HTML 5 y CSS 3

Nacho Viano

Índice - HTML5

- ¿ Qué es HMTL ?
- ¿ Qué necesitamos para empezar ?
- Creando un archivo HTML
- Estructura de una página HTML
- Doctype. Elementos inline vs block
- Atributos de las etiquetas
- Etiquetas semánticas
- DOM

1. ¿ Qué es HTML ?

- Las siglas vienen del inglés HyperText Markup Language o lenguaje de marcado de hipertexto.
- No es un lenguaje de programación.
- Lenguaje de marcado para crear páginas web o documentos.
- Son los bloques con los que se construye la web.

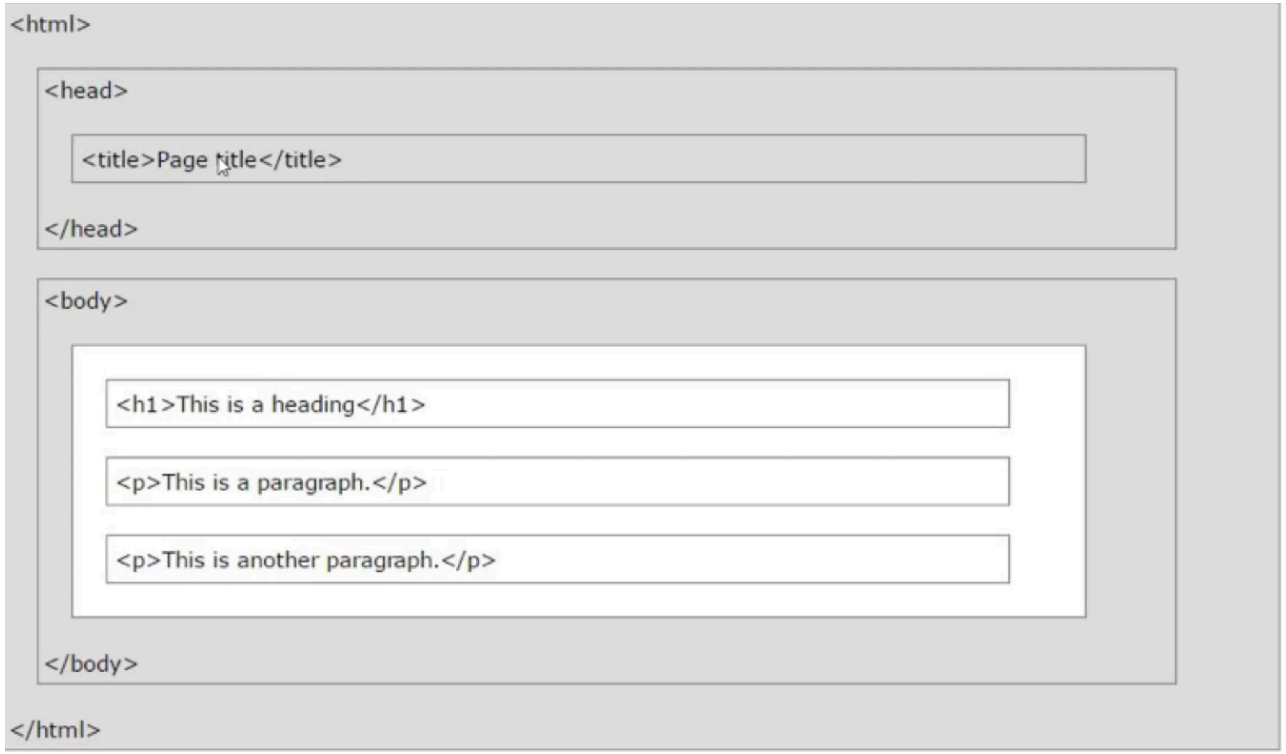
2. ¿ Qué necesitamos para empezar ?

- Un navegador web (cliente)
  - Google Chrome
  - Mozilla Firefox
  - Safari
  - Edge
  - ...
- Un editor de texto
  - Visual Studio Code
  - Atom.io
  - Sublime Text
  - ...

3. Creando un archivo HTML

- No necesita un servidor para ejecutarse.
- Los archivos tienen que acabar con la extensión .html.
- Se puede visualizar en los navegadores.
- index.html es la raíz o página principal de una web.

4. Estructura de una página HTML




```
<!DOCTYPE html>
<html>
    <head>
        <title>Título de la página</title>
    </head>
    <body>
        <h1>Mi primer título</h1>
        <p>Mi primer párrafo</p>
        <p>Mi segundo parrafo</p>
    </body>
</html>
```

5. DocType

- Define qué tipo de documento es la página.
- HTML4, HTML5, XHTML,etc.
- HTML4

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://w3.org/TR/html4/strict.dtd">
```

- HTML 5

```
<!DOCTYPE html>
```

6. Elementos inline vs Blocks

- Elementos inline
  - No empiezan una nueva linea.
  - Toman sólo el espacio que necesitan.

- Elementos blocks (bloques)
  - Empiezan una nueva linea
  - Toman todo el ancho posible

Ejemplos

Blocks: , <h1> - <h6>, <p>, <form>

Inline: , <img>, <a>

7. Atributos de las etiquetas

- Todas las etiquetas o tags pueden tener atributos
- Proporcionan información sobre el elemento
- Se colocan dentro de la parte inicial de la etiqueta
- Tienen un nombre y un valor

Estructura:

```
<nombre_etiqueta nombre_atributo="valor_atributo">contenido</nombre_etiqueta>
```

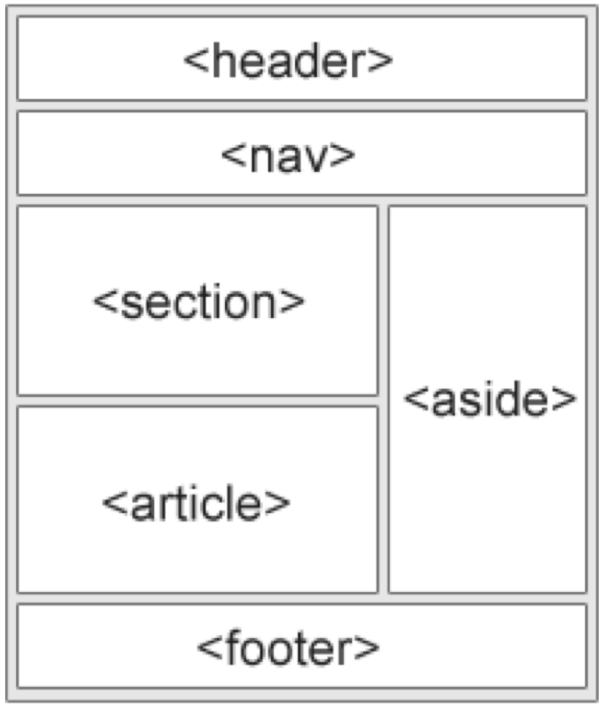
Ejemplo:

```
<h1 id="main_title" class="cover_title">Releevant</h1>
```

8. Etiquetas semánticas

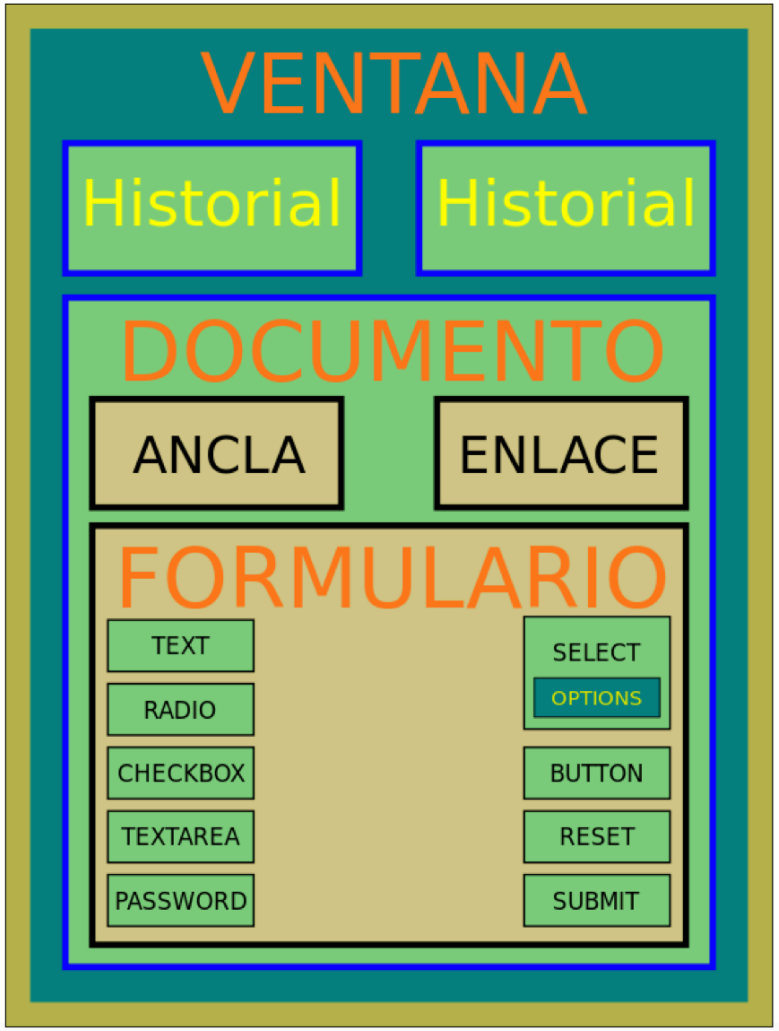
Hay etiquetas cuyo objetivo no es darle formato o estilo al contenido, sino que están diseñadas para decirle al navegador y al desarrollador qué tipo de contenido tienen.

```
<header>    </header>
<footer>    </footer>
<aside>     </aside>
<main>      </main>
<article>   </article>
<nav>       </nav>
<section>   </section>
<details>   </details>
```



8. DOM

- Siglas de Document Object Model o Modelo de Objetos del Documento.
- Conjunto estudar de objetos para representar documentos HTML, XML, etc.
- Interfaz que permitirá acceder a los elementos y modificar el contenido, estructuras o estilos



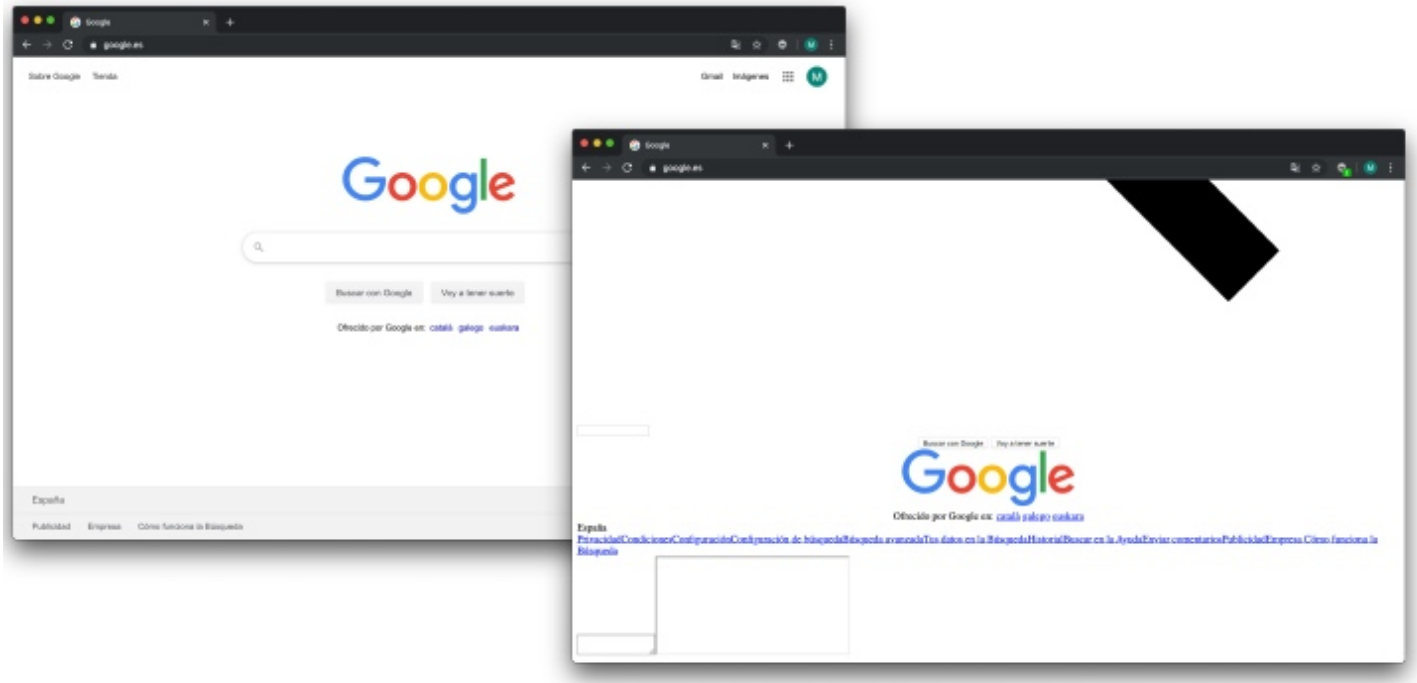
Índice - CSS3

- ¿ Qué es CSS ?
- Sintaxis CSS
- Cómo aplicarlo
- Selectores
- Colores y transparencia en CSS
- Tamaños
- Modelo de caja
- Posicionamiento
- Diseño Web Responsive / Mobile-first

1. ¿ Qué es Css ?

CSS (Cascading Style Sheets) u hojas de estilo en cascada al igual que HTML, no es un lenguaje de programación. En cambio, describe la presentación de un documento. Se utiliza para estilizar elementos escritos en un lenguaje de marcado como HTML. CSS separa el contenido de la representación visual del sitio.

La relación entre HTML y CSS es muy fuerte. Dado que HTML es un lenguaje de marcado (es decir, constituye la base de un sitio) y CSS enfatiza el estilo toda la parte estética de un sitio web), van de la mano.



CSS3 en específico, añade nuevas funcionalidades respecto a versiones anteriores.

1. ¿ Qué es Css ?

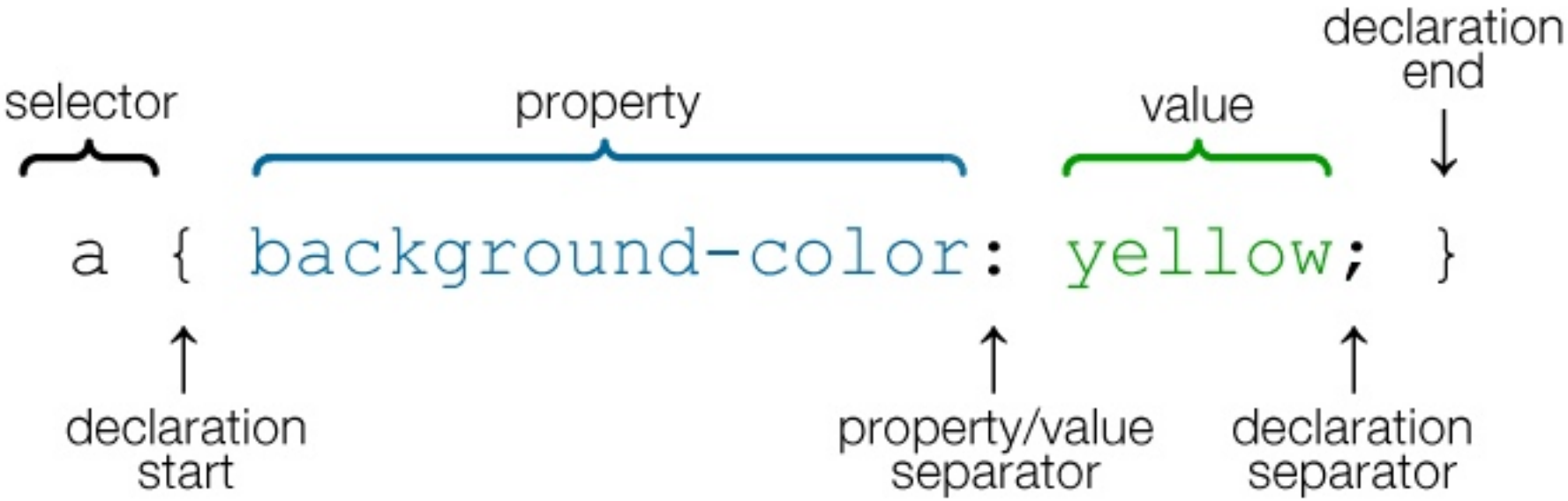
Estilos por defecto de los navegadores

Cuando los navegadores muestran una página web, además de aplicar las hojas de estilo de los diseñadores, siempre aplican otras dos hojas de estilos: la del navegador y la del usuario.

Las reglas que menos prioridad tienen son las del CSS de los navegadores, ya que son las primeras que se aplican. A continuación se aplican las reglas definidas por los usuarios y por último se aplican las reglas CSS definidas por el diseñador, que por tanto, son las que más prioridad tienen.

2. Sintaxis CSS

Los elementos fundamentales de la sintaxis de CSS son la Regla, el Selector, la Propiedad y su Valor



3. Cómo aplicarlo

- En linea (Inline)
  Los estilos inline son aquellos que se escriben directamente en la etiqueta (tag) del documento. Solamente afectan a la etiqueta en la que son aplicados. Son los más prioritarios.

```
<a href="blog.html" style="text-decoration: none;">Blog</a>
```

- Incrustados (Embedded) o internos
  Los estilos embedded son incrustados dentro de la sección <head> del documento. Estos sólo afectan a los elementos de la página en la que están incrustados. Son los segundos más prioritarios.

```
<style>
    p {
        color: aqua;
    }
</style>
```

3. Cómo aplicarlo

- Externos (External)
  Estos son escritos en un documento separado independiente y posteriormente vinculado a una o varias páginas web. Los estilos externos afectan a cualquier documento vinculado a ellos. Es la opción generalmente recomendada. Son los últimos en cuanto a prioridad.

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

4. Selectores

Hay diferentes formas de hacer referencia a uno o más elementos.

- Selector universal: *
- Selector de etiqueta: elementname
- Selector de clase: .classname (recomendado)
- Selector de ID: #idname
- Selector de atributo: [attr=value]

4. Selectores

Además, podemos combinarlos de las siguientes maneras:

• Por jerarquía:	Con un espacio “ “	div span
• Con el hijo de un elemento:	Con “>”	ul > li

Pseudo

• pseudo-elementos:	Con “:”	a:visited
• pseudo-classes:	Con “::”	p::first-line

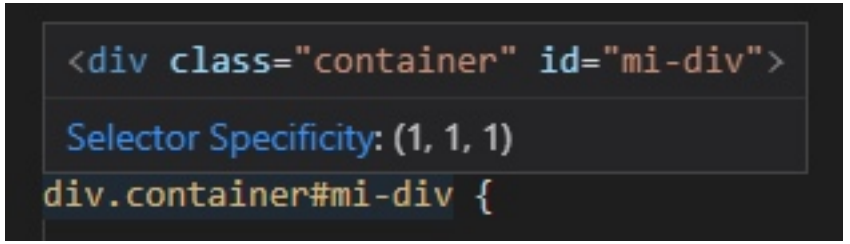
4. Selectores

Especificidad

A través de la especificidad de un conjunto de selectores, los navegadores van a decidir cuál es la propiedad con más relevancia para un elemento, y por lo tanto, la que será aplicada.

Se va calcular en función de lo específico que sea un selector o la suma de varios. En orden creciente:

- Selectores de tipo o etiqueta y pseudo-elementos (p.ej. ::before)
- Selectores de clase, atributo y pseudo-classes (p.ej. :hover)
- Selectores de ID



!important

Podemos escribir !important tras una propiedad para que sobreescriba cualquier declaración previa. Es una mala práctica y debe evitarse. Debe usarse la especificidad en su lugar.

5. Colores y transparencia en CSS

Colores

- Nombres de colores en HTML 5
- Hexadecimal
- RGM

Transparencia

Opacity o rgba(R,G,B,O)

```
h1 {
    color: red;
}

h2 {
    color: #ff0000;
}

.h3 {
    color: rgb(255, 0, 0);
}
```

5. Colores y transparencia en CSS

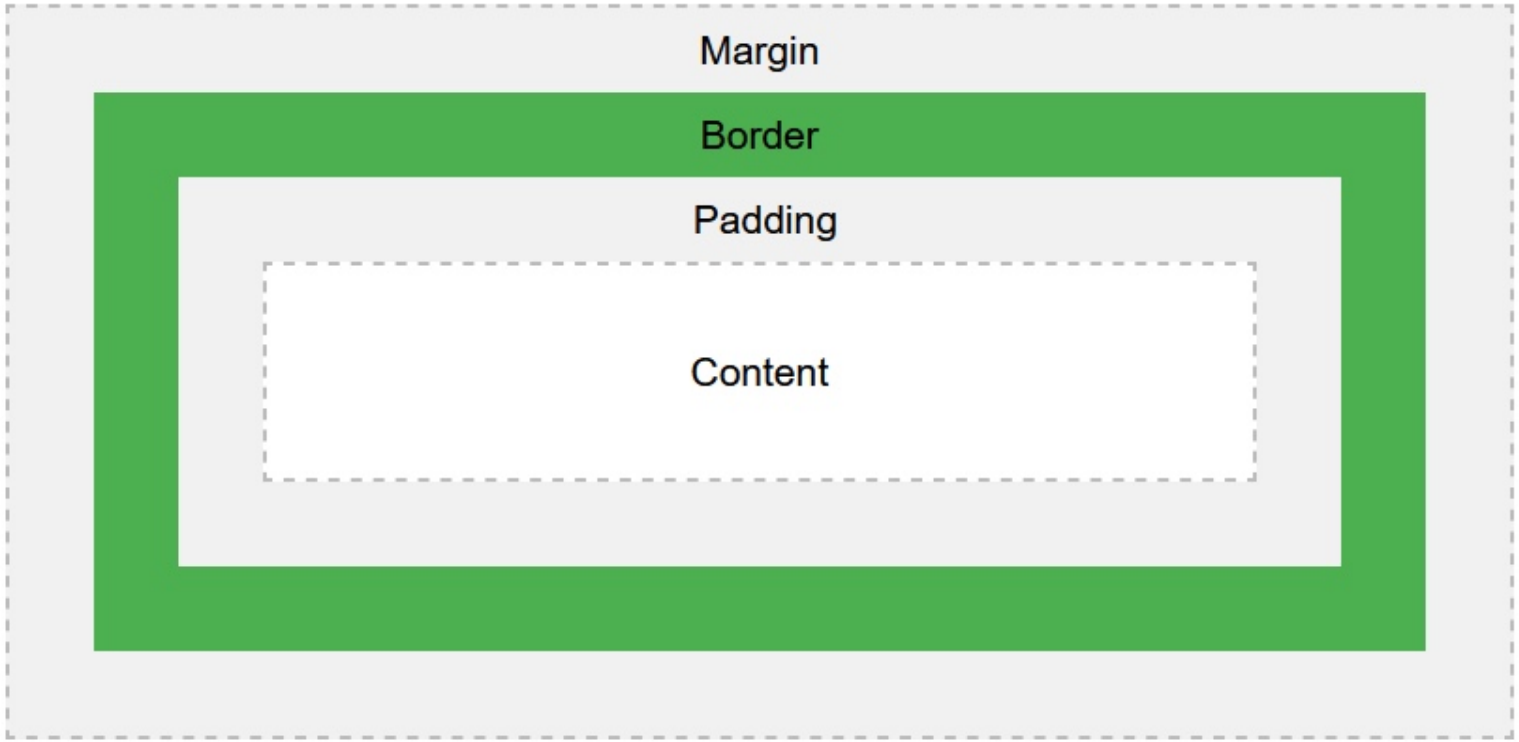
Imágenes

Además, podemos establecer imágenes como fondo de un elemento.

```
div.width-background {
    height: 50vh;
    background-image: url(../images/background.jpg);
    background-position: center bottom;
    background-attachment: fixed;
    background-size: cover;
}
```

6. Modelo de cajas

Cada elemento que encontramos dentro de un documento HTML se encuentra contenido en una caja rectangular que cuenta con una serie de propiedades que afectaran el cómo se muestran los elementos.



7. Tamaño

Unidades absolutas

Es la unidad que es fija y que no depende de otro valor de referencia:
in, pulgadas (inches), cm, mm...

Unidades relativas

No son fijas ya que están relacionadas con otro valor.

- em: Relativa al tamaño de letra del elemento.
- rem: Relativa al tamaño de letra del elemento raíz.
- px: Píxel, relativa a la resolución de la pantalla del dispositivo.
- %: Relativo a su elemento padre.
- vw, vh.
- vmin, vmax, ex, ch...

7. Tamaño

Unidades relativas al Viewport

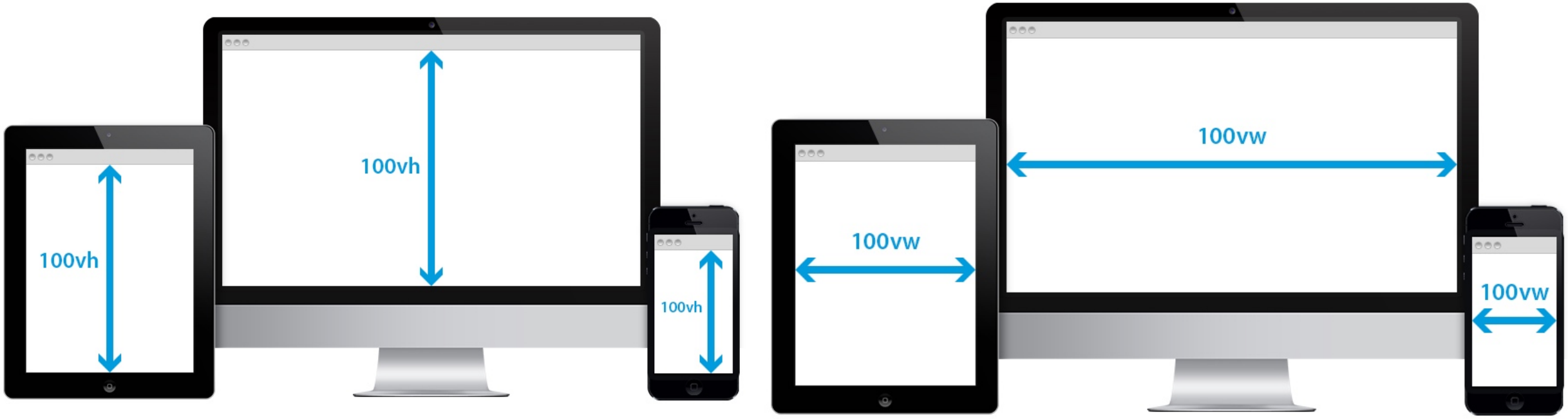
Las unidades de longitud porcentuales de ventana de visualización (viewport-percentage lengths) o más comúnmente llamadas viewport units son unidades css relativas a la ventana gráfica del navegador o visualizador.

La ventana gráfica puede considerarse como la capa padre de toda nuestra web, y las unidades “vh” y “vw” son las unidades en altura y anchura que hacen referencia a esta ventana.

7. Tamaño

vh (viewport-height) = Altura de la ventana gráfica.
vw (viewport-width) = Anchura de la ventana gráfica.

Sus valores se expresan en porcentaje de 0 a 100.



7. Tamaño

Display

Define el tipo de caja generada por un elemento.

- none
- inline
- block
- inline-block
- ...

8. Posicionamiento

- static - Por defecto
- relative - relativo a su posición original
- fixed - fijo en un punto de la ventana
- absolute - Posición absoluta respecto a su contenedor padre.
- sticky - Alterna entre relative y fixed en función del scroll.
- flotante - Se convierte en cajas flotantes que pueden desplazarse a la izquierda o derecha.

El eje Z

Podemos controlas la superposición de los elementos gracias a la propiedad z-index.

8. Posicionamiento

Overflow

Esta propiedad gestiona el comportamiento cuando un elemento es demasiado grande para caber dentro del bloque que lo contiene, produciéndose un desbordamiento del contenido.

overflow: [visible I hidden I clip I scroll I auto]{1,2}

Nota: dotar de la propiedad overflow a un contenedor sin altura específica y cuyos hijos son flotantes puede servirnos para que se adapte al tamaño de los hijos y no colapse a una altura de cero, también evitamos la alternativa de utilizar “ clearfix”.

9. Diseño Web Responsive / Mobile-first

Una página responsive adapta la visualización del contenido a las diferentes pantallas de los dispositivos para mantener una experiencia correcta en todos los usuarios. Podemos hacerlo con CSS3 a través de la regla @media definiendo diferentes breakpoints.

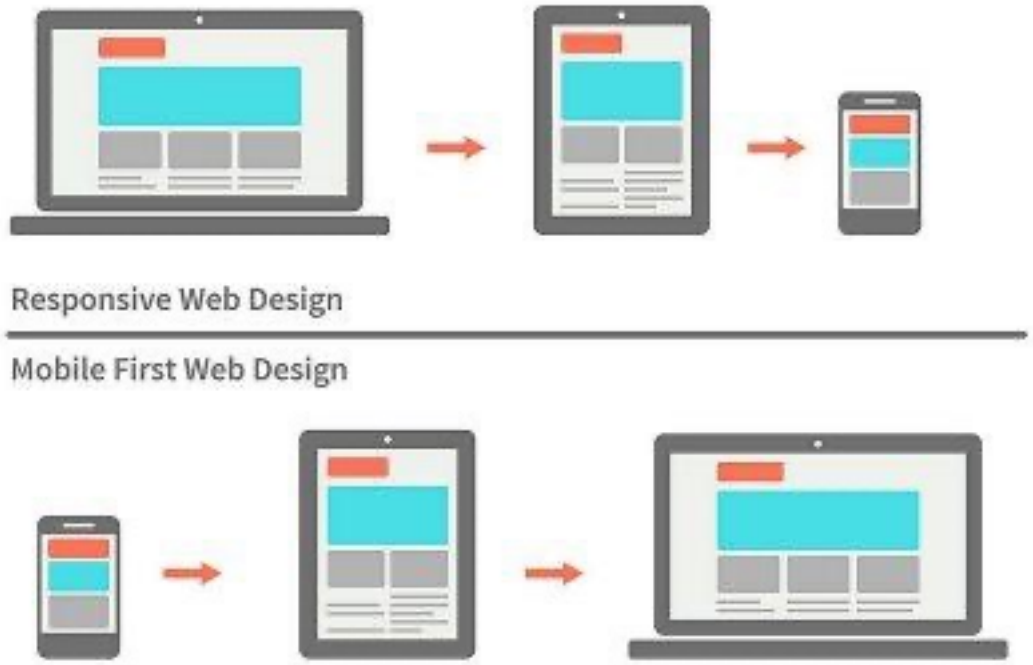
```
@media only screen and (max-width: 758px) {

    .container-1, .container-2, .container-3 {
        float: none;
        width: 100%;
    }
}
```

9. Diseño Web Responsive / Mobile-first

Al diseñar primero para móviles, partimos de un diseño más sencillo, ya que suelen tener estructuras más simples. Posteriormente, añadir reglas que modifiquen dicha estructura en pantallas más grandes.

Para hacer esto, primero escribiremos nuestro CSS para móviles y después añadiremos reglas con una condición min-width y así poder personalizar los siguientes tamaños.



Índice - CSS3 - Segunda Parte

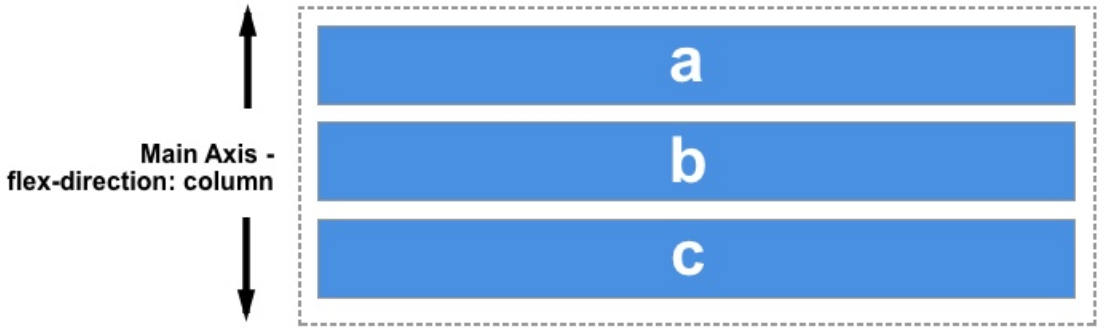
- Flexbox
- CSS Grid
- Iconos
- Gradientes y sombras
- Transformaciones
- Transiciones
- Animaciones

10. Flexbox

Es un modelo de disposición de elementos en una dimensión: fila / columna. Para trabajar con flexbox, necesitamos tener bajo control sus dos ejes: main y cross, que dependerán de la dirección que elijamos.

display: flex – Activa flexbox en el contenedor.

flex direction: row | row-reverse | column | column-reverse



10. Flexbox

Para organizar el contenido de un contenedor flex, utilizaremos las siguientes propiedades que dependerán de la dirección:

- flex-wrap: nowrap I wrap I wrap-reverse - Creación de nueva linea si no hay espacio.
- flex-flow: flex-direction I flex-wrap - Atajo para combinar ambas propiedades.

Para controlar el posicionamiento en eje principal (main axis), en el contenedor, podemos configurar:

- justify-content: flex-start I flex-end I center I space-between I space-around I space-evenly

Y en el eje perpendicular (cross axis):

- align-items: stretch I flex-start I flex-end I center

11. CSS Grid

Es un modelo de disposición de elementos en DOS dimensiones: filas y columnas.

Proporciona más libertad en la gestión de distribuciones más complejas.

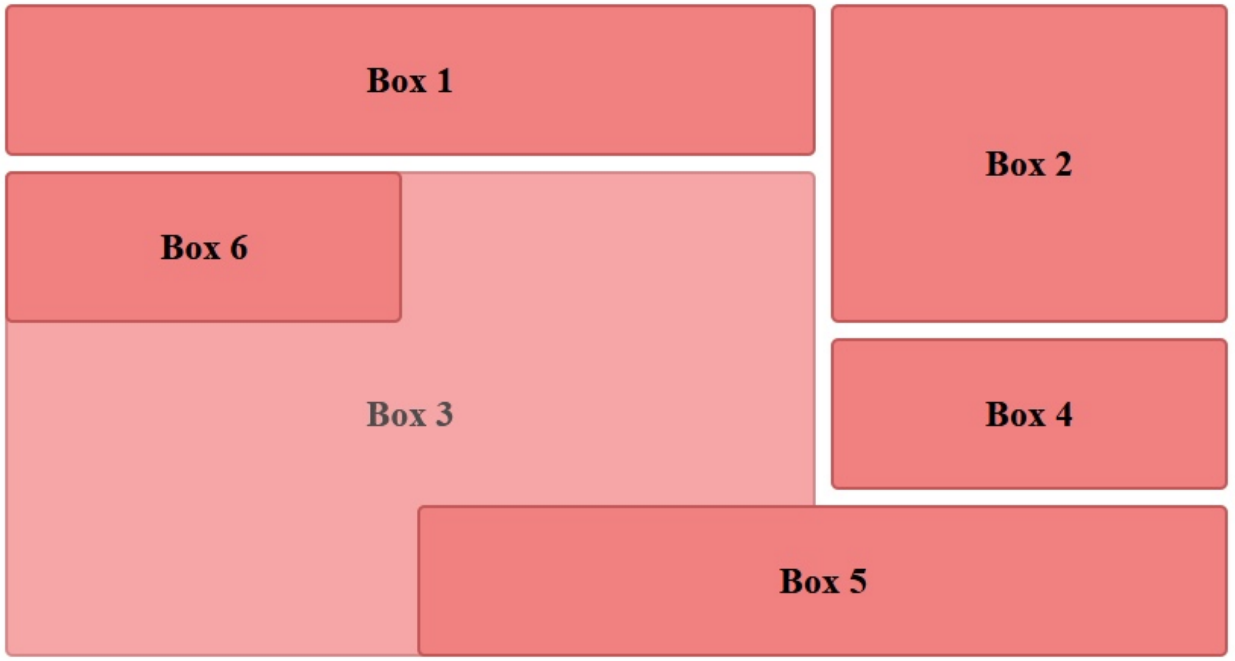
display: grid

Propiedades comunes:

grid-template-columns: 1fr 2fr

grid-template-rows: 1fr 2fr 1fr

-gap: 10px;



https://developer.mozilla.org/es/docs/Web/CSS/CSS_Grid_Layout/Conceptos_Basicos_del_Posicionamiento_con_Rejillas

12. Iconos

Para el uso de iconos en nuestras interfaces, podemos utilizar imágenes con un tamaño adecuado pero en la mayoría de casos será más rápido y cómodo el uso de librerías específicas de iconos.

Una de ellas, es fontawesome icons. Descargando los ficheros necesarios o haciendo uso de su CDN, podemos disponer de los iconos de forma casi inmediata.

```
    <link rel="stylesheet" href="css/all.css">
</head>
<body>
    <i class="fas fa-user"></i> <!-- uses solid style -->
    <i class="far fa-user"></i> <!-- uses regular style -->
    <i class="fab fa-github-square"></i> <!-- uses brands style -->
```

13. Gradientes y sombras

Un gradiente de color o degradado puede formar parte de un buen diseño, habitualmente debe utilizarse muy sutilmente. Puede ser un gradiente lineal o radial.

- background: linear-gradient(to right, blueviolet, white)
- background: linear-gradient(to bottom, blue, white 80%, orange)
- background: linear-gradient(to right, rgba(255,255,255,0), rgba(255,255,255,1) 30%),url(image.jpg)
- background: radial-gradient(lightcoral 5%, yellow 25%, #1E90FF 50%)

Sombras

box.shadow: offset-x | offset-y | radio-desenfoque | color

14. Transformaciones

Las transformaciones que podemos realizar a través de la propiedad transform son:

- trasladar: translate(x,y)
- rotar: rotate(angle)
- escalar: scale(x)
- sesgar: skew(angle)

15. Transiciones

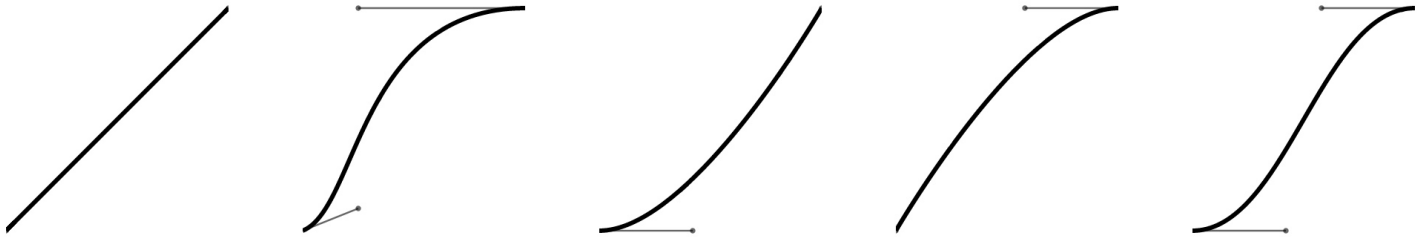
Las transiciones nos permiten animar un elemento HTML partiendo de un estado A hacia un estado B.

Debemos usar dos parámetros:

- transition-property - Propiedades que se verán afectadas
- transition-duration - Duración de la transición

Adicionalmente:

- transition-delay: <segundos>
- transition-timing-function: ease I ease-in I ease-out I ease-in-out I linear



16. Animaciones

Las animaciones, a diferencia de las transiciones, nos permite controlar estados intermedios y repeticiones permitiéndonos realizar diseños más complejos.

- animation-name: nombre de la regla @keyframes
- animation-duration - tiempo en realizar 1 ciclo

```
.text-animation {
    animation-duration: 3s;
    animation-name: slidein;
    animation-iteration-count: infinite;
}

@keyframes slidein {
    from {
        margin-left: 100%;
    }

    to {
        margin-left: 0%;
    }
}
```

Algunas de la propiedades:

- animation-iteration-count
- animation-timing-function
- animation-delay

Ejemplo: https://codepen.io/Wujek_Greg/pen/KRXYpg

HTML 5 y CSS 3

Nacho Viano