Display, width i height

Introducció

Hem vist com hi ha principalment 2 tipus d'elements HTML: elements **block** i **inline**. També hem esmentat algunes alternatives, com ara **list-item** o **table-cell**.

La propietat `display` permet canviar el *tipus* de l'element HTML. De manera predeterminada, un element `<div>` (un element **de tipus block**) tindrà un valor de `display` predeterminat de `block`, però es pot representar com a **inline**:

```
div { display: inline; }
```

Per què canviar el display d'un element?

Perquè s'han de triar els elements HTML pel seu **significat**, no per la seva representació. Amb `display` podem alterar el **tipus** d'un element *sense* canviar-ne el **significat**.

Per exemple, un menú de navegació és una llista d'enllaços, és a dir, amb etiquetes `<li>`, però potser els volem visualitzar horitzontalment, o potser volem que uns enllaços ocupin mes espai que els hi correspondria per tractar-se d'elements en línia.

Valors de display

Cada opció de "visualització" té comportaments de representació específics:

- `block` ocuparà tota l'amplada disponible
- `inline` actuarà com a text senzill
- `inline-block` és, com el seu nom indica, un compost de bloc i comportament en línia, una opció *"el millor dels dos mons"*
- `list-item` és similar a `block` ja que ocupa tota l'amplada disponible, però mostra una icona en forma de punt addicional
- `table`, `table-row` i `table-cell` tenen un comportament molt específic, encara que inesperat, que permet dissenys més interessants.

display: block

Això convertirà qualsevol element en un element **block**.

Aquesta tècnica s'utilitza sovint als **enllaços** per augmentar la seva zona de clic, que es pot comprovar fàcilment establint un color de fons.

```
<ul class="menu">
  <li>
    <a>Inici</a>
  </li>
  <li>
    <a>Contacte</a>
  </li>
  <li>
    <a>Quant a</a>
  </li>
</ul>
```

```
.menu a{ background-color: red; color: white; display: block; }
```

display: inline

Això converteix qualsevol element en elements **inline**, com si només fossin un simple **text**.

Sovint s'utilitza per crear **navegacions horitzontals**, on els **elements de llista** són útils semànticament però no visualment.

```
<nav>
    <ul>
        <li><a>Introducció</a></li>
        <li><a>Objectius</a></li>
    </ul>
</nav>
```

```
nav li { display: inline; }
```

display: inline-block & display: list-item

Amb el valor `inline-block` es mostra un element com un element en línia, però es poden aplicar valors d'alçada i amplada.

Amb `list-item` el mostra com si fos un element d'una llista, amb el seu punt i tot.

display: none

Si apliqueu `display: none;` a un element HTML, l'elimina de la vostra pàgina web, com si mai hagués existit al vostre codi.

```
.mort { display: none;}
```

visibility: hidden

La propietat CSS `visibility` és lleugerament semblant a `display`. Aplicar `visibility: hidden;` *amaga* un element de la vostra pàgina, però només el converteix en **invisible**: encara ocupa l'espai que se suposava.

```
.mort { visibility: hidden;}
```

width i height

Les dimensions (o alçada i amplada) d'un element són **dinàmiques**, ja que fluctuen per adaptar-se al contingut. D'alguna manera és possible establir dimensions **específiques**.

Amb `width` i `height` es *fixa* l'amplada i l'alçada d'un element:

```
div { width: 600px; height: 200px;}
```

I si sols s'especifica widht o height?

Per exemple:

```
div { width: 600px; }
```

El `div` ocuparà tota l'amplada disponible, però es mantindrà de 600 píxels d'amplada **en qualsevol situació**:

- si la finestra del navegador és menys ampla que 600 píxels, mostrarà una barra de desplaçament horitzontal
- si la finestra del navegador és més ampla que 600 px, es mantindrà 600 px d'ample i no ocuparà tot l'espai

Com que només hem establert l'amplada, el `div` continua sent fluida en **height**: l'alçada es converteix en la dimensió variable per adaptar-se al contingut del `<div>`.

I si el contingut no hi quep?

En establir les dimensions d'un element, es mantindrà fix sense importar la longitud del seu contingut.

Què passa si el contingut és més llarg del que pot contenir l'element? Per defecte el contingut es mostrarà igual, **desbordant**!

overflow

La propietat CSS `overflow` ens permet gestionar el cas que el contingut sigui més llarg que el seu contenidor.

El valor per defecte és `visible`: el contingut es mostrarà de totes maneres, perquè *"Per què voldríeu evitar que l'usuari llegeixi contingut si està present al codi?"*.

En aplicar `overflow: hidden;`, simplement *prohibeu* que es vegi qualsevol contingut desbordat.

Barres de desplaçament

Si voleu que el vostre contingut es desbordi, però encara voleu que sigui accessible, podeu decidir mostrar barres de desplaçament aplicant `overflow: scroll`.

Una millor opció és utilitzar `overflow: auto`, ja que les barres de desplaçament només apareixeran *si* el contingut està desbordant, però romandran ocultes fins aleshores.

Compte amb les dimensions fixes

Sovint es requereix l'aplicació de dimensions específiques perquè un disseny sembli visualment atractiu, però pot tenir conseqüències no desitjades. En aquest sentit:

- Assegureu-vos que el vostre contingut no es desbordi
- si ho fa, utilitzeu "overflow: hidden" o "overflow: auto" per evitar que el vostre disseny es trenqui

min-width i min-height

La propietat `min-width` defineix l'amplada mínima d'un element.

- Si el contingut és inferior a l'amplada mínima, s'aplicarà l'amplada mínima.
- Si el contingut és més gran que l'amplada mínima, la propietat `min-width` no té cap efecte.

Això evita que el valor de la propietat d'amplada sigui més petit que l'amplada mínima, util per exemple quan el `width` es posa en percentatge però es vol evitar que sigui mes petit d'una certa quantitat de píxels.

La propietat `min-height` té un comportament similar.

max-width i max-height

La propietat `max-height` defineix l'alçada màxima d'un element.

- Si el contingut és més gran que l'alçada màxima, es desbordarà. La propietat `overflow` defineix com es gestionarà el contingut desbordat.
- Si el contingut és més petit que l'alçada màxima, la propietat `max-height` no té cap efecte.

Igual que amb el `min-height`, es fa servir quan el `height` s'especifica amb percentatge i es vol evitar que sigui excesivament gran.

La propietat `max-width` té un comportament simular.