

CSS



CSS. Conceptos. Selectores



Origen y evolución de CSS

- 1994 -> Se funda W3C
- 1996 -> Publica la especificación CSS1 [HTML3]
- 1998 -> Nueva especificación CSS2
- 2011 -> CSS3 [HTML5 = 2014]
 - Módulos, niveles y estados.

https://www.w3.org/standards/techs/css#w3c_all

Propiedades de fabricantes. Prefijos

Fuentes del CSS

- estilos en línea. atributo style
Propiedades CSS. Ejemplos de propiedades.
 - font-size
 - color
 - background-color
- estilos independientes
Selectores CSS. Selector de etiqueta
 - embebidos: etiqueta style
 - **ficheros css**

Proyecto. Creación del fichero CSS

Ejemplo de propiedad: `body {background-color}`

Selectores, propiedades y directivas.

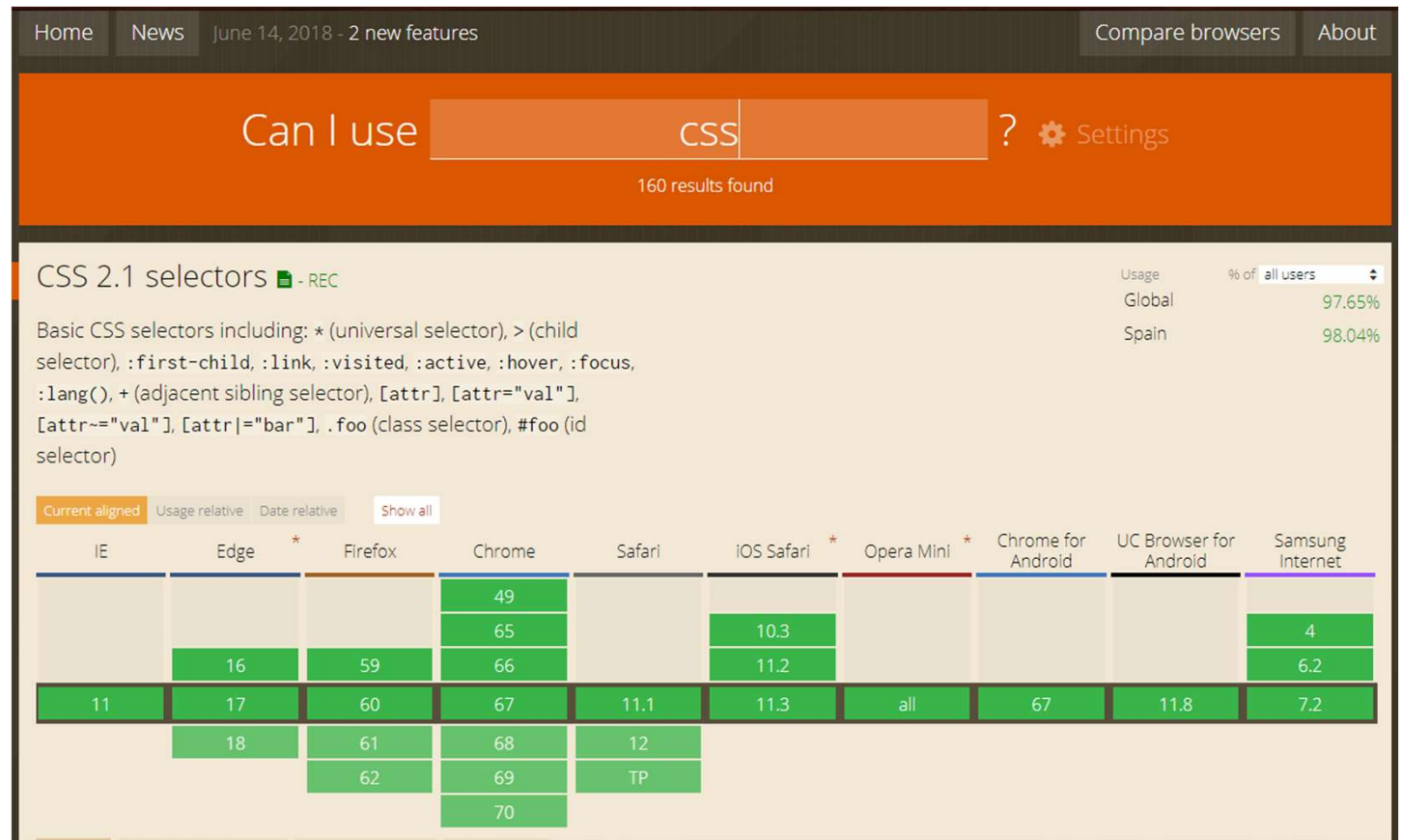
- Selectores y declaraciones: propiedades



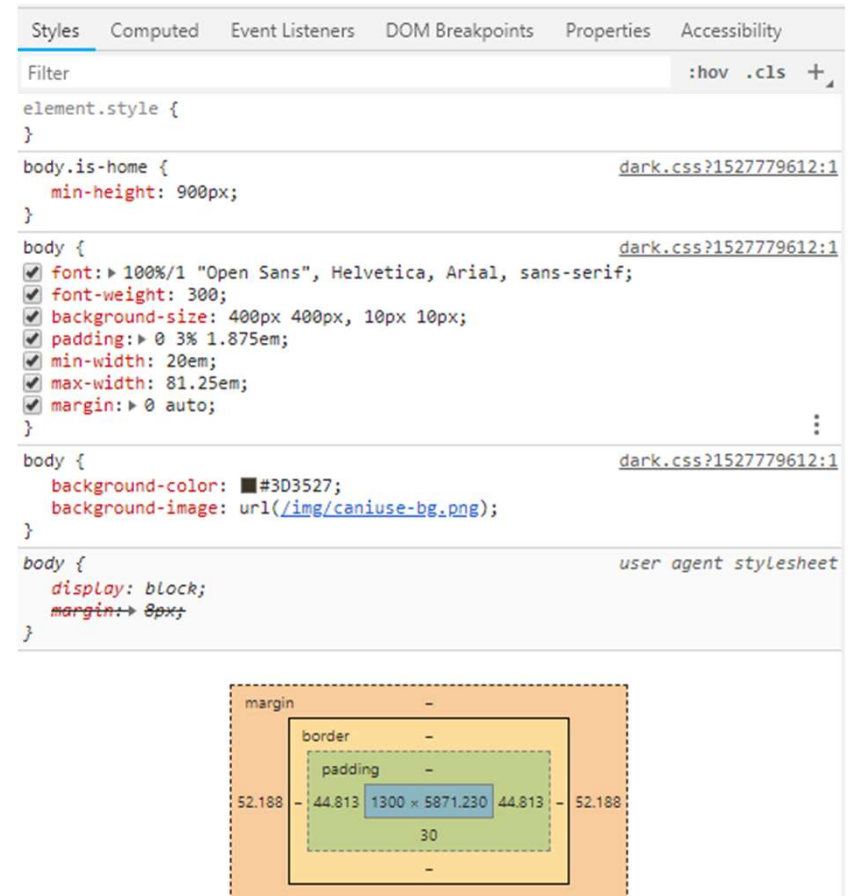
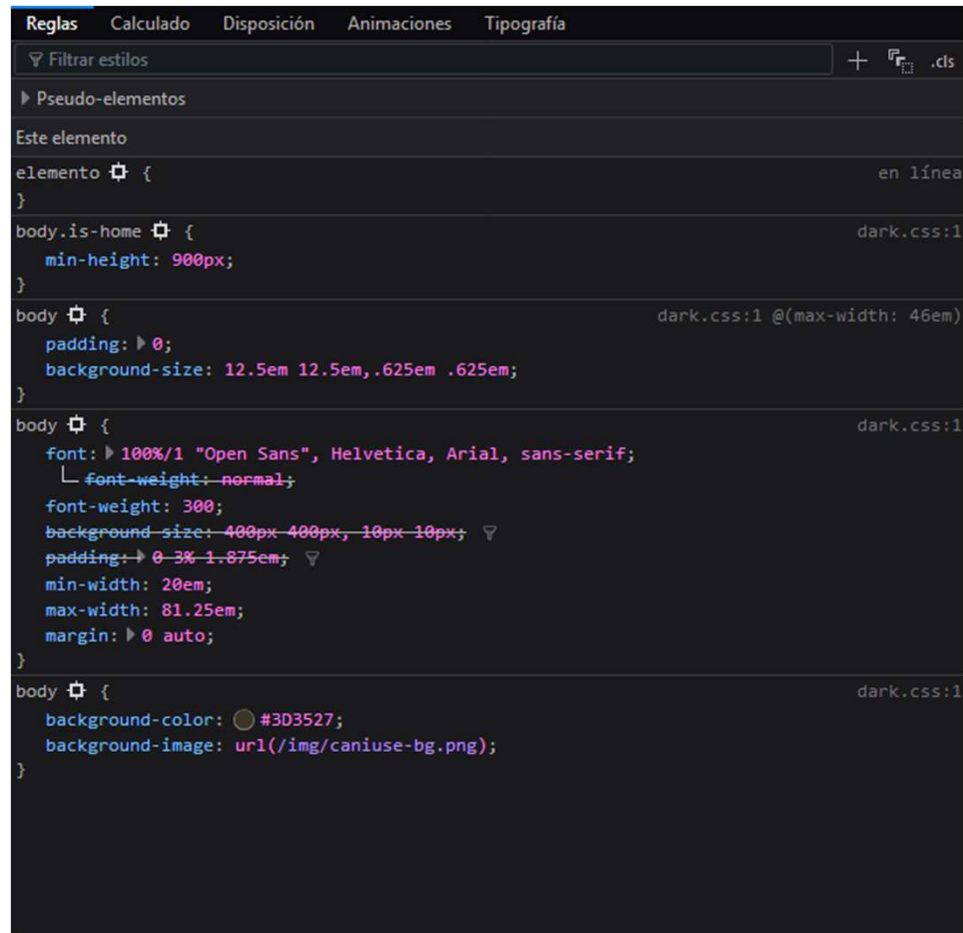
- Directivas
 - @import
 - @media
 - @font-face

Herramientas. CanIUse

<https://caniuse.com/>



Herramientas en el navegador. Inspector



Selectores básicos

- selector de etiquetas (tags) p {...}
- selector de id #id {...}
- **selector de clases .clase {}**
 - Sistemas de nomenclatura
BEM (Block, Element, Modifier) <http://getbem.com/>
- selectores múltiples: h1, h2, h3 {...}
- selector de atributo: [name] {...}

[atributo = valor] elementos con ese valor

[atributo ^= valor] elementos que comienzan con el valor provisto

[atributo \$= valor] elementos que terminan con ese valor

[atributo *= valor] elementos que contienen el texto provisto

Selectores combinados: anidación

Selector “espacio” (Descendant Selector)

div p {...}

Selector > (Child Selector)

div>p {...}

Selector + (Adjacent Sibling Selector)

div+p {...}

Selector ~ (General Sibling Selector)

div~p {...}

```
<p></p>
<div>
    <p><p>
    <address>
        <p></p>
    </address>
</div>
<p></p>
<p></p>
```

Proyecto. Uso de clases CSS o de anidación

h1 en el header principal: 1.25rem

h1 en el header de sección: 5rem

h1 en el header de article: 3.5rem

Pseudoelementos y pseudoclases.

Pseudoelementos: elementos definidos desde CSS.

::first-letter

::first-line

::before {content=""}

::after {content=""}

Pseudoclases: clases aplicadas dinámicamente a elementos reales de HTML en función de su estado

:link, :visited, :active, :focus :hover, :target, :lang()

:nth-child(n), :first-child, last-child, only-child

:nth-of-type(n), :first-of-type, :last-of-type, only-of-type

:not()

Proyecto. PseudoElementos

Ejemplo de `::after` / `::before` para añadir al `blockquote` comillas (open-quote, close-quote).

Le añadimos también `title`

Aplicación de múltiples estilos

- Herencia
 - Propiedades que se heredan en los elementos hijos
font-family, color... -> afectan al contenido
 - Propiedades que no se heredan
margin, padding, border... -> afectan al contenedor
- Cascada
 - agente de usuario: por defecto / definidos por el usuario
 - bloques de CSS: externos / embebidos
 - estilos en línea
- Especificidad

Especificidad

a | b | c | d

- a** es igual a 1 si la declaración está definida como estilos en línea
- b** es igual al número de id's
- c** es igual al número de otros atributos y pseudo-clases
- d** es igual al número de elementos y pseudo-elementos

Unidades de medida. Relativas y absolutas.

Los EM y los REM.

- EM
 - tamaño de fuente del elemento (distancia entre las líneas base; por defecto 16px)
 - varía cuando se modifica el tamaño de fuente
- REM
 - tamaño de fuente en el root: <html> (distancia entre las líneas base; por defecto 16px)
 - constante una vez definido en una página

VW (viewport width) / VH (viewport height)

Variables. Uso de calc()

Variables definidas globalmente

```
:root {  
  --color-principal: #06c;  
}
```

Se utilizan mediante la función var()

```
#foo h1 {  
  color: var(--color-principal);  
}
```

La función cal() opera con variables y números

```
{  
  --separacion: 20;  
  margin-top: calc(var(--separacion) * 2px);  
}
```

Proyecto. Uso de variables

Definición de los colores:

--color-texto

--color-fondo

CSS. Propiedades básicas



Propiedades básicas

- tipografías. Fuentes. Google fonts. Fuentes "locales"
- colores: transparencia
- degradados
- sombras
- backgrounds: imágenes y patrones

Tipografías

- Propiedades de las fuentes.
 - font-size, font-family, font-weight, font-style
 - text-decoration, text-transform
 - line-height, letter-spacing, word-spacing
 - text-align, vertical-align, text-indent
- Google fonts.
- Fuentes "locales". @font-face
- Fuentes de iconos. FontAwesome

Proyecto. Uso de Google Fonts. Uso de Font Ions

Lora:400,700,400italic,700italic -> body

Open+Sans:300italic,400italic,600italic,700italic

0italic,800italic,400,300,600,700,800 ->

h1..h6

FontAwesome

Colores

- Sistemas de colores
 - Nombres
 - `rgb()` - red, green, blue
 - hexadecimal (#)
 - `hsl()` - hue° (matiz), saturation%, luminosity%
- Transparencia

canal alfa entre 0 (transparente) y 1 (opaco)

 - `rgba()`
 - `hsla()`
 - Opacity / se hereda

Degradados y sombras

- Degradados (gradient)
 - Lineales
background: linear-gradient(to top, #573636, #f00);
 - Radiales
background: radial-gradient(circle, #573636, #f00);
- Sombras (shadow)
 - De cajas
box-shadow: 1px 1px 1px 1px grey;
 - De textos
text-shadow: 1px 1px 1px grey

Backgrounds: imágenes y patrones

- `background-image: url(./assets/about-bg.jpg);`
- `background-repeat: no-repeat;`
- `background-size: cover;`
- `background-position: center center;`
- `background-attachment: fixed;`
- `background-clip: border-box;`
- `background-blend-mode: normal;`

Proyecto. Imágenes de fondo

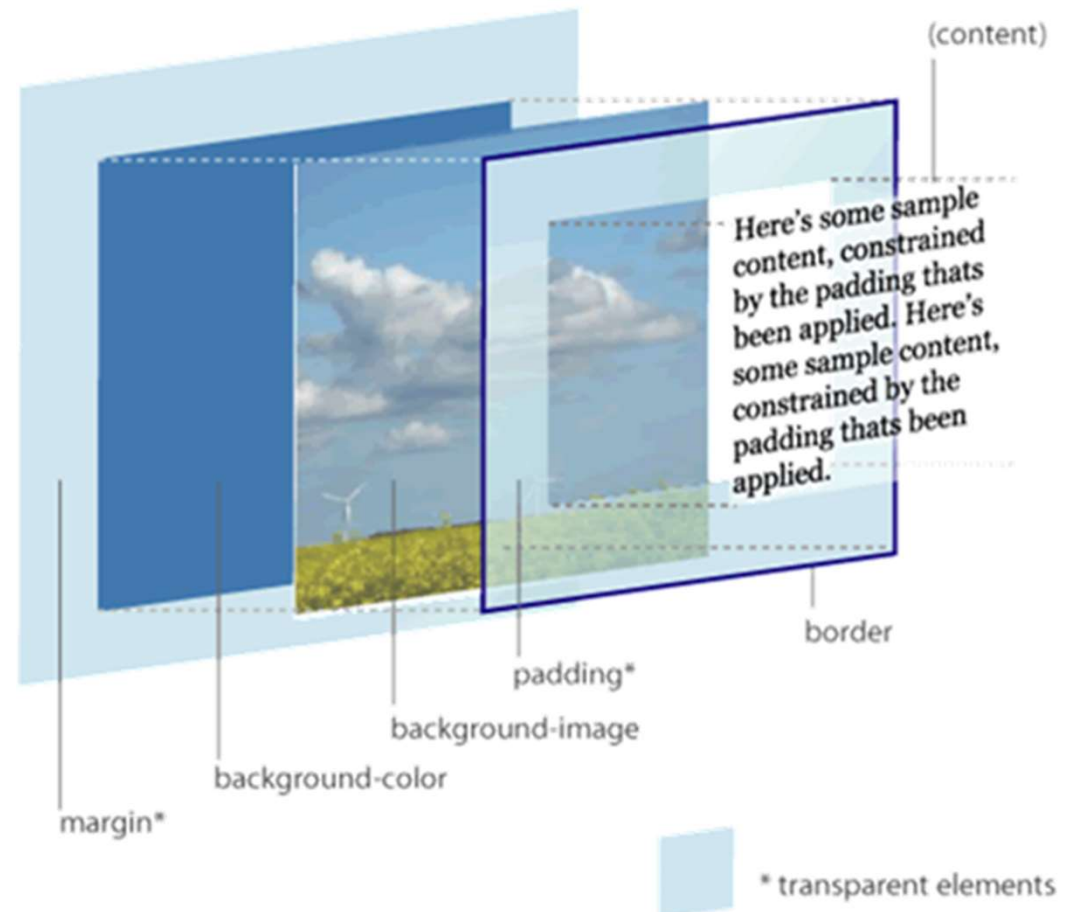
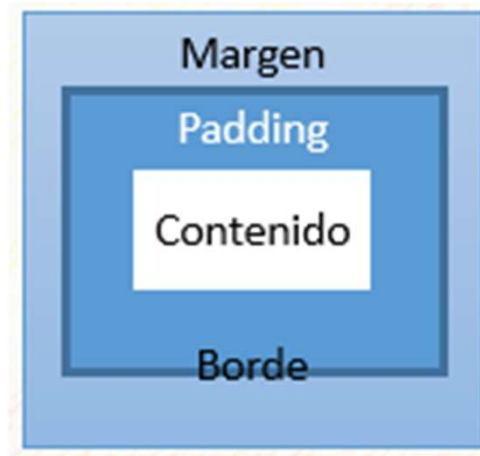
cada sección header con una altura
(100vh)
y un background-image de tamaño cover

CSS. Modelo de caja



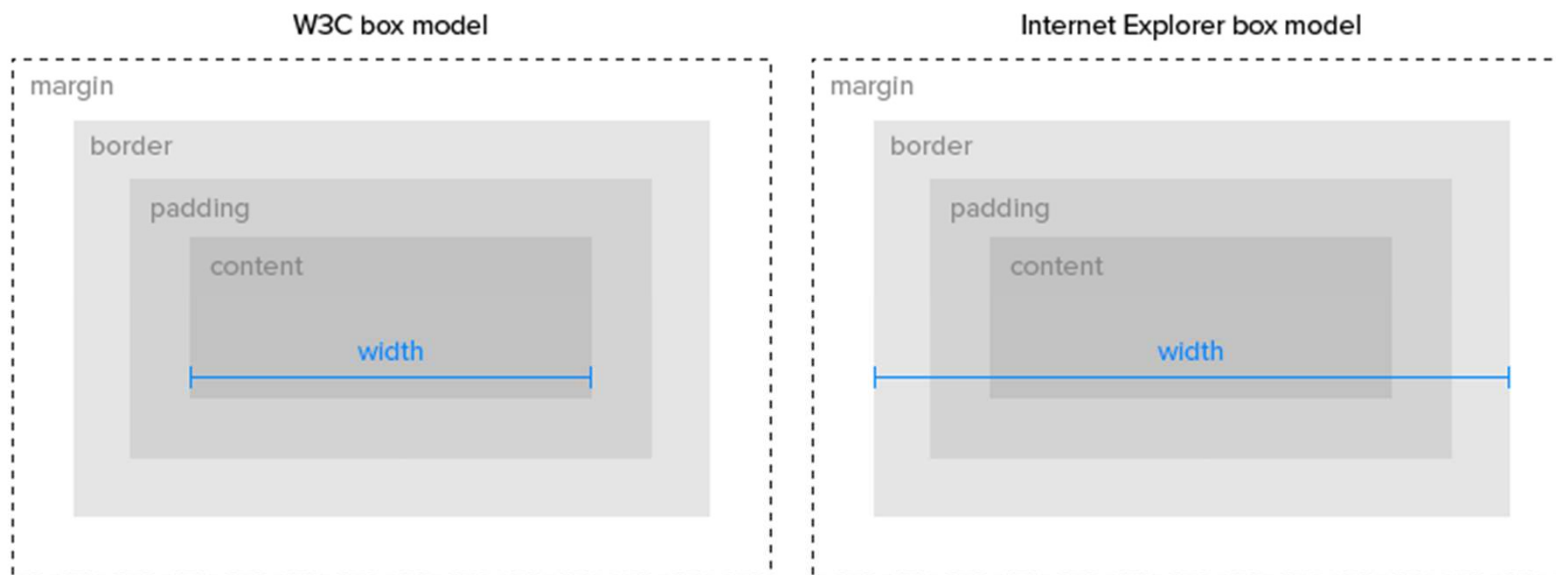
Modelo de caja. Elementos

- Contenido
- Aire (padding)
- Borde
- Fondo (background)
- Margen



Modelo de caja. Sistemas de medida

- box-sizing
 - content-box
 - border-box



Proyecto. Uso de Reset

Reset parcial: box-sizing

Modelo de caja. Propiedades

- Propiedades
 - margin
 - padding
 - border
 - border-radius
 - border-image
- Usos
 - wrappers
 - centrado de bloques

Centrado de bloques

Horizontal:

width: %

margin-left / margin-right: auto

Vertical

- line-height = height (elementos de 1 linea)
- display table
- position absolute 50% / 50% margin -w/2 - h/2
- position absolute 50% / 50% transform translate -50% -50%

Alternativa -> display
flex

Proyecto. Uso del espacio

Márgenes y paddings.
content-wrapper -< centrado horizontal

CSS. Display y posicionamiento



Display. Opciones básicas

- Block
- Inline
- Inline-Block
- None.

Etiqueta Visibility

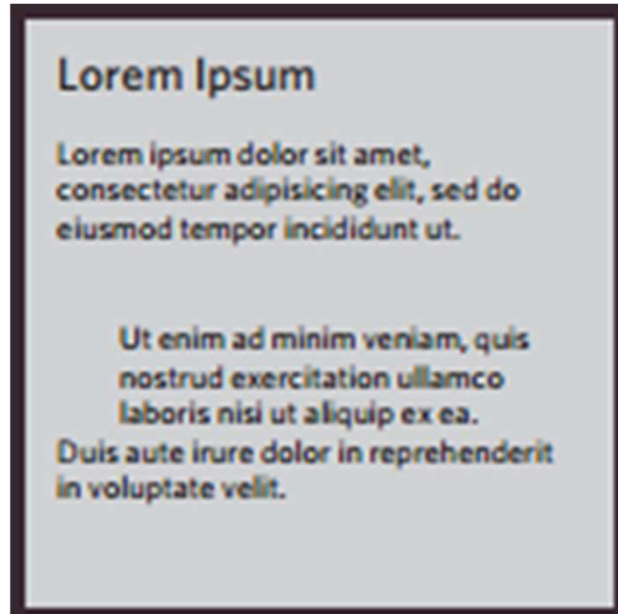
un elemento en bloque fluye como si fuera en línea, pero conserva el resto de sus propiedades en bloque

Posicionamiento.

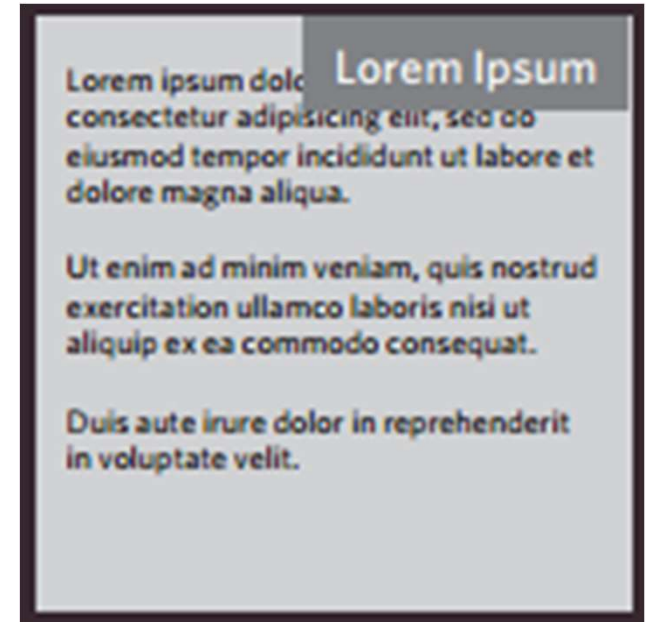
- Position:
 - static
 - relative
 - absolute
 - fixed



pasan a otra capa
pueden
superponerse
les afecta z-index



Position: relative
Párrafo desplazado al
darle un valor a top y
left



Encabezado desplazado al
darle un valor a top y left en
términos absolutos. El primer
párrafo se coloca como si el
encabezado no existiera.

Proyecto. Creación del menú y las cabeceras

Nav: inline block

Nav y header > posición absoluta / fixed

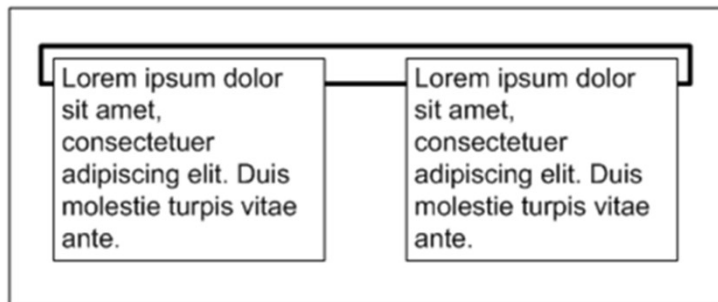
Section header: title-wrapper: centrado horizontal / vertical

sección header : añadiendo capa de opacidad

Cambios del flujo static

- Float: left / right
- Clear: left / right /both

“Cuando un elemento tiene a todos sus hijos flotando, pierde su altura”



OVERFLOW : hidden / auto

CLEARFIX AFTER.

```
::after {  
    content : "  
    display:  
block;  
clear: both;  
}
```

Columnas de texto

column-count: 2;
column-width: 50% (alternativa)
column-gap: 1rem;
column-rule: 1px solid black;

column-fill: balance (default) / auto;
column-span: none (default) / all;

break-inside: auto (default) / avoid;

The Final Frontier

There can be no thought of finishing for 'aiming for the stars.' Both figuratively and literally, it is a task to occupy the generations. And no matter how much progress one makes, there is always the thrill of just beginning.

There can be no thought of finishing for 'aiming for the stars.' Both figuratively and literally, it is a task to occupy the generations. And no matter how much progress one makes, there is always the thrill of just beginning.

« The dreams of yesterday are the hopes of today and the reality of tomorrow. Science has not yet mastered prophecy. We predict too much for the next year and yet far too little for the next ten. »

Spaceflights cannot be stopped. This is not the work of any one man or even a group of men. It is a historical process which mankind is carrying out in accordance with the natural laws of human development.

Proyecto. Cambios en el post

Float para posicionar una imagen

Float para crear columnas // problema de colapso

Uso de la etiqueta column

Diseño de layouts



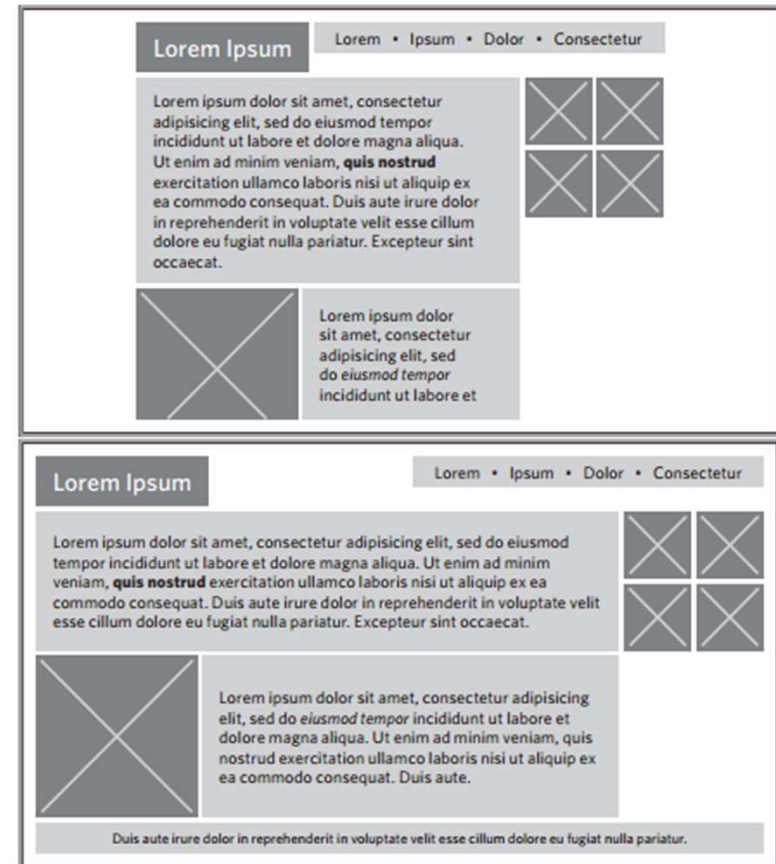
Diseño de layouts

- Maquetación con divs
 - divs semánticos
 - divs genéricos con clases
- Estilos
 - height / width
 - float / clear / overflow
 - position / z-index
 - display / visibility
 - margin / padding / background



Disposiciones: la anchura

- Disposición fija
(Fixed width layout)
- Disposición fluida o
“líquida”
(Liquid layout)
- Disposición elástica
(Elastic layout)
- Disposición híbrida
(Hybrid Layout)



El problema del “Holy Grail Layout”: la altura

```
<div class="grid-container">  
  <header>Header</header>  
  <aside class="sidebar-left">  
    Left Sidebar  
  </aside>  
  <main>Sections / Articles </main>  
  <aside class="sidebar-right">  
    Right Sidebar  
  </aside>  
  <footer>Footer</footer>  
</div>
```



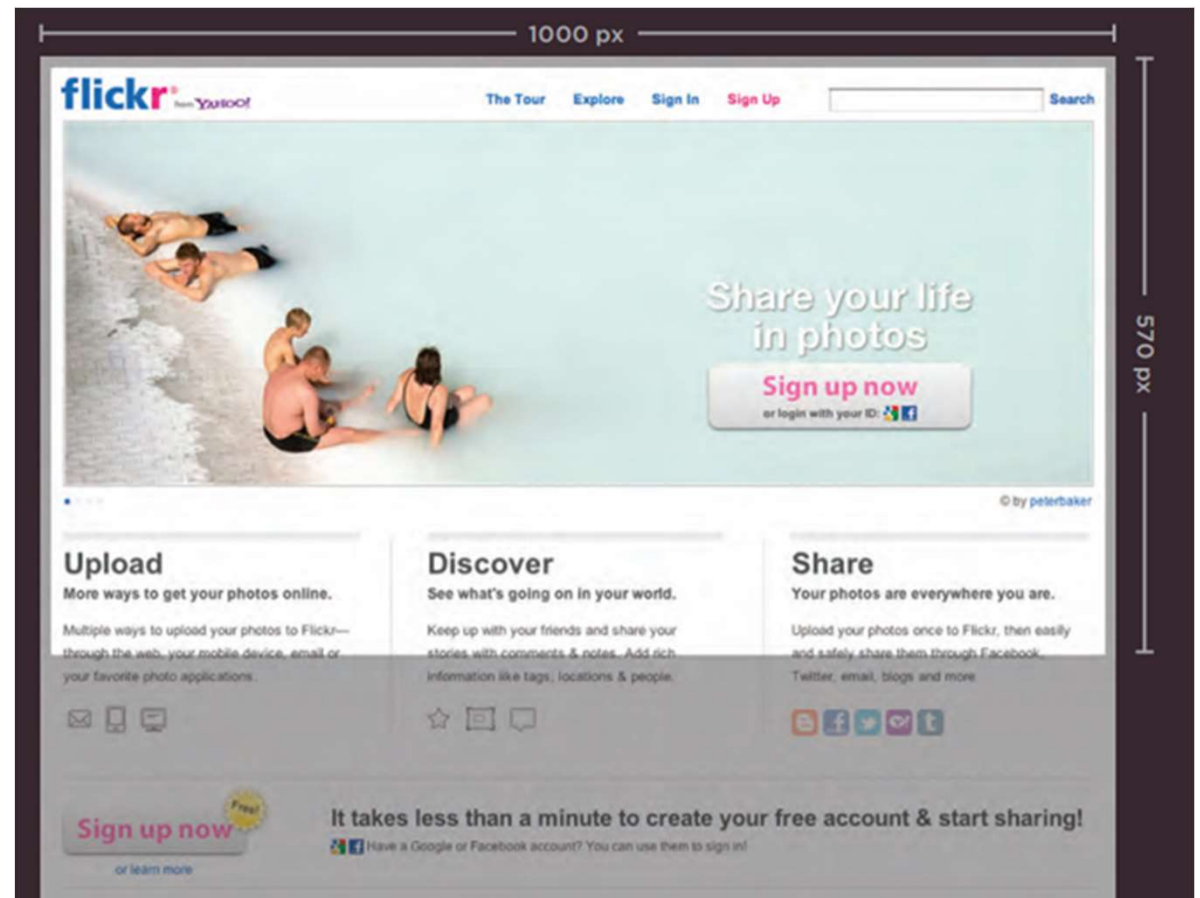
Ejercicio. Simulando el “Holy Grail Layout”

Uso de float para crearlo

Primer pantallazo (above the fold)

En sobremesa y portátiles suele considerarse

- de unos 570-600 px de altura y
- unos 960 px de anchura



Distribución y proporcionalidad: grid

Como ayuda para distribuir correctamente los elementos en alguna proporción se utiliza una rejilla (como una cuadrícula sólo en vertical)






Ejercicio. Creación de un grid

Librerías / Frameworks con grids

<https://getbootstrap.com/>

B

Home Documentation Examples Themes Jobs Expo Blog

v4.0   

Download

Bootstrap


Build responsive, mobile-first projects on the web with the world's most popular front-end component library.

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery.

Get started

Download

Currently v4.0.0



Display Table

- `display: table;`
- `display: table-cell;`
- `display: table-column;`
- `display: table-colgroup;`
- `display: table-header-group;`
- `display: table-row-group;`
- `display: table-footer-group;`
- `display: table-row;`
- `display: table-caption;`

COL1	COL2	COL3
<p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p> <p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p> <p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p>	<p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p> <p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p> <p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p>	<p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p> <p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p> <p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p>

Ejercicio. Consiguiendo el “Holy Grail Layout”

Uso de display table para crearlo

Flex



Flex Box

FlexBox: un módulo completo de layout- > defino como se muestran los elementos y cómo se relacionan con el resto.

CONTENEDOR / ÍTEMS El contenedor va a poder modificar las dimensiones y el orden de los ítems para acomodarlos según nuestras indicaciones.

¿Qué cosas podemos hacer con flexbox que antes no podíamos hacer de manera fácil?

- Alineación vertical
- Columnas de igual altura independientemente del contenido
- Cambiar el orden en el que se muestran los elementos sin que cambie el HTML

Contenedor Flex

display:flex

display:inline-flex

- define un “ contenedor flexible
- convierte de forma automática a sus “hijos” directos en “ elementos flexibles
- sus tamaños y disposición se ajustará siempre al espacio disponible en el contenedor



Contenedor Flex: propiedades

- **flex-direction:** column / row / reverse-column / reverse-row;
- **flex-wrap:** nowrap - wrap;
- **justify-content:** flex-start;
alineación en el eje principal
- **align-items:** stretch;
alineación en el eje secundario
- **align-content:** flex-start;
alineación de FILAS en el eje secundario

Items flexibles

- order
- flex
 - flex-grow: default 0
cuánto crecerá el elemento, si puede
 - flex-shrink: default 1
cuánto decrecerá el elemento, si lo necesita
 - flex-basis: default auto
tamaño inicial en el eje principal
- align-self

Ejercicio. Consiguiendo el “Holy Grail Layout”

Uso de display flex para crearlo

Proyecto. Utilizando FlexBox

Posts: colocando el botón
Footer: redes sociales

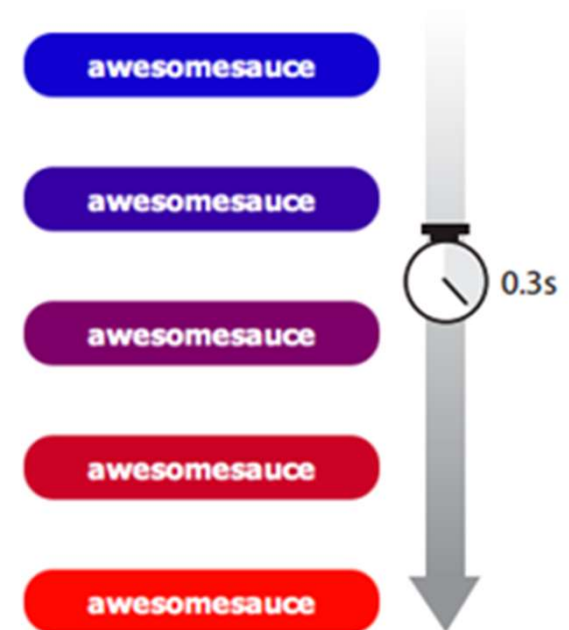
CSS Dinámico



CSS Dinámico

- Transiciones
- Transformaciones
- Animaciones

CSS3 incorpora la posibilidad de efectos animados que hasta el momento sólo eran posibles con Flash o JavaScript



Transiciones

permite suavizar los cambios (tweening), creando automáticamente el resto de los pasos que se encuentran implícitos en un cambio de estado (e.g. cambio de color, cambio de fuente...)

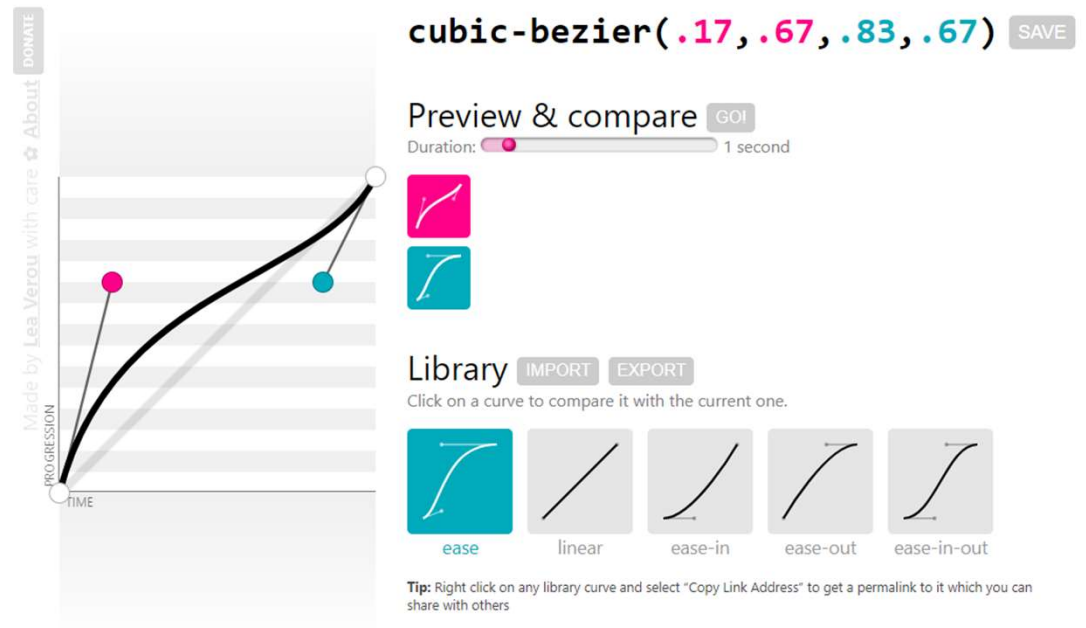
El proceso de transición suele aparecer ligado a una pseudoclase como :hover, :focus, or :active

```
a.smooth {...  
    transition----}  
a.smooth:hover,  
a.smooth:focus {  
    background-color: red;  
}
```

Transition: propiedades

- transition-property: all;
- transition-duration: 1s;
- transition-timing-function: linear;
- transition-delay: 1s;

<http://cubic-bezier.com>



Transformaciones

transform -> puede operar cuatro transformaciones básicas en un elemento

- escalar: `scaleX(x)` , `scaleY(y)`, `scale(x,y)`
- rotar `rotate(grados: deg)`

El centro de rotación se puede modificar con `transform-origin`:
`percentage` | `length` | `left` | `center` | `right` | `top` | `bottom`

- inclinar `skewX(deg)` `skewY(deg)` `skew(deg, [deg])`
- trasladar o mover `translateX(x)`, `translateY(y)`, `translate(x,y)`

Se pueden aplicar a la vez diversas transformaciones

Se pueden combinar con transiciones

Proyecto. Utilizando CSS Dinámico

Transiciones / Transformaciones en el menú

Animaciones: keyframes

@keyframes: Definimos la serie de etapas que constituyen la

```
@keyframes colors {  
  0% { background-color: red; }  
  20% { background-color:  
orange; }  
  40% { background-color:  
yellow; }  
  60% { background-color:  
green; }  
  80% { background-color:  
blue; }  
  100% { background-color:  
purple; }  
}
```



Animaciones: animation

- animation-name nombre de la secuencia de animación
- animation-duration tiempo en segundos
- animation-timing-function forma de aceleración
- animation-iteration-count número de veces que se repite
- animation-direction posible dirección adelante y/o atrás
- animation-play-state estado reproduciéndose o en pausa
- animation-delay tiempo hasta que se inicia
- animation-fill-mode que sucede cuando termina (límites en los que puede ser aplicada)

Proyecto. Utilizando CSS Dinámico

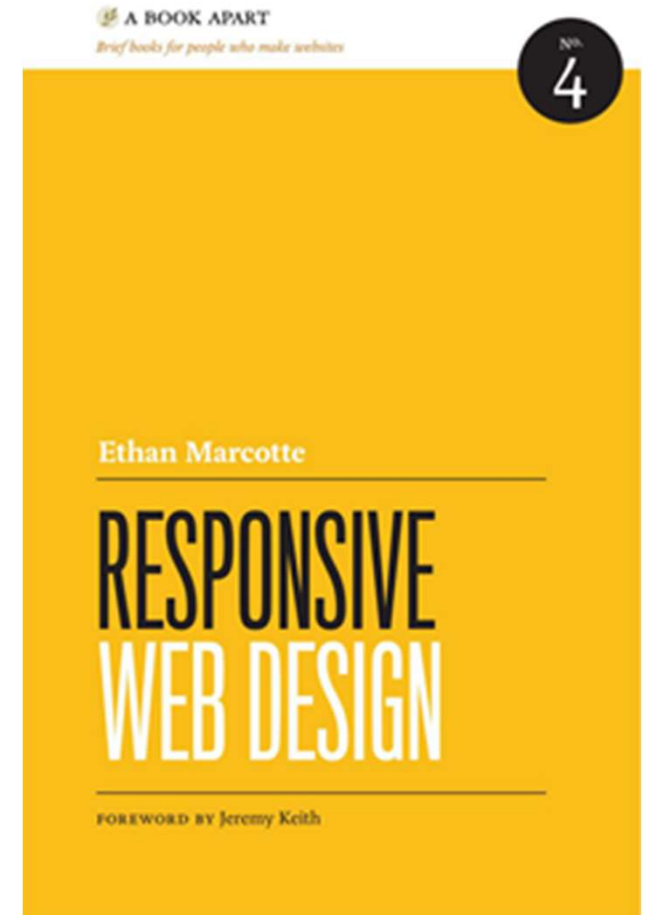
Animación en el título de la sección

RWD



Diseño Responsivo (RWD)

- ventanas de ancho inconsistente
 - resoluciones de pantalla
 - preferencias de los usuarios
 - fuentes instaladas
-
- Layouts fluidos
 - CSS3 Media Queries
 - Imágenes fluidas



Media Queries

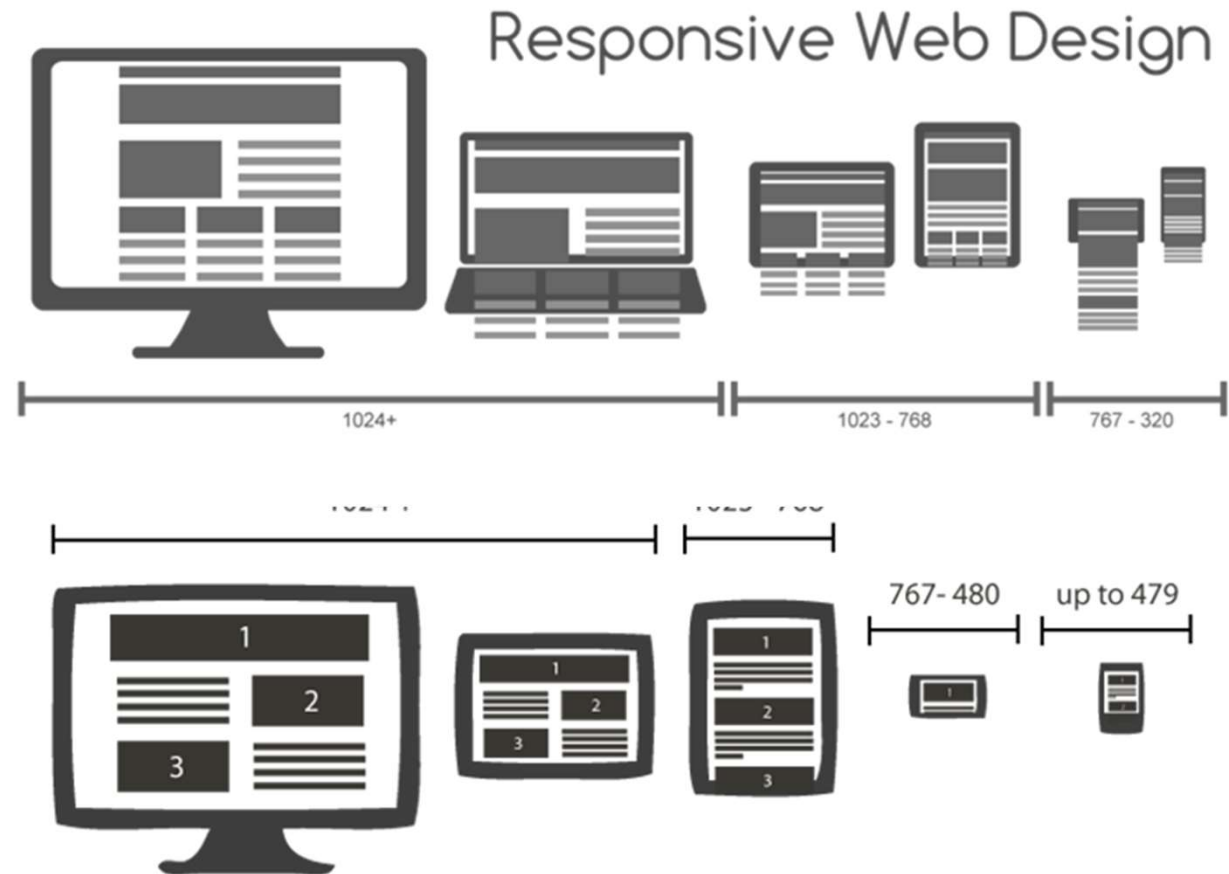
CSS2 -> media types: screen, print, voice...

CSS3 -> media queries:

media type + query del tamaño del dispositivo

HTML	{	<code><link rel="stylesheet" media="screen and (max-device-width: 480px)" href="screen.css" /></code>
L		<code>@import url("shetland.css") screen and (max-device-width: 480px);</code>
CSS	{	<code>@media screen and (max-device-width: 480px) { .column { float: none; } }</code>

Tamaños de pantalla



Procedimiento

Diseño
bottom-up
o mobile
first

escritorio / tablet horizontal: más de 1024px	
@media (min-width: 1024px)	Default
tablet vertical, mas de 768px (y menos de 1024px)	
@media (min-width: 768px)	@media (max-width: 1023px))
móvil horizontal, más de 480px (y menos de 768px)	
@media (min-width: 480px)	@media (max-width: 767px)

Diseño
top-
bottom

Viewport

Existe el metadata **viewport** en el head, solo válido para dispositivos móviles, que permite cambiar el valor por defecto y el comportamiento del navegador

```
<meta name="viewport"  
      content="width=device-width,  
              initial-scale=1">
```



zoom
min-zoom
max-zoom
user-zoom
orientation
initial-scale
maximum-scale

Patrones de navegación

[Brad Frost: Responsive Navigation Patterns](#)

- Top Nav or “Do Nothing” Approach
- The Footer Anchor
- The Select Menu
- The Toggle
- The Left Nav Flyout
- The Footer Only
- The “Hide and Cry”

[Brad Frost: Complex Responsive Navigation Patterns](#)

Imágenes

- Imágenes fluidas
contenedor (figure, div) -> width %
img -> width 100% / max-width 100%
- Distintas imágenes en distintos dispositivos

img srcset / sizes
picture source

<picture>

<source media="(min-width: 650px)"
srcset="img_pink_flowers.jpg">

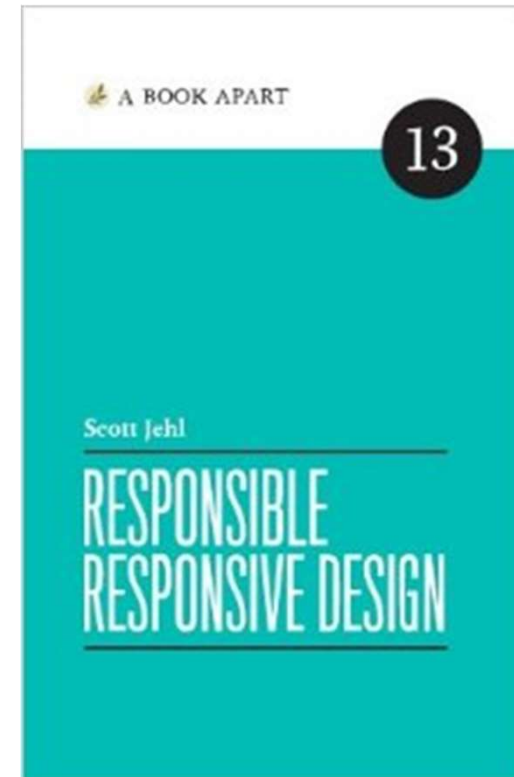
<source media="(min-width: 465px)"
srcset="img_white_flower.jpg">

</picture>

Diseño “Responsive” Responsable

El diseño debe adecuarse a las características de los dispositivos móviles más allá del simple tamaño de la pantalla.

- **Usabilidad**
- **Accesibilidad**
- **Sostenibilidad** (*Sustainability*)
- **Rendimiento** (*Performance:*)



Ejercicio. RWD

Ejemplo básico de RWD

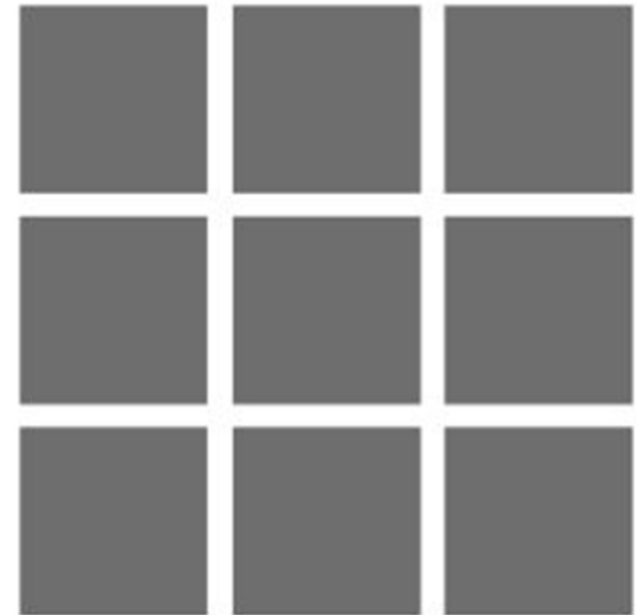
CSS Grid



Display Grid

Sistema de rejilla en 2 DIMENSIONES

- ▶ Voy a poder colocar los items DONDE QUIERA...
- ▶ ...PERO habrá ítems que SE COLOQUEN SOLOS (AUTO-PLACEMENT)
- ▶ Puedo HACER LO MISMO DE MUCHAS FORMAS...
- ▶ ...PERO NO TODAS HACEN LO MISMO.
- ▶ Controlo las 2 dimensiones, NO ES FLEXBOX.
- ▶ La colocación de los items es muy libre, NO ES UNA TABLA.
- ▶ Tiene una EXTENSA SINTAXIS.
- ▶ VA A CAMBIAR EL CSS PARA SIEMPRE.



Soporte

CSS Grid Layout - CR

Method of using a grid concept to lay out content, providing a mechanism for authors to divide available space for layout into columns and rows using a set of predictable sizing behaviors. Includes support for all `grid-*` properties and the `fr` unit.

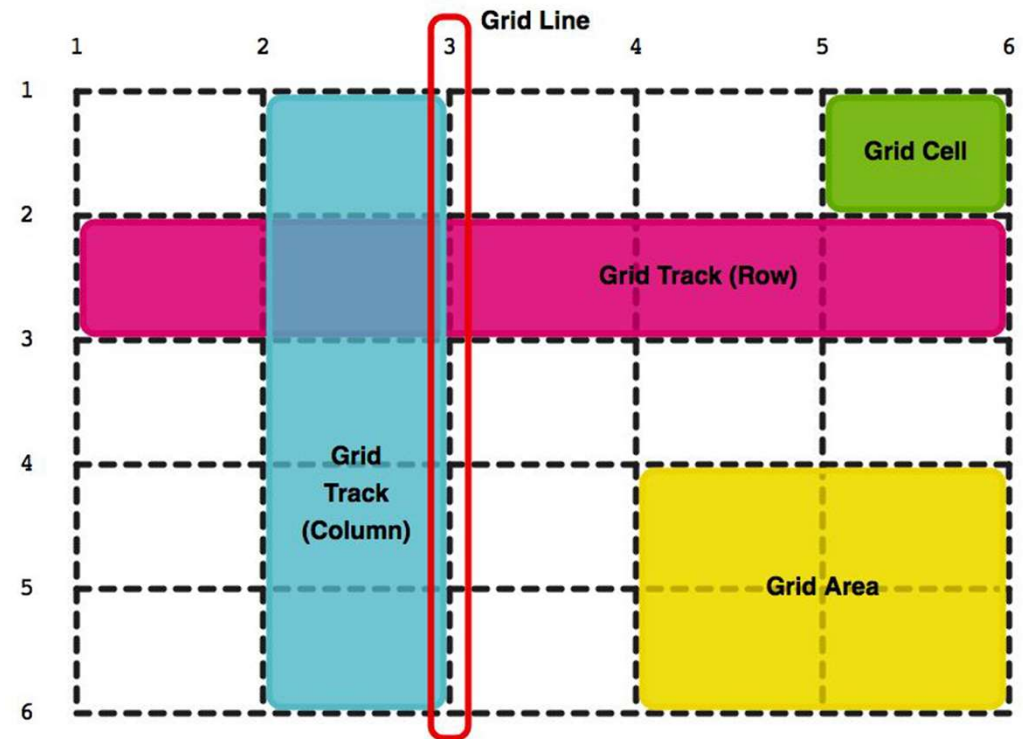
Usage	% of all users	
Global	84.9%	+ 3.22% = 88.11%
unprefixed:	84.9%	
Spain	91.39%	+ 1.77% = 93.16%
unprefixed:	91.39%	

Current aligned Usage relative Date relative Show all

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			49						
			65		10.3				4
	16	59	66		11.2				6.2
11	17	60	67	11.1	11.3	all	67	11.8	7.2
	18	61	68	12					
		62	69	TP					
			70						

Display Grid: Conceptos básicos

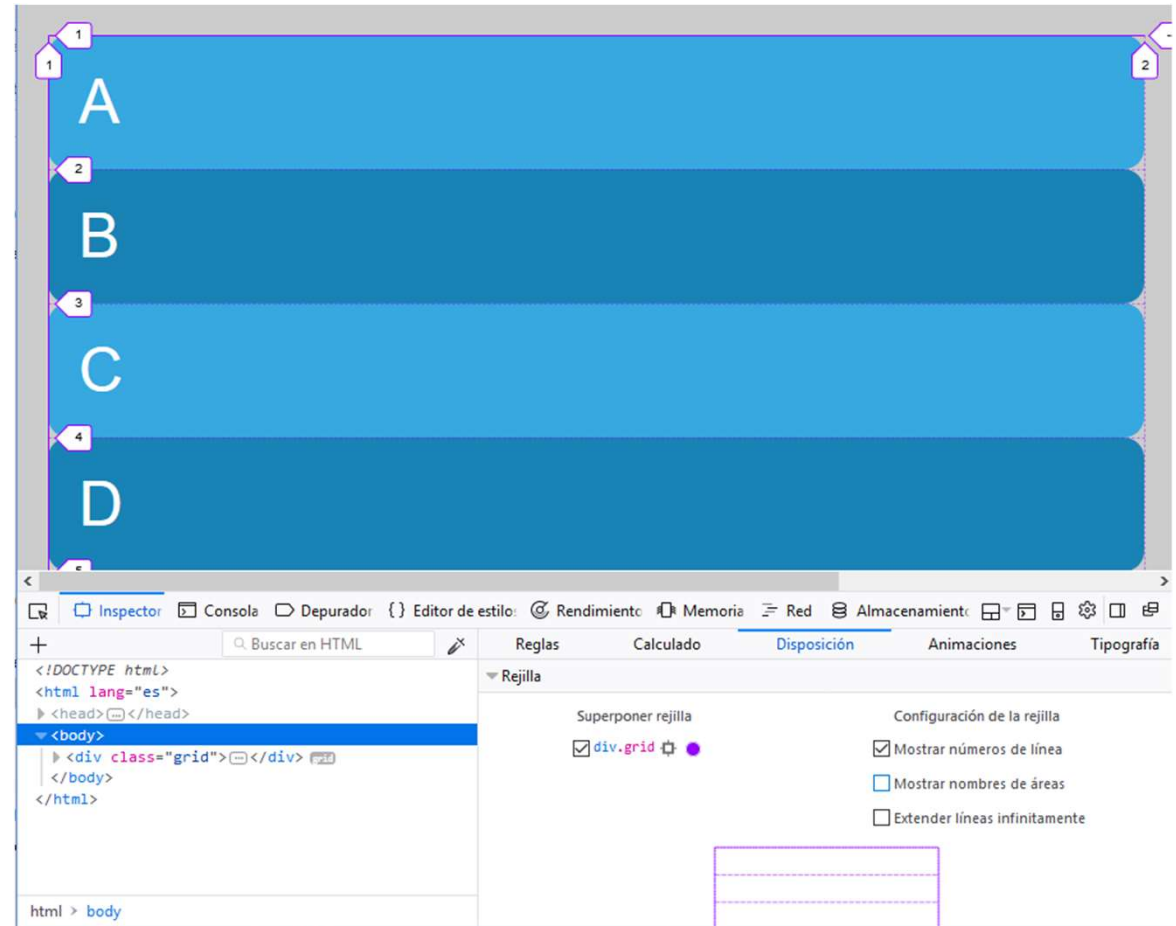
- Se define en base a **líneas** (Grid Line) que el sistema numera automáticamente
- 2 líneas consecutivas (i.e. de igual dirección) definen un **track** (Grid Track): filas y columnas
- el cruce de 2 tracks define una **celda** (Grid Cell)
- un conjunto de celdas adyacentes es un **área** (Grid Area)



Contenedor Grid

En cuanto se declara `DISPLAY: GRID` o `DISPLAY: INLINE-GRID`, los hijos directos de ese elemento pasan a ser `GRID ITEMS`.

Se ve especialmente bien en el inspector de firefox



Tracks

- Se definen cuando creamos el grid explícitamente.
Al contenedor grid,
le indico cómo deben ser las filas y las columnas:
- Existen múltiples sintaxis y formas de hacerlo.

grid-template-columns: ancho ancho
ancho

grid-template-rows: alto alto alto
alto

grid-column-gap: ancho;

grid-row-gap: ancho;

grid-auto-rows: alto

- en rem
- en fr
- repeat (n, valor)

Se definen como serán las filas que aparezcan
cuando haya más que las definidas en el grid
explícito

Tracks: mejor definición

- `repeat(num_tracks, ancho/anchos)`
- `minmax(número {fijo}, máximo {fr})`
- `repeat(auto-fill / auto-fit, ancho/anchos)`
se define el tamaño de los track -> el sistema calcula cuantos caben. Si se combina con un minmax, se consigue un layout totalmente responsive

```
grid-template-columns: repeat(auto-fill, 10rem); grid-template-columns: repeat(auto-fill, minmax(10rem, 1fr))
```

Items

La enorme potencia de GRID LAYOUT en parte viene dada porque una vez generada la rejilla, se puede posicionar cualquier item en cualquier posición dentro de la rejilla

- indicando las líneas de inicio / final
- indicando cuántas líneas se extiende (span)
- definiendo un área completa con grid-area
- es posible colocar 2 items en la misma posición (capas)

```
/* grid-column-  
start: 2;  
grid-column-end: 3;  
grid-row-start: 2;  
grid-row-end: 3; */  
grid-column: 2/3;  
grid-row: 2/span 2;
```

Líneas con nombres

- ayudan a recordar cómo van los tracks en layouts complejos.
- En RWD evitan sobrecribir la colocación de algunos ítems.
- Si cambia el número de tracks, no hay que sobre escribir todos los elementos

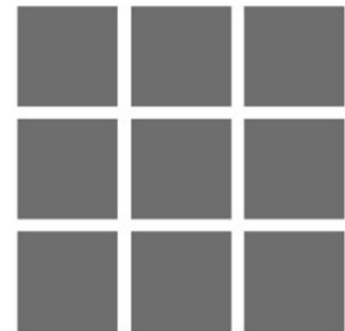
```
grid-template-columns: [start] minmax(200px, 20%) [main] 1fr [end];  
grid-template-rows: [start] auto [main] 80vh [footer] auto [end];
```

Áreas con nombres

En el contenedor se utiliza la propiedad grid-template-areas

grid-template-areas:

```
"header      header
... "
"sidebar     main  main"
"footer      footer
footer";
```

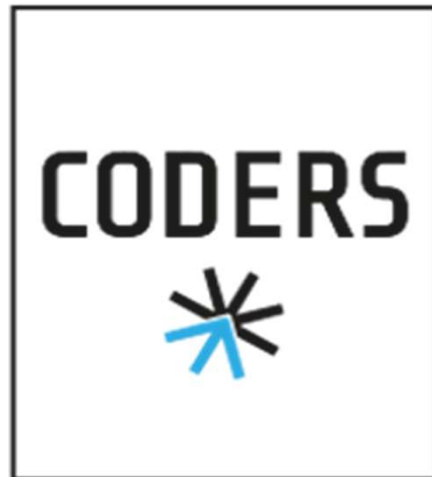


En los ítems se utiliza la propiedad grid-area

```
.header{
  grid-area:
  header;
}
```

Ejercicio. Display Grid

Ejemplo básico de RWD



- One
- Two
- Three
- Four



