

Mòdul professional: Llenguatges de Marques i Sistemes de Gestió d'Informació



Codi: 0373

Total: 96 hores

Continguts:

Tema 1. Reconeixement de les característiques de llenguatges de marques

Tema 2. Utilització de llenguatges de marques en entorns web: HTML5

Tema 3. Utilització de llenguatges de marques en entorns web: CSS3

Tema 4. JavaScript HTML DOM

Tema 5. Definició d'esquemes i vocabularis en llenguatges de marques

Tema 6. Conversió i adaptació de documents per a l'intercanvi d'informació

Tema 7. Emmagatzematge d'informació

Tema 8. Sistemes de gestió empresarial

Xavi Blanes

Tema 1. Reconeixement de les característiques de llenguatges de marques



1. [Concepte de llenguatge de marques.](#)
 2. [Avantatges. Necessitat d'ús.](#)
 3. [Característiques comunes.](#)
 4. [Classificació.](#)
 5. [Què és XML?](#)
 6. [Com es pot utilitzar XML?](#)
 7. [Arbre XML](#)
 8. [Sintaxi XML](#)
 9. [Elements XML](#)
 10. [Atributs XML](#)
 11. [Espais de noms XML](#)
-

1 Concepte de llenguatge de marques.

És un sistema de codificació que s'utilitza per representar informació de manera estructurada en un document de text. En comptes d'utilitzar un llenguatge natural com l'espanyol o l'anglès, es fa servir un conjunt d'etiquetes o marques que indiquen com s'ha de formar o estructurar la informació dins del document.

Aquestes etiquetes són interpretades per programes o aplicacions específiques per renderitzar el contingut segons les instruccions proporcionades. Un dels llenguatges de marques més coneguts és HTML (HyperText Markup Language), que es fa servir per crear pàgines web. En HTML, les etiquetes s'utilitzen per definir elements com ara capçaleres, paràgrafs, enllaços, imatges i altres elements d'una pàgina web. Altres exemples de llenguatges de marques inclouen XML (eXtensible Markup Language), que s'utilitza per a representar dades estructurades, i Markdown, que es fa servir per formatar text de manera simple i ràpida, especialment en entorns com blocs i wikis.

Els llenguatges de marques són importants perquè permeten la creació de documents estructurats que poden ser interpretats i presentats de manera coherent a diferents plataformes i dispositius. A més, faciliten la separació de contingut i presentació, cosa que significa que el contingut d'un document pot ser formatat de manera diferent sense necessitat de canviar-ne l'estructura subjacent. Això és fonamental en la creació de contingut web, on es busca que les pàgines es vegean bé en una varietat de dispositius i mides de pantalla.

2 Avantatges. Necessitat d'ús.

L'ús de llenguatges de marques proporciona diversos avantatges en la representació i la gestió d'informació estructurada. Alguns dels principals avantatges inclouen:

1- Estructura i organització: Els llenguatges de marques permeten organitzar i estructurar la informació de manera coherent i jeràrquica. Això facilita la comprensió i el processament de les dades.

2- Separació de contingut i presentació: Un dels beneficis clau és la separació de contingut i presentació. Això significa que el contingut s'emmagatzema en un format estructurat, mentre que la presentació (com el format, l'estil i el disseny) el tenim per separat. Això facilita l'adaptació del contingut a diferents contextos i dispositius sense necessitat de modificar l'estructura subjacent.

3- Interoperabilitat: Els llenguatges de marques solen ser estàndards oberts i àmpliament acceptats. Això promou la interoperabilitat, la qual cosa significa que les dades i els documents creats en un llenguatge de marques específic poden ser interpretats i processats per una varietat d'aplicacions i sistemes.

4- Portabilitat: Els documents creats amb llenguatges de marques són generalment portables entre diferents plataformes i sistemes operatius. Això garanteix que la informació es puga compartir i visualitzar de manera consistent en una àmplia gamma de dispositius i entorns.

5- Eficiència en l'emmagatzematge: Els llenguatges de marques solen ser més eficients pel que fa a l'espai demmagatzematge en comparació amb formats de dades binàries.

Això és especialment rellevant en aplicacions on cal emmagatzemar grans quantitats d'informació de manera eficient.

6- Facilita l'automatització: L'estrucció predefinida dels llenguatges de marques facilita l'automatització de tasques relacionades amb la gestió i el processament de dades. Els programes informàtics poden analitzar i manipular documents de llenguatge de marques de manera més eficient que els documents en format de text sense estructurar.

7- Facilita l'accessibilitat: Els llenguatges de marques es poden enriquir amb metadades que milloren l'accessibilitat per a persones amb discapacitats. Per exemple, HTML permet etiquetar elements multimèdia perquè els lectors de pantalla puguen interpretar-los i descripcions per a imatges.

8- Flexibilitat: La flexibilitat dels llenguatges de marques permet adaptar la representació de dades a una varietat de necessitats i contextos. Això és especialment útil en la creació de contingut web, on l'adaptació a diferents dispositius i pantalles és essencial.

En resum, l'ús de llenguatges de marques proporciona una sèrie d'avantatges clau que faciliten la representació, la gestió i la presentació d'informació estructurada en una àmplia gamma d'aplicacions i entorns.

L'ús de llenguatges de marques és necessari en una gran varietat de situacions a causa dels avantatges que ofereixen en la representació i la gestió d'informació estructurada. Ací es presenten algunes de les necessitats principals que justifiquen l'ús de llenguatges de marques:

1- Web i desenvolupament de llocs: Els llenguatges de marques com ara HTML, CSS i JavaScript són essencials per crear i dissenyar llocs web. HTML s'utilitza per estructurar el contingut, CSS per al disseny i l'estil, i JavaScript per a la interactivitat. Aquests llenguatges permeten crear pàgines web amb una aparença i funcionalitat específiques.

2- Intercanvi de dades estructurades: En moltes aplicacions, cal intercanviar dades estructurades entre diferents sistemes i plataformes. Llenguatges de marques com XML i JSON s'utilitzen per representar dades de manera estructurada i facilitar la comunicació entre sistemes heterogenis.

3- Documentació tècnica: Per documentar protocols, especificacions tècniques, estàndards o qualsevol tipus d'informació estructurada, els llenguatges de marques com XML o Markdown són valuosos. Permeten representar de manera clara i organitzada informació tècnica complexa.

4- Publicació en línia: Tant per a blocs com per a wikis, els llenguatges de marques com Markdown són populars a causa de la seva simplicitat i llegibilitat. Faciliten la publicació ràpida de contingut en línia sense necessitat de coneixements tècnics avançats.

5- Presentació d'informes i documents tècnics: En entorns empresarials i acadèmics, s'utilitzen llenguatges de marques com ara LaTeX per crear documents tècnics i científics. Aquests llenguatges ofereixen un alt grau de control sobre el format i estructura del document.

6- Automatització de processos: Els llenguatges de marques són fonamentals en l'automatització de tasques, ja que les dades estructurades són més fàcils de processar

automàticament mitjançant scripts i programes. Això s'aplica a l'automatització d'informes, processament de dades i més.

7- Accessibilitat i estàndards: Per garantir l'accessibilitat en línia i el compliment d'estàndards web, com ara WCAG (Web Content Accessibility Guidelines), cal utilitzar llenguatges de marques per etiquetar i estructurar adequadament el contingut web.

En resum, la necessitat de fer servir llenguatges de marques sorgeix en situacions on és fonamental representar informació de manera estructurada, facilitar la comunicació entre sistemes, aconseguir la interoperabilitat, automatitzar tasques i complir estàndards específics. Aquests llenguatges tenen un paper essencial en la informàtica, la comunicació en línia i la gestió de dades.

3 Característiques comunes.

Els llenguatges de marques comparteixen algunes característiques comunes que els distingeixen dels llenguatges de programació convencionals i els fan adequats per a la representació estructurada de dades i documents. Aquestes característiques inclouen:

1-ús d'etiquetes o marques: Els llenguatges de marques utilitzen etiquetes o marques per definir l'estructura i el format del contingut. Aquestes etiquetes solen estar delimitades per símbols com <> en HTML o {} a LaTeX.

2- Sintaxi predefinida: Els llenguatges de marques tenen una sintaxi predefinida que dicta com s'han d'utilitzar les etiquetes i com han d'estar estructurats els documents. Això proporciona coherència i consistència en la representació de dades.

3- Estructura jeràrquica: Els documents creats amb llenguatges de marques solen tenir una estructura jeràrquica on els elements estan els uns dins d'altres, creant una gerarquia. Això permet representar informació complexa de manera organitzada.

4- Text llegible: A diferència dels llenguatges de programació, els llenguatges de marques solen ser llegibles per humans. Això facilita la creació i l'edició de documents utilitzant un editor de text estàndard.

5- Separació de contingut i presentació: Els llenguatges de marques promouen la separació de contingut i presentació. El contingut es defineix al document utilitzant etiquetes, mentre que la presentació es controla per separat, cosa que facilita l'adaptació a diferents estils i dispositius.

6- Estàndards oberts: Molts llenguatges de marques són estàndards oberts i àmpliament acceptats. Això facilita la interoperabilitat i l'intercanvi de dades entre diferents sistemes i aplicacions.

7- Portabilitat: Els documents creats amb llenguatges de marques solen ser portables entre diferents sistemes operatius i plataformes, cosa que garanteix que la informació es puga compartir i visualitzar de manera consistent.

8- Flexibilitat: Els llenguatges de marques són flexibles i es poden adaptar a una varietat de necessitats i contextos. Poden utilitzar-se per representar des de contingut web fins a dades estructurades o documents tècnics.

9- Enriquiment semàntic: Alguns llenguatges de marques permeten l'enriquiment semàntic de la informació mitjançant l'ús d'atributs, metadades i altres elements que afegeixen significat addicional a les dades.

10- Faciliten l'automatització: A causa de la seva estructura predefinida i llegibilitat, els llenguatges de marques són adequats per a l'automatització de tasques relacionades amb el processament de dades, cosa que els fa útils en entorns de programació i sistemes de gestió de dades.

Aquestes característiques comunes fan que els llenguatges de marques siguin eines poderoses per representar informació estructurada de manera coherent i efectiva en una gran varietat de contextos.

4 Classificació.

A continuació, et presento una classificació general dels llenguatges de marques:

1. Llenguatges de marques de marcatge lleuger (Lightweight Markup Languages): Aquests llenguatges estan dissenyats per ser simples i fàcils de llegir i escriure, amb una sintaxi minimalista. Alguns exemples de llenguatges de marcatge lleuger inclouen Markdown i reStructuredText. S'utilitzen comunament en la creació de documents de text enriquit i per prendre notes.

2. Llenguatges de marques de marcatge extensible (Extensible Markup Languages - XML): XML és un llenguatge de marques que permet crear etiquetes personalitzades per descriure dades estructurades. S'utilitza en una àmplia varietat d'aplicacions, com la representació de dades a la web (per exemple, a RSS i Atom), a la configuració de fitxers (per exemple, a fitxers de configuració de programari) i a la representació de dades a bases de dades.

3. Llenguatges de marques d'hipertext (Hypertext Markup Languages – HTML): HTML és el llenguatge de marques utilitzat per crear pàgines web. Defineix l'estructura i el format dels elements en una pàgina web, com ara capçaleres, paràgrafs, imatges, enllaços i més. HTML s'utilitza juntament amb CSS (Cascading Style Sheets) per donar estil i disseny a les pàgines web.

4. Llenguatges de marques de presentació (Presentation Markup Languages): Aquests llenguatges s'utilitzen per definir la presentació i el disseny de documents. Un exemple és CSS, que s'utilitza per a controlar l'aparença de les pàgines web en definir estils, colors, mides de font i altres atributs visuals.

5. Llenguatges de marques de dades (Data Markup Languages): Aquests llenguatges es fan servir per descriure dades estructurades i el seu significat. Un exemple és JSON (JavaScript Object Notation), que s'utilitza molt per a l'intercanvi de dades en aplicacions web i serveis web.

6. Llenguatges de marques específics del domini (Domain-Specific Markup Languages): Aquests llenguatges estan dissenyats per a un propòsit o indústria específics. Per exemple, MathML s'utilitza per representar fórmules matemàtiques a documents web i científics.

7. Llenguatges de marques d'anotació semàntica (Semantic Markup Languages): Aquests llenguatges s'utilitzen per afegir informació semàntica als elements en un document, cosa que permet a les màquines comprendre millor el significat del contingut. RDF

(Resource Description Framework) i Microdata són exemples de llenguatges de marques d'anotació semàntica.

Aquestes són algunes de les categories més comunes de llenguatges de marques. Cadascú té el seu propi propòsit i aplicació en diferents contextos tecnològics i de comunicació.

5. Què és XML?

XML significa eXtensible Markup Language.

XML va ser dissenyat per emmagatzemar i transportar dades.

XML va ser dissenyat per ser llegible tant per l'home com per la màquina.

XML Exemple 1

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>
```

XML Exemple 2

```
<?xml version="1.0" encoding="UTF-8"?>
<breakfast_menu>
    <food>
        <name>Belgian Waffles</name>
        <price>$5.95</price>
        <description>
            Two of our famous Belgian Waffles with plenty of real maple
            syrup
        </description>
        <calories>650</calories>
    </food>
    <food>
        <name>Strawberry Belgian Waffles</name>
        <price>$7.95</price>
        <description>
            Light Belgian waffles covered with strawberries and whipped
            cream
        </description>
        <calories>900</calories>
    </food>
    <food>
        <name>Berry-Berry Belgian Waffles</name>
        <price>$8.95</price>
        <description>
            Belgian waffles covered with assorted fresh berries and whipped
            cream
        </description>
        <calories>900</calories>
    </food>
</breakfast_menu>
```

Per què estudiar XML?

XML té un paper important en molts sistemes informàtics diferents.

XML s'utilitza sovint per distribuir dades per Internet.

És important (per a tot tipus de desenvolupadors de programari) tindre una bona comprensió d'XML.

Què Aprendràs

- Què és XML?
- Com funciona XML?
- Com puc utilitzar XML?
- Per a què puc utilitzar XML?

Què és XML?

- XML significa eXtensible Markup Language
- XML és un llenguatge de marques semblant a HTML
- XML va ser dissenyat per emmagatzemar i transportar dades
- XML va ser dissenyat per ser autodescriptiu
- XML és una recomanació del W3C

XML no fa res

Potser és sobtant, però XML no fa res.

Aquesta nota és una nota per a Tove de Jani, emmagatzemada com a XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>
```

L'XML anterior és molt autodescriptiu:

- Té informació del remitent
- Té informació del receptor
- Té un encapçalament
- Té un cos de missatge

Però tot i així, l'XML anterior no fa res. XML és només informació embolicada en etiquetes.

Algú ha d'escriure un programari per enviar-lo, rebre-lo, emmagatzemar-lo o mostrar-lo. Podria mostrar-se així:

```
nota:
per a: Tove
de: Jani
tema: Reminder
cos: Don't forget me this weekend!
```

La diferència entre XML i HTML

XML i HTML es van dissenyar amb diferents objectius:

- XML va ser dissenyat per transportar dades, centrant-se en què són les dades
- L'HTML es va dissenyar per mostrar dades, centrant-se en com es veuen les dades
- Les etiquetes XML no estan predefinides com les etiquetes HTML

El llenguatge XML no té etiquetes predefinides.

Les etiquetes de l'exemple anterior (com <per_a> i <de>) no estan definides en cap estàndard XML. Aquestes etiquetes són "inventades" per l'autor del document XML.

HTML funciona amb etiquetes predefinides com <p>, <h1>, <table>, etc.

Amb XML, l'autor ha de definir tant les etiquetes com l'estruatura del document.

XML és extensible

La majoria de les aplicacions XML funcionaran com s'esperava, fins i tot si s'afegeixen (o s'eliminen) dades noves.

Imagineu una aplicació dissenyada per mostrar la versió original de note.xml (<a> <from> <heading> <body>).

A continuació, imagineu una versió més nova de note.xml amb elements <date> i <hour> afegits i un <heading> eliminat.

De la manera com es construeix XML, la versió anterior de l'aplicació encara pot funcionar:

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
    <date>2015-09-01</date>
    <hour>08:30</hour>
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</cos>
</note>
```

XML simplifica les coses

- XML simplifica l'intercanvi de dades
- XML simplifica el transport de dades
- XML simplifica els canvis de plataforma
- XML simplifica la disponibilitat de dades

Molts sistemes informàtics contenen dades en formats incompatibles. L'intercanvi de dades entre sistemes incompatibles (o sistemes actualitzats) és una tasca que requereix molt de temps per als desenvolupadors web. S'han de convertir grans quantitats de dades i sovint es perdren dades incompatibles.

XML emmagatzema les dades en format de text senzill. Això proporciona una manera independent del programari i del maquinari d'emmagatzemar, transportar i compartir dades.

XML també facilita l'expansió o l'actualització a nous sistemes operatius, noves aplicacions o nous navegadors, sense perdre dades.

Amb XML, les dades poden estar disponibles per a tot tipus de "màquines de lectura" com persones, ordinadors, màquines de veu, canals de notícies, etc.

XML és una recomanació del W3C

XML es va convertir en una recomanació del W3C el febrer de 1998.

6. Com es pot utilitzar XML?

XML s'utilitza en molts aspectes del desenvolupament web.

XML s'utilitza sovint per separar les dades de la presentació.

XML separa les dades de la presentació

XML no inclou cap informació sobre com es mostra.

Les mateixes dades XML es poden utilitzar en molts escenaris de presentació diferents.

Per això, amb XML, hi ha una separació total entre dades i presentació.

XML és sovint un complement a HTML

En moltes aplicacions HTML, XML s'utilitza per emmagatzemar o transportar dades, mentre que HTML s'utilitza per formatar i mostrar les mateixes dades.

XML Separa les dades de l'HTML

Quan es mostren dades en HTML, no cal editar el fitxer HTML quan les dades canvien.

Amb XML, les dades es poden emmagatzemar en fitxers XML separats.

Amb unes poques línies de codi JavaScript, podeu llegir un fitxer XML i actualitzar el contingut de dades de qualsevol pàgina HTML.

Dades de transacció

Existeixen milers de formats XML, en moltes indústries diferents, per descriure les transaccions de dades del dia a dia:

- Transaccions financeres
- Dades mèdiques
- Dades matemàtiques
- Mesures científiques
- Informació de notícies
- Serveis meteorològics

Exemple: XML News

XMLNews és una especificació per intercanviar notícies i altra informació.

L'ús d'un estàndard facilita tant als productors de notícies com als consumidors de notícies produir, rebre i arxivjar qualsevol tipus d'informació de notícies a través de diferents maquinari, programari i llenguatges de programació.

Un exemple de document XMLNews:

```

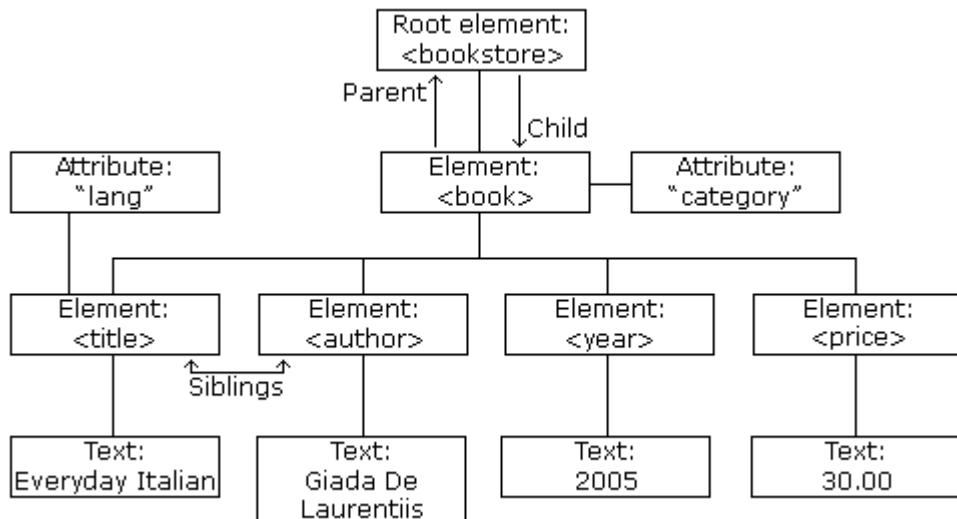
<?xml version="1.0" encoding="UTF-8"?>
<nitf>
  <head>
    <title>Colombia Earthquake</title>
  </head>
  <body>
    <headline>
      <h1>143 Dead in Colombia Earthquake</h1>
    </headline>
    <byline>
      <bytag>By Jared Kotler, Associated Press Writer</bytag>
    </byline>
    <dateline>
      <location>Bogota, Colombia</location>
      <date>Monday January 25 1999 7:28 ET</date>
    </dateline>
  </body>
</nitf>

```

7. Arbre XML

Els documents XML formen una estructura d'arbre que comença a "l'arrel" i es ramifica a "les fulles".

L'estructura d'arbre XML



Un exemple de document XML

La imatge de dalt representa llibres i el corresponent XML:

```

<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J. K. Rowling</author>
  </book>

```

```

        <year>2005</year>
        <price>29.99</price>
    </book>
    <book category="web">
        <title lang="en">Learning XML</title>
        <author>Erik T. Ray</author>
        <year>2003</year>
        <price>39.95</price>
    </book>
</bookstore>
```

Estructura d'arbre XML

Els documents XML es formen com a **arbres d'elements**.

Un arbre XML comença en un **element arrel** i es ramifica des de l'arrel als **elements secundaris**.

Tots els elements poden tenir subelements (elements secundaris):

```

<root>
    <child>
        <subchild>.....</subchild>
    </child>
</root>
```

Els termes pare, fill i germà s'utilitzen per descriure les relacions entre els elements.

Els pares tenen fills. Els fills tenen pares. Els germans estan en el mateix nivell (germans i germanes).

Tots els elements poden tenir contingut de text (Harry Potter) i atributs (category="cooking").

Sintaxi autodescriptiva

XML utilitza una sintaxi molt autodescriptiva.

Un pròleg defineix la versió XML i la codificació de caràcters:

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

La línia següent és l' **element arrel** del document:

```
<bookstore>
```

La línia següent comença un element <book>:

```
<book category="cooking">
```

Els elements <book> tenen **4 elements secundaris** : <títol>, <autor>, <any>, <preu>.

```

<title lang="en">Everyday Italian</title>
<author>Giada De Laurentiis</author>
<year>2005</year>
<price>30.00</price>
```

La línia següent acaba l'element del llibre:

```
</book>
```

A partir d'aquest exemple, podeu suposar que el document XML conté informació sobre llibres d'una llibreria.

8. Regles de sintaxi XML

Les regles de sintaxi de XML són molt senzilles i lògiques. Les regles són fàcils d'aprendre i d'utilitzar.

Els documents XML han de tenir un element arrel

Els documents XML han de contenir un element **arrel que siga el pare** de tots els altres elements:

```
<root>
    <child>
        <subchild>.....</subchild>
    </child>
</root>
```

En aquest exemple **<note>** és l'element arrel:

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
    <to>Tomàs</to>
    <from>Joana</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>
```

El pròleg XML

Aquesta línia s'anomena **pròleg** XML :

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

El pròleg XML és opcional. Si existeix, ha de figurar primer al document.

Els documents XML poden contenir caràcters internacionals, com el noruec øæå o el francès êèé.

Per evitar errors, heu d'especificar la codificació utilitzada o desar els fitxers XML com a UTF-8.

UTF-8 és la codificació de caràcters predeterminada per als documents XML.

La codificació de caràcters es pot estudiar al nostre [Tutorial del conjunt de caràcters](https://www.w3schools.com/charsets/default.asp) (<https://www.w3schools.com/charsets/default.asp>).

UTF-8 també és la codificació predeterminada per a HTML5, CSS, JavaScript, PHP i SQL.

Tots els elements XML han de tenir una etiqueta de tancament

En XML, és il·legal ometre l'etiqueta de tancament. Tots els elements han de tenir una etiqueta de tancament:

```
<p>This is a paragraph.</p>
```

Nota: el pròleg XML no té una etiqueta de tancament! Això no és un error. El pròleg no forma part del document XML.

Les etiquetes XML distingeixen entre majúscules i minúscules

Les etiquetes XML distingeixen entre majúscules i minúscules. L'etiqueta <Carta> és diferent de l'etiqueta <carta>.

Les etiquetes d'obertura i tancament s'han d'escriure amb el mateix identificador:

<**message**>This is correct</**message**>

Les "etiquetes d'obertura i tancament" sovint s'anomenen "etiquetes d'inici i final". Pots dir-ho com vullgues. És exactament el mateix.

Els elements XML han d'anar correctament

En HTML, és possible que vegeu elements imbricats incorrectament:

<**b**><**i**>This text is bold and italic</**b**></**i**>

En XML, tots els elements han d'estar correctament imbricats entre si:

<**b**><**i**>This text is bold and italic</**i**></**b**>

A l'exemple anterior, "Anidat correctament" simplement vol dir que com que l'element <**i**> s'obre dins de l'element <**b**>, s'ha de tancar dins de l'element <**b**>.

Els valors dels atributs XML s'han de citar sempre

Els elements XML poden tenir atributs en parells nom/valor igual que en HTML.

En XML, els valors dels atributs sempre s'han de citar:

```
<note date="12/11/2007">
  <to>Tove</to>
  <from>Jani</from>
</note>
```

Referències d'entitats

Alguns caràcters tenen un significat especial en XML.

Si col·loqueu un caràcter com "<" dins d'un element XML, generarà un error perquè l'analitzador l'interpreta com l'inici d'un element nou.

Això generarà un error XML:

<**message**>salary < 1000</**message**>

Per evitar aquest error, substituïu el caràcter "<" per una **referència d'entitat**:

<**message**>salary < 1000</**message**>

Hi ha 5 referències d'entitats predefinides en XML:

| | | |
|--------|---|----------------|
| < | < | less than |
| > | > | greater than |
| & | & | ampersand |
| ' | ' | apostrophe |
| " | " | quotation mark |

Només < i & són estrictament il·legals en XML, però és un bon hàbit substituir > per >; també.

Comentaris en XML

La sintaxi per escriure comentaris en XML és similar a la d'HTML:

<!-- This is a comment -->

No es permeten dos guions al mig d'un comentari:

<!-- This is an invalid -- comment -->

L'espai en blanc es conserva en XML

XML no trunca diversos espais en blanc (HTML trunca diversos espais en blanc a un únic espai en blanc):

XML: Hello Tove

HTML: Hello Tove

XML ben format

Es diu que els documents XML que s'ajusten a les regles de sintaxi anteriors són documents XML "ben formats".

9. Elements XML

Un document XML conté elements XML.

Què és un element XML?

Un element XML és tot, des de l'etiqueta inicial de l'element fins l'etiqueta final de l'element.

<price>29.99</price>

Un element pot contenir:

- text
- atributs
- altres elements
- o una barreja de les anteriors

```
<bookstore>
    <book category="children">
        <title>Harry Potter</title>
        <author>J. K. Rowling</author>
        <year>2005</year>
        <price>29.99</price>
    </book>
    <book category="web">
        <title>Learning XML</title>
        <author>Erik T. Ray</author>
        <year>2003</year>
        <price>39.95</price>
    </book>
</bookstore>
```

A l'exemple anterior:

`<title>, <author>, <year> i <price>` tenen **contingut de text** perquè contenen text (com ara 29,99).

`<book> i </bookstore>` tenen **contingut d'elements**, perquè contenen elements.

`<book>` té un **atribut** (`category="..."`).

Elements XML buits

Es diu que un element sense contingut està buit.

En XML, podeu indicar un element buit com aquest:

`<element></element>`

També podeu utilitzar l'anomenada etiqueta de tancament automàtic: `<element/>`

Les dues formes produueixen resultats idèntics en programari XML (lectors, analitzadors, navegadors).

Els elements buits poden tenir atributs.

Regles de nomenclatura XML

Els elements XML han de seguir aquestes regles de denominació:

- Els noms dels elements distingeixen entre majúscules i minúscules
- Els noms dels elements han de començar amb una lletra o un guió baix
- Els noms dels elements no poden començar amb les lletres xml (o XML, o Xml, etc.)
- Els noms dels elements poden contenir lletres, dígits, guions, guions baixos i punts
- Els noms dels elements no poden contenir espais

Es pot utilitzar qualsevol nom, no es reserven paraules (excepte xml).

Bones pràctiques de denominació

Creeu noms descriptius, com aquest: `<persona>`, `<nom>`, `<cognom>`.

Creeu noms curts i senzills, com aquest: `<títol_llibre>` no com aquest: `<el_títol_del_llibre>`.

Eviteu "-". Si anomeneu alguna cosa "nom", alguns programaris poden pensar que voleu restar "nom" de "primer".

Eviteu ". ". Si anomeneu alguna cosa "first.name", algun programari pot pensar que "name" és una propietat de l'objecte "first".

Eviteu ":". Els dos punts es reserven per als espais de noms (més endavant).

Les lletres no angleses com éòá són perfectament legals en XML, però vés amb compte amb els problemes si el teu programari no els admet!

Convencions de de nominació

Algunes convencions de nomenclatura d'ús habitual per als elements XML:

Estil

Exemple

Descripció

| | | |
|-----------|--------------|--|
| minúscula | <nom> | Totes les lletres minúscules |
| majúscula | <NOM> | Totes les lletres majúscules |
| serp | <nom_de_nom> | El guió baix separa paraules (utilitzat habitualment a les bases de dades SQL) |
| Pascal | <Nom> | Primera lletra majúscula de cada paraula (utilitzada habitualment pels programadors de C) |
| camell | <nom> | Primera lletra majúscula de cada paraula excepte la primera (utilitzada habitualment a JavaScript) |

Consell! Tria el teu estil de nom i sigues coherent!

Els documents XML sovint tenen una base de dades corresponent. Una pràctica habitual és utilitzar les regles de denominació de la base de dades per als elements XML.

Els elements XML són extensibles

Els elements XML es poden ampliar per portar més informació.

Mireu l'exemple XML següent:

```
<note>
  <to>Tove</to>
  <from>Jani</from>
  <body>Don't forget me this weekend!</body>
</note>
```

Imaginem que hem creat una aplicació que extreu els elements <a>, <from> i <body> del document XML per produir aquesta eixida:

MISSATGE

A: Tove

De: Jani

Don't forget me this weekend!

Imagineu que l'autor del document XML hi ha afegit informació addicional:

```
<note>
  <date>2008-01-10</date>
  <to>Tomàs</to>
  <from>Joana</from>
  <heading>Recordatori</heading>
  <body>No m'oblides aquest cap de setmana!</body>
</note>
```

L'aplicació donara errades o fallarà?

No. L'aplicació hauria de poder trobar els elements <a>, <from> i <body> al document XML i produir la mateixa eixida.

Aquesta és una de les fortaleses d'XML. Es pot ampliar sense trencar aplicacions.

10. Atributs XML

Els elements XML poden tenir atributs, igual que HTML.

Els atributs estan dissenyats per contenir dades relacionades amb un element específic.

Els atributs XML s'han de citar

Els valors dels atributs sempre s'han de citar. Es poden utilitzar cometes simples o dobles.

Per al gènere d'una persona, l'element <person> es pot escriure així:

```
<person gender="female">
```

o així:

```
<person gender='female'>
```

Si el valor de l'atribut conté cometes dobles, podeu utilitzar cometes simples, com en aquest exemple:

```
<gangster name='George "Shotgun" Ziegler'>
```

o podeu utilitzar entitats de caràcters:

```
<gangster name="George &quot;Shotgun&quot; Ziegler">
```

Elements XML vs. atributs

Mireu aquests dos exemples:

```
<person gender="female">
    <firstname>Anna</firstname>
    <lastname>Smith</lastname>
</person>
<person>
    <gender>female</gender>
    <firstname>Anna</firstname>
    <lastname>Smith</lastname>
</person>
```

En el primer exemple, el gènere és un atribut. En l'últim exemple, el gènere és un element. Tots dos exemples proporcionen la mateixa informació.

No hi ha regles sobre quan utilitzar atributs o quan utilitzar elements en XML.

La meua manera preferida

Els tres documents XML següents contenen exactament la mateixa informació:

En el primer exemple s'utilitza un atribut de data:

```
<note date="2008-01-10">
    <to>Tomàs</to>
    <from>Joana</from>
</note>
```

En el segon exemple s'utilitza un element <date>:

```
<note>
    <date>2008-01-10</date>
    <to>Tomàs</to>
    <from>Joana</from>
</note>
```

Al tercer exemple s'utilitza un element <date> ampliat: (AQUEST ÉS EL MEU PREFERIT):

```

<note>
  <date>
    <year>2008</year>
    <month>01</month>
    <day>10</day>
  </date>
  <to>Tomàs</to>
  <from>Joan</from>
</note>

```

Evitar els atributs XML?

Algunes coses a tenir en compte quan s'utilitzen atributs són:

- els atributs no poden contenir diversos valors (els elements poden)
- els atributs no poden contenir estructures d'arbre (els elements poden)
- els atributs no es poden ampliar fàcilment (per a canvis futurs)

No acabes així:

```

<note day="10" month="01" year="2008" to="Tove" from="Jani"
heading="Reminder"
body="Don't forget me this weekend!">
</note>

```

Atributs XML per a metadades

De vegades, les referències d'ID s'assignen als elements. Aquests identificadors es poden utilitzar per identificar elements XML de la mateixa manera que l'atribut id en HTML. Aquest exemple ho demostra:

```

<messages>
  <note id="501">
    <to>Tove</to>
    <from>Jani</from>
    <heading>Recordatori</heading>
    <body>No m'oblides aquest cap de setmana!</body>
  </note>
  <note id="502">
    <to>Jani</to>
    <from>Tove</from>
    <heading>Re: Recordatori</heading>
    <body>No ho faré</body>
  </note>
</messages>

```

Els atributs id anteriors serveixen per identificar les diferents notes. No forma part de la nota en si.

El que estic intentant dir és que les metadades (dades sobre dades) s'han d'emmagatzemar com a atributs i les dades en si s'han d'emmagatzemar com a elements.

11. Espais de noms XML

Els espais de noms XML proporcionen un mètode per a evitar conflictes de noms d'elements.

Conflictes de noms

En XML, els noms dels elements els defineix el desenvolupador. Això sovint provoca un conflicte quan s'intenta barrejar documents XML de diferents aplicacions XML.

Aquest XML conté informació de la taula HTML:

```
<table>
  <tr>
    <td>Apples</td>
    <td>Bananas</td>
  </tr>
</table>
```

Aquest XML porta informació sobre una taula (un moble):

```
<table>
  <name>African Coffee Table</name>
  <width>80</width>
  <length>120</length>
</table>
```

Si afegirem aquests fragments XML, hi hauria un conflicte de nom. Tots dos contenen un element `<table>`, però els elements tenen contingut i significat diferents.

Un usuari o una aplicació XML no sabrà com gestionar aquestes diferències.

Resolució del conflicte de noms mitjançant un prefix

Els conflictes de noms en XML es poden evitar fàcilment mitjançant un prefix de nom.

Aquest XML porta informació sobre una taula HTML i un moble:

```
<h:table>
  <h:tr>
    <h:td>Apples</h:td>
    <h:td>Bananas</h:td>
  </h:tr>
</h:table>

<f:table>
  <f:name>African Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>
```

A l'exemple anterior, no hi haurà cap conflicte perquè els dos elements `<table>` tenen noms diferents.

Espais de noms XML: l'atribut `xmlns`

Quan s'utilitzen prefixes en XML, s'ha de definir un **espai de noms per al prefix**.

L'espai de noms es pot definir mitjançant un atribut `xmlns` a l'etiqueta inicial d'un element.

La declaració d'espai de noms té la sintaxi següent `prefix = "URI"`.

```
<root>
<h:table xmlns:h="http://www.w3.org/TR/html4/">
  <h:tr>
    <h:td>Apples</h:td>
    <h:td>Bananas</h:td>
```

```

        </h:tr>
    </h:table>

    <f:table xmlns:f="https://www.w3schools.com/furniture">
        <f:name>African Coffee Table</f:name>
        <f:width>80</f:width>
        <f:length>120</f:length>
    </f:table>
</root>

```

A l'exemple anterior:

L'atribut xmlns del primer element <table> dóna al prefix h: un espai de noms qualificat.

L'atribut xmlns del segon element <table> dóna al prefix f: un espai de noms qualificat.

Quan es defineix un espai de noms per a un element, tots els elements fills amb el mateix prefix s'associen amb el mateix espai de noms.

Els espais de noms també es poden declarar a l'element arrel XML:

```

<root xmlns:h="http://www.w3.org/TR/html4/"
      xmlns:f="https://www.w3schools.com/furniture">
    <h:table>
        <h:tr>
            <h:td>Apples</h:td>
            <h:td>Bananas</h:td>
        </h:tr>
    </h:table>

    <f:table>
        <f:name>African Coffee Table</f:name>
        <f:width>80</f:width>
        <f:length>120</f:length>
    </f:table>
</root>

```

Nota: l'URI de l'espai de noms no l'utilitza l'analitzador per cercar informació.

El propòsit d'utilitzar un URI és donar a l'espai de noms un nom únic.

Tanmateix, les empreses sovint utilitzen l'espai de noms com a punter a una pàgina web que conté informació sobre l'espai de noms.

Identificador uniforme de recursos (URI)

Un **identificador uniforme de recursos** (URI) és una cadena de caràcters que identifica un recurs d'Internet.

L'URI més comú és l' URL (**Uniform Resource Locator**) que identifica una adreça de domini d'Internet. Un altre tipus d'URI no tan comú és el **nom de recurs uniforme** (URN).

Espais de noms per defecte

Definir un espai de noms predeterminat per a un element ens estalvia l'ús de prefixos en tots els elements secundaris. Té la sintaxi següent:

`xmlns="namespaceURI"`

Aquest XML conté informació de la taula HTML:

```
<table xmlns="http://www.w3.org/TR/html4/">
  <tr>
    <td>Apples</td>
    <td>Bananas</td>
  </tr>
</table>
```

Aquest XML porta informació sobre un móble:

```
<table xmlns="https://www.w3schools.com/furniture">
  <name>African Coffee Table</name>
  <width>80</width>
  <length>120</length>
</table>
```

Tema 2. Utilització de llenguatges de marques en entorns web: HTML5



1. [Introducció HTML](#)
 2. [HTML bàsic](#)
 3. [Elements HTML](#)
 4. [Atributs HTML](#)
 5. [Encapçalaments HTML](#)
 6. [Paràgrafs HTML](#)
 7. [Estils HTML](#)
 8. [Format HTML](#)
 9. [Cites HTML](#)
 10. [Comentaris HTML](#)
 11. [Colors HTML](#)
 12. [HTML CSS](#)
 13. [Enllaços HTML](#)
 14. [Imatges HTML](#)
 15. [Favicon HTML](#)
 16. [Títol de la pàgina HTML](#)
 17. [Taules HTML](#)
 18. [Llistes HTML](#)
 19. [Bloc HTML i en línia](#)
 20. [Classes HTML](#)
 21. [Id HTML](#)
 22. [Iframes HTML](#)
 23. [HTML JavaScript](#)
 24. [Rutes dels fitxers HTML](#)
 25. [Capçalera HTML](#)
 26. [Disseny HTML](#)
 27. [HTML Responsive](#)
 28. [Codi informàtic HTML](#)
 29. [Semàntica HTML](#)
 30. [Guia d'estil HTML](#)
 31. [Entitats HTML](#)
 32. [Símbols HTML](#)
 33. [Conjunt de caràcters HTML](#)
 34. [HTML vs. XHTML](#)
 35. [Formularis HTML](#)
 36. [Atributs de formulari HTML](#)
 37. [Elements del formulari HTML](#)
 38. [Tipus de dades d'entrada en formularis HTML](#)
 39. [Atributs d'entrada en formularis HTML](#)
-

1. Introducció HTML

HTML és el llenguatge de marques estàndard per a pàgines web.

Amb HTML podeu crear el vostre propi lloc web.

L'HTML és fàcil d'aprendre: el gaudiràs!

HTML és el llenguatge de marques estàndard per crear pàgines web.

Què és HTML?

- HTML significa Hyper Text Markup Language
- HTML és el llenguatge de marques estàndard per crear pàgines web
- HTML descriu l'estructura d'una pàgina web
- HTML consta d'una sèrie d'elements
- Els elements HTML indiquen al navegador com mostrar el contingut
- Els elements HTML etiqueten fragments de contingut com ara "això és un encapçalament", "això és un paràgraf", "això és un enllaç", etc.

Exemple

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
    <h1>My First Heading</h1>
    <p>My first paragraph.</p>
  </body>
</html>
```

Exemple explicat

- La <!DOCTYPE html> declaració defineix que aquest document és un document HTML5
- L'element <html> és l'element arrel d'una pàgina HTML
- L'element <head> conté metainformació sobre la pàgina HTML
- L'element <title> especifica un títol per a la pàgina HTML (que es mostra a la barra de títol del navegador o a la pestanya de la pàgina)
- L'element <body> defineix el cos del document, i és un contenidor per a tots els continguts visibles, com ara encapçalaments, paràgrafs, imatges, hiperenllaços, taules, llistes, etc.
- L'element <h1> defineix un encapçalament gran
- L'element <p> defineix un paràgraf

Què és un element HTML?

Un element HTML es defineix per una etiqueta d'inici, una mica de contingut i una etiqueta final:

< tagname > El contingut va aquí... < /tagname >

L'element HTML és tot, des de l'etiqueta inicial fins a l'etiqueta final:

< h1 > El meu primer encapçalament < /h1 >

< p > El meu primer paràgraf. < /p >

| Start tag | Element content | End tag |
|-----------|---------------------|-------------|
| < h1 > | My First Heading | </h1> |
| < p > | My first paragraph. | </p> |
| | <i>none</i> | <i>none</i> |

Nota: alguns elements HTML no tenen contingut (com l'element
). Aquests elements s'anomenen elements buits. Els elements buits no tenen una etiqueta final!

Navegadors web

La finalitat d'un navegador web (Chrome, Edge, Firefox, Safari) és llegir documents HTML i mostrar-los correctament.

Un navegador no mostra les etiquetes HTML, però les utilitza per determinar com mostrar el document:

Història HTML

Des dels primers dies de la World Wide Web, hi ha hagut moltes versions d'HTML:

| Year | Version |
|------|---|
| 1989 | Tim Berners-Lee invented www |
| 1991 | Tim Berners-Lee invented HTML |
| 1993 | Dave Raggett drafted HTML+ |
| 1995 | HTML Working Group defined HTML 2.0 |
| 1997 | W3C Recommendation: HTML 3.2 |
| 1999 | W3C Recommendation: HTML 4.01 |
| 2000 | W3C Recommendation: XHTML 1.0 |
| 2008 | WHATWG HTML5 First Public Draft |
| 2012 | WHATWG HTML5 Living Standard |
| 2014 | W3C Recommendation: HTML5 |
| 2016 | W3C Candidate Recommendation: HTML5.1 |
| 2017 | W3C Recommendation: HTML5.1 2nd Edition |
| 2017 | W3C Recommendation: HTML5.2 |

2. HTML bàsic

En aquest capítol mostrarem alguns exemples bàsics d'HTML. No us preocupeu si fem servir etiquetes que encara no coneixeu.

Documents HTML

Tots els documents HTML han de començar amb una declaració de tipus de document:

<!DOCTYPE html>.

El document HTML en si comença amb <html> i acaba amb </html>.

La part visible del document HTML es troba entre <body> i </body>. Exemple:

```
<!DOCTYPE html>
<html>
  <body>
    <h1>My First Heading</h1>
    <p>My first paragraph.</p>
  </body>
</html>
```

La declaració <!DOCTYPE>

La declaració <!DOCTYPE> representa el tipus de document i ajuda els navegadors a mostrar les pàgines web correctament.

Només ha d'aparèixer una vegada, a la part superior de la pàgina (davant de les etiquetes HTML).

La declaració <!DOCTYPE> no distingeix entre majúscules i minúscules.

La declaració <!DOCTYPE> per a HTML5 és:

```
<!DOCTYPE html>
```

Encapçalaments HTML

Els encapçalaments HTML es defineixen amb les etiquetes de <h1> a <h6>.

<h1> defineix l'encapçalament més important. <h6> defineix l'encapçalament menys important. Exemple:

```
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
```

Paràgrafs HTML

Els paràgrafs HTML es defineixen amb l'etiqueta <p>. Exemple:

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

Enllaços HTML

Els enllaços HTML es defineixen amb l'etiqueta <a>. Exemple:

```
<a href="https://www.w3schools.com">This is a link</a>
```

La destinació de l'enllaç s'especifica a l'atribut href.

Els atributs s'utilitzen per proporcionar informació addicional sobre els elements HTML.

Imatges HTML

Les imatges HTML es defineixen amb l'etiqueta .

El fitxer font (src), el text alternatiu (alt), width i height es proporcionen com a atributs:

```

```

Veure codi font HTML:

Feu clic amb el botó dret en una pàgina HTML i seleccioneu "Mostra la font de la pàgina" (a Chrome) o "Mostra la font" (a l'Edge) o similar en altres navegadors. Això obrirà una finestra que conté el codi font HTML de la pàgina.

Inspeccioneu un element HTML:

Feu clic amb el botó dret sobre un element (o una àrea en blanc) i trieu "Inspeccionar" o "Inspecció de l'element" per veure de quins elements estan formats (veureu tant l'HTML com el CSS). També podeu editar l'HTML o CSS sobre la marxa al tauler Elements o Estils que s'obre.

3. Elements HTML

Un element HTML es defineix per una etiqueta d'inici, una mica de contingut i una etiqueta final.

Nota: alguns elements HTML no tenen contingut (com l'element
). Aquests elements s'anomenen elements buits. Els elements buits no tenen una etiqueta final!

Elements dins d'elements en HTML

Els elements HTML es poden imbricar (això vol dir que els elements poden contenir altres elements).

Tots els documents HTML consisteixen en elements HTML imbricats.

L'exemple següent conté quatre elements HTML (<html>, <body>, <h1> i <p>). Exemple:

```
<!DOCTYPE html>
<html>
  <body>
    <h1>My First Heading</h1>
    <p>My first paragraph.</p>
  </body>
</html>
```

Exemple explicat

L'element <html> és l'element arrel i defineix tot el document HTML.

Té una etiqueta d'inici <html> i una de final </html>.

Aleshores, dins de l'element <html> hi ha un element <body>:

L'element <body> defineix el cos del document.

Té una etiqueta d'inici <body> i una de final </body>.

Aleshores, dins de l'element <body> hi ha altres dos elements: <h1> i <p>

L'element <h1> defineix un encapçalament.

Té una etiqueta d'inici <h1> i una de final </h1>:

L'element <p> defineix un paràgraf.

Té una etiqueta d'inici <p> i una de final </p>:

No salteu mai l'etiqueta final

Alguns elements HTML es mostraran correctament, fins i tot si oblideu l'etiqueta final:

```
<html>
  <body>
    <p>This is a paragraph <!-- falta tancar p -->
    <p>This is a paragraph <!-- falta tancar p -->
  </body>
</html>
```

Mai confieu en això! Es poden produir resultats i errors inesperats si oblideu l'etiqueta final!

Elements HTML buits

Els elements HTML sense contingut s'anomenen elements buits.

L'etiqueta `
` defineix un salt de línia i és un element buit sense una etiqueta de tancament. Exemple:

```
<p>This is a <br> paragraph with a line break.</p>
```

HTML no distingeix entre majúscules i minúscules

Les etiquetes HTML no distingeixen entre majúscules i minúscules: `<P>` significa el mateix que `<p>`.

L'estàndard HTML no requereix etiquetes en minúscules, però W3C **recomana** minúscules en HTML i **exigeix** minúscules per a tipus de document més estrictes com XHTML.

4. Atributs HTML

Els atributs HTML proporcionen informació addicional sobre els elements HTML.

Atributs HTML

- Tots els elements HTML poden tenir **atributs**
- Els atributs proporcionen **informació addicional** sobre els elements
- Els atributs sempre s'especifiquen a l'**etiqueta d'inici**
- Els atributs solen venir en parells nom/valor com: **nom="valor"**

L'atribut href

L'etiqueta `<a>` defineix un hiperenllaç. L' `href` atribut especifica l'URL de la pàgina a la qual va l'enllaç. Exemple:

```
<a href="https://www.w3schools.com">Visit W3Schools</a>
```

L'atribut src

L'etiqueta `` s'utilitza per incrustar una imatge en una pàgina HTML. L'atribut `src` especifica el camí a la imatge que es mostrarà. Exemple:

```

```

Hi ha dues maneres d'especificar l'URL a l'atribut `src`:

1. URL absolut - Enllaços a una imatge externa allotjada en un altre lloc web. Exemple: src="https://www.w3schools.com/images/img_girl.jpg".

Notes: les imatges externes poden estar sota copyright. Si no obtenui permís per utilitzar-lo, és possible que infringiu les lleis de drets d'autor. A més, no podeu controlar imatges externes; es pot eliminar o canviar de sobte.

2. URL relatiu - Enllaços a una imatge allotjada dins del lloc web. Ací, l'URL no inclou el nom de domini. Si l'URL comença sense barra inclinada, serà relatiu a la pàgina actual. Exemple: src="img_girl.jpg". Si l'URL comença amb una barra inclinada, serà relatiu al domini. Exemple: src="/images/img_girl.jpg".

Consell: sempre és millor utilitzar URL relatius. No es trencaran si canvies de domini.

Els atributs d'amplada i alçada

L'etiqueta també ha de contenir els atributs width i height, que especificuen l'amplada i l'alçada de la imatge (en píxels). Exemple:

```

```

L'atribut alt

L' atribut Alt obligatori per a l'etiqueta especifica un text alternatiu per a una imatge, si la imatge per algun motiu no es pot mostrar. Això pot ser degut a una connexió lenta, a un error en l'atribut src o si l'usuari utilitza un lector de pantalla. Exemple:

```

```

L'atribut d'estil

L'atribut style s'utilitza per afegir estils a un element, com ara el color, el tipus de lletra, la mida i molt més. Exemple:

```
<p style="color:red;">This is a red paragraph.</p>
```

L'atribut lang

Sempre hauríeu d'incloure l'atribut lang dins de l'etiqueta <html> , per declarar l'idioma de la pàgina web. Això està pensat per ajudar els motors de cerca i els navegadors.

L'exemple següent especifica l'anglès com a idioma:

```
<!DOCTYPE html>
<html lang="en">
<body>
...
</body>
</html>
```

Els codis de país també es poden afegir al codi d'idioma de l'atribut lang . Així, els dos primers caràcters defineixen l'idioma de la pàgina HTML i els dos últims caràcters defineixen el país.

L'exemple següent especifica l'anglès com a idioma i els Estats Units com a país:

```
<!DOCTYPE html>
<html lang="en-US">
<body>
...
```

```
</body>  
</html>
```

L'atribut del títol

L'atribut title defineix informació addicional sobre un element.

El valor de l'atribut title es mostrarà com a informació en un recuadre quan passeu el ratolí per sobre de l'element. Exemple:

```
<p title="I'm a tooltip">This is a paragraph.</p>
```

Suggerim: Utilitzeu sempre els atributs en minúscules

L'estàndard HTML no requereix noms d'atributs en minúscules.

L'atribut title (i tots els altres atributs) es pot escriure amb majúscules o minúscules com **title** o **TITLE**.

Tanmateix, W3C **recomana** atributs en minúscules en HTML i **exigeix** atributs en minúscules per a tipus de documents més estrictes com XHTML.

Us suggerim: citar sempre els valors dels atributs

L'estàndard HTML no requereix cometes al voltant dels valors dels atributs.

No obstant això, W3C **recomana** cites en HTML i **exigeix** cites per a tipus de documents més estrictes com XHTML.

Cometes simples o dobles?

Les cometes dobles al voltant dels valors dels atributs són les més habituals en HTML, però també es poden utilitzar cometes simples.

En algunes situacions, quan el valor de l'atribut conté cometes dobles, és necessari utilitzar cometes simples:

```
<p title='John "ShotGun" Nelson'>
```

O viceversa:

```
<p title="John 'ShotGun' Nelson">
```

5. Encapçalaments HTML

Els encapçalaments HTML són títols o subtítols que voleu mostrar en una pàgina web.

Encapçalaments HTML

Els encapçalaments HTML es defineixen amb les etiquetes de **<h1>** a **<h6>**.

<h1> defineix l'encapçalament més important. **<h6>** defineix l'encapçalament menys important. Exemple:

```
<h1>Heading 1</h1>  
<h2>Heading 2</h2>  
<h3>Heading 3</h3>  
<h4>Heading 4</h4>  
<h5>Heading 5</h5>  
<h6>Heading 6</h6>
```

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

Nota: els navegadors afegeixen automàticament un espai en blanc (un marge) abans i després d'un encapçalament.

Els encapçalamens són importants

Els motors de cerca utilitzen els encapçalamens per indexar l'estructura i el contingut de les vostres pàgines web.

Els usuaris sovint passen per una pàgina pels seus encapçalamens. És important utilitzar els encapçalamens per mostrar l'estructura del document.

<h1> els encapçalamens s'han d'utilitzar per als encapçalamens principals, seguits dels <h2> encapçalamens, després els menys importants <h3>, i així successivament.

Nota: Utilitzeu encapçalamens HTML només per als encapçalamens. No utilitzeu encapçalamens per fer que el text siga **GRAN** o **en negreta**.

Títols més grans

Cada encapçalament HTML té una mida predeterminada. Tanmateix, podeu especificar la mida de qualsevol encapçalament amb l' style atribut, utilitzant la font-size propietat CSS. Exemple:

```
<h1 style="font-size:60px;">Heading 1</h1>
```

6. Paràgrafs HTML

Un paràgraf sempre comença en una línia nova i sol ser un bloc de text.

Paràgrafs HTML

L'element HTML <p> defineix un paràgraf.

Un paràgraf sempre comença en una línia nova i els navegadors afegeixen automàticament algun espai en blanc (un marge) abans i després d'un paràgraf. Exemple:

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

Visualització HTML

No podeu estar segur de com es mostrerà l'HTML.

Les pantalles grans o menudes i les finestres redimensionades crearan resultats diferents.

Amb HTML, no podeu canviar la visualització afegint espais addicionals o línies addicionals al vostre codi HTML.

El navegador eliminarà automàticament els espais i línies addicionals quan es mostre la pàgina. Exemple:

```
<p>  
This paragraph  
contains a lot of lines  
in the source code,  
but the browser  
ignores it.  
</p>
```

Regles horizontals HTML

L'etiqueta `<hr>` defineix un salt temàtic en una pàgina HTML i es mostra com a una regla horitzontal. L'element `<hr>` s'utilitza per separar el contingut (o definir un canvi) en una pàgina HTML. Exemple:

```
<h1>This is heading 1</h1>  
<p>This is some text.</p>  
<hr>  
<h2>This is heading 2</h2>  
<p>This is some other text.</p>  
<hr>
```

L'etiqueta `<hr>` és una etiqueta buida, el que significa que no té cap etiqueta final.

Salts de línia HTML

L'element HTML `
` defineix un salt de línia.

Utilitzeu `
` si voleu un salt de línia (una línia nova) sense començar un paràgraf nou:

```
<p>This is<br>a paragraph<br>with line breaks.</p>
```

L'etiqueta `
` és una etiqueta buida, el que significa que no té cap etiqueta final.

El problema del poema

Aquest poema es mostrerà en una sola línia. Exemple:

```
<p>  
My Bonnie lies over the ocean.  
My Bonnie lies over the sea.  
My Bonnie lies over the ocean.  
Oh, bring back my Bonnie to me.  
</p>
```

Solució: l'element HTML `<pre>`

L'element HTML `<pre>` defineix el text preformatat.

El text dins d'un element `<pre>` es mostra amb un tipus de lletra d'amplada fixa (normalment Courier) i conserva tant els espais com els salts de línia. Exemple:

```
<pre>  
My Bonnie lies over the ocean.
```

```
My Bonnie lies over the sea.  
My Bonnie lies over the ocean.  
Oh, bring back my Bonnie to me.  
</pre>
```

7. Estils HTML

L'atribut d'estil HTML s'utilitza per afegir estils a un element, com ara el color, el tipus de lletra, la mida i molt més.

```
<!DOCTYPE html>  
<html>  
<body>  
    <p>I am normal</p>  
    <p style="color:red;">I am red</p>  
    <p style="color:blue;">I am blue</p>  
    <p style="font-size:50px;">I am big</p>  
</body>  
</html>
```

L'atribut d'estil HTML

La configuració de l'estil d'un element HTML es pot fer amb l'atribut style. L'atribut d'estil HTML té la sintaxi següent:

```
<tagname style="property:value;">
```

La propietat és una propietat CSS. El valor és un valor CSS. Aprendràs més sobre CSS més adavant.

Color de fons

La propietat CSS background-color defineix el color de fons per a un element HTML:

```
<body style="background-color:powderblue;">  
    <h1>This is a heading</h1>  
    <p>This is a paragraph.</p>  
</body>
```

Altre exemple:

```
<body>  
    <h1 style="background-color:powderblue;">This is a heading</h1>  
    <p style="background-color:tomato;">This is a paragraph.</p>  
</body>
```

Color del text

La propietat de color CSS defineix el color del text per a un element HTML. Exemple:

```
<h1 style="color:blue;">This is a heading</h1>  
<p style="color:red;">This is a paragraph.</p>
```

Fonts

La propietat CSS font-family defineix el tipus de lletra que s'utilitzarà per a un element HTML. Exemple:

```
<h1 style="font-family:verdana;">This is a heading</h1>  
<p style="font-family:courier;">This is a paragraph.</p>
```

Mida del text

La propietat CSS font-size defineix la mida del text per a un element HTML. Exemple:

```
<h1 style="font-size:300%;">This is a heading</h1>
<p style="font-size:160%;">This is a paragraph.</p>
```

Alineació de text

La propietat CSS text-align defineix l'alignació horitzontal del text per a un element HTML. Exemple:

```
<h1 style="text-align:center;">Centered Heading</h1>
<p style="text-align:center;">Centered paragraph.</p>
```

8. Format de text HTML

HTML conté diversos elements per definir text amb un significat especial: negreta, cursiva...

Elements de format HTML

Els elements de format es van dissenyar per mostrar tipus especials de text:

- **** - Text en negreta
- **<i>** - Text en cursiva
- **<mark>** - Text marcat
- **<small>** - Text més xicotet
- **** - Text eliminat
- **<ins>** - Text inserit
- **<sub>** - Text subíndex
- **<sup>** - Text en superíndex

Element **<small>** HTML

L'element HTML **<small>** defineix el text més xicotet:

Element **<mark>** HTML

L'element HTML **<mark>** defineix el text que s'ha de marcar o ressaltar. Exemple:

```
<p>Do not forget to buy <mark> milk </mark> today.</p>
```

HTML **** Element

L'element HTML **** defineix el text que s'ha suprimit d'un document. Els navegadors solen tocar una línia a través del text suprimit. Exemple:

```
<p>My favorite color is<del>blue</del>red.</p>
```

Element **<ins>** HTML

L'element **<ins>** HTML defineix un text que s'ha inserit en un document. Els navegadors solen subratllar el text inserit. Exemple:

```
<p>My favorite color is <del>blue</del> <ins>red</ins>.</p>
```

Element <sub> HTML

L'element HTML `<sub>` defineix el text de subíndex. El text de subíndex apareix mig caràcter per baix de la línia normal i, de vegades, es representa amb un tipus de lletra més xicotet. El text d'índex es pot utilitzar per a fórmules químiques, com H₂O:

Element <sup> HTML

L'element HTML `<sup>` defineix el text en superíndex. El text en superíndex apareix mig caràcter per sobre de la línia normal i, de vegades, es representa amb un tipus de lletra més xicotet.

9. Elements de citació i citació HTML

En aquest punt repassarem els elements HTML `<blockquote>`, `<q>`, `<abbr>`, `<address>`, `<cite>` i `<bdo>`.

HTML <blockquote> per a cites

L'element `<blockquote>` HTML defineix una secció que es cita d'una altra font.

Els navegadors soLEN sagnar els emENTS `<blockquote>`

HTML <q> per a cites curtes

L'etiqueta HTML `<q>` defineix una cita curta.

Els navegadors normalment insereixen cometes al voltant de la citació.

Abreviatures HTML

L'etiqueta HTML `<abbr>` defineix una abreviatura o un acrònim, com ara "HTML", "CSS", "Mr.", "Dr.", "ASAP", "ATM".

Marcar abreviatures pot proporcionar informació útil als navegadors, sistemes de traducció i motors de cerca. Exemple:

```
<p>The <abbr title="World Health Organization">WHO</abbr> was founded in  
1948.</p>
```

HTML <address> per a la informació de contacte

L'etiqueta HTML `<address>` defineix la informació de contacte de l'autor/propietari d'un document o d'un article.

La informació de contacte pot ser una adreça de correu electrònic, URL, adreça física, número de telèfon, identificador de xarxes socials, etc.

El text de l'element `<address>` normalment es mostra en *cursiva* i els navegadors sempre afegiran un salt de línia abans i després de l'element `<address>`. Exemple:

```
<address>  
Written by John Doe.<br>  
Visit us at:<br>  
Example.com<br>  
Box 564, Disneyland<br>  
USA  
</address>
```

HTML <cite> per al títol del treball

L'etiqueta **<cite>** HTML defineix el títol d'una obra creativa (per exemple, un llibre, un poema, una cançó, una pel·lícula, una pintura, una escultura, etc.).

Nota: el nom d'una persona no és el títol d'una obra.

El text de l'element **<cite>** normalment es mostra en *cursiva*. Exemple:

```
<p><cite>The Scream</cite>by Edvard Munch. Painted in 1893.</p>
```

HTML <bdo> per a la substitució bidireccional

BDO són les sigles de Bi-Directional Override.

L'etiqueta HTML **<bdo>** s'utilitza per anul·lar la direcció del text actual. Exemple:

```
<bdo dir="rtl">This text will be written from right to left</bdo>
```

10. Comentaris HTML

Els comentaris HTML no es mostren al navegador, però poden ajudar a documentar el vostre codi font HTML.

Etiqueta de comentari HTML

Podeu afegir comentaris a la vostra font HTML utilitzant la sintaxi següent:

```
<!-- Write your comments here -->
```

Tingueu en compte que hi ha un signe d'exclamació (!) a l'etiqueta inicial, però no a l'etiqueta final.

Nota: el navegador no mostra els comentaris, però poden ajudar a documentar el vostre codi font HTML.

Afegeix comentaris

Amb els comentaris podeu col·locar notificacions i recordatoris al vostre codi HTML:

```
<!-- This is a comment -->  
<p>This is a paragraph.</p>  
<!-- Remember to add more information here -->
```

Amaga contingut

Els comentaris es poden utilitzar per ocultar contingut. Això pot ser útil si amagueu el contingut temporalment.

També podeu amagar més d'una línia. Tot el que hi ha entre el **<!--** i el **-->** s'amagarà de la pantalla.

Els comentaris també són excellents per depurar HTML, perquè podeu comentar les línies de codi HTML, una a la vegada, per cercar errors.

Amaga el contingut en línia

Els comentaris es poden utilitzar per ocultar parts al mig del codi HTML.

11. Colors HTML

Els colors HTML s'especifiquen amb noms de color predefinits o amb valors RGB, HEX, HSL, RGBA o HSLA.

Noms de colors

En HTML, es pot especificar un color mitjançant un nom de color:

https://www.w3schools.com/colors/colors_names.asp

HTML admet 140 noms de colors estàndard.

Color de fons

Podeu establir el color de fons per als elements HTML:

```
<h1 style="background-color:DodgerBlue;">Hello World</h1>
<p style="background-color:Tomato;">Lorem ipsum...</p>
```

Color del text

Podeu definir el color del text:

```
<h1 style="color:Tomato;">Hello World</h1>
<p style="color:DodgerBlue;">Lorem ipsum...</p>
<p style="color:MediumSeaGreen;">Ut wisi enim...</p>
```

Color de la vora

Podeu definir el color de les vores:

```
<h1 style="border:2px solid Tomato;">Hello World</h1>
<h1 style="border:2px solid DodgerBlue;">Hello World</h1>
<h1 style="border:2px solid Violet;">Hello World</h1>
```

Valors de color

En HTML, els colors també es poden especificar mitjançant valors RGB, valors HEX, valors HSL, valors RGBA i valors HSLA. Exemple:

```
<h1 style="background-color:rgb(255, 99, 71);">...</h1>
<h1 style="background-color:#ff6347;">...</h1>
<h1 style="background-color:hsl(9, 100%, 64%);">...</h1>
<h1 style="background-color:rgba(255, 99, 71, 0.5);">...</h1>
<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">...</h1>
```

Valors de color RGB

En HTML, un color es pot especificar com a valor RGB, utilitzant aquesta fórmula:

rgb (vermell, verd, blau)

Cada paràmetre (vermell, verd i blau) defineix la intensitat del color amb un valor entre 0 i 255.

Això vol dir que hi ha $256 \times 256 \times 256 = 16777216$ colors possibles!

Per exemple, `rgb(255, 0, 0)` es mostra en roig, perquè el roig s'estableix en el seu valor més alt (255) i els altres dos (verd i blau) es defineixen en 0.

Un altre exemple, `rgb(0, 255, 0)` es mostra com a verd, perquè el verd s'estableix en el seu valor més alt (255) i els altres dos (vermell i blau) es defineixen en 0.

Per mostrar el negre, establiu tots els paràmetres de color a 0, com aquest: `rgb(0, 0, 0)`.

Per mostrar el blanc, establiu tots els paràmetres de color a 255, com aquest: `rgb(255, 255, 255)`.

Valors de color HEX

En HTML, es pot especificar un color mitjançant un valor hexadecimal de la forma:

`#rrggbb`

On rr (roig), gg (verd) i bb (blau) són valors hexadecimals entre 00 i ff (igual que el decimal 0-255).

Per exemple, `#ff0000` es mostra en roig, perquè el roig s'estableix en el seu valor més alt (ff) i els altres dos (verd i blau) es defineixen en 00.

Un altre exemple, `#00ff00` es mostra com a verd, perquè el verd s'estableix al seu valor més alt (ff) i els altres dos (vermell i blau) es defineixen en 00.

Per mostrar el negre, configureu tots els paràmetres de color a 00, com aquest: `#000000`.

Per mostrar el blanc, configureu tots els paràmetres de color a ff, com aquest: `#ffffff`

12. Estils HTML - CSS

CSS són les sigles de Cascading Style Sheets. CSS estalvia molta feina. Pot controlar el disseny de diverses pàgines web alhora.

Què és CSS?

Els fulls d'estil en cascada (CSS) s'utilitzen per donar format al disseny d'una pàgina web.

Amb CSS, podeu controlar el color, la font, la mida del text, l'espaiat entre els elements, la posició i la disposició dels elements, quines imatges de fons o colors de fons s'han d'utilitzar, diferents pantalles per a diferents dispositius i mides de pantalla, i molt més!

Consell: la paraula **cascada** significa que un estil aplicat a un element pare també s'aplicarà a tots els elements secundaris dins del pare. Per tant, si configureu el color del text del cos com a "blau", tots els encapçalaments, paràgrafs i altres elements de text del cos també tindran el mateix color (tret que especifiqueu alguna cosa més)!

Utilitzant CSS

El CSS es pot afegir als documents HTML de 3 maneres:

- **En línia:** utilitzant l'atribut `style` dins dels elements HTML
- **Interna:** utilitzant un element `<style>` a la `<head>` secció
- **Extern:** utilitzant un element `<link>` per enllaçar a un fitxer CSS extern

La forma més habitual d'afegir CSS és mantenir els estils en fitxers CSS externs. No obstant això, en aquest tutorial farem servir estils en línia i interns, perquè això és més fàcil de mostrar i més fàcil de provar-ho tu mateix.

CSS en línia

Un CSS en línia s'utilitza per aplicar un estil únic a un únic element HTML.

Un CSS en línia utilitza l'atribut `style` d'un element HTML.

L'exemple següent estableix el color del text de l'element `<h1>` en blau i el color del text de l'element `<p>` en roig. Exemple:

```
<h1 style="color:blue;">A Blue Heading</h1>
<p style="color:red;">A red paragraph.</p>
```

CSS intern

S'utilitza un CSS intern per definir un estil per a una única pàgina HTML.

Un CSS intern es defineix a la secció `<head>` d'una pàgina HTML, dins d'un element `<style>`.

L'exemple següent estableix el color del text de TOTS els elements `<h1>` (d'aquesta pàgina) en blau i el color del text de TOTS els elements `<p>` en roig. A més, la pàgina es mostrerà amb un color de fons "powderblue":

```
<!DOCTYPE html>
<html>
<head>
    <style>
        body{ background-color: powderblue;}
        h1{color: blue;}
        p{color: red;}
    </style>
</head>
<body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
</body>
</html>
```

CSS extern

S'utilitza un full d'estil extern per definir l'estil de moltes pàgines HTML.

Per utilitzar un full d'estil extern, afegiu un element `<link>` a la secció `<head>` de cada pàgina HTML. Exemple:

```
<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
</body>
</html>
```

El full d'estil extern es pot escriure en qualsevol editor de text. El fitxer no ha de contenir cap codi HTML i s'ha de desar amb una extensió .css. A continuació, es mostra el fitxer "styles.css":

```
body{background-color: powderblue;}  
h1{color: blue;}  
p{color: red;}
```

Consell: amb un full d'estil extern, podeu canviar l'aspecte d'un lloc web sencer canviant un fitxer!

Colors, tipus de lletra i mides CSS

Ací, demostrarem algunes propietats CSS d'ús habitual. Més endavant aprendràs més sobre ells.

La propietat CSS **color** defineix el color del text que s'utilitzarà.

La propietat CSS **font-family** defineix el tipus de lletra que s'utilitzarà.

La propietat CSS **font-size** defineix la mida del text que s'utilitzarà.

```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
h1{  
    color: blue;  
    font-family: verdana;  
    font-size: 300%;  
}  
p{  
    color: red;  
    font-family: courier;  
    font-size: 160%;  
}  
</style>  
</head>  
<body>  
  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
  
</body>  
</html>
```

Border CSS

La propietat CSS **border** defineix una vora al voltant d'un element HTML. Exemple:

```
p{border: 2px solid powderblue;}
```

Padding CSS

La propietat **padding** CSS defineix un farciment (espai) entre el text i la vora. Exemple:

```
p{border: 2px solid powderblue; padding: 30px;}
```

Margin CSS

La propietat CSS **margin** defineix un marge (espai) fora de la vora. Exemple:

```
p{border: 2px solid powderblue; margin: 50px;}
```

13. Enllaços HTML

Els enllaços es troben en totes les pàgines web. Els enllaços permeten als usuaris fer clic d'una pàgina a una altra.

Enllaços HTML - Hiperenllaços

Els enllaços HTML són hiperenllaços. Podeu fer clic a un enllaç i anar a un altre document. Quan moveu el ratolí per sobre d'un enllaç, la fletxa del ratolí es convertirà en una xicoteta mà.

Nota: un enllaç no només ha de ser de text. Un enllaç pot ser una imatge o qualsevol altre element HTML!

Enllaços HTML - Sintaxi

L'etiqueta HTML `<a>` defineix un hiperenllaç. Té la sintaxi següent:

```
<a href="url">link text</a>
```

L'atribut més important de l'element `<a>` és l'atribut `href`, que indica la destinació de l'enllaç. El text de l'enllaç és la part que serà visible per al lector.

Si feu clic al text de l'enllaç, s'enviarà el lector a l'adreça URL especificada. Exemple:

```
<a href="https://www.w3schools.com/">Visit W3Schools!</a>
```

De manera predeterminada, els enllaços apareixeran de la manera següent a tots els navegadors:

- Un enllaç no visitat està subratllat i blau
- Un enllaç visitat està subratllat i lila
- Un enllaç actiu està subratllat i roig

Consell: per descomptat, els enllaços es poden dissenyar amb CSS, per a obtenir un altre aspecte!

Enllaços HTML: l'atribut target

Per defecte, la pàgina enllaçada es mostrarà a la finestra del navegador actual. Per canviar-ho, heu d'especificar un altre objectiu per a l'enllaç.

L'atribut `target` especifica on s'ha d'obrir el document enllaçat. Pot tenir un dels valors següents:

- `_self`- Per defecte. Obre el document a la mateixa finestra/pestanya en què es va fer clic
- `_blank`- Obre el document en una finestra o pestanya nova
- `_parent`- Obre el document al marc principal
- `_top`- Obre el document a tot el cos de la finestra

Utilitzeu target="_blank" per obrir el document enllaçat en una nova finestra o pestanya del navegador:

```
<a href="https://www.w3schools.com/" target="_blank">Visit W3Schools!</a>
```

URL absoluts versus URL relatius

Els dos exemples anteriors utilitzen un **URL absolut** (una adreça web completa) a l'atribut **href**.

S'especifica un enllaç local (un enllaç a una pàgina dins del mateix lloc web) amb un **URL relatiu** (sense la part "[https://www](#)"). Exemple:

```
<h2>Absolute URLs</h2>
<p><a href="https://www.w3.org/">W3C</a></p>
<p><a href="https://www.google.com/">Google</a></p>
<h2>Relative URLs</h2>
<p><a href="html_images.asp">HTML Images</a></p>
<p><a href="/css/default.asp">CSS Tutorial</a></p>
```

Enllaços HTML: utilitzeu una imatge com a enllaç

Per utilitzar una imatge com a enllaç, només cal posar l'etiqueta **** dins de l'etiqueta **<a>**. Exemple:

```
<a href="default.asp"></a>
```

Enllaç a una adreça de correu electrònic

Utilitzeu **mailto:** dins de l'atribut **href** per crear un enllaç que obliga el programa de correu electrònic de l'usuari (per permetre que s'envie un correu electrònic nou):

```
<a href="mailto:someone@example.com">Send email</a>
```

Botó com a enllaç

Per utilitzar un botó HTML com a enllaç, heu d'afegir algun codi JavaScript.

JavaScript us permet especificar què passa en determinats esdeveniments, com ara un clic en un botó. Exemple:

```
<button onclick="document.location='default.asp'">HTML Tutorial</button>
```

Títols d'enllaç

L'atribut **title** especifica informació addicional sobre un element. La informació es mostra més sovint com a text d'informació eines quan el ratolí es mou sobre un element:

```
<a href="https://www.w3schools.com/html/" title="Go to W3Schools HTML section">Visit our HTML Tutorial</a>
```

Enllaços HTML : diferents colors

Un enllaç HTML es mostra amb un color diferent en funció de si s'ha visitat, no visitat o està actiu. Però també podeu canviar els colors de l'enllaç mitjançant CSS:

Ací un exemple: un enllaç no visitat serà de color verd sense subratllat. Un enllaç visitat serà de color rosa sense subratllat. Un enllaç actiu serà groc i subratllat. A més, quan passeu el ratolí per sobre d'un enllaç (`a:hover`) es tornarà roig i subratllat:

```

<style>
a:link {
  color: green;
  background-color: transparent;
  text-decoration: none;
}

a:visited {
  color: pink;
  background-color: transparent;
  text-decoration: none;
}

a:hover {
  color: red;
  background-color: transparent;
  text-decoration: underline;
}

a:active {
  color: yellow;
  background-color: transparent;
  text-decoration: underline;
}
</style>

```

Crea adreces d'interès

Els enllaços HTML es poden utilitzar per crear adreces d'interès, de manera que els lectors puguen anar a parts específiques d'una pàgina web.

Creeu un marcador en HTML. Els marcadors poden ser útils si una pàgina web és molt llarga.

Per crear un marcador: primer creeu el marcador i, a continuació, afegiu-hi un enllaç.

Quan es fa clic a l'enllaç, la pàgina es desplaçarà cap avall o cap amunt fins a la ubicació amb el marcador. Exemple:

Primer, utilitzeu l'atribut **`id`** per crear un marcador:

```
<h2 id="C4">Chapter 4</h2>
```

A continuació, afegiu un enllaç al marcador ("Va al capítol 4") des de la mateixa pàgina:

```
<a href="#C4">Jump to Chapter 4</a>
```

També podeu afegir un enllaç a un marcador d'una altra pàgina:

```
<a href="html_demo.html#C4">Jump to Chapter 4</a>
```

14. Imatges HTML

Les imatges poden millorar el disseny i l'aparença d'una pàgina web. Exemple:

```

```

Sintaxi d'imatges HTML

L'etiqueta HTML **``** s'utilitza per incrustar una imatge en una pàgina web.

Les imatges no s'insereixen tècnicament en una pàgina web. Les imatges estan enllaçades a pàgines web. L'etiqueta `` crea un espai de retenció per a la imatge de referència.

L'etiqueta `` està buida, només conté atributs i no té cap etiqueta de tancament.

L'etiqueta `` té dos atributs obligatoris:

- `src` - Especifica el camí a la imatge
- `alt` - Especifica un text alternatiu per a la imatge

Sintaxi:

```

```

L'atribut `src`

L'atribut `src` obligatori especifica el camí (URL) a la imatge.

Nota: Quan es carrega una pàgina web, és el navegador, en aquest moment, qui rep la imatge d'un servidor web i l'insereix a la pàgina. Per tant, assegureu-vos que la imatge es mantinga al mateix lloc en relació a la pàgina web, en cas contrari, els vostres visitants rebran una icona d'enllaç trencat. La icona d'enllaç trencat i el `alt` text es mostren si el navegador no troba la imatge.

L'atribut `alt`

L'atribut `alt` obligatori proporciona un text alternatiu per a una imatge, si l'usuari per algun motiu no la pot visualitzar (a causa d'una connexió lenta, un error a l'atribut `src` o si l'usuari utilitza un lector de pantalla).

El valor de l'atribut `alt` ha de descriure la imatge: Si un navegador no pot trobar una imatge, mostrarà el valor de l'atribut `alt`

Mida de la imatge: amplada i alçada

Podeu utilitzar l'atribut `style` per especificar l'amplada i l'alçada d'una imatge. Exemple:

```

```

Alternativament, podeu utilitzar els atributs `width` i `height`:

```

```

Imatges en una altra carpeta

Si teniu les vostres imatges en una subcarpeta, heu d'incloure el nom de la carpeta a l'atribut `src`. Exemple:

```

```

Imatges en un altre servidor/lloc web

Alguns llocs web apunten a una imatge en un altre servidor.

Per apuntar a una imatge en un altre servidor, heu d'especificar un URL absolut (complet) a l'atribut `src`. Exemple:

```

```

Imatges animades

HTML permet GIF animats:

Imatge com a enllaç

Per utilitzar una imatge com a enllaç, poseu l'etiqueta `` dins de l'etiqueta `<a>`:

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

Imatge flotant

Utilitzeu la propietat CSS `float` per deixar que la imatge flote a la dreta o a l'esquerra d'un text. Exemple:

```
<p>
The image will float to the right of the text.</p>
<p>
The image will float to the left of the text.</p>
```

Imatges de fons HTML

Es pot especificar una imatge de fons per a quasi qualsevol element HTML.

Imatge de fons en un element HTML

Per afegir una imatge de fons a un element HTML, utilitzeu l'atribut HTML `style` i la propietat CSS `background-image`. Afegiu una imatge de fons a un element HTML:

```
<p style="background-image: url('img_girl.jpg');">
```

Imatge de fons en una pàgina

Si voleu que tota la pàgina tinga una imatge de fons, heu d'especificar la imatge de fons a l'element `<body>`. Afegiu una imatge de fons per a tota la pàgina:

```
<style>
body {
    background-image: url('img_girl.jpg');
}
</style>
```

Repetició de fons

Si la imatge de fons és més menuda que l'element, la imatge es repetirà, horitzontalment i verticalment, fins que arriba al final de l'element. Per a evitar que la imatge de fons es repeteisca, establiu la propietat `background-repeat` a `no-repeat`:

```
<style>
body {
    background-image: url('example_img_girl.jpg');
    background-repeat: no-repeat;
}
</style>
```

Portada de fons

Si voleu que la imatge de fons cobreisca tot l'element, podeu establir la propietat **background-size** a **cover**.

A més, per assegurar-vos que tot l'element estiga sempre cobert, configureu la propietat **background-attachment** a **fixed**:

D'aquesta manera, la imatge de fons cobrirà tot l'element, sense estirar-se (la imatge mantindrà les seues proporcions originals). Exemple:

```
<style>
body {
    background-image: url('img_girl.jpg');
    background-repeat: no-repeat;
    background-attachment: fixed;
    background-size: cover;
}
</style>
```

Estirament de fons

Si voleu que la imatge de fons s'estire per adaptar-se a tot l'element, podeu establir la **background-size** propietat a **100% 100%**.

Proveu de canviar la mida de la finestra del navegador i veureu que la imatge s'estirarà, però sempre cobreix tot l'element. Exemple:

```
<style>
body {
    background-image: url('img_girl.jpg');
    background-repeat: no-repeat;
    background-attachment: fixed;
    background-size: 100% 100%;
}
</style>
```

Element HTML <picture>

L'element HTML **<picture>** us permet mostrar imatges diferents per a diferents dispositius o mides de pantalla. L'element HTML **<picture>** ofereix als desenvolupadors web més flexibilitat a l'hora d'especificar recursos d'imatge.

L'element **<picture>** conté un o més **<source>** elements, cadascun referint-se a diferents imatges a través de l'atribut **srcset**. D'aquesta manera, el navegador pot triar la imatge que millor s'adapte a la vista i/o dispositiu actuals.

Cada element **<source>** té un **media** atribut que defineix quan la imatge és la més adequada. Exemple:

```
<picture>
    <source media="(min-width: 650px)" srcset="img_food.jpg">
    <source media="(min-width: 465px)" srcset="img_car.jpg">
    
</picture>
```

Nota: especifiqueu sempre un element **** com a darrer element fill de l'element **<picture>**. L'element **** l'utilitzen els navegadors que no admeten l'element **<picture>** o si cap de les etiquetes **<source>** coincideix.

Quan utilitzar l'element d'picture

Hi ha dos propòsits principals per a l'element <picture>:

1. Ample de banda

Si teniu una pantalla o un dispositiu xicotet, no cal carregar un fitxer d'imatge gran. El navegador utilitzarà el primer element <source> amb valors d'atribut coincidents i ignorarà qualsevol dels elements següents.

2. Suport de format

És possible que alguns navegadors o dispositius no admeten tots els formats d'imatge. En utilitzar l'element <picture>, podeu afegir imatges de tots els formats i el navegador utilitzarà el primer format que reconeix i ignorarà qualsevol dels elements següents. El navegador utilitzarà el primer format d'imatge que reconeix:

```
<picture>
  <source srcset="img_avatar.png">
  <source srcset="img_girl.jpg">
  
</picture>
```

Nota: el navegador utilitzarà el primer element <source> amb valors d'atribut coincidents i ignorarà els elements <source> següents.

15. Favicon HTML

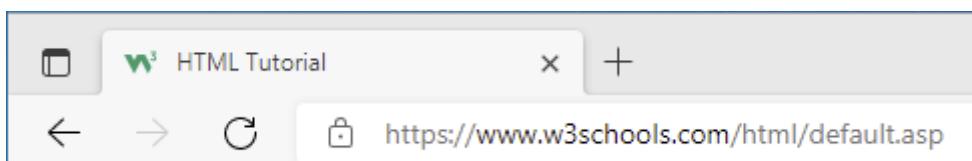
Un favicon és una xicoteta imatge que es mostra al costat del títol de la pàgina a la pestanya del navegador.

Com afegir un favicon en HTML

Podeu utilitzar qualsevol imatge que us agrade com a favicon. També podeu crear el vostre propi favicon a llocs com <https://www.favicon.cc>.

Consell: un favicon és una imatge menuda, de manera que hauria de ser una imatge senzilla amb un alt contrast.

Es mostra una imatge de favicon a l'esquerra del títol de la pàgina a la pestanya del navegador, com aquesta:



Per afegir un favicon al vostre lloc web, deseu la vostra imatge de favicon al directori arrel del vostre servidor web o creeu una carpeta al directori arrel anomenada imatges i deseu la vostra imatge de favicon en aquesta carpeta. Un nom comú per a una imatge de favicon és "favicon.ico".

A continuació, afegiu un element `<link>` al fitxer "index.html", després de l'element `<title>`, com aquest:

```
<!DOCTYPE html>
<html>
<head>
  <title>My Page Title</title>
  <link rel="icon" type="image/x-icon" href="/images/favicon.ico">
</head>
<body>
  <h1>This is a Heading</h1>
  <p>This is a paragraph.</p>
</body>
</html>
```

16. Títol de la pàgina HTML

Cada pàgina web hauria de tenir un títol per descriure el significat de la pàgina. L'element `<title>` afegeix un títol a la vostra pàgina:

```
<!DOCTYPE html>
<html>
<head>
  <title>HTML Tutorial</title>
</head>
<body>
The content of the document.....
</body>
</html>
```

El títol es mostra a la barra de títol del navegador. El títol ha de descriure el contingut i el significat de la pàgina.

El títol de la pàgina és molt important per a l'optimització de motors de cerca (SEO). Els algorismes dels motors de cerca utilitzen el text per decidir l'ordre quan s'enlacen pàgines als resultats de la cerca.

L'element `<title>`:

- defineix un títol a la barra d'eines del navegador
- proporciona un títol per a la pàgina quan s'afegeix als preferits
- mostra un títol per a la pàgina als resultats del motor de cerca

Per tant, intenta que el títol siga com més precís i significatiu millor!

17. Taules HTML

Les taules HTML permeten als desenvolupadors web organitzar les dades en files i columnes. Exemple:

| Company | Contact | Country |
|----------------------------|-----------------|---------|
| Alfreds Futterkiste | Maria Anders | Germany |
| Centro comercial Moctezuma | Francisco Chang | Mexico |

| | | |
|------------------------------|-----------------|---------|
| Ernst Handel | Roland Mendel | Austria |
| Island Trading | Helen Bennett | UK |
| Laughing Bacchus Winecellars | Yoshi Tannamuri | Canada |

Definiu una taula HTML

Una taula en HTML consta de cel·les de taula dins de files i columnes. Una taula HTML senzilla:

```
<table>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
</table>
```

Cel·les de la taula

Cada cel·la de la taula està definida per una `<td>` i una etiqueta `</td>`. `td` representa les dades de la taula.

Nota: una cel·la de taula pot contenir tot tipus d'elements HTML: text, imatges, llistes, enllaços, altres taules, etc.

Files de taula

Cada fila de la taula comença amb una `<tr>` i acaba amb una etiqueta `</tr>`. `tr` significa fila de taula. Exemple:

```
<table>
  <tr>
    <td>Emil</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
  <tr>
    <td>16</td>
    <td>14</td>
    <td>10</td>
  </tr>
</table>
```

Podeu tenir tantes files com vulgueu en una taula. Només assegureu-vos que el nombre de cel·les siga el mateix a cada fila.

Capçaleres de la taula

De vegades voleu que les vostres cel·les siguin cel·les de capçalera de taula. En aquests casos, utilitzeu l'etiqueta `<th>` en lloc de l'etiqueta `<td>`. `th` representa la capçalera de la taula. Exemple:

```
<table>
  <tr>
    <th>Person 1</th>
    <th>Person 2</th>
    <th>Person 3</th>
  </tr>
  <tr>
    <td>Emil</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
  <tr>
    <td>16</td>
    <td>14</td>
    <td>10</td>
  </tr>
</table>
```

De manera predeterminada, el text dels elements `<th>` està en negreta i centralitzat, però ho podeu canviar amb CSS.

Etiquetes de taula HTML

| Tag | Description |
|-------------------------------|--|
| <code><table></code> | Defines a table |
| <code><th></code> | Defines a header cell in a table |
| <code><tr></code> | Defines a row in a table |
| <code><td></code> | Defines a cell in a table |
| <code><caption></code> | Defines a table caption |
| <code><colgroup></code> | Specifies a group of one or more columns in a table for formatting |
| <code><col></code> | Specifies column properties for each column within a <code><colgroup></code> element |
| <code><thead></code> | Groups the header content in a table |
| <code><tbody></code> | Groups the body content in a table |
| <code><tfoot></code> | Groups the footer content in a table |

Vores de la taula HTML

Les taules HTML poden tenir vores de diferents estils i formes.

Com afegir una vora

Per afegir una vora, utilitzeu la propietat `border` CSS als elements `table`, `th`, i `td`:

A small 2x3 grid table with a border, consisting of 6 cells arranged in two rows and three columns.

| | | |
|--|--|--|
| | | |
|--|--|--|

```
table, th, td {
    border: 1px solid black;
}
```

Vores de taula col·lapsades

Per evitar tenir vores dobles com a l'exemple anterior, establiu la `border-collapse` propietat CSS a `collapse`.

Això farà que les vores col·lapsen en una única vora:

| | | |
|--|--|--|
| | | |
| | | |
| | | |
| | | |

```
table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}
```

Vores de taula amb estil

Si configureu un color de fons per a cada cel·la i doneu a la vora un color blanc (el mateix que el fons del document), obtindreu la impressió d'una vora invisible:

| | | |
|--|--|--|
| | | |
| | | |
| | | |
| | | |

```
table, th, td {
    border: 1px solid white;
    border-collapse: collapse;
}
th, td {
    background-color: #96D4D4;
}
```

Vores de la taula rodones

Amb la propietat `border-radius`, les vores tenen cantonades arrodonides:

| | | |
|--|--|--|
| | | |
| | | |
| | | |
| | | |

```
table, th, td {
    border: 1px solid black;
    border-radius: 10px;
}
```

Ometeu la vora de la taula deixant de banda el selector CSS `table`:

| | | |
|--|--|--|
| | | |
| | | |
| | | |
| | | |

```
th, td {  
    border: 1px solid black;  
    border-radius: 10px;  
}
```

Vores de taula amb punts

Amb la propietat **border-style**, podeu definir l'aspecte de la vora.

| | | |
|--|--|--|
| | | |
| | | |
| | | |

Es permeten els valors següents:

- dotted
- dashed
- solid
- double
- groove
- ridge
- inset
- outset
- none
- hidden

```
th, td {  
    border-style: dotted;  
}
```

Color de la vora

Amb la propietat **border-color**, podeu establir el color de la vora.

| | | |
|--|--|--|
| | | |
| | | |

```
th, td {  
    border-color: #96D4D4;  
}
```

Mides de la taula HTML

Les taules HTML poden tenir diferents mides per a cada columna, fila o tota la taula.

Utilitzeu l'atribut **style** amb les propietats **width** o **height** per especificar la mida d'una taula, fila o columna.

Amplada de la taula HTML

Per establir l'amplada d'una taula, afegiu l'atribut **style** a l'element **<table>**. Estableix l'amplada de la taula al 100%:

Nota: L'ús d'un percentatge com a unitat de mida d'una amplada significa quina amplada es compararà aquest element amb el seu element pare, que en aquest cas és l'element **<body>**.

Amplada de la columna de la taula HTML

Per establir la mida d'una columna específica, afegiu l'atribut `style` a un element `<th>` o `<td>`. Exemple: Estableix l'amplada de la primera columna al 70% i també exemple d'amplada 100% de la taula:

```
<!DOCTYPE html>
<html>
<style>
table, th, td {
    border:1px solid black;
    border-collapse: collapse;
}
</style>
<body>

<h2>Set the first column to 70% of the table width</h2>

<table style="width:100%">
<tr>
    <th style="width:70%">Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
</tr>
<tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
</tr>
<tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
</tr>
</table>
</body>
</html>
```

Alçada de fila de la taula HTML

Per establir l'alçada d'una fila específica, afegiu l'atribut `style` a un element de fila de taula:

Exemple: Estableix l'alçada de la segona fila a 200 píxels.

```
<table style="width:100%">
<tr style="height:200px">
    <th style="width:70%">Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
</tr>
<tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
</tr>
</table>
```

Capçaleres HTML

Les taules HTML poden tenir capçaleres per a cada columna o fila, o per a moltes columnes/files.

| EMIL | TOBIES | LINUS |
|------|--------|-------|
| | | |
| | | |
| | | |
| | | |

| | | |
|-------|--|--|
| 8:00 | | |
| 9:00 | | |
| 10:00 | | |
| 11:00 | | |
| 12:00 | | |
| 13:00 | | |

| | EL MEU | DIMAR | DIM | COL-LECCIONAR | DIV |
|-------|-----------|-------|-----|---------------|-----|
| 8:00 | | | | | |
| 9:00 | | | | | |
| 10:00 | | | | | |
| 11:00 | | | | | |
| 12:00 | | | | | |

| DESEMBRE | | | | | |
|----------|--|--|--|--|--|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Capçaleres de taula HTML

Les capçaleres de les taules es defineixen amb elements `th`. Cada element `th` representa una cel·la de taula.

Capçaleres de taules verticals

Per utilitzar la primera columna com a capçaleres de taula, definiu la primera cel·la de cada fila com a element `<th>`:

Alinea les capçaleres de la taula

De manera predeterminada, les capçaleres de la taula estan en negreta i centrades:

| Nom | Cognom | Edat |
|------|--------|------|
| Jill | Smith | 50 |

Per alinear a l'esquerra les capçaleres de la taula, utilitzeu la propietat CSS `text-align: left;`:

```
th {
    text-align: left;
}
```

Capçalera per a diverses columnes

Podeu tenir una capçalera que abaste dues o més columnes.

| Nom | Edat |
|---------|---------|
| Jill | Smith |
| Vigília | Jackson |

Per fer-ho, utilitzeu l'atribut `colspan` de l' element `<th>`. Exemple:

```
<table>
  <tr>
    <th colspan="2">Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
</table>
```

Títol de la taula

Podeu afegir un títol que serveisca d'encapçalament per a tota la taula.

Estalvi mensual

| Mes | Estalvis |
|-------|----------|
| gener | \$100 |

Per afegir un títol a una taula, utilitzeu l'etiqueta `<caption>`. Exemple:

```
<table style="width:100%">
  <caption>Estalvi mensual</caption>
  <tr>
    <th>Month</th><th>Savings</th>
  </tr>
  <tr>
    <td>January</td><td>$100</td>
  </tr>
</table>
```

Emplenat i espaiat de la taula HTML

Les taules HTML poden ajustar el farciment, i també l'espai entre les cel·les.

| Amb farciment | | | Amb espaiat | | |
|---------------|------|------|-------------|------|------|
| Hola | Hola | Hola | Hola | Hola | Hola |
| Hola | Hola | Hola | Hola | Hola | Hola |
| Hola | Hola | Hola | Hola | Hola | Hola |

Taula HTML: farciment de cel·les

El farciment de la cel·la és l'espai entre les vores de la cel·la i el contingut de la cel·la. Per defecte, el farciment s'estableix a 0. Per afegir farciment a les cel·les de la taula, utilitzeu la propietat `padding` CSS. Exemple:

```
th, td {
  padding: 15px;
}
```

Per afegir farciment només per dalt del contingut, utilitzeu la propietat `padding-top`.

I els altres costats amb les propietats `padding-bottom`, `padding-left`, i `padding-right`:

```
th, td {
  padding-top: 10px;
  padding-bottom: 20px;
  padding-left: 30px;
  padding-right: 40px;
}
```

Taula HTML - Espaiat de cel·les

L'espai entre cel·les és l'espai entre cada cel·la. Per defecte, l'espai està configurat en 2 píxels. Per canviar l'espai entre cel·les de la taula, utilitzeu la propietat CSS `border-spacing` de l'element: `table`. Exemple:

```
table {
    border-spacing: 30px;
}
```

Taula HTML Colspan i Rowspan

Les taules HTML poden tenir cel·les que s'estenen per diverses files i/o columnes.

| NOM | | |
|-----|--|--|
| | | |
| | | |
| | | |
| | | |

| | | |
|-------|--|--|
| ABRIL | | |
| | | |
| | | |
| | | |
| | | |

| 2022 | | |
|--------|--|--|
| | | |
| FIESTA | | |
| | | |
| | | |

Taula HTML - Colspan

Per fer que una cel·la s'estenga en diverses columnes, utilitzeu l'atribut **colspan** (ja vist).

Taula HTML - Cowspan

Per fer que una cel·la s'estenga en diverses files, utilitzeu l'atribut **rowspan**:

```
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}
</style>
</head>
<body>

<h2>Cell that spans two rows</h2>
<p>To make a cell span more than one row, use the rowspan attribute.</p>

<table style="width:100%">
<tr>
<th>Name</th>
<td>Jill</td>
</tr>
<tr>
<th rowspan="2">Phone</th>
<td>555-1234</td>
</tr>
<tr>
<td>555-8745</td>
</tr>
</table>
</body>
</html>
```

Estil de taula HTML

Utilitzeu CSS per fer que les vostres taules es vegein millor.

Taula HTML - Zebra Stripes

Si afegiu un color de fons a una fila sí i altra no de la taula, obtindreu un bon efecte de ratlles de zebra.

Per estilitzar tots els altres elements de fila de la taula, utilitzeu el `:nth-child(even)` selector com aquest:

```
<!DOCTYPE html>
<html>
<head>
    <style>
        table {
            border-collapse: collapse;
            width: 100%;
        }
        th, td {
            text-align: left;
            padding: 8px;
        }

        tr:nth-child(even) {
            background-color: #D6EEEE;
        }
    </style>
</head>
<body>
<h2>Zebra Striped Table</h2>
<p>For zebra-striped tables, use the nth-child() selector and add a background-color to all even (or odd) table rows:</p>
<table>
    <tr>
        <th>First Name</th>
        <th>Last Name</th>
        <th>Points</th>
    </tr>
    <tr>
        <td>Peter</td>
        <td>Griffin</td>
        <td>$100</td>
    </tr>
    <tr>
        <td>Lois</td>
        <td>Griffin</td>
        <td>$150</td>
    </tr>
</table>
</body>
</html>
```

Zebra Striped Table

For zebra-striped tables, use the `nth-child()` selector and add a `background-color` to all even (or odd) table rows:

| First Name | Last Name | Points |
|------------|-----------|--------|
| Peter | Griffin | \$100 |
| Lois | Griffin | \$150 |
| Joe | Swanson | \$300 |
| Cleveland | Brown | \$250 |

Nota: si feu servir `(odd)` en comptes de `(even)`, l'estil es produirà a la fila 1, 3, 5, etc. en lloc de 2, 4, 6, etc.

Taula HTML: ratlles verticals de zebra

Per fer ratlles de zebra verticals, estileu totes les altres columnes , en lloc de totes les altres files .

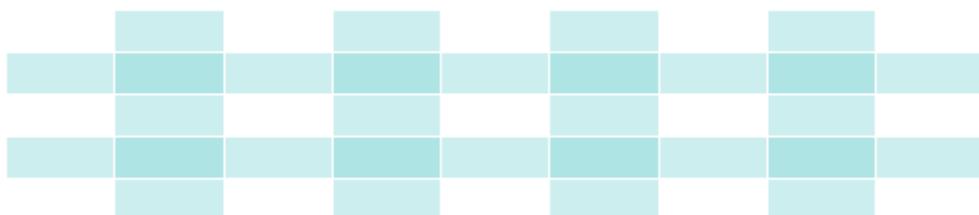
Estableix els elements `:nth-child(even)` de dades de la taula com aquest:

```
td:nth-child(even), th:nth-child(even) {  
    background-color: #D6EEEE;  
}
```

Nota: colloqueu el `:nth-child()` selector a tots dos `th` i `td` als elements si voleu tindre l'estil tant a les capçaleres com a les cel·les normals de la taula.

Combina ratlles de zebra verticals i horitzontals

Podeu combinar l'estil dels dos exemples anteriors i tindreu ratlles a cada altra fila i a cada altra columna.



Si utilitzeu un color transparent obtindreu un efecte de superposició.

Utilitzeu un `rgba()` color per especificar la transparència del color. Exemple:

```
tr:nth-child(even) {  
    background-color: rgba(150, 212, 212, 0.4);  
}  
  
th:nth-child(even), td:nth-child(even) {
```

```
        background-color: rgba(150, 212, 212, 0.4);  
    }
```

Divisors horitzontals

| Nom | Cognom | Estalvis |
|------|---------|----------|
| Pere | Grifó | \$100 |
| Lois | Grifó | \$150 |
| Joe | Swanson | \$300 |

Si especifiqueu vores només a la part inferior de cada fila de la taula, tindreu una taula amb divisors horitzontals.

Afegiu la propietat `border-bottom` a tots els elements `<tr>` per obtenir divisors horitzontals. Exemple:

```
tr {  
    border-bottom: 1px solid #ddd;  
}
```

Taula Hoverable

Utilitzeu el selector `:hover` per essaltar les files de la taula sobre el ratolí. Exemple:

```
tr:hover {background-color: #D6EEEE;}
```

18. Llistes HTML

Les llistes HTML permeten als desenvolupadors web agrupar un conjunt d'elements relacionats en llistes.

Exemple

Una llista HTML no ordenada:

- Article
- Article
- Article
- Article

Una llista HTML ordenada:

1. Primer element
2. Segon article
3. Tercer element
4. Quart element

Llista HTML no ordenada

Una llista no ordenada comença amb l'etiqueta ``. Cada element de la llista comença amb l'etiqueta ``.

Els elements de la llista es marcaran amb vinyetes (xicotets cercles negres) per defecte:

```
<ul>  
    <li>Coffee</li>  
    <li>Tea</li>
```

```
<li>Milk</li>
</ul>
```

Llista HTML ordenada

Una llista ordenada comença amb l'etiqueta ``. Cada element de la llista comença amb l'etiqueta ``.

Els elements de la llista es marcaran amb números per defecte:

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

Llistes de descripció HTML

HTML també admet llistes de descripcions.

Una llista de descripció és una llista de termes, amb una descripció de cada terme.

L'etiqueta `<dl>` defineix la llista de descripció, l'etiqueta `<dt>` defineix el terme (nom) i l'etiqueta `<dd>` descriu cada terme:

```
<dl>
  <dt>Coffee</dt>
  <dd>- black hot drink</dd>
  <dt>Milk</dt>
  <dd>- white cold drink</dd>
</dl>
```

Etiquetes de llista HTML

Tag Description

| | |
|-------------------------|--|
| <code></code> | Defines an unordered list |
| <code></code> | Defines an ordered list |
| <code></code> | Defines a list item |
| <code><dl></code> | Defines a description list |
| <code><dt></code> | Defines a term in a description list |
| <code><dd></code> | Describes the term in a description list |

19. Bloc HTML i elements en línia

Cada element HTML té un valor de visualització per defecte, depenent del tipus d'element que siga. Hi ha dos valors de visualització: block i inline.

Elements a nivell de bloc

Un element a nivell de bloc sempre comença en una línia nova i els navegadors afegeixen automàticament algun espai (un marge) abans i després de l'element.

Un element a nivell de bloc sempre ocupa tota l'amplada disponible (s'estén cap a l'esquerra i la dreta tant com pot).

Dos elements de bloc utilitzats habitualment són: `<p>` i `<div>`.

L'element `<p>` defineix un paràgraf en un document HTML.

L'element `<div>` defineix una divisió o una secció en un document HTML. Exemple:

```
<p>Hello World</p>
<div>Hello World</div>
```

Aquests són els elements a nivell de bloc en HTML:

| | | | | |
|--------------|-----------|----------|--------------|------------|
| <address> | <article> | <aside> | <blockquote> | <canvas> |
| <dd> | <div> | <dl> | <dt> | <fieldset> |
| <figcaption> | <figure> | <footer> | <form> | <h1>-<h6> |
| <header> | <hr> | | <main> | <nav> |
| <noscript> | | <p> | <pre> | <section> |
| <table> | <tfoot> | | <video> | |

Elements en línia

Un element en línia no comença en una línia nova.

Un element en línia només ocupa l'amplada necessària.

Aquest és un element dins d' un paràgraf. Exemple:

```
<!DOCTYPE html>
<html>
<body>
<p>This is an inline span <span style="border: 1px solid black">Hello
World</span> element inside a paragraph.</p>
<p>The SPAN element is an inline element, and will not start on a new line
and only takes up as much width as necessary.</p>
</body>
</html>
```

Aquests són els elements en línia en HTML:

| | | | | |
|--------|----------|-----------|----------|------------|
| <a> | <abbr> | <acronym> | | <bdo> |
| <big> | | <button> | <cite> | <code> |
| <dfn> | | <i> | | <input> |
| <kbd> | <label> | <map> | <object> | <output> |
| <q> | <samp> | <script> | <select> | <small> |
| | | <sub> | <sup> | <textarea> |
| <time> | <tt> | <var> | | |

20. Atribut de classe

L'atribut HTML `class` s'utilitza per especificar una classe per a un element HTML.

Diversos elements HTML poden compartir la mateixa classe.

Ús de l'atribut de classe

L'atribut `class` s'utilitza normalment per assenyalar un nom de classe en un full d'estil. També pot ser utilitzat per un JavaScript per accedir i manipular elements amb el nom de classe específic.

En l'exemple següent tenim tres `<div>` elements amb un atribut `class` amb el valor de "ciutat". Tots els tres `<div>` elements tindran un estil igual d'acord amb la definició d'estil a la secció `.city` de capçalera. Exemple:

```
<!DOCTYPE html>
<html>
<head>
<style>
.city {
    background-color: tomato;
    color: white;
    border: 2px solid black;
    margin: 20px;
    padding: 20px;
}
</style>
</head>
<body>

<div class="city">
<h2>London</h2>
<p>London is the capital of England.</p>
</div>

<div class="city">
<h2>Paris</h2>
<p>Paris is the capital of France.</p>
</div>

<div class="city">
<h2>Tokyo</h2>
<p>Tokyo is the capital of Japan.</p>
</div>

</body>
</html>
```

En l'exemple següent tenim dos elements `` amb un atribut `class` amb el valor de "nota". Els dos elements `` tindran un estil igual d'acord amb la definició d'estil `.note` a la secció de capçalera. Exemple:

```
<!DOCTYPE html>
<html>
<head>
<style>
.note {
    font-size: 120%;
    color: red;
}
</style>
</head>
<body>

<h1>My <span class="note">Important</span> Heading</h1>
<p>This is some <span class="note">important</span> text.</p>
```

```
</body>
</html>
```

L'atribut **class** es pot utilitzar en **qualsevol** element HTML. El nom de la classe distingeix entre majúscules i minúscules!

La sintaxi per a la classe

Per crear una classe escriu un punt (.), seguit d'un nom de classe. A continuació, definiu les propietats CSS dins de claus {}. Ja hem vist .city abans:

```
.city {
    background-color: tomato;
    color: white;
    border: 2px solid black;
    margin: 20px;
    padding: 20px;
}
```

Classes múltiples

Els elements HTML poden pertànyer a més d'una classe.

Per definir diverses classes, separeu els noms de classe amb un espai, p. ex. <div class="city main">. L'element s'estilitzarà segons totes les classes especificades.

A l'exemple següent, el primer <h2> element pertany tant a la **city** classe com a la **main** classe, i obtindrà els estils CSS de les dues classes. Exemple:

```
<h2 class="city main">London</h2>
<h2 class="city">Paris</h2>
<h2 class="city">Tokyo</h2>
```

Diferents elements poden compartir la mateixa classe

Diferents elements HTML poden apuntar al mateix nom de classe.

A l'exemple següent, tots dos <h2> i <p> assenyalen la classe "ciutat" i compartiran el mateix estil. Exemple:

```
<h2 class="city">Paris</h2>
<p class="city">Paris is the capital of France</p>
```

Ús de l'atribut de classe en JavaScript

JavaScript també pot utilitzar el nom de la classe per realitzar determinades tasques per a elements específics.

JavaScript pot accedir a elements amb un nom de classe específic amb el **getElementsByClassName()** mètode:

Exemple. Feu clic a un botó per amagar tots els elements amb el nom de classe "city":

```
<script>
function myFunction() {
    var x = document.getElementsByClassName("city");
    for (var i = 0; i < x.length; i++) {
        x[i].style.display = "none";
    }
}
```

```
}
```

```
</script>
```

21. Atribut id HTML

L'atribut HTML `id` s'utilitza per especificar un identificador únic per a un element HTML.

No podeu tenir més d'un element amb el mateix identificador en un document HTML.

Ús de l'atribut id

L'atribut `id` especifica un identificador únic per a un element HTML. El valor de l'atribut `id` ha de ser únic dins del document HTML.

L'atribut `id` s'utilitza per apuntar a una declaració d'estil específica en un full d'estil. JavaScript també l'utilitza per accedir i manipular l'element amb l'identificador específic.

La sintaxi per a `id` és: escriviu un caràcter hash (#), seguit d'un nom d'identificador. A continuació, definiu les propietats CSS dins de claus {}.

A l'exemple següent tenim un `<h1>` element que apunta al nom d'identificador "myHeader". Aquest `<h1>` element s'estilitzarà segons la `#myHeader` definició d'estil a la secció de capçalera. Exemple:

```
<!DOCTYPE html>
<html>
<head>
<style>
#myHeader {
    background-color: lightblue;
    color: black;
    padding: 40px;
    text-align: center;
}
</style>
</head>
<body>
<h1 id="myHeader">My Header</h1>
</body>
</html>
```

Nota: el nom d'identificador distingeix entre majúscules i minúscules! El nom d'identificador ha de contenir almenys un caràcter, no pot començar amb un número i no ha de contenir espais en blanc (espais, tabulacions, etc.).

Diferència entre classe i ID

Un nom de classe pot ser utilitzat per diversos elements HTML, mentre que un nom d'identificació només l'ha d'utilitzar un element HTML dins de la pàgina. Exemple:

```
<style>
/* Style the element with the id "myHeader" */
#myHeader {
    background-color: lightblue;
    color: black;
    padding: 40px;
    text-align: center;
```

```

}

/* Style all elements with the class name "city" */
.city {
    background-color: tomato;
    color: white;
    padding: 10px;
}
</style>

<!-- An element with a unique id -->
<h1 id="myHeader">My Cities</h1>

<!-- Multiple elements with same class -->
<h2 class="city">London</h2>
<p>London is the capital of England.</p>

<h2 class="city">Paris</h2>
<p>Paris is the capital of France.</p>

<h2 class="city">Tokyo</h2>
<p>Tokyo is the capital of Japan.</p>

```

Ús de l'atribut id a JavaScript

JavaScript també pot utilitzar l' atribut **Id** per realitzar algunes tasques per a aquest element específic.

JavaScript pot accedir a un element amb un identificador específic amb el **getElementById()** mètode:

Exemple. Utilitzeu l'atribut **id** per manipular text amb JavaScript:

```

<script>
function displayResult() {
    document.getElementById("myHeader").innerHTML = "Have a nice day!";
}
</script>

```

22. Iframes HTML

Un iframe HTML s'utilitza per mostrar una pàgina web dins d'una pàgina web.

Sintaxi HTML Iframe

L'etiqueta HTML **<iframe>** especifica un marc en línia.

Un marc en línia s'utilitza per incrustar un altre document dins del document HTML actual.

Sintaxi

```
<iframe src="url" title="description"></iframe>
```

Consell: és una bona pràctica incloure sempre un atribut **title** per a **<iframe>**. Els lectors de pantalla l'utilitzen per llegir quin és el contingut de l'iframe.

Iframe: estableix l'alçada i l'amplada

Utilitzeu els atributs **height** i **width** per especificar la mida de l'iframe.

L'alçada i l'amplada s'especifiquen en píxels de manera predeterminada. Exemple:

```
<iframe src="demo_iframe.htm" height="200" width="300" title="Iframe Example"></iframe>
```

Iframe: eliminar la vora

Per defecte, un iframe té una vora al seu voltant.

Per eliminar la vora, afegiu l'atribut **style** i utilitzeu la propietat **border** CSS. Exemple:

```
<iframe src="demo_iframe.htm" style="border:none;" title="Iframe Example"></iframe>
```

Amb CSS, també podeu canviar la mida, l'estil i el color de la vora de l'iframe:

```
<iframe src="demo_iframe.htm" style="border:2px solid red;" title="Iframe Example"></iframe>
```

Iframe: objectiu per a un enllaç

Un iframe es pot utilitzar com a marc de destinació per a un enllaç.

L'atribut **target** de l'enllaç ha de fer referència a l'atribut **name** de l'iframe. Exemple:

```
<!DOCTYPE html>
<html>
<body>

    <h2>Iframe - Target for a Link</h2>

    <iframe src="demo_iframe.htm" name="iframe_a" height="300px" width="50%" title="Iframe Example"></iframe>

    <p><a href="https://www.w3schools.com" target="iframe_a">W3Schools.com</a></p>

    <p>When the target attribute of a link matches the name of an iframe, the link will open in the iframe.</p>

</body>
</html>
```

23. JavaScript HTML

JavaScript fa que les pàgines HTML siguin més dinàmiques i interactives.

L'etiqueta HTML **<script>**

L'etiqueta HTML **<script>** s'utilitza per definir un script del costat del client (JavaScript).

L'element **<script>** conté declaracions d'script o apunta a un fitxer de script extern mitjançant l'atribut **src**.

Els usos habituals de JavaScript són la manipulació d'imatges, la validació de formularis i els canvis dinàmics de contingut.

Per seleccionar un element HTML, JavaScript fa servir el mètode **document.getElementById()**

Aquest exemple de JavaScript escriu "Hola JavaScript!" en un element HTML amb id="demo":

```
<script>
document.getElementById("demo").innerHTML = "Hello JavaScript!";
</script>
```

Una xicoteta mostra de JavaScript

Aquests són alguns exemples del que pot fer JavaScript:

Exemple. JavaScript pot canviar el contingut:

```
document.getElementById("demo").innerHTML = "Hello JavaScript!";
```

JavaScript pot canviar els estils:

```
document.getElementById("demo").style.fontSize = "25px";
document.getElementById("demo").style.color = "red";
document.getElementById("demo").style.backgroundColor = "yellow";
```

JavaScript pot canviar els atributs:

```
document.getElementById("image").src = "picture.gif";
```

L'etiqueta HTML <noscript>

L'etiqueta HTML <noscript> defineix un contingut alternatiu que es mostrerà als usuaris que tinguen scripts desactivats al seu navegador o que tinguen un navegador que no admet scripts. Exemple:

```
<script>
document.getElementById("demo").innerHTML = "Hello JavaScript!";
</script>
<noscript>Sorry, your browser does not support JavaScript!</noscript>
```

24. Rutes dels fitxers HTML

Una ruta de fitxer descriu la ubicació d'un fitxer a l'estructura de carpetes d'un lloc web.

Exemples de ruta de fitxer

| Camí | Descripció |
|---------------------------------|--|
| | El fitxer "picture.jpg" es troba a la mateixa carpeta que la pàgina actual |
| | El fitxer "picture.jpg" es troba a la carpeta d'imatges de la carpeta actual |
| | El fitxer "picture.jpg" es troba a la carpeta d'imatges a l'arrel del web actual |
| | El fitxer "picture.jpg" es troba a la carpeta un nivell més amunt de la carpeta actual |

Rutes dels fitxers HTML

Una ruta de fitxer descriu la ubicació d'un fitxer a l'estructura de carpetes d'un lloc web.

Els camins dels fitxers s'utilitzen quan s'enllaça amb fitxers externs, com ara:

- Pàgines web
- Imatges
- Fulls d'estil
- JavaScripts

Rutes de fitxer absolutes

Una ruta de fitxer absoluta és l'URL complet d'un fitxer:

```

```

Camins de fitxers relatius

Una ruta relativa del fitxer apunta a un fitxer relatiu a la pàgina actual.

A l'exemple següent, la ruta del fitxer apunta a un fitxer de la carpeta d'imatges situada a l'arrel del web actual:

```

```

Millors pràctiques

La millor pràctica és utilitzar camins de fitxer relatius (si és possible).

Quan utilizeu camins de fitxer relatius, les vostres pàgines web no estaran vinculades a la vostra URL base actual. Tots els enllaços funcionaran al vostre propi ordinador (localhost), així com al vostre domini públic actual i als vostres futurs dominis públics.

25. Capçalera HTML

L'element HTML `<head>` és un contenidor per als elements següents: `<title>`, `<style>`, `<meta>`, `<link>`, `<script>` i `<base>`.

L'element HTML `<head>`

L'element `<head>` és un contenidor per a metadades (dades sobre dades) i es col·loca entre l'etiqueta `<html>` i l'etiqueta `<body>`.

Les metadades HTML són dades sobre el document HTML. Les metadades no es mostren.

Les metadades normalment defineixen el títol del document, el conjunt de caràcters, els estils, els scripts i altres metainformació.

L'element HTML `<title>`

L'element `<title>` defineix el títol del document. El títol ha de ser només de text i es mostra a la barra de títol del navegador o a la pestanya de la pàgina.

El contingut d'un títol de pàgina és molt important per a l'optimització de motors de cerca. Els algorismes dels motors de cerca utilitzen el títol de la pàgina per decidir l'ordre quan s'enumeren pàgines als resultats de la cerca.

L'element `<title>`:

- defineix un títol a la barra d'eines del navegador

- proporciona un títol per a la pàgina quan s'afegeix als preferits
- mostra un títol per a la pàgina als resultats del motor de cerca

Per tant, intenta que el títol siga com més precís i significatiu millor!

L'element HTML <style>

L'element <style> s'utilitza per definir informació d'estil per a una única pàgina HTML:

```
<style>
    body {background-color: powderblue;}
    h1 {color: red;}
    p {color: blue;}
</style>
```

L'element HTML <link>

L'element <link> defineix la relació entre el document actual i un recurs extern.

L'etiqueta <link> s'utilitza sovint per enllaçar a fulls d'estil externs:

```
<link rel="stylesheet" href="mystyle.css">
```

L'element HTML <meta>

L'element <meta> s'utilitza normalment per especificar el conjunt de caràcters, la descripció de la pàgina, les paraules clau, l'autor del document i la configuració de la finestra gràfica.

Les metadades no es mostraran a la pàgina, però les fan servir els navegadors (com mostrar el contingut o tornar a carregar la pàgina), els motors de cerca (paraules clau) i altres serveis web. Exemple:

```
<meta charset="UTF-8">
<meta name="description" content="Free Web tutorials">
<meta name="keywords" content="HTML, CSS, JavaScript">
<meta name="author" content="John Doe">
```

Configuració de la finestra gràfica

La finestra gràfica és l'àrea visible de l'usuari d'una pàgina web. Varia segons el dispositiu: serà més xicoteta en un telèfon mòbil que en una pantalla d'ordinador.

Heu d'incloure l'element <meta> següent a totes les vostres pàgines web:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Això dóna al navegador instruccions sobre com controlar les dimensions i l'escala de la pàgina.

La `width=device-width` estableix l'amplada de la pàgina a l'amplada de la pantalla del dispositiu (que variarà segons el dispositiu).

La `initial-scale=1.0` estableix el nivell de zoom inicial quan el navegador carrega la pàgina per primera vegada.

L'element HTML <script>

Ja vistos.

L'element HTML <base>

L'element '`<base>`' especifica l'URL base i/o l'objectiu per a tots els URL relatius d'una pàgina.

L'etiqueta `<base>` ha de tenir present un atribut `href` o `target`, o tots dos.

Només hi pot haver un únic `<base>` element en un document!

Exemple. Especifica un URL predeterminat i un objectiu predeterminat per a tots els enllaços d'una pàgina:

```
<head>
  <base href="https://www.w3schools.com/" target="_blank">
</head>
```

Elements de capçalera HTML

Tag Description

| | |
|-----------------------------|---|
| <code><head></code> | Defines information about the document |
| <code><title></code> | Defines the title of a document |
| <code><base></code> | Defines a default address or a default target for all links on a page |
| <code><link></code> | Defines the relationship between a document and an external resource |
| <code><meta></code> | Defines metadata about an HTML document |
| <code><script></code> | Defines a client-side script |
| <code><style></code> | Defines style information for a document |

26. Elements i tècniques de disseny HTML

Els llocs web normalment mostren contingut en diverses columnes (com una revista o un diari).

Elements de disseny HTML

HTML té diversos elements semàntics que defineixen les diferents parts d'una pàgina web:



- `<header>`- Defineix una capçalera per a un document o una secció
- `<nav>`- Defineix un conjunt d'enllaços de navegació
- `<section>`- Defineix una secció en un document
- `<article>`- Defineix un contingut independent i autònom
- `<aside>`- Defineix contingut a part del contingut (com una barra lateral)
- `<footer>`- Defineix un peu de pàgina per a un document o una secció
- `<details>`- Defineix detalls addicionals que l'usuari pot obrir i tancar sota demanda
- `<summary>`- Defineix un encapsament per a l'element `<details>`

Tècniques de disseny HTML

Hi ha quatre tècniques diferents per crear dissenys multicolumnes. Cada tècnica té els seus pros i contres:

- marc CSS
- Propietat flotant CSS
- Flexbox CSS
- quadrícula CSS

Frameworks CSS

Si voleu crear el vostre disseny ràpidament, podeu utilitzar un marc CSS, com [W3.CSS](#) o [Bootstrap](#).

Disseny flotant CSS

És habitual fer dissenys web sencers mitjançant la propietat `float` CSS. `Float` és fàcil d'aprendre: només cal recordar com funcionen les propietats `float` i `clear`.

Desavantatges: els elements flotants estan lligats al flux de documents, cosa que pot perjudicar la flexibilitat.

Disseny CSS Flexbox

L'ús de flexbox garanteix que els elements es comporten de manera previsible quan el disseny de la pàgina ha d'adaptar-se a diferents mides de pantalla i diferents dispositius de visualització.

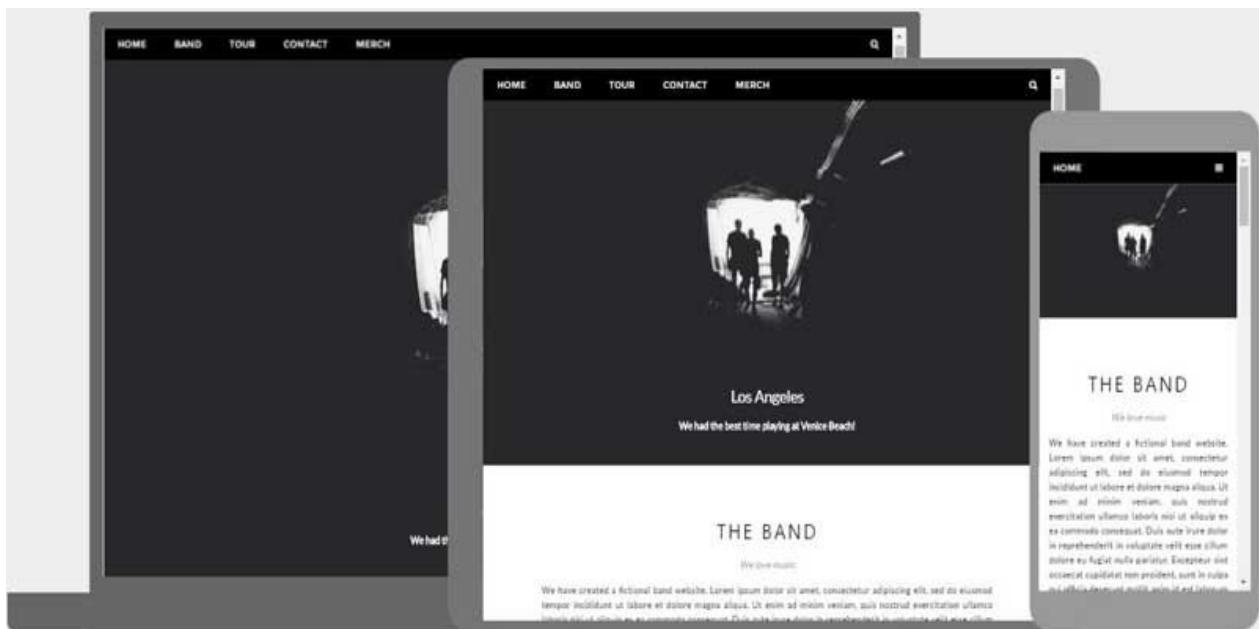
Disseny de quadrícula CSS

El mòdul de disseny de quadrícula CSS ofereix un sistema de disseny basat en quadrícula, amb files i columnes, que facilita el disseny de pàgines web sense haver d'utilitzar flotadors i posicionament.

27. Disseny web responsiu HTML

El disseny web responsiu consisteix a crear pàgines web que es vegen bé en tots els dispositius!

Un disseny web responsiu s'ajustarà automàticament a diferents mides de pantalla i finestres.



Què és el disseny web responsiu?

El disseny web responsiu consisteix a utilitzar HTML i CSS per redimensionar, amagar, reduir o ampliar automàticament un lloc web per fer-lo veure bé en tots els dispositius (ordinadors, tauletes i telèfons):

Configuració de la finestra gràfica

Per crear un lloc web responsiu, afegiu l'etiqueta <meta> següent a totes les vostres pàgines web. Exemple:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Això establirà la finestra gràfica de la vostra pàgina, que donarà instruccions al navegador sobre com controlar les dimensions i l'escala de la pàgina.

Imatges sensibles o responsives

Les imatges responsives són imatges que s'ajusten bé a qualsevol mida del navegador.

Utilitzant la propietat width. Si la propietat width CSS s'estableix al 100%, la imatge respondrà i augmentarà i baixarà l'escala. Exemple:

```

```

Tingueu en compte que a l'exemple anterior, la imatge es pot escalar per a ser més gran que la seva mida original. Una millor solució, en molts casos, serà utilitzar la propietat max-width.

Utilitzant la propietat max-width

Si la propietat max-width s'estableix al 100%, la imatge es reduirà si cal, però mai augmentarà per ser més gran que la seva mida original. Exemple:

```

```

Mostra imatges diferents segons l'amplada del navegador

L'element <picture>. Vist en el punt sobre imatges.

Mida del text sensible

La mida del text es pot configurar amb una unitat "vw", que significa "amplada de la finestra gràfica".

D'aquesta manera, la mida del text seguirà la mida de la finestra del navegador. Exemple

```
<h1 style="font-size:10vw">Hello World</h1>
```

Viewport és la mida de la finestra del navegador. 1vw = 1% de l'amplada de la finestra visual. Si el visor té 50 cm d'ample, 1vw és 0,5 cm.

Consultes de mitjans

A més de canviar la mida del text i les imatges, també és habitual utilitzar consultes multimèdia en pàgines web responsives.

Amb les consultes multimèdia podeu definir estils completament diferents per a diferents mides de navegador.

Exemple: canvieu la mida de la finestra del navegador per veure que els tres elements div següents es mostraran horitzontalment en pantalles grans i s'apilen verticalment en pantalles xicotetes:

```
<style>
.left, .right {
    float: left;
    width: 20%; /* The width is 20%, by default */
}

.main {
    float: left;
    width: 60%; /* The width is 60%, by default */
}
/* Use a media query to add a breakpoint at 800px: */
@media screen and (max-width: 800px) {
    .left, .main, .right {
        width: 100%; /* The width is 100%, when the viewport is 800px or
smaller */
    }
}
</style>
```

Pàgina web responsiva: exemple complet

Una pàgina web responsiva hauria de tenir un bon aspecte en pantalles grans d'escriptori i en telèfons mòbils xicotets. Prova aquest exemple:

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
* {
    box-sizing: border-box;
}
```

```

.menu {
    float: left;
    width: 20%;
    text-align: center;
}

.menu a {
    background-color: #e5e5e5;
    padding: 8px;
    margin-top: 7px;
    display: block;
    width: 100%;
    color: black;
}

.main {
    float: left;
    width: 60%;
    padding: 0 20px;
}

.right {
    background-color: #e5e5e5;
    float: left;
    width: 20%;
    padding: 15px;
    margin-top: 7px;
    text-align: center;
}

@media only screen and (max-width: 620px) {
    /* For mobile phones: */
    .menu, .main, .right {
        width: 100%;
    }
}

</style>
</head>
<body style="font-family:Verdana;color:#aaaaaa;">

<div style="background-color:#e5e5e5;padding:15px;text-align:center;">
    <h1>Hello World</h1>
</div>

<div style="overflow:auto">
    <div class="menu">
        <a href="#">Link 1</a>
        <a href="#">Link 2</a>
        <a href="#">Link 3</a>
        <a href="#">Link 4</a>
    </div>

    <div class="main">
        <h2>Lorum Ipsum</h2>
        <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
    </div>

    <div class="right">

```

```

<h2>About</h2>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
</div>
</div>

<div style="background-color:#e5e5e5;text-align:center;padding:10px;margin-top:7px;">© copyright w3schools.com</div>

</body>
</html>

```

Bootstrap

Un altre marc CSS popular és Bootstrap:

```

<!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap 5 Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js"></script>
</head>
<body>

<div class="container-fluid p-5 bg-primary text-white text-center">
  <h1>My First Bootstrap Page</h1>
  <p>Resize this responsive page to see the effect!</p>
</div>

<div class="container mt-5">
  <div class="row">
    <div class="col-sm-4">
      <h3>Column 1</h3>
      <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit...</p>
      <p>Ut enim ad minim veniam, quis nostrud exercitation ullamco
laboris...</p>
    </div>
    <div class="col-sm-4">
      <h3>Column 2</h3>
      <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit...</p>
      <p>Ut enim ad minim veniam, quis nostrud exercitation ullamco
laboris...</p>
    </div>
    <div class="col-sm-4">
      <h3>Column 3</h3>
      <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit...</p>
      <p>Ut enim ad minim veniam, quis nostrud exercitation ullamco
laboris...</p>
    </div>
  </div>
</div>

</body>
</html>

```

28. Elements del codi informàtic HTML

HTML <code> per al codi informàtic

L'element <code> HTML s'utilitza per definir un fragment de codi informàtic. El contingut interior es mostra amb el tipus de lletra monoespai predeterminat del navegador.

Penseu que l'element <code> no conserva espais en blanc addicionals i salts de línia.

Per solucionar-ho, podeu posar l'element <code> dins d'un element <pre>. Exemple:

```
<pre>
<code>
x = 5;
y = 6;
z = x + y;
</code>
</pre>
```

29. Elements semàntics HTML

Elements semàntics = elements amb significat.

Què són els elements semàntics?

Un element semàntic descriu clarament el seu significat tant per al navegador com per al desenvolupador.

Exemples d'elements **no semàntics**: <div> i - No diu res sobre el seu contingut.

Exemples d'elements **semàntics**: <form>, <table>, i <article> - Defineix clarament el seu contingut.

Elements semàntics en HTML

Molts llocs web contenen codi HTML com: <div id="nav"> <div class="header"> <div id="footer"> per indicar la navegació, la capçalera i el peu de pàgina.

En HTML hi ha alguns elements semàntics que es poden utilitzar per definir diferents parts d'una pàgina web:

```
<article> <aside> <details> <figcaption> <figure> <footer> <header> <main>
<mark> <nav> <section> <summary> <time>
```



Element <secció> HTML

L'element <section> defineix una secció en un document.

Segons la documentació HTML del W3C: "Una secció és una agrupació temàtica de contingut, normalment amb un encapçalament".

Exemples d'on es pot utilitzar un element <section>:

- Capítols
- Introducció
- Notícies
- Informació de contacte

Una pàgina web normalment es pot dividir en seccions per a la introducció, el contingut i la informació de contacte.

Exemple. Dues seccions en un document:

```

<section>
  <h1>WWF</h1>
  <p>The World Wide Fund for Nature (WWF) is an international organization
  working on issues regarding the conservation, research and restoration of
  the environment, formerly named the World Wildlife Fund. WWF was founded in
  1961.</p>
</section>

<section>
  <h1>WWF's Panda symbol</h1>
  <p>The Panda has become the symbol of WWF. The well-known panda logo of WWF
  originated from a panda named Chi Chi that was transferred from the Beijing
  Zoo to the London Zoo in the same year of the establishment of WWF.</p>
</section>

```

Element <article> HTML

L'element <article> especifica contingut independent i autònom.

Un article hauria de tenir sentit per si mateix i hauria de ser possible distribuir-lo independentment de la resta del lloc web.

Exemples d'on es pot utilitzar l'element <article>:

- Publicacions del fòrum
- Publicacions del blog
- Comentaris dels usuaris
- Targetes de productes
- Articles periodístics

Exemple . Dos articles amb contingut independent i autònom:

```
<article>
  <h2>Google Chrome</h2>
  <p>Google Chrome is a web browser developed by Google, released in 2008.
  Chrome is the world's most popular web browser today!</p>
</article>

<article>
  <h2>Mozilla Firefox</h2>
  <p>Mozilla Firefox is an open-source web browser developed by Mozilla.
  Firefox has been the second most popular web browser since January,
  2018.</p>
</article>
```

Exemple 2. Utilitzeu CSS per estilitzar l'element <article>:

```
<html>
<head>
<style>
.all-browsers {
  margin: 0;
  padding: 5px;
  background-color: lightgray;
}

.all-browsers > h1, .browser {
  margin: 10px;
  padding: 5px;
}

.browser {
  background: white;
}

.browser > h2, p {
  margin: 4px;
  font-size: 90%;
}
</style>
</head>
<body>
<article class="all-browsers">
  <h1>Most Popular Browsers</h1>
  <article class="browser">
    <h2>Google Chrome</h2>
    <p>Google Chrome is a web browser developed by Google, released in
    2008. Chrome is the world's most popular web browser today!</p>
  </article>
  <article class="browser">
    <h2>Mozilla Firefox</h2>
    <p>Mozilla Firefox is an open-source web browser developed by Mozilla.
    Firefox has been the second most popular web browser since January,
    2018.</p>
  </article>
</article>
```

```

<article class="browser">
  <h2>Microsoft Edge</h2>
  <p>Microsoft Edge is a web browser developed by Microsoft, released in
2015. Microsoft Edge replaced Internet Explorer.</p>
</article>
</article>
</body>
</html>

```

Nidificar `<article>` a `<section>` o viceversa?

L'element `<article>` especifica contingut independent i autònom.

L'element `<section>` defineix la secció d'un document.

Podem utilitzar les definicions per decidir com aniar aquests elements? No podem!

Per tant, trobareu pàgines HTML amb `<section>` elements que contenen `<article>` elements i `<article>` elements que contenen elements `<section>`.

Element HTML `<capçalera>`

L'element `<header>` representa un contingut introductorí o un conjunt d'enllaços de navegació.

Un element `<header>` normalment conté:

- un o més elements d'encapçalament (`<h1>` - `<h6>`)
- logotip o icona
- informació d'autoria

Nota: podeu tenir diversos `<header>` elements en un document HTML. Tanmateix, `<header>` no es pot collocar dins d'un `<footer>`, `<address>` o d'un altre `<header>` element.

Exemple. Una capçalera per a un `<article>`:

```

<article>
  <header>
    <h1>What Does WWF Do?</h1>
    <p>WWF's mission:</p>
  </header>
  <p>WWF's mission is to stop the degradation of our planet's natural
environment,
and build a future in which humans live in harmony with nature.</p>
</article>

```

Element HTML `<footer>`

L'element `<footer>` defineix un peu de pàgina per a un document o secció.

Un element `<footer>` normalment conté:

- informació d'autoria
- informació de copyright
- informació de contacte
- mapa del lloc
- tornar als enllaços superiors
- documents relacionats

Podeu tenir diversos elements `<footer>` en un document.

Exemple. Una secció de peu de pàgina d'un document:

```
<footer>
  <p>Author: Hege Refsnes</p>
  <p><a href="mailto:hege@example.com">hege@example.com</a></p>
</footer>
```

Element <nav> HTML

L'element <nav> defineix un conjunt d'enllaços de navegació.

Tingueu en compte que NO tots els enllaços d'un document haurien d'estar dins d'un element <nav>. L'element <nav> només està pensat per als principals blocs d'enllaços de navegació.

Els navegadors, com ara els lectors de pantalla per a usuaris amb discapacitat, poden utilitzar aquest element per determinar si s'ometen la representació inicial d'aquest contingut.

Exemple. Un conjunt d'enllaços de navegació:

```
<nav>
  <a href="/html/">HTML</a> |
  <a href="/css/">CSS</a> |
  <a href="/js/">JavaScript</a> |
  <a href="/jquery/">jQuery</a>
</nav>
```

Element HTML <aside>

L'element <aside> defineix algun contingut a part del contingut en què es col·loca (com una barra lateral).

El contingut <aside> ha d'estar indirectament relacionat amb el contingut que l'envolta.

Exemple. Mostra alguns continguts a part del contingut on es col·loca:

```
<p>My family and I visited The Epcot center this summer. The weather was
nice, and Epcot was amazing! I had a great summer together with my family!
</p>

<aside>
<h4>Epcot Center</h4>
<p>Epcot is a theme park at Walt Disney World Resort featuring exciting
attractions, international pavilions, award-winning fireworks and seasonal
special events.</p>
</aside>
```

Exemple 2. Utilitzeu CSS per estilitzar l'element <aside>:

```
<html>
<head>
<style>
  aside {
    width: 30%;
    padding-left: 15px;
    margin-left: 15px;
    float: right;
    font-style: italic;
    background-color: lightgray;
  }
</style>
</head>
<body>
<h1>My family and I visited The Epcot center this summer. The weather was
nice, and Epcot was amazing! I had a great summer together with my family!
</h1>
<aside>
<h4>Epcot Center</h4>
<p>Epcot is a theme park at Walt Disney World Resort featuring exciting
attractions, international pavilions, award-winning fireworks and seasonal
special events.</p>
</aside>
</body>
</html>
```

```

</style>
</head>
<body>

<p>My family and I visited The Epcot center this summer. The weather was nice, and Epcot was amazing! I had a great summer together with my family!</p>

<aside>
<p>The Epcot center is a theme park at Walt Disney World Resort featuring exciting attractions, international pavilions, award-winning fireworks and seasonal special events.</p>
</aside>

<p>My family and I visited The Epcot center this summer. The weather was nice, and Epcot was amazing! I had a great summer together with my family!</p>
<p>My family and I visited The Epcot center this summer. The weather was nice, and Epcot was amazing! I had a great summer together with my family!</p>

</body>
</html>

```

Elements HTML <figure> i <figcaption>

L'etiqueta <figure> especifica contingut autònom, com ara il·lustracions, diagrames, fotos, llistats de codi, etc.

L'etiqueta <figcaption> defineix un títol per a un element <figure>. L'element <figcaption> es pot col·locar com a primer o com a darrer fill d'un element <figure>.

L'element defineix la imatge/il·lustració real. Exemple:

```

<figure>
    
    <figcaption>Fig1. - Trulli, Puglia, Italy.</figcaption>
</figure>

```

Per què els elements semàntics?

Segons el W3C: "Una web semàntica permet compartir i reutilitzar dades entre aplicacions, empreses i comunitats".

Elements semàntics en HTML

A continuació es mostra una llista d'alguns dels elements semàntics en HTML.

| Tag | Description |
|--------------|---|
| <article> | Defines independent, self-contained content |
| <aside> | Defines content aside from the page content |
| <details> | Defines additional details that the user can view or hide |
| <figcaption> | Defines a caption for a <figure> element |
| <figure> | Specifies self-contained content, like illustrations, diagrams, photos, code listings, etc. |
| <footer> | Defines a footer for a document or section |
| <header> | Specifies a header for a document or section |
| <main> | Specifies the main content of a document |
| <mark> | Defines marked/highlighted text |
| <nav> | Defines navigation links |

| | |
|------------------------|---|
| <u><section></u> | Defines a section in a document |
| <u><summary></u> | Defines a visible heading for a <details> element |
| <u><time></u> | Defines a date/time |

30. Guia d'estil HTML

Un codi HTML coherent, net i ordenat fa que siga més fàcil per als altres llegir i entendre el vostre codi.

Ací teniu algunes directrius i consells per crear un bon codi HTML.

Declarar sempre el tipus de document

Declara sempre el tipus de document com a primera línia del document.

El tipus de document correcte per a HTML és:

`<!DOCTYPE html>`

Utilitzeu noms d'elements en minúscules

HTML permet barrejar lletres majúscules i minúscules en els noms dels elements.

Però, us recomanem que utilitzeu noms d'elements en minúscules, perquè:

- Barrejar noms en majúscules i minúscules sembla malament
- Els desenvolupadors normalment utilitzen noms en minúscula
- Les minúscules es veuen més netes
- Les minúscules són més fàcils d'escriure

Tanca tots els elements HTML

En HTML, no cal que tanqueu tots els elements (per exemple, l'element `<p>`).

Però, us recomanem fermament que tanqueu tots els elements HTML.

Utilitzeu noms d'atributs en minúscules

HTML permet barrejar lletres majúscules i minúscules en els noms dels atributs.

Bé :

`Visit our HTML tutorial`

Mal:

`Visit our HTML tutorial`

Citeu sempre els valors dels atributs

HTML permet valors d'atributs sense cometes. Però, recomanem citar els valors dels atributs, perquè:

- Els desenvolupadors solen citar valors d'atributs
- Els valors citats són més fàcils de llegir
- Heu d'utilitzar cometes si el valor conté espais

Especifiqueu sempre alt, amplada i alçada per a les imatges

Especifiqueu sempre l'atribut `alt` de les imatges. Aquest atribut és important si la imatge per algun motiu no es pot mostrar.

A més, defineix sempre el **width** i **height** de les imatges. Això redueix el parpelleig, perquè el navegador pot reservar espai per a la imatge abans de carregar-la.

Espais i signes iguals

HTML permet espais al voltant dels signes iguals. Però sense espai és més fàcil de llegir i agrupa millor les entitats.

Eviteu les línies de codi llargues

Quan utilitzeu un editor HTML, NO és convenient desplaçar-vos cap a la dreta i l'esquerra per llegir el codi HTML. Intenteu evitar línies de codi massa llargues.

Línies en blanc i sagnat

No afegiu línies, espais o sagnat en blanc sense un motiu.

Per facilitar la lectura, afegiu línies en blanc per separar blocs de codi grans o lògics.

Exemple de bona taula:

```
<table>
  <tr>
    <th>Name</th>
    <th>Description</th>
  </tr>
  <tr>
    <td>A</td>
    <td>Description of A</td>
  </tr>
  <tr>
    <td>B</td>
    <td>Description of B</td>
  </tr>
</table>
```

Exemple de bona llista:

```
<ul>
  <li>London</li>
  <li>Paris</li>
  <li>Tokyo</li>
</ul>
```

No ometeu mai l'element <title>

L'element **<title>** és obligatori en HTML.

El contingut d'un títol de pàgina és molt important per a l'optimització de motors de cerca (SEO)! Els algorismes dels motors de cerca utilitzen el títol de la pàgina per decidir l'ordre quan s'enllacen pàgines als resultats de la cerca.

L'element **<title>**:

- defineix un títol a la barra d'eines del navegador
- proporciona un títol per a la pàgina quan s'afegeix als preferits
- mostra un títol per a la pàgina als resultats del motor de cerca

Per tant, intenta que el títol siga com més precís i significatiu millor:

```
<title>HTML Style Guide and Coding Conventions</title>
```

Vols ometre <html> i <body>?

Una pàgina HTML es validarà sense les etiquetes <html> i <body>:

Però, us recomanem que afegiu sempre les etiquetes <html> i <body>!

L'omissió <body> pot produir errors en navegadors antics.

Omet <html> i <body> també pot bloquejar el programari DOM i XML.

Vols ometre <head>?

L'etiqueta HTML <head> també es pot ometre.

Els navegadors afegiran tots els elements abans de <body>, a un <head> element predeterminat.

Però, recomanem utilitzar l'etiqueta <head>.

Tanqueu els elements HTML buits?

En HTML, és opcional tancar els elements buits.

Permès:

```
<meta charset="utf-8">
```

També es permet:

```
<meta charset="utf-8" />
```

Si espereu que el programari XML/XHTML accedeisca a la vostra pàgina, manteniu la barra inclinada de tancament (/), perquè és obligatori en XML i XHTML.

Afegiu l'atribut lang

Sempre hauríeu d'incloure l'atribut **lang** dins de l'etiqueta <html>, per declarar l'idioma de la pàgina web. Això està pensat per ajudar els motors de cerca i els navegadors.

```
<!DOCTYPE html>
<html lang="en-us">
<head>
  ...
</head>
```

Metadades

Per garantir una interpretació adequada i una indexació correcta del motor de cerca, tant l'idioma com la codificació de caràcters s'han de definir en un document HTML:

```
<head>
  <meta charset="UTF-8">
  <title>Page Title</title>
</head>
```

Configuració de la finestra gràfica

La finestra gràfica és l'àrea visible de l'usuari d'una pàgina web. Varia segons el dispositiu: serà més xicoteta en un telèfon mòbil que en una pantalla d'ordinador.

Heu d'incloure l'element <meta> següent a totes les vostres pàgines web:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Comentaris HTML

Els comentaris breus s'han d'escriure en una línia, com aquesta:

```
<!-- This is a comment -->
```

Els comentaris que abasten més d'una línia s'han d'escriure així:

```
<!--  
    This is a long comment example. This is a long comment example.  
    This is a long comment example. This is a long comment example.  
-->
```

Els comentaris llargs són més fàcils d'observar si estan sagnats amb dos espais.

Ús de fulls d'estil

Utilitzeu una sintaxi senzilla per enllaçar amb fulls d'estil :

```
<link rel="stylesheet" href="styles.css">
```

Les regles CSS curtes es poden escriure comprimides, com aquesta:

```
p.intro {font-family:Verdana;font-size:16em;}
```

Les regles CSS llargues s'han d'escriure en diverses línies:

```
body {  
    background-color: lightgrey;  
    font-family: "Arial Black", Helvetica, sans-serif;  
    font-size: 16em;  
    color: black;  
}
```

- Colloqueu el suport d'obertura a la mateixa línia que el selector
- Feu servir un espai abans del suport d'obertura
- Utilitzeu dos espais de sagnat
- Utilitzeu punt i coma després de cada parell propietat-valor, inclòs l'últim
- Només utilitzeu cometes al voltant dels valors si el valor conté espais
- Colloqueu el suport de tancament en una línia nova, sense espais davanters

S'està carregant JavaScript en HTML

Utilitzeu una sintaxi senzilla per carregar scripts externs:

```
<script src="myscript.js">
```

Accés a elements HTML amb JavaScript

L'ús de codi HTML "desordenat" pot provocar errors de JavaScript.

Aquestes dues declaracions de JavaScript produiran resultats diferents:

```
getElementById("Demo").innerHTML = "Hello";  
getElementById("demo").innerHTML = "Hello";
```

Utilitzeu noms de fitxers en minúscules

Alguns servidors web (Apache, Unix) distingeixen entre majúscules i minúscules sobre els noms de fitxers: no es pot accedir a "london.jpg" com a "London.jpg".

Altres servidors web (Microsoft, IIS) no distingeixen entre majúscules i minúscules: es pot accedir a "london.jpg" com a "London.jpg".

Si feu servir una barreja de majúscules i minúscules, n'heu de ser conscient.

Si passeu d'un servidor que no distingeix entre majúscules i minúscules a un servidor, fins i tot xicotets errors trencaran la vostra web!

Per evitar aquests problemes, utilitzeu sempre noms de fitxers en minúscules!

Extensions de fitxer

Els fitxers HTML han de tenir una extensió **.html** (es permet **.htm**).

Els fitxers CSS haurien de tenir una extensió **.css**.

Els fitxers JavaScript haurien de tenir una extensió **.js**.

Diferències entre .htm i .html?

No hi ha cap diferència entre les extensions de fitxer **.htm** i **.html**!

Tots dos seran tractats com a HTML per qualsevol navegador web i servidor web.

Noms de fitxer per defecte

Quan un URL no especifica un nom de fitxer al final (com ara "<https://www.w3schools.com/>"), el servidor només afegeix un nom de fitxer predeterminat, com ara "index.html", "index.htm", "default.html", o "default.htm".

Si el vostre servidor només està configurat amb "index.html" com a nom de fitxer predeterminat, el vostre fitxer s'ha de denominar "index.html" i no "default.html".

Tanmateix, els servidors es poden configurar amb més d'un nom de fitxer predeterminat. Normalment podeu configurar tants noms de fitxer predeterminats com vulgueu.

31. Entitats HTML

Els caràcters reservats en HTML s'han de substituir per entitats de caràcters.

Entitats HTML

Alguns caràcters estan reservats en HTML.

Si feu servir els signes de menys de (<) o de major que (>) al vostre text, el navegador pot barrejar-los amb etiquetes.

Les entitats de caràcters s'utilitzen per mostrar caràcters reservats en HTML.

Una entitat de caràcter té aquest aspecte:

&entity_name;

OR

#entity_number;

Per mostrar un signe menor que (<) hem d'escriure: < or <

Avantatge d'utilitzar un nom d'entitat: un nom d'entitat és fàcil de recordar.

Desavantatge d'utilitzar un nom d'entitat: és possible que els navegadors no admeten tots els noms d'entitat, però el suport per als números d'entitat és bo.

Espai ininterromput

Una entitat que s'utilitza habitualment en HTML és l'espai ininterromput:

Un espai que no es trenca és un espai que no trencarà en una línia nova.

Dues paraules separades per un espai ininterromput s'enganxaran (no es trenquen en una línia nova). Això és útil quan trencar les paraules pot ser pertorbador. Exemple:

•10 km/h

Un altre ús comú de l'espai no trencat és evitar que els navegadors trunquen espais a les pàgines HTML.

Si escriviu 10 espais al vostre text, el navegador n'eliminarà 9. Per afegir espais reals al vostre text, podeu utilitzar el entitat de caràcter.

Algunes entitats de caràcters HTML útils

| Result Description | Entity Name | Entity Number |
|--------------------------------------|-------------|---------------|
| non-breaking space | | |
| < less than | < | < |
| > greater than | > | > |
| & ampersand | & | & |
| " double quotation mark | " | " |
| ' single quotation mark (apostrophe) | ' | ' |
| ¢ cent | ¢ | ¢ |
| £ pound | £ | £ |
| ¥ yen | ¥ | ¥ |
| € euro | € | € |
| © copyright | © | © |
| ® registered trademark | ® | ® |

Nota: els noms de les entitats distingeixen entre majúscules i minúscules.

Combinació de signes diacrítics

Un signe diacrític és un "glif" afegit a una lletra.

Alguns signes diacrítics, com greu (') i agut (") s'anomenen accents.

Els signes diacrítics poden aparèixer tant sobre com baix d'una lletra, dins d'una lletra i entre dues lletres.

Els signes diacrítics es poden utilitzar en combinació amb caràcters alfanumèrics per produir un caràcter que no està present en el conjunt de caràcters (codificació) utilitzat a la pàgina.

Aquests són alguns exemples:

| Mark Character Construct | Result |
|--------------------------|---------|
| ` a | à |
| ' a | á |
| ^ a | â |
| ~ a | ã |
| ` O | Ò |
| ' O | Ó |
| ^ O | Ô |
| ~ O | Õ |

32. Símbols HTML

Els símbols que no estan presents al teclat també es poden afegir utilitzant entitats.

Entitats de símbol HTML

Les entitats HTML es van descriure al punt anterior.

Molts símbols matemàtics, tècnics i de moneda no estan presents en un teclat normal.

Per afegir aquests símbols a una pàgina HTML, podeu utilitzar el nom de l'entitat o el número d'entitat (una referència decimal o hexadecimal) per al símbol.

Exemple: mostra el signe de l'euro, €, amb un nom d'entitat, un decimal i un valor hexadecimal.

```
<p>I will display &euro;</p>
<p>I will display &#8364;</p>
<p>I will display &#x20AC;</p>
```

Alguns símbols matemàtics compatibles amb HTML

| Char Number | Entity | Description |
|-------------|---------|-------------|
| ∀ | ∀ | ∀ |
| ∂ | ∂ | ∂ |
| ∃ | ∃ | ∃ |
| ∅ | ∅ | ∅ |
| ∇ | ∇ | ∇ |
| ∈ | ∈ | ∈ |
| ∉ | ∉ | ∉ |
| ∋ | ∋ | ∋ |
| ∏ | ∏ | ∏ |
| Σ | ∑ | ∑ |

Algunes lletres gregues compatibles amb HTML

| Char Number | Entity | Description |
|-------------|--------|-------------|
| Α | Α | Α |
| Β | Β | Β |
| Γ | Γ | Γ |
| Δ | Δ | Δ |

| | | | |
|---|--------|-----------|------------------------------|
| E | Ε | Ε | GREEK CAPITAL LETTER EPSILON |
| Z | Ζ | Ζ | GREEK CAPITAL LETTER ZETA |

Algunes altres entitats compatibles amb HTML

| Char Number | Entity | Description |
|-------------|---------|-------------|
| © | © | © |
| ® | ® | ® |
| € | € | € |
| ™ | ™ | ™ |
| ← | ← | ← |
| ↑ | ↑ | ↑ |
| → | → | → |
| ↓ | ↓ | ↓ |
| ♠ | ♠ | ♠ |
| ♣ | ♣ | ♣ |
| ♥ | ♥ | ♥ |
| ♦ | ♦ | ♦ |

33. Codificació HTML (conjunts de caràcters)

Per mostrar correctament una pàgina HTML, un navegador web ha de saber quin conjunt de caràcters utilitzar.

D'ASCII a UTF-8

ASCII va ser el primer estàndard de codificació de caràcters. ASCII va definir 128 caràcters diferents que es podrien utilitzar a Internet: números (0-9), lletres angleses (AZ) i alguns caràcters especials com ! \$ + - () @ < > .

ISO-8859-1 era el conjunt de caràcters predeterminat per a HTML 4. Aquest conjunt de caràcters admetia 256 codis de caràcters diferents. HTML 4 també suportava UTF-8.

ANSI (Windows-1252) era el conjunt de caràcters original de Windows. ANSI és idèntic a ISO-8859-1, excepte que ANSI té 32 caràcters addicionals.

L'especificació HTML5 anima els desenvolupadors web a utilitzar el conjunt de caràcters UTF-8, que cobreix gairebé tots els caràcters i símbols del món!

L'atribut del conjunt de caràcters HTML

Per mostrar correctament una pàgina HTML, un navegador web ha de conèixer el conjunt de caràcters utilitzat a la pàgina.

Això s'especifica a l'etiqueta `<meta>`:

```
<meta charset="UTF-8">
```

34. HTML versus XHTML

XHTML és una versió d'HTML més estricta i basada en XML.

Què és XHTML?

- XHTML significa E X tensible H yper T ext M arkup L anguage
- XHTML és una versió d'HTML més estricta i basada en XML

- XHTML és HTML definit com una aplicació XML
- XHTML és compatible amb tots els navegadors principals

Per què XHTML?

XML és un llenguatge de marques on tots els documents han d'estar marcats correctament (estar "ben formats").

XHTML es va desenvolupar per fer HTML més extensible i flexible per treballar amb altres formats de dades (com ara XML). A més, els navegadors ignoren els errors a les pàgines HTML i intenten mostrar el lloc web encara que tinga alguns errors en el marcantge. Així, XHTML inclou un tractament d'errors molt més estricta.

Les diferències més importants amb l'HTML

- <!DOCTYPE> és **obligatori**
- L'atribut xmlns a <html> és **obligatori**
- <html>, <head>, <title> i <body> són **obligatoris**
- Els elements sempre s'han d'**anar correctament**
- Els elements han d'estar sempre **tancats**
- Els elements han d'estar sempre en **minúscula**
- Els noms dels atributs sempre han d'estar en **minúscules**
- Els valors dels atributs sempre s'han de **citar**
- La minimització d'atributs està **prohibida**

XHTML - <!DOCTYPE> És obligatori

Un document XHTML ha de tenir una declaració XHTML <!DOCTYPE>.

Els elements <html>, <head>, <title> i <body> també han d'estar presents, i l'atribut xmlns a <html> ha d'especificar l'espai de noms xml per al document. Exemple:

Ací teniu un document XHTML amb un mínim d'etiquetes requerides:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Title of document</title>
</head>
<body>
    some content here...
</body>
</html>
```

Els elements XHTML s'han d'anar correctament imbrincats

En XHTML, els elements sempre s'han d'imbricar correctament entre si.

Els elements XHTML han d'estar sempre tancats

En XHTML, els elements sempre han d'estar tancats.

Els elements buits XHTML s'han de tancar sempre

En XHTML, els elements buits sempre s'han de tancar.

Els elements XHTML han d'estar en minúscules

En XHTML, els noms dels elements sempre han d'estar en minúscules.

Els noms dels atributs XHTML han d'estar en minúscules

En XHTML, els noms dels atributs sempre han d'estar en minúscules.

Els valors dels atributs XHTML s'han de citar

En XHTML, els valors dels atributs sempre s'han de citar.

La minimització d'atributs XHTML està prohibida

En XHTML, la minimització d'atributs està prohibida.

35. Formularis HTML

S'utilitza un formulari HTML per recollir les entrades de l'usuari. L'entrada de l'usuari s'envia sovint a un servidor per processar-la. Exemple:

The image shows a screenshot of a web page with a form. It contains two text input fields, one with the value "John" and another with "Doe". Below the inputs is a grey rectangular button labeled "Submit".

L'element <form>

L'element `<form>` HTML s'utilitza per crear un formulari HTML per a l'entrada de l'usuari:

```
<form>
  .
  form elements
  .
</form>
```

L'element `<form>` és un contenidor per a diferents tipus d'elements d'entrada, com ara: camps de text, caselles de selecció, botons d'opció, botons d'enviament, etc.

L'element <input>

L'element HTML `<input>` és l'element de formulari més utilitzat.

Un element `<input>` es pot mostrar de moltes maneres, depenent de l'atribut `type`.

Aquests són alguns exemples:

| Tipus | Descripció |
|-------|------------|
|-------|------------|

| | |
|--|---|
| <code><input type="text"></code> | Mostra un camp d'entrada de text d'una sola línia |
| <code><input type="radio"></code> | Mostra un botó d'opcio (per seleccionar una de moltes opcions) |
| <code><input type="checkbox"></code> | Mostra una casella de selecció (per seleccionar zero o més de moltes opcions) |
| <code><input type="submit"></code> | Mostra un botó d'enviament (per enviar el formulari) |
| <code><input type="button"></code> | Mostra un botó en què es pot fer clic |

Camps de text

El `<input type="text">` defineix un camp d'entrada d'una sola línia per a l'entrada de text. Exemple. Un formulari amb camps d'entrada de text:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe"><br><br>
  <input type="submit" value="Submit">
</form>
```

Així és com es mostrarà el codi HTML anterior en un navegador:

First name:

Last name:

Nota: el formulari en si no és visible. Tingueu en compte també que l'amplada predeterminada d'un camp d'entrada és de 20 caràcters.

L'element `<label>`

Observeu l'ús de l'element `<label>` a l'exemple anterior.

L'etiqueta `<label>` defineix una etiqueta per a molts elements del formulari.

L'element `<label>` és útil per als usuaris de lectors de pantalla, perquè el lector de pantalla llegirà en veu alta l'etiqueta quan l'usuari se centra en l'element d'entrada.

L'element `<label>` també ajuda els usuaris que tenen dificultats per fer clic a regions molt xicotetes (com ara botons d'opcio o caselles de verificació), perquè quan l'usuari fa clic al text de l'element `<label>`, canvia el botó d'opcio o la casella de selecció.

L'atribut `for` de l'etiqueta `<label>` ha de ser igual a l'atribut `id` de l' `<input>` element per unir-los.

Botons de ràdio

El `<input type="radio">` defineix un botó d'opcio.

Els botons d'opcio permeten a l'usuari seleccionar UNA d'un nombre limitat d'opcions.

Exemple. Un formulari amb botons d'opció. Prova el codi per vore el resultat:

```
<form>
  <input type="radio" id="html" name="fav_language" value="HTML">
  <label for="html">HTML</label><br>
  <input type="radio" id="css" name="fav_language" value="CSS">
  <label for="css">CSS</label><br>
  <input type="radio" id="js" name="fav_language" value="JavaScript">
  <label for="javascript">JavaScript</label>
</form>
```

Caselles de verificació

El `<input type="checkbox">` defineix una **casella de selecció**.

Les caselles de selecció permeten a l'usuari seleccionar ZERO o MÉS opcions d'un nombre limitat d'opcions.

Exemple. Un formulari amb caselles de selecció:

```
<form action="/action_page.php">
  <input type="checkbox" id="vehicle1" name="vehicle1" value="Bike">
  <label for="vehicle1"> I have a bike</label><br>
  <input type="checkbox" id="vehicle2" name="vehicle2" value="Car">
  <label for="vehicle2"> I have a car</label><br>
  <input type="checkbox" id="vehicle3" name="vehicle3" value="Boat">
  <label for="vehicle3"> I have a boat</label><br><br>
  <input type="submit" value="Submit">
</form>
```

El botó d'enviament

El `<input type="submit">` defineix un botó per enviar les dades del formulari a un gestor de formularis.

El gestor de formularis és normalment un fitxer al servidor amb un script per processar les dades d'entrada.

El gestor de formularis s'especifica a l'atribut `action` del formulari.

Exemple. Un formulari amb un botó d'enviament:

```
<form action="action_page.php">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe"><br><br>
  <input type="submit" value="Send">
</form>
```

L'atribut de `name` per a `<entrada>`

Tingueu en compte que cada camp d'entrada ha de tenir un atribut `name` per ser enviat.

Si s'omet l'atribut `name`, el valor del camp d'entrada no s'enviarà en absolut.

36. Atributs de formulari HTML

L'atribut action

L'atribut **action** defineix l'acció que s'ha de realitzar quan s'envia el formulari.

Normalment, les dades del formulari s'envien a un fitxer del servidor quan l'usuari fa clic al botó d'enviament.

A l'exemple següent, les dades del formulari s'envien a un fitxer anomenat "action_page.php". Aquest fitxer conté un script del servidor que gestiona les dades del formulari:

Exemple. En enviar, envieu les dades del formulari a "action_page.php":

```
<!DOCTYPE html>
<html>
<body>

<h2>HTML Forms</h2>

<form action="action_page.php">
    <label for="fname">First name:</label><br>
    <input type="text" id="fname" name="fname" value="John"><br>
    <label for="lname">Last name:</label><br>
    <input type="text" id="lname" name="lname" value="Doe"><br><br>
    <input type="submit" value="Submit">
</form>

<p>If you click the "Submit" button, the form-data will be sent to a page
called "/action_page.php".</p>

</body>
</html>
```

Consell: si s'omet l'atribut **action**, l'acció s'estableix a la pàgina actual.

L'atribut target

L'atribut **target** especifica on mostrar la resposta que es rep després d'enviar el formulari.

L'atribut **target** pot tenir un dels valors següents:

| Value | Description |
|-----------|--|
| _blank | The response is displayed in a new window or tab |
| _self | The response is displayed in the current window |
| _parent | The response is displayed in the parent frame |
| _top | The response is displayed in the full body of the window |
| framename | The response is displayed in a named iframe |

El valor predeterminat és **_self** la qual cosa significa que la resposta s'obrirà a la finestra actual.

Exemple. Ací, el resultat enviat s'obrirà en una nova pestanya del navegador:

```
<form action="action_page.php" target="_blank">
```

L'atribut del method

L'atribut **method** especifica el mètode HTTP que s'utilitzarà en enviar les dades del formulari.

Les dades del formulari es poden enviar com a variables URL (amb **method="get"**) o com a transacció posterior HTTP (amb **method="post"**).

El mètode HTTP predeterminat quan s'envien dades del formulari és GET.

Exemple. Aquest exemple utilitza el mètode GET quan s'envien les dades del formulari:

```
<form action="action_page.php" method="get">
```

Exemple. Aquest exemple utilitza el mètode POST quan s'envien les dades del formulari:

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

Notes sobre GET:

- Afegeix les dades del formulari a l'URL, en parells nom/valor
- NO utilitzeu MAI GET per enviar dades sensibles! (les dades del formulari enviat són visibles a l'URL!)
- La longitud d'un URL és limitada (2048 caràcters)
- Útil per a l'enviament de formularis on un usuari vol marcar el resultat
- GET és bo per a dades no segures, com ara les cadenes de consulta a Google

Notes sobre POST:

- Afegeix les dades del formulari dins del cos de la sol·licitud HTTP (les dades del formulari enviat no es mostren a l'URL)
- POST no té limitacions de mida i es pot utilitzar per enviar grans quantitats de dades.
- Els enviaments de formularis amb POST no es poden afegir a favorits

Consell: feu servir sempre POST si les dades del formulari contenen informació sensible o personal.

L'atribut d'autocomplete

L'atribut **autocomplete** especifica si un formulari ha d'activar o desactivar l'emplenament automàtic.

Quan l'emplenament automàtic està activat, el navegador completa automàticament els valors en funció dels valors que l'usuari haja introduït abans.

Exemple. Un formulari amb autocompletar a:

```
<form action="action_page.php" autocomplete="on">
```

L'atribut novalidate

L'atribut **novalidate** és un atribut booleà. Quan està present, especifica que les dades del formulari (entrada) no s'han de validar quan s'envien.

Exemple. Un formulari amb un atribut novalidate:

```
<form action="action_page.php" novalidate>
```

Llista de tots els atributs <form>

| Atribut | Descripció |
|-----------------------|---|
| <u>accept-charset</u> | Especifica les codificacions de caràcters utilitzades per a l'enviament del formulari |
| <u>accion</u> | Especifica on enviar les dades del formulari quan s'envii un formulari |
| <u>autocomplete</u> | Especifica si un formulari ha d'activar o desactivar l'emplenament automàtic |
| <u>enctype</u> | Especifica com s'han de codificar les dades del formulari en enviar-les al servidor (només per a method="post") |
| <u>metod</u> | Especifica el mètode HTTP que cal utilitzar quan s'envien dades de formulari |
| <u>name</u> | Especifica el nom del formulari |
| <u>novalidate</u> | Especifica que el formulari no s'ha de validar quan s'envii |
| <u>rel</u> | Especifica la relació entre un recurs enllaçat i el document actual |
| <u>Target</u> | Especifica on mostrar la resposta que es rep després d'enviar el formulari |

37. Elements del formulari HTML

L'element HTML <form> pot contenir un o més dels elements de formulari següents:

- <input>
- <label>
- <select>
- <textarea>
- <button>
- <fieldset>
- <legend>
- <output>
- <option>
- <optgroup>
- <datalist>

L'element <input>

Un dels elements de forma més utilitzats és l'element <input>.

L'element <input> es pot mostrar de diverses maneres, depenent de l'atribut **type**.

```
<label for="fname">First name:</label>
<input type="text" id="fname" name="fname">
```

Tots els diferents valors de l' **type** atribut els varem al punt següent:

L'element <label>

L'element <label> defineix una etiqueta per a diversos elements del formulari.

L'element <label> és útil per als usuaris de lectors de pantalla, perquè el lector de pantalla llegirà en veu alta l'etiqueta quan l'usuari se centre en l'element d'entrada.

L'element <label> també ajuda els usuaris que tenen dificultats per fer clic en regions molt xicotetes (com ara botons d'opció o caselles de selecció), perquè quan l'usuari fa clic al text de l'element <label>, activa el botó d'opció o la casella de selecció.

L'atribut **for** de l'etiqueta <label> ha de ser igual a l'atribut **id** de l' element <input> per unir-los.

L'element <select>

L'element <select> defineix una llista desplegable. Exemple:

```

<label for="cars">Choose a car:</label>
<select id="cars" name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>

```

L'element `<option>` defineix una opció que es pot seleccionar.

Per defecte, el primer element de la llista desplegable està seleccionat.

Per definir una opció preseleccionada, afegiu l'atribut `selected` a l'opció. Exemple:

```
<option value="fiat" selected>Fiat</option>
```

Valors visibles:

Utilitzeu l'atribut `size` per especificar el nombre de valors visibles. Exemple:

```

<label for="cars">Choose a car:</label>
<select id="cars" name="cars" size="3">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>

```

Permet seleccions múltiples:

Utilitzeu l'atribut `multiple` per permetre a l'usuari seleccionar més d'un valor. Exemple:

```

<label for="cars">Choose a car:</label>
<select id="cars" name="cars" size="4" multiple>
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>

```

L'element `<textarea>`

L'element `<textarea>` defineix un camp d'entrada de diverses línies (una àrea de text):

```

<textarea name="message" rows="10" cols="30">
The cat was playing in the garden.
</textarea>

```

L'atribut `rows` especifica el nombre visible de línies en una àrea de text.

L'atribut `cols` especifica l'amplada visible d'una àrea de text.

També podeu definir la mida de l'àrea de text mitjançant CSS. Exemple:

```

<textarea name="message" style="width:200px; height:600px;">
The cat was playing in the garden.
</textarea>

```

L'element `<button>`

L'element `<button>` defineix un botó clicable. Exemple:

```
<button type="button" onclick="alert('Hello World!')">Click Me!</button>
```

Els elements <fieldset> i <legend>

L'element <fieldset> s'utilitza per agrupar dades relacionades en un formulari.

L'element <legend> defineix un títol per a l' element <fieldset>. Exemple:

```
<form action="action_page.php">
  <fieldset>
    <legend>Personalia:</legend>
    <label for="fname">First name:</label><br>
    <input type="text" id="fname" name="fname" value="John"><br>
    <label for="lname">Last name:</label><br>
    <input type="text" id="lname" name="lname" value="Doe"><br><br>
    <input type="submit" value="Submit">
  </fieldset>
</form>
```

L'element <datalist>

L'element <datalist> especifica una llista d'opcions predefinides per a un element <input>.

Els usuaris veuran una llista desplegable de les opcions predefinides mentre introdueixen dades.

L'atribut **list** de l'element <input>, ha de fer referència a l'atribut **id** de l'element <datalist>. Exemple:

```
<form action="action_page.php">
  <input list="browsers">
  <datalist id="browsers">
    <option value="Internet Explorer">
    <option value="Firefox">
    <option value="Chrome">
    <option value="Opera">
    <option value="Safari">
  </datalist>
</form>
```

L'element <output>

L'element <output> representa el resultat d'un càlcul (com el realitzat per un script).

```
<!DOCTYPE html>
<html>
<body>

<h2>The output Element</h2>
<p>The output element represents the result of a calculation.</p>

<form action="/action_page.php"
oninput="x.value=parseInt(a.value)+parseInt(b.value)"> 0
  <input type="range" id="a" name="a" value="50"> 100 +
  <input type="number" id="b" name="b" value="50"> =
  <output name="x" for="a b"></output>
  <br><br>
  <input type="submit">
</form>

</body>
</html>
```

Elements del formulari HTML

| Tag | Description |
|-------------------------|--|
| <u><form></u> | Defines an HTML form for user input |
| <u><input></u> | Defines an input control |
| <u><textarea></u> | Defines a multiline input control (text area) |
| <u><label></u> | Defines a label for an <input> element |
| <u><fieldset></u> | Groups related elements in a form |
| <u><legend></u> | Defines a caption for a <fieldset> element |
| <u><select></u> | Defines a drop-down list |
| <u><optgroup></u> | Defines a group of related options in a drop-down list |
| <u><option></u> | Defines an option in a drop-down list |
| <u><button></u> | Defines a clickable button |
| <u><datalist></u> | Specifies a list of pre-defined options for input controls |
| <u><output></u> | Defines the result of a calculation |

38. Tipus de dades d'entrada en formularis HTML

En aquest punt es descriu els diferents tipus de l'element <input> HTML.

Tipus d'entrada HTML

Aquests són els diferents tipus d'entrada que podeu utilitzar en HTML:

- `<input type="button">`
- `<input type="checkbox">`
- `<input type="color">`
- `<input type="date">`
- `<input type="datetime-local">`
- `<input type="email">`
- `<input type="file">`
- `<input type="hidden">`
- `<input type="image">`
- `<input type="month">`
- `<input type="number">`
- `<input type="password">`
- `<input type="radio">`
- `<input type="range">`
- `<input type="reset">`
- `<input type="search">`
- `<input type="submit">`
- `<input type="tel">`
- `<input type="text">`
- `<input type="time">`
- `<input type="url">`
- `<input type="week">`

ConSELL: el valor predeterminat de l'atribut `type` és "text".

Restriccions d'entrada

Ací hi ha una llista d'algunes restriccions d'entrada habituals:

| Attribute | Description |
|------------------------|---|
| <code>checked</code> | Specifies that an input field should be pre-selected when the page loads (for <code>type="checkbox"</code> or <code>type="radio"</code>) |
| <code>disabled</code> | Specifies that an input field should be disabled |
| <code>max</code> | Specifies the maximum value for an input field |
| <code>maxlength</code> | Specifies the maximum number of character for an input field |
| <code>min</code> | Specifies the minimum value for an input field |
| <code>pattern</code> | Specifies a regular expression to check the input value against |
| <code>readonly</code> | Specifies that an input field is read only (cannot be changed) |

| | |
|----------|--|
| required | Specifies that an input field is required (must be filled out) |
| size | Specifies the width (in characters) of an input field |
| step | Specifies the legal number intervals for an input field |
| value | Specifies the default value for an input field |

39. Atributs d'entrada HTML

Aquest punt descriu els diferents atributs de l'element `<input>` HTML.

L'Atribut value

L'atribut `value` d'entrada especifica un valor inicial per a un camp d'entrada:

Exemple. Camps d'entrada amb valors inicials (per defecte):

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe">
</form>
```

L'atribut de només lectura

L'atribut `readonly` d'entrada especifica que un camp d'entrada és només de lectura.

No es pot modificar un camp d'entrada només de lectura (no obstant això, un usuari pot fer-hi tabulació, ressaltar-lo i copiar-ne el text).

El valor d'un camp d'entrada només de lectura s'enviarà en enviar el formulari!

Exemple. Un camp d'entrada de només lectura:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John" readonly><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe">
</form>
```

L'atribut desactivat

L'atribut `disabled` d'entrada especifica que s'ha de desactivar un camp d'entrada.

Un camp d'entrada desactivat és inutilitzable i no es pot fer clic.

El valor d'un camp d'entrada desactivat no s'enviarà en enviar el formulari!

Exemple. Un camp d'entrada desactivat:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John" disabled><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe">
</form>
```

L'atribut de mida

L'atribut `size` d'entrada especifica l'amplada visible, en caràcters, d'un camp d'entrada.

El valor per defecte de `size` és 20.

Nota: l'atribut `size` funciona amb els tipus d'entrada següents: text, cerca, tel, url, correu electrònic i contrasenya.

Exemple. Estableix una amplada per a un camp d'entrada:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" size="50"><br>
  <label for="pin">PIN:</label><br>
  <input type="text" id="pin" name="pin" size="4">
</form>
```

L'atribut `maxlength`

L'atribut `maxlength` d'entrada especifica el nombre màxim de caràcters permès en un camp d'entrada.

Nota: quan `maxlength` s'estableix a, el camp d'entrada no acceptarà més del nombre de caràcters especificat. Tanmateix, aquest atribut no proporciona cap comentari. Per tant, si voleu avisar l'usuari, heu d'escriure codi JavaScript.

Exemple. Estableix una longitud màxima per a un camp d'entrada:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" size="50"><br>
  <label for="pin">PIN:</label><br>
  <input type="text" id="pin" name="pin" maxlength="4" size="4">
</form>
```

Els atributs mínim i màxim

L'entrada `min` i `max` els atributs especificuen els valors mínims i màxims per a un camp d'entrada.

Els atributs `min` i `max` funcionen amb els tipus d'entrada següents: nombre, interval, data, datahora local, mes, hora i setmana.

ConSELL: utilitzeu els atributs `max` i `min` junts per crear una sèrie de valors legals.

Exemple. Estableix una data màxima, una data mínima i un interval de valors legals:

```
<form>
  <label for="datemax">Enter a date before 1980-01-01:</label>
  <input type="date" id="datemax" name="datemax" max="1979-12-31"><br><br>

  <label for="datemin">Enter a date after 2000-01-01:</label>
  <input type="date" id="datemin" name="datemin" min="2000-01-02"><br><br>

  <label for="quantity">Quantity (between 1 and 5):</label>
  <input type="number" id="quantity" name="quantity" min="1" max="5">
</form>
```

L'atribut múltiple

L'atribut `multiple` d'entrada especifica que l'usuari pot introduir més d'un valor en un camp d'entrada.

L'atribut `multiple` funciona amb els tipus d'entrada següents: correu electrònic i fitxer.

Exemple. Un camp de càrrega de fitxers que accepta diversos valors:

```
<form>
  <label for="files">Select files:</label>
  <input type="file" id="files" name="files" multiple>
</form>
```

L'atribut del pattern

L'atribut **pattern** d'entrada especifica una expressió regular amb la qual es contrasta el valor del camp d'entrada quan s'envia el formulari.

L'atribut **pattern** funciona amb els tipus d'entrada següents: text, data, cerca, url, tel, correu electrònic i contrasenya.

Exemple. Un camp d'entrada que només pot contenir tres lletres (sense números ni caràcters especials):

```
<form>
  <label for="country_code">Country code:</label>
  <input type="text" id="country_code" name="country_code"
    pattern="[A-Za-z]{3}" title="Three letter country code">
</form>
```

L'atribut de marcador de posició

L'atribut **placeholder** d'entrada especifica una pista breu que descriu el valor esperat d'un camp d'entrada (un valor de mostra o una breu descripció del format esperat).

La pista breu es mostra al camp d'entrada abans que l'usuari introduceixi un valor.

L'atribut **placeholder** funciona amb els tipus d'entrada següents: text, cerca, URL, tel, correu electrònic i contrasenya.

Exemple. Un camp d'entrada amb un text de marcador de posició:

```
<form>
  <label for="phone">Enter a phone number:</label>
  <input type="tel" id="phone" name="phone"
    placeholder="123-45-678"
    pattern="[0-9]{3}-[0-9]{2}-[0-9]{3}">
</form>
```

L'atribut requerit

L'atribut **required** d'entrada especifica que s'ha d'omplir un camp d'entrada abans d'enviar el formulari.

L'atribut **required** funciona amb els tipus d'entrada següents: text, cerca, URL, tel, correu electrònic, contrasenya, selector de dates, número, casella de selecció, ràdio i fitxer.

Exemple. Un camp d'entrada obligatori:

```
<form>
  <label for="username">Username:</label>
  <input type="text" id="username" name="username" required>
</form>
```

L'atribut de step

L'atribut **step** d'entrada especifica els intervals de nombre legal per a un camp d'entrada.

Exemple: si step="3", els números legals podrien ser -3, 0, 3, 6, etc.

Consell: aquest atribut es pot utilitzar juntament amb els atributs max i min per crear una sèrie de valors legals.

L'atribut `step` funciona amb els tipus d'entrada següents: nombre, interval, data, datahora-local, mes, hora i setmana.

Exemple. Un camp d'entrada amb un interval de nombre legal especificat:

```
<form>
  <label for="points">Points:</label>
  <input type="number" id="points" name="points" step="3">
</form>
```

les restriccions d'entrada no són infal·libles i JavaScript ofereix moltes maneres d'afegir entrades il·legals. Per restringir l'entrada de manera segura, també l'ha de comprovar el receptor (el servidor)!

L'atribut d'enfocament automàtic

L'atribut `autofocus` d'entrada especifica que un camp d'entrada hauria de centrar-se automàticament quan es carrega la pàgina.

Exemple. Permet que el camp d'entrada "Nom" es concentre automàticament quan es carregue la pàgina:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" autofocus><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```

Els atributs d'alçada i amplada

L'entrada `height` i `width` els atributs especificuen l'alçada i l'amplada d'un `<input type="image">` element.

Consell: especifiqueu sempre els atributs d'alçada i amplada de les imatges. Si s'estableixen alçada i amplada, l'espai necessari per a la imatge es reserva quan es carrega la pàgina. Sense aquests atributs, el navegador no coneix la mida de la imatge i no pot reservar-li l'espai adequat. L'efecte serà que el disseny de la pàgina canviarà durant la càrrega (mentre es carreguen les imatges).

Exemple. Definiu una imatge com a botó d'enviament, amb atributs d'alçada i amplada:

```
<form>
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <input type="image" src="img_submit.gif" alt="S" width="48" height="48">
</form>
```

L'atribut de la llista

L'atribut **list** d'entrada fa referència a un **<datalist>** element que conté opcions predefinides per a un element **<input>**.

Exemple. Un element **<input>** amb valors predefinitos en una **<datalist>**:

```
<form>
  <input list="browsers">
  <datalist id="browsers">
    <option value="Internet Explorer">
    <option value="Firefox">
    <option value="Chrome">
    <option value="Opera">
    <option value="Safari">
  </datalist>
</form>
```

L'atribut d'emplenament automàtic

L'atribut **autocomplete** d'entrada especifica si un formulari o un camp d'entrada ha d'activar o desactivar l'emplenament automàtic.

L'emplenament automàtic permet al navegador predir el valor. Quan un usuari comença a escriure un camp, el navegador hauria de mostrar opcions per omplir el camp, basant-se en els valors escrits anteriorment.

L'atribut **autocomplete** funciona amb **<form>** i els **<input>** tipus següents: text, cerca, URL, tel, correu electrònic, contrasenya, selectors de dates, rang i color.

Exemple. Un formulari HTML amb l'emplenament automàtic activat i desactivat per a un camp d'entrada:

```
<form action="/action_page.php" autocomplete="on">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <label for="email">Email:</label>
  <input type="email" id="email" name="email" autocomplete="off"><br><br>
  <input type="submit" value="Submit">
</form>
```

Tema 3. Utilització de llenguatges de marques en entorns web: CSS3



- [1. Introducció a CSS](#)
 - [2. Sintaxi CSS](#)
 - [3. Selectors CSS](#)
 - [4. Com afegir CSS](#)
 - [5. Comentaris CSS](#)
 - [6. Colors CSS](#)
 - [7. Fons CSS](#)
 - [8. Vora CSS](#)
 - [9. Marges CSS](#)
 - [10. Padding CSS](#)
 - [11. Alçada/amplada CSS](#)
 - [12. Model de caixa CSS](#)
 - [13. Contorn CSS](#)
 - [14. Text CSS](#)
 - [15. Tipus de lletra CSS](#)
 - [16. Ícones CSS](#)
 - [17. Enllaços CSS](#)
 - [18. Llistes CSS](#)
 - [19. Taules CSS](#)
 - [20. Visualització CSS](#)
 - [21. CSS Amplada màxima](#)
 - [22. Posició CSS](#)
 - [23. CSS índex Z](#)
 - [24. Desbordament de CSS](#)
 - [25. CSS flotant](#)
 - [26. Bloc en línia CSS](#)
 - [27. Alineació CSS](#)
 - [28. Combinadors CSS](#)
 - [29. Pseudoclasse CSS](#)
 - [30. Pseudoelement CSS](#)
 - [31. Opacitat CSS](#)
 - [32. Barra de navegació CSS](#)
 - [33. Menú desplegable CSS](#)
 - [34. Galeria d'imatges CSS](#)
 - [35. Selectors CSS d'atribut](#)
 - [36. Formularis CSS](#)
 - [37. Comptadors CSS](#)
 - [38. Disseny del lloc web CSS](#)
 - [39. Unitats CSS](#)
 - [40. Especificitat CSS](#)
 - [41. Funcions matemàtiques CSS](#)
-

1. Introducció a CSS

CSS és el llenguatge que fem servir per dissenyar una pàgina web.

Què és CSS?

- CSS són les sigles de Cascading Style Sheets
- CSS descriu com s'han de mostrar els elements HTML a la pantalla, al paper o en altres mitjans
- CSS estalvia molta feina. Pot controlar el disseny de diverses pàgines web alhora
- Els fulls d'estil externs s'emmagatzemen en fitxers CSS

Per què utilitzar CSS?

CSS s'utilitza per definir estils per a les vostres pàgines web, inclòs el disseny, la disposició i les variacions de visualització per a diferents dispositius i mides de pantalla. Exemple CSS:

```
body {  
    background-color: lightblue;  
}  
  
h1 {  
    color: white;  
    text-align: center;  
}  
  
p {  
    font-family: verdana;  
    font-size: 20px;  
}
```

CSS va resoldre un gran problema

L'HTML mai va tindre la intenció de contenir etiquetes per donar format a una pàgina web!

HTML es va crear per descriure el contingut d'una pàgina web, com ara:

```
<h1>Aquest és un encapçalament</h1>  
<p>Aquest és un paràgraf.</p>
```

Quan es van afegir etiquetes com i atributs de color a l'especificació HTML 3.2, va començar un malson per als desenvolupadors web. El desenvolupament de grans llocs web, on s'afegeixen fonts i informació de color a cada pàgina, es va convertir en un procés llarg i costós.

Per resoldre aquest problema, el World Wide Web Consortium (W3C) va crear CSS.

CSS ha eliminat el format d'estil de la pàgina HTML!

CSS estalvia molta feina!

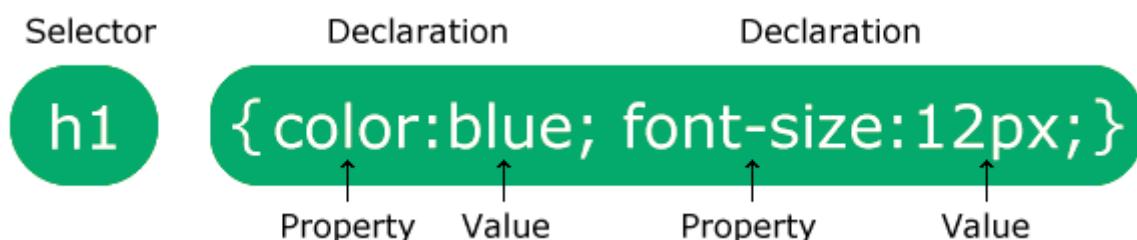
Les definicions d'estil es guarden normalment en fitxers .css externs.

Amb un fitxer de full d'estil extern, podeu canviar l'aspecte d'un lloc web sencer canviant només un fitxer!

2. Sintaxi CSS

Una regla CSS consta d'un selector i un bloc de declaració.

Sintaxi CSS



El selector apunta a l'element HTML que voleu estilitzar.

El bloc de declaracions conté una o més declaracions separades per punt i coma.

Cada declaració inclou un nom de propietat CSS i un valor, separats per dos punts.

Diverses declaracions CSS es separen amb punt i coma i els blocs de declaració estan envoltats per claus.

En aquest exemple, tots els elements <p> estaran alineats al centre, amb un color de text roig:

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

Exemple explicat

- p és un selector en CSS (apunta a l'element HTML que voleu estilitzar: <p>)
- color és una propietat, i red és el valor de la propietat
- text-align és una propietat, i center és el valor de la propietat

3. Selectors CSS

Un selector CSS selecciona els elements HTML als quals voleu estilitzar.

Selectors CSS

Els selectors CSS s'utilitzen per "trobar" (o seleccionar) els elements HTML que voleu estilitzar.

Podem dividir els selectores CSS en cinc categories:

- Selectors simples (seleccioneu elements basats en nom, identificador, classe)
- Selectors combinadors (seleccioneu elements basats en una relació específica entre ells)
- Selectors de pseudoclasse (seleccioenen elements basats en un estat determinat)
- Selectors de pseudoelements (seleccioneu i estilitzeu una part d'un element)

- Selectors d'atribut (seleccioneu elements basats en un atribut o valor d'atribut)

Explicarem els selectors CSS més bàsics.

El selector d'elements CSS

El selector d'elements selecciona elements HTML en funció del nom de l'element. Ací, tots els elements <p> de la pàgina estaran alineats al centre, amb un color de text roig:

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

El selector d'identificadors de CSS

El selector id utilitza l'atribut id d'un element HTML per seleccionar un element específic.

L'identificador d'un element és únic dins d'una pàgina, de manera que el selector d'identificació s'utilitza per seleccionar un element únic!

Per seleccionar un element amb un identificador específic, escriviu un caràcter hash (#), seguit de l'identificador de l'element.

La regla CSS següent s'aplicarà a l'element HTML amb id="para1":

```
#para1 {
    text-align: center;
    color: red;
}
```

El selector de classes CSS

El selector de classe selecciona elements HTML amb un atribut de classe específic.

Per seleccionar elements amb una classe específica, escriviu un punt (.), seguit del nom de la classe.

En aquest exemple, tots els elements HTML amb class="center" seran rojos i alineats al centre:

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

També podeu especificar que només els elements HTML específics s'hagen de veure afectats per una classe.

En aquest exemple, només els elements <p> amb class="center" seran rojos i alineats al centre:

```
p.center {
    text-align: center;
```

```
    color: red;  
}
```

El selector universal CSS

El selector universal (*) selecciona tots els elements HTML de la pàgina.

La regla CSS següent afectarà tots els elements HTML de la pàgina:

```
* {  
    text-align: center;  
    color: blue;  
}
```

El selector d'agrupació CSS

El selector d'agrupament selecciona tots els elements HTML amb les mateixes definicions d'estil.

Mireu el següent codi CSS (els elements h1, h2 i p tenen les mateixes definicions d'estil):

```
h1, h2, p {  
    text-align: center;  
    color: red;  
}
```

Tots els selectors simples de CSS

| Selector | Example | Example description |
|---------------------------|------------|---|
| <u>#id</u> | #firstname | Selects the element with id="firstname" |
| <u>.class</u> | .intro | Selects all elements with class="intro" |
| <u>element.class</u> | p.intro | Selects only <p> elements with class="intro" |
| * | * | Selects all elements |
| <u>element</u> | p | Selects all <p> elements |
| <u>element,element,..</u> | div, p | Selects all <div> elements and all <p> elements |

4. Com afegir CSS

Quan un navegador llegeix un full d'estil, formatarà el document HTML segons la informació del full d'estil.

Hi ha tres maneres d'inserir un full d'estil:

- CSS extern
- CSS intern (vist en el tema 2)
- CSS en línia (vist en el tema 2)

CSS extern

Amb un full d'estil extern, podeu canviar l'aspecte d'un lloc web sencer canviant només un fitxer!

Cada pàgina HTML ha d'incloure una referència al fitxer de full d'estil extern dins de l'element <link>, dins de la secció de capçalera.

Exemple. Els estils externs es defineixen dins de l'element <link>, dins de la secció <head> d'una pàgina HTML:

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="mystyle.css">
</head>
```

Un full d'estil extern es pot escriure en qualsevol editor de text i s'ha de desar amb una extensió .css. El fitxer .css extern no ha de contenir cap etiqueta HTML.

Així és com es veu el fitxer "mystyle.css":

```
body {
    background-color: lightblue;
}

h1 {
    color: navy;
    margin-left: 20px;
}
```

Nota: No afegiu un espai entre el valor de la propietat (20) i la unitat (px):

Incorrecte (espai): margin-left: 20 px;

Correcte (sense espai):margin-left: 20px;

Múltiplesfulls d'estil

Si s'han definit algunes propietats per al mateix selector (element) en diferents fulls d'estil, s'utilitzarà el valor de l'últim full d'estil llegit.

Si l'estil intern es defineix **després** de l'enllaç al full d'estil extern, els elements <h1> seran "taronja":

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
<style>
h1 {
    color: orange;
}
</style>
</head>
```

Tanmateix, si l'estil intern es defineix **abans** de l'enllaç al full d'estil extern, els elements <h1> seran "marí":

Ordre en cascada

Quin estil s'utilitzarà quan hi haja més d'un estil especificat per a un element HTML?

Tots els estils d'una pàgina "en cascada" en un nou full d'estil "virtual" segons les regles següents, on el número 1 té la prioritat més alta:

1. Estil en línia (dins d'un element HTML)
2. Fulls d'estil externs i interns (a la secció de capçalera)
3. Navegador predeterminat

Per tant, un estil en línia té la prioritat més alta i anul·larà els estils externs i interns i els valors predeterminats del navegador.

5. Comentaris CSS

Els comentaris CSS no es mostren al navegador, però poden ajudar a documentar el vostre codi font.

Comentaris CSS

Els comentaris s'utilitzen per explicar el codi i poden ser útils quan editeu el codi font en una data posterior. Els navegadors ignoren els comentaris.

Un comentari CSS comença `/*` i acaba amb `*/`. Exemple:

```
/* This is a single-line comment */  
p {  
    color: red;  
}
```

6. Colors CSS

Els colors s'especifiquen mitjançant noms de color predefinits o valors RGB, HEX, HSL, RGBA i HSLA.

Noms de colors CSS

En CSS, es pot especificar un color mitjançant un nom de color predefinit:

CSS/HTML admet [140 noms de colors estàndard](#)
(https://www.w3schools.com/colors/colors_names.asp).

Color de fons CSS

Podeu establir el color de fons per als elements HTML. Exemple:

```
<h1 style="background-color:DodgerBlue;">Hello World</h1>  
<p style="background-color:Tomato;">Lorem ipsum...</p>
```

Color del text CSS

Podeu definir el color del text. Exemple:

```
<h1 style="color:Tomato;">Hello World</h1>  
<p style="color:DodgerBlue;">Lorem ipsum...</p>  
<p style="color:MediumSeaGreen;">Ut wisi enim...</p>
```

Color de la vora CSS

Podeu definir el color de les vores. Exemple:

```
<h1 style="border:2px solid Tomato;">Hello World</h1>  
<h1 style="border:2px solid DodgerBlue;">Hello World</h1>  
<h1 style="border:2px solid Violet;">Hello World</h1>
```

Valors de color CSS

En CSS, els colors també es poden especificar mitjançant valors RGB, valors HEX, valors HSL, valors RGBA i valors HSLA:

* 4 exemples del color "Tomato":

rgb(255, 99, 71)

#ff6347

hsl(9, 100%, 64%)

* Ara el color "Tomato", però 50% transparent:

rgba(255, 99, 71, 0,5)

hsla(9, 100%, 64%, 0,5)

Exemple

```
<h1 style="background-color:rgb(255, 99, 71);">...</h1>
<h1 style="background-color:#ff6347;">...</h1>
<h1 style="background-color:hsl(9, 100%, 64%);">...</h1>

<h1 style="background-color:rgba(255, 99, 71, 0.5);">...</h1>
<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">...</h1>
```

7. Fons CSS

Les propietats de fons CSS s'utilitzen per afegir efectes de fons per als elements.

En aquests punt, aprendràs sobre les propietats de fons CSS següents:

- background-color
- background-image
- background-repeat
- background-attachment
- background-position
- background (proprietat de taquigrafia)

Color de fons CSS

La **background-color** propietat especifica el color de fons d'un element. El color de fons d'una pàgina s'estableix així:

```
body {
    background-color: lightblue;
}
```

Opacitat / Transparència

La propietat **opacity** especifica l'opacitat/transparència d'un element. Pot prendre un valor entre 0,0 i 1,0. Com més baix siga el valor, més transparent. Exemple:

```
div {
    background-color: green;
    opacity: 0.3;
}
```

Nota: quan s'utilitza la propietat **opacity** per afegir transparència al fons d'un element, tots els seus elements secundaris hereten la mateixa transparència. Això pot fer que el text dins d'un element totalment transparent siga difícil de llegir.

Imatge de fons CSS

El background-image és la propietat específica una imatge per utilitzar-la com a fons d'un element. Per defecte, la imatge es repeteix de manera que cobrixa tot l'element.

Exemple

Estableix la imatge de fons d'una pàgina:

```
body {  
    background-image: url("paper.gif");  
}
```

Nota: quan utilitzeu una imatge de fons, feu servir una imatge que no moleste a la lectura del text.

La imatge de fons també es pot configurar per a elements específics, com l'element <p>

Repetició de fons CSS

Per defecte, el background-imagepropietat repeteix una imatge tant horitzontalment com verticalment.

Si la imatge es repeteix només horitzontalment establirem la propietat a: background-repeat: repeat-x; per repetir una imatge verticalment, configureu background-repeat: repeat-y;

CSS background-repeat: sense repetició

Mostrar la imatge de fons només una vegada serà posar: **background-repeat: no-repeat;**

Posició de fons CSS

La propietat background-position es per a especificar la posició de la imatge de fons. Per exemple **background-position: right top;**

Adjunt de fons CSS

El background-attachmentespecifica la propietat si la imatge de fons s'ha de desplaçar o es fixa (no es desplaçarà amb el resta de la pàgina). Exemple de fixa:

```
body {  
    background-attachment: fixed;  
}
```

Exemple de no fixa:

```
body {  
    background-attachment: scroll;  
}
```

Fons CSS: propietat taquigràfica

Per acurtar el codi, també és possible especificar totes les propietats de fons en una propietat única. Això s'anomena propietat taquigràfica.

En lloc d'escriure:

```
body {  
    background-color: #ffffff;  
    background-image: url("img_tree.png");  
    background-repeat: no-repeat;  
    background-position: right top;  
}
```

Podeu utilitzar la propietat abreviatura `background`:

Exemple

Utilitzeu la propietat abreviatura per establir les propietats de fons en una declaració:

```
body {  
    background: #ffffff url("img_tree.png") no-repeat right top;  
}
```

8. Vora CSS

Les propietats de la vora CSS us permeten especificar l'estil, l'amplada i el color de la vora d'un element.

Estil de vora CSS

La propietat `border-style` especifica quin tipus de vora es mostra. Es permeten els valors següents:

- `dotted`- Defineix una vora puntejada
- `dashed`- Defineix una vora discontinua
- `solid`- Defineix una vora sòlida
- `double`- Defineix una vora doble
- `groove`- Defineix una vora ranurada en 3D.
- `ridge`- Defineix una vora encrescada en 3D.
- `inset`- Defineix una vora insertada en 3D.
- `outset`- Defineix una vora inicial 3D.
- `none`- No defineix cap frontera
- `hidden`- Defineix una vora oculta

La propietat `border-style` pot tenir d'un a quatre valors (per a la vora superior, la vora dreta, la vora inferior i la vora esquerra).

Amplada de la vora CSS

La propietat `border-width` especifica l'amplada de les quatre vores.

L'amplada es pot definir com una mida específica (en px, pt, cm, em, etc.) o utilitzant un dels tres valors predefinits: prim, mitjà o gros:

Color de la vora CSS

La propietat `border-color` s'utilitza per establir el color de les quatre vores.

Nota: si `border-color` no s'estableix, hereta el color de l'element.

Vores CSS: costats individuals

En CSS, també hi ha propietats per especificar cadascuna de les vores (superior, dreta, inferior i esquerra). Exemple:

```
p {  
    border-top-style: dotted;  
    border-right-style: solid;  
    border-bottom-style: dotted;  
    border-left-style: solid;  
}
```

Vores arrodonides CSS

La propietat `border-radius` s'utilitza per afegir vores arrodonides a un element. Exemple:

```
p {  
    border: 2px solid red;  
    border-radius: 5px;  
}
```

9. Marges CSS

Els marges s'utilitzen per crear espai al voltant dels elements, fora de les vores definides.

Marges CSS

Les propietats `margin` CSS s'utilitzen per crear espai al voltant dels elements, fora de les vores definides.

Amb CSS, teniu un control total sobre els marges. Hi ha propietats per establir el marge per a cada costat d'un element (superior, dret, inferior i esquerre).

Marge: costats individuals

CSS té propietats per especificar el marge de cada costat d'un element:

- `margin-top`
- `margin-right`
- `margin-bottom`
- `margin-left`

Totes les propietats del marge poden tenir els valors següents:

- automàtic: el navegador calcula el marge
- longitud : especifica un marge en px, pt, cm, etc.
- % - especifica un marge en % de l'amplada de l'element que conté
- inherit - especifica que el marge s'ha d'heretar de l'element pare

ConSELL: es permeten valors negatius.

Exemple. Estableix marges diferents per als quatre costats d'un element <p>:

```
p {  
    margin-top: 100px;  
    margin-bottom: 100px;  
    margin-right: 150px;  
    margin-left: 80px;  
}
```

La propietat **margin** és una propietat abreviada per a les següents propietats de marge individuals:

- **margin-top**
- **margin-right**
- **margin-bottom**
- **margin-left**

Funciona així:

Si la propietat **margin** té quatre valors:

- **marge: 25px 50px 75px 100px;**
 - el marge superior és de 25 píxels
 - el marge dret és de 50 píxels
 - el marge inferior és de 75 píxels
 - el marge esquerre és de 100 píxels

Utilitzeu la propietat abreviatura de marge amb quatre valors:

```
p {  
    margin: 25px 50px 75px 100px;  
}
```

Si la propietat **margin** té tres valors:

- **marge: 25px 50px 75px;**
 - el marge superior és de 25 píxels
 - els marges dret i esquerre són de 50 píxels
 - el marge inferior és de 75 píxels

Si la **margin** propietat té dos valors:

- **marge: 25px 50px;**
 - els marges superior i inferior són de 25 píxels
 - els marges dret i esquerre són de 50 píxels

Si la **margin** propietat té un valor:

- **marge: 25 píxels;**
 - els quatre marges són de 25 píxels

El valor automàtic

Podeu establir la propietat del marge a `auto` per centrar horitzontalment l'element dins del seu contenidor.

Aleshores, l'element ocuparà l'amplada especificada i l'espai restant es dividirà a parts iguals entre els marges esquerre i dret.

Exemple. Utilitza el marge: automàtic:

```
div {  
    width: 300px;  
    margin: auto;  
    border: 1px solid red;  
}
```

Totes les propietats del marge CSS

| Property | Description |
|--------------------------------------|---|
| <u>margin</u> | A shorthand property for setting all the margin properties in one declaration |
| <u>margin-bottom</u> | Sets the bottom margin of an element |
| <u>margin-left</u> | Sets the left margin of an element |
| <u>margin-right</u> | Sets the right margin of an element |
| <u>margin-top</u> | Sets the top margin of an element |

Col·lapse del marge

Els marges superior i inferior dels elements de vegades es redueixen en un únic marge que és igual al més gran dels dos marges.

Això no passa als marges esquerre i dret! Només els marges superior i inferior!

Mireu l'exemple següent com a demostració del col·lapse del marge:

```
h1 {  
    margin: 0 0 50px 0;  
}  
  
h2 {  
    margin: 20px 0 0 0;  
}
```

A l'exemple anterior, l'element `<h1>` té un marge inferior de 50 píxels i l'element `<h2>` té un marge superior establert en 20 píxels.

El sentit comú sembla suggerir que el marge vertical entre `<h1>` i `<h2>` seria un total de 70px (50px + 20px). Però a causa del col·lapse del marge, el marge real acaba sent de 50 píxels.

10. Padding CSS

El farciment (padding) s'utilitza per crear espai al voltant del contingut d'un element, dins de les vores definides.

Padding (Encoixinat) CSS

Amb CSS, teniu un control total sobre el farciment. Hi ha propietats per configurar el farciment per a cada costat d'un element (superior, dret, inferior i esquerre).

Encoixinat - Laterals individuals

CSS té propietats per especificar el farciment per a cada costat d'un element:

- `padding-top`
- `padding-right`
- `padding-bottom`
- `padding-left`

Totes les propietats de farciment poden tenir els valors següents:

- longitud : especifica un farciment en px, pt, cm, etc.
- % - especifica un farciment en % de l'amplada de l'element que conté
- hereta: especifica que el farciment s'ha d'heretar de l'element pare

Estableix un farciment diferent per als quatre costats d'un element <div>:

```
div {  
    padding-top: 50px;  
    padding-right: 30px;  
    padding-bottom: 50px;  
    padding-left: 80px;  
}
```

Encoixinat - propietat taquigràfica

Per a acurtar el codi, és possible especificar totes les propietats de farciment en una propietat.

La propietat `padding` és una propietat abreviada. Així doncs, ací tens com funciona:

Si la propietat `padding` té quatre valors:

- **Farciment: 25px 50px 75px 100px;**
 - el farciment superior és de 25 píxels
 - el farciment dret és de 50 píxels
 - el farciment inferior és de 75 píxels
 - el farciment esquerre és de 100 píxels

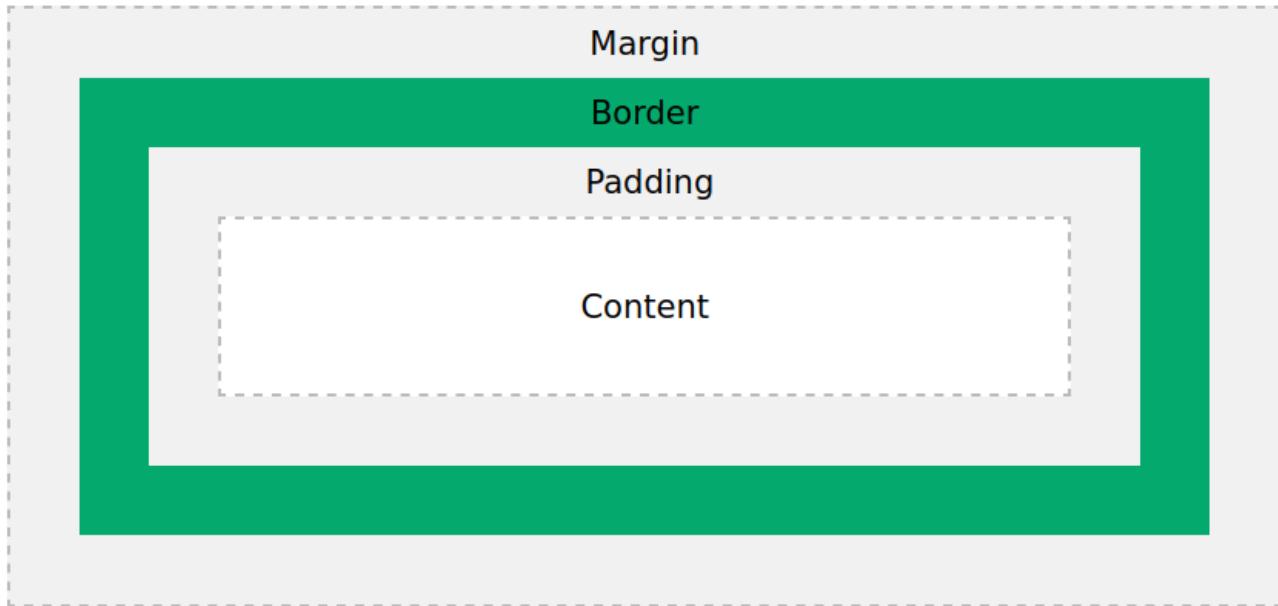
Utilitzeu la propietat d'abreviatura de farciment amb quatre valors:

```
div {  
    padding: 25px 50px 75px 100px;  
}
```

Si la propietat `padding` té tres, dos, un.. Serà igual que amb el margin.

Encoixinat i amplada de l'element

La propietat **width** CSS especifica l'amplada de l'àrea de contingut de l'element. L'àrea de contingut és la part dins del farciment, la vora i el marge d'un element (model de caixa).



Per tant, si un element té una amplada especificada, he de sumar-li margin, border i padding per a saber l'amplada del contingut. Un exemple:

```
div {  
    width: 300px;  
    padding: 25px;  
}
```

Ací, l'element <div> té una amplada de 300 píxels. Tanmateix, l'amplada real de l'element <div> serà de 350px (300px + 25px de farciment esquerre + 25px de farciment dret)

Per mantenir l'amplada a 300 píxels, sense importar la quantitat de farciment, podeu utilitzar la propietat **box-sizing**. Això fa que l'element mantinga la seva amplada real. Si augmenteu el farciment, aleshores, l'espai de contingut disponible disminuirà.

Exemple: Utilitzeu la propietat **box-sizing** de la caixa per mantenir l'amplada a 300 píxels, independentment de la quantitat de farciment:

```
div {  
    width: 300px;  
    padding: 25px;  
    box-sizing: border-box;  
}
```

11. Alçada, amplada i amplada màxima CSS

Les propietats CSS **height** i **width** s'utilitzen per establir l'alçada i l'amplada d'un element.

La propietat **max-width** CSS s'utilitza per establir l'amplada màxima d'un element.

CSS Configuració d'alçada i amplada

Les propietats d'alçada i amplada no inclouen el farciment, les vores ni els marges. Estableix l'alçada/amplada de l'àrea dins del farciment, la vora i el marge de l'element.

Valors d'alçada i amplada CSS

Les propietats **height** i **width** poden tenir els valors següents:

- **auto**- Això és per defecte. El navegador calcula l'alçada i l'amplada
- **length**- Defineix l'alçada/amplada en px, cm, etc.
- **%**- Defineix l'alçada/amplada en percentatge del bloc que conté
- **initial**- Estableix l'alçada/amplada al seu valor predeterminat
- **inherit**- L'alçada/amplada s'heretarà del seu valor principal

Exemple: Estableix l'alçada i l'amplada d'un element <div>

```
div {  
    height: 200px;  
    width: 50%;  
    background-color: powderblue;  
}
```

Configuració de l'amplada màxima

La propietat **max-width** s'utilitza per establir l'amplada màxima d'un element.

Es pot especificar **max-width** en valors de longitud, com ara px, cm, etc., o en percentatge (%) del bloc que conté, o establir-se en cap (això és per defecte. Significa que no hi ha una amplada màxima).

El problema ve quan la finestra del navegador és més xicoteta que l'amplada de l'element. Aleshores, el navegador afegeix una barra de desplaçament horitzontal a la pàgina.

En canvi, l'ús **max-width**, en aquesta situació, millorarà el maneig del navegador de finestres menudes.

Exemple. Aquest element <div> té una alçada de 100 píxels i una amplada màxima de 500 píxels:

```
div {  
    max-width: 500px;  
    height: 100px;  
    background-color: powderblue;  
}
```

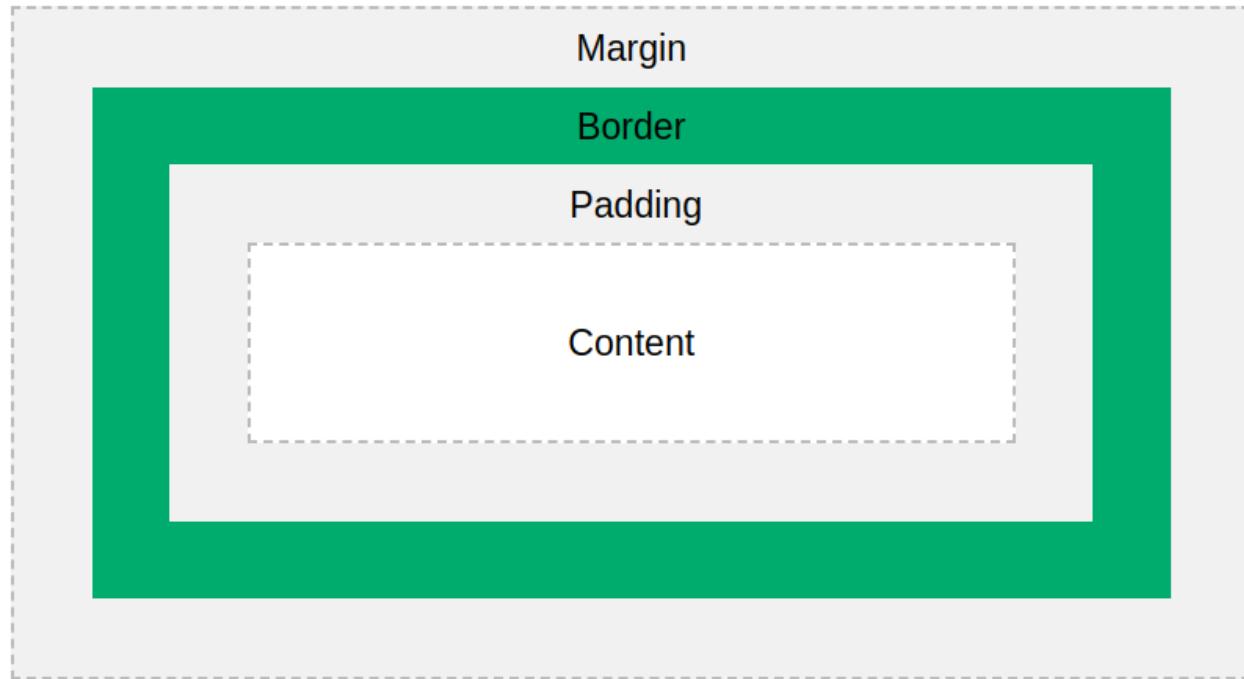
12. Model de caixa CSS

Tots els elements HTML es poden considerar com a quadres.

El model de caixa CSS

En CSS, el terme "model de caixa" s'utilitza quan es parla de disseny i maquetació.

El model de caixa CSS és essencialment una caixa que envolta tots els elements HTML. Consta de: marges, vores, farciment i el contingut real. La imatge següent il·lustra el model de caixa:



Explicació de les diferents parts:

- **Contingut**: el contingut del quadre, on apareixen text i imatges
- **Encoixinat**: esborra una àrea al voltant del contingut. El farciment és transparent
- **Frontera**: una vora que envolta el farciment i el contingut
- **Marge**: neteja una àrea fora de la frontera. El marge és transparent

El model de caixa ens permet afegir una vora al voltant dels elements i definir l'espai entre elements. Exemple:

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
    background-color: lightgrey;
    width: 400px;
    border: 15px solid green;
    padding: 50px;
    margin: 40px;
}
</style>
</head>
<body>

<h2>Demonstrating the Box Model</h2>

<p>The CSS box model is essentially a box that wraps around every HTML element. It consists of: borders, padding, margins, and the actual content.</p>

</body>
</html>
```

Amplada i alçada d'un element

Per configurar correctament l'amplada i l'alçada d'un element en tots els navegadors, cal saber com funciona el model de caixa.

Important: quan configureu les propietats d'amplada i alçada d'un element amb CSS és només l'amplada i l'alçada de l'àrea de contingut. Per calcular la mida completa d'un element, també heu d'afegir farciment, vores i marges.

Aquest element <div> tindrà una amplada total de 350 píxels:

```
div {  
    width: 320px;  
    padding: 10px;  
    border: 5px solid gray;  
    margin: 0;  
}
```

Aquí teniu el càlcul:

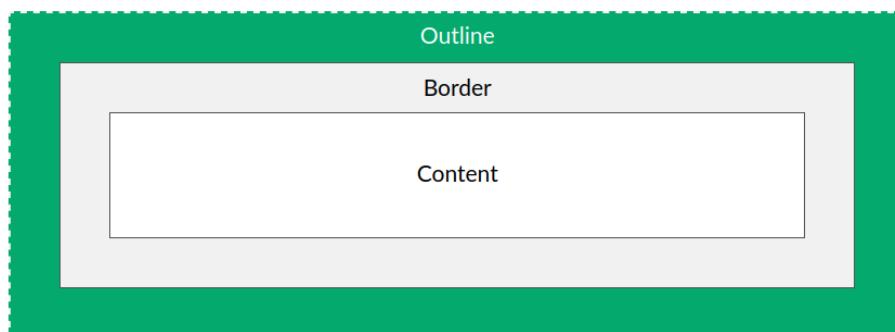
320 px (amplada) + 20 px (esquerre + farciment dret) + 10 px (bord esquerre + dret) + 0

px (marge esquerre + dret) = **350 px**

13. Contorn CSS

Contorn CSS

Un contorn és una línia que es dibuixa al voltant dels elements, FORA de les vores, per fer que l'element "destaque".



CSS té les següents propietats de contorn:

- `outline-style`
- `outline-color`
- `outline-width`
- `outline-offset`
- `outline`

Nota: el contorn difereix de les vores! A diferència de la vora, el contorn es dibuixa fora de la vora de l'element i pot superposar-se a altres continguts. A més, el contorn NO

forma part de les dimensions de l'element; l'amplada i l'alçada totals de l'element no es veuen afectades per l'amplada del contorn.

Esquema CSS: propietat taquigràfica

La propietat **outline** és una propietat abreviada per establir les següents propietats d'esquema individuals:

- **outline-width**
- **outline-style**
- **outline-color**

Exemple

```
p.ex1 {outline: dashed;}
p.ex2 {outline: dotted red;}
p.ex3 {outline: 5px solid yellow;}
p.ex4 {outline: thick ridge pink;}
```

outline-offset

La propietat **outline-offset** afegeix espai entre un contorn i la vora d'un element. L'espai entre un element i el seu contorn és transparent.

L'exemple següent especifica un contorn de 15 píxels fora de la vora: Aquest paràgraf té un contorn de 15 píxels fora de la vora. Exemple:

```
p {
    margin-top: 100px;
    margin-bottom: 100px;
    margin-right: 150px;
    margin-left: 80px;
}
```

14. Text CSS

CSS té moltes propietats per donar format al text.

Color del text i color de fons

En aquest exemple, definim tant la propietat **background-color** com la propietat **color**

```
body {
    background-color: lightgrey;
    color: blue;
}
```

Alineació de text

La propietat **text-align** s'utilitza per establir l'alignació horitzontal d'un text. Un text pot estar alineat a l'esquerra o a la dreta, centrat o justificat.

L'exemple següent mostra el text alineat al centre i alineat a l'esquerra i a la dreta (l'alignació a l'esquerra és per defecte si la direcció del text és d'esquerra a dreta, i l'alignació dreta és per defecte si la direcció del text és de dreta a esquerra):

```
h1{text-align: center;}
h2{text-align: left;}
h3{text-align: right;}
```

Quan la propietat `text-align` s'estableix com a "justificar", cada línia s'estira de manera que cada línia tinga la mateixa amplada i els marges esquerre i dret siguen rectes (com les revistes i diaris):

Alinea el text darrera línia

La propietat `text-align-last` especifica com alinear l'última línia d'un text.

Direcció del text

Les propietats `direction` i `unicode-bidi` es poden utilitzar per a canviar la direcció del text d'un element. Exemple:

```
p {  
    direction: rtl;  
    unicode-bidi: bidi-override;  
}
```

Alineació vertical

La propietat `vertical-align` estableix l'alineació vertical d'un element.

Aquest exemple estableix l'alineació vertical d'una imatge en un text, pots provar-ho:

```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
    img.a { vertical-align: baseline; }  
    img.b { vertical-align: text-top; }  
    img.c { vertical-align: text-bottom; }  
    img.d { vertical-align: sub; }  
    img.e { vertical-align: super; }  
</style>  
</head>  
<body>  
  
<h1>The vertical-align Property</h1>  
  
<h2>vertical-align: baseline (default):</h2>  
<p>An  image with a  
default alignment.</p>  
  
<h2>vertical-align: text-top:</h2>  
<p>An  image with a  
text-top alignment.</p>  
  
<h2>vertical-align: text-bottom:</h2>  
<p>An  image with a  
text-bottom alignment.</p>  
  
<h2>vertical-align: sub:</h2>  
<p>An  image with a  
sub alignment.</p>  
  
<h2>vertical-align: sup:</h2>  
<p>An  image with a  
super alignment.</p>
```

```
</body>
</html>
```

Les propietats d'alignació/direcció del text CSS

| Property | Description |
|---------------------------------|---|
| direction | Specifies the text direction/writing direction |
| text-align | Specifies the horizontal alignment of text |
| text-align-last | Specifies how to align the last line of a text |
| unicode-bidi | Used together with the direction property to set or return whether the text should be overridden to support multiple languages in the same document |
| vertical-align | Sets the vertical alignment of an element |

Decoració de text CSS

En aquest apartat coneixeràs les propietats següents:

- [text-decoration-line](#)
- [text-decoration-color](#)
- [text-decoration-style](#)
- [text-decoration-thickness](#)
- [text-decoration](#)

Afegeiu una línia de decoració al text

La propietat [text-decoration-line](#) s'utilitza per afegir una línia de decoració al text.

Consell: podeu combinar més d'un valor, com ara el sobre i el subratllat, per mostrar línies tant per dalt com per baix d'un text. Exemple:

```
h1 {text-decoration-line: overline;}
h2 {text-decoration-line: line-through;}
h3 {text-decoration-line: underline;}
p {text-decoration-line: overline underline;}
```

Nota: no es recomana subratllar el text que no siga un enllaç, ja que això sovint confon el lector.

Especifiqueu un color per a la línia de decoració

La propietat [text-decoration-color](#) s'utilitza per establir el color de la línia de decoració.

```
h1 {
  text-decoration-line: overline;
  text-decoration-color: red;
}
```

Especifiqueu un estil per a la línia de decoració

La propietat [text-decoration-style](#) s'utilitza per definir l'estil de la línia de decoració.

```
h1 {
  text-decoration-line: underline;
  text-decoration-style: solid;
  text-decoration-thickness: 5px;
}
```

La propietat taquigràfica

La propietat `text-decoration` és una propietat taquigràfica per a:

- `text-decoration-line` (obligatori)
- `text-decoration-color` (opcional)
- `text-decoration-style` (opcional)
- `text-decoration-thickness` (opcional)

Exemples:

```
h1 {text-decoration: underline;}  
h2 {text-decoration: underline red;}  
h3 {text-decoration: underline red double;}  
p {text-decoration: underline red double 5px;}
```

Un consell: Tots els enllaços en HTML estan subratllats per defecte. De vegades veus que els enllaços tenen un estil sense subratllat. S'utilitza `text-decoration: none;` per eliminar el subratllat dels enllaços.

Totes les propietats de decoració de text CSS

| Property | Description |
|--|--|
| <code>text-decoration</code> | Sets all the text-decoration properties in one declaration |
| <code>text-decoration-color</code> | Specifies the color of the text-decoration |
| <code>text-decoration-line</code> | Specifies the kind of text decoration to be used (underline, overline, etc.) |
| <code>text-decoration-style</code> | Specifies the style of the text decoration (solid, dotted, etc.) |
| <code>text-decoration-thickness</code> | Specifies the thickness of the text decoration line |

Transformació de textos

La propietat `text-transform` s'utilitza per especificar lletres majúscules i minúscules en un text.

Es pot utilitzar per convertir-ho tot en lletres majúscules o minúscules, o posar en majúscula la primera lletra de cada paraula. Exemple:

```
p.uppercase {text-transform: uppercase;}  
p.lowercase {text-transform: lowercase;}  
p.capitalize {text-transform: capitalize;}
```

Espaiat del text

En aquest apartat tenim les propietats següents:

- `text-indent`
- `letter-spacing`
- `line-height`
- `word-spacing`
- `white-space`

Sagnat de text

La propietat `text-indent` s'utilitza per especificar el sagnat de la primera línia d'un text:

```
p {text-indent: 50px;}
```

Espaiat entre lletres

La propietat `letter-spacing` s'utilitza per especificar l'espai entre els caràcters d'un text.

L'exemple següent mostra com augmentar o reduir l'espai entre caràcters. Exemple:

```
h1 {letter-spacing: 5px;}  
h2 {letter-spacing: -2px;}
```

Alçada de la línia

La propietat `line-height` s'utilitza per especificar l'espai entre línies. Exemple:

```
p.small {line-height: 0.8;}  
p.big {line-height: 1.8;}
```

Espaiat de paraules

La propietat `word-spacing` s'utilitza per especificar l'espai entre les paraules d'un text.

L'exemple següent mostra com augmentar o disminuir l'espai entre paraules. Exemple:

```
p.one {word-spacing: 10px;}  
p.two {word-spacing: -2px;}
```

Les propietats d'espaiat de text CSS

| Property | Description |
|-----------------------------|---|
| <code>letter-spacing</code> | Specifies the space between characters in a text |
| <code>line-height</code> | Specifies the line height |
| <code>text-indent</code> | Specifies the indentation of the first line in a text-block |
| <code>white-space</code> | Specifies how to handle white-space inside an element |
| <code>word-spacing</code> | Specifies the space between words in a text |

Ombra de text

La propietat `text-shadow` afegeix ombra al text.

En el seu ús més senzill, només especifiqueu l'ombra horitzontal (2px) i l'ombra vertical (2px):

Efecte d'ombra de text! Exemple:

```
h1 {text-shadow: 2px 2px;}
```

A continuació, afegiu un color (roig) a l'ombra. Efecte d'ombra de text!

```
h1 {text-shadow: 2px 2px red;}
```

15. Tipus de lletra CSS

És important triar el tipus de lletra adequat per al vostre lloc web!

Escollir el tipus de lletra adequat té un gran impacte en la manera com els lectors experimenten un lloc web. El tipus de lletra adequat pot crear una identitat forta per a la vostra marca.

És important utilitzar un tipus de lletra fàcil de llegir. El tipus de lletra afegeix valor al vostre text. També és important triar el color i la mida del text correctes per a la font.

Famílies de fonts genèriques

En CSS hi ha cinc famílies de tipus de lletra genèriques:

1. **Els tipus de lletra serif** tenen un xicotet traç a les vores de cada lletra. Creen una sensació de formalitat i elegància.
2. **Els tipus de lletra sans-serif** tenen línies netes (no hi ha traços xicotets). Creen un aspecte modern i minimalist.
3. **Tipus de lletra monospace**: ací totes les lletres tenen la mateixa amplada fixa. Creen un aspecte mecànic.
4. **Els tipus de lletra cursive** imiten l'escriptura humana.
5. **Els tipus de lletra de fantasy** són tipus de lletra decoratius o lúdics.

Tots els diferents noms de tipus de lletra pertanyen a una de les famílies de tipus de lletra genèriques.

Diferència entre tipus de lletra Serif i Sans-serif



Nota: a les pantalles d'ordinador, els tipus de lletra sans-serif es consideren més fàcils de llegir que els tipus de lletra serif.

Alguns exemples de fonts

| Generic Font Family | Examples of Font Names |
|---------------------|---|
| Serif | Times New Roman Georgia Garamond Arial |
| Sans-serif | Verdana Helvetica |
| Monospace | Courier New Lucida Console Monaco |
| Cursive | Brush Script MT Lucida Handwriting |
| Fantasy | Copperplate Papyrus |

La propietat de la família de fonts CSS

En CSS, utilitzem la propietat **font-family** per especificar el tipus de lletra d'un text.

Nota: si el nom del tipus de lletra és més d'una paraula, ha d'estar entre cometes, com ara: "Times New Roman".

Consell: la propietat **font-family** hauria de contenir diversos noms de font com a sistema "de reserva", per garantir la màxima compatibilitat entre navegadors/sistemes operatius. Comenceu amb el tipus de lletra que vulgueu i acabeu amb una família genèrica (per permetre que el navegador trieu un tipus de lletra similar a la família genèrica, si no hi ha altres tipus de lletra disponibles). Els noms dels tipus de lletra s'han de separar amb comes. Exemple:

```
.p1 {font-family: "Times New Roman", Times, serif;}  
.p2 {font-family: Arial, Helvetica, sans-serif;}  
.p3 {font-family: "Lucida Console", "Courier New", monospace;}
```

Tipus de lletra web CSS segurs

Què són els tipus de lletra web segurs?

Els tipus de lletra web segurs són tipus de lletra que s'instal·len de manera universal a tots els navegadors i dispositius.

Els millors tipus de lletra web segurs per a HTML i CSS

La llista següent són les millors fonts segures web per a HTML i CSS:

- Arial (sense serif)
- Verdana (sans-serif)
- Tahoma (sense serif)
- Trebuchet MS (sense serif)
- Times New Roman (serif)
- Geòrgia (serif)
- Garamond (serif)
- Courier New (monoespai)
- Brush Script MT (cursiu)

Nota: abans de publicar el vostre lloc web, comproveu sempre com apareixen els vostres tipus de lletra en diferents navegadors i dispositius, i feu servir sempre tipus de lletra alternatius.

Estil de lletra CSS

La propietat **font-style** s'utilitza principalment per especificar text en cursiva.

Aquesta propietat té tres valors:

- normal: el text es mostra amb normalitat

- cursiva: el text es mostra en cursiva
- oblic: el text és "inclinat" (l'oblic és molt semblant a la cursiva, però amb menys suport)

```
p.normal {font-style: normal;}
p.italic {font-style: italic;}
p.oblique {font-style: oblique;}
```

Pes de la lletra

La propietat **font-weight** especifica el pes d'una font. Exemple:

```
p.normal {font-weight: normal;}
p.thick {font-weight: bold;}
```

Font Variant

La propietat **font-variant** especifica si un text s'ha de mostrar o no amb un tipus de lletra en majúscules xicotetes.

En un tipus de lletra amb majúscules xicotetes, totes les lletres minúscules es converteixen en lletres majúscules. Tanmateix, les lletres majúscules convertides apareixen amb una mida de lletra més xicoteta que les lletres majúscules originals al text. Exemple:

```
p.normal {font-variant: normal;}
p.small {font-variant: small-caps;}
```

Mida del tipus de lletra CSS

La propietat **font-size** estableix la mida del text.

Ser capaç de gestionar la mida del text és important en el disseny web. Tanmateix, no hauríeu d'utilitzar ajustaments de mida de lletra perquè els paràgrafs semblen encapçalaments o els encapçalaments semblen paràgrafs.

Feu servir sempre les etiquetes HTML adequades, com ara **<h1>** - **<h6>** per als encapçalaments i **<p>** per als paràgrafs.

El valor de la mida de la font pot ser una mida absoluta o relativa.

Mida absoluta:

- Estableix el text a una mida especificada
- No permet que un usuari canvie la mida del text en tots els navegadors (mal per raons d'accessibilitat)
- La mida absoluta és útil quan es coneix la mida física de l'eixida

Mida relativa:

- Estableix la mida relativa als elements circumdants
- Permet a un usuari canviar la mida del text als navegadors

Nota: si no especifiqueu una mida de lletra, la mida predeterminada del text normal, com ara els paràgrafs, és de 16 píxels (16 px=1 em).

Establir la mida del text amb píxels us ofereix un control total sobre la mida del text. Exemple:

```
h1 {font-size: 40px;}  
h2 {font-size: 30px;}  
p {font-size: 14px;}
```

Consell: si feu servir píxels, encara podeu utilitzar l'eina de zoom per canviar la mida de la pàgina sencera.

Estableix la mida de la lletra amb Em

Per permetre als usuaris canviar la mida del text (al menú del navegador), molts desenvolupadors utilitzen em en lloc de píxels.

1em és igual a la mida de lletra actual. La mida del text predeterminada als navegadors és de 16 píxels. Per tant, la mida predeterminada d'1 em és de 16 píxels.

La mida es pot calcular de píxels a em utilitzant aquesta fórmula: $\text{píxels} / 16 = \text{em}$

```
h1 {font-size: 2.5em; /* 40px/16=2.5em */}  
h2 {font-size: 1.875em; /* 30px/16=1.875em */}  
p {font-size: 0.875em; /* 14px/16=0.875em */}
```

A l'exemple anterior, la mida del text en em és la mateixa que l'exemple anterior en píxels. Tanmateix, amb la mida em, és possible ajustar la mida del text en tots els navegadors.

Utilitzeu una combinació de Percentatge i Em

La solució que funciona en tots els navegadors és establir una mida de font predeterminada en percentatge per a l'element <body>. Exemple:

```
body {font-size: 100%;}  
h1 {font-size: 2.5em;}  
h2 {font-size: 1.875em;}  
p {font-size: 0.875em;}
```

El nostre codi ara funciona molt bé! Mostra la mateixa mida de text a tots els navegadors i permet a tots els navegadors fer zoom o canviar la mida del text!

Mida de lletra sensible

La mida del text es pot definir amb una unitat **vw**, que significa "amplada de la finestra gràfica".

D'aquesta manera, la mida del text seguirà la mida de la finestra del navegador. Canvieu la mida de la finestra del navegador per veure com s'escala la mida de la lletra:

```
<h1 style="font-size:10vw">Hello World</h1>
```

Viewport és la mida de la finestra del navegador. 1vw = 1% de l'amplada de la finestra visual. Si el visor té 50 cm d'ample, 1vw és 0,5 cm.

CSS Google Fonts

Si no voleu utilitzar cap tipus de lletra estàndard en HTML, podeu utilitzar Google Fonts.

Les fonts de Google són gratuïtes i tenen més de 1000 tipus de lletra per triar.

Com utilitzar Google Fonts

Només cal que afegiu un enllaç especial al full d'estil a la secció <head> i, a continuació, consulteu el tipus de lletra al CSS. Ací, volem utilitzar una font anomenada "Sofia" de Google Fonts:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Sofia">
<style>
body {
    font-family: "Sofia", sans-serif;
}
</style>
</head>
```

Utilitzeu diversos tipus de lletra de Google

Per utilitzar diversos tipus de lletra de Google, només cal que separeu els noms dels tipus de lletra amb un caràcter pipe (|), així:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Audiowide|Sofia|Trirong">
<style>
h1.a {font-family: "Audiowide", sans-serif;}
h1.b {font-family: "Sofia", sans-serif;}
h1.c {font-family: "Trirong", serif;}
</style>
</head>
```

Nota: sol·licitar diversos tipus de lletra pot alentir les vostres pàgines web! Així que aneu amb compte amb això.

La propietat taquigràfica del tipus de lletra CSS

Per a compactar el codi, també és possible especificar totes les propietats de tipus de lletra individuals en una propietat.

La propietat **Font** és una propietat taquigràfica per a:

- **font-style**
- **font-variant**
- **font-weight**
- **font-size/line-height**
- **font-family**

Nota: els valors **font-size** i **font-family** són obligatoris. Si falta un dels altres valors, s'utilitza el seu valor per defecte. Exemple:

```
p {
    font: 20px Arial, sans-serif;
}
```

Totes les propietats del tipus de lletra CSS

| Property | Description |
|--------------------|---|
| <u>font</u> | Sets all the font properties in one declaration |
| <u>font-family</u> | Specifies the font family for text |
| <u>font-size</u> | Specifies the font size of text |

| | |
|---------------------|--|
| <u>font-style</u> | Specifies the font style for text |
| <u>font-variant</u> | Specifies whether or not a text should be displayed in a small-caps font |
| <u>font-weight</u> | Specifies the weight of a font |

16. Icons CSS

Les icones es poden afegir fàcilment a la vostra pàgina HTML mitjançant una biblioteca d'icônes.

Com afegir icônes de Google

La manera més senzilla d'afegir una icôna a la vostra pàgina HTML anar a icons de Google. Busqueu la icôna a agregar i trieu mida i color.



Podeu copiar i apear el codi corresponent o descarregar el svg o png i insertar la imatge. També teniu **Icônes de Font Awesome** en fontawesome.com o Bootstrap icon library.

17. Enllaços CSS

Amb CSS, els enllaços es poden dissenyar de moltes maneres diferents.

Estil d'enllaços

Els enllaços es poden dissenyar amb qualsevol propietat CSS (per exemple **color**, **font-family**, **background**, etc.). Exemple:

```
a {
  color: hotpink;
}
```

A més, els enllaços poden tenir un estil diferent en funció de l'**estat** en què es troben. Els quatre estats d'enllaç són:

- **a:link**- un enllaç normal i no visitat
- **a:visited**- un enllaç que l'usuari ha visitat
- **a:hover**- un enllaç quan l'usuari passa el ratolí per sobre
- **a:active**- un enllaç en el moment en què es fa clic

Exemple

```
/* unvisited link */
a:link {
  color: red;
```

```

}

/* visited link */
a:visited {
  color: green;
}

/* mouse over link */
a:hover {
  color: hotpink;
}

/* selected link */
a:active {
  color: blue;
}

```

Quan s'estableix l'estil per a diversos estats d'enllaç, hi ha algunes regles d'ordre:

- a:hover ha d'anar després de a:link i a:visited
- a:active ha d'anar després de a:hover

Decoració de text

La propietat **text-decoration** s'utilitza principalment per eliminar els subratllats dels enllaços. Exemple:

```
a:link {
  text-decoration: none;
}
```

Color de fons

La propietat **background-color** es pot utilitzar per especificar un color de fons per als enllaços. Exemple

```
a:link {
  background-color: yellow;
}
```

Aquest exemple mostra els **diferents tipus de cursors** (pot ser útil per als enllaços):

```
<!DOCTYPE html>
<html>
<body>

<h2>The cursor Property</h2>

<p>Mouse over the words to change the cursor.</p>
<span style="cursor:auto">auto</span><br>
<span style="cursor:crosshair">crosshair</span><br>
<span style="cursor:default">default</span><br>
<span style="cursor:e-resize">e-resize</span><br>
<span style="cursor:help">help</span><br>
<span style="cursor:move">move</span><br>
<span style="cursor:n-resize">n-resize</span><br>
<span style="cursor:ne-resize">ne-resize</span><br>
<span style="cursor:nw-resize">nw-resize</span><br>
<span style="cursor:pointer">pointer</span><br>
<span style="cursor:progress">progress</span><br>
<span style="cursor:s-resize">s-resize</span><br>
```

```

<span style="cursor:se-resize">se-resize</span><br>
<span style="cursor:sw-resize">sw-resize</span><br>
<span style="cursor:text">text</span><br>
<span style="cursor:w-resize">w-resize</span><br>
<span style="cursor:wait">wait</span><br>

</body>
</html>

```

18. Llistes CSS

Llistes no ordenades:

- Cafè
- Té
- Coca Cola

- Cafè
- Té
- Coca Cola

Llistes ordenades:

1. Cafè
2. Té
3. Coca Cola

- I. Cafè
- II. Té
- III. Coca Cola

Llistes HTML i propietats de llista CSS

En HTML, hi ha dos tipus principals de llistes:

- llistes no ordenades (): els elements de la llista estan marcats amb vinyetes
- llistes ordenades (): els elements de la llista estan marcats amb números o lletres

Les propietats de la llista CSS us permeten:

- Estableix diferents marcadors d'elements de llista per a llistes ordenades
- Estableix diferents marcadors d'elements de llista per a llistes no ordenades
- Estableix una imatge com a marcador d'elements de la llista
- Afegiu colors de fons a llistes i elements de llista

Diferents marcadors d'elements de llista

La propietat **list-style-type** especifica el tipus de marcador d'element de llista.

L'exemple següent mostra alguns dels marcadors d'elements de llista disponibles:

```

ul.a {
    list-style-type: circle;
}

ul.b {
    list-style-type: square;
}

ol.c {
    list-style-type: upper-roman;
}

```

```
}

ol.d {
    list-style-type: lower-alpha;
}
```

Nota: alguns dels valors són per a llistes no ordenades i altres per a llistes ordenades.

Una imatge com a marcador d'elements de llista

La propietat `list-style-image` especifica una imatge com a marcador d'element de llista:

```
ul {
    list-style-image: url('sqpurple.gif');
}
```

Elimina la configuració predeterminada

La propietat `list-style-type:none` també es pot utilitzar per eliminar els bullets. Tingueu en compte que la llista també té marge i farciment predeterminats. Per eliminar-ho, afegiu `margin:0` i `padding:0` a `` o ``. Exemple:

```
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
}
```

Llista - Propietat taquigràfica

La propietat `list-style` és una propietat taquigràfica. S'utilitza per establir totes les propietats de la llista en una declaració. Exemple:

```
ul {
    list-style: square inside url("sqpurple.gif");
}
```

Quan s'utilitza la propietat abreviatura, l'ordre dels valors de la propietat és:

- `list-style-type` (si s'especifica una imatge d'estil de llista, el valor d'aquesta propietat es mostrarà si la imatge per algun motiu no es pot mostrar)
- `list-style-position` (especifica si els marcadors d'elements de la llista han d'aparèixer dins o fora del flux de contingut)
- `list-style-image` (especifica una imatge com a marcador d'element de la llista)

Si falta un dels valors de propietat anteriors, s'inserirà el valor predeterminat de la propietat que falta, si n'hi ha.

Llista d'estils per a colors

També podem estilitzar llistes amb colors, perquè semblen més interessants.

Qualsevol cosa que s'afegeix a l'etiqueta `` o `` afecta tota la llista, mentre que les propietats afegides a l'etiqueta `` afectaran els elements individuals de la llista:

Exemple

```
ol {
    background: #ff9999;
    padding: 20px;
}
```

```

ul {
    background: #3399ff;
    padding: 20px;
}

ol li {
    background: #ffe5e5;
    color: darkred;
    padding: 5px;
    margin-left: 35px;
}

ul li {
    background: #cce5ff;
    color: darkblue;
    margin: 5px;
}

```

Totes les propietats de la llista CSS:

| Property | Description |
|--|---|
| <u>list-style</u> | Sets all the properties for a list in one declaration |
| <u>list-style-image</u> | Specifies an image as the list-item marker |
| <u>list-style-position</u> | Specifies the position of the list-item markers (bullet points) |
| <u>list-style-type</u> | Specifies the type of list-item marker |

19. Taules CSS

L'aspecte d'una taula HTML es pot millorar molt amb CSS.

Vores d'una taula

Per especificar vores de taula en CSS, utilitzeu la propietat **border**.

L'exemple següent especifica una vora sòlida per als elements <taula>, <th> i <td>:

```

table, th, td {
    border: 1px solid;
}

```

Fronteres dobles

Tingueu en compte que la taula dels exemples anteriors té vores dobles. Això es perquè tant la taula com els elements <th> i <td> tenen vores separades. Per eliminar vores dobles useu la propietat **border-collapse** estableix si les vores de la taula s'han de replegar en una única vora:

| Nom | Cognom |
|------|--------|
| Pere | Grifó |
| Lois | Grifó |

```

table, td, th {
    border: 1px solid black;
    border-collapse: collapse;
}

```

Amplada i alçada de la taula

L'amplada i l'alçada d'una taula es defineixen per les propietats **width** i **height**.

L'exemple següent estableix l'amplada de la taula al 100% i l'alçada dels elements **<th>** a 70 píxels. Exemple:

```
table {  
    width: 100%;  
}  
  
th {  
    height: 70px;  
}
```

Per crear una taula que només hauria d'omplir la meitat de la pàgina, utilitzeu **width: 50%**. Exemple:

```
table {  
    width: 50%;  
}
```

Alineació horitzontal

La propietat **text-align** estableix l'alignació horitzontal (esquerra, dreta o centre) del contingut a **<th>** o **<td>**.

Per defecte, el contingut dels elements **<th>** està alineat al centre i el contingut dels elements **<td>** està alineat a l'esquerra.

Per alinear al centre el contingut dels elements **<td>** també, utilitzeu **text-align: center**:

Alineació vertical

La propietat **vertical-align** estableix l'alignació vertical (com ara la part superior, inferior o mitjana) del contingut a **<th>** o **<td>**.

Per defecte, l'alignació vertical del contingut d'una taula és mitjana (tant per als elements **<th>** com per **<td>**).

Padding de taula

Per controlar l'espai entre la vora i el contingut d'una taula, utilitzeu la propietat **padding** dels elements **<td>** i **<th>**:

Taula Hoverable

Utilitzeu el selector **:hover** de **<tr>** per ressaltar les files de la taula sobre el ratolí.

```
tr:hover {background-color: coral;}
```

Taules a ratlles (zebra)

Per a les taules amb ratlles de zebra, utilitzeu el selector **nth-child()** i afegiu a totes les files de la taula parells (o senars) **background-color**. Exemple:

```
tr:nth-child(even) {background-color: #f2f2f2;}
```

Color de la taula

L'exemple següent especifica el color de fons i el color del text dels elements **<th>**.

```
th {  
    background-color: #04AA6D;  
    color: white;  
}
```

Taula responsiva

Una taula responsiva mostrarà una barra de desplaçament horitzontal si la pantalla és massa xicoteta per a mostrar el contingut complet:

Afegiu un element contenidor (com <div>) amb **overflow-x:auto** al voltant de l'element <table> perquè responga. Exemple:

```
<div style="overflow-x:auto;">  
  
<table>  
... table content ...  
</table>  
  
</div>
```

Propietats de la taula CSS

| Property | Description |
|---------------------------------|--|
| border | Sets all the border properties in one declaration |
| border-collapse | Specifies whether or not table borders should be collapsed |
| border-spacing | Specifies the distance between the borders of adjacent cells |
| caption-side | Specifies the placement of a table caption |
| empty-cells | Specifies whether or not to display borders and background on empty cells in a table |
| table-layout | Sets the layout algorithm to be used for a table |

20. La propietat de visualització

La propietat **display** és la propietat CSS més important per controlar el disseny.

La propietat de visualització

La propietat **display** especifica com es mostra un element.

Cada element HTML té un valor de visualització predeterminat en funció del tipus d'element que siga. El valor de visualització predeterminat per a la majoria dels elements és **block** o **inline**.

Elements a nivell de bloc

Un element a nivell de bloc sempre comença en una línia nova i ocupa tota l'amplada disponible (s'estén a l'esquerra i a la dreta tant com pot).

Exemples d'elements a nivell de bloc:

- <div>
- <h1> - <h6>
- <p>
- <form>
- <header>

- <footer>
- <section>

Elements en línia

Un element en línia no comença en una línia nova i només ocupa l'amplada necessària.

Aquest és un element en línia dins d' un paràgraf.

Exemples d'elements en línia:

-
- <a>
-

Visualització: cap;

display: none; s'utilitza habitualment amb JavaScript per ocultar i mostrar elements sense suprimir-los.

Substituï el valor de visualització predeterminat

Com s'ha esmentat, cada element té un valor de visualització predeterminat. Tanmateix, podeu anul·lar-ho.

Canviar un element en línia a un element de bloc, o viceversa, pot ser útil per fer que la pàgina sembla d'una manera específica i seguir els estàndards web.

Un exemple comú és fer elements en línia per als menús horizontals. Exemple:

```
li {  
    display: inline;  
}
```

Nota: establir la propietat de visualització d'un element només canvia com es mostra l'element, NO quin tipus d'element és. Per tant, un element en línia amb display: block; no pot tenir altres elements de bloc al seu interior.

L'exemple següent mostra elements com a elements de bloc:

```
span {  
    display: block;  
}
```

L'exemple següent mostra elements <a> com a elements de bloc:

```
a {  
    display: block;  
}
```

Amaga un element - display:none o visibility:hidden?

Es pot amagar un element establint la propietat display a none. L'element s'amagará i la pàgina es mostrerà com si l'element no estiguera.

Per contra, visibility:hidden; també amaga un element, però encara ocuparà el mateix espai que abans. L'element s'amagará, però encara afectarà el disseny.

Propietats de visualització/visibilitat CSS

Property Description

| | |
|-------------------|---|
| <u>display</u> | Specifies how an element should be displayed |
| <u>visibility</u> | Specifies whether or not an element should be visible |

21. CSS: amplada màxima

Utilitzant l'amplada, l'amplada màxima i el marge: automàtic.

Com s'ha esmentat en el punt anterior, un element a nivell de bloc sempre ocupa tota l'amplada disponible (s'estén cap a l'esquerra i la dreta tant com pot).

La configuració **width** d'un element a nivell de bloc evitarà que s'estinga fins a les vores del seu contingut. A continuació, podeu establir els marges com a automàtic, per centrar horitzontalment l'element dins del seu contingut. L'element ocuparà l'amplada especificada i l'espai restant es dividirà a parts iguals entre els dos marges.

El problema amb l'anterior **<div>** es produeix quan la finestra del navegador és més xicoteta que l'amplada de l'element. Aleshores, el navegador afegeix una barra de desplaçament horitzontal a la pàgina.

En canvi, l'ús **max-width**, en aquesta situació, millorarà el maneig del navegador de finestres menudes. Això és important a l'hora de fer que un lloc es puga utilitzar en dispositius mòbils. Mira aquest exemple aclaridor:

```
<!DOCTYPE html>
<html>
<head>
<style>
div.ex1 {
    width: 500px;
    margin: auto;
    border: 3px solid #73AD21;
}

div.ex2 {
    max-width: 500px;
    margin: auto;
    border: 3px solid #73AD21;
}
</style>
</head>
<body>

<h2>CSS Max-width</h2>

<div class="ex1">This div element has width: 500px;</div>
<br>

<div class="ex2">This div element has max-width: 500px;</div>

<p><strong>Tip:</strong> Drag the browser window to smaller than 500px wide, to see the difference between the two divs!</p>

</body>
</html>
```

22. Disseny CSS : la propietat de la posició

La propietat **position** especifica el tipus de mètode de posicionament utilitzat per a un element (estàtic, relatiu, fix, absolut o enganxós). Hi ha cinc valors de posició diferents:

- **static**
- **relative**
- **fixed**
- **absolute**
- **sticky**

Posició: estàtica;

Els elements HTML es col·loquen estàtics per defecte. Els elements posicionats estàtics no es veuen afectats per les propietats **top**, **left**, **right**, **bottom** o **z-index**.

Un element amb **position: static;** no està posicionat de cap manera especial. Sempre es posiciona segons el flux normal de la pàgina:

Posició: relatiu;

Un element amb **position: relative;** es posiciona en relació a la seva posició normal.

Li afecten les propietats superior, dreta, inferior i esquerra, i cupa l'espai que ha d'ocupar.

L'altre contingut no s'ajustarà per adaptar-se a cap buit deixat per l'element.

Posició: fixa;

Un element amb **position: fixed;** es posiciona en relació a la finestra gràfica (viewport), la qual cosa significa que sempre es manté al mateix lloc encara que es desplace la pàgina. Les propietats superior, dreta, inferior i esquerra s'utilitzen per posicionar l'element.

Un element fix no deixa un buit a la pàgina on normalment s'hauria situat.

Posició: absoluta;

Un element amb **position: absolute;** es posiciona en relació amb l'avantpassat més proper (en lloc de posicionar-se en relació a la finestra gràfica, com si fos fix).

Tot i això, si un element posicionat absolut no té avantpassats posicionats, utilitza el cos del document i es mou amb el desplaçament de la pàgina.

Nota: els elements posicionats en absolut s'eliminen del flux normal i es poden solapar amb elements.

Posició: enganxosa;

Un element amb **position: sticky;** es posiciona en funció de la posició de desplaçament de l'usuari.

Un element enganxós alterna entre **relative** i **fixed**, depenent de la posició de desplaçament. Es col·loca relativament fins que es troba una posició de desplaçament determinada a la finestra gràfica; després s'enganxa al seu lloc (com la posició: fixa).

Un exemple:

```

<!DOCTYPE html>
<html>
<head>
<style>
div.sticky {
    position: -webkit-sticky;
    position: sticky;
    top: 0;
    padding: 5px;
    background-color: #cae8ca;
    border: 2px solid #4CAF50;
}
</style>
</head>
<body>

<p>Try to <b>scroll</b> inside this frame to understand how sticky positioning works.</p>

<div class="sticky">I am sticky!</div>

<div style="padding-bottom:2000px">
    <p>In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position.</p>
    <p>Scroll back up to remove the stickyness.</p>
    <p>Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>
    <p>Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>
</div>

</body>
</html>

```

Posicionament del text en una imatge

Com situar el text sobre una imatge:

```

<!DOCTYPE html>
<html>
<head>
<style>
.container {
    position: relative;
}

.topleft {
    position: absolute;
    top: 8px;
    left: 16px;
    font-size: 18px;
}

img {
    width: 100%;
    height: auto;
}

```

```

    opacity: 0.3;
}
</style>
</head>
<body>

<h2>Image Text</h2>
<p>Add some text to an image in the top left corner:</p>

<div class="container">
    
        <div class="topleft">Top Left</div>
    </div>

</body>
</html>

```

Aquest exemple es top i topleft, però pot ser qualsevol combinació.

Totes les propietats de posicionament CSS

| Property | Description |
|-----------------|--|
| <u>bottom</u> | Sets the bottom margin edge for a positioned box |
| <u>clip</u> | Clips an absolutely positioned element |
| <u>left</u> | Sets the left margin edge for a positioned box |
| <u>position</u> | Specifies the type of positioning for an element |
| <u>right</u> | Sets the right margin edge for a positioned box |
| <u>top</u> | Sets the top margin edge for a positioned box |

23. Disseny CSS : la propietat z-index

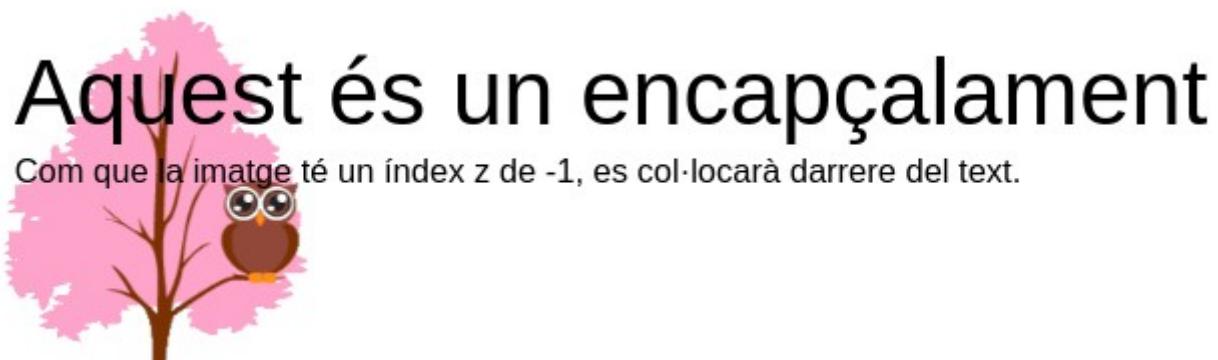
La propietat **z-index** especifica l'ordre de pila d'un element.

La propietat z-index

Quan es col·loquen elements, es poden solapar amb altres elements.

La propietat **z-index** especifica l'ordre de pila d'un element (quin element s'ha de col·locar davant o darrere dels altres).

Un element pot tenir un ordre de pila positiu o negatiu:



Exemple

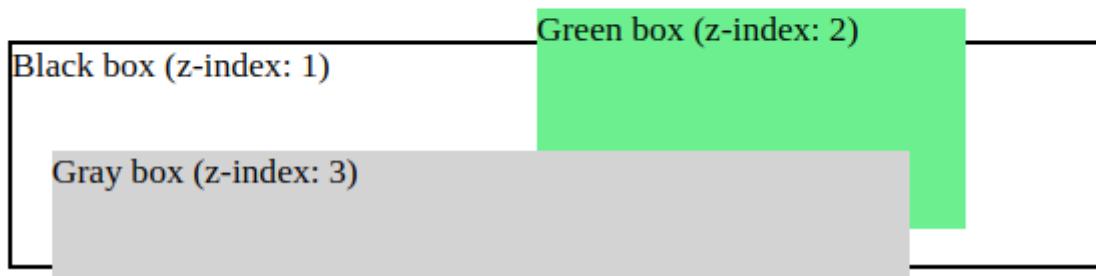
```
img {
    position: absolute;
```

```
left: 0px;  
top: 0px;  
z-index: -1;  
}
```

Un altre exemple d'índex z

Z-index Example

An element with greater stack order is always above an element with a lower stack order.



```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
.container {  
  position: relative;  
}  
  
.black-box {  
  position: relative;  
  z-index: 1;  
  border: 2px solid black;  
  height: 100px;  
  margin: 30px;  
}  
  
.gray-box {  
  position: absolute;  
  z-index: 3; /* gray box will be above both green and black box */  
  background: lightgray;  
  height: 60px;  
  width: 70%;  
  left: 50px;  
  top: 50px;  
}  
  
.green-box {  
  position: absolute;  
  z-index: 2; /* green box will be above black box */  
  background: lightgreen;  
  width: 35%;  
  left: 270px;  
  top: -15px;  
  height: 100px;
```

```

}
</style>
</head>
<body>

<h1>Z-index Example</h1>

<p>An element with greater stack order is always above an element with a lower stack order.</p>

<div class="container">
  <div class="black-box">Black box (z-index: 1)</div>
  <div class="gray-box">Gray box (z-index: 3)</div>
  <div class="green-box">Green box (z-index: 2)</div>
</div>

</body>
</html>

```

Sense índex z

Si dos elements se superposen sense **z-index** especificat, l'element definit **per últim al codi HTML** es mostrerà per damunt.

24. Disseny CSS - Desbordament

La propietat **overflow** CSS controla què passa amb el contingut massa gran per a cabre en una àrea.

Desbordament CSS

La propietat **overflow** especifica si cal retallar el contingut o afegir barres de desplaçament quan el contingut d'un element és massa gran per cabre a l'àrea especificada.

La propietat **overflow** té els valors següents:

- **visible** - Per defecte. El desbordament no està retallat. El contingut es representa fora de la caixa de l'element
- **hidden** - El desbordament es retalla i la resta del contingut serà invisible
- **scroll** - El desbordament es retalla i s'afegeix una barra de desplaçament per veure la resta del contingut
- **auto** - Similar a **scroll**, però afegeix barres de desplaçament només quan cal

Desbordament-x i desbordament-y

Les propietats **overflow-x** i **overflow-y** especifica si s'ha de canviar el desbordament del contingut només horitzontalment o verticalment (o tots dos):

overflow-x especifica què fer amb les vores esquerra/dreta del contingut.

overflow-y especifica què fer amb les vores superior/inferior del contingut.

Totes les propietats de desbordament de CSS

| Property | Description |
|----------------------|--|
| <u>overflow</u> | Specifies what happens if content overflows an element's box |
| <u>overflow-wrap</u> | Specifies whether or not the browser can break lines with long words, if they overflow its container |

| | |
|-------------------|--|
| <u>overflow-x</u> | Specifies what to do with the left/right edges of the content if it overflows the element's content area |
| <u>overflow-y</u> | Specifies what to do with the top/bottom edges of the content if it overflows the element's content area |

25. Disseny CSS : float i clear

La propietat CSS **float** especifica com ha de flotar un element.

La propietat **clear** CSS especifica quins elements poden flotar al costat de l'element esborrat i de quin costat.

La propietat del float

La propietat **float** s'utilitza per col·locar i donar format al contingut, per exemple, deixar flotar una imatge al text d'un contingut.

La propietat **float** pot tenir un dels valors següents:

- **left** - L'element flota a l'esquerra del seu contingut
- **right** - L'element flota a la dreta del seu contingut
- **none** - L'element no flota (es mostrarà just on apareix el text). Això és per defecte
- **inherit** - L'element hereta el valor flotant del seu pare

En el seu ús més senzill, la propietat **float** es pot utilitzar per a distribuir el text al voltant de les imatges.

Exemple - flotant: dreta;

```
<!DOCTYPE html>
<html>
<head>
<style>
  img {
    float: right;
  }
</style>
</head>
<body>
  <h2>Float Right</h2>
  <p>In this example, the image will float to the right in the paragraph, and the text in the paragraph will wrap around the image.</p>

  <p>
  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum dignissim ac. In at libero sed nunc venenatis imperdiet sed ornare turpis. Donec vitae dui eget tellus gravida venenatis. Integer fringilla congue eros non fermentum. Sed dapibus pulvinar nibh tempor porta. Cras ac leo purus. Mauris quis diam velit.</p>
</body>
</html>
```

Float Right

In this example, the image will float to the right in the paragraph, and the text in the paragraph will wrap around the image.

Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Phasellus imperdiet, nulla et dictum interdum, nisi lorem
egestas odio, vitae scelerisque enim ligula venenatis
dolor. Maecenas nisl est, ultrices nec congue eget, auctor
vitae massa. Fusce luctus vestibulum augue ut aliquet.
Mauris ante ligula, facilisis sed ornare eu, lobortis in
odio. Praesent convallis urna a lacus interdum ut
hendrerit risus congue. Nunc sagittis dictum nisi, sed
ullamcorper ipsum dignissim ac. In at libero sed nunc
venenatis imperdiet sed ornare turpis. Donec vitae dui eget tellus gravida venenatis.
Integer fringilla congue eros non fermentum. Sed dapibus pulvinar nibh tempor porta.
Cras ac leo purus. Mauris quis diam velit.



flotar: esquerra;

```
img {  
    float: left;  
}
```

sense flotació

En l'exemple següent, la imatge es mostrerà just on apareix al text (float: none):

```
img {  
    float: none;  
}
```



El client és molt important, el client serà seguit pel client. De vegades es

diu que no hi ha inversió a la flota, excepte que cal odiar el món, perquè la xocolata de la vida és un dolor verinós. Mecenas és un aficionat, no necessita deures, és l'autor de la vida de masses.

Flotar l'un al costat de l'altre

Normalment, els elements `div` es mostraran uns sobre els altres. Tanmateix, si fem servir `float: left` podem deixar que els elements floten uns al costat dels altres:

Float Next To Each Other

In this example, the three divs will float next to each other.



```

<!DOCTYPE html>
<html>
<head>
<style>
  div {
    float: left;
    padding: 15px;
  }

  .div1 {
    background: red;
  }

  .div2 {
    background: yellow;
  }

  .div3 {
    background: green;
  }
</style>
</head>
<body>

  <h2>Float Next To Each Other</h2>

  <p>In this example, the three divs will float next to each other.</p>

  <div class="div1">Div 1</div>
  <div class="div2">Div 2</div>
  <div class="div3">Div 3</div>

</body>
</html>

```

La propietat clear

Quan utilitzem la propietat **float**, i volem el següent element a baix (no a la dreta ni a l'esquerra), haurem d'utilitzar la propietat **clear**.

La propietat **clear** especifica què ha de passar amb l'element que està al costat d'un element flotant.

La propietat **clear** pot tenir un dels valors següents:

- **none** - L'element no s'empeny per sota dels elements flotants esquerre o dret. Això és per defecte
- **left**- L'element s'empeny per sota dels elements flotants esquerre
- **right**- L'element s'empeny per sota dels elements flotants a la dreta
- **both**- L'element s'empeny per sota dels elements flotants esquerre i dret
- **inherit**- L'element hereta el valor clar del seu pare

Exemple

```

<!DOCTYPE html>
<html>
<head>
<style>
  .div1 {
    float: left;

```

```

padding: 10px;
border: 3px solid #73AD21;
}

.div2 {
padding: 10px;
border: 3px solid red;
}

.div3 {
float: left;
padding: 10px;
border: 3px solid #73AD21;
}

.div4 {
padding: 10px;
border: 3px solid red;
clear: left;
}
</style>
</head>
<body>

<h2>Without clear</h2>
<div class="div1">div1</div>
<div class="div2">div2 - Notice that div2 is after div1 in the HTML code.
However, since div1 floats to the left, the text in div2 flows around
div1.</div>
<br><br>

<h2>With clear</h2>
<div class="div3">div3</div>
<div class="div4">div4 - Here, clear: left; moves div4 down below the
floating div3. The value "left" clears elements floated to the left. You can
also clear "right" and "both".</div>

</body>
</html>

```

Without clear

div1 div2 - Notice that div2 is after div1 in the HTML code. However, since div1 floats to the left, the text in div2 flows around div1.

With clear

div3

div4 - Here, clear: left; moves div4 down below the floating div3. The value "left" clears elements floated to the left. You can also clear "right" and "both".

El clearfix Hack

Si un element flotant és més alt que l'element que el conté, "desbordarà" fora del seu contingut. A continuació, podem afegir un hack clearfix per resoldre aquest problema:

Sense Clearfix

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum...



Amb Clearfix

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum...



Exemple

```
.clearfix {  
    overflow: auto;  
}
```

Exemple: menú de navegació

També podeu utilitzar **float** amb una llista d'hiperenllaços per crear un menú horitzontal:

```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
ul {  
    list-style-type: none;  
    margin: 0;  
    padding: 0;
```

```

        overflow: hidden;
        background-color: #333;
    }

    li {
        float: left;
    }

    li a {
        display: inline-block;
        color: white;
        text-align: center;
        padding: 14px 16px;
        text-decoration: none;
    }

    li a:hover {
        background-color: #111;
    }

    .active {
        background-color: red;
    }

```

</style>

</head>

<body>

- Home
- News
- Contact
- About

</body>

</html>

26. Disseny CSS - visualització: bloc en línia

Comparem **display: inline** i **display: inline-block**, la diferència principal és que **display: inline-block** permet establir una amplada i una alçada a l'element.

A més, amb **display: inline-block**, es respecten els marges/encoixinats superior i inferior, però amb **display: inline** no.

En comparació amb **display: block**, la diferència principal és que **display: inline-block** no afegeix un salt de línia després de l'element, de manera que l'element es pot situar al costat d'altres elements.

L'exemple següent mostra el comportament diferent de: **display: inline**, **display: inline-block** i **display: block**

Exemple

```

<!DOCTYPE html>
<html>
<head>
<style>
span.a {
    display: inline; /* the default for span */

```

```

width: 100px;
height: 100px;
padding: 5px;
border: 1px solid blue;
background-color: yellow;
}

span.b {
  display: inline-block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}

span.c {
  display: block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}

```

</style>

</head>

<body>

<h1>The display Property</h1>

<h2>display: inline</h2>

<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet. </div>

<h2>display: inline-block</h2>

<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet. </div>

<h2>display: block</h2>

<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet. </div>

</body>

</html>

Utilitzant el bloc en línia per crear enllaços de navegació

Un ús comú **display: inline-block** és mostrar els elements de la llista horitzontalment en lloc de verticalment. L'exemple següent crea enllaços de navegació horitzontal:

Exemple

```

<!DOCTYPE html>
<html>
<head>

```

```

<style>
.nav {
  background-color: yellow;
  list-style-type: none;
  text-align: center;
  margin: 0;
  padding: 0;
}

.nav li {
  display: inline-block;
  font-size: 20px;
  padding: 20px;
}
</style>
</head>
<body>

<h1>Horizontal Navigation Links</h1>

<ul class="nav">
  <li><a href="#home">Home</a></li>
  <li><a href="#about">About Us</a></li>
  <li><a href="#clients">Our Clients</a></li>
  <li><a href="#contact">Contact Us</a></li>
</ul>

</body>
</html>

```

27. Alineació CSS

Elements d'alignació central

Per centrar horitzontalment un element de bloc (com `<div>`), utilitzeu `margin: auto;`. Establir l'amplada de l'element evitarà que abaste fins a les vores del seu contingut. Aleshores, l'element ocuparà l'amplada especificada i l'espai restant es dividirà a parts iguals entre els dos marges:

Aquest element `div` està centrat.

Exemple

```

<!DOCTYPE html>
<html>
<head>
<style>
.center {
  margin: auto;
  width: 60%;
  border: 3px solid #73AD21;
  padding: 10px;
}
</style>
</head>
<body>

<h2>Center Align Elements</h2>
<p>To horizontally center a block element (like div), use margin: auto;</p>

```

```
<div class="center">
  <p>Hello World!</p>
</div>

</body>
</html>
```

Alineació al centre del text

Per centrar només el text dins d'un element, feu servir `text-align: center;`

Aquest text està centrat. Exemple:

```
.center {
  text-align: center;
  border: 3px solid green;
}
```

Centra una imatge

Per centrar una imatge, establiu el marge esquerre i dret `auto` i convertiu-lo en un element `block`:

Exemple

```
img {
  display: block;
  margin-left: auto;
  margin-right: auto;
  width: 40%;
}
```

Alineació esquerra i dreta: utilitzant la posició

Un mètode per alinear elements és utilitzar `position: absolute;`

Right align with the position property

An example of how to right align elements with the position property:

In my younger and more vulnerable years my father gave me some advice that I've been turning over in my mind ever since.

```
<!DOCTYPE html>
<html>
<head>
<style>
.right {
  position: absolute;
  right: 0px;
  width: 300px;
  border: 3px solid #73AD21;
```

```

        padding: 10px;
    }
</style>
</head>
<body>

<h2>Right align with the position property</h2>

<p>An example of how to right align elements with the position property:</p>

<div class="right">
    <p>In my younger and more vulnerable years my father gave me some advice that I've been turning over in my mind ever since.</p>
</div>

</body>
</html>

```

Nota: els elements posicionats en absolut s'eliminen del flux normal i es poden solapar amb elements.

Alineació esquerra i dreta: utilitzant flotant

Un altre mètode per alinear elements és utilitzar la propietat **float**. Exemple:

```

.right {
    float: right;
    width: 300px;
    border: 3px solid #73AD21;
    padding: 10px;
}

```

Centreu verticalment: utilitzant el farciment

Hi ha moltes maneres de centrar un element verticalment en CSS. Una solució senzilla és utilitzar la part superior i inferior **padding**:

Estic centrat verticalment. Exemple:

```

.center {
    padding: 70px 0;
    border: 3px solid green;
}

```

Per centrar tant verticalment com horitzontalment, utilitzeu **padding** i **text-align: center**:

Estic centrat verticalment i horitzontalment:

```

.center {
    padding: 70px 0;
    border: 3px solid green;
    text-align: center;
}

```

Centreu verticalment: utilitzant Flexbox

També podeu utilitzar flexbox per centrar les coses.

Estic centrat verticalment i horitzontalment. Exemple:

```

.center {
    display: flex;
    justify-content: center;
}

```

```
    align-items: center;
    height: 200px;
    border: 3px solid green;
}
```

28. Combinadors CSS

Un combinador és alguna cosa que explica la relació entre els selectors.

Un selector CSS pot contenir més d'un selector simple. Entre els selectors simples, podem incloure un combinador.

Hi ha quatre combinadors diferents en CSS:

- selector descendent (espai)
- selector de fills (>)
- selector de germans adjacents (+)
- selector general de germans (~)

Selector descendent

El selector descendent coincideix amb tots els elements que són descendents d'un element especificat.

L'exemple següent selecciona tots els elements <p> dins dels elements <div>:

```
div p {
    background-color: yellow;
}
```

Selector de fills (>)

El selector fill selecciona tots els elements que són fills d'un element especificat.

L'exemple següent selecciona tots els elements <p> que són fills d'un element <div>. Els nets no, per a que quede clar.

```
div > p {
    background-color: yellow;
}
```

Selector de germans adjacents (+)

El selector de germans adjacents s'utilitza per seleccionar un element que està directament darrere d'un altre element específic.

Els elements germans han de tenir el mateix element pare, i "adjacent" significa "immediatament següent".

L'exemple següent selecciona el primer element <p> que es col·loca immediatament després dels elements <div>:

```
div + p {
    background-color: yellow;
}
```

Selector general de germans (~)

El selector general de germans selecciona tots els elements que són germans següents d'un element especificat.

L'exemple següent selecciona tots els elements <p> que són els següents germans dels elements <div>:

```
div ~ p {  
    background-color: yellow;  
}
```

Tots els selectors de combinadors CSS

| Selector | Example | Example description |
|---------------------------|---------|--|
| <u>element element</u> | div p | Selects all <p> elements inside <div> elements |
| <u>element>element</u> | div > p | Selects all <p> elements where the parent is a <div> element |
| <u>element+element</u> | div + p | Selects the first <p> element that are placed immediately after <div> elements |
| <u>element1~element2</u> | p ~ ul | Selects every element that are preceded by a <p> element |

29. Pseudo-classes de CSS

Què són les pseudo-classes?

Una pseudo-classe s'utilitza per definir un estat especial d'un element.

Per exemple, es pot utilitzar per:

- Estil d'un element quan un usuari hi passa el ratolí
- Estil dels enllaços visitats i no visitats de manera diferent
- Estil d'un element quan se centra

Sintaxi

La sintaxi de les pseudo-classes:

```
selector:pseudo-class {  
    property: value;  
}
```

Els enllaços es poden mostrar de diferents maneres:

```
/* unvisited link */  
a:link {  
    color: #FF0000;  
}  
  
/* visited link */  
a:visited {  
    color: #00FF00;  
}  
  
/* mouse over link */  
a:hover {  
    color: #FF00FF;  
}  
  
/* selected link */  
a:active {
```

```
        color: #0000FF;  
    }
```

Nota: `a:hover` ha d'anar després `a:link` i `a:visited` a la definició CSS, per tal de ser efectiu! `a:active` ha d'anar després `a:hover` a la definició CSS per tal de ser efectiu! Els noms de pseudoclasses no distingeixen entre majúscules i minúscules.

Pseudo-classes i classes HTML

Les pseudoclasses es poden combinar amb classes HTML:

Quan passeu el cursor per sobre de l'enllaç de l'exemple, canviarà de color. Exemple:

```
a.highlight:hover {  
    color: #ff0000;  
}
```

Passeu el cursor a <div>

Un exemple d'ús de la pseudo-classe `:hover` en un element `<div>`. Exemple:

```
div:hover {  
    background-color: blue;  
}
```

Passeu el cursor per sobre d'un `<div>` per mostrar un element `<p>`. Prova el codi:

```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
p {  
    display: none;  
    background-color: yellow;  
    padding: 20px;  
}  
  
div:hover p {  
    display: block;  
}  
</style>  
</head>  
<body>  
  
<div>Hover over this div element to show the p element  
    <p>Tada! Here I am!</p>  
</div>  
  
</body>  
</html>
```

CSS - La pseudo-classe del primer fill

La pseudo-classe `:first-child` coincideix amb un element especificat que és el primer fill d'un altre element.

Coincideix amb el primer element `<p>`

A l'exemple següent, el selector coincideix amb qualsevol element que siga el primer fill de qualsevol element `<p>`. Exemple:

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

A l'exemple següent, el selector coincideix amb el primer element <i> de tots els elements <p>. Exemple:

```
p i:first-child {
    color: blue;
}
```

CSS - La pseudo-classe :lang

La pseudo-classe `:lang` us permet definir regles especials per a diferents idiomes.

A l'exemple següent, `:lang` defineix les cometes per als elements <q> amb lang="no":

```
<!DOCTYPE html>
<html>
<head>
<style>
q:lang(no) {
    quotes: "~" "~";
}
</style>
</head>
<body>

<p>Some text <q lang="no">A quote in a paragraph</q> Some text.</p>

</body>
</html>
```

Totes les Pseudo Classes

| Selector | Example | Example description |
|------------------------------|------------------------------|--|
| <code>:active</code> | <code>a:active</code> | Selects the active link |
| <code>:checked</code> | <code>input:checked</code> | Selects every checked <input> element |
| <code>:disabled</code> | <code>input:disabled</code> | Selects every disabled <input> element |
| <code>:empty</code> | <code>p:empty</code> | Selects every <p> element that has no children |
| <code>:enabled</code> | <code>input:enabled</code> | Selects every enabled <input> element |
| <code>:first-child</code> | <code>p:first-child</code> | Selects every <p> elements that is the first child of its parent |
| <code>:first-of-type</code> | <code>p:first-of-type</code> | Selects every <p> element that is the first <p> element of its parent |
| <code>:focus</code> | <code>input:focus</code> | Selects the <input> element that has focus |
| <code>:hover</code> | <code>a:hover</code> | Selects links on mouse over |
| <code>:in-range</code> | <code>input:in-range</code> | Selects <input> elements with a value within a specified range |
| <code>:invalid</code> | <code>input:invalid</code> | Selects all <input> elements with an invalid value |
| <code>:lang(language)</code> | <code>p:lang(it)</code> | Selects every <p> element with a lang attribute value starting with "it" |
| <code>:last-child</code> | <code>p:last-child</code> | Selects every <p> elements that is the last child of its parent |
| <code>:last-of-type</code> | <code>p:last-of-type</code> | Selects every <p> element that is the last <p> element of its parent |
| <code>:link</code> | <code>a:link</code> | Selects all unvisited links |
| <code>:not(selector)</code> | <code>:not(p)</code> | Selects every element that is not a <p> element |

| | | |
|---|-----------------------|--|
| <u>:nth-child(n)</u> | p:nth-child(2) | Selects every <p> element that is the second child of its parent |
| <u>:nth-last-child(n)</u> | p:nth-last-child(2) | Selects every <p> element that is the second child of its parent, counting from the last child |
| <u>:nth-last-of-type(n)</u> | p:nth-last-of-type(2) | Selects every <p> element that is the second <p> element of its parent, counting from the last child |
| <u>:nth-of-type(n)</u> | p:nth-of-type(2) | Selects every <p> element that is the second <p> element of its parent |
| <u>:only-of-type</u> | p:only-of-type | Selects every <p> element that is the only <p> element of its parent |
| <u>:only-child</u> | p:only-child | Selects every <p> element that is the only child of its parent |
| <u>:optional</u> | input:optional | Selects <input> elements with no "required" attribute |
| <u>:out-of-range</u> | input:out-of-range | Selects <input> elements with a value outside a specified range |
| <u>:read-only</u> | input:read-only | Selects <input> elements with a "readonly" attribute specified |
| <u>:read-write</u> | input:read-write | Selects <input> elements with no "readonly" attribute |
| <u>:required</u> | input:required | Selects <input> elements with a "required" attribute specified |
| <u>:root</u> | root | Selects the document's root element |
| <u>:target</u> | #news:target | Selects the current active #news element (clicked on a URL containing that anchor name) |
| <u>:valid</u> | input:valid | Selects all <input> elements with a valid value |
| <u>:visited</u> | a:visited | Selects all visited links |

30. Pseudoelements CSS

Què són els pseudo-elements?

Un pseudoelement CSS s'utilitza per estilitzar les parts específiques d'un element.

Per exemple, es pot utilitzar per:

- Estil la primera lletra, o línia, d'un element
- Inserir contingut abans o després del contingut d'un element

Sintaxi

La sintaxi dels pseudoelements:

```
selector::pseudo-element {
    property: value;
}
```

El pseudoelement ::de primera línia

El pseudoelement **::first-line** s'utilitza per afegir un estil especial a la primera línia d'un text.

L'exemple següent formata la primera línia del text en tots els elements <p>. Exemple:

```
p::first-line {  
    color: #ff0000;  
    font-variant: small-caps;  
}
```

Nota: el pseudoelement `::first-line` només es pot aplicar als elements de bloc.

Les propietats següents s'apliquen al `::first-line` pseudoelement:

- propietats del tipus de lletra
- propietats del color
- propietats de fons
- espaiat de paraules
- espai entre lletres
- text-decoració
- alineació vertical
- text-transformació
- alçada de línia
- clar

El pseudoelement `::primera lletra`

El pseudoelement `::first-letter` s'utilitza per afegir un estil especial a la primera lletra d'un text.

L'exemple següent formata la primera lletra del text en tots els elements `<p>`:

```
p::first-letter {  
    color: #ff0000;  
    font-size: xx-large;  
}
```

Nota: el pseudoelement `::first-letter` només es pot aplicar als elements de bloc.

Les propietats següents s'apliquen al pseudoelement `::first-letter`:

- propietats del tipus de lletra
- propietats del color
- propietats de fons
- propietats del marge
- propietats de farciment
- propietats frontereres
- text-decoració
- alineació vertical (només si "float" és "cap")
- text-transformació
- alçada de línia
- flotar
- clar

Pseudoelements i classes HTML

Els pseudoelements es poden combinar amb classes HTML:

```
p.intro::first-letter {  
    color: #ff0000;  
    font-size: 200%;  
}
```

L'exemple anterior mostrarà la primera lletra dels paràgrafs amb class="intro", en roig i en una mida més gran.

Pseudelements múltiples

També es poden combinar diversos pseudelements.

En l'exemple següent, la primera lletra d'un paràgraf serà roja, amb una mida de lletra xx large. La resta de la primera línia serà blava i en majúscules xicotetes. La resta del paràgraf serà la mida i el color de la lletra per defecte:

```
p::first-letter {  
    color: #ff0000;  
    font-size: xx-large;  
}  
  
p::first-line {  
    color: #0000ff;  
    font-variant: small-caps;  
}
```

CSS - El pseudelement ::before

El pseudelement **::before** es pot utilitzar per inserir contingut abans del contingut d'un element.

L'exemple següent insereix una imatge abans del contingut de cada element <h1>:

```
h1::before {  
    content: url(smiley.gif);  
}
```

CSS - El pseudelement ::after

El pseudelement **::after** es pot utilitzar per inserir contingut després del contingut d'un element. Funciona com before.

CSS - El pseudelement ::marker

El pseudelement **::marker** selecciona els marcadors dels elements de la llista.

L'exemple següent estilitza els marcadors dels elements de la llista:

```
::marker {  
    color: red;  
    font-size: 23px;  
}
```

CSS - El pseudelement ::selection

El pseudelement **::selection** coincideix amb la part d'un element seleccionada per un usuari.

Les propietats CSS següents es poden aplicar a **::selection: color, background, cursor i outline**.

L'exemple següent fa que el text seleccionat siga roig sobre un fons groc:

```
::selection {  
    color: red;
```

```
    background: yellow;  
}
```

Tots els pseudoelements CSS

| Selector | Example | Example description |
|----------------|-----------------|--|
| ::after | p::after | Insert something after the content of each <p> element |
| ::before | p::before | Insert something before the content of each <p> element |
| ::first-letter | p::first-letter | Selects the first letter of each <p> element |
| ::first-line | p::first-line | Selects the first line of each <p> element |
| ::marker | ::marker | Selects the markers of list items |
| ::selection | p::selection | Selects the portion of an element that is selected by a user |

31. CSS Opacitat / Transparència

La propietat **opacity** especifica l'opacitat/transparència d'un element.

imatge Transparent

La propietat **opacity** pot prendre un valor entre 0.0 i 1.0. Com més baix siga el valor, més transparent:

```
img {  
    opacity: 0.5;  
}
```

Efecte Hover transparent

La propietat **opacity** s'utilitza sovint juntament amb el **:hover** selector per canviar l'opacitat en passar el ratolí per sobre:

```
img {  
    opacity: 0.5;  
}  
  
img:hover {  
    opacity: 1.0;  
}
```

32. Barra de navegació CSS

Barres de navegació

Tenir una navegació fàcil d'utilitzar és important per a qualsevol lloc web.

Amb CSS podeu transformar menús HTML avorrits en barres de navegació atractives.

Barra de navegació = Llista d'enllaços

Una barra de navegació necessita HTML estàndard com a base.

En els nostres exemples construirem la barra de navegació a partir d'una llista HTML estàndard.

Una barra de navegació és bàsicament una llista d'enllaços, de manera que utilitzar els elements **** i ****. Exemple:

```

<ul>
  <li><a href="default.asp">Home</a></li>
  <li><a href="news.asp">News</a></li>
  <li><a href="contact.asp">Contact</a></li>
  <li><a href="about.asp">About</a></li>
</ul>

```

Ara eliminem les vinyetes i els marges i el farciment de la llista:

```

ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
}

```

Exemple explicat:

- **list-style-type: none;**- Lleva els marcadors de llista. Una barra de navegació no necessita marcadors de llista
- Estableix **margin: 0;** i **padding: 0;** elimina la configuració predeterminada del navegador

El codi de l'exemple anterior és el codi estàndard que s'utilitza tant a les barres de navegació verticals com horitzontals.

Barra de navegació vertical

Per crear una barra de navegació vertical, podeu estilitzar els elements `<a>` dins de la llista, a més del codi de l'exemple anterior. Exemple:

```

li a {
  display: block;
  width: 60px;
}

```

Exemple explicat:

- **display: block;**- Mostrar els enllaços com a elements de bloc fa que es puga fer clic a tota l'àrea de l'enllaç (no només el text), i ens permet especificar l'amplada (i el farciment, el marge, l'alçada, etc. si es vol)
- **width: 60px;**- Els elements de bloc ocupen tota l'amplada disponible per defecte. Volem especificar una amplada de 60 pixels

També podeu establir l'amplada de `` i eliminar l'amplada de `<a>`, ja que ocuparan tota l'amplada disponible quan es mostren com a elements de bloc. Això produirà el mateix resultat que el nostre exemple anterior. Exemple:

```

ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  width: 60px;
}

li a {
  display: block;
}

```

Exemples de barres de navegació vertical

Creeu una barra de navegació vertical bàsica amb un color de fons gris i canvieu el color de fons dels enllaços quan l'usuari mou el ratolí per sobre d'ells. Exemple:

```
<!DOCTYPE html>
<html>
<head>
<style>
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  width: 200px;
  background-color: #f1f1f1;
}

li a {
  display: block;
  color: #000;
  padding: 8px 16px;
  text-decoration: none;
}

/* Change the link color on hover */
li a:hover {
  background-color: #555;
  color: white;
}
</style>
</head>
<body>

<h2>Vertical Navigation Bar</h2>

<ul>
  <li><a href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>

</body>
</html>
```

Enllaç de navegació actiu/actual

Afegiu una classe "activa" a l'enllaç actual per fer saber a l'usuari a quina pàgina es troba:

```
.active {
  background-color: #04AA6D;
  color: white;
}
```

Centreu els enllaços i afegiu vores

Afegeix `text-align:center` a `` o `<a>` per centrar els enllaços.

Afegeix la **border** propietat a afegeix una vora al voltant de la barra de navegació. Si també voleu vores dins de la barra de navegació, afegiu a **border-bottom** tots els elements , excepte a l'últim. Exemple:

```
ul {  
    border: 1px solid #555;  
}  
  
li {  
    text-align: center;  
    border-bottom: 1px solid #555;  
}  
  
li:last-child {  
    border-bottom: none;  
}
```

Vertical fixa d'altura completa

Creeu una navegació lateral "adhesiva" d'alçada completa. Exemple:

```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
body {  
    margin: 0;  
}  
  
ul {  
    list-style-type: none;  
    margin: 0;  
    padding: 0;  
    width: 25%;  
    background-color: #f1f1f1;  
    position: fixed;  
    height: 100%;  
    overflow: auto;  
}  
  
li a {  
    display: block;  
    color: #000;  
    padding: 8px 16px;  
    text-decoration: none;  
}  
  
li a.active {  
    background-color: #04AA6D;  
    color: white;  
}  
  
li a:hover:not(.active) {  
    background-color: #555;  
    color: white;  
}  
</style>  
</head>  
<body>
```

```

<ul>
  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>

<div style="margin-left:25%;padding:1px 16px;height:1000px;">
  <h2>Fixed Full-height Side Nav</h2>
  <h3>Try to scroll this area, and see how the sidenav sticks to the page</h3>
  <p>Notice that this div element has a left margin of 25%. This is because the side navigation is set to 25% width. If you remove the margin, the sidenav will overlay/sit on top of this div.</p>
  <p>Also notice that we have set overflow:auto to sidenav. This will add a scrollbar when the sidenav is too long (for example if it has over 50 links inside of it).</p>
  <p>Some text..</p>
  <p>Some text..</p>
  <p>Some text..</p>
  <p>Some text..</p>
  <p>Some text..</p>
  <p>Some text..</p>
  <p>Some text..</p>
</div>

</body>
</html>

```

Barra de navegació horitzontal

Hi ha dues maneres de crear una barra de navegació horitzontal. Ús d'elements de llista **flotants** o **en línia**.

Elements de la llista en línia

Una manera de crear una barra de navegació horitzontal és especificar els elements `` com a en línia, a més del codi "estàndard" de la pàgina anterior. Exemple:

```

li {
  display: inline;
}

```

Exemple explicat:

- **display: inline;** - Per defecte, els elements `` són elements de bloc. Aquí, eliminem els salts de línia abans i després de cada element de la llista, per mostrar-los en una línia

Elements de llista flotant

Una altra manera de crear una barra de navegació horitzontal és fer flotar els elements `` i especificar un disseny per als enllaços de navegació. Exemple:

```

li {
  float: left;
}

a {
  display: block;
}

```

```

padding: 8px;
background-color: #dddddd;
}

```

Exemple explicat:

- **float: left;** - Utilitzeu float per aconseguir que els elements de bloc floten uns al costat dels altres
- **display: block;** - Ens permet especificar el farciment (i alçada, amplada, marges, etc. si es vol)
- **padding: 8px;** - Especifiqueu algun farciment entre cada element <a>, per fer-los veure bé
- **background-color: #dddddd;** - Afegiu un color de fons gris a cada element <a>

Consell: afegiu el color de fons a en comptes de cada element <a> si voleu un color de fons d'amplada completa.

Exemples de barres de navegació horitzontal

Creeu una barra de navegació horitzontal bàsica amb un color de fons fosc i canvieu el color de fons dels enllaços quan l'usuari mou el ratolí per sobre d'ells. Exemple:

```

<!DOCTYPE html>
<html>
<head>
<style>
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  background-color: #333;
}

li {
  float: left;
}

li a {
  display: block;
  color: white;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
}

li a:hover {
  background-color: #111;
}
</style>
</head>
<body>

<ul>
  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>

```

```
</ul>

</body>
</html>
```

Enllaços alineats a la dreta

Alineeu els enllaços a la dreta fent flotar els elements de la llista a la dreta (`float:right;`):

```
<ul>
  <li><a href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li style="float:right"><a class="active" href="#about">About</a></li>
</ul>
```

Separadors de fronteres

Afegiu la propietat `border-right` a `` per crear divisors d'enllaços. Exemple:

```
/* Add a gray right border to all list items, except the last item (last-child) */
li {
  border-right: 1px solid #bbb;
}

li:last-child {
  border-right: none;
}
```

Barra de navegació fixa

Feu que la barra de navegació es mantinga a la part superior o inferior de la pàgina, fins i tot quan l'usuari es desplaça per la pàgina:

```
<!DOCTYPE html>
<html>
<head>
<style>
body {margin:0;}

ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  background-color: #333;
  position: fixed;
  top: 0;
  width: 100%;
}

li {
  float: left;
}

li a {
  display: block;
  color: white;
  text-align: center;
```

```

padding: 14px 16px;
text-decoration: none;
}

li a:hover:not(.active) {
  background-color: #111;
}

.active {
  background-color: #04AA6D;
}
</style>
</head>
<body>

<ul>
  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>

<div style="padding:20px; margin-top:30px; background-
color:#1abc9c; height:1500px;">
  <h1>Fixed Top Navigation Bar</h1>
  <h2>Scroll this page to see the effect</h2>
  <h2>The navigation bar will stay at the top of the page while scrolling</h2>

  <p>Some text some text some text some text..</p>
  <p>Some text some text some text some text..</p>
</div>

</body>
</html>

```

Nota: És possible que la posició fixa no funcione correctament als dispositius mòbils.

Barra de navegació enganxosa

Afegiu **position: sticky;** a **** per crear una barra de navegació enganxosa.

Un element enganxós alterna entre relatiu i fix, depenent de la posició de desplaçament. Es col·loca relativament fins que es troba una posició de desplaçament determinada a la finestra gràfica; després s'enganxa al seu lloc (com la posició: fixa):

```

ul {
  position: -webkit-sticky; /* Safari */
  position: sticky;
  top: 0;
}

```

33. Menú desplegable

Menú desplegable bàsic

Creeu un quadre desplegable que apareix quan l'usuari mou el ratolí per sobre d'un element. Exemple:

```
<!DOCTYPE html>
<html>
<head>
<style>
.dropdown {
  position: relative;
  display: inline-block;
}

.dropdown-content {
  display: none;
  position: absolute;
  background-color: #f9f9f9;
  min-width: 160px;
  box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
  padding: 12px 16px;
  z-index: 1;
}

.dropdown:hover .dropdown-content {
  display: block;
}
</style>
</head>
<body>

<h2>Hoverable Dropdown</h2>
<p>Move the mouse over the text below to open the dropdown content.</p>

<div class="dropdown">
  <span>Mouse over me</span>
  <div class="dropdown-content">
    <p>Hello World!</p>
  </div>
</div>

</body>
</html>
```

Exemple explicat

Utilitzeu qualsevol element per obrir el contingut desplegable, per exemple, un element `` o un element `<button>`.

Utilitzeu un element contenidor (com `<div>`) per crear el contingut desplegable i afegir-hi el que vulgueu.

Emboliqueu un element `<div>` al voltant dels elements per posicionar correctament el contingut del desplegable amb CSS.

La classe `.dropdown` utilitza `position:relative`, que és necessari quan volem que el contingut desplegable es col·loque just a baix del botó desplegable (utilitzant `position:absolute`).

La classe `.dropdown-content` conté el contingut real del desplegable. S'amaga per defecte i es mostrerà al passar el cursor (vegeu més avall). Tingueu en compte que `min-width` està establert en 160 píxels. No dubteu a canviar-ho.

Consell: si voleu que l'amplada del contingut desplegable siga tan ample com el botó desplegable, configureu-lo `width` al 100% (i `overflow:auto` per activar el desplaçament a les pantalles xicotetes).

En lloc d'utilitzar una vora, hem utilitzat la `box-shadow` propietat CSS per fer que el menú desplegable sembla una "targeta".

El selector `:hover` s'utilitza per mostrar el menú desplegable quan l'usuari mou el ratolí sobre el botó desplegable.

Menú desplegable

Creeu un menú desplegable que permeta a l'usuari triar una opció d'una llista:

Aquest exemple és similar a l'anterior, excepte que afegim enllaços dins del quadre desplegable i els estilem per adaptar-se a un botó desplegable amb estil. Exemple:

```
<!DOCTYPE html>
<html>
<head>
<style>
.dropbtn {
    background-color: #4CAF50;
    color: white;
    padding: 16px;
    font-size: 16px;
    border: none;
    cursor: pointer;
}

.dropdown {
    position: relative;
    display: inline-block;
}

.dropdown-content {
    display: none;
    position: absolute;
    background-color: #f9f9f9;
    min-width: 160px;
    box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
    z-index: 1;
}

.dropdown-content a {
    color: black;
    padding: 12px 16px;
    text-decoration: none;
    display: block;
}

.dropdown-content a:hover {background-color: #f1f1f1}
```

```

.dropdown:hover .dropdown-content {
  display: block;
}

.dropdown:hover .dropbtn {
  background-color: #3e8e41;
}

```

</style>

</head>

<body>

<h2>Dropdown Menu</h2>

<p>Move the mouse over the button to open the dropdown menu.</p>

<div class="dropdown">

<button class="dropbtn">Dropdown</button>

<div class="dropdown-content">

Link 1

Link 2

Link 3

</div>

</div>

<p>Note: We use href="#" for test links. In a real web site this would be URLs.</p>

</body>

</html>

34. Galeria d'imatges CSS

CSS es pot utilitzar per crear una galeria d'imatges.

La següent galeria d'imatges es crea amb CSS:

```

<html>
<head>
<style>
div.gallery {
  margin: 5px;
  border: 1px solid #ccc;
  float: left;
  width: 180px;
}

div.gallery:hover {
  border: 1px solid #777;
}

div.gallery img {
  width: 100%;
  height: auto;
}

div.desc {
  padding: 15px;
  text-align: center;
}

```

</style>

</head>

```

<body>

<div class="gallery">
  <a target="_blank" href="img_5terre.jpg">
    
  </a>
  <div class="desc">Add a description of the image here</div>
</div>

<div class="gallery">
  <a target="_blank" href="img_forest.jpg">
    
  </a>
  <div class="desc">Add a description of the image here</div>
</div>

<div class="gallery">
  <a target="_blank" href="img_lights.jpg">
    
  </a>
  <div class="desc">Add a description of the image here</div>
</div>

<div class="gallery">
  <a target="_blank" href="img_mountains.jpg">
    
  </a>
  <div class="desc">Add a description of the image here</div>
</div>

</body>
</html>

```

Galeria d'imatges responsives

Com utilitzar les consultes multimèdia CSS per crear una galeria d'imatges responsives que es veurà bé en ordinadors de sobretaula, tauletes i telèfons intel·ligents.

```

<!DOCTYPE html>
<html>
<head>
<style>
div.gallery {
  border: 1px solid #ccc;
}

div.gallery:hover {
  border: 1px solid #777;
}

div.gallery img {
  width: 100%;
  height: auto;
}

div.desc {
  padding: 15px;
  text-align: center;
}

```

```

* {
  box-sizing: border-box;
}

.responsive {
  padding: 0 6px;
  float: left;
  width: 24.99999%;
}

@media only screen and (max-width: 700px) {
  .responsive {
    width: 49.99999%;
    margin: 6px 0;
  }
}

@media only screen and (max-width: 500px) {
  .responsive {
    width: 100%;
  }
}

.clearfix:after {
  content: "";
  display: table;
  clear: both;
}

```

</style>

</head>

<body>

<h2>Responsive Image Gallery</h2>

<h4>Resize the browser window to see the effect.</h4>

<div class="responsive">

<div class="gallery">

<div class="desc">Add a description of the image here</div>

</div>

</div>

<div class="responsive">

<div class="gallery">

<div class="desc">Add a description of the image here</div>

</div>

</div>

<div class="responsive">

<div class="gallery">

<div class="desc">Add a description of the image here</div>

```

        </div>
    </div>

<div class="responsive">
    <div class="gallery">
        <a target="_blank" href="img_mountains.jpg">
            
        </a>
        <div class="desc">Add a description of the image here</div>
    </div>
</div>

<div class="clearfix"></div>

<div style="padding:6px;">
    <p>This example use media queries to re-arrange the images on different screen sizes: for screens larger than 700px wide, it will show four images side by side, for screens smaller than 700px, it will show two images side by side. For screens smaller than 500px, the images will stack vertically (100%).</p>
    <p>You will learn more about media queries and responsive web design later in our CSS Tutorial.</p>
</div>

</body>
</html>

```

35. Selectors d'atribut CSS

És possible crear un estil d'elements HTML que tinguen atributs o valors d'atributs específics.

CSS [atribut] Selector

El **[attribute]** selector s'utilitza per seleccionar elements amb un atribut especificat.

L'exemple següent selecciona tots els elements `<a>` amb un atribut `target`:

```
a[target] {
    background-color: yellow;
}
```

CSS [atribut~="valor"] Selector

El **[attribute~="value"]** selector s'utilitza per seleccionar elements amb un valor d'atribut que conté una paraula específica.

L'exemple següent selecciona tots els elements amb un atribut de títol que conté una llista de paraules separades per espais, una de les quals és "flor":

```
[title~="flower"] {
    border: 5px solid yellow;
}
```

L'exemple anterior farà coincidir els elements amb `title="flor"`, `title="flor d'estiu"` i `title="flor nova"`, però no `= "flors"`.

CSS [atribut|= "valor"] Selector

El **[attribute|= "value"]** selector s'utilitza per seleccionar elements amb l'atribut especificat, el valor dels quals pot ser exactament el valor especificat, o el valor especificat seguit d'un guionet (-).

Nota: el valor ha de ser una paraula sencera, ja siga sola, com class="top", o seguida d'un guionet (-), com class="top-text":

```
[class|="top"] {  
    background: yellow;  
}
```

CSS [atribut^="valor"] Selector

El [attribute^="value"] selector s'utilitza per seleccionar elements amb l'atribut especificat, el valor dels quals comença amb el valor especificat.

L'exemple següent selecciona tots els elements amb un valor d'atribut de classe que comença per "top":

Nota: el valor no cal que siga una paraula sencera!

```
[class^="top"] {  
    background: yellow;  
}
```

CSS [atribut\$="valor"] Selector

El [attribute\$="value"] selector s'utilitza per seleccionar elements el valor de l'atribut dels quals acaba amb un valor especificat.

L'exemple següent selecciona tots els elements amb un valor d'atribut de classe que acaba amb "test":

Nota: el valor no cal que siga una paraula sencera!

```
[class$="test"] {  
    background: yellow;  
}
```

CSS [atribut*= "valor"] Selector

El [attribute*= "value"] selector s'utilitza per seleccionar elements el valor d'atribut dels quals conté un valor especificat.

L'exemple següent selecciona tots els elements amb un valor d'atribut de classe que conté "te":

Nota: el valor no cal que siga una paraula sencera!

```
[class*="te"] {  
    background: yellow;  
}
```

Exemples d'ús:

Els selectors d'atributs poden ser útils per dissenyar formularis sense classe o identificador:

```
input[type="text"] {  
    width: 150px;  
    display: block;  
    margin-bottom: 10px;  
    background-color: yellow;  
}  
  
input[type="button"] {  
    width: 120px;  
    margin-left: 35px;
```

```

        display: block;
    }

```

Tots els selectors d'atributs CSS

| Selector | Example | Example description |
|---------------------|----------------------|---|
| [attribute] | [target] | Selects all elements with a target attribute |
| [attribute=value] | [target="_blank"] | Selects all elements with target="_blank" |
| [attribute~=value] | [title~="flower"] | Selects all elements with a title attribute that contains a space-separated list of words, one of which is "flower" |
| [attribute =value] | [lang = "en"] | Selects all elements with a lang attribute value starting with "en" |
| [attribute^=value] | a[href^="https"] | Selects all <a> elements with a href attribute value starting with "https" |
| [attribute\$=value] | a[href\$=".pdf"] | Selects all <a> elements with a href attribute value ending with ".pdf" |
| [attribute*=value] | a[href*="w3schools"] | Selects all <a> elements with a href attribute value containing the substring "w3schools" |

36. Formularis CSS

L'aspecte d'un formulari HTML es pot millorar molt amb CSS:

Estil dels camps d'entrada

Utilitzeu la propietat **width** per determinar l'amplada del camp d'entrada:

```

input {
    width: 100%;
}

```

L'exemple anterior s'aplica a tots els elements <input>. Si només voleu estilitzar un tipus d'entrada específic, podeu utilitzar selectors d'atributs:

- **input[type=text]**- només seleccionarà camps de text
- **input[type=password]**- només seleccionarà camps de contrasenya
- **input[type=number]**- només seleccionarà camps numèrics
- etc..

Entrades encoixinats

Utilitzeu la propietat **padding** per afegir espai dins del camp de text.

Consell: quan tingueu moltes entrades una darrere de l'altra, és possible que també vulgueu afegir-ne algunes **margin** per afegir més espai fora d'elles:

```

input[type=text] {
    width: 100%;
    padding: 12px 20px;
    margin: 8px 0;
    box-sizing: border-box;
}

```

Entrades vorejades

Utilitzeu la propietat **border** per canviar la mida i el color de la vora i utilitzeu la propietat **border-radius** per afegir cantons arrodonits:

```
input[type=text] {  
    border: 2px solid red;  
    border-radius: 4px;  
}
```

Si només voleu una vora inferior, utilitzeu la propietat **border-bottom**:

Entrades de colors

Utilitzeu la propietat **background-color** per afegir un color de fons a l'entrada i la propietat **color** per canviar el color del text:

```
input[type=text] {  
    background-color: #3CBC8D;  
    color: white;  
}
```

Entrades enfocades

De manera predeterminada, alguns navegadors afegiran un contorn blau al voltant de l'entrada quan es focalitze. Podeu eliminar aquest comportament afegint a l'entrada **outline: none;**:

Utilitzeu el selector **:focus** per fer alguna cosa amb el camp d'entrada quan es centra.

Entrada amb icona/imatge

Si voleu una icona dins de l'entrada, utilitzeu la propietat **background-image** i colloqueu-la amb la propietat **background-position**. Observeu també que afegim un farciment esquerre gran per reservar l'espai de la icona:



```
input[type=text] {  
    background-color: white;  
    background-image: url('searchicon.png');  
    background-position: 10px 10px;  
    background-repeat: no-repeat;  
    padding-left: 40px;  
}
```

Entrada de cerca animada

En aquest exemple utilitzem la propietat **transition** CSS per animar l'amplada de l'entrada de cerca quan es centra. Més endavant aprendràs més sobre la propietat **transition**.

```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
input[type=text] {  
    width: 130px;  
    box-sizing: border-box;  
    border: 2px solid #ccc;  
    border-radius: 4px;  
    font-size: 16px;  
    background-color: white;
```

```

background-image: url('searchicon.png');
background-position: 10px 10px;
background-repeat: no-repeat;
padding: 12px 20px 12px 40px;
transition: width 0.4s ease-in-out;
}

input[type=text]:focus {
  width: 100%;
}
</style>
</head>
<body>

<h2>Animate width of search input</h2>

<form>
  <input type="text" name="search" placeholder="Search..">
</form>

</body>
</html>

```

Estilització d'àrees de text

Consell: utilitzeu la propietat **resize** per evitar que les àrees de text es canvien de mida (desactiveu el "grabber" a la cantonada inferior dreta):

```

textarea {
  width: 100%;
  height: 150px;
  padding: 12px 20px;
  box-sizing: border-box;
  border: 2px solid #ccc;
  border-radius: 4px;
  background-color: #f8f8f8;
  resize: none;
}

```

Estil dels menús de selecció

```

select {
  width: 100%;
  padding: 16px 20px;
  border: none;
  border-radius: 4px;
  background-color: #f1f1f1;
}

```

Estilitza els botons d'entrada

```

input[type=button], input[type=submit], input[type=reset] {
  background-color: #04AA6D;
  border: none;
  color: white;
  padding: 16px 32px;
  text-decoration: none;
  margin: 4px 2px;
  cursor: pointer;
}

/* Tip: use width: 100% for full-width buttons */

```

Formulari responsiu

Canvieu la mida de la finestra del navegador per veure l'efecte. Quan la pantalla tinga menys de 600 píxels d'amplada, fa que les dues columnes s'apilen una sobre l'altra en lloc d'estar una al costat de l'altra.

```
<!DOCTYPE html>
<html>
<head>
<style>
* {
  box-sizing: border-box;
}

input[type=text], select, textarea {
  width: 100%;
  padding: 12px;
  border: 1px solid #ccc;
  border-radius: 4px;
  resize: vertical;
}

label {
  padding: 12px 12px 12px 0;
  display: inline-block;
}

input[type=submit] {
  background-color: #04AA6D;
  color: white;
  padding: 12px 20px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  float: right;
}

input[type=submit]:hover {
  background-color: #45a049;
}

.container {
  border-radius: 5px;
  background-color: #f2f2f2;
  padding: 20px;
}

.col-25 {
  float: left;
  width: 25%;
  margin-top: 6px;
}

.col-75 {
  float: left;
  width: 75%;
  margin-top: 6px;
}

/* Clear floats after the columns */
.row::after {
```

```

        content: "";
        display: table;
        clear: both;
    }

/* Responsive layout - when the screen is less than 600px wide, make the
two columns stack on top of each other instead of next to each other */
@media screen and (max-width: 600px) {
    .col-25, .col-75, input[type=submit] {
        width: 100%;
        margin-top: 0;
    }
}
</style>
</head>
<body>

<h2>Responsive Form</h2>
<p>Resize the browser window to see the effect. When the screen is less
than 600px wide, make the two columns stack on top of each other instead of
next to each other.</p>

<div class="container">
    <form action="/action_page.php">
        <div class="row">
            <div class="col-25">
                <label for="fname">First Name</label>
            </div>
            <div class="col-75">
                <input type="text" id="fname" name="firstname" placeholder="Your
name...">
            </div>
        </div>
        <div class="row">
            <div class="col-25">
                <label for="lname">Last Name</label>
            </div>
            <div class="col-75">
                <input type="text" id="lname" name="lastname" placeholder="Your last
name...">
            </div>
        </div>
        <div class="row">
            <div class="col-25">
                <label for="country">Country</label>
            </div>
            <div class="col-75">
                <select id="country" name="country">
                    <option value="australia">Australia</option>
                    <option value="canada">Canada</option>
                    <option value="usa">USA</option>
                </select>
            </div>
        </div>
        <div class="row">
            <div class="col-25">
                <label for="subject">Subject</label>
            </div>
            <div class="col-75">
                <textarea id="subject" name="subject" placeholder="Write something..
style="height:200px"></textarea>

```

```

        </div>
    </div>
    <br>
    <div class="row">
        <input type="submit" value="Submit">
    </div>
    </form>
</div>

</body>
</html>

```

37. Comptadors CSS

Numeració automàtica amb comptadors

Els comptadors CSS són com "variables". Els valors de les variables es poden incrementar mitjançant regles CSS (que farà un seguiment de quantes vegades s'utilitzen).

Per treballar amb comptadors CSS utilitzarem les propietats següents:

- **counter-reset** - Crea o reinicia un comptador
- **counter-increment** - Incrementa un valor de comptador
- **content** - Insereix contingut generat
- **counter()** o **counters()** funció: afegeix el valor d'un comptador a un element

Per utilitzar un comptador CSS, primer s'ha de crear amb **counter-reset**.

L'exemple següent crea un comptador per a la pàgina (al selector del cos), després augmenta el valor del comptador per a cada element `<h2>` i afegeix "Secció < valor del comptador >:" al començament de cada element `<h2>`. Prova aquest codi:

```

<!DOCTYPE html>
<html>
<head>
<style>
body {
    counter-reset: section;
}

h2::before {
    counter-increment: section;
    content: "Secció " counter(section) ": ";
}
</style>
</head>
<body>

<h1>Using CSS Counters</h1>

<h2>HTML Tutorial</h2>
<h2>CSS Tutorial</h2>
<h2>JavaScript Tutorial</h2>
<h2>Python Tutorial</h2>
<h2>SQL Tutorial</h2>

</body>
</html>

```

Comptadors de nidificació

L'exemple següent crea un comptador per a la pàgina (secció) i un comptador per a cada element `<h1>` (subsecció). El comptador de "secció" es comptarà per a cada element `<h1>` amb "<valor del comptador de secció>.", i el comptador de "subsecció" es comptarà per a cada element `<h2>` amb "<valor del comptador de secció>.>.<valor del comptador de subsecció>":

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
    counter-reset: section;
}

h1 {
    counter-reset: subsection;
}

h1::before {
    counter-increment: section;
    content: "Section " counter(section) ". ";
}

h2::before {
    counter-increment: subsection;
    content: counter(section) "." counter(subsection) " ";
}
</style>
</head>
<body>

<h1>HTML/CSS Tutorials</h1>
<h2>HTML</h2>
<h2>CSS</h2>
<h2>Bootstrap</h2>
<h2>W3.CSS</h2>

<h1>Scripting Tutorials</h1>
<h2>JavaScript</h2>
<h2>jQuery</h2>
<h2>React</h2>

<h1>Programming Tutorials</h1>
<h2>Python</h2>
<h2>Java</h2>
<h2>C++</h2>

</body>
</html>
```

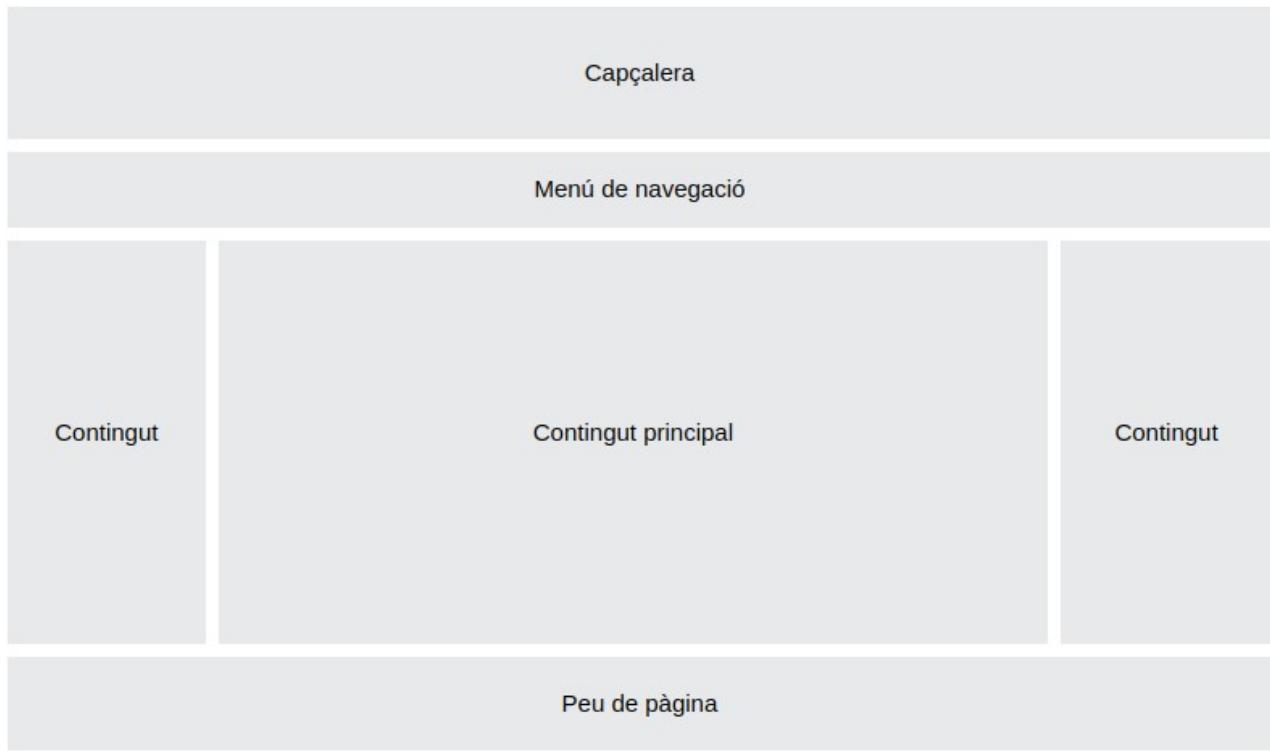
Propietats del comptador CSS

| Property | Description |
|--------------------------|---|
| <u>content</u> | Used with the ::before and ::after pseudo-elements, to insert generated content |
| <u>counter-increment</u> | Increments one or more counter values |
| <u>counter-reset</u> | Creates or resets one or more counters |

[counter\(\)](#) Returns the current value of the named counter

38. Disseny del lloc web CSS

Un lloc web sovint es divideix en capçaleres, menús, contingut i peu de pàgina:



Hi ha molts dissenys diferents per triar. No obstant això, l'estruccura anterior és una de les més habituals i la veurem més de prop en aquest tutorial.

Capçalera

Normalment, una capçalera es troba a la part superior del lloc web (o just baix d'un menú de navegació superior). Sovint conté un logotip o el nom del lloc web:

Barra de navegació

Una barra de navegació conté una llista d'enllaços per ajudar els visitants a navegar pel vostre lloc web:

Contingut

El disseny d'aquesta secció, sovint depèn dels usuaris objectiu. El disseny més comú és un (o combinant-los) dels següents:

- **1 columna** (sovint s'utilitza per a navegadors mòbils)
- **2 columnes** (sovint s'utilitza per a tauletes i ordinadors portàtils)
- **Disseny de 3 columnes** (només s'utilitza per a escriptoris)

Crearem un disseny de 3 columnes i el canviarem a un disseny d'1 columna en pantalles més xicotetes:

Consell: per crear un disseny de 2 columnes, canvieu l'amplada al 50%. Per crear un disseny de 4 columnes, utilitzeu 25%, etc.

Columnes desiguals

El contingut principal és la part més gran i important del vostre lloc.

És comú amb amplades de columnes **desiguals**, de manera que la major part de l'espai es reserva per al contingut principal. El contingut secundari (si n'hi ha) s'utilitza sovint com a navegació alternativa o per especificar informació rellevant per al contingut principal. Canvieu les amplades com vulgueu, només recordeu que hauria de sumar fins al 100% en total:

Peu de pàgina

El peu de pàgina es col·loca a la part inferior de la pàgina. Sovint conté informació com ara els drets d'autor i la informació de contacte:

Disseny responsiu del lloc web

Mira aquest disseny web responsiu, que varia entre dues columnes i columnes d'amplada completa segons l'amplada de la pantalla:

```
<!DOCTYPE html>
<html>
<head>
<style>
* {
  box-sizing: border-box;
}

body {
  font-family: Arial;
  padding: 10px;
  background: #f1f1f1;
}

/* Header/Blog Title */
.header {
  padding: 30px;
  text-align: center;
  background: white;
}

.header h1 {
  font-size: 50px;
}

/* Style the top navigation bar */
.topnav {
  overflow: hidden;
  background-color: #333;
}

/* Style the topnav links */
.topnav a {
  float: left;
  display: block;
```

```

color: #f2f2f2;
text-align: center;
padding: 14px 16px;
text-decoration: none;
}

/* Change color on hover */
.topnav a:hover {
background-color: #ddd;
color: black;
}

/* Create two unequal columns that floats next to each other */
/* Left column */
.leftcolumn {
float: left;
width: 75%;
}

/* Right column */
.rightcolumn {
float: left;
width: 25%;
background-color: #f1f1f1;
padding-left: 20px;
}

/* Fake image */
.fakeimg {
background-color: #aaa;
width: 100%;
padding: 20px;
}

/* Add a card effect for articles */
.card {
background-color: white;
padding: 20px;
margin-top: 20px;
}

/* Clear floats after the columns */
.row::after {
content: "";
display: table;
clear: both;
}

/* Footer */
.footer {
padding: 20px;
text-align: center;
background: #ddd;
margin-top: 20px;
}

/* Responsive layout - when the screen is less than 800px wide, make the
two columns stack on top of each other instead of next to each other */
@media screen and (max-width: 800px) {
.leftcolumn, .rightcolumn {
width: 100%;
```

```

        padding: 0;
    }
}

/* Responsive layout - when the screen is less than 400px wide, make the
navigation links stack on top of each other instead of next to each other
*/
@media screen and (max-width: 400px) {
    .topnav a {
        float: none;
        width: 100%;
    }
}
</style>
</head>
<body>

<div class="header">
    <h1>My Website</h1>
    <p>Resize the browser window to see the effect.</p>
</div>

<div class="topnav">
    <a href="#">Link</a>
    <a href="#">Link</a>
    <a href="#">Link</a>
    <a href="#" style="float:right">Link</a>
</div>

<div class="row">
    <div class="leftcolumn">
        <div class="card">
            <h2>TITLE HEADING</h2>
            <h5>Title description, Dec 7, 2017</h5>
            <div class="fakeimg" style="height:200px;">Image</div>
            <p>Some text..</p>
            <p>Sunt in culpa qui officia deserunt mollit anim id est laborum
consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et
dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation
ullamco.</p>
        </div>
        <div class="card">
            <h2>TITLE HEADING</h2>
            <h5>Title description, Sep 2, 2017</h5>
            <div class="fakeimg" style="height:200px;">Image</div>
            <p>Some text..</p>
            <p>Sunt in culpa qui officia deserunt mollit anim id est laborum
consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et
dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation
ullamco.</p>
        </div>
    </div>
    <div class="rightcolumn">
        <div class="card">
            <h2>About Me</h2>
            <div class="fakeimg" style="height:100px;">Image</div>
            <p>Some text about me in culpa qui officia deserunt mollit anim..</p>
        </div>
        <div class="card">
            <h3>Popular Post</h3>
            <div class="fakeimg"><p>Image</p></div>

```

```

<div class="fakeimg"><p>Image</p></div>
<div class="fakeimg"><p>Image</p></div>
</div>
<div class="card">
  <h3>Follow Me</h3>
  <p>Some text...</p>
</div>
</div>
</div>

<div class="footer">
  <h2>Footer</h2>
</div>

</body>
</html>

```

39. Unitats CSS

CSS té diverses unitats diferents per expressar una longitud.

Moltes propietats CSS prenen valors de "longitud", com ara **width**, **margin**, **padding**, **font-size**, etc.

La longitud és un nombre seguit d'una unitat de longitud, com ara **10px**, **2em**, etc.

Nota: No pot aparèixer un espai en blanc entre el número i la unitat. Tanmateix, si el valor és **0**, la unitat es pot ometre.

Per a algunes propietats CSS, es permeten longituds negatives.

Hi ha dos tipus d'unitats de longitud: **absolutes** i **relatives**.

Longituds absolutes

Les unitats de longitud absolutes són fixes i una longitud expressada en qualsevol d'aquestes apareixerà exactament amb aquesta mida.

No es recomana utilitzar unitats de longitud absoluta a la pantalla, perquè les mides de la pantalla varien molt. Tanmateix, es poden utilitzar si es coneix el mitjà d'eixida, com per exemple per a la disposició d'impressió.

Unit Description

| | |
|----|------------------------------|
| cm | centimeters |
| mm | millimeters |
| in | inches (1in = 96px = 2.54cm) |
| px | pixels (1px = 1/96th of 1in) |
| pt | points (1pt = 1/72 of 1in) |
| pc | picas (1pc = 12 pt) |

Longituds relatives

Les unitats de longitud relativa especificuen una longitud relativa a una altra propietat de longitud. Les unitats de longitud relativa s'escalen millor entre diferents mitjans de representació.

Unit Description

| | |
|------|---|
| em | Relative to the font-size of the element (2em means 2 times the size of the current font) |
| ex | Relative to the x-height of the current font (rarely used) |
| ch | Relative to width of the "0" (zero) |
| rem | Relative to font-size of the root element |
| vw | Relative to 1% of the width of the viewport* |
| vh | Relative to 1% of the height of the viewport* |
| vmin | Relative to 1% of viewport's* smaller dimension |
| vmax | Relative to 1% of viewport's* larger dimension |
| % | Relative to the parent element |

Consell: les unitats em i rem són pràctiques per crear un disseny perfectament escalable!

* Viewport = la mida de la finestra del navegador. Si el visor té 50 cm d'ample, 1vw = 0,5 cm.

40. Especificitat CSS

Què és l'especificitat?

Si hi ha dues o més regles CSS que apunten al mateix element, el selector amb el valor d'especificitat més alt "guanyarà" i la seva declaració d'estil s'aplicarà a aquest element HTML.

Penseu en l'especificitat com una puntuació/ranking que determina quina declaració d'estil s'aplica finalment a un element.

Mireu els exemples següents:

Exemple 1: En aquest exemple, hem utilitzat l'element "p" com a selector i hem especificat un color roig per a aquest element. El text serà roig:

```
<html>
<head>
  <style>
    p {color: red;}
  </style>
</head>
<body>

<p>Hello World!</p>

</body>
</html>
```

Ara, mireu l'exemple 2: En aquest exemple, hem afegit un selector de classe (anomenat "prova") i hem especificat un color verd per a aquesta classe. El text ara serà verd (tot i que hem especificat un color roig per al selector d'elements "p"). Això és perquè el selector de classe té una prioritat més alta:

```
<html>
<head>
  <style>
    .test {color: green;}
    p {color: red;}
  </style>
```

```

</head>
<body>

<p class="test">Hello World!</p>

</body>
</html>

```

Ara, mireu l'exemple 3: En aquest exemple, hem afegit el selector d'identificació (anomenat "demo"). El text ara serà blau, perquè el selector d'identificació té una prioritat més alta:

```

<html>
<head>
  <style>
    #demo {color: blue;}
    .test {color: green;}
    p {color: red;}
  </style>
</head>
<body>

<p id="demo" class="test">Hello World!</p>

</body>
</html>

```

Ara, mireu l'exemple 4: En aquest exemple, hem afegit un estil en línia per a l'element "p". El text ara serà de color rosa, perquè l'estil en línia té la màxima prioritat:

```

<html>
<head>
  <style>
    #demo {color: blue;}
    .test {color: green;}
    p {color: red;}
  </style>
</head>
<body>

<p id="demo" class="test" style="color: pink;">Hello World!</p>

</body>
</html>

```

Jerarquia d'especificitat

Cada selector CSS té el seu lloc a la jerarquia d'especificitat.

Hi ha quatre categories que defineixen el nivell d'especificitat d'un selector:

- Estils en línia** - Exemple: `<h1 style="color: pink;">`
- ID** - Exemple: `#navbar`
- Classes, pseudo-classes, selectors d'atributs** - Exemple: `.test, :hover, [href]`
- Elements i pseudoelements** - Exemple: `h1, ::before`

Especificitat igual: guanya l'última regla: si la mateixa regla s'escriu dues vegades al full d'estil extern, guanya l'última regla.

Els selectors d'identificació tenen una especificitat més alta que els selectors d'atributs: mireu les tres línies de codi següents. Exemple:

```
div#a {background-color: green;}  
#a {background-color: yellow;}  
div[id=a] {background-color: blue;}
```

la primera regla és més específica que les altres dues i, per tant, s'aplicarà.

Els selectors contextuales són més específics que un selector d'element únic: el full d'estil incrustat està més a prop de l'element que s'ha d'estilitzar. Així que en la següent situació:

```
From external CSS file:  
#content h1 {background-color: red;}  
  
In HTML file:  
<style>  
#content h1 {background-color: yellow;}  
</style>
```

s'aplicarà aquesta darrera norma.

Un selector de classe supera qualsevol selector d'elements: un selector de classe com ara .intro bat h1, p, div, etc.:

El selector universal (*) i els valors heretats tenen una especificitat de 0 - El selector universal (*) i els valors heretats s'ignoren!

41. Funcions matemàtiques

Les funcions matemàtiques CSS permeten utilitzar expressions matemàtiques com a valors de propietat. Aquí, explicarem les funcions calc(), max() i min().

La funció calc().

La funció calc() realitza un càlcul que s'utilitzarà com a valor de propietat.

Sintaxi CSS

calc(expression)

Value Description

| | |
|------------|--|
| expression | Required. A mathematical expression. The result will be used as the value. The following operators can be used: + - * / |
|------------|--|

Un exemple. Utilitzeu calc() per calcular l'amplada d'un element <div>:

```
#div1 {  
    position: absolute;  
    left: 50px;  
    width: calc(100% - 100px);  
    border: 1px solid black;  
    background-color: yellow;  
    padding: 5px;  
}
```

La funció max().

La funció **max()** utilitza el valor més gran, d'una llista de valors separats per comes, com a valor de propietat.

Sintaxi CSS

`max(value1, value2, ...)`

| Value | Description |
|----------------------------|--|
| <i>Value1, value2, ...</i> | Required. A list of comma-separated values - where the largest value is chosen |

Un exemple. Utilitzeu max() per establir l'amplada de #div1 al valor més gran, 50% o 300px:

```
#div1 {  
    background-color: yellow;  
    height: 100px;  
    width: max(50%, 300px);  
}
```

La funció min().

La funció **min()** utilitza el valor més xicotet, d'una llista de valors separats per comes, com a valor de propietat.

Sintaxi CSS

`min(value1, value2, ...)`

| Value | Description |
|----------------------------|---|
| <i>Value1, value2, ...</i> | Required. A list of comma-separated values - where the smallest value is chosen |

Totes les funcions matemàtiques CSS

| Function | Description |
|------------------------|---|
| calc() | Allows you to perform calculations to determine CSS property values |
| max() | Uses the largest value, from a comma-separated list of values, as the property value |
| min() | Uses the smallest value, from a comma-separated list of values, as the property value |

Tema 4: JavaScript HTML DOM



0. Introducció

1. Mètodes HTML DOM
 2. Document DOM HTML JavaScript
 3. Elements DOM HTML JavaScript
 4. JavaScript HTML DOM - Canvi d'HTML
 5. Formularis JavaScript
 6. JavaScript HTML DOM - Canvi de CSS
 7. JavaScript HTML DOM Animació
 8. Esdeveniments DOM JavaScript HTML
 9. JavaScript HTML DOM EventListener
 10. JavaScript HTML Navegació DOM
 11. Elements DOM HTML JavaScript (nodes)
-

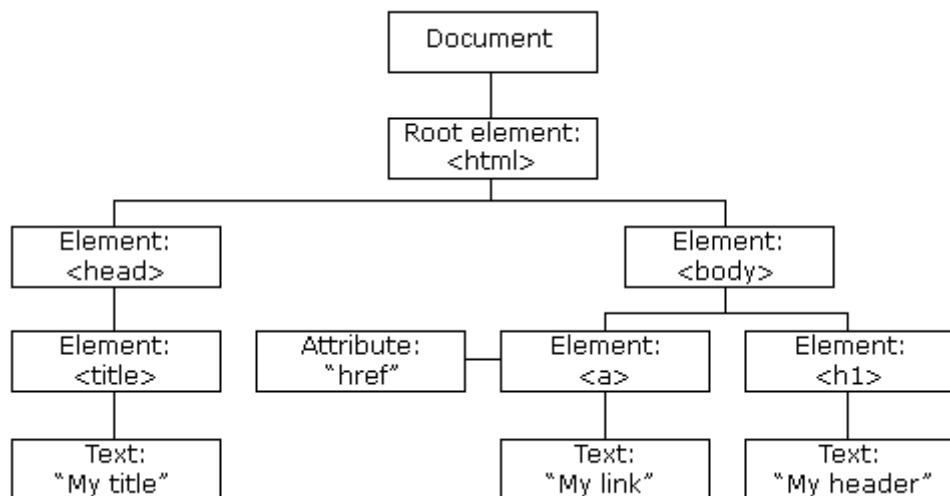
0. Introducció

El DOM HTML (Model d'objectes de document)

Quan es carrega una pàgina web, el navegador crea un Model d'objecte de document de la pàgina.

El model **HTML DOM** es construeix com un arbre d' **objectes** :

L'arbre d'objectes HTML DOM



Amb el model d'objectes, JavaScript obté tot el poder que necessita per crear HTML dinàmic:

- JavaScript pot canviar tots els elements HTML de la pàgina
- JavaScript pot canviar tots els atributs HTML de la pàgina
- JavaScript pot canviar tots els estils CSS de la pàgina
- JavaScript pot eliminar elements i atributs HTML existents
- JavaScript pot afegir nous elements i atributs HTML
- JavaScript pot reaccionar a tots els esdeveniments HTML existents a la pàgina
- JavaScript pot crear nous esdeveniments HTML a la pàgina

Què Aprendràs

En els següents punts del tema aprendràs:

- Com canviar el contingut dels elements HTML
- Com canviar l'estil (CSS) dels elements HTML
- Com reaccionar als esdeveniments HTML DOM
- Com afegir i eliminar elements HTML

Què és el DOM?

El DOM és un estàndard del W3C (World Wide Web Consortium).

El DOM defineix un estàndard per accedir als documents:

"El Document Object Model (DOM) del W3C és una plataforma i una interfície neutral en el llenguatge que permet als programes i scripts accedir i actualitzar dinàmicament el contingut, l'estructura i l'estil d'un document".

L'estàndard W3C DOM es divideix en 3 parts diferents:

- Core DOM: model estàndard per a tots els tipus de documents
- XML DOM: model estàndard per a documents XML
- HTML DOM: model estàndard per a documents HTML

Què és el DOM HTML?

El DOM HTML és un model **d'objectes** estàndard i **una interfície de programació** per a HTML. Defineix:

- Els elements HTML com a **objectes**
- Les **proprietats** de tots els elements HTML
- Els **mètodes** per accedir a tots els elements HTML
- Els **esdeveniments** per a tots els elements HTML

En altres paraules: **el DOM HTML és un estàndard per obtenir, canviar, afegir o suprimir elements HTML.**

1. Mètodes HTML DOM

Els mètodes HTML DOM són **accions** que podeu realitzar (en elements HTML).

Les propietats HTML DOM són **valors** (d'elements HTML) que podeu establir o canviar.

La interfície de programació DOM

Es pot accedir al DOM HTML amb JavaScript (i amb altres llenguatges de programació).

Al DOM, tots els elements HTML es defineixen com a **objectes**.

La interfície de programació són les propietats i mètodes de cada objecte.

Una **proprietat** és un valor que podeu obtenir o establir (com canviar el contingut d'un element HTML).

Un **mètode** és una acció que podeu fer (com afegir o suprimir un element HTML).

L'exemple següent canvia el contingut (el innerHTML) de l'element <p> amb id="demo":

```
<!DOCTYPE html>
<html>
  <body>
    <h2>My First Page</h2>
    <p id="demo"></p>
    <script>
      document.getElementById("demo").innerHTML = "Hello World!";
    </script>
  </body>
</html>
```

A l'exemple anterior, getElementById és un **mètode**, mentre que innerHTML és una **proprietat**.

El mètode getElementById

La manera més habitual d'accendir a un element HTML és utilitzar l'id de l'element.

A l'exemple anterior, el getElementById és un mètode que utilitzat id="demo" per trobar l'element.

La propietat innerHTML

La manera més senzilla d'obtenir el contingut d'un element és utilitzant la propietat innerHTML.

La propietat innerHTML és útil per obtenir o substituir el contingut dels elements HTML.

2. Document DOM HTML JavaScript

L'objecte de document HTML DOM és el propietari de tots els altres objectes de la vostra pàgina web.

L'objecte de document HTML DOM

L'objecte document representa la vostra pàgina web.

Si voleu accedir a qualsevol element d'una pàgina HTML, sempre comenceu per accedir a l'objecte del document.

A continuació es mostren alguns exemples de com podeu utilitzar l'objecte document per accedir i manipular HTML.

Trobar elements HTML

| Method | Description |
|--|-------------------------------|
| <code>document.getElementById(id)</code> | Find an element by element id |
| <code>document.getElementsByTagName(name)</code> | Find elements by tag name |
| <code>document.getElementsByClassName(name)</code> | Find elements by class name |

Canvi d'elements HTML

| Method | Description |
|---|---|
| <code>element.innerHTML = new html content</code> | Change the inner HTML of an element |
| <code>element.setAttribute = new value</code> | Change the attribute value of an HTML element |
| <code>element.style.property = new style</code> | Change the style of an HTML element |
| <code>element.setAttribute(attribute, value)</code> | Change the attribute value of an HTML element |

Afegir i eliminar elements

| Method | Description |
|--|-------------------------|
| <code>document.createElement(element)</code> | Create an HTML element |
| <code>document.removeChild(element)</code> | Remove an HTML element |
| <code>document.appendChild(element)</code> | Add an HTML element |
| <code>document.replaceChild(new, old)</code> | Replace an HTML element |

| | |
|-----------------------------------|-----------------------------------|
| <code>document.write(text)</code> | Write into the HTML output stream |
|-----------------------------------|-----------------------------------|

Afegir gestors d'esdeveniments

| Method | Description |
|---|---|
| <code>document.getElementById(id).onclick = function(){code}</code> | Adding event handler code to an onclick event |

Trobar objectes HTML

El nivell 1 de DOM HTML (1998), va definir 11 objectes HTML, col·leccions d'objectes i propietats. Aquests encara són vàlids en HTML5.

Més tard, a HTML DOM Nivell 3, es van afegir més objectes, col·leccions i propietats.

| Property | Description | DOM |
|---|--|-----|
| <code>document.anchors</code> | Returns all <a> elements that have a name attribute | 1 |
| <code>document.applets</code> | Deprecated | 1 |
| <code>document.baseURI</code> | Returns the absolute base URI of the document | 3 |
| <code>document.body</code> | Returns the <body> element | 1 |
| <code>document.cookie</code> | Returns the document's cookie | 1 |
| <code>document.doctype</code> | Returns the document's doctype | 3 |
| <code>document.documentElement</code> | Returns the <html> element | 3 |
| <code>document.documentElement</code> | Returns the mode used by the browser | 3 |
| <code>document.documentElement</code> | Returns the URI of the document | 3 |
| <code>document.domain</code> | Returns the domain name of the document server | 1 |
| <code>document.domConfig</code> | Obsolete. | 3 |
| <code>document.embeds</code> | Returns all <embed> elements | 3 |
| <code>document.forms</code> | Returns all <form> elements | 1 |
| <code>document.head</code> | Returns the <head> element | 3 |
| <code>document.images</code> | Returns all elements | 1 |
| <code>document.implementation</code> | Returns the DOM implementation | 3 |
| <code>document.inputEncoding</code> | Returns the document's encoding (character set) | 3 |
| <code>document.lastModified</code> | Returns the date and time the document was updated | 3 |
| <code>document.links</code> | Returns all <area> and <a> elements that have a href attribute | 1 |
| <code>document.readyState</code> | Returns the (loading) status of the document | 3 |
| <code>document.referrer</code> | Returns the URI of the referrer (the linking document) | 1 |
| <code>document.scripts</code> | Returns all <script> elements | 3 |
| <code>document.strictErrorChecking</code> | Returns if error checking is enforced | 3 |
| <code>document.title</code> | Returns the <title> element | 1 |

3. Elements DOM HTML JavaScript

Aquest punt us ensenya com trobar i accedir als elements HTML en una pàgina HTML.

Trobar elements HTML

Sovint, amb JavaScript, voleu manipular elements HTML.

Per fer-ho, primer heu de trobar els elements. Hi ha diverses maneres de fer-ho:

- Cercar elements HTML per id
- Cercar elements HTML pel nom de l'etiqueta
- Cercar elements HTML pel nom de classe
- Trobar elements HTML mitjançant selectors CSS
- Trobar elements HTML per col·leccions d'objectes HTML

Trobar l'element HTML per Id

La manera més senzilla de trobar un element HTML al DOM és utilitzant l'element id.

Aquest exemple troba l'element amb id="intro":

```
<!DOCTYPE html>
<html>
<body>
    <h2>JavaScript HTML DOM</h2>
    <p id="intro">Finding HTML Elements by Id</p>
    <p>This example demonstrates the <b>getElementsById</b> method.</p>
    <p id="demo"></p>
    <script>
        const element = document.getElementById("intro");
        document.getElementById("demo").innerHTML =
            "The text from the intro paragraph is: " + element.innerHTML;
    </script>
</body>
</html>
```

Si es troba l'element, el mètode retornarà l'element com a objecte (a l'element).

Si no es troba l'element, l'element contindrà null.

Cerca d'elements HTML pel nom de l'etiqueta

Aquest exemple troba tots els <p> elements. Exemple:

```
const element = document.getElementsByTagName("p");
```

Aquest exemple troba l'element amb id="main", i després troba tots els elements <p> dins "main":

```
<!DOCTYPE html>
<html>
<body>
    <h2>JavaScript HTML DOM</h2>
    <p>Finding HTML Elements by Tag Name.</p>
    <p>This example demonstrates the <b>getElementsByTagName</b> method.</p>
```

```

<p id="demo"></p>

<script>
  const element = document.getElementsByTagName("p");
  document.getElementById("demo").innerHTML = 'The text in
  first paragraph (index 0) is: ' + element[0].innerHTML;
</script>
</body>
</html>

```

Si es troba l'element, el mètode retornarà l'element com a objecte (a l'element).

Si no es troba l'element, l'element contindrà null.

Trobar elements HTML per nom de classe

Si voleu trobar tots els elements HTML amb el mateix nom de classe, utilitzeu `getElementsByClassName()`.

Aquest exemple retorna una llista de tots els elements amb `class="intro"`.

```

const x = document.getElementsByClassName("intro");

<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript HTML DOM</h2>
  <p>Finding HTML Elements by Class Name.</p>
  <p class="intro">Hello World!</p>
  <p class="intro">This example demonstrates the
  <b>getElementsByClassName</b> method.</p>
  <p id="demo"></p>
  <script>
    const x = document.getElementsByClassName("intro");
    document.getElementById("demo").innerHTML =
      'The first paragraph (index 0) with class="intro" is: '
      + x[0].innerHTML;
  </script>
</body>
</html>

```

Trobar elements HTML mitjançant selectors CSS

Si voleu trobar tots els elements HTML que coincideixen amb un selector CSS especificat (identificador, noms de classe, tipus, atributs, valors d'atributs, etc.), utilitzeu el mètode `querySelectorAll()`.

Aquest exemple retorna una llista de tots els elements `<p>` amb `class="intro"`.

```

const x = document.querySelectorAll("p.intro");

<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript HTML DOM</h2>
  <p>Finding HTML Elements by Query Selector</p>
  <p class="intro">Hello World!</p>
  <p class="intro">This example demonstrates the
  <b>querySelectorAll</b> method.</p>
  <p id="demo"></p>

```

```

<script>
    const x = document.querySelectorAll("p.intro");
    document.getElementById("demo").innerHTML =
        'The first paragraph (index 0) with class="intro" is: '
        + x[0].innerHTML;
</script>
</body>
</html>

```

Trobar elements HTML per col·leccions d'objectes HTML

Aquest exemple cerca l'element de formulari amb id="frm1", a la col·lecció de formularis i mostra tots els valors de l'element:

```

<!DOCTYPE html>
<html>
<body>
    <h2>JavaScript HTML DOM</h2>
    <p>Finding HTML Elements Using <b>document.forms</b>. </p>
    <form id="frm1" action="/action_page.php">
        First name: <input type="text" name="fname" value="Donald"><br>
        Last name: <input type="text" name="lname" value="Duck"><br><br>
        <input type="submit" value="Submit">
    </form>
    <p>These are the values of each element in the form:</p>
    <p id="demo"></p>
    <script>
        const x = document.forms["frm1"];
        let text = "";
        for (let i = 0; i < x.length ;i++) {
            text += x.elements[i].value + "<br>";
        }
        document.getElementById("demo").innerHTML = text;
    </script>
</body>
</html>

```

Els següents objectes HTML (i col·leccions d'objectes) també són accessibles:

- [document.anchors](#)
- [document.body](#)
- [document.documentElement](#)
- [document.embeds](#)
- [document.forms](#)
- [document.head](#)
- [document.images](#)
- [document.links](#)
- [document.scripts](#)
- [document.title](#)

4. JavaScript HTML DOM - Canvi d'HTML

El DOM HTML permet a JavaScript canviar el contingut dels elements HTML. La manera més senzilla de modificar el contingut d'un element HTML és utilitzant la propietat innerHTML.

Per canviar el contingut d'un element HTML, utilitzeu aquesta sintaxi:

```
document.getElementById(id).innerHTML = new HTML;
```

Aquest exemple canvia el contingut d'un element <p>:

```
<html>
<body>
  <p id="p1">Hello World!</p>
  <script>
    document.getElementById("p1").innerHTML = "New text!";
  </script>
</body>
</html>
```

Exemple explicat:

- El document HTML anterior conté un element <p> amb id="p1"
- Utilitzem el DOM HTML per obtenir l'element amb id="p1"
- Un JavaScript canvia el contingut (innerHTML) d'aquest element a "New text!"

Aquest exemple canvia el contingut d'un element <h1>:

```
<!DOCTYPE html>
<html>
<body>
  <h1 id="id01">Old Heading</h1>
  <script>
    const element = document.getElementById("id01");
    element.innerHTML = "New Heading";
  </script>
</body>
</html>
```

Canviar el valor d'un atribut

Per canviar el valor d'un atribut HTML, utilitzeu aquesta sintaxi:

```
document.getElementById(id).attribute = new value;
```

Aquest exemple canvia el valor de l'atribut src d'un element :

```
<!DOCTYPE html>
<html>
<body>
  
  <script>
    document.getElementById("myImage").src = "landscape.jpg";
  </script>
</body>
</html>
```

Exemple explicat:

- El document HTML anterior conté un element amb id="myImage"
- Utilitzem el DOM HTML per obtenir l'element amb id="myImage"
- Un JavaScript canvia l' src atribut d'aquest element de "smiley.gif" a "landscape.jpg"

Contingut HTML dinàmic

JavaScript pot crear contingut HTML dinàmic:

```

<!DOCTYPE html>
<html>
<body>
    <p id="demo"></p>
    <script>
        document.getElementById("demo").innerHTML = "Date : " + Date();
    </script>
</body>
</html>

```

document.write()

En JavaScript, `document.write()` es pot utilitzar per escriure directament al flux d'eixida HTML:

```

<!DOCTYPE html>
<html>
<body>
    <p>Bla bla bla</p>
    <script>
        document.write(Date());
    </script>
    <p>Bla bla bla</p>
</body>
</html>

```

5. Formularis JavaScript

Validació del formulari JavaScript

La validació del formulari HTML es pot fer mitjançant JavaScript.

Si un camp de formulari (`fname`) està buit, aquesta funció alerta amb un missatge i retorna `false` per evitar que s'envie el formulari:

Exemple de JavaScript

```

function validateForm() {
    let x = document.forms["myForm"]["fname"].value;
    if (x == "") {
        alert("Name must be filled out");
        return false;
    }
}

```

La funció es pot cridar quan s'envia el formulari:

Exemple de formulari HTML

```

<!DOCTYPE html>
<html>
<head>
    <script>
        function validateForm() {
            let x = document.forms["myForm"]["fname"].value;
            if (x == "") {
                alert("Name must be filled out");
                return false;
            }
        }
    </script>
</head>

```

```

<body>
    <h2>JavaScript Validation</h2>
    <form name="myForm" action="/action_page.php" onsubmit="return
validateForm()" method="post">
        Name: <input type="text" name="fname">
        <input type="submit" value="Submit">
    </form>

</body>
</html>

```

JavaScript pot validar l'entrada numèrica

JavaScript s'utilitza sovint per validar l'entrada numèrica:

Introduïu un número entre 1 i 10. Exemple:

```

<!DOCTYPE html>
<html>
<body>
    <h2>JavaScript Validation</h2>

    <p>Please input a number between 1 and 10:</p>

    <input id="numb">

    <button type="button" onclick="myFunction()">Submit</button>

    <p id="demo"></p>

    <script>
        function myFunction() {
            // Get the value of the input field with id="numb"
            let x = document.getElementById("numb").value;
            // If x is Not a Number or less than one or greater than 10
            let text;
            if (isNaN(x) || x < 1 || x > 10) {
                text = "Input not valid";
            } else {
                text = "Input OK";
            }
            document.getElementById("demo").innerHTML = text;
        }
    </script>

</body>
</html>

```

Validació automàtica de formularis HTML

La validació del formulari HTML es pot realitzar automàticament pel navegador:

Si un camp de formulari (fname) està buit, l'atribut required impedeix que aquest formulari s'envie:

Exemple de formulari HTML

```

<!DOCTYPE html>
<html>
<body>
    <h2>JavaScript Validation</h2>
    <form action="/action_page.php" method="post">

```

```

<input type="text" name="fname" required>
<input type="submit" value="Submit">
</form>
<p>If you click submit, without filling out the text field,
your browser will display an error message.</p>
</body>
</html>

```

Validació de dades

La validació de dades és el procés per garantir que l'entrada de l'usuari siga neta, correcta i útil.

Les tasques típiques de validació són:

- l'usuari ha emplenat tots els camps obligatoris?
- l'usuari ha introduït una data vàlida?
- l'usuari ha introduït text en un camp numèric?

Molt sovint, l'objectiu de la validació de dades és garantir l'entrada correcta de l'usuari.

La validació es pot definir per molts mètodes diferents i desplegar-se de moltes maneres diferents.

La validació del costat del servidor la realitza un servidor web, després que l'entrada s'haja enviat al servidor.

La validació del costat del client la realitza un navegador web, abans que l'entrada s'envie a un servidor web.

Validació de restriccions HTML

HTML5 va introduir un nou concepte de validació HTML anomenat **validació de restriccions**.

La validació de restriccions HTML es basa en:

- Validació de restriccions **Atributs d'entrada HTML**
- **Pseudoselectors CSS** de validació de restriccions
- Validació de restriccions **Propietats i mètodes DOM**

Atributs d'entrada HTML de validació de restriccions

| Attribute | Description |
|-----------|---|
| disabled | Specifies that the input element should be disabled |
| max | Specifies the maximum value of an input element |
| min | Specifies the minimum value of an input element |
| pattern | Specifies the value pattern of an input element |
| required | Specifies that the input field requires an element |
| type | Specifies the type of an input element |

Pseudoselectors CSS de validació de restriccions

| Selector | Description |
|-----------|--|
| :disabled | Selects input elements with the "disabled" attribute specified |

| | |
|-----------|--|
| :invalid | Selects input elements with invalid values |
| :optional | Selects input elements with no "required" attribute specified |
| :required | Selects input elements with the "required" attribute specified |
| :valid | Selects input elements with valid values |

6. JavaScript HTML DOM - Canvi de CSS

El DOM HTML permet a JavaScript canviar l'estil dels elements HTML.

Canvi d'estil HTML

Per canviar l'estil d'un element HTML, utilitzeu aquesta sintaxi:

document.getElementById(id).style.property = new style

L'exemple següent canvia l'estil d'un <p> element. Exemple:

```
<html>
<body>
    <p id="p2">Hello World!</p>
    <script>
        document.getElementById("p2").style.color = "blue";
    </script>
    <!-- Mostrarà Hello World! En blau -->
</body>
</html>
```

Ús d'esdeveniments

El DOM HTML us permet executar codi quan es produeix un esdeveniment.

Els esdeveniments els genera el navegador quan "passen coses" als elements HTML:

- Es fa clic en un element
- La pàgina s'ha carregat
- Els camps d'entrada es canvien

Aprendràs més sobre els esdeveniments al següent punt d'aquests apunts.

Aquest exemple canvia l'estil de l'element HTML amb id="id1", quan l'usuari fa clic en un botó. Exemple:

```
<!DOCTYPE html>
<html>
    <body>
        <h1 id="id1">My Heading 1</h1>
        <button type="button"
            onclick="document.getElementById('id1').style.color = 'red'"
            Click Me!
        </button>
    </body>
</html>
```

Més exemples: visibilitat

Com fer invisible un element. Vols mostrar l'element o no?

```
<!DOCTYPE html>
<html>
```

```

<body>
    <p id="p1">
        This is a text.
    </p>
    <input type="button" value="Hide text"
    onclick="document.getElementById('p1').style.visibility='hidden'">
    <input type="button" value="Show text"
    onclick="document.getElementById('p1').style.visibility='visible'">
</body>
</html>

```

7. JavaScript HTML DOM Animació

Aprèn a crear animacions HTML amb JavaScript.

Per demostrar com crear animacions HTML amb JavaScript, utilitzarem una pàgina web senzilla. Exemple:

```

<!DOCTYPE html>
<html>
<body>
    <h1>My First JavaScript Animation</h1>
    <div id="animation">My animation will go here</div>
</body>
</html>

```

Crea un contenidor d'animació

Totes les animacions han de ser relatives a un element contenidor.

```

<div id ="container">
    <div id ="animate">My animation will go here</div>
</div>

```

Estil els elements

L'element contenidor s'ha de crear amb style = " position: relative".

L'element d'animació s'ha de crear amb style = " position: absolute".

Exemple

```

#container {
    width: 400px;
    height: 400px;
    position: relative;
    background: yellow;
}
#animate {
    width: 50px;
    height: 50px;
    position: absolute;
    background: red;
}

```

Codi d'animació

Les animacions de JavaScript es fan programant canvis graduals en l'estil d'un element.

Els canvis són cridats per un temporitzador. Quan l'interval del temporitzador és xicotet, l'animació sembla contínua.

El codi bàsic és:

```
id = setInterval(frame, 5);
function frame() {
    if /* test for finished */) {
        clearInterval(id);
    } else {
        /* code to change the element style */
    }
}
```

Creeu l'animació completa amb JavaScript

```
<!DOCTYPE html>
<html>
    <style>
        #container {
            width: 400px;
            height: 400px;
            position: relative;
            background: yellow;
        }
        #animate {
            width: 50px;
            height: 50px;
            position: absolute;
            background-color: red;
        }
    </style>
    <body>
        <p><button onclick="myMove()">Click Me</button></p>
        <div id ="container">
            <div id ="animate"></div>
        </div>
        <script>
            function myMove() {
                let id = null;
                const elem = document.getElementById("animate");
                let pos = 0;
                clearInterval(id);
                id = setInterval(frame, 5);
                function frame() {
                    if (pos == 350) {
                        clearInterval(id);
                    } else {
                        pos++;
                        elem.style.top = pos + "px";
                        elem.style.left = pos + "px";
                    }
                }
            }
        </script>
    </body>
</html>
```

8. Esdeveniments DOM JavaScript HTML

HTML DOM permet a JavaScript reaccionar als esdeveniments HTML

Per executar codi quan un usuari fa clic en un element, afegiu codi JavaScript a un atribut d'esdeveniment HTML:

onclick=JavaScript

Exemples d'esdeveniments HTML:

- Quan un usuari fa clic amb el ratolí
- Quan s'ha carregat una pàgina web
- Quan s'ha carregat una imatge
- Quan el ratolí es mou sobre un element
- Quan es canvia un camp d'entrada
- Quan s'envia un formulari HTML
- Quan un usuari toca una tecla

En aquest exemple, el contingut de l'element `<h1>` es modifica quan un usuari hi fa clic:

```
<!DOCTYPE html>
<html>
<body>
  <h1 onclick="this.innerHTML = 'Ooops!'">Click on this text!</h1>
</body>
</html>
```

En aquest exemple, es crida una funció des del controlador d'esdeveniments. El resultat és el mateix:

```
<!DOCTYPE html>
<html>
<body>
  <h1 onclick="changeText(this)">Click on this text!</h1>
  <script>
    function changeText(id) {
      id.innerHTML = "Ooops!";
    }
  </script>
</body>
</html>
```

Atributs d'esdeveniment HTML

Per assignar esdeveniments a elements HTML podeu utilitzar atributs d'esdeveniment.

A l'exemple s'executarà una funció anomenada `displayDate` quan es façà clic al botó.

```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript HTML Events</h2>
  <p>Click the button to display the date.</p>
  <button onclick="displayDate()">The time is?</button>
  <script>
    function displayDate() {
      document.getElementById("demo").innerHTML = Date();
    }
  </script>
  <p id="demo"></p>
</body>
</html>
```

Els esdeveniments de càrrega i descàrrega de la pàgina

Els esdeveniments onload i onunload es desencadenen quan l'usuari entra o ix de la pàgina.

L'esdeveniment onload es pot utilitzar per comprovar el tipus de navegador del visitant i la versió del navegador, i carregar la versió adequada de la pàgina web en funció de la informació.

Els esdeveniments onload i onunload es poden utilitzar per tractar les cookies.

Exemple

```
<!DOCTYPE html>
<html>
<body onload="checkCookies()">
    <h2>JavaScript HTML Events</h2>
    <p id="demo"></p>
    <script>
        function checkCookies() {
            var text = "";
            if (navigator.cookieEnabled == true) {
                text = "Cookies are enabled.";
            }
            else {
                text = "Cookies are not enabled.";
            }
            document.getElementById("demo").innerHTML = text;
        }
    </script>
</body>
</html>
```

L'esdeveniment onchange

L'esdeveniment onchange s'utilitza sovint en combinació amb la validació dels camps d'entrada.

A continuació es mostra un exemple de com utilitzar l'onchange. La uppercase() funció es cridarà quan un usuari canvie el contingut d'un camp d'entrada.

Exemple

```
<!DOCTYPE html>
<html>
<body>
    <h2>JavaScript HTML Events</h2>
    Enter your name: <input type="text" id="fname"
    onchange="uppercase()">
    <p>When you leave the input field, a function is triggered which
    transforms the input text to upper case.</p>
    <script>
        function uppercase() {
            const x = document.getElementById("fname");
            x.value = x.value.toUpperCase();
        }
    </script>
</body>
</html>
```

Els esdeveniments onmouseover i onmouseout

Els esdeveniments onmouseover i onmouseout poden utilitzar per activar una funció quan l'usuari passa el ratolí per damunt o fora d'un element HTML:

```
<!DOCTYPE html>
<html>
<body>
    <div onmouseover="mOver(this)" onmouseout="mOut(this)"
        style="background-color:#D94A38;
        width:120px;height:20px;padding:40px;">
        Mouse Over Me
    </div>
    <script>
        function mOver(obj) {
            obj.innerHTML = "Thank You"
        }
        function mOut(obj) {
            obj.innerHTML = "Mouse Over Me"
        }
    </script>
</body>
</html>
```

Els esdeveniments onmousedown, onmouseup i onclick

Els esdeveniments onmousedown, onmouseup, i onclick són totes parts d'un clic amb el ratolí. Primer quan es fa clic al botó del ratolí, s'activa l'esdeveniment onmousedown, després, quan es deixa anar el botó del ratolí, s'activa l'esdeveniment onmouseup i, finalment, quan es completa el clic del ratolí, s'activa l'esdeveniment onclick.

```
<!DOCTYPE html>
<html>
<body>
    <div onmousedown="mDown(this)" onmouseup="mUp(this)" style=
        "background-color:#D94A38;width:90px;height:20px;padding:40px;">
        Click Me</div>
    <script>
        function mDown(obj) {
            obj.style.backgroundColor = "#1ec5e5";
            obj.innerHTML = "Release Me";
        }
        function mUp(obj) {
            obj.style.backgroundColor="#D94A38";
            obj.innerHTML="Thank You";
        }
    </script>
</body>
</html>
```

Més exemples

onmousedown i onmouseup

Canvia una imatge quan un usuari manté premut el botó del ratolí.

onload

Mostra un quadre d'alerta quan la pàgina s'hagi acabat de carregar.

onfocus

Canvia el color de fons d'un camp d'entrada quan es centra.

Esdeveniments del ratolí

Canvia el color d'un element quan el cursor es mou per sobre.

Referència d'objectes d'esdeveniment HTML DOM

Per obtenir una llista de tots els esdeveniments HTML DOM, consulteu:

https://www.w3schools.com/jsref/dom_obj_event.asp

9. JavaScript HTML DOM EventListener

El mètode addEventListener().

Afegiu un oient d'esdeveniments que s'activa quan un usuari fa clic en un botó:

```
<!DOCTYPE html>
<html>
<body>
    <h2>JavaScript addEventListener()</h2>
    <p>This example uses the addEventListener() method to attach
    a click event to a button.</p>
    <button id="myBtn">Try it</button>
    <p id="demo"></p>
    <script>
        document.getElementById("myBtn").addEventListener
        ("click", displayDate);
        function displayDate() {
            document.getElementById("demo").innerHTML = Date();
        }
    </script>
</body>
</html>
```

El mètode addEventListener() enllaça un controlador d'esdeveniments a l'element especificat.

Podeu afegir molts controladors d'esdeveniments a un element.

Podeu afegir molts controladors d'esdeveniments del mateix tipus a un element, és a dir, dos esdeveniments de "clic".

Podeu afegir oients d'esdeveniments a qualsevol objecte DOM no només elements HTML. és a dir, l'objecte finestra.

Quan s'utilitza el addEventListener()mètode, el JavaScript està separat de l'etiquetatge HTML, per a una millor llegibilitat i us permet afegir oients d'esdeveniments fins i tot quan no controlieu l'etiquetatge HTML.

Podeu eliminar fàcilment un oient d'esdeveniments mitjançant el removeEventListener()mètode.

Sintaxi

element.addEventListener(event, function, useCapture);

El primer paràmetre és el tipus d'esdeveniment (com " click" o " mousedown" o qualsevol altre [esdeveniment HTML DOM](#)).

El segon paràmetre és la funció que volem cridar quan es produeix l'esdeveniment.

El tercer paràmetre és un valor booleà que especifica si s'ha d'utilitzar el bubbling d'esdeveniments o la captura d'esdeveniments. Aquest paràmetre és opcional.

Afegiu un gestor d'esdeveniments a un element

Exemple. Alerta "Hola món!" quan l'usuari fa clic en un element:

```
element.addEventListener("click", function(){ alert("Hello World!"); });
```

També podeu fer referència a una funció externa:

```
element.addEventListener("click", myFunction);
```

```
<!DOCTYPE html>
<html>
<body>
    <h2>JavaScript addEventListener()</h2>
    <p>This example uses the addEventListener() method to attach a click event to a button.</p>
    <button id="myBtn">Try it</button>
    <script>
        document.getElementById("myBtn").addEventListener("click",
            function() {
                alert("Hello World!");
            });
    </script>
</body>
</html>
```

Afegiu molts gestors d'esdeveniments al mateix element

El addEventListener()mètode us permet afegir molts esdeveniments al mateix element, sense sobreescrivir els esdeveniments existents:

Exemple

```
element.addEventListener("click", myFunction);
element.addEventListener("click", mySecondFunction);
```

Podeu afegir esdeveniments de diferents tipus al mateix element:

Exemple

```
element.addEventListener("mouseover", myFunction);
element.addEventListener("click", mySecondFunction);
element.addEventListener("mouseout", myThirdFunction);
```

```
<!DOCTYPE html>
<html>
<body>
    <h2>JavaScript addEventListener()</h2>
    <p>This example uses the addEventListener() method to add many events on the same button.</p>
    <button id="myBtn">Try it</button>
    <p id="demo"></p>
    <script>
        var x = document.getElementById("myBtn");
        x.addEventListener("mouseover", myFunction);
        x.addEventListener("click", mySecondFunction);
        x.addEventListener("mouseout", myThirdFunction);
        function myFunction() {
```

```

        document.getElementById("demo").innerHTML += "Moused over!<br>";
    }
    function mySecondFunction() {
        document.getElementById("demo").innerHTML += "Clicked!<br>";
    }
    function myThirdFunction() {
        document.getElementById("demo").innerHTML += "Moused out!<br>";
    }
</script>
</body>
</html>
```

Afegiu un gestor d'esdeveniments a l'objecte de la finestra

El mètode `addEventListener()` us permet afegir escoltes d'esdeveniments a qualsevol objecte DOM HTML, com ara elements HTML, el document HTML, l'objecte finestra o altres objectes que admeten esdeveniments, com XMLHttpRequestobjecte.

Exemple

Afegiu un oient d'esdeveniments que s'activa quan un usuari canvia la mida de la finestra:

```

<!DOCTYPE html>
<html>
<body>
<h2>JavaScript addEventListener()</h2>
<p>This example uses the addEventListener() method on the window
object.</p>
<p>Try resizing this browser window to trigger the "resize"
event handler.</p>
<p id="demo"></p>
<script>
    window.addEventListener("resize", function(){
        document.getElementById("demo").innerHTML = Math.random();
    });
</script>
</body>
</html>
```

Passant paràmetres

Per a passar els valors dels paràmetres, utilitzeu una "funció anònima" que cride a la funció especificada amb els paràmetres:

Exemple

```
element.addEventListener("click", function(){ myFunction(p1, p2); });
```

```

<!DOCTYPE html>
<html>
<body>
<h2>JavaScript addEventListener()</h2>
<p>This example demonstrates how to pass parameter values when using the
addEventListener() method.</p>
<p>Click the button to perform a calculation.</p>
<button id="myBtn">Try it</button>
<p id="demo"></p>
<script>
    let p1 = 5;
    let p2 = 7;
```

```

        document.getElementById("myBtn").addEventListener("click",
        function() {
        myFunction(p1, p2);
        });
        function myFunction(a, b) {
        document.getElementById("demo").innerHTML = a * b;
        }
    </script>
</body>
</html>

```

Bubbling d'esdeveniments o captura d'esdeveniments?

Hi ha dues maneres de propagació d'esdeveniments al DOM HTML, el bubbling i la captura.

La propagació d'esdeveniments és una manera de definir l'ordre dels elements quan es produeix un esdeveniment. Si teniu un element `<p>` dins d'un element `<div>` i l'usuari fa clic a l'element `<p>`, l'esdeveniment "clic" de quin element s'ha de gestionar primer?

En *bombolles*, primer es gestiona l'esdeveniment de l'element més intern i després l'exterior: primer es gestiona l'esdeveniment de clic de l'element `<p>`, després l'esdeveniment de clic de l'element `<div>`.

En *capturar* l'esdeveniment de l'element més exterior es gestiona primer i després l'interior: primer es gestionarà l'esdeveniment de clic de l'element `<div>` i després l'esdeveniment de clic de l'element `<p>`.

Amb el mètode `addEventListener()` podeu especificar el tipus de propagació mitjançant el paràmetre "useCapture":

`addEventListener(event, function, useCapture);`

El valor per defecte és fals, que utilitzarà la propagació de bombolles, quan el valor s'estableix en true, l'esdeveniment utilitza la propagació de captura.

Exemple

```

document.getElementById("myP").addEventListener("click", myFunction, true);
document.getElementById("myDiv").addEventListener("click", myFunction, true);

<!DOCTYPE html>
<html>
<head>
<style>
#myDiv1, #myDiv2 {
    background-color: coral;
    padding: 50px;
}

#myP1, #myP2 {
    background-color: white;
    font-size: 20px;
    border: 1px solid;
    padding: 20px;
}
</style>
<meta content="text/html; charset=utf-8" http-equiv="Content-Type">
</head>
<body>

```

```

<h2>JavaScript addEventListener()</h2>

<div id="myDiv1">
  <h2>Bubbling:</h2>
  <p id="myP1">Click me!</p>
</div><br>

<div id="myDiv2">
  <h2>Capturing:</h2>
  <p id="myP2">Click me!</p>
</div>

<script>
  document.getElementById("myP1").addEventListener("click", function() {
    alert("You clicked the white element!");
  }, false);

  document.getElementById("myDiv1").addEventListener("click", function() {
    alert("You clicked the orange element!");
  }, false);

  document.getElementById("myP2").addEventListener("click", function() {
    alert("You clicked the white element!");
  }, true);

  document.getElementById("myDiv2").addEventListener("click", function() {
    alert("You clicked the orange element!");
  }, true);
</script>

</body>
</html>

```

El mètode removeEventListener().

El removeEventListener() mètode elimina els controladors d'esdeveniments que s'han adjuntat amb el mètode addEventListener():

Exemple

```
element.removeEventListener("mousemove", myFunction);
```

```

<!DOCTYPE html>
<html>
<head>
<style>
  #myDIV {
    background-color: coral;
    border: 1px solid;
    padding: 50px;
    color: white;
    font-size: 20px;
  }
</style>
</head>
<body>
  <h2>JavaScript removeEventListener()</h2>
  <div id="myDIV">

```

```

<p>This div element has an onmousemove event handler that displays a random number every time you move your mouse inside this orange field.</p>
<p>Click the button to remove the div's event handler.</p>
<button onclick="removeHandler()" id="myBtn">Remove</button>
</div>
<p id="demo"></p>
<script>
    document.getElementById("myDIV").addEventListener("mousemove",
    myFunction);
    function myFunction() {
        document.getElementById("demo").innerHTML = Math.random();
    }
    function removeHandler() {
        document.getElementById("myDIV").removeEventListener("mousemove",
        myFunction);
    }
</script>
</body>
</html>

```

Referència d'objectes d'esdeveniment HTML DOM

Per obtenir una llista de tots els esdeveniments HTML DOM, consulteu:

https://www.w3schools.com/jsref/dom_obj_event.asp

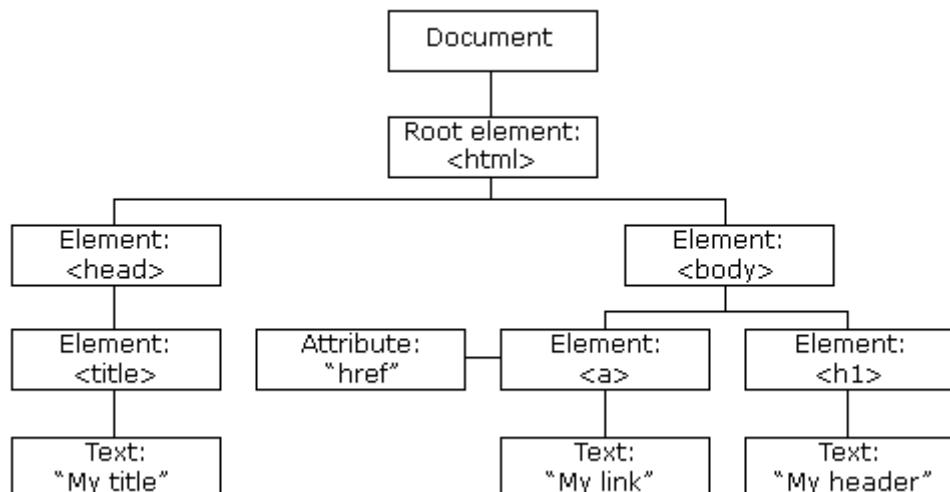
10. JavaScript HTML Navegació DOM

Amb el DOM HTML, podeu navegar per l'arbre de nodes mitjançant relacions de nodes.

Nodes DOM

Segons l'estàndard HTML DOM del W3C, tot en un document HTML és un node:

- Tot el document és un node de document
- Cada element HTML és un node d'element
- El text dins dels elements HTML són nodes de text
- Cada atribut HTML és un node d'atribut (obsolet)
- Tots els comentaris són nodes de comentaris



Amb el DOM HTML, es pot accedir a tots els nodes de l'arbre de nodes mitjançant JavaScript.

Es poden crear nous nodes i tots els nodes es poden modificar o suprimir.

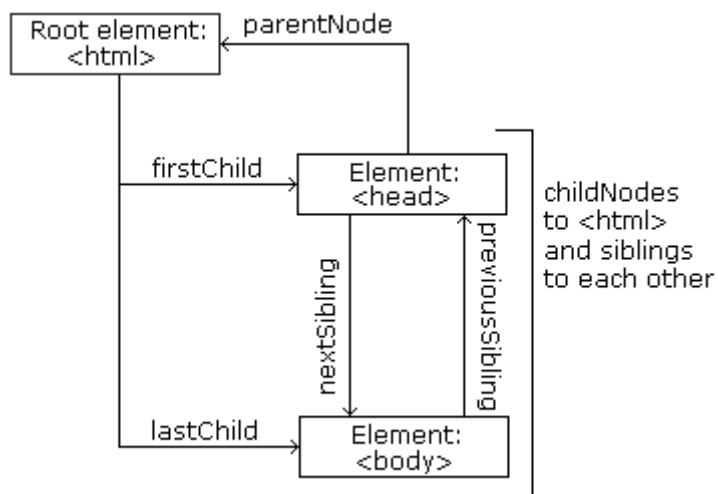
Relacions de nodes

Els nodes de l'arbre de nodes tenen una relació jeràrquica entre ells.

Els termes pare, fill i germà s'utilitzen per descriure les relacions.

- En un arbre de nodes, el node superior s'anomena arrel (o node arrel)
- Cada node té exactament un pare, excepte l'arrel (que no té cap pare)
- Un node pot tenir un nombre de fills de 0 a n
- Els germans (germans o germanes) són nodes amb el mateix progenitor

```
<html>
  <head>
    <title>DOM Tutorial</title>
  </head>
  <body>
    <h1>DOM Lesson one</h1>
    <p>Hello world!</p>
  </body>
</html>
```



Des de l'HTML anterior podeu llegir:

- <html> és el node arrel
- <html> no té pares
- <html> és el pare de <head> i <body>
- <head> és el primer fill de <html>
- <body> és l'últim fill de <html>
- <head> té un fill: <title>
- <title> té un fill (un node de text): "Tutorial DOM"
- <body> té dos fills: <h1> i <p>
- <h1> té un fill: "DOM Lliçó 1"
- <p> té un fill: "Hola món!"
- <h1> i <p> són germans

Navegació entre nodes

Podeu utilitzar les propietats de nodes següents per navegar entre nodes amb JavaScript:

- parentNode
- childNodes[nodenumber]
- firstChild
- lastChild
- nextSibling
- previousSibling

Nodes fills i valors de nodes

Un error comú en el processament DOM és esperar que un node d'element continga text.

Exemple:

```
<title id="demo">DOM Tutorial</title>
```

El node d'element `<title>` (a l'exemple anterior) no **conté** text. Conté un **node de text** amb el valor "DOM Tutorial".

Es pot accedir al valor del node de text mitjançant la `innerHTML` propietat del node:

```
myTitle = document.getElementById("demo").innerHTML;
```

Accedir a la propietat `innerHTML` és el mateix que accedir a la `nodeValue` del primer fill:

```
myTitle = document.getElementById("demo").firstChild.nodeValue;
```

L'accés al primer fill també es pot fer així:

```
myTitle = document.getElementById("demo").childNodes[0].nodeValue;
```

Tots els (3) exemples següents recuperen el text d'un `<h1>` element i el copien en un element `<p>`:

Exemple

```
<html>
<body>
    <h1 id="id01">My First Page</h1>
    <p id="id02"></p>
    <script>
        document.getElementById("id02").innerHTML =
            document.getElementById("id01").innerHTML;
    </script>
</body>
</html>
```

Exemple

```
<html>
<body>
    <h1 id="id01">My First Page</h1>
    <p id="id02"></p>
    <script>
        document.getElementById("id02").innerHTML =
            document.getElementById("id01").firstChild.nodeValue;
    </script>
```

```
</body>
</html>
```

Exemple

```
<html>
<body>
<h1 id="id01">My First Page</h1>
<p id="id02">Hello!</p>
<script>
document.getElementById("id02").innerHTML =
document.getElementById("id01").childNodes[0].nodeValue;
</script>
</body>
</html>
```

InnerHTML

En aquest tutorial fem servir la propietat innerHTML per recuperar el contingut d'un element HTML.

Tanmateix, aprendre els altres mètodes anteriors és útil per entendre l'estructura de l'arbre i la navegació del DOM.

Nodes arrel DOM

Hi ha dues propietats especials que permeten l'accés al document complet:

- `document.body` - El cos del document
- `document.documentElement` - El document complet

Exemple

```
<html>
<body>
<h2>JavaScript HTMLDOM</h2>
<p>Displaying document.body</p>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML = document.body.innerHTML;
</script>
</body>
</html>
```

Exemple

```
<html>
<body>
<h2>JavaScript HTMLDOM</h2>
<p>Displaying document.documentElement</p>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML =
document.documentElement.innerHTML;
</script>
</body>
</html>
```

La propietat nodeName

La nodeNameproprietat especifica el nom d'un node.

- nodeName és només de lectura
- nodeName d'un node d'element és el mateix que el nom de l'etiqueta
- nodeName d'un atribut node és el nom de l'atribut
- nodeName d'un node de text sempre és #text
- nodeName del node del document sempre és #document

Exemple

```
<!DOCTYPE html>
<html>
<body>

<h1 id="id01">My First Page</h1>
<p id="id02"></p>

<script>
document.getElementById("id02").innerHTML =
document.getElementById("id01").nodeName;
</script>

</body>
</html>
```

Nota: nodeName sempre conté el nom d'etiqueta en majúscula d'un element HTML.

La propietat nodeValue

La nodeValueproprietat especifica el valor d'un node.

- nodeValue per als nodes d'element és null
- nodeValue per als nodes de text és el mateix text
- nodeValue per als nodes d'atribut és el valor de l'atribut

La propietat.nodeType

La nodeType propietat és només de lectura. Retorna el tipus de node.

Exemple

```
<!DOCTYPE html>
<html>
<body>

<h1 id="id01">My First Page</h1>
<p id="id02"></p>

<script>
document.getElementById("id02").innerHTML =
document.getElementById("id01").nodeType;
</script>

</body>
</html>
```

Les propietats de nodeType més importants són:

| Node | Type | Example |
|--------------------|------|---|
| ELEMENT_NODE | 1 | <h1 class="heading">W3Schools</h1> |
| ATTRIBUTE_NODE | 2 | class = "heading" (deprecated) |
| TEXT_NODE | 3 | W3Schools |
| COMMENT_NODE | 8 | <!-- This is a comment --> |
| DOCUMENT_NODE | 9 | The HTML document itself (the parent of <html>) |
| DOCUMENT_TYPE_NODE | 10 | <!Doctype html> |

El tipus 2 està obsolet al DOM HTML (però funciona). No està obsolet al DOM XML.

11. Elements DOM HTML JavaScript (nodes)

Afegir i eliminar nodes (elements HTML)

Creació d'elements HTML nous (nodes)

Per afegir un element nou al DOM HTML, primer heu de crear l'element (node d'element) i després afegir-lo a un element existent.

Exemple

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript HTML DOM</h2>
<p>Add a new HTML Element.</p>

<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
const para = document.createElement("p");
const node = document.createTextNode("This is new.");
para.appendChild(node);
const element = document.getElementById("div1");
element.appendChild(para);
</script>

</body>
</html>
```

Exemple explicat

Aquest codi crea un element <p> nou:

```
const para = document.createElement("p");
```

Per afegir text a l'element <p> , primer heu de crear un node de text. Aquest codi crea un node de text:

```
const node = document.createTextNode("This is a new paragraph.");
```

A continuació, heu d'afegir el node de text a l' <p> element:

```
para.appendChild(node);
```

Finalment, heu d'afegir l'element nou a un element existent.

Aquest codi troba un element existent:

```
const element = document.getElementById("div1");
```

Aquest codi afegeix l'element nou a l'element existent:

```
element.appendChild(para);
```

Creació d'elements HTML nous - insertBefore()

El appendChild()mètode de l'exemple anterior va afegir el nou element com a darrer fill del pare.

Si no ho voleu, podeu utilitzar el insertBefore()mètode:

Exemple

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript HTML DOM</h2>
<p>Add a new HTML Element.</p>

<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
const para = document.createElement("p");
const node = document.createTextNode("This is new.");
para.appendChild(node);

const element = document.getElementById("div1");
const child = document.getElementById("p1");
element.insertBefore(para,child);
</script>

</body>
</html>
```

Eliminació d'elements HTML existents

Per eliminar un element HTML, utilitzeu el remove() mètode:

Exemple

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript HTML DOM</h2>
<h3>Remove an HTML Element.</h3>

<div>
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>
```

```

<button onclick="myFunction()">Remove Element</button>

<script>
function myFunction() {
document.getElementById("p1").remove();
}
</script>

</body>
</html>

```

Exemple explicat

El document HTML conté un element <div> amb dos nodes secundaris (dos <p> elements):

```

<div>
  <p id="p1">This is a paragraph.</p>
  <p id="p2">This is another paragraph.</p>
</div>

```

Cerqueu l'element que voleu eliminar:

```
const elmnt = document.getElementById("p1");
```

A continuació, executeu el mètode remove() en aquest element:

```
elmnt.remove();
```

Eliminació d'un node fill

Per als navegadors que no admeten el remove() mètode, heu de trobar el node pare per eliminar un element:

Exemple

```

<!DOCTYPE html>
<html>
<body>

<h2>JavaScript HTML DOM</h2>
<p>Remove Child Element</p>

<div id="div1">
  <p id="p1">This is a paragraph.</p>
  <p id="p2">This is another paragraph.</p>
</div>

<script>
const parent = document.getElementById("div1");
const child = document.getElementById("p1");
parent.removeChild(child);
</script>

</body>
</html>

```

Exemple explicat

Aquest document HTML conté un element <div> amb dos nodes secundaris (dos <p> elements):

```
<div id="div1">
    <p id="p1">This is a paragraph.</p>
    <p id="p2">This is another paragraph.</p>
</div>
```

Troba l'element amb id="div1":

```
const parent = document.getElementById("div1");
```

Troba l' <p>element amb id="p1":

```
const child = document.getElementById("p1");
```

Elimineu el fill dels pares:

```
parent.removeChild(child);
```

Substitució d'elements HTML

Per substituir un element al DOM HTML, utilitzeu el replaceChild() mètode:

Exemple

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript HTML DOM</h2>
<h3>Replace an HTML Element.</h3>

<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is a paragraph.</p>
</div>

<script>
const parent = document.getElementById("div1");
const child = document.getElementById("p1");
const para = document.createElement("p");
const node = document.createTextNode("This is new.");
para.appendChild(node);
parent.replaceChild(para,child);
</script>

</body>
</html>
```

Tema 5: Definició d'esquemes i vocabularis en llenguatges de marques



1. [Introducció al DTD](#)
 2. [Blocs de construcció DTD](#)
 3. [Elements DTD](#)
 4. [Atributs DTD](#)
 5. [Elements DTD vs Atributs](#)
 6. [Entitats DTD](#)
 7. [Exemples de DTD](#)
 8. [Introducció a XSD.](#)
 9. [XSD <esquema>](#)
 10. [Elements XSD](#)
 11. [Atributs XSD](#)
 12. [Restriccions XSD](#)
 13. [Elements complexos XSD](#)
 14. [Indicadors XSD](#)
-

1. Introducció al DTD

Un DTD és una definició de tipus de document. Una DTD defineix l'estructura i els elements i atributs legals d'un document XML.

Per què utilitzar un DTD?

Amb un DTD, grups independents de persones poden acordar un DTD estàndard per intercanviar dades.

Una aplicació pot utilitzar un DTD per a verificar que les dades XML són vàlides.

Una declaració interna de DTD

Si el DTD es declara dins del fitxer XML, s'ha d'embolicar dins de la definició <!DOCTYPE>:

Document XML amb una DTD interna

```
<?xml version="1.0"?>
<!DOCTYPE note [
    <!ELEMENT note (to,from,heading,body)>
    <!ELEMENT to (#PCDATA)>
    <!ELEMENT from (#PCDATA)>
    <!ELEMENT heading (#PCDATA)>
    <!ELEMENT body (#PCDATA)>
]>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend</body>
</note>
```

Si visualitzem el XML no veurem el DTD, cadrà que seleccioneu "visualitza el codi font" per veure el DTD.

El DTD anterior s'interpreta així:

- **!DOCTYPE note** defineix que l'element arrel d'aquest document és la nota
- **!ELEMENT nota** defineix que l'element nota ha de contenir quatre elements: "to, from, heading, body"
- **!ELEMENT to** defineix que l'element to serà de tipus "#PCDATA"
- **!ELEMENT from** defineix l'element de tipus "#PCDATA"
- **!ELEMENT heading** defineix l'element d'encapçalament que siga del tipus "#PCDATA"
- **!ELEMENT body** defineix l'element del cos com a tipus "#PCDATA"

Una declaració de DTD externa

Si el DTD es declara en un fitxer extern, la definició <!DOCTYPE> ha de contenir una referència al fitxer DTD:

Document XML amb una referència a una DTD externa

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note SYSTEM "Note.dtd">
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

I ací teniu el fitxer "note.dtd", que conté el DTD:

```
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

2. Blocs de construcció DTD

Els components bàsics dels documents XML i HTML són els elements.

Els blocs de construcció dels documents XML

Vist des del punt de vista de DTD, tots els documents XML estan formats pels següents blocs de construcció:

- Elements
- Atributs
- Entitats
- PCDATA
- CDATA

Elements

Els elements són els **principals blocs de construcció** dels documents XML i HTML.

Exemples d'elements HTML són "body" i "table". Exemples d'elements XML podrien ser "nota" i "missatge". Els elements poden contenir text, altres elements o estar buits. Exemples d'elements HTML buits són "hr", "br" i "img". Exemples:

```
<body>some text</body>
<message>some text</message>
```

Atributs

Els atributs proporcionen **informació addicional sobre els elements**.

Els atributs sempre es col·loquen dins de l'etiqueta d'obertura d'un element. Els atributs sempre vénen en parells nom/valor. L'element "img" següent té informació addicional sobre un fitxer font:

```

```

El nom de l'element és "img". El nom de l'atribut és "src". El valor de l'atribut és "computer.gif". Com que l'element en si està buit, es tanca amb un "/".

Entitats

Alguns caràcters tenen un significat especial en XML, com ara el signe inferior (<) que defineix l'inici d'una etiqueta XML.

Les entitats següents estan predefinides en XML:

Entity References Character

| | |
|--------|---|
| < | < |
| > | > |
| & | & |
| " | " |
| ' | ' |

PCDATA

PCDATA és *parsed character data* o, més clar, text pla. Però no pot contenir cap altre element.

Penseu en les dades PCDATA com a el text que es troba entre l'etiqueta inicial i l'etiqueta final d'un element XML.

CDATA

CDATA significa dades de caràcters.

CDATA és text que NO serà analitzat per un analitzador. Les etiquetes dins del text NO es tractaran com a marques i les entitats de caràcter no s'ampliaran.

3. DTD - Elements

En una DTD, els elements es declaren amb una declaració ELEMENT i amb la sintaxi següent:

```
<!ELEMENT element-name category>
or
<!ELEMENT element-name (element-content)>
```

Elements buits

Els elements buits es declaren amb la paraula clau EMPTY:

```
<!ELEMENT element-name EMPTY>
```

Exemple: `<!ELEMENT br EMPTY>`

Elements amb PCDATA

Els elements amb només dades de caràcters analitzats es declaren amb #PCDATA entre parèntesis:

```
<!ELEMENT element-name (#PCDATA)>
```

Example: <!ELEMENT from (#PCDATA)>

Elements amb qualsevol Contingut

Els elements declarats amb la paraula clau de categoria ANY poden contenir qualsevol combinació de dades analitzables:

```
<!ELEMENT element-name ANY>
```

Example: <!ELEMENT note ANY>

Elements amb fills (seqüències)

Els elements amb un o més fills es declaren amb el nom dels elements fills entre parèntesis:

```
<!ELEMENT element-name (child1)>  
or  
<!ELEMENT element-name (child1, child2, ...)>
```

Example: <!ELEMENT note (to, from, heading, body)>

Quan els fills es declaren en una seqüència separats per comes, els fills han d'aparèixer en la mateixa seqüència al document. En una declaració completa, també s'han de declarar els fills, i els fills també poden tenir fills. La declaració completa de l'element "nota" és:

```
<!ELEMENT note (to, from, heading, body)>  
<!ELEMENT to (#PCDATA)>  
<!ELEMENT from (#PCDATA)>  
<!ELEMENT heading (#PCDATA)>  
<!ELEMENT body (#PCDATA)>
```

Declarar només una ocurrència d'un element

```
<!ELEMENT element-name (child-name)>
```

Example: <!ELEMENT note (message)>

L'exemple anterior declara que l'element fill "message" s'ha de produir una vegada, i només una vegada dins de l'element "nota".

Declaració d'un mínim d'una ocurrència d'un element

```
<!ELEMENT element-name (child-name+)>
```

Example: <!ELEMENT note (message+)>

El signe + de l'exemple anterior declara que l'element secundari "message" s'ha de produir una o més vegades dins de l'element "nota".

Declaració de zero o més ocurrències d'un element

```
<!ELEMENT element-name (child-name*)>
```

Example: <!ELEMENT note (message*)>

El signe * de l'exemple anterior declara que l'element secundari "message" pot aparèixer zero o més vegades dins de l'element "nota".

Declaració de zero o una ocurrència d'un element

```
<!ELEMENT element-name (child-name?)>
```

Example: `<!ELEMENT note (message?)>`

El ? a l'exemple anterior declara que l'element fill "message" pot aparèixer zero o una vegada dins de l'element "nota".

Declaració d'un o altre contingut

```
<!ELEMENT note (to,from,header,(message| body))>
```

L'exemple anterior declara que l'element "nota" ha de contenir un element "to", un element "from", un element "header" i un element "message" o "body".

Declaració de contingut mixt

```
<!ELEMENT note (#PCDATA|to|from|header|message)*>
```

L'exemple anterior declara que l'element "nota" pot contenir zero o més ocurrències de dades de caràcters analitzats, elements "to", "from", "header" o "message".

4. DTD - Atributs

En una DTD, els atributs es declaren amb una declaració ATTLIST.

Declaració d'atributs

Una declaració d'atribut té la sintaxi següent:

```
<!ATTLIST element-name attribute-name attribute-type attribute-value>
```

DTD example: `<!ATTLIST payment type CDATA "check">`

XML example: `<payment type="check"/>`

El **tipus d'atribut** pot ser un dels següents:

| Type | Description |
|--------------|---|
| CDATA | The value is character data |
| (en1 en2 ..) | The value must be one from an enumerated list |
| ID | The value is a unique id |
| IDREF | The value is the id of another element |
| IDREFS | The value is a list of other ids |
| NMTOKEN | The value is a valid XML name |
| NMTOKENS | The value is a list of valid XML names |
| ENTITY | The value is an entity |
| ENTITIES | The value is a list of entities |
| NOTATION | The value is a name of a notation |
| xml: | The value is a predefined xml value |

El **valor-atribut** pot ser un dels següents:

| Value | Explanation |
|-------------|------------------------------------|
| value | The default value of the attribute |
| #REQUIRED | The attribute is required |
| #IMPLIED | The attribute is optional |
| #FIXEDvalue | The attribute value is fixed |

Un valor d'atribut per defecte

DTD:

```
<!ELEMENT square EMPTY>
<!ATTLIST square width CDATA "0">
```

Valid XML:

```
<square width="100" />
```

A l'exemple anterior, l'element "quadrat" es defineix com un element buit amb un atribut "amplada" de tipus CDATA. Si no s'especifica cap amplada, té un valor predeterminat de 0.

#REQUIRED

Sintaxi

```
<!ATTLIST element-name attribute-name attribute-type #REQUIRED>
```

DTD:

```
<!ATTLIST person number CDATA #REQUIRED>
```

Valid XML:

```
<person number="5677"/>
```

Invalid XML:

```
<person/>
```

Utilitzeu la paraula clau #REQUIRED si no teniu cap opció per a un valor predeterminat, però encara voleu forçar que l'atribut estiga present.

#IMPLIED

Sintaxi

```
<!ATTLIST element-name attribute-name attribute-type #IMPLIED>
```

DTD:

```
<!ATTLIST contact fax CDATA #IMPLIED>
```

Valid XML:

```
<contact fax="555-667788"/>
```

Valid XML:

```
<contact/>
```

Utilitzeu la paraula clau #IMPLIED si no voleu obligar l'autor a incloure un atribut i no teniu cap opció per a un valor predeterminat.

#FIXED

Sintaxi

```
<!ATTLIST element-name attribute-name attribute-type #FIXED "value">
```

DTD:

```
<!ATTLIST sender company CDATA #FIXED "Microsoft">
```

Valid XML:

```
<sender company="Microsoft"/>
```

Invalid XML:

```
<sender company="W3Schools"/>
```

Utilitzeu la paraula clau #FIXED quan vulgueu que un atribut tinga un valor fix sense permetre que l'autor el canvie. Si un autor inclou un altre valor, l'analitzador XML retornarà un error.

Valors d'atributs enumerats

Sintaxi

```
<!ATTLIST element-name attribute-name (en1|en2|..) default-value>
```

DTD:

```
<!ATTLIST payment type (check|cash) "cash">
```

XML example:

```
<payment type="check"/>  
or  
<payment type="cash"/>
```

Utilitzeu valors d'atribut enumerats quan vulgueu que el valor de l'atribut siga un d'un conjunt fix de valors legals.

5. Elements XML vs. atributs

En XML, no hi ha regles sobre quan s'han d'utilitzar els atributs i quan s'han d'utilitzar elements secundaris. Les dades es poden emmagatzemar en elements fills o en atributs.

Mireu aquests exemples:

```
<person sex="female">  
  <firstname>Anna</firstname>  
  <lastname>Smith</lastname>  
</person>  
  
<person>  
  <sex>female</sex>  
  <firstname>Anna</firstname>  
  <lastname>Smith</lastname>  
</person>
```

En el primer exemple, el sexe és un atribut. En l'últim, el sexe és un element fill. Tots dos exemples proporcionen la mateixa informació.

No hi ha regles sobre quan utilitzar atributs i quan utilitzar elements fills. La meua experiència és que els atributs són útils en HTML, però en XML hauríeu d'intentar evitar-los. Utilitzeu elements secundaris si la informació sembla dades.

La meva manera preferida: **M'agrada emmagatzemar dades en elements secundaris.**

Els tres documents XML següents contenen exactament la mateixa informació:

En el primer exemple s'utilitza un atribut de data:

```
<note date="12/11/2002">
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

En el segon exemple s'utilitza un element de data:

```
<note>
  <date>12/11/2002</date>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

S'utilitza un element de data ampliat al tercer: (AQUEST ÉS EL MEU PREFERIT):

```
<note>
  <date>
    <day>12</day>
    <month>11</month>
    <year>2002</year>
  </date>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Evitar utilitzar atributs?

Heu d'evitar l'ús d'atributs?

Alguns dels problemes amb els atributs són:

- els atributs no poden contenir diversos valors (els elements secundaris poden)
- els atributs no es poden ampliar fàcilment (per a canvis futurs)
- els atributs no poden descriure estructures (els elements secundaris poden)
- els atributs són més difícils de manipular pel codi del programa
- els valors dels atributs no són fàcils de provar amb un DTD

Si utilitzeu atributs com a contenidors de dades, acabeu amb documents difícils de llegir i mantenir. Intenta utilitzar **elements** per descriure dades. Utilitzeu els atributs només per proporcionar informació que no sigua rellevant per a les dades.

No acabes així:

```
<note day="12" month="11" year="2002"  
to="Tove" from="Jani" heading="Reminder"  
body="Don't forget me this weekend!">  
</note>
```

Una excepció a la meua regla d'atributs

Les regles sempre tenen excepcions. De vegades assigne referències d'identificació als elements. Aquestes referències d'ID es poden utilitzar per accedir als elements XML de la mateixa manera que els atributs NAME o ID en HTML. Aquest exemple ho demostra:

```
<messages>  
<note id="p501">  
  <to>Tove</to>  
  <from>Jani</from>  
  <heading>Reminder</heading>  
  <body>Don't forget me this weekend!</body>  
</note>  
  
<note id="p502">  
  <to>Jani</to>  
  <from>Tove</from>  
  <heading>Re: Reminder</heading>  
  <body>I will not!</body>  
</note>  
</messages>
```

L'ID d'aquests exemples és només un comptador, o un identificador únic, per identificar les diferents notes del fitxer XML, i no una part de les dades de la nota.

El que estic intentant dir és que les metadades (dades sobre les dades) s'han d'emmagatzemar com a atributs, i les mateixes dades s'han d'emmagatzemar com a elements.

6. DTD - Entitats

Les entitats s'utilitzen per definir dreceres a caràcters especials. Les entitats es poden declarar internes o externes.

Declaració interna de l'entitat

Sintaxi

```
<!ENTITY entity-name "entity-value">
```

Exemple

DTD Example:

```
<!ENTITY writer "Donald Duck.">  
<!ENTITY copyright "Copyright W3Schools.">
```

XML example:

```
<author>&writer;&copyright;</author>
```

Nota: una entitat té 3 parts: un ampersand (&), un nom de l'entitat i un punt i coma (;).

Una declaració d'entitat externa

Sintaxi

```
<!ENTITY entity-name SYSTEM "URI/URL">
```

DTD Example:

```
<!ENTITY writer SYSTEM "https://www.w3schools.com/entities.dtd">
<!ENTITY copyright SYSTEM "https://www.w3schools.com/entities.dtd">
```

XML example:

```
<author>&writer;&copyright;</author>
```

7. Exemples de DTD

TV Schedule DTD

```
<!DOCTYPE TVSCHEDULE [
    <!ELEMENT TVSCHEDULE (CHANNEL+)>
    <!ELEMENT CHANNEL (BANNER, DAY+)>
    <!ELEMENT BANNER (#PCDATA)>
    <!ELEMENT DAY (DATE, (HOLIDAY|PROGRAMSLIST)+)>
    <!ELEMENT HOLIDAY (#PCDATA)>
    <!ELEMENT DATE (#PCDATA)>
    <!ELEMENT PROGRAMSLIST (TIME, TITLE, DESCRIPTION?)>
    <!ELEMENT TIME (#PCDATA)>
    <!ELEMENT TITLE (#PCDATA)>
    <!ELEMENT DESCRIPTION (#PCDATA)>

    <!ATTLIST TVSCHEDULE NAME CDATA #REQUIRED>
    <!ATTLIST CHANNEL CHAN CDATA #REQUIRED>
    <!ATTLIST PROGRAMSLIST VTR CDATA #IMPLIED>
    <!ATTLIST TITLE RATING CDATA #IMPLIED>
    <!ATTLIST TITLE LANGUAGE CDATA #IMPLIED>
]>
```

Newspaper Article DTD

```
<!DOCTYPE NEWSPAPER [
    <!ELEMENT NEWSPAPER (ARTICLE+)>
    <!ELEMENT ARTICLE (HEADLINE, BYLINE, LEAD, BODY, NOTES)>
    <!ELEMENT HEADLINE (#PCDATA)>
    <!ELEMENT BYLINE (#PCDATA)>
    <!ELEMENT LEAD (#PCDATA)>
    <!ELEMENT BODY (#PCDATA)>
    <!ELEMENT NOTES (#PCDATA)>

    <!ATTLIST ARTICLE AUTHOR CDATA #REQUIRED>
    <!ATTLIST ARTICLE EDITOR CDATA #IMPLIED>
    <!ATTLIST ARTICLE DATE CDATA #IMPLIED>
    <!ATTLIST ARTICLE EDITION CDATA #IMPLIED>

    <!ENTITY NEWSPAPER "Vervet Logic Times">
    <!ENTITY PUBLISHER "Vervet Logic Press">
    <!ENTITY COPYRIGHT "Copyright 1998 Vervet Logic Press">
]>
```

8. Introducció a XSD.

Què és un esquema XML?

Un esquema XML és un document que defineix l'estructura i les regles d'un fitxer XML, assegurant que les dades segueixen un format específic. Els esquemes XML ajuden a validar la informació, establint quins elements poden aparèixer, en quin ordre i amb quins tipus de dades.

Ja hem vist el DTD (Document Type Definition) que és un tipus d'esquema més antic i poc flexible.

No permet definir tipus de dades amb precisió. Per contra:

XSD (XML Schema Definition)

És el més modern i utilitzat. Permet definir tipus de dades complexos, restriccions i regles més avançades.

Exemples d'esquemes XSD:

```
<persona>
    <nom>Anna</nom>
    <edat>30</edat>
</persona>
```

Tenim un XML, però no sabem si està ben format perquè no hi ha cap esquema que defineisca quins elements i tipus de dades s'accepten.

Exemple d'un esquema XSD que valida aquest XML:

```
<xss:schema xmlns:xss="http://www.w3.org/2001/XMLSchema">
    <xss:element name="persona">
        <xss:complexType>
            <xss:sequence>
                <xss:element name="nom" type="xs:string"/>
                <xss:element name="edat" type="xs:int"/>
            </xss:sequence>
        </xss:complexType>
    </xss:element>
</xss:schema>
```

Aquest esquema diu que:

L'element principal és `<persona>`.

Ha de contenir `<nom>` (de tipus text) i `<edat>` (de tipus número).

Així, si l'XML té errors (per exemple, si "edat" conté text en comptes d'un número), el validarem contra l'esquema i detectarem el problema.

9. XSD: l' element <schema>

En XSD, l'element <schema> és l'element arrel d'un esquema XML. Defineix les regles i les restriccions per a l'estructura dels documents XML que segueixen aquest esquema.

Sintaxi bàsica:

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <!-- Definició dels elements, tipus, atributs, etc. -->
</xs:schema>
```

Atributs habituals de <schema>:

xmlns:xs="http://www.w3.org/2001/XMLSchema" : Declara l'espai de noms estàndard per a XSD.

targetNamespace : Defineix l'espai de noms per als elements definits en aquest esquema.

elementFormDefault : Indica si els elements han d'estar qualificats amb un espai de noms (qualified o unqualified).

attributeFormDefault : Determina si els atributs han d'estar qualificats o no.

Exemple complet d'un XSD:

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.exemple.com"
    elementFormDefault="qualified">

    <xs:element name="persona">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="nom" type="xs:string"/>
                <xs:element name="edat" type="xs:int"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

Explicació:

Defineix un esquema XML amb un espai de noms http://www.exemple.com.

Conté un element <persona> que té dos subelements: <nom> (cadena de text) i <edat> (nombre enter).

10. Elements XSD

Què és un element simple?

Un element simple és un element XML que només pot contenir text. No pot contenir cap altre element o atribut.

Definició d'un element simple

La sintaxi per definir un element simple és:

```
<xs:element name="xxx" type="yyy"/>
```

XSD té molts tipus de dades integrats. Els tipus més comuns són:

- xs:cadena
- xs: decimal
- xs: nombre sencer
- xs:booleà
- xs:data
- xs: temps

Exemple. Aquests són alguns exemples d'elements:

```
<lastname>Refsnes</lastname>
<age>36</age>
<dateborn>1970-03-27</dateborn>
```

I ací tenim les definicions d'elements simples:

```
<xs:element name="lastname" type="xs:string"/>
<xs:element name="age" type="xs:integer"/>
<xs:element name="dateborn" type="xs:date"/>
```

Valors predeterminats i fixos per a elements simples

Els elements simples poden tenir un valor predeterminat o especificar un valor fix. S'assigna automàticament un valor per defecte a l'element quan no s'especifica cap altre valor.

A l'exemple següent, el valor per defecte és "red":

```
<xs:element name="color" type="xs:string" default="red"/>
```

També s'assigna automàticament un valor fix a l'element i no podeu especificar un altre valor.

A l'exemple següent, el valor fix és "red":

```
<xs:element name="color" type="xs:string" fixed="red"/>
```

11. Atributs XSD

Tots els atributs es declaren com a tipus simples.

Què és un atribut?

Els elements simples no poden tenir atributs. Si un element té atributs, es considera que és de tipus complex. Però l'atribut en si es declara sempre com un tipus simple.

Com definir un atribut?

La sintaxi per definir un atribut és:

```
<xs:attribute name="xxx" type="yyy"/>
```

Exemple. Ací hi ha un element XML amb un atribut:

```
<lastname lang="EN">Smith</lastname>
```

I ací teniu la definició d'atribut corresponent:

```
<xs:attribute name="lang" type="xs:string"/>
```

Valors predeterminats i fixos per als atributs

A l'exemple següent, el valor per defecte és "EN":

```
<xs:attribute name="lang" type="xs:string" default="EN"/>
```

A l'exemple següent, el valor fix és "EN":

```
<xs:attribute name="lang" type="xs:string" fixed="EN"/>
```

Atributs opcionals i obligatoris

Els atributs són opcionals per defecte. Per especificar que l'atribut és obligatori, utilitzeu l'atribut "use":

```
<xs:attribute name="lang" type="xs:string" use="required"/>
```

Restriccions de contingut

Quan un element o atribut XML té un tipus de dades definit, imposa restriccions al contingut de l'element o atribut.

Si un element XML és del tipus "xs:date" i conté una cadena com "Hello World", l'element no es validarà.

Amb els esquemes XML, també podeu afegir les vostres pròpies restriccions als vostres elements i atributs XML.

12. Restriccions XSD

Les restriccions s'utilitzen per definir valors acceptables per als elements o atributs XML. Les restriccions als elements XML s'anomenen facets.

Restriccions de valors

L'exemple següent defineix un element anomenat "edat" amb una restricció. El valor de l'edat no pot ser inferior a 0 ni superior a 120:

```
<xs:element name="age">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="120"/>
    </xs:restriction>
```

```
</xs:simpleType>
</xs:element>
```

Restriccions a un conjunt de valors

Per limitar el contingut d'un element XML a un conjunt de valors acceptables, utilitzaríem la restricció d'enumeració.

L'exemple següent defineix un element anomenat "cotxe" amb una restricció. Els únics valors acceptables són: Audi, Wolsvagen, BMW:

```
<xs:element name="car">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Audi"/>
      <xs:enumeration value="Wolsvagen"/>
      <xs:enumeration value="BMW"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

L'exemple anterior també es podria haver escrit així:

```
<xs:element name="car" type="carType"/>

<xs:simpleType name="carType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Audi"/>
    <xs:enumeration value="Wolsvagen"/>
    <xs:enumeration value="BMW"/>
  </xs:restriction>
</xs:simpleType>
```

Nota: En aquest cas, el tipus "carType" pot ser utilitzat per altres elements perquè no forma part de l'element "car".

Restriccions a una sèrie de valors

Per limitar el contingut d'un element XML per definir una sèrie de números o lletres que es poden utilitzar, utilitzaríem la restricció de patró.

L'exemple següent defineix un element anomenat "letter" amb una restricció. L'únic valor acceptable és una de les lletres minúscules de la a a la z:

```
<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

L'exemple següent defineix un element anomenat "initials" amb una restricció. L'únic valor acceptable són 3 de les lletres MAJÚSCULAS de la a a la z:

```
<xs:element name="initials">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z][A-Z][A-Z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

```

</xs:simpleType>
</xs:element>
```

L'exemple següent també defineix un element anomenat "initials" amb una restricció. L'únic valor acceptable són 3 de les lletres MINÚSCULES O MAJÚSCULAS de la a a la z:

```

<xs:element name="initials">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

L'exemple següent defineix un element anomenat "choice" amb una restricció. L'únic valor acceptable és una de les lletres següents: x, y, O z:

```

<xs:element name="choice">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[xyz]" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Altres restriccions a una sèrie de valors

L'exemple següent defineix un element anomenat "letter" amb una restricció. El valor acceptable és zero o més aparicions de lletres minúscules de la a a la z:

```

<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="([a-z])*" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

L'exemple següent també defineix un element anomenat "letter" amb una restricció. El valor acceptable és un o més parells de lletres, cada parell format per una lletra minúscula seguida d'una lletra majúscula. Per exemple, "sToP" es validarà amb aquest patró, però no "Stop" o "STOP" o "stop":

```

<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="([a-z][A-Z])+" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

L'exemple següent defineix un element anomenat "gender" amb una restricció. L'únic valor acceptable és home o dona:

```

<xs:element name="gender">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="male|female" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

```

</xs:simpleType>
</xs:element>

```

L'exemple següent defineix un element anomenat "password" amb una restricció. Hi ha d'haver exactament vuit caràcters seguits i aquests caràcters han de ser lletres minúscules o majúscules de la a a la z, o un número del 0 al 9:

```

<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z0-9]{8}" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>

```

Restriccions als caràcters d'espais en blanc

Per especificar com s'han de gestionar els caràcters d'espai en blanc, utilitzaríem la restricció d'espai en blanc.

Amb `whiteSpace value="preserve"` NO s'eliminarà cap caràcter d'espai en blanc:

```

<xs:element name="address">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="preserve" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>

```

Aquest exemple també defineix un element anomenat "address" amb una restricció. La restricció d'espai blanc s'estableix a "reemplaçar", la qual cosa significa que el processador XML SUBSTITUIRÀ tots els caràcters d'espais en blanc (avís de línia, tabulacions, espais i returns de carro) per espais:

```

<xs:element name="address">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="replace" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>

```

Aquest exemple també defineix un element anomenat "address" amb una restricció. La restricció d'espai en blanc s'estableix en "replega", el que significa que el processador XML ELIMINARÀ tots els caràcters d'espai en blanc (els salts de línia, les tabulacions, els espais, els returns de carro) es substitueixen per espais, els espais inicials i finals s'eliminen i els espais múltiples es redueixen a un sol espai):

```

<xs:element name="address">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="collapse" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>

```

Restriccions de longitud

Per limitar la longitud d'un valor en un element, utilitzaríem les restriccions `length`, `maxLength` i `minLength`.

Aquest exemple defineix un element anomenat "password" amb una restricció. El valor ha de tenir exactament vuit caràcters:

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:length value="8"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Aquest exemple defineix un altre element anomenat "password" amb una restricció. El valor ha de tenir un mínim de cinc caràcters i un màxim de vuit caràcters:

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:minLength value="5"/>
      <xs:maxLength value="8"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Restriccions per a tipus de dades

| Constraint | Description |
|----------------|---|
| enumeration | Defines a list of acceptable values |
| fractionDigits | Specifies the maximum number of decimal places allowed. Must be equal to or greater than zero |
| length | Specifies the exact number of characters or list items allowed. Must be equal to or greater than zero |
| maxExclusive | Specifies the upper bounds for numeric values (the value must be less than this value) |
| maxInclusive | Specifies the upper bounds for numeric values (the value must be less than or equal to this value) |
| maxLength | Specifies the maximum number of characters or list items allowed. Must be equal to or greater than zero |
| minExclusive | Specifies the lower bounds for numeric values (the value must be greater than this value) |
| minInclusive | Specifies the lower bounds for numeric values (the value must be greater than or equal to this value) |
| minLength | Specifies the minimum number of characters or list items allowed. Must be equal to or greater than zero |
| pattern | Defines the exact sequence of characters that are acceptable |
| totalDigits | Specifies the exact number of digits allowed. Must be greater than zero |
| whiteSpace | Specifies how white space (line feeds, tabs, spaces, and carriage returns) is handled |

13. Elements complexos XSD

Què és un element complex?

Un element complex és un element XML que conté altres elements i/o atributs.

Hi ha quatre tipus d'elements complexos:

- elements buits
- elements que només contenen altres elements
- elements que només contenen text
- elements que contenen tant altres elements com text

Nota: cadascun d'aquests elements també pot contenir atributs!

Exemples d'elements complexos

Un element XML complex, "product", que està buit:

```
<product pid="1345"/>
```

Un element XML complex, "employee", que només conté altres elements:

```
<employee>
  <firstname>John</firstname>
  <lastname>Smith</lastname>
</employee>
```

Un element XML complex, "food", que només conté text:

```
<food type="dessert">Ice cream</food>
```

Un element XML complex, "description", que conté tant elements com text:

```
<description>
  It happened on <date lang="norwegian">03.03.99</date> ....
</description>
```

Com definir un element complex

Mireu aquest element XML complex, "employee", que només conté altres elements:

```
<employee>
  <firstname>John</firstname>
  <lastname>Smith</lastname>
</employee>
```

Podem definir un element complex en un esquema XML de dues maneres diferents:

1. L'element "employee" es pot declarar directament anomenant l'element, així:

```
<xs:element name="employee">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Si utilitzeu el mètode descrit anteriorment, només l'element "empleat" pot utilitzar el tipus complex especificat. Tingueu en compte que els elements secundaris, "firstname" i "lastname", estan envoltats per l'indicador `<sequence>`. Això vol dir que els elements fills han d'aparèixer en el mateix ordre en què es declaren.

2. L'element "employee" pot tenir un atribut de tipus que fa referència al nom del tipus complex que cal utilitzar:

```
<xs:element name="employee" type="personinfo"/>
```

```

<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

```

Si utilitzeu el mètode descrit anteriorment, diversos elements poden fer referència al mateix tipus complex, com aquest:

```

<xs:element name="employee" type="personinfo"/>
<xs:element name="student" type="personinfo"/>
<xs:element name="member" type="personinfo"/>

<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

```

També podeu basar un tipus complex en un tipus complex existent i afegir alguns elements, com aquest:

```

<xs:element name="employee" type="fullpersoninfo"/>

<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="fullpersoninfo">
  <xs:complexContent>
    <xs:extension base="personinfo">
      <xs:sequence>
        <xs:element name="address" type="xs:string"/>
        <xs:element name="city" type="xs:string"/>
        <xs:element name="country" type="xs:string"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

14. Indicadors XSD

Els indicadors XSD són elements que s'utilitzen en els esquemes XSD per definir com es poden estructurar els elements dins d'un document XML. Aquests indicadors permeten especificar com es poden agrupar i repetir els elements dins d'un esquema XML.

Tipus principals d'indicadors XSD:

xs:sequence Indica que els elements fills han d'aparèixer en un ordre específic.

Exemple:

```

<xs:complexType>
  <xs:sequence>
    <xs:element name="nom" type="xs:string"/>

```

```
<xs:element name="cognom" type="xs:string"/>
</xs:sequence>
</xs:complexType>
```

En aquest cas, l'element <nom> ha d'aparèixer abans de <cognom> en l'XML.

xs:choice Permet que només un dels elements definits puga aparèixer.

Exemple:

```
<xs:complexType>
<xs:choice>
    <xs:element name="telèfon" type="xs:string"/>
    <xs:element name="email" type="xs:string"/>
</xs:choice>
</xs:complexType>
```

En l'XML resultant, només pot aparèixer o bé <telèfon> o bé <email>, però no tots dos.

xs:all Indica que tots els elements fills han d'aparèixer exactament una vegada, però poden estar en qualsevol ordre.

Exemple:

```
<xs:complexType>
<xs:all>
    <xs:element name="adreça" type="xs:string"/>
    <xs:element name="codiPostal" type="xs:string"/>
</xs:all>
</xs:complexType>
```

Tant <adreça> com <codiPostal> han d'aparèixer, però l'ordre no importa.

Altres propietats útils en els indicadors XSD:

minOccurs: especifica el nombre mínim de vegades que un element pot aparèixer.

maxOccurs: especifica el nombre màxim de vegades que un element pot aparèixer.

Exemple combinat:

```
<xs:sequence>
    <xs:element name="telèfon" type="xs:string" minOccurs="0" maxOccurs="3"/>
</xs:sequence>
```

Ací l'element <telèfon> pot aparèixer entre 0 i 3 vegades.

En resum, els indicadors XSD s'utilitzen per definir com s'agrupen i es repeteixen els elements dins d'un esquema XML, ajudant a estructurar correctament les dades.

Tema 6: Conversió i adaptació de documents XML (XSLT)



- [1. Introducció a XSLT](#)
 - [2. Llenguatges XSL](#)
 - [3. Transformació XSLT](#)
 - [4. L'element <xsl:template>](#)
 - [5. L'element <xsl:value-of>](#)
 - [6. L'element <xsl:for-each>](#)
 - [7. L'element <xsl:sort>](#)
 - [8. L'element <xsl:if>](#)
 - [9. L'element <xsl:choose>](#)
 - [10. L'element <xsl:apply-templates>](#)
 - [11. XSLT al client](#)
 - [12. XSLT al servidor](#)
-

1. Introducció a XSLT

XSL (eXtensible Stylesheet Language) és un llenguatge d'estil per a XML.

XSLT significa transformacions XSL. Aquest tema us ensenyarà com utilitzar XSLT per transformar documents XML en altres formats (bàsicament com transformar XML a HTML).

Un exemple de XSLT

```
<?xml version="1.0"?>

<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <html>
      <body>
        <h2>My CD Collection</h2>
        <table border="1">
          <tr bgcolor="#9acd32">
            <th>Title</th>
            <th>Artist</th>
          </tr>
          <xsl:for-each select="catalog/cd">
            <tr>
              <td><xsl:value-of select="title"/></td>
              <td><xsl:value-of select="artist"/></td>
            </tr>
          </xsl:for-each>
        </table>
      </body>
    </html>
  </xsl:template>

</xsl:stylesheet>
```

2. Llenguatges XSL.

XSLT és un llenguatge per a transformar documents XML.

XPath és un llenguatge per navegar per documents XML.

XQuery és un llenguatge per consultar documents XML.

Tot va començar amb XSL

XSL significa eXtensible Stylesheet Langage .

El World Wide Web Consortium (W3C) va començar a desenvolupar XSL perquè hi havia una necessitat d'un llenguatge de fulls d'estils basat en XML.

CSS = Fulls d'estil per a HTML

HTML utilitza etiquetes predefinides. S'entén bé el significat de cada etiqueta i com es mostra.

CSS s'utilitza per afegir estils als elements HTML.

XSL = Fulls d'estil per a XML

XML no utilitza etiquetes predefinides i, per tant, no s'entén bé el significat de cada etiqueta.

Un element <taula> podria indicar una taula HTML, un moble o una altra cosa, i els navegadors no saben com mostrar-ho!

Per tant, XSL descriu com s'han de mostrar els elements XML.

XSL - Més que un llenguatge de fulls d'estil

XSL consta de 3 parts:

- XSLT - un llenguatge per transformar documents XML
- XPath: un llenguatge per navegar per documents XML
- XQuery: un llenguatge per consultar documents XML

Què és XSLT?

- XSLT significa transformacions XSL
- XSLT és la part més important de XSL
- XSLT transforma un document XML en un altre document XML
- XSLT utilitza XPath per navegar per documents XML
- XSLT és una recomanació del W3C

XSLT = transformacions XSL

XSLT és la part més important de XSL. XSLT s'utilitza per transformar un document XML en un altre document XML o un altre tipus de document reconegut per un navegador, com ara HTML i XHTML. Normalment XSLT ho fa transformant cada element XML en un element (X)HTML.

Amb XSLT podeu afegir/eliminar elements i atributs al fitxer d'eixida o des del mateix. També podeu reordenar i ordenar els elements, fer proves i prendre decisions sobre quins elements voleu amagar i mostrar, i molt més.

Una manera habitual de descriure el procés de transformació és dir que XSLT transforma un arbre d'origen XML en un arbre de resultats XML .

XSLT utilitza Xpath per trobar informació en un document XML. XPath s'utilitza per navegar per elements i atributs en documents XML.

Com Funciona? En el procés de transformació, XSLT utilitza XPath per definir parts del document font que han de coincidir amb una o més plantilles predefinides. Quan es troba una coincidència, XSLT transformarà la part coincident del document d'origen en el document de resultats.

3. XSLT - Transformació

Exemple: Com transformar XML en XHTML mitjançant XSLT?

Declaració correcta del full d'estil

L'element arrel que declara que el document és un full d'estil XSL és `<xsl:stylesheet>` o `<xsl:transform>`.

Nota: <xsl:stylesheet> i <xsl:transform> són completament sinònims i es poden utilitzar qualsevol!

La manera correcta de declarar un full d'estil XSL segons la recomanació XSLT del W3C és:

```
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  o
<xsl:transform version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

Per accedir als elements, atributs i característiques XSLT hem de declarar l'espai de noms XSLT a la part superior del document.

El `xmlns:xsl="http://www.w3.org/1999/XSL/Transform"` apunta a l'espai de noms XSLT oficial del W3C. Si utilitzeu aquest espai de noms, també heu d'incloure l'atribut `version="1.0"`.

Comenceu amb un document XML en brut. Volem transformar el següent document XML ("cdcatalog.xml") en XHTML:

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
  .
  .
</catalog>
```

Creeu un full d'estil XSL

A continuació, creeu un full d'estil XSL ("cdcatalog.xsl") amb una plantilla de transformació:

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

<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <html>
      <body>
        <h2>My CD Collection</h2>
        <table border="1">
          <tr bgcolor="#9acd32">
            <th>Title</th>
            <th>Artist</th>
          </tr>
          <xsl:for-each select="catalog/cd">
```

```

<tr>
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="artist"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>

</xsl:stylesheet>

```

Enllaceu el full d'estil XSL al document XML. Afegiu la referència del full d'estil XSL al vostre document XML ("cdcatalog.xml"):

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="cdcatalog.xsl"?>
<catalog>
    <cd>
        <title>Empire Burlesque</title>
        <artist>Bob Dylan</artist>
        <country>USA</country>
        <company>Columbia</company>
        <price>10.90</price>
        <year>1985</year>
    </cd>
    .
    .
</catalog>

```

Si teniu un navegador compatible amb XSLT, transformarà el vostre XML en XHTML.

My CD Collection

| Title | Artist |
|------------------|--------------|
| Empire Burlesque | Bob Dylan |
| Hide your heart | Bonnie Tyler |

i continua...

4. L'element <xsl:template>

Un full d'estil XSL consta d'un o més conjunt de regles que s'anomenen plantilles. Una plantilla conté regles per aplicar quan coincideix un node específicat.

L'element `<xsl:template>` s'utilitza per crear plantilles. L'atribut `match` s'utilitza per associar una plantilla amb un element XML. L'atribut `match` també es pot utilitzar per definir una plantilla per a tot el document XML. El valor de l'atribut `match` és una expressió XPath (és a dir, `match="/"` defineix tot el document).

D'acord, mirem una versió simplificada del fitxer XSL del punt anterior:

Exemple

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0">

```

```

xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
    <body>
      <h2>My CD Collection</h2>
      <table border="1">
        <tr bgcolor="#9acd32">
          <th>Title</th>
          <th>Artist</th>
        </tr>
        <tr>
          <td>.</td>
          <td>.</td>
        </tr>
      </table>
    </body>
  </html>
</xsl:template>

</xsl:stylesheet>

```

Resultat:

My CD Collection

| Title | Artist |
|-------|--------|
| . | . |
| . | . |

Exemple explicat

Com que un full d'estil XSL és un document XML, sempre comença amb la declaració XML: `<?xml version="1.0" encoding="UTF-8"?>`.

El següent element, `<xsl:stylesheet>`, defineix que aquest document és un document de full d'estil XSLT (juntament amb el número de versió i els atributs de l'espai de noms XSLT).

L'element `<xsl:template>` defineix una plantilla. L'atribut `match="/"` associa la plantilla amb l'arrel del document font XML.

El contingut de l'element `<xsl:template>` defineix una mica d'HTML per escriure a l'eixida.

Les dues últimes línies defineixen el final de la plantilla i el final del full d'estil.

El resultat d'aquest exemple és una mica decebedor, perquè no es van copiar dades del document XML a l'eixida. En el següent punt aprendràs a utilitzar l'element `<xsl:value-of>` per seleccionar valors dels elements XML.

5. L'element `<xsl:value-of>`

L'element `<xsl:value-of>` s'utilitza per extreure el valor d'un node seleccionat.

L'element `<xsl:value-of>` es pot utilitzar per extreure el valor d'un element XML i afegir-lo al flux d'eixida de la transformació:

Exemple

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <html>
      <body>
        <h2>My CD Collection</h2>
        <table border="1">
          <tr bgcolor="#9acd32">
            <th>Title</th>
            <th>Artist</th>
          </tr>
          <tr>
            <td><xsl:value-of select="catalog/cd/title"/></td>
            <td><xsl:value-of select="catalog/cd/artist"/></td>
          </tr>
        </table>
      </body>
    </html>
  </xsl:template>

</xsl:stylesheet>
```

My CD Collection

| Title | Artist |
|------------------|-----------|
| Empire Burlesque | Bob Dylan |

Exemple explicat

Nota: l'atribut **select**, a l'exemple anterior, conté una expressió XPath. Una expressió XPath funciona com si navegarem per un sistema de fitxers. Una barra inclinada (/) selecciona subdirectoris.

El resultat de l'exemple anterior continua sent una mica decebedor. Només s'ha copiat una línia de dades del document XML a l'eixida. Al punt següent aprendràs a utilitzar l'element **<xsl:for-each>** per recórrer els elements XML i mostrar tots els registres.

6. L'element **<xsl:for-each>**

L'element **<xsl:for-each>** us permet fer bucles en XSLT.

L'element XSL **<xsl:for-each>** es pot utilitzar per seleccionar tots els elements XML d'un conjunt de nodes especificat:

Exemple

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <html>
      <body>
        <h2>My CD Collection</h2>
```

```

<table border="1">
  <tr bgcolor="#9acd32">
    <th>Title</th>
    <th>Artist</th>
  </tr>
  <xsl:for-each select="catalog/cd">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
    </tr>
  </xsl:for-each>
</table>
</body>
</html>
</xsl:template>

</xsl:stylesheet>

```

My CD Collection

| Title | Artist |
|---------------------|--------------|
| Empire Burlesque | Bob Dylan |
| Hide your heart | Bonnie Tyler |
| Greatest Hits | Dolly Parton |
| Still got the blues | Gary Moore |

i continua...

Filtrat de l'eixida

També podem filtrar l'eixida del fitxer XML, ¿com?: afegint un criteri a l'atribut select de l'element `<xsl:for-each>`.

```
<xsl:for-each select="catalog/cd[artist='Bob Dylan']">
```

Els operadors de filtres legals són:

- = (igual)
- != (no igual)
- < menys que
- > més gran que

Exemple

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <html>
      <body>
        <h2>My CD Collection</h2>
        <table border="1">
          <tr bgcolor="#9acd32">
            <th>Title</th>

```

```

<th>Artist</th>
</tr>
<xsl:for-each select="catalog/cd[artist='Bob Dylan']">
<tr>
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="artist"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>

</xsl:stylesheet>

```

Resultat:

| Title | Artist |
|------------------|-----------|
| Empire Burlesque | Bob Dylan |

7. Element <xsl:sort>

L'element <xsl:sort> s'utilitza per ordenar l'eixida.

On posar la informació d'ordenació

Per a ordenar l'eixida, només cal que afegiu un element <xsl:sort> dins de l'element <xsl:for-each> al fitxer XSL:

Exemple

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
    <body>
      <h2>My CD Collection</h2>
      <table border="1">
        <tr bgcolor="#9acd32">
          <th>Title</th>
          <th>Artist</th>
        </tr>
        <xsl:for-each select="catalog/cd">
          <xsl:sort select="artist"/>
          <tr>
            <td><xsl:value-of select="title"/></td>
            <td><xsl:value-of select="artist"/></td>
          </tr>
        </xsl:for-each>
      </table>
    </body>
  </html>
</xsl:template>

</xsl:stylesheet>

```

Resultat:

| Title | Artist |
|------------------|----------------|
| Romanza | Andrea Bocelli |
| One night only | Bee Gees |
| Empire Burlesque | Bob Dylan |
| Hide your heart | Bonnie Tyler |
| The very best of | Cat Stevens |
| Greatest Hits | Dolly Parton |
| | |

i continua...

Nota: l'atribut **select** indica quin element XML cal ordenar.

8. L'element <xsl:if>

L'element <xsl:if>

Per a posar un condicional al contingut del fitxer XML, afegiu un element **<xsl:if>** al document XSL.

Sintaxi

```
<xsl:if test="expression">
    ...some output if the expression is true...
</xsl:if>
```

On posar l'element <xsl:if>

Per a afegir un condicional, afegiu l'element **<xsl:if>** dins de l'element **<xsl:for-each>** al fitxer XSL:

Exemple

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <html>
      <body>
        <h2>My CD Collection</h2>
        <table border="1">
          <tr bgcolor="#9acd32">
            <th>Title</th>
            <th>Artist</th>
            <th>Price</th>
          </tr>
          <xsl:for-each select="catalog/cd">
            <xsl:if test="price > 10">
              <tr>
                <td><xsl:value-of select="title"/></td>
                <td><xsl:value-of select="artist"/></td>
                <td><xsl:value-of select="price"/></td>
              </tr>
            </xsl:if>
          </xsl:for-each>
        </table>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

```

</table>
</body>
</html>
</xsl:template>

</xsl:stylesheet>

```

Resultat:

| Title | Artist | Price |
|----------------------|----------------|-------|
| Empire Burlesque | Bob Dylan | 10.90 |
| Still got the blues | Gary Moore | 10.20 |
| One night only | Bee Gees | 10.90 |
| Romanza | Andrea Bocelli | 10.80 |
| Black angel | Savage Rose | 10.90 |
| 1999 Grammy Nominees | Many | 10.20 |

Nota: el valor de l' atribut **de prova** requerit conté l'expressió que cal avaluar. El codi anterior només mostrarà el títol i els elements de l'artista dels CD que tinguen un preu superior a 10.

9. L'element <xsl:choose>

L'element **<xsl:choose>** s'utilitza juntament amb **<xsl:when>** i **<xsl:otherwise>** per expressar múltiples proves condicionals.

Sintaxi

```

<xsl:choose>
  <xsl:when test="expression">
    ... some output ...
  </xsl:when>
  <xsl:otherwise>
    ... some output ....
  </xsl:otherwise>
</xsl:choose>

```

On posar la condició de selecció

Per a inserir una prova condicional múltiple en el fitxer XML, afegiu els elements **<xsl:choose>**, **<xsl:when>** i **<xsl:otherwise>** al fitxer XSL:

Exemple

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <html>
      <body>
        <h2>My CD Collection</h2>
        <table border="1">
          <tr bgcolor="#9acd32">
            <th>Title</th>
            <th>Artist</th>

```

```

</tr>
<xsl:for-each select="catalog/cd">
<tr>
  <td><xsl:value-of select="title"/></td>
  <xsl:choose>
    <xsl:when test="price > 10">
      <td bgcolor="#ff00ff">
        <xsl:value-of select="artist"/></td>
    </xsl:when>
    <xsl:otherwise>
      <td><xsl:value-of select="artist"/></td>
    </xsl:otherwise>
  </xsl:choose>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>

</xsl:stylesheet>

```

El codi anterior afegirà un color de fons rosa a la columna "Artista" QUAN el preu del CD siga superior a 10.

| Title | Artist |
|---------------------|-----------------|
| Empire Burlesque | Bob Dylan |
| Hide your heart | Bonnie Tyler |
| Greatest Hits | Dolly Parton |
| Still got the blues | Gary Moore |
| Eros | Eros Ramazzotti |
| One night only | Bee Gees |
| Sylvias Mother | Dr.Hook |
| Maggie May | Rod Stewart |
| Romanza | Andrea Bocelli |

Un altre exemple

Ací hi ha un altre exemple que conté dos elements `<xsl:when>`:

Exemple

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <html>
      <body>
        <h2>My CD Collection</h2>
        <table border="1">
          <tr bgcolor="#9acd32">
            <th>Title</th>
            <th>Artist</th>
          </tr>
          <xsl:for-each select="catalog/cd">

```

```

<tr>
    <td><xsl:value-of select="title"/></td>
    <xsl:choose>
        <xsl:when test="price &gt; 10">
            <td bgcolor="#ff00ff">
                <xsl:value-of select="artist"/></td>
        </xsl:when>
        <xsl:when test="price &gt; 9">
            <td bgcolor="#cccccc">
                <xsl:value-of select="artist"/></td>
        </xsl:when>
        <xsl:otherwise>
            <td><xsl:value-of select="artist"/></td>
        </xsl:otherwise>
    </xsl:choose>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>

</xsl:stylesheet>

```

El codi anterior afegirà un color de fons rosa a la columna "Artista" QUAN el preu del CD siga superior a 10, i un color de fons gris QUAN el preu del CD siga superior a 9 i inferior o igual a 10.

10. L'element <xsl:apply-templates>

L'element **<xsl:apply-templates>** aplica una plantilla a l'element actual o als nodes fills de l'element actual.

Si afegim un atribut "select" a l'element **<xsl:apply-templates>**, només processarà els elements secundaris que coincideisquen amb el valor de l'atribut. Podem utilitzar l'atribut "select" per especificar en quin ordre s'han de processar els nodes fills.

Mireu el següent full d'estil XSL:

Exemple

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
    <html>
        <body>
            <h2>My CD Collection</h2>
            <xsl:apply-templates/>
        </body>
    </html>
</xsl:template>

<xsl:template match="cd">
    <p>
        <xsl:apply-templates select="title"/>
        <xsl:apply-templates select="artist"/>
    </p>

```

```

</xsl:template>

<xsl:template match="title">
  Title: <span style="color:#ff0000">
    <xsl:value-of select="."/></span>
    <br />
  </xsl:template>

<xsl:template match="artist">
  Artist: <span style="color:#00ff00">
    <xsl:value-of select="."/></span>
    <br />
  </xsl:template>

</xsl:stylesheet>

```

My CD Collection

Title: Empire Burlesque

Artist: Bob Dylan

Title: Hide your heart

Artist: Bonnie Tyler

Title: Greatest Hits

Artist: Dolly Parton

Title: Still got the blues

Artist: Gary Moore

Title: Eros

Artist: Eros Ramazzotti

11. XSLT - Al client

XSLT es pot utilitzar per transformar un document XML en un document a XHTML al vostre navegador. Però no sempre funcionarà o serà suficient.

Una solució de JavaScript

En els punts anteriors hem explicat com es pot utilitzar XSLT per transformar un document d'XML a XHTML. Ho vam fer afegint un full d'estil XSL al fitxer XML i deixant que el navegador faca la transformació.

Fins i tot si això funciona bé, no sempre és desitjable incloure una referència de full d'estil en un fitxer XML (per exemple, no funcionarà en un navegador que no siga compatible amb XSLT).

Una solució més versàtil seria utilitzar un JavaScript per fer la transformació.

Mitjançant un JavaScript, podem:

- fer proves específiques del navegador
- utilitzar diferentsfulls d'estil segons les necessitats del navegador i de l'usuari

Aquesta és la bellesa de XSLT! Un dels objectius de disseny de XSLT era fer possible la transformació de dades d'un format a un altre, donant suport a diferents navegadors i diferents necessitats dels usuaris.

Transformar XML a XHTML al navegador

Ací teniu el codi font necessari per transformar el fitxer XML a XHTML al client:

Exemple

```
<!DOCTYPE html>
<html>
<head>
<script>
function loadXMLDoc(filename)
{
if (window.ActiveXObject)
{
    xhttp = new ActiveXObject("Msxml2.XMLHTTP");
}
else
{
    xhttp = new XMLHttpRequest();
}
xhttp.open("GET", filename, false);
try {xhttp.responseType = "msxml-document"} catch(err) {} // Helping IE11
xhttp.send("");
return xhttp.responseText;
}

function displayResult()
{
xml = loadXMLDoc("cdcatalog.xml");
xsl = loadXMLDoc("cdcatalog.xsl");
// code for IE
if (window.ActiveXObject || xhttp.responseType == "msxml-document")
{
    ex = xml.transformNode(xsl);
    document.getElementById("example").innerHTML = ex;
}
// code for Chrome, Firefox, Opera, etc.
else if (document.implementation && document.implementation.createDocument)
{
    xsltProcessor = new XSLTProcessor();
    xsltProcessor.importStylesheet(xsl);
    resultDocument = xsltProcessor.transformToFragment(xml, document);
    document.getElementById("example").appendChild(resultDocument);
}
}
</script>
</head>
<body onload="displayResult()">
<div id="example" />
</body>
</html>
```

Aquesta és una solució amb JavaScript.

12. XSLT - Al servidor

Per fer que les dades XML estiguin disponibles per a tot tipus de navegadors, podem transformar el document XML al SERVIDOR i enviar-lo de nou al navegador com a XHTML.

Una solució de navegador creuat

En el punt anterior vam explicar com es pot utilitzar XSLT per transformar un document d'XML a XHTML al navegador. Hem utilitzat un analitzador de JavaScript i XML per a la transformació. Però, això no funcionarà en un navegador que no tinga un analitzador XML.

Per fer que les dades XML estiguin disponibles per a tot tipus de navegadors, podem transformar el document XML al servidor i enviar-lo de nou al navegador com a XHTML.

Aquesta és una altra bellesa de XSLT. Un dels objectius de disseny de XSLT era fer possible transformar les dades d'un format a un altre en un servidor, retornant dades llegibles a tot tipus de navegadors.

Partim del XML i XSL del punt anterior. Tingueu en compte que el fitxer XML no té cap referència al fitxer XSL.

Codi PHP: transforma XML a XHTML al servidor

Ací teniu el codi font PHP necessari per transformar el fitxer XML a XHTML al servidor:

```
<?php
// Load XML file
$xml = new DOMDocument;
$xml->load('cdcatalog.xml');

// Load XSL file
$xsl = new DOMDocument;
$xsl->load('cdcatalog.xsl');

// Configure the transformer
$proc = new XSLTProcessor;

// Attach the xsl rules
$proc->importStyleSheet($xsl);

echo $proc->transformToXML($xml);
?>
```

Mireu el video: <https://youtu.be/DlELzYxARmE>

<xpath>

1. BaseX

2. Xpath

3. Nodes

 3.1 Terminologia Xpath

 3.2 Relació de nodes

4. Sintaxi Xpath

5. Eixos Xpath

6. Operadors Xpath

7. Exemples Xpath

8. Xquery

9. Exemple de XQuery

1. BaseX

BaseX és un sistema de gestió de bases de dades XML que es fa servir per emmagatzemar, consultar i gestionar dades XML d'una manera eficient. Pots realitzar una gran varietat de tasques amb BaseX i fitxers XML. Lloc oficial: <https://baseX.org>

Ací tens algunes coses que pots fer:

1. Emmagatzemar dades XML: Pots utilitzar BaseX per emmagatzemar arxius XML de manera eficient en una base de dades. Això facilita la gestió i l'accés a les dades XML, especialment quan es tracta de grans volums d'informació.
2. Consultar dades XML: BaseX et permet realitzar consultes sobre les dades XML emmagatzemades a la base de dades utilitzant el llenguatge de consulta XQuery. Pots buscar, filtrar i extreure informació específica dels documents XML.
3. Transformació XSLT: Pots utilitzar BaseX per aplicar transformacions XSLT (Extensible Stylesheet Language Transformations) als documents XML emmagatzemats. Això et permet canviar el format o l'estruccura de les dades XML d'acord amb les teves necessitats.
4. Indexació: BaseX proporciona un mecanisme eficient d'indexació de documents XML, la qual cosa facilita la recuperació ràpida d'informació específica en grans conjunts de dades XML.
5. Actualització i manteniment: Pots utilitzar BaseX per actualitzar documents XML emmagatzemats, inserir noves dades, eliminar documents i realitzar altres operacions de manteniment a la base de dades.
6. Administració de permisos: BaseX permet administrar permisos i control d'accés a documents XML emmagatzemats, la qual cosa és útil en entorns on es requereix seguretat en l'accés a les dades.
7. Integració amb altres sistemes: Pots integrar BaseX amb altres aplicacions i sistemes a través d'API i protocols estàndard com REST i WebDAV.
8. Consultes espacials: BaseX també admet extensions per a consultes espacials, la qual cosa és útil en aplicacions geoespaciales que utilitzen dades XML.
9. Exportació i importació: Pots exportar dades XML de la base de dades en diversos formats, com XML, JSON o text, i després importar dades en diferents formats per al seu posterior processament.
10. Recuperació de versions: BaseX ofereix capacitats de control de versions per a dades XML, la qual cosa permet fer un seguiment dels canvis en els documents XML al llarg del temps.

En resum, BaseX és una eina poderosa per a gestionar dades XML i realitzar una àmplia varietat de tasques, des de l'emmagatzematge i consulta de dades XML fins a la transformació, actualització i gestió de permisos. La seva flexibilitat i capacitat per gestionar grans conjunts de dades XML el fan valuós en una varietat d'aplicacions.

2. XPath

XPath és un llenguatge de consulta utilitzat per accedir i navegar a través d'elements d'un document XML. Aquest llenguatge permet especificar la ubicació d'elements o nodes dins d'un document XML mitjançant rutes i patrons.

Es fa servir àmpliament en el context de la recuperació d'informació de documents XML i la manipulació d'aquesta informació, com ara la selecció d'elements, l'extracció de dades i la navegació dins de jerarquies XML.

XPath és una eina important en el món del desenvolupament web i la gestió de dades estructurades.

3. Nodes

3.1. Terminologia Xpath

A XPath, hi ha set tipus de nodes: element, atribut, text, espai de noms, instrucció de processament, comentari i node arrel.

Els documents XML es tracten com arbres de nodes. L'element superior de l'arbre s'anomena element arrel.

Mireu el següent document XML:

```
<?xml version="1.0" encoding="UTF-8"?>  
  
<bookstore>  
  <book>  
    <title lang="en">Harry Potter</title>  
    <author>J K. Rowling</author>  
    <year>2005</year>  
    <price>29.99</price>  
  </book>  
</bookstore>
```

Exemple de nodes al document XML anterior:

```
<bookstore> (root element node)  
<author>J K. Rowling</author> (element node)  
lang="en" (attribute node)
```

Valors atòmics: Els valors atòmics són nodes sense fills ni pares. Com ara: J K. Rowling o "en"

Elements: Els elements són valors atòmics o nodes.

3.2. Relació de nodes

Pare: Cada element i atribut té un pare. En l'exemple anterior, l'element del llibre és el pare del títol, autor, any i preu.

Fills: Els elements node poden tenir zero, un o més fills.

En l'exemple anterior, el títol, l'autor, l'any i els elements de preu són tots fills de l'element llibre.

Germans: Nodes que tenen el mateix pare. En l'exemple anterior, el títol, l'autor, l'any i els elements de preu són tots germans.

Avantpassats: Pare d'un node, iaio d'un node, etc. En l'exemple anterior, els avantpassats de l'element títol són l'element llibre i l'element llibreria:

Descendents: Els fills d'un node, els nets, etc. En l'exemple anterior, els descendents de l'element llibreria són el llibre, el títol, l'autor, l'any i els elements de preu.

4. Sintaxi XPath

XPath utilitza expressions de camí per seleccionar nodes o conjunts de nodes en un document XML. El node es selecciona seguint un camí o passos.

Utilitzarem el document XML aquest als exemples següents.

```
<?xml version="1.0" encoding="UTF-8"?>  
  
<bookstore>  
  
  <book>  
    <title lang="en">Harry Potter</title>  
    <price>29.99</price>  
  </book>  
  
  <book>  
    <title lang="en">Learning XML</title>  
    <price>39.95</price>  
  </book>  
  
</bookstore>
```

Selecció de nodes

XPath utilitza expressions de camí per seleccionar nodes en un document XML. El node es selecciona seguint un camí o passos. Les expressions de camí més útils es mostren a continuació:

Expressió Descripció

`nodename` Selecciona tots els nodes amb el nom "nodename"

`/` Selecciona des del node arrel

`//` Selecciona els nodes del document del node actual que coincideixen amb la selecció, independentment d'on es troben

`.` Selecciona el node actual

`..` Selecciona el pare del node actual

`@` Selecciona atributs

A la taula següent hem enumerat algunes expressions de camí i el resultat de les expressions:

| Expressió | Resultat |
|-----------------|--|
| bookstore | Selecnia tots els nodes amb el nom "bookstore" Selecciona el element root bookstore |
| /bookstore | Nota: si el camí comença amb (/), sempre representa un camí absolut a un element! |
| bookstore/book | Selecciona tots els elements book que són fills de bookstore |
| //book | Selecciona tots els elements book sense importar on es troben al document |
| bookstore//book | Selecciona tots els elements book que són descendents de l'element bookstore, independentment d'on es troben |
| //@lang | Selecciona tots els atributs anomenats lang |

Predicats

Els predicats s'utilitzen per trobar un node específic o un node que continga un valor específic.

Els predicats sempre s'incorporen entre claudàtors.

A la taula següent hem enumerat algunes expressions de camí amb predicats i el resultat de les expressions:

| Expressió | Resultat |
|------------------------------------|--|
| /bookstore/book[1] | Selects the first book element that is the child of the bookstore element. |
| /bookstore/book[last()] | Selects the last book element that is the child of the bookstore element |
| /bookstore/book[last()-1] | Selects the last but one book element that is the child of the bookstore element |
| /bookstore/book[position()<3] | Selects the first two book elements that are children of the bookstore element |
| //title[@lang] | Selects all the title elements that have an attribute named lang |
| //title[@lang='en'] | Selects all the title elements that have a "lang" attribute with a value of "en" |
| /bookstore/book[price>35.00] | Selects all the book elements of the bookstore element that have a price element with a value greater than 35.00 |
| /bookstore/book[price>35.00]/title | Selects all the title elements of the book elements of the bookstore element that have a price element with a value greater than 35.00 |

Selecció de nodes desconeguts

Els comodins XPath es poden utilitzar per seleccionar nodes XML desconeguts.

| Wildcard comodí | Descripció |
|--------------------|--------------------------|
| * | Matches any element node |

| | |
|---------------------|------------------------------|
| <code>@*</code> | Matches any attribute node |
| <code>node()</code> | Matches any node of any kind |

A la taula següent hem enumerat algunes expressions de camí i el resultat de les expressions:

| Expressió | Resultat |
|---------------------------|--|
| <code>/bookstore/*</code> | Selecciona tots els nodes descendents de l'element bookstore |
| <code>//*</code> | Selecciona tots els elements del document |
| <code>//title[@*]</code> | Selecciona tots els elements title que tinguen almenys un atribut de qualsevol tipus |

Selecció de diversos camins

Mitjançant l'ús del operador (`|`) en una expressió XPath podeu seleccionar diversos camins.

A la taula següent hem enumerat algunes expressions de camí i el resultat de les expressions:

| Expressió | Resultat |
|---|--|
| <code>//book/title //book/price</code> | Selects all the title AND price elements of all book elements |
| <code>//title //price</code> | Selects all the title AND price elements in the document |
| <code>/bookstore/book/ title //price</code> | Selects all the title elements of the book element of the bookstore element AND all the price elements in the document |

5 Eixos XPath

Eixos Xpath

Un eix representa una relació amb el node de context (actual) i s'utilitza per localitzar nodes en relació amb aquest node a l'arbre.

| Nom de l'eix | Resultat |
|---------------------------------|---|
| <code>ancestor</code> | Selects all ancestors (parent, grandparent, etc.) of the current node |
| <code>ancestor-or-self</code> | Selects all ancestors (parent, grandparent, etc.) of the current node and the current node itself |
| <code>attribute</code> | Selects all attributes of the current node |
| <code>child</code> | Selects all children of the current node |
| <code>descendant</code> | Selects all descendants (children, grandchildren, etc.) of the current node |
| <code>descendant-or-self</code> | Selects all descendants (children, grandchildren, etc.) of the current node and the current node itself |
| <code>following</code> | Selects everything in the document after the closing tag of the current node |
| <code>following-sibling</code> | Selects all siblings after the current node |
| <code>namespace</code> | Selects all namespace nodes of the current node |
| <code>parent</code> | Selects the parent of the current node |
| <code>preceding</code> | Selects all nodes that appear before the current node in the |

| | |
|-------------------|---|
| | document, except ancestors, attribute nodes and namespace nodes |
| preceding-sibling | Selects all siblings before the current node |
| self | Selects the current node |

Expressió del camí d'ubicació

Un camí d'ubicació pot ser absolut o relatiu.

Una ruta d'ubicació absoluta comença amb una barra inclinada (/) i una ruta d'ubicació relativa no. En ambdós casos, el camí d'ubicació consta d'un o més passos, cadascun separat per una barra inclinada:

An absolute location path:

/step/step/...

A relative location path:

step/step/...

Cada pas s'avalua en comparació amb els nodes del conjunt de nodes actual.

Un pas consisteix en:

- un eix (defineix la relació d'arbre entre els nodes seleccionats i el node actual)
- una prova de nodes (identifica un node dins d'un eix)
- zero o més predicats (per refinar encara més el conjunt de nodes seleccionat)

La sintaxi d'un pas d'ubicació és: axisname::nodetest[predicate]

Exemples:

| Exemple | Resultat |
|------------------------|---|
| child::book | Selects all book nodes that are children of the current node |
| attribute::lang | Selects the lang attribute of the current node |
| child::* | Selects all element children of the current node |
| attribute::* | Selects all attributes of the current node |
| child::text() | Selects all text node children of the current node |
| child::node() | Selects all children of the current node |
| descendant::book | Selects all book descendants of the current node |
| ancestor::book | Selects all book ancestors of the current node |
| ancestor-or-self::book | Selects all book ancestors of the current node - and the current as well if it is a book node |
| child::*/child::price | Selects all price grandchildren of the current node |

6 Operadors Xpath

A continuació es mostra una llista dels operadors que es poden utilitzar en expressions Xpath:

| Operador | Descripció | Example |
|----------|------------------------|---------------|
| | Computes two node-sets | //book //cd |
| + | Addition | 6 + 4 |
| - | Subtraction | 6 - 4 |

| | | |
|-----|------------------------------|--|
| * | Multiplication | $6 * 4$ |
| div | Division | $8 \text{ div } 4$ |
| = | Equal | $\text{price} = 9.80$ |
| != | Not equal | $\text{price} \neq 9.80$ |
| < | Less than | $\text{price} < 9.80$ |
| <= | Less than or equal to | $\text{price} \leq 9.80$ |
| > | Greater than | $\text{price} > 9.80$ |
| >= | Greater than or equal to | $\text{price} \geq 9.80$ |
| or | or | $\text{price} = 9.80 \text{ or } \text{price} = 9.70$ |
| and | and | $\text{price} > 9.00 \text{ and } \text{price} < 9.90$ |
| mod | Modulus (division remainder) | $5 \text{ mod } 2$ |

7. Exemples XPath

Intentem aprendre una mica de sintaxi XPath bàsica mirant alguns exemples.

Utilitzarem el document XML aquest als exemples següents. "books.xml":

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

<book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
</book>

<book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
</book>

<book category="web">
    <title lang="en">XQuery Kick Start</title>
    <author>James McGovern</author>
    <author>Per Bothner</author>
    <author>Kurt Cagle</author>
    <author>James Linn</author>
    <author>Vaidyanathan Nagarajan</author>
    <year>2003</year>
    <price>49.99</price>
</book>

<book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
</book>

</bookstore>
```

Seleccioneu tots els títols

L'exemple següent selecciona tots els nodes de títol:

/bookstore/book/title

The screenshot shows the XQEdit XML editor interface. On the left, there is a file browser with a tree view showing 'xml' as the selected folder, containing 'books.xml', 'cd's.xq', and 'cd_catalog.xml'. The main area is titled 'Editor' and contains a query editor window with the XQuery code: `1 /bookstore/book/title`. Below the editor is an 'OK' button and a timestamp '1 : 22'. At the bottom, there is a toolbar with icons for file operations and a 'Resultado' section.

Editor

cd's.xq* ☰
1 /bookstore/book/title

OK 1 : 22

Resultado

```
<title lang="en">Everyday Italian</title>
<title lang="en">Harry Potter</title>
<title lang="en">XQuery Kick Start</title>
<title lang="en">Learning XML</title>
```

Seleccioneu el títol del primer llibre:

/bookstore/book[1]/title

The screenshot shows the XQEdit XML editor interface. On the left, there is a file browser with a tree view showing 'xml' as the selected folder, containing 'books.xml', 'cd's.xq', and 'cd_catalog.xml'. The main area is titled 'Editor' and contains a query editor window with the XQuery code: `1 /bookstore/book[1]/title`. Below the editor is an 'OK' button and a timestamp '1 : 25'. At the bottom, there is a toolbar with icons for file operations and a 'Resultado' section.

Editor

cd's.xq* ☰
1 /bookstore/book[1]/title

OK 1 : 25

Resultado

```
<title lang="en">Everyday Italian</title>
```

Seleccioneu tots els preus: /bookstore/book/price[text()]

The screenshot shows the Oxygen XML Editor interface. On the left, there's a file browser with an orange-highlighted folder named 'xml' containing files like 'books.xml', 'cd's.xq', and 'cd_catalog.xml'. The main area is titled 'Editor' and contains an XQuery code editor with the query: `/bookstore/book/price[text()]`. Below the editor is a status bar showing 'OK' and the time '1 : 30'. To the right, a results pane titled 'Resultado' displays the output of the XQuery query:

```
<price>30.00</price>
<price>29.99</price>
<price>49.99</price>
<price>39.95</price>
```

Seleccioneu els nodes de preu amb preu>35
`/bookstore/book[price>35]/price`

Seleccioneu els nodes de títol amb preu>35
`/bookstore/book[price>35]/title`

8. Xquery

XQuery és a XML el que SQL és a bases de dades. XQuery està dissenyat per consultar dades XML.

Mentre que XPath s'utilitza principalment per navegar i seleccionar nodes en documents XML, XQuery és un llenguatge més complet que permet fer consultes avançades i transformacions sobre dades XML.

Exemple de Xquery:

```
for $x in doc("books.xml")/bookstore/book
where $x/price>30
order by $x/title
return $x/title
```

Què és Xquery?

- XQuery és l' idioma per consultar dades XML
- XQuery per a XML és com SQL per a bases de dades
- XQuery es basa en expressions XPath
- XQuery és compatible amb totes les bases de dades principals
- XQuery és una recomanació del W3C

XQuery i XPath

XQuery 1.0 i XPath 2.0 comparteixen el mateix model de dades i admeten les mateixes funcions i operadors. Si ja has estudiat XPath, no tindreu problemes per entendre XQuery.

XQuery - Exemples d'ús

XQuery es pot utilitzar per:

- Extraure informació per utilitzar-la en un servei web
- Generar informes resums
- Transformar dades XML a XHTML
- Cerqueu documents web per obtenir informació rellevant

XQuery és una recomanació del W3C

XQuery és compatible amb diversos estàndards del W3C, com ara XML, Namespaces, XSLT, XPath i XML Schema.

XQuery 1.0 es va convertir en una recomanació del W3C el 2007.

9. Exemple de XQuery

Aprenem una mica de XQuery bàsic mirant un exemple. Utilitzarem altra vegada [books.xml](#) als exemples següents.

Com seleccionar els nodes de "books.xml"?

Funcions: XQuery utilitza funcions per extreure dades de documents XML. La funció doc() s'utilitza per obrir el fitxer "books.xml": [doc\("books.xml"\)](#)

Expressions del camí: XQuery utilitza expressions de camí per navegar pels elements d'un document XML.

La següent expressió de camí s'utilitza per seleccionar tots els elements del títol al fitxer "books.xml": [doc\("books.xml"\)/bookstore/book/title](#)

The screenshot shows the Oxygen XML Editor interface. On the left, there's a file browser with a folder named 'xml' containing files like 'books.xml', 'cd's.xq', and 'cd_catalog.xml'. In the center, the 'Editor' pane displays the XQuery code: `1 doc("books.xml")/bookstore/book/title`. Below it, a status bar shows 'OK' and '1 : 38'. On the right, the 'Resultado' (Result) pane shows the output of the query:

```
<title lang="en">Everyday Italian</title>
<title lang="en">Harry Potter</title>
<title lang="en">XQuery Kick Start</title>
<title lang="en">Learning XML</title>
```

(/bookstore selecciona l'element llibreria, /book selecciona tots els elements del llibre en l'element llibreria i /title selecciona tots els elements del títol en cada element del llibre)

Predicats

XQuery utilitza predicats per limitar les dades extretes dels documents XML.

El predicat següent s'utilitza per seleccionar tots els elements del llibre de l'element llibreria que tenen un element de preu amb un valor inferior a 30:

```
doc("books.xml")/bookstore/book[price<30]
```

The screenshot shows the Oxygen XML Editor interface. On the left, there's a file browser with a folder named 'xml' containing 'books.xml' (825 b), 'cd's.xq' (29 b), and 'cd_catalog.xml' (4). The main area is titled 'Editor' and contains the XQuery code:

```
1 doc ("books.xml") /bookstore/book[price<30]
```

. Below the editor is a status bar showing '1 : 42'. At the bottom right is a 'Resultado' (Result) panel displaying the XML output:

```
<book category="children">
  <title lang="en">Harry Potter</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
</book>
```

Podeu vore el vídeo: https://youtu.be/2Us68hOnQ54?si=6GGI4QqxsD_0tPK7

Tema 8: Sistemes de gestió d'informació



1. Sistemes de gestió d'informació

2. ERP

- 2.1. Característiques dels ERP
- 2.2. Tipus de ERP
- 2.3. Els ERP més populars

3. CRM

- 3.1. CRM més populars

4. Business Intelligence (BI)

- 4.1. BI més populars

1. Sistemes de gestió d'informació

Els sistemes de gestió de la informació són conjunts de processos dissenyats per a capturar, emmagatzemar, organitzar i distribuir informació de manera eficient i efectiva en una organització.

Aquests sistemes són essencials per a ajudar a les empreses i altres entitats a gestionar les seues dades i garantir que la informació estiga disponible quan calga.

Ací tens alguns components clau dels sistemes de gestió de la informació:

Captura de dades:

Els sistemes de gestió de la informació solen començar amb la captura de dades des de diverses fonts. Això pot incloure dades ingressades manualment, dades generades automàticament per sensors o sistemes de seguiment, o dades importades de fonts externes.

Emmagatzematge de dades:

La informació recopilada serà guarda en bases de dades o altres sistemes d'emmagatzematge. L'elecció de la tecnologia d'emmagatzematge depèn de la naturalesa de les dades i les necessitats de l'organització.

Organització i classificació:

Les dades s'organitzen i classifiquen de manera que siguen fàcils de buscar i recuperar quan siga necessari. Això implica la creació d'estructures de dades coherents i l'ús de metadades per descriure les dades.

Accés i recuperació d'informació:

Els usuaris autoritzats poden accedir i recuperar informació de manera eficient. Això s'aconsegueix a través d'interfícies d'usuari, consultes de bases de dades i sistemes de cerca.

Seguretat de la informació:

La seguretat de les dades és essencial. Els sistemes de gestió de la informació han d'implementar mesures de seguretat per a protegir la informació confidencial i assegurar que només les persones autoritzades tinguin accés. També cal parar atenció a la integritat i la disponibilitat d'aquestes dades.

Integració de sistemes:

En moltes organitzacions, les dades provenen d'una varietat de fonts i sistemes. Els sistemes de gestió de la informació sovint s'integren amb altres sistemes de l'empresa per a garantir una gestió coherent de la informació.

Anàlisi de dades:

La capacitat d'analitzar dades és un component important dels sistemes de gestió de la informació. Això pot incloure la generació d'informes, anàlisis estadístics i la identificació de tendències.

Flux de treball i automatització:

Els sistemes de gestió de la informació poden automatitzar processos i fluxes de treball per a millorar l'eficiència i la productivitat.

Compliment normatiu:

Moltes organitzacions han de complir amb regulacions i normatives específiques en allò que respecta a la gestió de la informació. Els sistemes de gestió de la informació poden ajudar a garantir el compliment d'aquestes normatives.

Còpia de seguretat i recuperació de dades:

És important tenir sistemes de còpia de seguretat i recuperació de dades en cas de pèrdua d'informació a causa de fallades tècniques, errors humans o desastres naturals.

Els sistemes de gestió de la informació tenen un paper fonamental en la presa de decisions, la millora de l'eficiència operativa i el compliment de les responsabilitats legals i regulatòries en una organització. Poden variar en complexitat: des de sistemes senzills de gestió de documents fins a sistemes empresarials de gestió de contingut (ECM) que abasten una àmplia gamma de funcions.

2. ERP

Els ERP (Enterprise Resource Planning) són sistemes de planificació de recursos empresarials que permeten a les empreses gestionar eficaçment els seus recursos i processos interns. Aquests sistemes integren diferents funcions i departaments dins d'una empresa en una única plataforma de software, cosa que facilita la col·laboració i la presa de decisions informades.

Entre les principals característiques dels ERP hi ha la gestió de comptabilitat, finances, inventari, compres, vendes, recursos humans i altres àrees clau de l'empresa. Els ERP poden ser personalitzats per adaptar-se a les necessitats específiques de cada empresa i poden millorar l'eficiència, la productivitat i la visibilitat de les operacions.

També poden ajudar a automatitzar processos, simplificar la generació d'informes i facilitar la gestió global de la companyia.

2.1. Característiques dels ERP

Els sistemes ERP (Enterprise Resource Planning) tenen diverses característiques importants que els fan útils per a les empreses. Algunes de les seues característiques més destacades són:

Integració: El principal punt fort d'un ERP és la seua capacitat per a integrar diferents àrees i processos de l'empresa en una única plataforma. Això inclou funcions com comptabilitat, finances, gestió d'estocks, vendes, compres i recursos humans.

Base de dades centralitzada: Totes les dades i la informació es centralitzen en una sola base de dades, la qual cosa facilita l'accés i la gestió de la informació.

Automatització de processos: Els ERP permeten automatitzar molts processos empresarials, com la generació d'informes financers, la gestió d'estocks i la facturació, la qual cosa pot augmentar l'eficiència i reduir els errors humans.

Personalització: Molts sistemes ERP poden ser personalitzats per a satisfer les necessitats específiques de l'empresa. Això inclou l'adaptació de fluxos de treball, la incorporació de mòduls addicionals i la personalització d'informes.

Visibilitat i presa de decisions: Els ERP ofereixen una visió completa de l'estat de l'empresa en temps real, la qual cosa facilita la presa de decisions informades per part de la direcció i els responsables de l'empresa.

Seguretat de dades: Els ERP inclouen funcions de seguretat per a garantir que les dades empresarials siguen protegides i només accessibles per a usuaris autoritzats.

Flux de treball i col·laboració: Els ERP poden facilitar la gestió de fluxos de treball i la col·laboració entre diferents departaments i equips de treball.

Escalabilitat: Molts sistemes ERP són escalables, la qual cosa vol dir que poden créixer juntament amb l'empresa, adaptant-se a les necessitats canviants.

Compatibilitat amb la normativa (compliance): Els ERP poden ser configurats per a complir amb les normatives i regulacions específiques del sector o del país on opera l'empresa.

Suport i actualitzacions: Les empreses que proporcionen sistemes ERP solen oferir serveis de suport i actualitzacions per a mantenir el sistema actualitzat i en funcionament.

2.2. Tipus de ERP

Hi ha diversos tipus d'ERP (Enterprise Resource Planning) dissenyats per satisfer les necessitats de diferents tipus d'empreses i sectors. A continuació, es detallen alguns dels tipus més comuns d'ERP:

ERP per a PYMES: Aquest tipus d'ERP està dissenyat específicament per a xicotetes i mitjanes empreses. Són més assequibles i tenen funcions adaptades a les necessitats d'aquest tipus d'organitzacions.

ERP per a empreses grans: Aquestes solucions estan dissenyades per a empreses de gran envergadura amb una complexitat i volum de dades més elevats. Ofereixen funcions més avançades i poden ser personalitzades segons les necessitats particulars de l'empresa.

ERP específics per sectors: Hi ha ERP dissenyats per a sectors o indústries específiques, com ara la manufactura, l'atenció sanitària, l'educació, la logística, l'hoteleria, etc. Aquests sistemes tenen funcions i característiques adaptades a les necessitats particulars d'aquests sectors.

ERP basats en la núvol (Cloud ERP): Aquests sistemes s'ofereixen com a servei en línia i es poden accedir des de qualsevol lloc amb connexió a Internet. Són adequats per a empreses que volen evitar les despeses inicials de hardware i manteniment.

ERP de codi obert (Open Source ERP): Aquest tipus d'ERP està basat en codi obert, el que significa que les empreses poden personalitzar-lo lliurement. Solen ser més flexibles, però requereixen una major inversió en desenvolupament.

ERP de mòbil: Aquest tipus d'ERP permet als usuaris accedir i gestionar les dades i els processos empresarials mitjançant dispositius mòbils, com tauletes i telèfons intel·ligents.

ERP verticals i horitzontals: Els ERP verticals estan dissenyats per a un sector específic, mentre que els ERP horitzontals ofereixen funcions generals i poden ser personalitzats per a una àmplia gamma d'indústries.

ERP locals: Aquests sistemes s'instal·len i es gestionen internament a les instal·lacions de l'empresa. Són més controlats per l'empresa, però requereixen una inversió més gran en infraestructura i manteniment.

ERP global: Aquesta categoria d'ERP està dissenyada per a empreses amb operacions a escala mundial i pot gestionar múltiples ubicacions, idiomes i monedes.

ERP de lliure servei (Freemium ERP): Algunes empreses ofereixen versions gratuïtes o de cost reduït dels seus ERP amb funcionalitats bàsiques, però cobren per funcions més avançades o serveis addicionals.

Cada tipus d'ERP té avantatges i desavantatges i és important triar el que millor s'ajusti a les necessitats específiques de l'empresa.

2.3. Els ERP més populars

Els ERP més populars poden variar amb el temps i les tendències del mercat, però en general, alguns dels ERP més coneguts i utilitzats a nivell global són:

1. **SAP:** SAP és una de les empreses líders en solucions ERP a escala mundial. El seu software empresarial ofereix una àmplia gamma de funcions per a diferents sectors i empreses de tots els mides.
2. **Oracle NetSuite:** Oracle NetSuite és un sistema ERP de núvol que ofereix una suite completa de funcions empresarials, incloent comptabilitat, gestió de clients, inventari i màrqueting.
3. **Microsoft Dynamics 365:** Aquesta és la solució ERP de Microsoft que inclou diverses aplicacions empresarials, com Dynamics 365 Finance, Dynamics 365 Supply Chain Management i més.
4. **Infor CloudSuite:** Infor és coneguda pels seus productes ERP i altres aplicacions empresarials, i ofereix solucions especialitzades per a diversos sectors.
5. **Epicor ERP:** Epicor és un altre proveïdor de solucions ERP amb una presència global, especialitzat en la indústria manufacturera i altres sectors.

6. **Sage**: Sage ofereix una varietat de solucions ERP per a empreses xicotetes i mitjanes. Els seus productes inclouen Sage X3 i Sage 300.
7. **IFS Applications**: Aquest ERP està especialitzat en sectors com l'enginyeria, la construcció i la indústria aeroespacial.
8. **Acumatica**: Acumatica és una solució ERP de núvol que ofereix una àmplia gamma de funcions empresarials i és coneguda per la seva flexibilitat i personalització.
9. **SYSPRO**: SYSPRO és una altra opció popular en el món de l'ERP, especialitzada en la indústria manufacturera i distribució.
10. **Workday**: Workday és una solució ERP que es centra principalment en la gestió de recursos humans i finances, amb un enfocament fort en la núvol.

3. CRM

Els CRM, o Customer Relationship Management, són sistemes i estratègies dissenyats per gestionar les relacions amb els clients i les interaccions amb ells. Aquestes eines són molt utilitzades per les empreses per a millorar la interacció amb els clients, augmentar la retenció de clients i impulsar les vendes. Els CRM es basen en la recopilació i l'anàlisi de dades relacionades amb els clients per proporcionar una visió més completa de les seues necessitats i preferències. Algunes de les característiques importants dels CRM inclouen:

Gestió de dades de clients: Un CRM recopila i emmagatzema dades detallades de clients, com ara informació de contacte, històric de compres, comunicacions anteriors i preferències.

Automatització de vendes i màrqueting: Els CRM permeten automatitzar tasques com l'enviament de correus electrònics de seguiment, la generació de pressupostos i la gestió de campanyes de màrqueting.

Seguiment de les interaccions dels clients: Les empreses poden registrar totes les interaccions amb els clients, com ara cridades telefòniques, correus electrònics i reunions, per garantir una comunicació coherent i personalitzada.

Servei d'atenció al client: Els CRM inclouen eines per gestionar i fer seguiment dels casos d'atenció al client, garantint que les qüestions i problemes siguin resolts ràpidament.

Anàlisi i informes: Els CRM ofereixen funcions d'anàlisi de dades per avaluar el rendiment de les vendes i el màrqueting, identificar tendències i prendre decisions informades.

Accés des de múltiples dispositius: Molts CRM ofereixen accés a les dades del client des de dispositius mòbils i altres plataformes, permetent als treballadors estar connectats en qualsevol moment i lloc.

Personalització: Amb la informació recopilada, les empreses poden oferir comunicacions i ofertes més personalitzades als clients.

Integració amb altres sistemes: Molts CRM poden integrar-se amb altres aplicacions empresarials, com sistemes de correu electrònic i sistemes ERP, per millorar la coherència de les dades.

Gestió de leads: Els CRM ajuden a seguir i qualificar clients potencials (leads) per convertir-los en clients actius.

Escalabilitat: Els CRM poden adaptar-se a les necessitats en creixement de les empreses, ja siga per a xicotetes empreses o grans corporacions.

Els CRM són eines valuoses per a qualsevol empresa que vulga millorar la seva relació amb els clients i oferir un servei més eficient i personalitzat. Ajuden a gestionar i analitzar la informació dels clients per millorar la seua experiència i augmentar la retenció de clients.

3.1. CRM més populars

Els CRM (Customer Relationship Management) més populars varien amb el temps i les necessitats específiques de cada empresa, però hi ha diversos CRM que gaudeixen d'una gran popularitat i són àmpliament utilitzats a nivell mundial. Algunes de les opcions més conegudes inclouen:

Salesforce: Salesforce és un dels CRM més reconeguts i àmpliament utilitzats. Ofereix una ampla gamma de funcions de gestió de clients, vendes i màrqueting, i es pot personalitzar segons les necessitats de cada empresa.

HubSpot CRM: HubSpot és conegut pel seu CRM gratuït, que inclou funcions d'automatització de màrqueting i vendes. És popular, sobretot, entre les petites i mitjanes empreses.

Microsoft Dynamics 365: Aquest CRM de Microsoft ofereix una suite completa d'eines per gestionar relacions amb els clients, vendes i servei d'atenció al client.

Zoho CRM: Zoho CRM és una opció popular per a xicotetes i mitjanes empreses. Inclou funcions de seguiment de clients, automatització de màrqueting i eines d'anàlisi.

Freshsales: Aquest CRM està dissenyat per a la gestió de vendes i inclou funcions de seguiment de clients, automatització de vendes i informes detallats.

Pipedrive: Pipedrive és conegut per la seva facilitat d'ús i està dissenyat específicament per a la gestió de vendes i l'automatització de processos comercials.

Bitrix24: Bitrix24 ofereix una àmplia gamma d'eines empresarials, incloent un CRM complet amb funcions de vendes, màrqueting i atenció al client.

Insightly: Aquest CRM és popular entre xicotetes empreses i equips que necessiten funcions de gestió de clients i projectes.

Agile CRM: Agile CRM és una solució àmpliament utilitzada per a xicotetes i mitjanes empreses que ofereix funcions de màrqueting, vendes i atenció al client.

SugarCRM: SugarCRM és un CRM personalitzable que s'utilitza a una àmplia gamma d'indústries i sectors.

És important destacar que la elecció del CRM adequat dependrà de les necessitats i pressupostos específics de cada empresa. Cadascun d'aquests CRM té les seues pròpies característiques i avantatges, i la selecció depèn de les prioritats de l'empresa en la gestió de les relacions amb els clients.

4. Business Intelligence (BI)

La Intel·ligència de Negoci (Business Intelligence o BI, en anglès) és un conjunt de processos i eines que permeten recopilar, emmagatzemar, analitzar i presentar dades relacionades amb les operacions i l'activitat d'una empresa o organització. L'objectiu de la Intel·ligència de Negoci és facilitar la presa de decisions basades en evidències i millorar el rendiment empresarial mitjançant la comprensió de les dades.

Algunes de les característiques i funcions principals de la Intel·ligència de Negoci inclouen:

Recopilació de dades: La BI implica la recopilació de dades de múltiples fonts dins i fora de l'organització, incloent dades internes, dades de clients, dades de la competència i dades de mercat.

Emmagatzemament de dades: Les dades recopilades s'emmagatzemen en bases de dades i sistemes de gestió de dades, assegurant que estiguin disponibles per a l'anàlisi.

Anàlisi de dades: Les dades són sotmeses a diverses tècniques d'anàlisi, com ara l'exploració de dades, l'anàlisi estadístic i l'anàlisi predictiu, per identificar tendències, oportunitats i desafiaments.

Generació de informes i visualització de dades: La BI ofereix eines per a la creació de informes i visualitzacions de dades significatives, com gràfics i taules, per ajudar a les persones a entendre millor les dades.

Presa de decisions informades: La informació obtinguda a través de la BI permet a les empreses prendre decisions basades en dades i evidències, en lloc d'intuïció o suposicions.

Optimització d'operacions: La BI pot identificar àrees d'ineficiència o problemes en les operacions empresarials i proporcionar recomanacions per a la millora del rendiment.

Gestió de rendiment empresarial (BPM): La BI pot integrar-se amb la gestió de rendiment empresarial per a seguir i avaluar els indicadors clau de rendiment (KPI) i garantir l'objectiu dels objectius empresarials.

Predictió i pronòstic: Mitjançant l'anàlisi predictiu, la BI pot ajudar a preveure tendències futures i anticipar-se a canvis en el mercat.

Accés mòbil: Moltes solucions de BI ofereixen accés a les dades des de dispositius mòbils, permetent que els treballadors puguin accedir a la informació en qualsevol moment i lloc.

Seguretat de dades: És essencial garantir la seguretat de les dades i la privacitat en les solucions de BI, especialment quan es tracten dades sensibles de clients o de l'empresa.

La Intel·ligència de Negoci és una eina fonamental per a les empreses que volen comprendre millor el seu entorn empresarial, millorar les seves operacions i prendre decisions informades per aconseguir l'èxit.

4.1. BI més populars

Els sistemes de Business Intelligence (BI) més populars poden variar amb el temps i les tendències del mercat, però hi ha diverses opcions que són àmpliament utilitzades per empreses i organitzacions de tot el món. Algunes de les solucions de BI més populars inclouen:

Tableau: Tableau és un programa d'anàlisi i visualització de dades que permet als usuaris crear informes interactius i gràfics per entendre les dades de manera més eficient.

Microsoft Power BI: Power BI és una eina de Business Intelligence de Microsoft que ofereix funcions d'anàlisi de dades, visualització i informes. Es pot integrar amb altres productes de Microsoft, com Excel.

QlikView / Qlik Sense: QlikView i Qlik Sense són solucions de BI que proporcionen eines per explorar i visualitzar dades de manera interactiva.

IBM Cognos: IBM Cognos és una plataforma de Business Intelligence que ofereix eines d'anàlisi, informes i planificació empresarial.

SAS Business Intelligence: SAS Business Intelligence és part de la suite d'anàlisi de dades de SAS i inclou eines per a l'anàlisi de dades i la generació de raports.

MicroStrategy: MicroStrategy és una plataforma de Business Intelligence que proporciona una àmplia gamma d'eines d'anàlisi de dades i visualització.

SAP BusinessObjects: SAP BusinessObjects és una solució de BI que permet als usuaris accedir, analitzar i compartir informació empresarial.

Domo: Domo és una plataforma de Business Intelligence que ofereix eines de visualització de dades i una plataforma d'intel·ligència empresarial núvol.

Looker: Looker és una solució de BI que ofereix eines d'anàlisi de dades i generació de raports, especialment adequades per a les empreses centrades en dades.

Aquestes són algunes de les opcions més populars en el món del Business Intelligence, però de manera més dispersa i si ens centrem en el núvol públic, per exemple Azure:

Azure ofereix diverses eines i serveis de Business Intelligence (BI) per a analitzar dades, crear informes i visualitzacions, i prendre millors decisions de negoci. Ací tens les principals:

- 1- Power BI (Power BI Service i Power BI Embedded)
 - Servei SaaS per visualització i anàlisi de dades.

- Es pot connectar fàcilment a dades a Azure (SQL Database, Synapse, Data Lake, etc.).
- Power BI Embedded permet integrar quadres de comandament i informes dins d'aplicacions.

2- Azure Synapse Analytics (antic SQL Data Warehouse)

- Plataforma d'analítica massiva (data warehouse a escala empresarial).
- Permet combinar dades estructurades i no estructurades.
- Ideal per a integració amb Power BI per crear informes avançats.

3- Azure Analysis Services

- Motor OLAP (Online Analytical Processing).
- Permet construir models semàntics complexos i reutilitzables per BI.
- Integra molt bé amb Power BI i Excel.

4- Azure Data Lake Storage

- Emmagatzematge escalable per dades estructurades i no estructurades.
- Normalment es combina amb Synapse i Databricks per preparar dades per BI.

5- Azure Data Factory

- ETL (Extract, Transform, Load) al núvol.
- Permet moure i transformar dades des de diverses fonts fins a plataformes d'anàlisi.

6- Azure Databricks

- Entorn de big data i Machine Learning.
- Es pot usar per preparar dades i després visualitzar-les amb Power BI o Synapse.

7- Azure Stream Analytics

- Permet analitzar dades en temps real (IoT, logs, transaccions).
 - Es pot connectar directament a Power BI per dashboards en temps real.
-