Els navegadors tenen un estil predeterminat per a cadascun dels elements (X)HTML. Si no s'ha esta-

blert cap propietat en els ítems anteriors, les propietats del navegador són les que prevalen.

Podem invertir l'ordre anterior posant la cadena `!important` després d'una declaració. Això fa que l'estil d'usuari prevalgui davant de la propietat especificada per l'autor. Aquest fet ens serveix perquè la pàgina sigui més accessible, ja que si, per exemple, algú té problemes de visió, li permetem que pugui fer la lletra més gran, tot i que l'aplicació estigui pensada per ser visualitzada amb una lletra més petita.

Si ens trobem en el cas que les propietats “empaten”, l'última propietat escrita és la que “guanya”. Si hem escrit el codi següent en un full d'estils extern:

---

```
p{
  color:red;
}
...
p{
  color:green;
}
...
```

---

En aquest cas la lletra dels paràgrafs serà de color verd, ja que és l'última escrita.

### 3.3.2. Herència

Cada element d'una pàgina (X)HTML està contingut en un altre. Quan parlem d'herència ens referim al fet que tot element hereta les propietats dels seus elements antecessors. Malgrat tot, hem de tenir en compte el següent:

- No totes les propietats s'hereten, aquesta característica està descrita en l'especificació de CSS corresponent.
- Si volem forçar l'herència en una propietat podem introduir-hi el valor `inherit`.
- Si posem un valor a una propietat, aquest valor preval sobre el valor heretat.
- Els elements hereten el valor computat, no el valor especificat. És a dir, si una propietat té per valor un valor relatiu (per exemple, un percentatge), el valor heretat és el resultat calculat.

### 3.4. Selectors

Un **selector** és la part de la regla CSS que identifica l'element o elements del document (X)HTML dels quals volem definir les propietats.

Hi ha diferents tipus de selectors: des dels més simples, en què simplement és posar la marca (X)HTML a la qual volem modificar l'estètica, fins a combinacions complexes d'aquests selectors.

### 3.4.1. Selectors de tipus

Els **selectors de tipus** són els selectors més simples: es tracta únicament d'introduir la marca (X)HTML. Afecten tots els elements (X)HTML amb aquella marca.

Imaginem que volem que tots els encapçalaments de primer ordre, amb marca `<h1>`, siguin de color blau. El codi és el següent:

---

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

---

Però, i si no volem modificar l'aparença de tots els elements d'un mateix tipus? Per aconseguir això hem d'usar altres tipus de selectors, com ara els selectors de classes i els selectors d'identificadors.

### 3.4.2. Selectors de classes

Els selectors de classes serveixen per associar unes propietats a tots els elements que tinguin l'atribut `class` d'igual valor. Si en el document (X)HTML tenim:

---

```
<tag class="nomclasse">...</tag>
```

---

podem introduir la regla següent amb un selector de classe:

---

```
tag.nomclasse{  
  declaracions  
}
```

---

Això fa que a tots els elements *tag* que tinguin l'atribut `class` amb valor *nomclasse* se'ls apliquin les *propietats* especificades.

També podem definir un selector de classe més genèric, que no depengui de la marca de l'element. És a dir, podem definir la regla següent:

---

```
.nomclasse{  
  declaracions  
}
```

---

Aquesta regla fa que qualsevol element, sigui quina sigui la seva marca, si té l'atribut `class` amb valor "nomclasse" tingui la lletra de color verd.

Posem un exemple: volem que tots els paràgrafs siguin de color verd, excepte els de classe `destacat`, que són de color vermell. El codi CSS és el següent:

---

```
p{
  color:green;
}
p.destacat{
  color:red;
}
```

---

Qualsevol paràgraf, amb marca `<p>` té el color de lletra verd, però si introduïm un paràgraf amb classe `destacat`, `<p class="destacat">`, la lletra d'aquest paràgraf és de color vermell.

Si el que volem és que qualsevol element amb `class="destacat"` tingui la lletra vermella hem d'escriure el codi següent:

---

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

---

### 3.4.3. Selectors d'identificador

Els selectors d'identificador serveixen per associar unes propietats a l'element que té un determinat identificador. L'identificador d'un element (X)HTML es declara amb l'atribut `id` de l'element. Hem de tenir en compte que en un document (X)HTML no hi pot haver dos elements amb el mateix identificador. Si en el document (X)HTML tenim:

---

```
<tag id="nomidentificador">...</tag>
```

---

podem escriure la regla següent amb un selector d'identificador:

---

```
tag#nomidentificador{
  declaracions
}
```

---

Podem introduir la mateixa regla prescindint de quin element porti l'identificador, és a dir, podem dir:

---

```
#nomidentificador{
  declaracions
}
```

---

Si, per exemple, volem que l'element amb identificador `principal`, per exemple `<h1 id="principal">`, estigui centrat hem d'escriure:

---

```
#principal{
  text-align:center;
}
```

---

### 3.4.4. Selectors d'atribut

Els selectors d'atribut serviran per aplicar les propietats de la regla als elements que tenen un atribut concret. El selector següent:

---

```
tag[atribut] {  
    declaracions  
}
```

---

afecta tots els elements (X)HTML amb la sintaxi següent:

---

```
<tag atribut="valor">...</tag>
```

---

Per exemple, si volem que hi hagi un marge dret de 10 px en tots els elements `<img>` que tinguin establert l'atribut `title`, hem d'escriure el següent:

---

```
img[title] {  
    margin-right:10px;  
}
```

---

Aquest selector permet diverses variants. Una d'elles és que podem jugar amb el valor de l'atribut. En l'exemple anterior podem haver decidit que volem un marge dret de 10 px en totes les imatges que tenen establert l'atribut `title` i que, a més, aquest atribut ha de tenir per valor "logo". El codi CSS ha de ser el següent:

---

```
img[title="logo"] {  
    margin-right:10px;  
}
```

---

També podem filtrar per més d'un atribut. Per exemple, si ara volem seleccionar tots els elements `<img>` que tenen l'atribut `title` i que són de classe important, tindrem el codi següent:

---

```
img[title][class="preferent"] {  
    margin-right:10px;  
}
```

---

Finalment, podem filtrar segons el valor de l'atribut. Disposem de dues fórmules per fer-ho:

- `[atribut~="valor"]`: s'usa per seleccionar els elements que tenen com a atribut una llista de paraules separades per espais, una de les quals és exactament *valor*.
- `[atribut|=valor]`: s'usa per seleccionar els elements que tenen com a atribut una llista de paraules separades per guions, començant per *valor*.

Posem un exemple per a cada cas:

---

```
img[alt~="logo"] {  
    border: solid  
}  
  
p[lang|="en"] {  
    font-family: "Times New Roman", Serif  
}
```

---

La primera regla selecciona les imatges en les quals l'atribut `alt` té per valor una cadena que conté la paraula `logo`, com per exemple `alt="logo de la web"`. La segona regla selecciona els paràgrafs que tenen com a valor de l'atribut `lang`, paraules que comencen amb `en`, com `en-US` o `en-cockney`.

### 3.4.5. Selector universal

El selector `*` s'anomena *selector universal* i afecta tots els elements del document. La sintaxi és la següent:

---

```
* {  
    declaracions  
}
```

---

### 3.4.6. Selectors de descendents

Diem que un element és descendent d'un altre quan aquest element està contingut en l'altre, no importa a quin nivell. Un selector de descendents ens permet seleccionar tots els elements que estan continguts en un altre. La sintaxi genèrica és la següent:

---

```
tag tagdescendent {  
    declaracions  
}
```

---

Per exemple, si volem posar en blau tots els elements `<em>` continguts en un paràgraf hem d'escriure:

---

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

---

Podem posar més nivells i barrejar altres tipus de selectors. Imaginem un exemple més complex: volem posar amb lletra vermella tots els elements de classe `destaca` que estiguin continguts en paràgrafs que alhora estan continguts en un `<div>`:

---

```
div p .destaca {  
    color:red;  
}
```

---

### 3.4.7. Selectors de fills

Un element és fill d'un altre si és descendent de primer nivell. Els selectors de fills ens permeten seleccionar els fills d'un determinat element. El símbol que hem d'usar és >:

---

```
tag>tagfill {
    declaracions
}
```

---

Per exemple, si tenim les regles següents en el nostre document CSS:

---

```
body>p { font-weight: bold; }
div ol>li p { text-decoration : underline; }
```

---

La primera línia de l'exemple fa que tots els paràgrafs que són fills de <body> estiguin en lletra negreta.

La segona regla és una mica més complexa: el selector afecta tots els elements <p> que són descendents d'un element <li>, on l'element <li> ha de ser fill d'un element <ol> que, alhora, ha de ser descendent d'un element <div>.

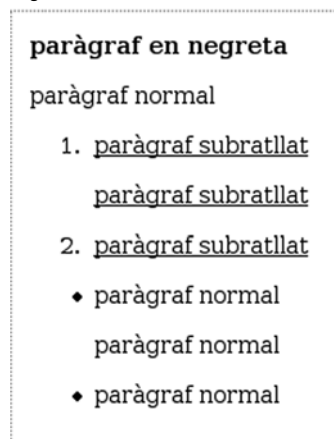
Aquests dos exemples combinats amb el codi (X)HTML següent fan que observem el resultat que es mostra en la figura 25 en el nostre navegador:

---

```
<body>
  <p>paràgraf en negreta</p>
  <div>
    <p>paràgraf normal</p>
    <div>
      <ol>
        <li>
          <p>paràgraf subratllat</p>
          <p>paràgraf subratllat</p>
        </li>
        <li>
          <div>
            <p>paràgraf subratllat</p>
          </div>
        </li>
      </ol>
      <ul>
        <li>
          <p>paràgraf normal</p>
          <p>paràgraf normal</p>
        </li>
        <li>
          <div>
            <p>paràgraf normal</p>
          </div>
        </li>
      </ul>
    </div>
  </div>
</body>
```

---

Figura 25. Selectors de fill



### 3.4.8. Selectors de germans adjacents

Els selectors de germans adjacents ens permeten, donat un element, conferir propietats al seu germà immediatament posterior. El símbol usat és +:

---

```
tag+taggermaadjacent {  
    declaracions  
}
```

---

Si, per exemple, volem reduir l'espai vertical que hi ha entre un <h1> i l'<h2> que el segueix immediatament, hem de definir la regla següent:

---

```
h1 + h2 { margin-top: -5mm }
```

---

Obtindrem el resultat que podem veure en la figura 26.

Figura 26. Selectors de germans adjacents



Complicant una mica la regla anterior, fa que la regla només s'apliqui si l'element <h1> és de classe oberta (class="obert"):

---

```
h1.obert + h2 { margin-top: -5mm }
```

---

### 3.4.9. Agrupament de selectors

Si diversos selectors tenen les mateixes propietats podem agrupar-los separant els selectors per una coma:

---

```
selector1,selector2,selector3 {  
    declaracions  
}
```

---

És a dir, si, per exemple, tenim les regles següents:

---

```
h1 { font-family: sans-serif }  
h2 { font-family: sans-serif }  
h3 { font-family: sans-serif }
```

---

és equivalent a la regla següent, en què hem agrupat els tres selectors:

---

```
h1, h2, h3 { font-family: sans-serif }
```

---



### 3.5. Pseudoclasses i pseudoelements

Les pseudoclasses i els pseudoelements serveixen per afegir alguns efectes addicionals als elements que ens interressi.

#### 3.5.1. Pseudoclasses

A continuació enumerem les pseudoclasses que hi ha amb la seva descripció i alguns exemples:

- **:first-child.** Selecciona un element que és el primer fill d'un altre element. Per exemple,

CSS:

---

```
/*els paràgrafs fills de div.nota tindran un sagnat d'1 em*/
div.nota > p { text-indent:1em; }

/* els paràgrafs primer fill de div.nota tindran un sagnat de 2em */
div.nota > p:first-child { text-indent:2em; }
```

---

(X)HTML:

---

```
<body>
  <p> Paràgraf exterior al <div></p>
  <div class="nota">
    <p>Paràgraf primer fill de <div></p>
    <p>Paràgraf segon fill de <div></p>
  </div>
</body>
```

---

En la figura 27 podem veure el resultat d'aquest exemple.

- **:link.** Permet definir l'estil dels enllaços de la nostra pàgina, quan encara no s'han visitat.
- **:visited.** Permet definir l'estil dels enllaços de la nostra pàgina, quan s'han visitat. Per exemple,

CSS:

---

```
/* Els enllaços són de lletra blanca damunt de fons negre */
a {color: white; background-color: black;}
/* Quan no han estat visitats no tindran cap tipus de subratllat */
a:link { text-decoration: none;}
/* Quan han estat visitats apareixeran tatxats */
a:visited { text-decoration: line-through;}
```

---

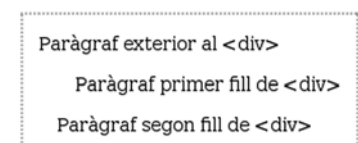
(X)HTML:

---

```
<p><a href="http://fsf.org">Enllaç normal</a></p>
<p><a href="http://gnu.org">Enllaç visitat</a></p>
```

---

Figura 27. Pseudoclasse  
:first-child



Podem veure el resultat en la figura 28.

Figura 28. Pseudoclasses d'enllaços



- **:active**. Permet definir l'estil dels elements quan s'activen (quan premem damunt seu).
- **:hover**. Permet definir l'estil dels elements quan passem per damunt seu.
- **:focus**. Permet definir l'estil dels elements quan reben el focus. Per exemple,

CSS:

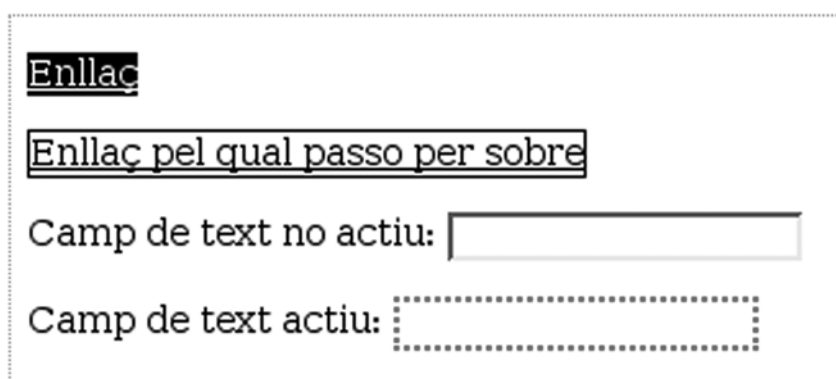
```
/* Els enllaços (visitats i no visitats) són de lletra blanca damunt de fons negre */
a:link, a:visited { color:white; background-color:black; }
/* Quan hi passem per damunt tenen una vora negra, la lletra serà negra i el fons blanc */
a:hover { border: solid thin black; color: black; background-color:white; }
/* Quan els premem, la lletra és blanca i el fons vermell */
a:active { color:white; background-color:red; }
/* Quan ens situem en un control de formulari <input> tindrà una vora vermella puntejada,
i un fraccionament de 2px */
input:focus { border: red 2px dotted; padding: 2px; }
```

(X)HTML:

```
<p><a href="http://fsf.org">Enllaç</a></p>
<p><a href="http://gnu.org">Enllaç pel qual passo per sobre</a></p>
<p>Camp de text no actiu: <input type="text" name="input" /></p>
<p>Camp de text actiu: <input type="text" name="input" /></p>
```

Una mostra del que veuríem la tenim en la figura 29.

Figura 29. Exemple de pseudoclasa :focus



- **:lang**. Permet definir les propietats segons l'idioma. Per exemple, si volem establir que els paràgrafs que tenen l'atribut lang amb valor ca (<p lang="ca">...</p>) estan amb lletra vermella, hem de definir-ho amb la regla següent:

```
p:lang(ca) {
  color:red;
}
```

### 3.5.2. Pseudoelements

A continuació enumerem els pseudoelements que hi ha amb la seva descripció:

- **:first-line.** Permet definir l'estil de la primera línia de l'element.
- **:first-letter.** Permet posar propietats d'estil a la primera lletra de l'element.
- **:after.** Permet introduir contingut al final de l'element. A l'element li hem de donar la propietat `content` amb el valor desitjat.
- **:before.** Permet introduir contingut a l'inici de l'element. A l'element li hem de donar la propietat `content` amb el valor desitjat.

A continuació exposem un exemple on participen tots aquests pseudoelements. Podeu observar el resultat en la figura 30.

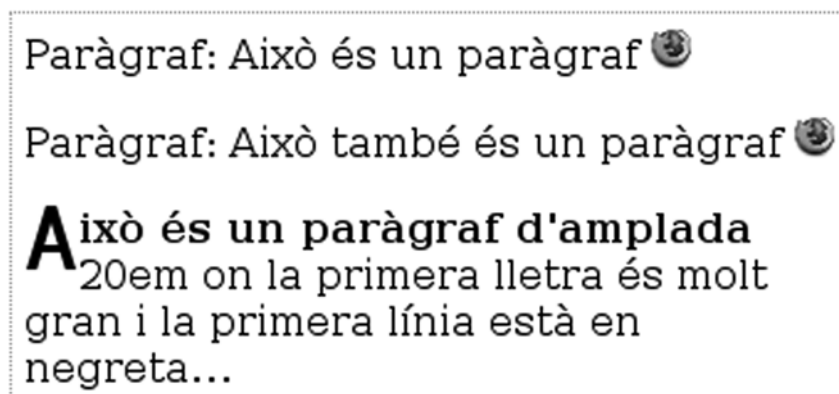
CSS:

```
/* La primera línia dels paràgrafs de classe ll estan en lletra negreta */
p.ll:first-line { font-weight:bold;}
/* La primera lletra dels paràgrafs de classe ll està flotant a l'esquerra amb lletra de
tipus monospace tres cops més gran */
p.ll:first-letter {float: left; font-size: 300%; font-family:monospace;}
/* Davant de cada paràgraf de classe pf hi haurà el text 'Paràgraf: ', després hi haurà un
logotip del navegador Firefox */
p.pf:before { content: "Paràgraf: "; }
p.pf:after { content: url("imatges/firefox.png"); }
```

(X)HTML:

```
<p class="pf">Això és un paràgraf </p>
<p class="pf">Això també és un paràgraf </p>
<p class="ll">Això és un paràgraf d'amplada 20em on la primera lletra és molt gran i la
primera línia està en negreta...</p>
```

Figura 30. Exemple sobre pseudoelements



### 3.6. Propietats bàsiques de format

Per tal de definir les declaracions d'una regla tenim disponibles un seguit de propietats que ens permeten modificar l'aparença dels elements (X)HTML: els colors, la tipografia, etc.

#### 3.6.1. Mesures i colors

Algunes de les propietats que podem establir en el full d'estils tenen com a valor una **unitat de mesura** que ens indica la mida de la propietat (mides de lletra, amplades, marges...). Aquestes unitats de mesura poden ser unitats relatives o absolutes:

- Les **unitats relatives** les podem mesurar amb la mesura *em*, amb la mesura *ex*, amb píxels (*px*) o amb percentatges (%).
- Les **unitats absolutes** es poden mesurar amb centímetres (*cm*), amb mil·límetres (*mm*), amb polzades (*in*) o amb punts (*pt*).

Quant a les unitats relatives, hem de destacar que una distància definida en *px* no canvia si l'usuari canvia la mida del text en el seu navegador o si l'usuari canvia la mida de la finestra del navegador, però sí que canvia si l'usuari canvia la resolució de la pantalla. En canvi, una distància definida en *em* o en percentatge és proporcional a la mida de la font establerta pel navegador. Per tant, si l'usuari canvia la mida del text en el seu navegador, la mesura augmenta o disminueix proporcionalment.

Algunes propietats tenen com a valor un **color** (color de la lletra, color de fons...). Els colors es poden expressar amb paraules clau (`red` (vermell), `green` (verd), `blue` (blau)...) o amb el model RGB.

Si usem les paraules clau estem limitats a usar un cert nombre de colors, els que tenen associat un nom. En canvi, amb la fórmula RGB podem obtenir tota la gamma de colors. Per expressar un color amb el format RGB, tenim dues maneres de fer-ho: per exemple, si volem donar el color vermell, hem d'escriure el següent: `#ff0000` o bé `rgb(255, 0, 0)`.

#### 3.6.2. Fonts

En la taula 9 podem veure les diferents propietats que ens ajuden a modificar la font de la lletra.



Podeu trobar unes guies de referència ràpida de totes les propietats de CSS (tant per a CSS1 com per a CSS2) en la secció "Annexos" del web del crèdit.



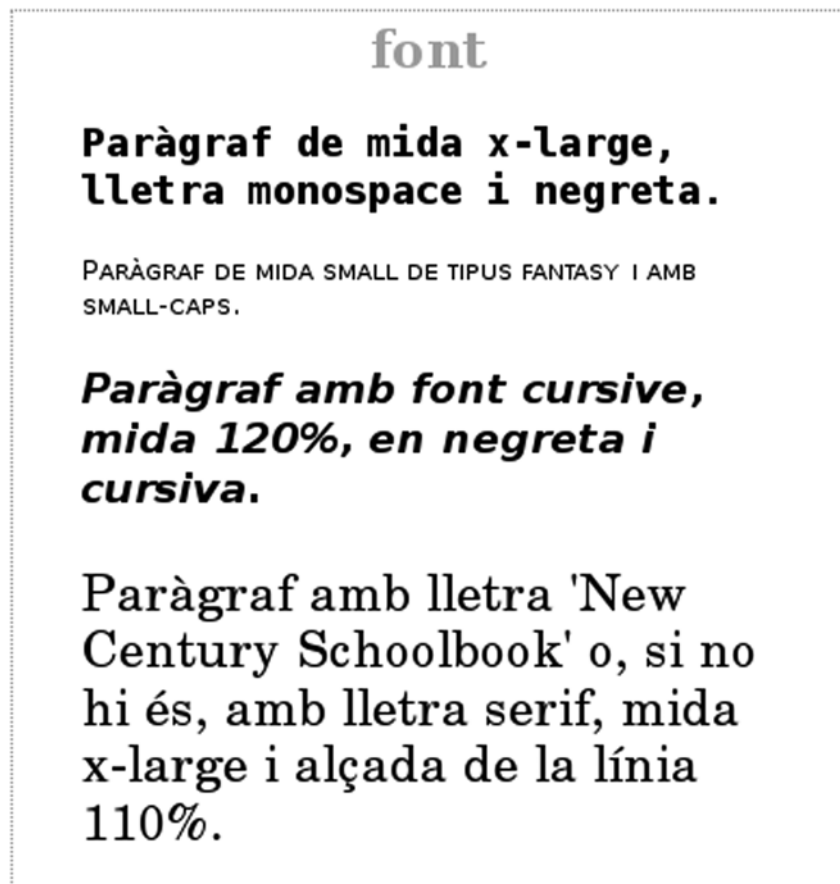
Podeu trobar totes les paraules clau amb el seu color corresponent en la secció "Annexos" del web d'aquest crèdit.

Taula 9. Propietats CSS per a la font de la lletra

Propietat	Descripció	Exemples
font-family	Canvia el tipus de lletra. Se'n poden posar diversos, separats per coma, per si el navegador no disposés d'algun d'ells.	<pre>body {   font-family: Gill, Helvetica, sans-serif; }</pre>
font-style	Serveix per posar ( <i>italic</i> ) o treure ( <i>normal</i> ) la lletra en cursiva.	<pre>h1, h2, h3 { font-style: italic; } h1 em { font-style: normal; }</pre>
font-variant	Posa ( <i>small-caps</i> ) o treu ( <i>normal</i> ) les lletres minúscules en majúscules però amb una tipografia més petita.	<pre>h3 { font-variant: small-caps }</pre>
font-weight	Serveix per posar ( <b>bold</b> ) o treure ( <i>normal</i> ) la lletra en negreta.	<pre>p { font-weight: normal } h1 { font-weight: bold }</pre>
font-size	Defineix la mida de la lletra.	<pre>p { font-size: 16px; } @media print {   p { font-size: 12pt; } }</pre>
font	Propietat que resumeix les propietats anteriors.	<pre>p { font: 12px sans-serif } p { font: x-large Helvetica, serif } p { font: bold italic large serif } p { font: normal small-caps 120% fantasy }</pre>
line-height	Estableix l'alçada de cadascuna de les línies de l'element.	<pre>p { line-height: 2em; }</pre>

La figura 31 mostra més exemples d'aquestes propietats.

Figura 31. Propietats de font



Trobareu el codi dels exemples en la secció "Recursos de contingut" del web d'aquest crèdit.

### 3.6.3. Aspecte del text

En la taula 10 podem veure les diferents propietats que ens ajuden a modificar l'aspecte del text.

Taula 10. Propietats CSS per a la font de la lletra

Propietat	Descripció	Exemples
text-indent	Defineix el sagnat, pot ser una unitat positiva o negativa.	<code>p { text-indent: 3em; }</code>
text-align	Defineix el tipus d'alienació.	<code>div.important { text-align: center; }</code>
text-decoration	Permet posar una línia a sobre (underline), a sota (overline) o taltant el text (line-through).	<code>.destaca { text-decoration: underline; }</code>
letter-spacing	Incrementa la distància entre lletres.	<code>blockquote { letter-spacing: 0.1em; }</code>
word-spacing	Incrementa la distància entre paraules.	<code>h1 { word-spacing: 1em; }</code>
text-transform	Escriu les lletres en majúscules (uppercase), minúscules (lowercase) o la primera lletra de cada paraula en majúscula (capitalize) o anul·la la propietat (none).	<code>h1 { text-transform: uppercase; }</code> <code>.titol { text-transform: capitalize; }</code>
white-space	Defineix com s'han de tractar els espais en blanc que contingui l'element. El valor perquè tingui en compte els espais en blanc és pre.	<code>pre { white-space: pre; }</code> <code>p { white-space: normal; }</code>

La figura 32 mostra més exemples d'aquestes propietats.

Figura 32. Propietats text-indent i text-decoration

#### text-indent

Paràgraf sense indentació.

Paràgraf amb indentació positiva (1em), la segona línia apareix amb indentació normal.

Paràgraf amb indentació negativa (-1em), la segona línia apareix amb indentació normal.

#### text-decoration


Paràgraf sense decoració.

Paràgraf amb subratllat.

Paràgraf amb superratllat.

~~Paràgraf taltat.~~

Paràgraf parpallejant.



Recordeu que trobareu el codi dels exemples en la secció "Recursos de contingut" del web d'aquest crèdit.

### 3.6.4. Color i fons

En la taula 11 podem veure les diferents propietats que ens ajuden a modificar el color del text i el seu color de fons.

Taula 11. Propietats CSS per al color i el fons dels elements

Propietat	Descripció	Exemples
color	Estableix el color de la lletra.	<code>em { color: red; }</code> <code>em { color: rgb(255,0,0); }</code>
background-color	Estableix el color de fons.	<code>h1 { background-color: #FF0000; }</code>
background-image	Treu (none) la imatge de fons.	<code>body{background-image: url("fons.png");}</code> <code>p { background-image: none; }</code>

Propietat	Descripció	Exemples
background-attachment	Estableix la fixació de la imatge de fons. Pot ser fixa ( <i>fixed</i> ) o movable ( <i>scroll</i> ).	<pre>body {   background-image: url("fons.png");   background-attachment: fixed; }</pre>
background-repeat	Especifica com volem que la imatge de fons es repeteixi ( <i>repeat</i> , <i>repeat-x</i> , <i>repeat-y</i> ) o si no volem que es repeteixi ( <i>no-repeat</i> ).	<pre>body {   background: white url("fons.png");   background-repeat: repeat-y; }</pre>
background-position	Especifica la posició de la imatge de fons. El primer valor correspon a l'eix X ( <i>left</i> , <i>center</i> o <i>right</i> ) i el segon a l'eix Y ( <i>top</i> , <i>center</i> o <i>bottom</i> ). També podem introduir percentatges.	<pre>body{   background-image:url("fons.png");   background-position: right top; }</pre>
background	Propietat que resumeix les propietats anteriors.	<pre>body { background: red; } p{ background: gray url("fons.png")   repeat fixed left top; }</pre>

La figura 33 mostra més exemples d'aquestes propietats.

Figura 33. Propietats de *background*



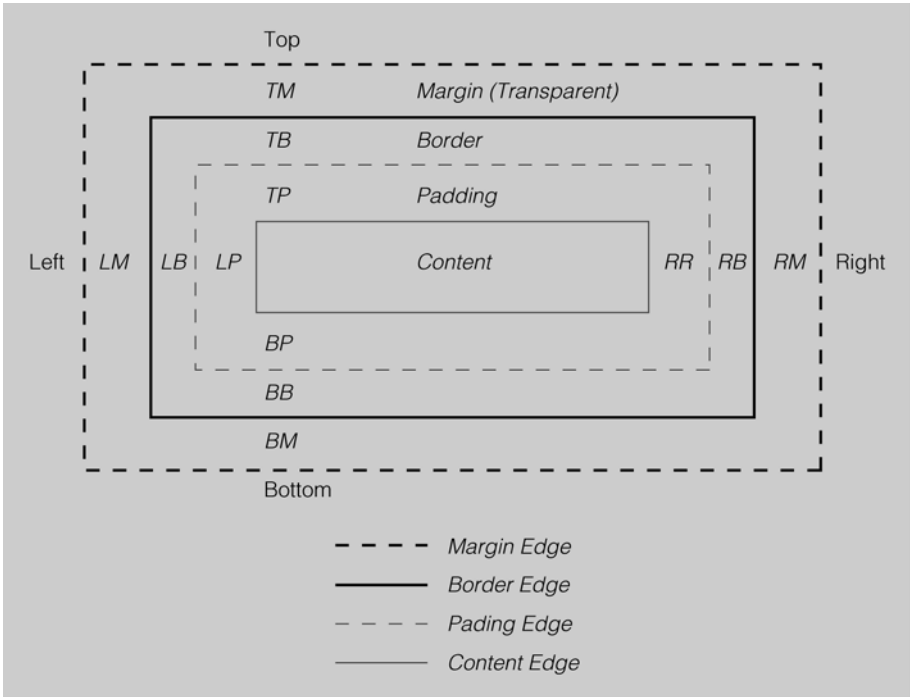
### 3.7. El model de caixa

Hi ha diverses propietats, com els marges o les vores, que hem d'establir a partir de l'anomenat **model de caixa**. Per entendre aquest model hem de pensar cada element del document (X)HTML com si fos una caixa. De cadascuna d'aquestes caixes podem distingir els elements següents:

- El contingut (*content*) de l'element: text o altres elements.
- El farciment (*padding*): espai entre el contingut i la vora de la caixa.
- La vora (*border*): vora que delimita la caixa.
- El marge (*margin*): espai entre la vora de la caixa i la resta d'elements.

A més, cada caixa té quatre segments: el superior (*top*), l'inferior (*bottom*), l'esquerre (*left*) i el dret (*right*). Podem observar tots aquests elements en la figura 34.

Figura 34. El model de caixa



La taula 12 ens ofereix les propietats que ens permeten definir les mesures de tots aquests elements.

Taula 12. Propietats del model de caixa

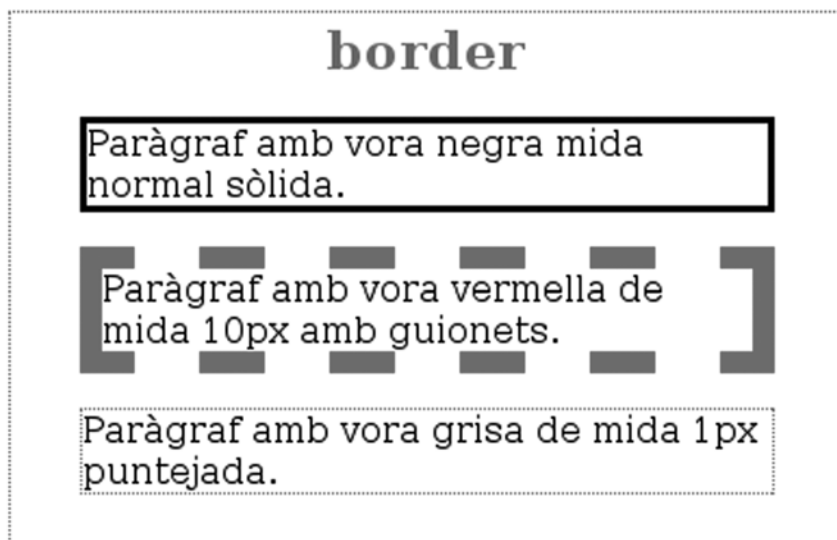
Propietat	Descripció	Descripció
<code>margin-top</code> <code>margin-right</code> <code>margin-bottom</code> <code>margin-left</code>	Mesura cadascun dels marges de la caixa (superior, dret, inferior i esquerre).	<pre>body {   margin-top: 1em;   margin-right: 2em;   margin-bottom: 3em;   margin-left: 4em; }</pre>
<code>margin</code>	Propietat que resumeix les propietats <code>margin-*</code> : es poden introduir d'un a quatre valors separats per espai: 1 valor: afecta els quatre marges 2 valors: <code>top-bottom left-right</code> 3 valors: <code>top right-left bottom</code> 4 valors: <code>top right bottom left</code>	<pre>body { margin: 1em 2em 3em 4em; }</pre>
<code>padding-top</code> <code>padding-right</code> <code>padding-bottom</code> <code>padding-left</code>	Mesura cadascun dels farciments: superior, dret, inferior i esquerre.	<pre>body {   padding-top: 1em;   padding -right: 2em;   padding -bottom: 1em;   padding-left: 2em; }</pre>
<code>padding</code>	Propietat que resumeix les propietats <code>padding-*</code> . Els valors s'introdueixen igual que a la propietat <code>margin</code> .	<pre>h1 { padding: 1em 2em; }</pre>
<code>border-top-width</code> <code>border-right-width</code> <code>border-bottom-width</code> <code>border-left-width</code>	Especifica l'amplada de cadascuna de les vores superior, dreta, inferior i esquerra. Com a valor podem posar <i>thin</i> (prim), <i>mitjà</i> (medium), <i>gruixut</i> (thick) o una mesura.	<pre>h1 {   border-top-width: thin;   border-right-width: thick;   border-botom-width: medium;   border-left-width: thick; }</pre>



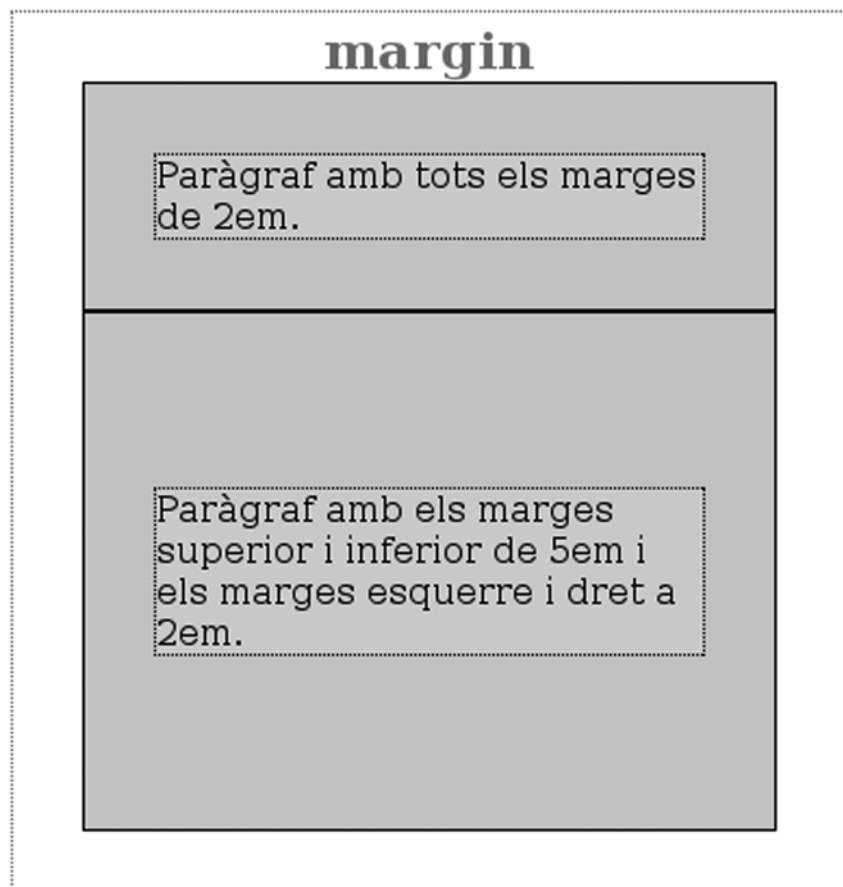
Propietat	Descripció	Descripció
<code>border-width</code>	Propietat que resumeix les propietats <code>border-*-width</code> . Els valors s'introdueixen igual que a la propietat <code>margin</code> .	<code>h1 {border-width: thin thick medium;}</code>
<code>border-top-color</code> <code>border-right-color</code> <code>border-bottom-color</code> <code>border-left-color</code>	Especifica el color de cadascuna de les vores superior, dreta, inferior i esquerra.	<pre>p {   border-top-color: black;   border-right-color: red;   border-bottom-color: green;   border-left-color: blue; }</pre>
<code>border-color</code>	Propietat que resumeix les propietats <code>border-*-color</code> . Els valors s'introdueixen igual que a la propietat <code>margin</code> .	<code>p { border-color: gray; }</code>
<code>border-top-style</code> <code>border-right-style</code> <code>border-bottom-style</code> <code>border-left-style</code>	Defineix l'estil de cadascuna de les vores superior, dreta, inferior i esquerra. Podrem introduir els valors següents: <code>none</code> o <code>hidden</code> : cap vora <code>dotted</code> : punts <code>dashed</code> : segments <code>solid</code> : línia contínua <code>double</code> : dues línies contínues <code>groove</code> : enfonsat <code>ridge</code> : sortit <code>inset</code> : encastat <code>outset</code> : sobresortit	<pre>#caixa {   border-top-style: solid;   border-right-style: dotted;   border-bottom-style: solid;   border-left-style: dotted; }</pre>
<code>border-style</code>	Propietat que resumeix les propietats <code>border-*-style</code> . Els valors s'introdueixen igual que a la propietat <code>margin</code> .	<code>#caixa { border-style: solid dotted }</code>
<code>border-top</code> <code>border-right</code> <code>border-bottom</code> <code>border-left</code>	Propietats que resumeixen les propietats <code>border</code> per posició.	<pre>h1 {   border-top: thick solid red;   border-right: thick solid red;   border-bottom: thick solid red;   border-left: thick solid red; }</pre>
<code>border</code>	Propietat que resumeix les propietats <code>border</code> . Afecten les quatre posicions igual.	<code>h1{ border: thick solid red; }</code>

La figura 35 i la figura 36 mostren més exemples d'aquestes propietats.

Figura 35. Propietats de `border`



Recordeu que trobareu el codi dels exemples en la secció "Recursos de contingut" del web d'aquest crèdit.

Figura 36. Propietats de `margin`

### 3.8. El format visual dels elements d'un document

El format visual defineix com són les caixes (és a dir, els elements (X)HTML vistos com a caixes) en el document i el comportament d'aquestes caixes envers la resta de caixes de la pàgina.

En la taula 13 es mostren les propietats que ens permeten modificar el format visual de cadascun dels elements del nostre document.

Taula 13. Propietats per a modificar el format visual dels elements d'un document

Propietat	Descripció	Exemples
<code>display</code>	<p>Canvia el tipus de visualització de l'element. Els valors que pot prendre són els següents:</p> <ul style="list-style-type: none"> <li><code>block</code>: element de bloc.</li> <li><code>inline</code>: element de línia.</li> <li><code>list-item</code>: element de llista.</li> <li><code>marker</code>: aquest valor declara que el contingut generat davant o després d'una caixa és un marcador.</li> <li><code>none</code>: l'element amb aquesta propietat no genera cap caixa, no és visible ni ocupa espai en el document.</li> <li><code>run-in i compact</code>: creen caixes de bloc o de línia segons el context.</li> <li><code>table, inline-table, table-row-group, table-column, table-column-group, table-header-group, table-footer-group, table-row, table-cell i table-caption</code>: l'element es comporta com una taula.</li> </ul>	<pre>/* Per defecte: */ p { display: block; } em { display: inline; } li { display: list-item; } /* Si no volem mostrar imatges: */ img { display: none; }</pre>

Propietat	Descripció	Exemples
position	Describeu la posició envers la resta d'elements. Els valors poden ser els següents: relative: guarda l'espai i mou el que li diguem. absolute: no guarda l'espai. fixed: com absolute, però sempre està allà encara que ens desplacem amb la pàgina.	hl#primer { position: fixed }
top right bottom left	Indica a quina posició posem la caixa dins la pàgina.	#header { top: 0; right: 0; bottom: auto; left: 0; }
width height	Indica la mida de la caixa: amplada i alçada.	#peu { position: fixed; width: 100%; height: 100px; top: auto; right: 0; bottom: 0; left: 0; }
min-width min-height max-width max-height	Assenyala les dimensions mínimes i màximes que poden prendre les caixes, sigui quina sigui la mida de la finestra: amplada mínima, alçada mínima, amplada màxima i alçada màxima.	
float	Fa que la caixa quedi flotant a l'esquerra (left) o a la dreta (right).	img { float: right; }
clear	Serveix per desactivar elements flotants al voltant de l'element que té la propietat. Podem eliminar els elements flotants a la part esquerra (left), a la part dreta (right), a ambdós costats (both) i anul·lar la propietat (none).	#peu { clear: both; }
z-index	Marca la profunditat d'una caixa. Com més petit sigui el valor més al fons se situarà la caixa.	p.fons { z-index: 1; } p.primer { z-index: 2; }
overflow	Indica què fer quan el contingut mesura més que la caixa. Els possibles valors són: visible: el contingut es veu sobrepassant la taula. hidden: el contingut sobrant no es veu. scroll: apareix una barra de desplaçament. auto: apareix una barra de desplaçament, si és necessari.	div.desplaca { height: 10em; overflow: auto; }

La figura 37 i la figura 38 mostren més exemples d'aquestes propietats.

Figura 37. Exemple de les propietats de posicionament absolut

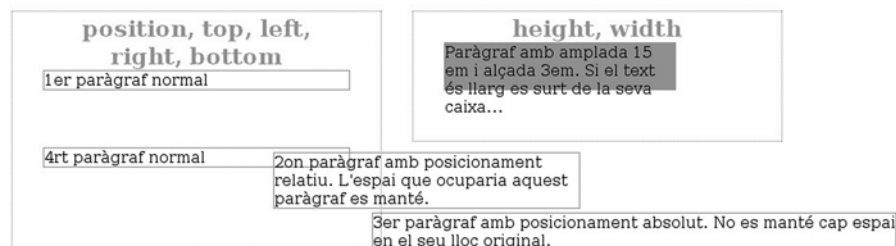
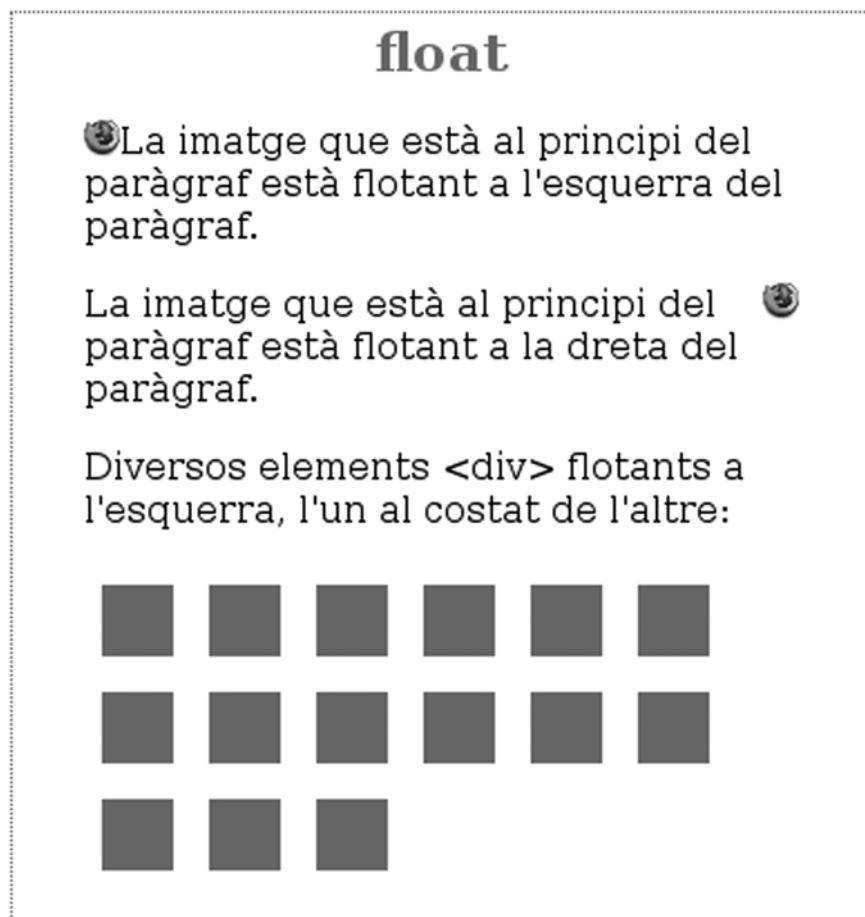


Figura 38. Exemple de la propietat `float`

Recordeu que trobareu el codi dels exemples en la secció "Recursos de contingut" del web d'aquest crèdit.

### 3.9. Exemples pràctics amb CSS

L'art del CSS no és saber-se totes les propietats que hi ha sinó saber-les combinar adequadament per aconseguir els efectes desitjats. Hi ha centenars de tècniques per a problemes comuns.

Casos típics on és necessari dominar les propietats CSS són fer menús a partir de llistes o maquetar webs sense usar taules.

#### 3.9.1. Fer menús amb llistes

Gairebé tota web disposa d'un menú de navegació que ens permet anar enllaçant a les diferents seccions del lloc. Una manera molt elegant de fer aquests menús és pensar-los com una llista amb ítems (i subítems, si s'escau). El problema que tenim és que les llistes són "lletges".

Vegem com podem fer un menú "bonic" a partir d'una llista. Es tracta d'un menú vertical que té l'aspecte que mostra la figura 39.

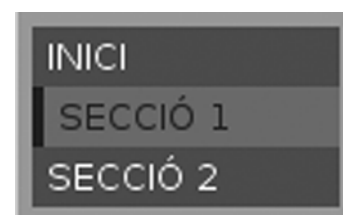
#### Més propietats

L'especificació CSS2 estableix encara més propietats que les que hem presentat. Les podeu consultar directament a l'especificació que trobareu en el lloc web del W3C.



En la secció "Recursos de contingut" del web del crèdit trobareu més exemples de menús fets amb llistes.

Figura 39. Menú fet amb una llista



El codi (X)HTML és molt simple. Es tracta simplement d'una llista desordenada:

---

```
<div id="menu">
  <ul>
    <li><a href="">Inici</a></li>
    <li><a href="">Secció 1</a></li>
    <li><a href="">Secció 2</a></li>
  </ul>
</div>
```

---

Vegem quines regles CSS hem d'introduir per obtenir els resultats desitjats.

A continuació, combinarem diverses de les regles CSS existents, i així aconseguirem que aquesta simple llista desordenada acabi essent un menú atractiu i accessible per a l'usuari.

Primer de tot hem de dir a la llista que no volem pics a cada ítem, això ho podem aconseguir aplicant una propietat específica de llistes que es diu `list-style-type`. A més, hi traiem marges i farciments i hi posem una vora:

---

```
#menu ul {
  list-style-type: none ;
  margin: 0;
  padding: 0;
  border: thick solid #D68EFF;
}
```

---

A cada ítem de la llista hi canviem el color de fons:

---

```
#menu ul li {
  background-color: #B20298;
}
```

---

Ara, definim l'estil dels enllaços: canviem el color de la lletra, traiem el subratllat, fem que les lletres siguin sempre majúscules i hi posem una mica de farciment. Hem de destacar que la declaració `display: block` serveix perquè l'enllaç passi a ser un element de bloc (per defecte, és un element de línia) i així aconseguim poder prémer l'enllaç en tot el bloc, no només on hi ha les lletres de l'enllaç:

---

```
#menu ul li a {
  color: white ;
  text-decoration: none ;
  text-transform: uppercase ;
  display: block ;
  padding: 0.2em 0.2em 0.2em 0.4em ;
}
```

---

Finalment, definim quina estètica han de tenir els enllaços quan hi passem per damunt. En aquest cas, els canviem el color de lletra i de fons, i hi fem aparèixer una vora a l'esquerra:

```
#menu ul li a:hover {  
    background: #CC4ABD;  
    border-left: 0.3em solid #58014B;  
    color: #58014B;  
}
```

### 3.9.2. Maquetació de webs sense ús de taules

Podem trobar en molts llocs web que la maquetatció està feta a partir de taules. L'ús de taules per maquetar és feixuc i hem d'evitar-lo. En aquest exemple mostrem com podem fer una maquetatció sense taules (*tableless*) fent ús únicament de la marca `<div>` i de CSS.

La maquetatció que ens proposem fer és la d'un lloc web típic, amb una capçalera, una barra lateral per al menú, el cos de la pàgina amb el contingut del lloc i un peu de pàgina. El resultat desitjat el podeu observar en la figura 40.

Figura 40. Maquetació *tableless*



El codi (X)HTML és força senzill, simplement marquem amb marques `<div>` cadascuna de les capes que formen el web. Englobem tot el contingut en un `<div>` contenidor, en què posem la resta de capes (cap, lateral amb el menú, cos i peu):

```
<body>  
<div id="contenidor">  
    <div id="cap">  
        <h1>Títol de la web</h1>
```

```
<h2>Subtítol de la web</h2>
</div>
<div id="lateral">
  <div id="menu">
    <ul>
      <li><a href="">Inici</a></li>
      <li><a href="">Secció 1</a></li>
      <li><a href="">Secció 2</a></li>
    </ul>
  </div>
</div>
<div id="cos">
  <div id="inici">
    <h3>Inici</h3>
    <p>..</p>
  </div>
</div>
<div id="peu">
  <p>Peu de pàgina </p>
</div>
</body>
```

---

Ara definim les regles pertinents per aconseguir els nostres objectius.

En la marca `<body>` definim totes les propietats genèriques: color de fons, color de lletra i tipus de lletra. A més, posem el text centrat (perquè Internet Explorer entengui que volem el contingut centrat) i traiem qual-sevol tipus de marge.

---

```
body {
  text-align: center ;
  background-color:#FFC7FC;
  color: #58014B;
  font: small sans-serif;
  margin: 0;
}
```

---

Al `<div>` amb identificador `contenedor` hem de dir-li que no ocupi tota la pàgina (li diem que n'ocupi només un 80%), que quedi centrat (això s'aconsegueix posant els marges esquerre i dret amb valor `auto`) i que el fons sigui blanc. A més, fem que tot el text del web estigui alineat a l'esquerra.

---

```
#contenedor{
  width: 80%;
  margin: 0px auto;
  text-align: left;
  background-color:white;
}
```

---

La capçalera i el peu de pàgina tenen propietats comunes: tenen la mateixa imatge de fons, el color de lletra és blanc i en negreta, i el text està centrat.

---

```
#cap, #peu{
  background-image: url("imatges/banner-web.png");
  text-align:center;
  font-weight: bold;
  color:white;
}
```

---

Especifiquem també les propietats que només afecten la capçalera: volem que els títols que contingui no tinguin marges:

```
#cap h1, #cap h2{ margin:0; }
```

Posem, a més, les mides per defecte que han de tenir tots els encapçalaments <h1> i <h2> de la pàgina:

```
h1{ font-size: 5em; }  
h2{ font-size: 4em; }
```

Per al peu de pàgina també tenim propietats específiques: la lletra ha de ser petita i fem que no pugui tenir cap element flotant. Així, la barra lateral, que és un element flotant, mai no es posarà per damunt del peu de pàgina.

```
#peu {  
    font-size: small;  
    clear: both ;  
}
```

Ara definim la barra lateral: és un element flotant a l'esquerra i té una amplada del 15% de la pàgina:

```
#lateral {  
    width: 15% ;  
    float: left ;  
}
```

Per maquetar el cos del web diem que el marge esquerre és d'un 15%, que és just l'espai que ocupa la barra lateral. A més, posem una mica de farciment a l'esquerra, perquè el text del cos no quedi enganxat al menú.

```
#cos {  
    margin-left: 15% ;  
    padding-left: 1em ;  
}
```

Amb totes aquestes regles hem aconseguit el nostre objectiu: tenim una web bonica, ben estructurada, amb un codi (X)HTML senzill i sense fer ús de taules. (Només ens manca el codi per convertir la llista que conforma el menú en l'estètica proposada.)



Recordeu que trobareu el codi dels exemples en la secció "Recursos de contingut" del web d'aquest crèdit.