UD03 - Posicionament i Ancoratge

Índex

Posicionament i ancoratge

Tipus de posicionament

```
(Propietat `position`)
```

Ancoratge dels elements

```
(Propietat `display`)
```

Model de Capes (stacking context)

Model de Caixa (Box Model)

Font: https://www.w3schools.com/css/css_boxmodel.asp

Propietat `box-sizing`

Estableix què mesuren les propietats `height`, `width` i derivades (`min-height`, `max-height`, ...).

Sense comptar `initial` (que en aquest cas equivaldria a `content-box` i `inherit`, la propietat `box-sizing` pot agafar dos valors:

- content-box.

- border-box.

Així, quan està establerta a `content-box` (per defecte), `height`, `width`, etc... defineixen el tamany del contingut. **Per això, si afegim voreres, marges o, fins i tot *padding*, l'element ens ocuparà més espaï del que hem especificat**.

En canvi, si l'establim a `border-box` aquesta mesura inclourà també el *padding* i les vores (*border*), fent molt més senzilla la tasca de sumar els espaïs que ens ocupa cada element a l'hora de maquetar.

> ⚠ **Avís:** Així i tot, encara haurem de tenir en compte els marges, si en tenim.

Establir el box-sizing de forma global (WIP)

Si ho preferim, podem establir el `box-sizing` a `border-box` de forma global amb una senzilla regla:

```
*,
::before,
::after {
  box-sizing: border-box;
}
```

> :bomb: Però si estam treballant en un projecte preexistent, podria passar que tinguéssim codi css anterior que, al estar calculat segons el comportament per defecte, deixas de funcionar corrèctament.
>
> El mateix ens pot passar si fem servir frameworks css externs que hagin estat dissenyats en base a `border-sizing: content-box`. Si be molts, com *Bootstrap* ja fan servir `border-box` i/o fixen explícitament aquesta propietat per al seu css.

En qualsevol cas, el següent exemple ens mostra una forma més refinada de fer-ho:

```
:root {
  box-sizing: border-box;
}


*,
::before,
::after {
  box-sizing: inherit;
}
```

Establint `box-sizing` a `inherit` obrim la possibilitat de canviar aquesta propietat recursivament de forma selectiva a les parts que ens interessi del DOM (per exemple `.oldCss {box-sizing: content-box;}`) mentre que la regla que hem definit sobre el selector `:root` ens garanteix que la resta heredarà el valor `border-box`.

Marges

Marges negatius

Marges colapsats

Casos en que els marges no colapsen:

- Establint la propietat `overflow` a qualsevol valor distint de `visible` a un contenidor s'evita que els marges interiors colapsin amb els exteriors.
- Si hi ha una vorera (border) entre dos marges (per exemple transparent, si la volem fer servir com a truc per evitar el colapsament).
- Cap a fora d'un contenidor dels següents contenidors:
  - Flotant ("floated")
  - Amb la propietat `display` a `inline-block`.
  - Amb posicionament absolut (`position: absolute`) o fix (`position: fixed`)
- A dins d'un contenidor flexbox (`display: flex`) o grid (`display: grid`).
- Els elements amb `display: table-cell`, `table-row` i la majoria de la resta de tipus de display de taules a excepció de `table`, `table-inline` i `table-caption` no tenen marges, així que no poden colapsar.

Voreres

Javascript - Components CSS

CSS Modular

Components Javascript/jQuery + CSS