



# UT1–1. Usabilidad



# Índice

1. ¿Qué es la usabilidad?
2. Principios de usabilidad de Steve Krug.
3. Navegación en la Web.
4. Principios del diseño de la interacción.
5. Análisis de la usabilidad.
6. Referencias.

# 1. ¿Qué es la usabilidad?

- Esta palabra no existe en el diccionario de la RAE.
- Concepto utilizado para **cualquier producto**, no sólo para páginas web.
- **Expresiones “equivalentes”:**
  - “Facilidad de uso”.
  - **Diseño amigable**.
  - “Experiencia de usuario” (UX, User eXperience).
- Consecuencia: para que algo sea usable, hay que diseñarlo pensando en el usuario (**diseño centrado en el usuario**).



- Los principios de usabilidad no cambian\*.
- La usabilidad no va de tecnología, va de cómo la gente utiliza las cosas.
- Se aplica al diseño de cualquier cosa con la que la gente tenga que interactuar.

\*Steve Krug. "No me hagas pensar"  
1<sup>a</sup> ed: 2000; 3<sup>a</sup> ed: 2016

## Definición formal

*ISO/IEC 9241-11: Consejos de usabilidad*

*"Usabilidad es la **eficacia**, **eficiencia** y **satisfacción** con la que un producto permite alcanzar **objetivos específicos** a **usuarios específicos** en un **contexto de uso específico**".*

- **Eficacia:** ¿pueden los usuarios completar las tareas, conseguir sus objetivos con el producto, es decir, hace lo que ellos quieren?
- **Eficiencia:** ¿cuánto esfuerzo tienen que hacer los usuarios para conseguirlo? (a menudo se mide en tiempo).
- **Satisfacción:** ¿qué piensan los usuarios sobre la facilidad de uso de los productos?

De la definición anterior, se deduce que la usabilidad dependerá de:

- **El objetivo del sitio Web.**
- **Tipos de usuarios. Necesidades.**
  - Mayores / Jóvenes / Conocimientos...
- **Contexto de uso:**
  - Navegador.
  - Dispositivo de acceso.
  - Velocidad de conexión.
  - ...

# Estándares de diseño Web

- Seguir los estándares de diseño Web es un **primer paso para construir sitios Web usables**,
  - tanto los estándares de los lenguajes **HTML**, **XHTML**, **CSS**, etc.,
  - como otros estándares: **Accesibilidad**, etc.
- De este modo, es más fácil que pueda ser usable desde distintos **navegadores**, **dispositivos** y por **usuarios** con diferentes necesidades.

## 2. Principios de usabilidad de Steve Krug

- **Principio 1: "No me hagas pensar".**
- Significa: saber utilizar una página sin hacerte demasiadas preguntas.
- Cosas que hacen pensar:
  - Nombres de las cosas, deben ser familiares (ej. "buscar").
  - ¿Dónde hacer clic?

## SIN PENSAR

Veamos. Esto tiene aspecto de ser las categorías de productos...



Memoria,  
módem...  
Aquí están:  
los monitores.  
Clic.



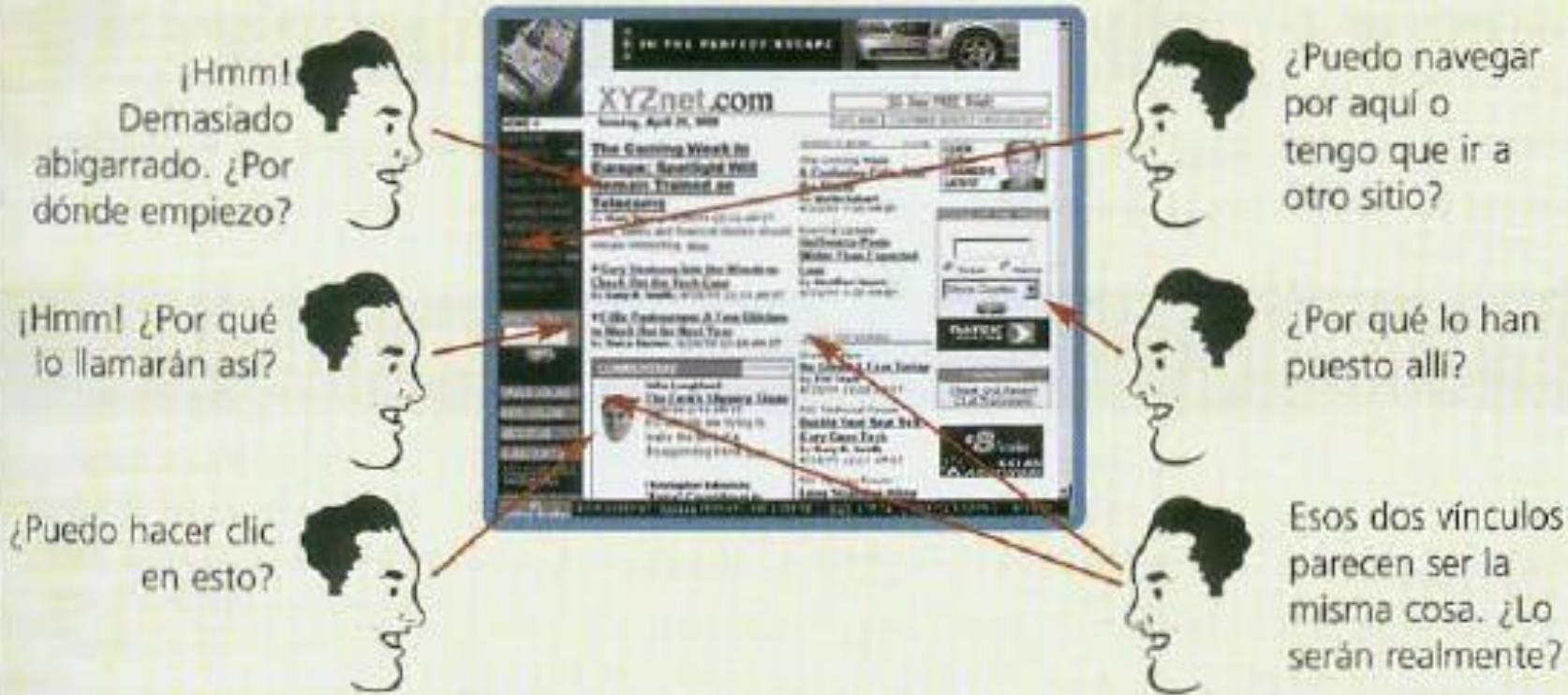
• ... y éstas son las compras especiales del día.

## 2. Principios de usabilidad de Steve Krug

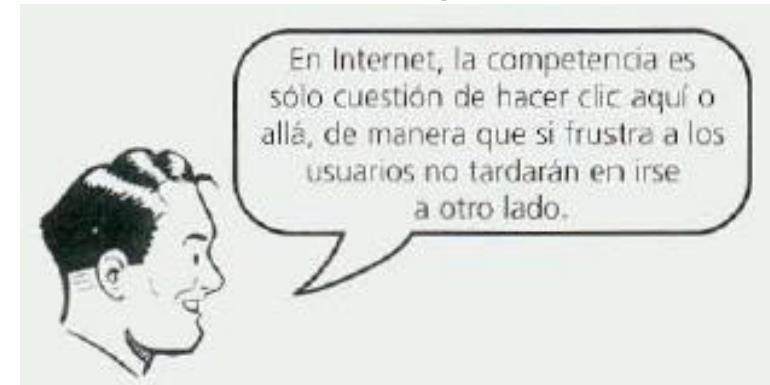
< EVIDENTE		REQUIERE PENSAR >
<p>¡Trabajos!</p> <p>Clic</p> 	<p>¡Hmm! (Milésimas de segundo pensando).</p> <p>Trabajos.</p> <p>Clic.</p> 	<p>¡Hmm! Podría tratarse de las ofertas de empleo, pero parece que es más que eso. ¿Debería hacer clic o seguir buscando?</p> 

< EVIDENTE HACER CLIC	REQUIERE PENSARLO >	
<p>Clic</p> 	<p>¡Hmm! (Milésimas de segundo pensando)</p> <p>Supongo que éste es el botón.</p> <p>Clic</p> 	<p>¡Hmml!</p> <p>¿Es éste el botón?</p> 

### PENSANDO

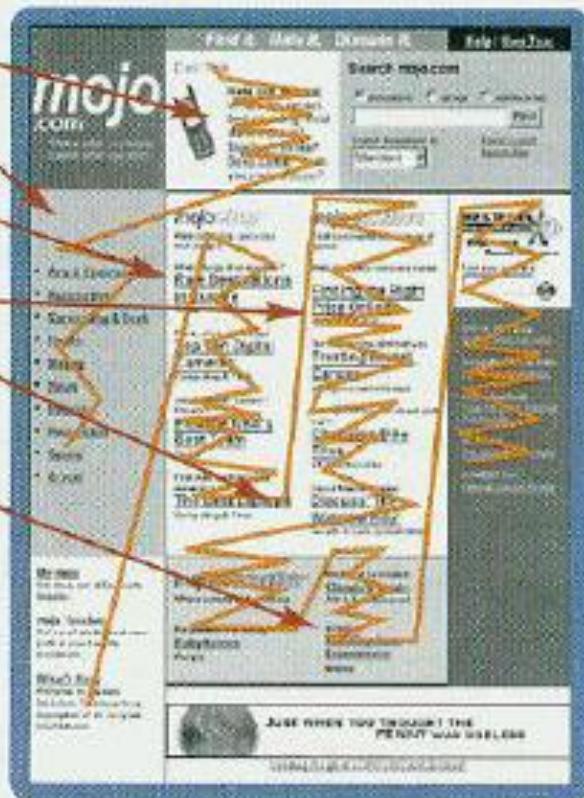


- ¿Por qué es importante no tener que pensar?: la gente dedica mucho menos tiempo a las páginas Web de lo que pensamos.

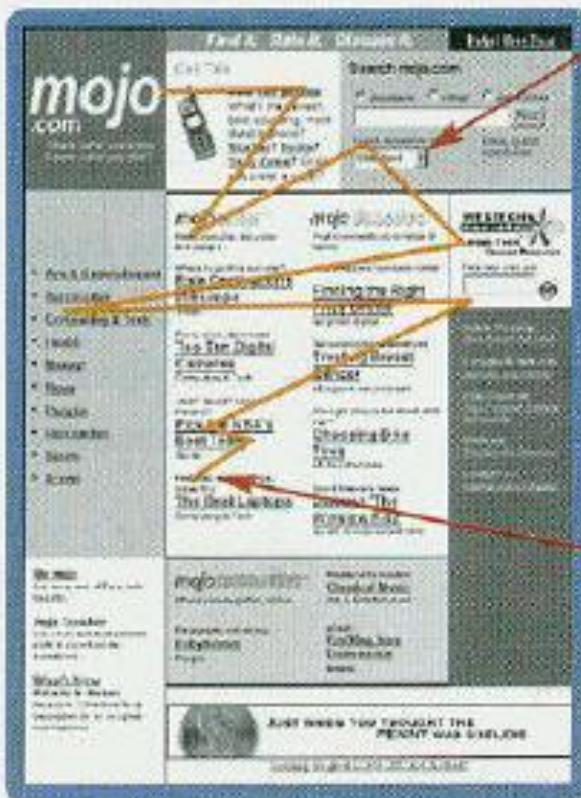


- ¿Cómo usamos realmente la Web?:
  - No leemos, escaneamos.
  - No tomamos decisiones óptimas, sino decisiones suficientes.
  - No estudiamos cómo funcionan las cosas, simplemente las utilizamos.
- Si nuestra página es fácil de entender, será fácil de usar.

### LO QUE DISEÑAMOS PARA...



### LA REALIDAD...



Mira alrededor febrilmente en busca de algo que:

- a) que es interesante o se parece vagamente a lo que está buscando, y
- b) en lo que puede hacer clic.

Tan pronto como encuentra algo que se ajusta bastante bien, hace clic.

Si no resulta, hace clic en el botón Atrás y lo intenta otra vez.

### Diseñar páginas para ser escaneadas:

#### ■ Seguir los convencionalismos:

- Dónde suelen estar las cosas. Ej. Logo esquina superior izquierda, navegación primaria en lado izquierdo.
- Cómo suelen funcionar las cosas. Ej. Carrito de compra.
- Qué aspecto suelen tener las cosas. Ej. Link vídeo, ícono búsqueda, opciones para compartir en redes sociales.

➔ Diferentes clases de sitios: comercio, universidades, blogs, películas...

#### ■ Jerarquías visuales claras en cada página.

- Los elementos más importantes deben ser los más destacados.
- Lo que está relacionado por lógica, también debe estarlo de forma visual.
- Los elementos se agrupan visualmente para mostrar que son parte de algo.

#### ■ Dividir la página en zonas claramente definidas: Ayuda al usuario a centrarse sólo en lo que le interesa.

#### ■ Dejar claro dónde se puede hacer clic.

#### ■ Eliminar el "ruido visual": todo parece importante, desorganización y desorden.

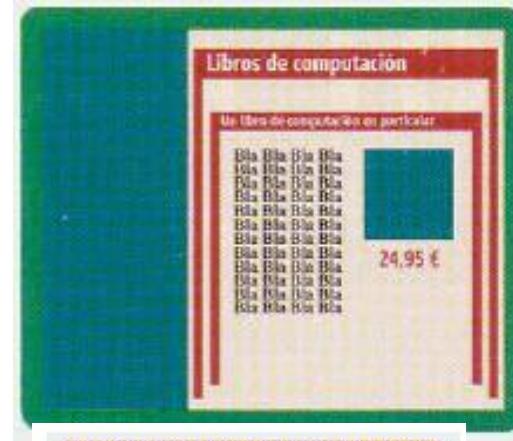
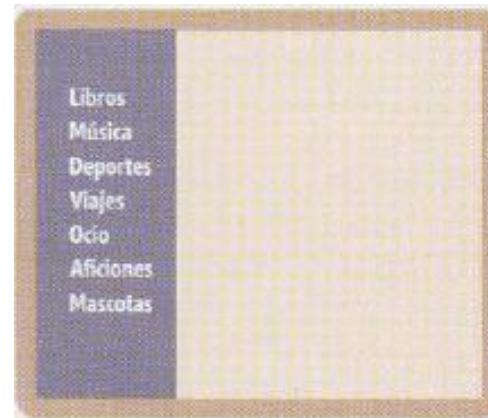
#### ■ Formatear el texto para que sea fácil de leer:

- Utilizar encabezados: cuidar niveles, no dejar “que floten”.
- Párrafos cortos.
- Utilizar viñetas.
- Destacar los términos clave (pocos).

# Diseñar páginas para ser escaneadas:

## ■ Ejemplos:

- Utilidad convencionalismos: <http://news.tbs.co.jp/>
- Jerarquía visual clara:



El titular que abarca estas tres columnas muestra, obviamente, que dichas columnas son parte del mismo artículo.

El tamaño del titular revela en tan sólo una ojeada que se trata del artículo más importante.



Esta jerarquía visual engañosa sugiere que todas las secciones del sitio forman parte de la sección "Libros sobre computadoras".

- **Principio 2: “No importa el número de veces que hay que hacer clic en algo, si la opción es mecánica e inequívoca.”**
- Por ejemplo:
  - Formularios: preguntas fáciles de contestar.
    - Elecciones con menos información, aunque más clics, son más fáciles que con mucha información en pocos clics .
  - Ayuda:
    - Breve: Poca información que ayude.
    - Oportuna: Situada para que se encuentre cuando se necesita.
    - Ineludible: que sea fácil de descubrir y nunca pase desapercibida.

- **Principio 3: “Elimine la mitad de las palabras de cada página. Después, deshágase de la mitad restante.”**

- Es una forma de decir que hay que eliminar el texto innecesario, de verdad.
  - Beneficios de suprimir palabras que no van a leerse:
    - Reduce el nivel de ruido del sitio web.
    - Hace que el contenido principal sea más relevante.
    - Se acortan las páginas, permitiendo que los usuarios vean más contenido de un vistazo.

- **El discurso excesivo tiene que desaparecer.**

- **Ejemplos:** Excesivos textos preliminares con bienvenida al sitio, publicidad del sitio.

- **Eliminar instrucciones.**

- Si son necesarias, reducirlas a lo esencial.

### Ejemplo instrucciones formulario:

ANTES: 103 PALABRAS	
<p>El siguiente cuestionario se ha diseñado para proporcionarnos la información necesaria que nos ayudará a mejorar el sitio y hacerlo más adecuado a sus necesidades.</p>	<p>La primera frase es sólo un discurso innecesario introductorio. Sabemos lo que es un estudio (una encuesta); lo único que necesito son las palabras "nos ayudará" para que me muestren que entienden que les estoy haciendo un favor cumpliéndolo.</p>
<p>Por favor, seleccione sus respuestas en los menús desplegables y en los botones de radio inferiores.</p>	<p>La mayoría de los usuarios no necesitan que se les diga cómo llenar un formulario web y los que necesitan saberlo desconocen lo que es un "menú desplegable" y "un botón de radio".</p>
<p>El cuestionario ha de completarse en tan sólo 2-3 minutos.</p>	<p>En este punto aún trato de decidirme si me molesto con el cuestionario, pero saber que es corto me dice que la información es útil.</p>
<p>En la parte inferior de este formulario puede dejar, si lo desea, su nombre, su dirección y número de teléfono. De hacerlo, contactaremos con usted en un futuro para que participe en una encuesta que nos ayude a mejorar este sitio.</p>	<p>Esta instrucción carece de utilidad para mí en este punto. Sería, al final del cuestionario, cuando podría hacer algo con él. El único efecto que produce es que las instrucciones parezcan más desalentadoras.</p>
<p>Si tiene comentarios o preguntas que hacer y que requieran una respuesta, por favor, contacte con el Servicio al cliente.</p>	<p>El hecho de no utilizar este formulario en caso de necesitar una respuesta es una información importante y práctica. Desgraciadamente, sin embargo, no se molestan en indicarme cómo contactar con el departamento de Servicio al cliente (o aún mejor, en facilitarme un vínculo con el que poder hacerlo desde aquí mismo).</p>

### Ejemplo instrucciones formulario:

#### DESPUÉS: 43 PALABRAS

Ayúdenos, por favor, a mejorar este sitio contestando a las siguientes preguntas. Sólo tardará 2-3 minutos en completar este cuestionario.

ADVERTENCIA: Si tiene alguna pregunta o comentario que requiera respuesta no use este formulario. Contacte entonces, por favor, con nuestro Servicio al cliente.

# 3. Navegación en la Web\*.

## ■ Ejemplo de búsqueda en un supermercado.

Normalmente buscamos algo.

- Decidimos si nos orientamos por los **letreros** o **preguntamos**.
- Los letreros siguen unas **convenciones** y tienen una **jerarquía**.
- Cuando llegamos a la estantería, **buscamos el producto**.

## ■ En un sitio Web, buscamos de forma similar.

Necesitamos saber, en cualquier momento:

- ¿**Dónde estoy**? ¿Cómo se llega hasta aquí?
- ¿**Cómo puedo ir** a donde me interesa?

## ■ **Algunas técnicas que facilitan la navegación en un sitio Web:**

1. Navegación constante.
2. Nombre de la página.
3. Coherencia del diseño.
4. Marcar la posición en los menús.
5. Breadcrumbs (migas de pan).

## 1. Navegación constante

- La navegación constante (también navegación global o coherente) **es el conjunto de elementos de navegación que aparecen en todas las páginas** de un sitio (excepto en los formularios, donde puede ser una distracción).
- **Proporcionan consistencia a las páginas de un sitio web:** Siempre está en el mismo lugar, con un aspecto similar, funciona igual. Nos informa de que seguimos en el mismo sitio y de que en todas las páginas se trabaja igual.



## 1. Navegación constante

Suele incluir **cinco elementos importantes**:

### 1. Identificación del sitio.

Además, normalmente, nos permite ir a la **página de inicio**.

En la parte superior izquierda (si el idioma se lee de izquierda a derecha).

### 2. Secciones (llamadas navegación principal).

Enlaces a las **secciones principales del sitio**.

La navegación constante puede incluir un espacio para la **navegación secundaria** (subsecciones de una sección).

### 3. Utilidades.

Enlaces a **elementos importantes del sitio que no son parte de la jerarquía**. Son cosas que ayudan a utilizar el sitio: Registro, Ayuda, Mapa del sitio, Carrito de la compra), o proporcionan información sobre el sitio (Sobre nosotros, Contacto).

Menos llamativas que las secciones.

Normalmente no habrá más de cuatro o cinco utilidades.

Las utilidades menos utilizadas pueden ir en la parte inferior de la página.

### 4. Un enlace a la página de inicio

### 5. Un modo de buscar



## 1. Navegación constante

- En las páginas donde hay que llenar un **formulario**, la navegación constante puede ser una distracción innecesaria.
  - Ejemplo: Pago de compras.
- Resulta práctica una **mínima versión** de la navegación constante que cuente sólo con:
  - La identificación del sitio.
  - Un vínculo a la página principal.
  - Cualquier servicio que pueda ser de ayuda para llenar el formulario.

## 2. Nombre de la página.

- **Todas las páginas** lo necesitan.
- Debe estar en el **lugar adecuado**. Debe dar la impresión de abarcar el contenido que es propio de esa página (después de todo, esto es lo que estamos nombrando, no la navegación o los anuncios).
- Debe ser **más llamativo** que el resto de la página. Combinando posición, tamaño, color y tipografía, para que quede claro que "Esta es la cabecera de la página".
- El nombre debe **coincidir con el texto del enlace** en el que hemos hecho clic. Si no es posible, deberían parecerse lo más posible.

#### Haciendo clic en Todos los departamentos – Libros - Libros

The screenshot shows the Amazon.es homepage. On the left, there's a sidebar with various categories like 'Música digital', 'Amazon Cloud Drive', and 'Libros'. The 'Libros' category is highlighted with a red box. A blue arrow points from this 'Libros' box down to the main content area. The main content area has a header 'Todos los departamentos' with a dropdown menu, followed by navigation links: 'Amazon.es de Ana', 'Nuestras ofertas', 'Cheques regalo', 'Vender', and 'Ayuda'. Below this is a search bar and a 'Libros' section. The 'Libros' section title is also highlighted with a red box. Underneath it are several sub-categories: 'eBooks Kindle', 'Libros en inglés', 'Libros en idiomas extranjeros', 'Libros infantiles y juveniles', and 'Libros de texto y universitarios'. The 'Libros' section title and its sub-categories are all enclosed in a large red box.

Llegamos a esta página que tiene como nombre "Libros"

This screenshot shows the 'Libros' page on Amazon.es. At the top, there's a header with 'Todos los departamentos' and other navigation links. The main content area features a large 'Libros' title, which is highlighted with a red box. Below this are sections for 'Ahora en Libros', 'Top 100 en Libros', and 'Libros a precios bajos', each with a list of sub-categories. The entire 'Libros' title and its surrounding area are enclosed in a large red box.

- La página tiene nombre.
- El nombre destaca.
- Coincide con el texto del enlace.

#### 3. Coherencia del diseño.

Si el diseño es coherente en todas las páginas del sitio, estamos diciendo al usuario que sigue en nuestra Web.

## 4. Marcar la posición en los menús.

[HOME](#)[NOVEDADES](#)[CATÁLOGO](#)[SOBRE NOSOTROS](#)[CONTACTO](#)[COLECCIONES](#)[DESCARGA DE CATALOGOS](#)[EBOOKS](#)

Poniendo un puntero junto a lo que se desea seleccionar	Cambiando el color del texto	Usando texto en negrita	Invirtiendo el color	Cambiando el color del botón
Sports Business ► Entertainment Politics	Sports Business Entertainment Politics	Sports Business Entertainment Politics	Sports Business Entertainment Politics	Sports Business Entertainment Politics

## 5. Migas de pan (Breadcrumbs)

- Nos indican **dónde estamos**.
- Proporcionan enlaces para **ir a las páginas de nivel superior**.
- No es necesario utilizarlas siempre, sólo si el sitio Web tiene muchos niveles.
- En el caso de que se utilicen, se suelen seguir estas reglas:
  - **Se sitúan en la parte superior.**
  - **Utilizar > entre niveles.**
  - Poner en **negrita** el último nombre de la lista. Si es el nombre de la página actual, no será un enlace.
  - Menos llamativas que el resto del contenido de la página.

## Prueba del maletero

¿Buena navegación?

- Selecciona al azar una página del sitio e imprímela.
- Sujétala y mantenla a distancia, con el brazo estirado, entorna los ojos.
- Tan rápido como puedas, identifica:
  - Identificador del sitio.
  - Nombre de la página.
  - Secciones (navegación global).
  - Navegación local. ¿Qué opciones tengo en este nivel?
  - Indicadores "Usted está aquí".
  - Búsqueda.



# 4. Principios de diseño de la interacción

Nielsen Norman Group

## NN/g Principals



Jakob Nielsen



Don Norman



Bruce "Tog"  
Tognazzini

*"The Guru of Web  
Page Usability"  
(New York Times)*

*"The Guru of  
Workable  
Technology"  
(Newsweek)*

*"The User Interface  
Guru" (Wired)*

Algunos principios de la interacción, según Bruce Tognazzini ("Tog"):

- 1. Consistencia.**
- 2. Anticipación** a las necesidades y deseos del usuario.
- 3. Autonomía** para el usuario.
- 4. Eficiencia** del usuario.
- 5. Interfaces explorables.**
- 6. Protección del trabajo del usuario.**
- 7. Uso de metáforas** que permitan comprender el funcionamiento del sitio Web.
- 8. Legibilidad.**



Bruce "Tog" Tognazzini

"The User Interface Guru" (Wired)

<http://asktog.com/atc/principles-of-interaction-design/>

## 1. Consistencia

- Consistencia: también se denomina a veces "coherencia".
- Cómo se consigue la consistencia:
  - Diseño consistente**. No cambia en cada página del sitio.
  - Navegación constante**.
  - Seguir las convenciones**. La consistencia más importante es la consistencia con las expectativas del usuario. No importa lo brillante que sea nuestra idea. Si los usuarios esperan otra cosa, y nuestra idea no ofrece una clara ventaja, tener en cuenta lo que esperan los usuarios.
  - No utilizar palabras distintas para hacer lo mismo**.
- La consistencia se traduce en **eficiencia** del usuario.



### 2. Anticipación a las necesidades y deseos del usuario

- Proporcionar al usuario toda la información y herramientas necesarias para cada paso del proceso.
- El sitio Web debe anticiparse a las necesidades y deseos del usuario.
- No esperes que el usuario deje la pantalla actual para buscar la información que necesita. La información debe estar a mano, y las herramientas necesarias, presentes y visibles.



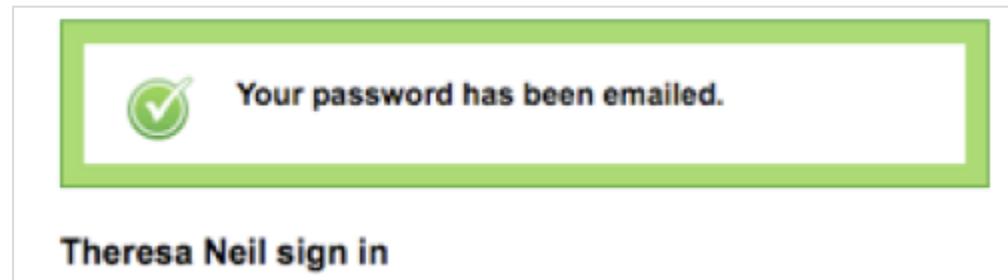
### 3. Autonomía para el usuario

- Para que haya autonomía, hace falta información. Por eso, el sistema debe mantener informado al usuario del estado del sistema. Además, la información del estado del sistema, se debe ver fácilmente y debe estar actualizada.

[UT01-Usabilidad-v0.ppt \(2.822 K\)](#)



Cargando un archivo en gmail.

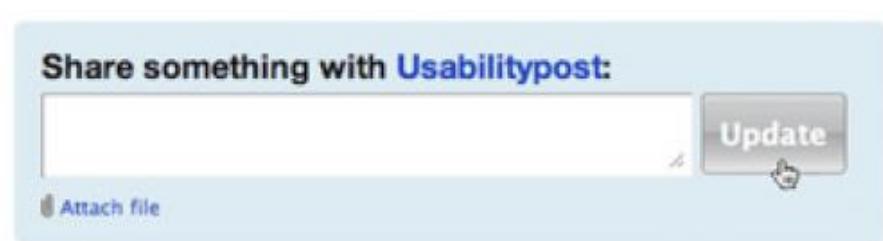


Type new password:  \* \* \* \* \* \* \* \*  
Six-characters minimum; case sensitive

Password strength: Strong

### 3. Autonomía para el usuario

- Prevención de errores. Ayudar al usuario a que no cometa errores.



#### Wikipedia

El autofocus en la entrada evita una común fuente de frustración, empezar a escribir antes de que el foco esté dentro del cuadro.

#### Yammer

Deshabilita el botón de actualizar después de hace clic en él, para que una persona no pueda actualizar dos veces el mismo post por error.

## 4. Eficiencia del usuario

- Ayudar al usuario a recuperarse de los errores
- Mensajes de error que ayuden:
  - Indicar dónde está el error.
  - Explicar al usuario cuál es la forma correcta de hacerlo.
  - Dejar abierta la posibilidad de que el mensaje se haya generado de forma errónea por un fallo del sistema.

### Ejemplo: formularios.

Proporciona información inmediata de los errores, con instrucciones específicas.

Or start a new account

Choose a username (no spaces)  
bert

Choose a password  
\*\*\*

Retype password

Email address (must be real)  
not an email

Send me occasional Digg updates.

⚠ bert is already taken. Please choose a different username.

⚠ Passwords must be at least 6 characters and can only contain letters and numbers.

⚠ The email provided does not appear to be valid.

9.0 Digg

### 4. Eficiencia del usuario

Ejemplo: Error 404: Página no encontrada.

Un texto claro que proporcione alternativas: ir a la página de inicio, un buscador.

También puede tener una imagen divertida.



Una de las páginas que salen cuando se da el error 404 en el Museo Reina Sofía.  
Idea tomada de: [http://www.eldiario.es/cultura/tecnologia/Not-found-enmarcar\\_0\\_559594492.html](http://www.eldiario.es/cultura/tecnologia/Not-found-enmarcar_0_559594492.html)

### 5. Interfaces explorables.

- Interfaces que se puedan recorrer fácilmente, que no te quedes atrapado sin poder volver atrás, que tengas la opción de deshacer...
- Navegación bien diseñada.
- Permitir el "deshacer". Mejor que la pregunta "realmente estás seguro de que quieres hacer esto"?

La conversación se ha enviado a la Papelera. [Más información](#) [Deshacer](#)

*Mensaje de gmail, cuando se elimina una conversación.*

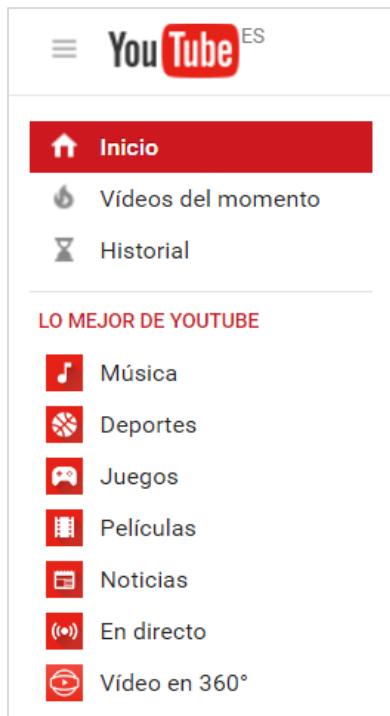
### 6. Protección del trabajo del usuario

- **Prioritario.**
- **Que el usuario nunca pierda su trabajo** como resultado de un error, los problemas de Internet u otro tipo de problemas inevitables, como un apagón.

Un ejemplo sería la carpeta de "Borradores" de gmail.

### 7. Uso de metáforas que permitan comprender el funcionamiento del sitio Web.

- Utilizar **metáforas**, que relacionen los sitios web con el mundo real.
- Utilizar palabras, imágenes, conceptos, que son **familiares al usuario**, más que palabras o conceptos técnicos.



### 8. Legibilidad.

- El texto debe tener suficiente **contraste** con el fondo.
- El **tamaño de letra**, se debe ver bien en las pantallas.
- Recogiendo los consejos de S. Krug:
  - Jerarquías visuales claras en cada página.
  - Texto fácil de leer:
    - Utilizar encabezados.
    - Párrafos cortos.
    - Utilizar viñetas.
    - Destacar los términos clave (pocos).

# 5. Análisis de la usabilidad

- En el caso de Diseño de Interfaces, el análisis de la usabilidad se utiliza para obtener retroalimentación que nos permita **mejorar el diseño**.
- El momento de hacerlo es... "cuanto antes".

## 1. Métodos para evaluar la usabilidad

- **Evaluación heurística:** Comprobar si la interfaz sigue unos **principios** de usabilidad. Ej. 10 principios heurísticos de Nielsen.  
<https://www.nngroup.com/articles/ten-usability-heuristics/>
- **Evaluación centrada en estándares:** Comprobar si la interfaz cumple uno o varios estándares. Ej. ISO/IEC 9241
- **Análisis de prototipos,** asumiendo el rol de usuarios.
- **Pruebas de usabilidad:** Probar la usabilidad de prototipos o de sitios Web estudiando cómo la utilizan los usuarios y analizando su experiencia.

## 2. Pruebas de usabilidad

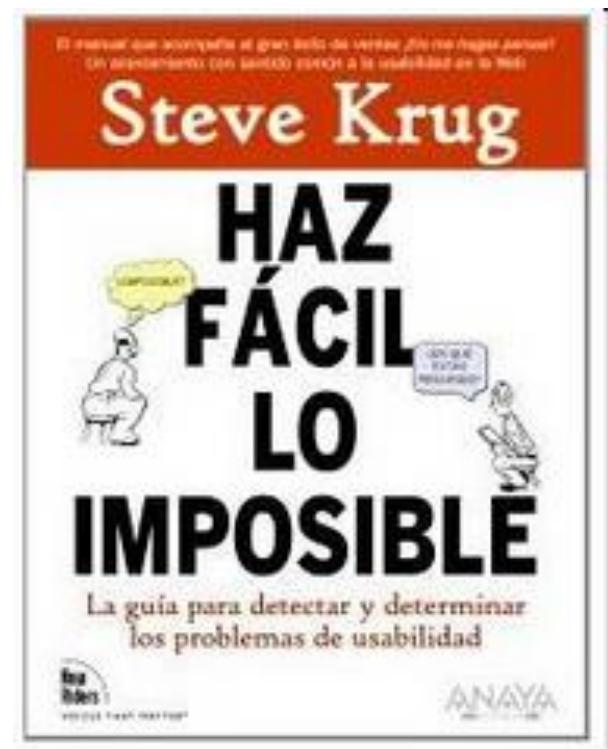


- Las pruebas de usabilidad básicamente consisten en **proponer a un usuario que realice una tarea** en un sitio Web.
- **El usuario va realizando la tarea** y comentando los problemas que se encuentra (o las cosas que le ayudan en su tarea).
- Una serie de **observadores** analizan lo que ha pasado en esa prueba, toman notas de problemas de usabilidad y proponen cambios en el sitio Web para mejorarla.
- Los observadores pueden grabar toda la sesión en vídeo, o simplemente tomar notas.

## 2. Pruebas de usabilidad

**Cómo hacer las pruebas.** Steve Krug: "Haz fácil lo imposible". Pruebas del tipo "hágalo usted mismo".

- **Una mañana al mes.** Con regularidad.
- **Empezar lo más pronto que crea que tiene sentido.** Desde los prototipos.
- No ser exigente en la **selección de "usuarios"**. Los errores serios los va a encontrar "casi cualquiera".
- Convertirlo en un **deporte de masas**. Involucrar a las personas de la empresa en las pruebas.
- Centrarse implacablemente en los **problemas más serios**. Siempre hay más problemas que recursos.
- Cuando solucione problemas, intente **resolver lo menos posible**. Cambios pequeños y sencillos.



### 3. Solucionar problemas de usabilidad

Una de las cosas que la gente tiene la tentación de hacer cuando hay un problema de usabilidad es **añadir algo**.

- Si alguien no ha entendido las instrucciones, **añada más instrucciones**.
- Si alguien no ha podido encontrar lo que estaba buscando en el texto, **añada más texto**.
- Si alguien no se ha dado cuenta de algo que necesitaba, **añádale más color, póngalo en negrita, o más grande**.

Pero muy a menudo la mejor forma de solucionar un problema de usabilidad es hacer justo lo contrario: **quitar algo**.

### 3. Herramientas

#### ■ **Herramientas que generan mapas de calor.**

- Estas herramientas hacen un mapa de colores en función del número de clics que han realizado los visitantes a estos sitios web.
- Estas herramientas se utilizan introduciendo código en la página web.
- También existen herramientas para hacer mapas de calor en función de dónde mira el usuario cuando visita el sitio web (eye-tracking).

<https://metrica.yandex.com/about/info/behavior#clickheatmaps>

#### ■ **Herramientas de grabación de pantallas.**

- Crean un vídeo donde se ve la actividad desarrollada por el usuario

#### ■ **Feedback.**

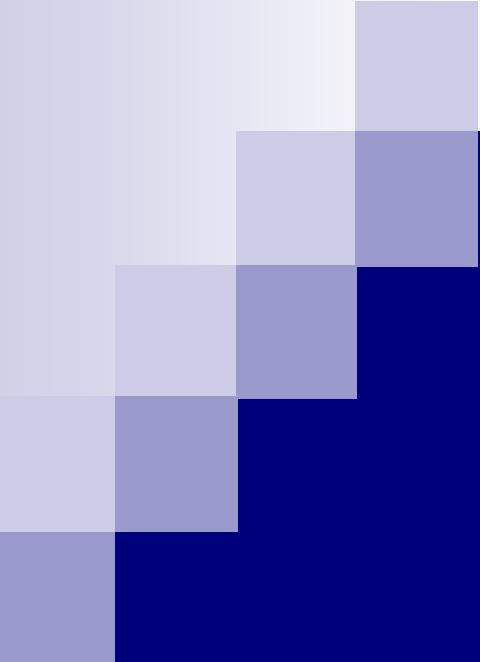
- No es propiamente una herramienta.
- Consiste en incluir en los sitios Web un modo de que los usuarios puedan enviar comentarios o sugerencias.

### 4. Análisis con distintos navegadores y dispositivos

- Qué comprobar del sitio web en distintos navegadores y dispositivos: que se visualiza correctamente y que funciona de forma adecuada.
- Herramientas en los propios navegadores:
  - Herramientas del desarrollador (F12).
  - Internet Explorer. Emulador de Windows Phone y de distintas versiones de Internet Explorer.
  - Chrome. Emulador de móviles y tablets.
- Otras herramientas:
  - Ejemplo: Emulador de OperaMobile.  
<https://dev.opera.com/articles/opera-mobile-emulator/>

# 6. Referencias

- KRUG, S. No me hagas pensar. Actualización. 3<sup>a</sup> ed. rev. y aum. Madrid: Anaya Multimedia, 2015.
- CARRERAS, Olga. Reseña "No me hagas pensar" y "Haz fácil lo imposible" de Steve Krug [en línea]. Disponible en: <http://olgacarreras.blogspot.com.es/2013/11/resena-no-me-hagas-pensar-y-haz-facil.html> [Consulta: 20/09/16]
- TOGNAZZINI, B. ("Tog"). First Principles of Interaction Design (Revised & Expanded) [en línea]. Disponible en: <http://asktog.com/atc/principles-of-interaction-design/> [Consulta: 20/09/16]



# UT1–2. Planificar interfaces Web I.

# Índice

1. Fases del diseño de un sitio Web
2. Planificación
  - 2.1. Investigación inicial
  - 2.2. Mapa del sitio
  - 2.3. Prototipos
  - 2.4. Componentes de una página web
  - 2.5. Diseño adaptativo
  - 2.6. Principios de diseño: cómo distribuir el contenido en una página

Referencias

# 1. Fases del diseño de un sitio Web

## ■ Planificación

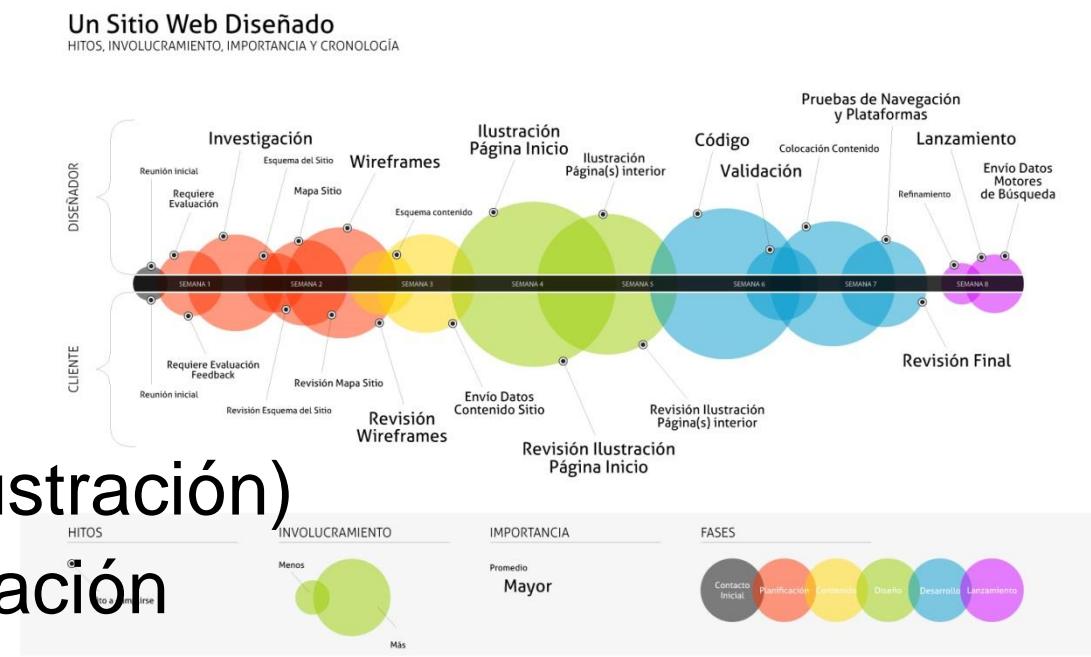
- Mapa del sitio
- Wireframes

## ■ Contenido

## ■ Diseño gráfico (ilustración)

## ■ Desarrollo: codificación

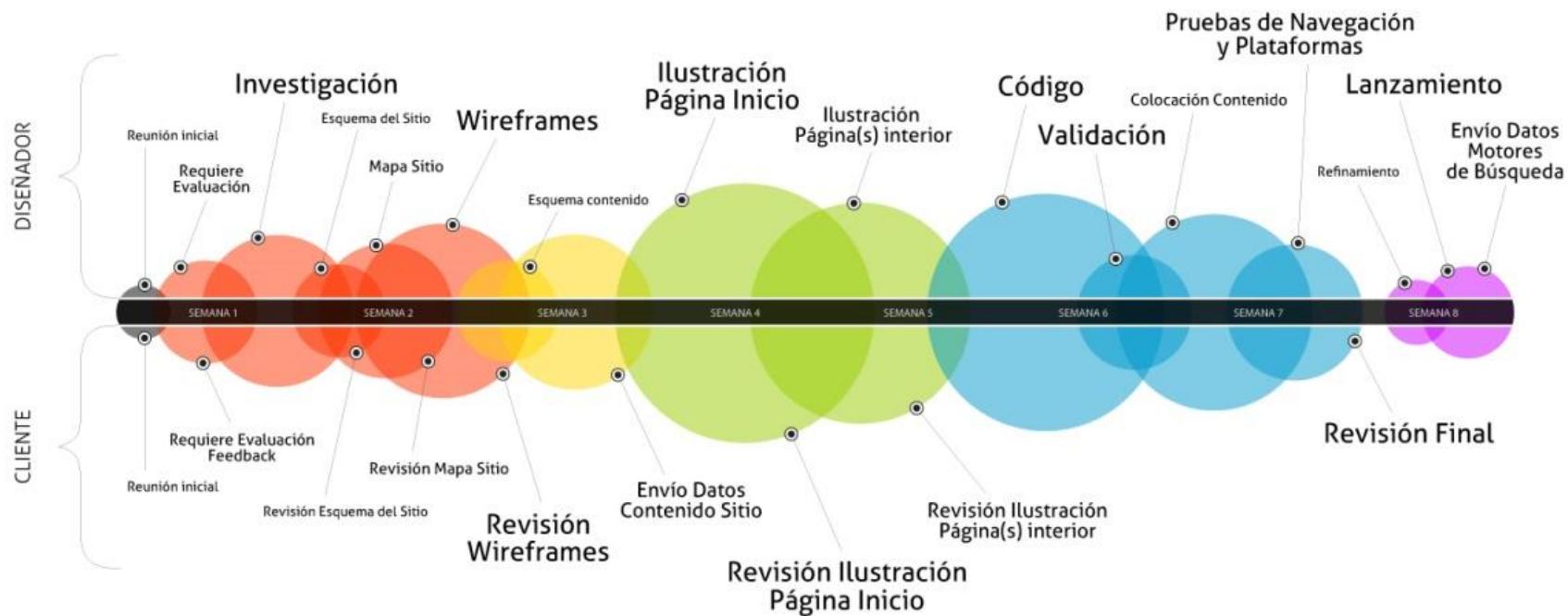
## ■ Lanzamiento



# 1. Fases del diseño de un sitio

## Un Sitio Web Diseñado

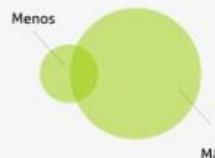
HITOS, INVOLUCRAMIENTO, IMPORTANCIA Y CRONOLOGÍA



## HITOS



## INVOLUCRAMIENTO



## IMPORTANCIA

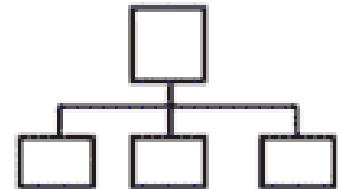
Promedio  
Mayor

## FASES



# 2.1. Investigación inicial

- Antes de crear el sitio web, hay que realizar una investigación, **para definir con precisión el sitio Web a construir.**
- **Técnicas**
  - *Card sorting* (ordenar tarjetas)
  - Tormenta de ideas
  - Diagramas...
  - Estas técnicas permiten que el **usuario participe en el desarrollo**, verificando si la solución que se propone es correcta o no.



## 2.2. Mapa del sitio

- El objetivo es **representar las secciones** en que está dividido un sitio web y las **relaciones entre los contenidos**.
- Se suelen representar con textos, cajas y flechas.
- Se puede hacer a mano o utilizando cualquier herramienta que permita hacer diagramas (ej. Microsoft Visio).

## 2.3. Prototipos

- También se llaman: maquetas, mockups, ...
- **Objetivo** es definir los interfaces.
- **Tipos** de prototipos:
  - Según el **nivel de detalle**:
    - de **baja fidelidad**: (sketches, wireframes): sin elementos gráficos, sin detallar color ni tipografía
    - de **fidelidad intermedia**: incorporan elementos gráficos, color, tipografía
    - de **alta fidelidad**: muy parecidos a la aplicación final
  - **Dinámicos o no dinámicos**: según permitan interactuar con ellos.

# Prototipos de baja fidelidad

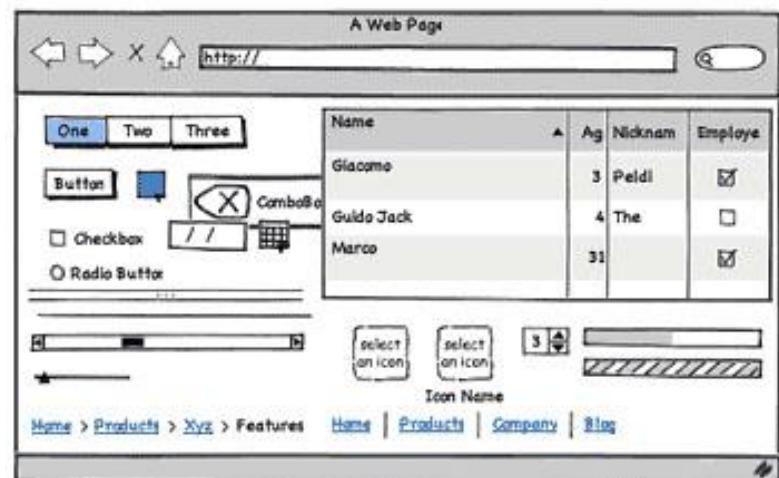
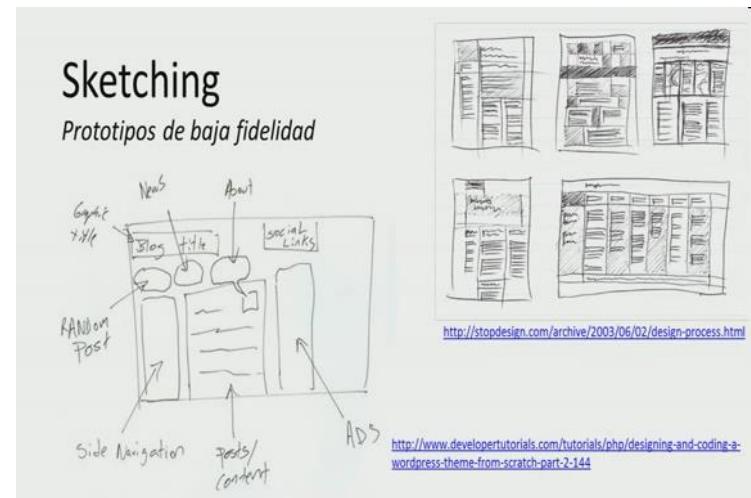
## Características:

- En **blanco y negro/grises**
- Misma **tipografía** para todo
- **No imágenes** reales
- **No texto** real
- **Trazos** simples

**Prototipos  
de  
baja fidelidad**

**Sketches**  
simulan dibujo a mano

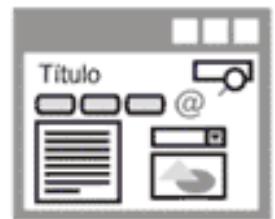
**Wireframes**  
trazos simples



# Prototipos de baja fidelidad

## Qué definir en el prototipo de cada interfaz

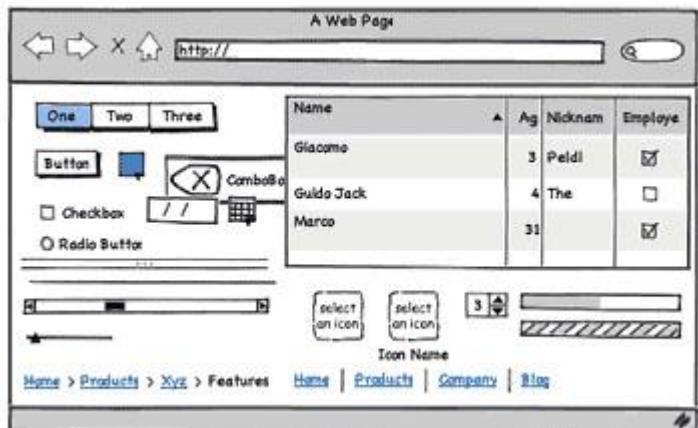
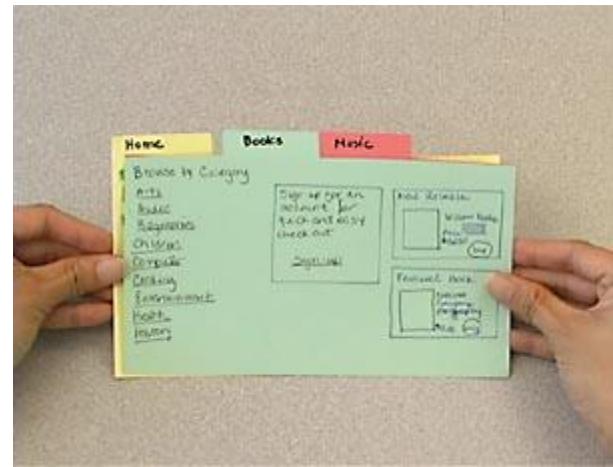
- **Elementos:**
  - **Componentes** que están presentes en la misma
  - **Layout** (distribución de los contenidos)
  - Estrategia de **navegación**
  - **Jerarquía** de los contenidos.
  - **Etiquetado** (texto) de los enlaces y de los títulos
- **Funcionamiento** de la interfaz, por ejemplo, mediante notas asociadas a los elementos.



*No todos los diseñadores son partidarios de estos prototipos.*

# Cómo hacer prototipos

- Se puede utilizar papel y lápiz, una pizarra...
- **Paper prototyping:** consiste en hacer maquetas con papel, lápiz y otros elementos (tijeras, etc.). Es una técnica sencilla utilizada para centrar el proceso de diseño en el usuario y para probar interfaces de usuario.
- **Software** para crear prototipos: es más fácil modificar los prototipos y que el cliente lo pruebe.



# Cómo hacer prototipos

## ■ Software de prototipado

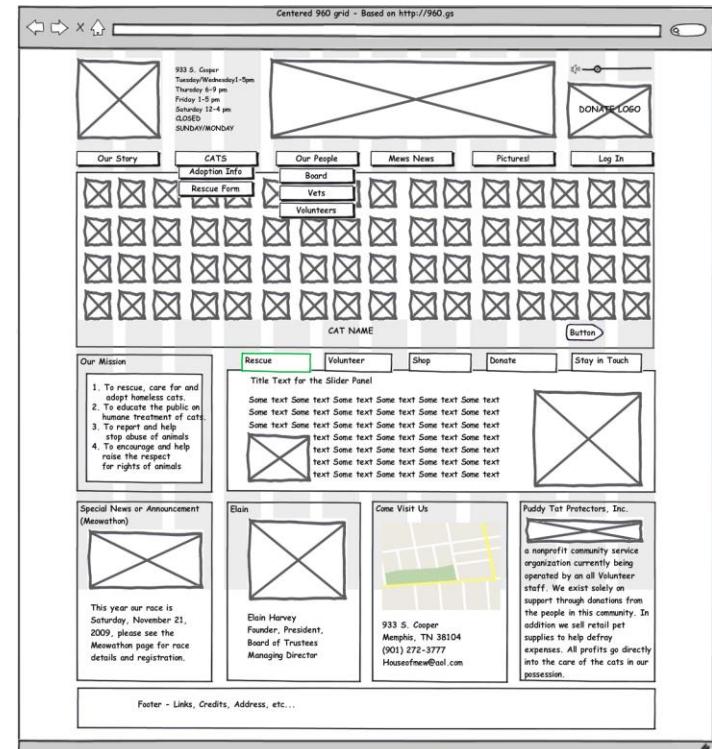
- [Balsamiq](#)
- [Axure](#)
- [Gliffy](#)
- [Pencil](#)
- [Microsoft SketchFlow](#)
- [Flairbuilder](#)
- [iPLOTZ](#)
- [Mockups](#)
- [Gomockingbird](#)
- [proto.io](#)
- [Adobe XD](#)

## ■ Otras herramientas

- [960 grid](#): es un conjunto de plantillas para hacer prototipos pensando en un sistema de 960 px.

# 2.4. Componentes de una página Web

- **Cabecera**
- **Sistema de navegación** (puede estar incluido en la cabecera)
  - **Sistema de navegación global** para acceder a las páginas principales.
  - **Sistema de navegación local** para acceder a las páginas secundarias.
- **Cuerpo de la página**
- **Pie de página**
  
- **Espacio en blanco**
  - No es necesario llenar todo el espacio



# Cabecera

- Elementos: navegación constante
- Puede haber páginas sin cabecera en un sitio Web, ej.
  - Página de inicio
  - Formularios
  - ...

### Sistema de navegación: ¿izquierda o derecha?

- Los usuarios **se adaptan** rápidamente a la navegación con la columna a la derecha o a la izquierda. Lo único que esperan es que el modo de colocar la navegación en todas las páginas sea **consistente**.
- **Si se coloca a la izquierda**
  - Normalmente, ahí va el sistema de navegación.
  - Ventaja en Usabilidad: los usuarios están acostumbrados a que estén aquí.
- **Si se coloca a la derecha**
  - Normalmente, ahí van enlaces externos, enlaces relacionados o anuncios.
  - Usabilidad: los usuarios esperan anuncios. Si se quiere utilizar para enlaces de navegación, su aspecto debe ser muy diferente a los anuncios o será ignorada por la mayoría de los usuarios.

### Cuerpo de la página

- Es donde se presenta al usuario la **información** referente a los contenidos de la página.
- Ocupará el **espacio mayor de la página**, normalmente entre un 50% y 85% del total.
- Su **ubicación** siempre es **central**, debajo de la cabecera (si la hay) y al lado del menú lateral de navegación (si lo hay).
- Normalmente, el cuerpo de la página llevará un **título** que identifique a la página, y que debe resaltar respecto al resto del texto de la página: con mayor tamaño, o con un color diferente.

### Pie de página

- Contenido:
  - **copyright**, información de **contacto**, **información legal**, algunos **enlaces** a las principales secciones del sitio.
  - puede contener un **menú auxiliar**.
- Se debe **distinguir bien** el bloque de pie de página del bloque de contenido para indicar al usuario claramente que se encuentra al final de la página.

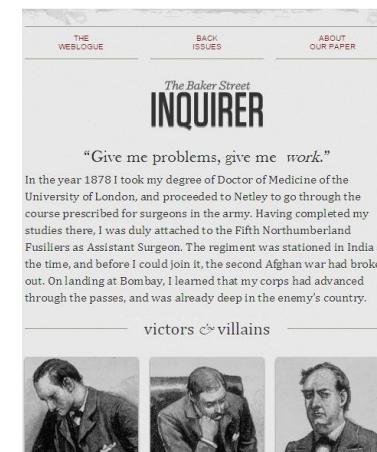
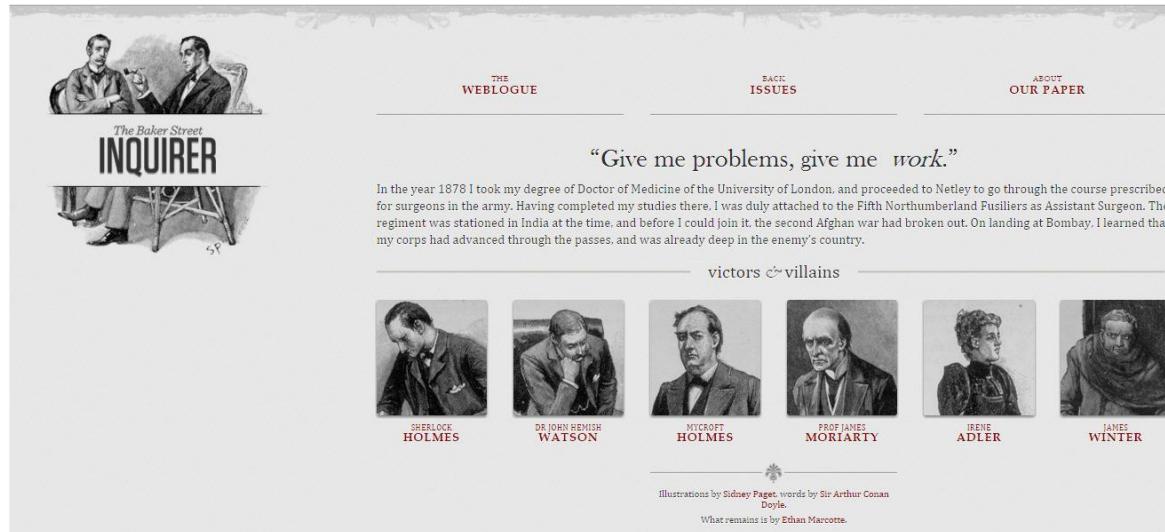
# 2.5. Diseño adaptativo

## ■ Qué es.

- También llamado: “*Responsive design*” o diseño responsivo.
- El fundamento del diseño adaptativo es que la página sea capaz de adaptarse al ancho del dispositivo.
- El contenido de la página (imágenes y texto) es siempre el mismo, lo que cambia es cómo se presenta (CSS).

## ■ Prototipado y diseño adaptativo:

- Al hacer el prototipo de las páginas web, hay que tener en cuenta los distintos anchos de dispositivos.
- **Estrategia "mobile first":** empezar diseñando para móviles.



Cómo se consigue:

- Columnas de ancho flexible %
- Tamaño de las imágenes flexible %
- Diferente distribución de las capas

## 2.6. Principios de diseño: cómo distribuir el contenido en una página

- **Jerarquía visual:** qué es más importante.
- **Flujo visual:** qué debería ver a continuación.
- **Agrupar y alinear:** qué va con qué.

# Jerarquía visual

- Los **títulos** deberían parecer títulos y el **contenido secundario** debe parecer contenido secundario.
- Un lector debe poder:
  - Entender la **estructura de la información** a partir de la estructura de la página
  - **Ver lo importante** con un golpe de vista
- ¿Cómo?
  - **Fuentes de mayor tamaño** o **negrita** para lo más importante.
  - **Espacios en blanco** para separar la información.
  - **Indentar el texto**: el texto indentado está subordinado al que está encima.

# Jerarquía visual

Screenshot of the BBC News website illustrating visual hierarchy:

- Header:** BBC logo, Sign in button, navigation menu (Home, News, Sport, Weather, Shop, Earth, Travel, Capital, More), Search bar.
- Section Header:** NEWS banner.
- Main Article:**
  - Title:** Police search home over London Tube bomb.
  - Image:** A photograph of police officers at a crime scene.
  - Text:** Police arrest an 18-year-old man on suspicion of a terror offence before armed officers search a house in Surrey.
  - Time/Categories:** 5m | UK.
  - Summary:** Passengers describe panic and 'stampede'.
  - Summary:** Witness describes 'smell of burning'.
- Live Updates Box:** LIVELatest updates on hunt for attacker
  - 3m Suspect arrested at Dover port
  - 40m Terror threat level remains at critical
  - 1h Woman tells of evacuation rush
- Thumbnail Grid:** Four news items with images and titles:
  - Bangladesh to restrict Rohingya movement
  - Ryanair to cancel 40-50 flights per day
  - Flying into Syria with the Russian Army
  - DR Congo forces kill Burundi 'refugees'
- Bottom News Items:**
  - Slender Man stabbing girl 'mentally ill'
  - Catalan separatists rally around mayors
  - North Korea vows to complete nuclear plan

# Flujo visual

- Es el recorrido que los ojos de los usuarios tienden a hacer cuando miran una página.
- A la hora de diseñar la página, tener en cuenta **cómo miran los usuarios**:
  - Tener en cuenta que el **tipo de la página** influye en cómo miramos una página: ej. página de tipo "periódico" o de tipo "ventas".
  - Normalmente miramos de **arriba a abajo y de izquierda a derecha**: (en algunos idiomas).
  - **Puntos fuertes**, que resaltan por tamaño, color, contraste...: atraerán primero la atención, seguidos de los más débiles.

# Flujo visual



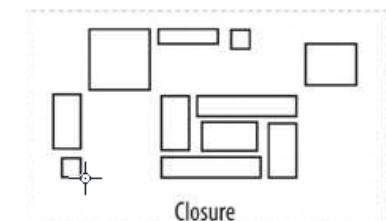
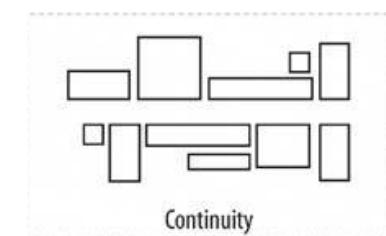
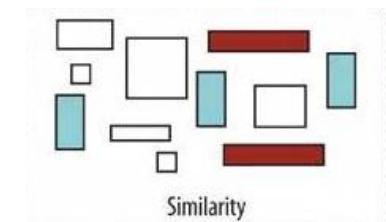
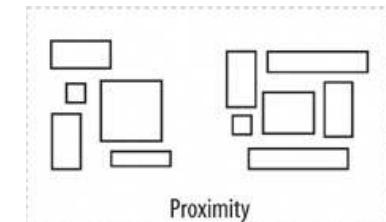
Los puntos fuertes (DONATE) atraen la vista (tamaño, contraste con el fondo, colocación con flechas alrededor).

# Agrupar visualmente

- **Agrupar visualmente** cosas equivale a indicar que esas cosas están *relacionadas*. Y al contrario, si se colocan las cosas muy lejos, se está diciendo que no están relacionadas.
- Por eso, se intenta **agrupar visualmente las cosas relacionadas** y separar las que no están relacionadas. Ej: asociar un campo de texto con su etiqueta; una imagen con su título; un gráfico con la barra que lo controla.

# Cómo agrupar visualmente

- **PROXIMIDAD:** Los elementos que están cerca se consideran relacionados
- **SIMILITUD:** Los elementos que tienen características visuales (forma, color, orientación) similares, se consideran relacionados
- **CONTINUIDAD:** Nuestros ojos tienden a ver líneas y curvas formadas por la alineación de elementos pequeños.
- **CIERRE:** Tendemos a ver formas simples cerradas, como rectángulos, aunque no estén explícitamente dibujadas.



## 2. Planificación > 2.6. Principios de diseño: distribuir el contenido

### Ejemplo:

- proximidad
- similitud
- continuidad
- cierre

The screenshot shows the homepage of [A Book Apart](http://abookapart.com/). The main focus is the book cover for 'CSS3 FOR WEB DESIGNERS' by Dan Cederholm, featuring a large green background and white text. Below the book cover, there's a section titled 'ALSO FROM A BOOK APART' displaying other book covers and gift card options.

**A Book Apart**  
Brief books for people who make websites

HOME STORE PRESS ABOUT HELP

0

Dan Cederholm  
**CSS3 FOR WEB DESIGNERS**  
foreword by JEFFREY ZELDMAN

SECOND EDITION

Rediscover a universe of creative possibilities with CSS3. Dan Cederholm updates his essential guide with new examples, polished code, and a new chapter on micro layouts.

Available in paperback, ePub, PDF, and mobi

**BUY NOW**

ALSO FROM A BOOK APART

**Gift cards**  
\$25, \$50, and \$100  
**BUY**

**Responsive Web Design**  
by ETHAN MARCOTTE  
**BUY**

**Responsive Responsive Design**  
by SCOTT JEHL  
**BUY**

**You're My Favorite Client**  
by MIKE MONTEIRO  
**BUY**

**On Web Typography**  
by JASON SANTA MARIA  
**BUY**

**Sass For Web Designers**  
by DAN CEDERHOLM  
**BUY**

<http://abookapart.com/>

# Referencias

- Video [¿Qué es el desarrollo web?](#) (del curso iDesWeb de la Universidad de Alicante). Explica las tareas que se realizan en la construcción de un sitio web, diferenciando las que realizan los desarrolladores web (más técnicas) y las que realizan los diseñadores web (más estéticas o visuales).
- Video [Diseño de una aplicación web](#). Sobre los tipos de diagramas: mapas de arquitectura y prototipos.
- [La diagramación en la arquitectura de información](#). Artículo de nosolousabilidad.com que explica la función de los diagramas (mapas de navegación, prototipos) en el desarrollo web.
- [Wiki del Web Education Community Group, de W3C](#) (en inglés, algunos documentos están traducidos al castellano) Contiene recursos para ayudar a enseñar o aprender desarrollo web moderno.
- [Arquitectura de la información: planificación de una web](#) (del curso Conceptos de diseño web, de la UOC, Universitat Oberta de Catalunya).
- [Web Style Guide Online 3<sup>a</sup> ed](#) (en inglés). Versión online del libro del mismo nombre.



# UT1–2. Planificar Interfaces Web II.

# Índice

## 1. El color

- 1.1. Modelos aditivo y sustractivo
- 1.2. Vocabulario sobre el color
- 1.3. Codificación del color
- 1.4. Elegir colores para la web
- 1.5. Color y legibilidad

## 2. Tipografía

- 2.1. Tipos de fuentes
- 2.2. Elegir fuentes para la web
- 2.3. Tipografía y legibilidad

## 3. Guías de estilo

## Referencias

# 1. El color

- Importante por:
  - **Usabilidad:** puede facilitar o dificultar el uso del interfaz.
  - **Estética.**
  - **Herramienta de comunicación.**

# 1.1. Modelos aditivo y sustractivo

- Se pueden obtener colores:
  - **Modelo aditivo**: mezclando luces que suman distintas longitudes de onda.
  - **Modelo sustractivo**: mezclando pigmentos que restan distintas longitudes de onda.

# Modelo aditivo



- Es la mezcla que se produce al combinar focos de luz (teatro) o luz generada por monitores.
- La suma de colores aumenta la intensidad de la luz y nos acerca al blanco.
- Los **colores primarios** son el Rojo, Verde y Azul (RGB).

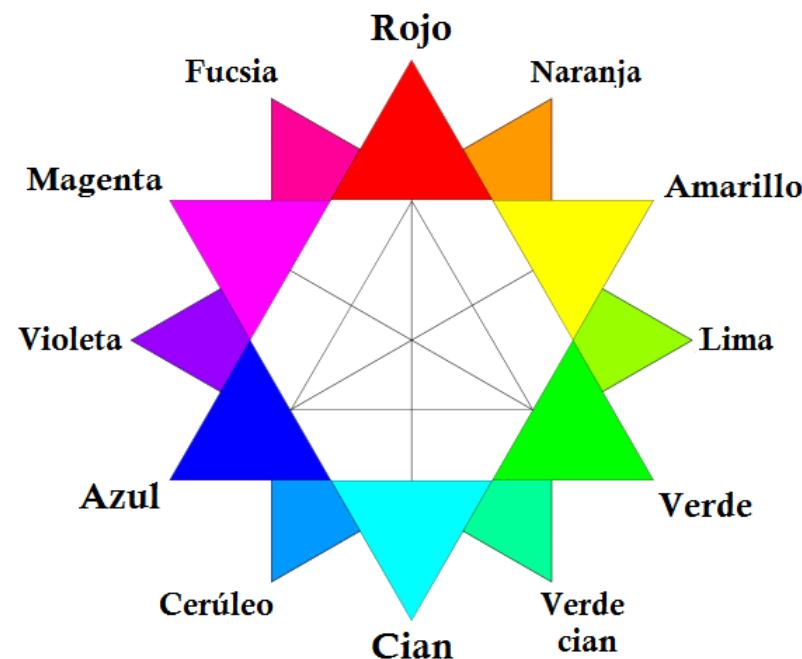
# Modelo sustractivo



- Es la mezcla resultante de la combinación de pigmentos. Se produce en la mezcla de pinturas y en la mezcla de tintas de impresora.
- Los pigmentos tienen la capacidad de sustraer parte de la luz.
- En la mezcla sustractiva, la suma de pigmentos de color distintos da un color más oscuro (cercano al negro).
- Los **colores primarios** son el Cyan, Magenta y Amarillo (CMY).
- Se suele utilizar un cartucho para cada uno de esos colores, y otro más para el negro (CMYK).

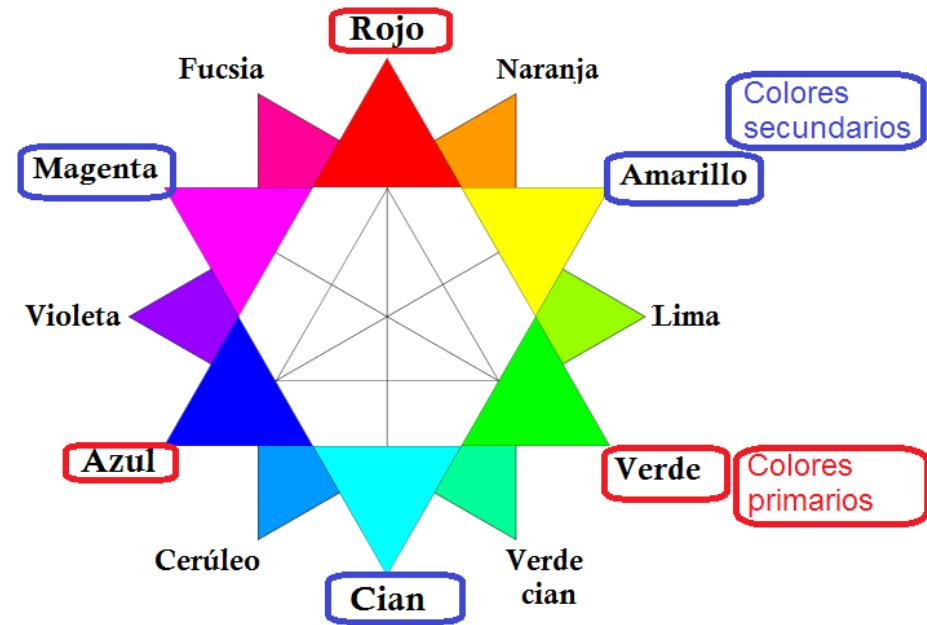
# 1.2. Vocabulario sobre el color

- **El círculo cromático o rueda de colores**, es una herramienta que se utiliza para representar los colores y su relación entre ellos; así como para facilitar la combinación de colores.



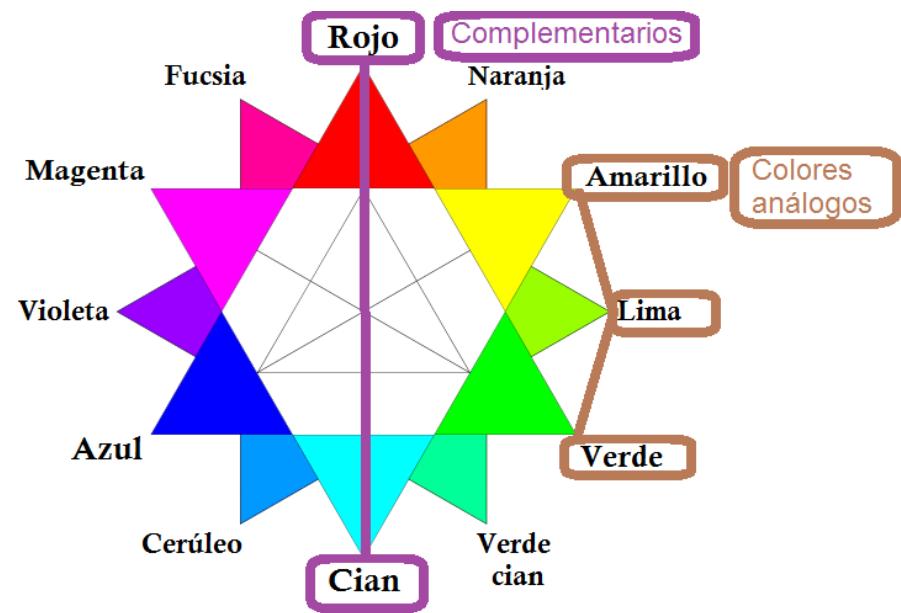
**Círculo cromático** para el modelo  
De síntesis aditiva

- **Colores primarios:** colores a partir de los cuales se generan los otros colores. Por ejemplo: Rojo, Verde y Azul.
- **Colores secundarios:** los que se obtienen combinando dos colores primarios.

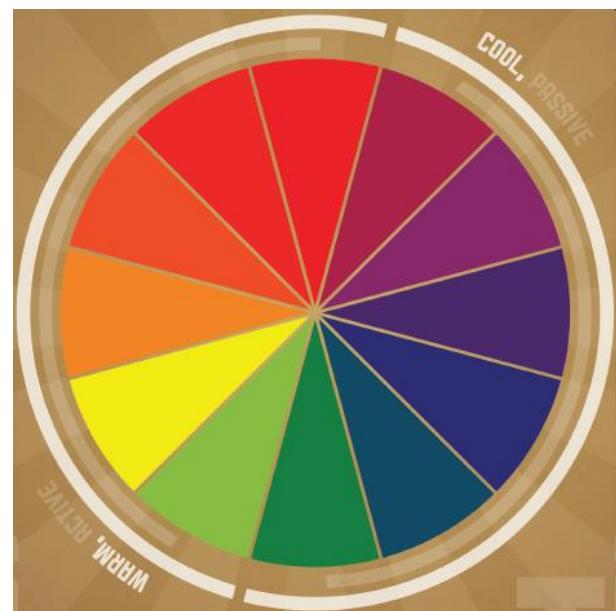


Colores primarios y secundarios en el modelo aditivo.

- **Colores complementarios:** dos colores complementarios están en lugares opuestos en el círculo cromático. Entre ellos se da el máximo contraste.
- **Colores análogos:** Son colores que están cerca en el círculo cromático.  
Los colores se relacionan por algún tono común a todos ellos.



- Los **colores cálidos y fríos** se definen tomando como base el círculo cromático tradicional (donde los colores primarios son el amarillo, azul y rojo).
- La división en colores cálidos y fríos es **subjetiva**: radica en la sensación y experiencia humana.
- **Colores cálidos:** van del rojo al amarillo
  - Producen un efecto alegre, vivo, estimulante
  - Tienden a **sobresalir**.
- **Colores fríos:** van del azul al verde.
  - Producen un efecto tranquilo, sedante, silencioso, frío, a veces triste.
  - **No sobresalen**, no abruman



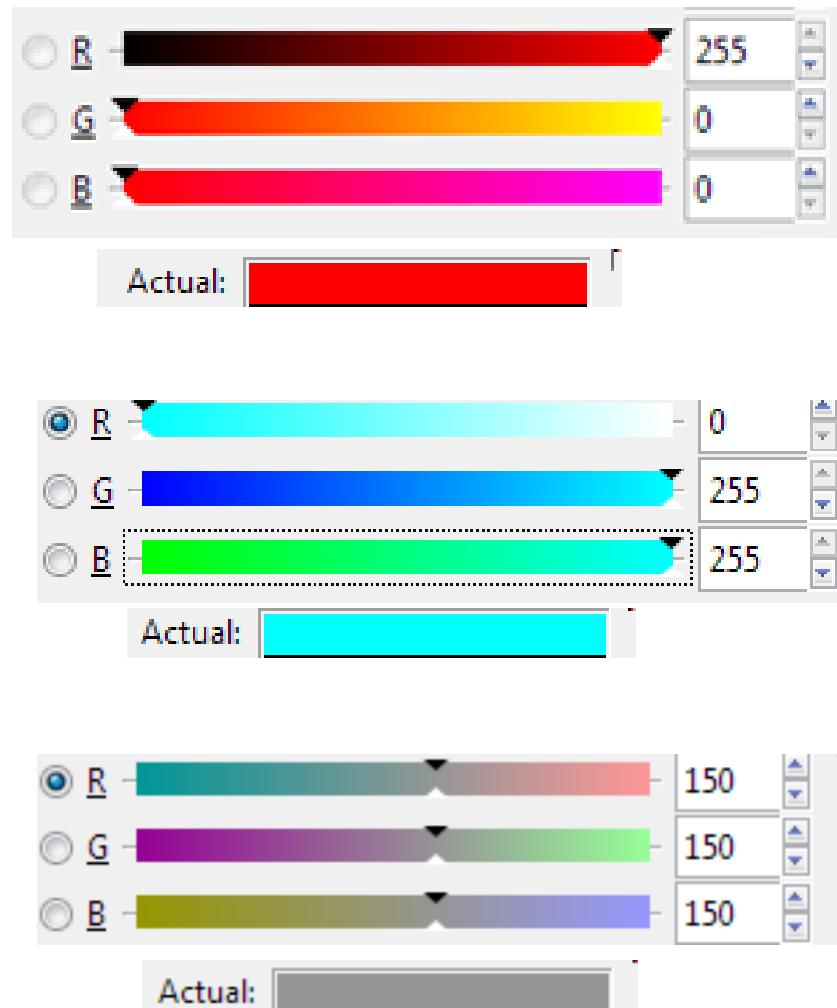
# 1.3. Codificación del color

- Para utilizar el color en un sistema informático, necesitamos "codificarlos", asignarles un número. Existen varios métodos, que a veces se llaman modos de color.
- Hay varios modos de representar los colores:
  - **Modo RGB**: es el método utilizado por los monitores CRT, está orientado al hardware.
  - **Modo CMKY**: es el método utilizado por las impresoras; también orientado al hardware, no utilizado para la web.
  - **Modo HSV**: utilizado por algunos programas, como GIMP o Paint.net.
  - **Modo HSL**: utilizado en CSS3.
- Los modos HSV y HSL son **más intuitivos** que el modo RGB. Por ejemplo, podemos quitar brillo a un color, sin que cambie de tono.

# Modo RGB

Corresponde al *modelo aditivo*:

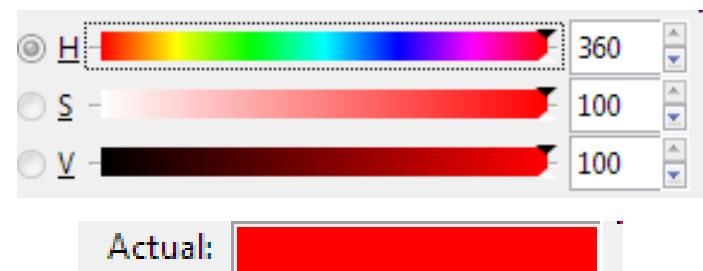
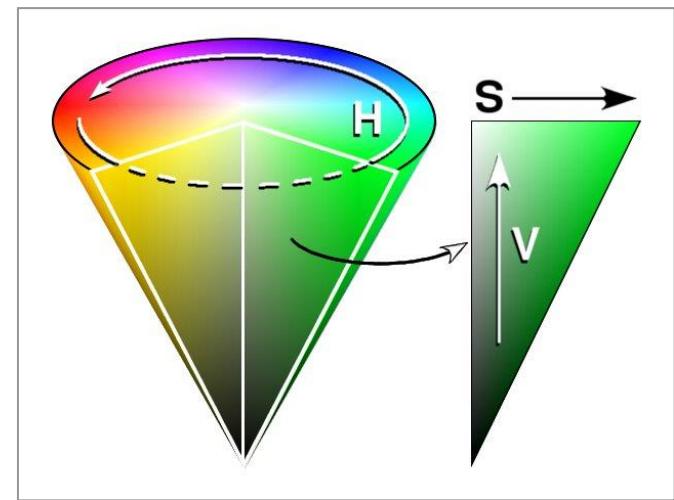
- Red-Green-Blue (Rojo-Verde-Azul)
- Los colores se representan con **tres valores numéricos** que indican la intensidad del verde, el rojo y el azul (**RGB**).
- Normalmente se utiliza una escala de **0 a 255** para indicar la intensidad de cada uno de los colores primarios.
- Si los tres colores son iguales, el color obtenido estará entre blanco (255,255,255) y negro (0,0,0), pasando por la escala de grises.



# Modo HSV (GIMP, Paint.net)

En este modelo, se utilizan tres propiedades del color:

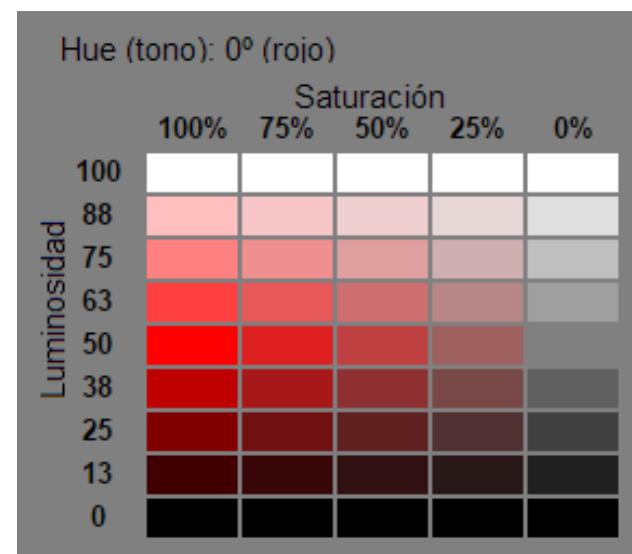
- **Tono o matiz (hue):** es el color. Valores que toma: de 0 a 360º.
- **Saturación:** es el nivel de *pureza de un color*. Un color se puede saturar mezclando blanco, negro, gris o un color complementario. En este caso, se mezcla con blanco. 100% es el color totalmente saturado, y 0% es blanco.
- **Valor (valor o value):** Indica el *grado de luminosidad de un color* (la cantidad de luz que emite). 100% es el valor normal; 0% es negro.



# Modo HSL (CSS3)

En este modelo, se utilizan tres propiedades del color:

- **H: Tono o matiz (hue):** es el color. Valores que toma: de 0 a 360º.
- **S: Saturación:** es el nivel de *pureza de un color*. 100% es total saturación y 0% una sombra de gris.
- **L: Luminosidad:** Indica el grado de *luminosidad de un color*. 100% es blanco, 0% es negro y 50% es normal.



# ¿Colores seguros para Internet?

- El color en las páginas web ha estado limitado a una paleta formada por **216 colores**, llamados "**colores seguros**", que era utilizada cuando los usuarios tenían monitores que reconocían 256 colores.
- Los colores seguros eran los que cumplían estas **características**:
  - Dígitos múltiplos de 3: 0, 3, 6, 9, C (12), F (15)
  - Además, los dos dígitos de cada par de dígitos hexadecimales son iguales.
  - Ejemplo: #0033FF
- Sin embargo, actualmente, los monitores pueden mostrar miles o millones de colores, por lo que no tiene sentido limitarse a ese conjunto de colores.

# 1.4. Elegir colores para la Web

## ■ **Inspiración:**

- Colores corporativos
- Otras páginas Web
- Fotografías del lugar
- "Significado" del color: psicología del color
- ...

## ■ **Cuántos colores:** Entre tres y cinco colores principales del sitio Web.

# Herramientas para generar la paleta de colores

- [Crear esquemas de colores \(Adobe\)](#) (castellano). Herramienta para crear o buscar paletas de colores. Se puede crear con la rueda cromática (aplicando algún esquema o de forma libre) o a partir de una imagen.
- [Paletton.com](#). Otra herramienta para crear paletas de colores aplicando un esquema.
- [Color Combos](#). Conjunto de herramientas para probar combinaciones de colores.
- [Color hunter](#). Para elegir paletas de colores ya creadas, o crear la propia paleta de colores a partir de una fotografía.
- [ColorHexa](#). Es una herramienta que proporciona información sobre cualquier color, y genera varias paletas de colores. También informa sobre cómo ven ese color personas con ciertas enfermedades.
- [ColorPix](#). Herramienta para obtener el color de cualquier punto de la pantalla.
- **Firefox > Desarrollador web > Recogecolor**. Permite obtener el color de un pixel.
  
- Paletas de colores ya creadas:
  - [COLRD](#)
  - [ColourLovers](#)

[http://cv.uoc.edu/annotation/2a8c76657e215adb7c99901a3020ebbe/505423/PID\\_00216991/modul\\_4.html#w26aac11b7c11](http://cv.uoc.edu/annotation/2a8c76657e215adb7c99901a3020ebbe/505423/PID_00216991/modul_4.html#w26aac11b7c11)

# Generar la paleta de colores

- Algunos esquemas que se utilizan:
  - Colores análogos
  - Esquema monocromático
  - Tríadas de colores
  - Colores complementarios
- Para obtener estos esquemas de colores se utiliza la rueda de color.

# Esquemas de colores

## Colores análogos

- Se forman a partir de varios colores adyacentes de la rueda de colores.
- Suelen ser armoniosos. Fáciles de usar.



## Monocromático

- Se forma con variaciones de un único tono.
- Uno de los más fáciles de usar.
- Suelen tener un aspecto unificado, armonioso y profesional... Aunque está el peligro de ser aburridos.



# Esquemas de colores

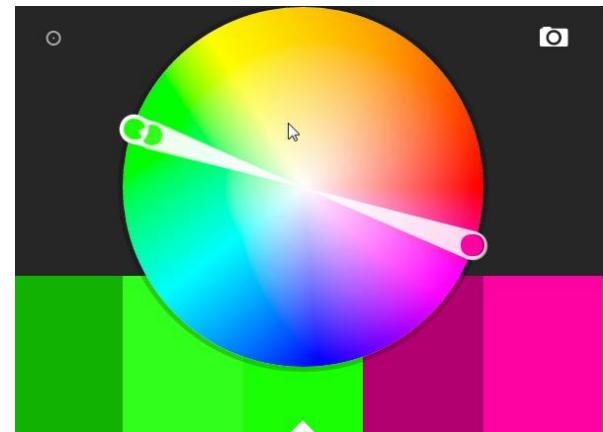
## Triada de colores

- Utiliza **tres colores separados uniformemente**.
- Combinación colores llamativas y vibrantes, aunque no tanto como los colores complementarios.
- La triada más utilizada es la formada por los tres colores primarios tradicionales: amarillo, azul y rojo.



## Complementarios

- Utiliza dos **colores complementarios**.
- Esquemas de colores llamativos, creando sensación de movimiento. Se utilizan en equipos deportivos.
- Se pueden atenuar los tonos para que haga otro efecto.



# 1.5. Color y legibilidad

- Al elegir el color del fondo y del texto lo más importante es la **legibilidad**.
- Algunas pautas:
  - **Suficiente contraste entre el fondo y el texto.** Ej. fondos oscuros y texto claro o viceversa.
  - Tener en cuenta que muchas personas no distinguen bien **el rojo del verde.**
  - **El color del texto y el del fondo no deben ser complementarios** porque cuesta leerlos. Ej. no poner texto azul sobre fondo amarillo o naranja
  - **Fondo.** Sin **texturas** o con texturas que no interfieran con el texto.
  - Mejor Colores **poco brillantes**, que pueden ser molestos si se utilizan para el texto o para el fondo.

# 2. Tipografía

- La función principal de la tipografía es su utilidad: que permita leer bien las palabras, es decir, que facilite la **legibilidad**.
- Además, la propia tipografía sirve como herramienta de **comunicación**. Por ejemplo, se puede utilizar para **decorar** o **llamar la atención**. Así.
- Hay que considerar también el **aspecto estético** de la tipografía empleada. Para muchos diseñadores es básico.
- Se debe utilizar una **tipografía apropiada** al mensaje que se quiere transmitir en el sitio Web. Por ejemplo, en general, no sería apropiada una tipografía "comic" en una gran empresa.

## 2.1. Tipos de fuentes

- Las fuentes se pueden agrupar en distintas categorías:
  - Serif y Sans serif
  - **Monoespaciadas** y de espaciado proporcional
  - **DECORATIVAS** y caligráficas
  - Dingbat ( dingbat )

# Fuentes Serif y Sans Serif

## Serif

- Es un tipo de letra **con remates** en los extremos.
- **En textos impresos** se utiliza mucho para bloques grandes de texto, normalmente anchos, al proporcionar una línea horizontal de referencia.
- **En la Web** se encuentran sobre todo en los **títulos y textos cortos**.
- Algunas fuentes Serif:
  - Times New Roman
  - Georgia
  - Courier New
- En CSS existe una fuente genérica llamada **serif**.

## Sans Serif

- Es un tipo de letra **sin remates** en los extremos.
- **En la Web** se suele utilizar para el **cuerpo** de la página.
- Algunas fuentes Sans Serif:
  - Helvética
  - Arial
  - Verdana
- En CSS existe una fuente genérica **sans-serif**.

\* *Serif: podría ser traducido como “remate”*

# Fuentes monoespaciadas y de espaciado proporcional

## Monoespaciadas

- Fuentes en las que todos los caracteres ocupan el mismo espacio.
- Se suelen utilizar para presentar **código de programación** porque permiten ver claramente el código.
- Algunas fuentes monoespaciadas:
  - Courier New
  - Liberation Mono
  - Lucida Console
- En CSS existe una fuente genérica llamada **monospace**.

## De espaciado proporcional

- Fuentes en las que los caracteres ocupan diferente espacio horizontal.
- La mayoría de las fuentes son de este tipo.

# Fuentes decorativas y caligráficas

## DECORATIVAS Y CALIGRÁFICAS

- Hay cientos de fuentes decorativas y caligráficas.
- En general, son **difíciles de leer** por lo que no se utilizan. Pueden aparecer en títulos y textos cortos.
- Se aconseja utilizarlas **con moderación**.
- Pueden dar personalidad y estilo al diseño.
- Algunas fuentes decorativas o caligráficas:
  - blade runner movie
  - ALGERIAN
  - Comic MS
- En CSS existen las fuentes genéricas **cursive** y **fantasy**.

# Fuentes dingbat\*



- Fuentes compuestas por símbolos y formas en lugar de por letras, números, etc.
- Pueden ser útiles para iconos.
- Algunas fuentes dingbat:
  - Wingdings: Fuente Wingdings
  - Webdings: Fuente Webdings

\* *Ding: persona tonta; florituras*

# 2.2. Elegir fuentes para la Web

## ■ Tipografía del cuerpo de la página

- "Suele" ser una fuente sans serif.
- Normalmente será una tipografía simple, legible y no demasiado pequeña.

## ■ Tipografía de títulos y textos cortos de tamaño grande

- Suelen tener un **tamaño de letra grande**, de forma que hasta las tipografías decorativas son legibles.
- Por este motivo hay más libertad para elegirlas.

# ¿Tipografías seguras?

- La fuente utilizada en un sitio Web tiene que estar instalada en el sistema.
- Las **fuentes seguras** (*safety fonts*) son las que están disponibles en “casi” todos los sistemas.
- Fuentes alternativas
  - CSS (font-family)
- @font-face

Andale Mono  
Times New Roman  
Georgia  
Verdana  
Arial  
Courier New  
Trebuchet MS  
Comic Sans  
**Impact**

## 2.3. Tipografía y legibilidad

- **Limitar el número de fuentes** a usar en un sitio. Se aconseja no pasar de tres; de esas, una está incluida en el logo, así que sólo quedarían dos.
- **Tipo de fuente fácil de leer.** En tamaño grande, y en tamaño pequeño.
- **Espacio entre caracteres y entre palabras.** Debe ser suficiente para facilitar la lectura.  
*Se puede configurar con CSS (letter-spacing & word-spacing).*
- **Tamaño de la fuente (CSS):** utilizar **proporciones** para indicar el tamaño de la letra para que las fuentes se puedan adaptar a diferentes resoluciones.  
*En CSS: especificar el tamaño en % o "em".*

- **Una sola fuente por párrafo.**
- **Interlineado.** El texto con espacio vertical añadido es mucho más sencillo de leer. Se aconseja que el espacio entre líneas sea 1,2em (1,2 veces el tamaño de la fuente).  
**Propiedad CSS (line-height).**
- **Ancho de la línea:** evitar alargar el texto central a toda la anchura de una pantalla.  
40-60 caracteres se consideran en general una buena longitud de línea.

- **Alineación.** Lo ideal es que el texto esté **alineado a la izquierda**. Justificar el texto puede llevar a generar espacios raros entre las palabras.

### Propiedad CSS (text-align)



The image shows two side-by-side text boxes. The left box, containing justified text, has a large red checkmark in the bottom-left corner. The right box, containing aligned text, has a small red cursor icon pointing at the bottom-right corner of the text area. Both boxes contain identical text about the W3C Consortium.

El World Wide Web Consortium (W3C) es una comunidad internacional que desarrolla estándares que aseguran el crecimiento de la Web a largo plazo. Únete a los grupos, y participa en los blogs del W3C [inglés] y en otras discusiones. Agradecemos tu ayuda para llevar a cabo el objetivo del W3C: guiar la Web hacia su máximo potencial.

El World Wide Web Consortium (W3C) es una comunidad internacional que desarrolla estándares que aseguran el crecimiento de la Web a largo plazo. Únete a los grupos, y participa en los blogs del W3C [inglés] y en otras discusiones. Agradecemos tu ayuda para llevar a cabo el objetivo del W3C: guiar la Web hacia su máximo potencial.

- **Títulos realmente diferentes del texto central.** Dividir el texto con títulos y subtítulos o con listas facilita que los usuarios puedan encontrar lo que buscan en una página.
- **Escribir párrafos cortos.**
- **Evitar los errores tipográficos y ortográficos.** Dificulta la lectura y es la imagen de la empresa.
- **Dejar espacios en blanco.** Es decir, darle a los ojos del lector espacios en blanco para que descance la vista.

# 3. Guías de estilo

- Una guía de estilo es uno o varios documentos que definen las **normas que deben seguir las interfaces de un sitio web**.
- Gracias a la guía de estilo se garantiza la **coherencia** del sitio, ya que proporciona un aspecto homogéneo.

- Algunos elementos que se pueden definir en la guía de estilo son:
  - Colores
  - Tipografía
  - Logotipo
  - Iconos
  - Estructura del sitio web

#### Ejemplos:

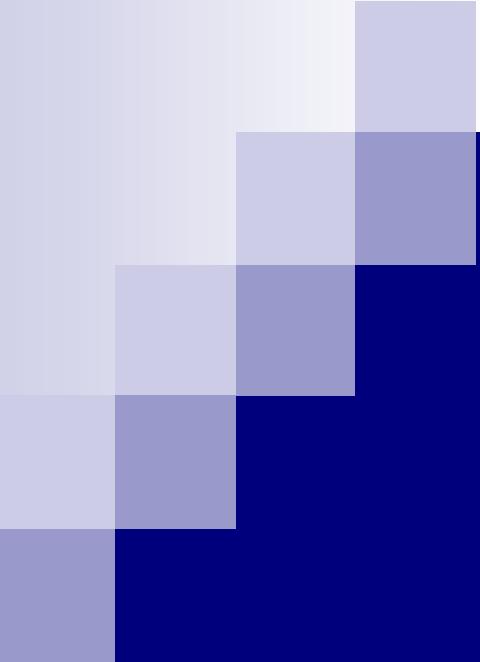
- Guía de estilo de la Universidad de Cádiz:  
<https://docwp.uca.es/wp-content/uploads/2019/07/Manual-Estilo-Web.pdf>
- Estilo visual de la Universidad del País Vasco:  
<https://www.ehu.eus/es/web/marka/estilo-bisuala>
- Guía de estilo del Gobierno de Navarra:  
[http://www.navarra.es/home\\_es/ManualEstilo/](http://www.navarra.es/home_es/ManualEstilo/)
- Guía de estilo de la BBC:  
<http://www.bbc.co.uk/guidelines/futuremedia/>. Destacan las Visual Language Guidelines.

# Referencias sobre color

- Curso idesWeb. El uso del color.
  - Diseño gráfico de una aplicación Web (del curso iDesWeb):  
<http://idesweb.es/temario/diseno-grafico-de-una-aplicacion-web> ¿Qué es el color? <http://www.youtube.com/watch?v=teW0H1GFRys>
  - Significado de los colores. <http://www.youtube.com/watch?v=CMhO3WKBd4g>
  - Diseñando con color. <http://www.youtube.com/watch?v=ZJKxATHWaBg>
- Combinaciones de color y modelos de diseño (del curso Conceptos de diseño Web, de la UOC):  
<http://mosaic.uoc.edu/ac/le/es/m2/ud5/index.html>
- Hoja de referencia de la teoría del color: <http://www.paper-leaf.com/blog/2010/01/color-theory-quick-reference-poster/>
- Comunicación visual gráfica. Material didáctico:  
<http://visualgrafica.wordpress.com/material-didactico/>
- Material Design:  
<https://material.io/design/color/the-color-system.html#color-theme-creation>

# Referencias sobre tipografía

- Web Style Guide 3<sup>rd</sup> edition, 8.Typography.  
<http://webstyleguide.com/wsg3/8-typography/index.html>
- UOC. Conceptos de diseño Web. La tipografía en la Web. <http://mosaic.uoc.edu/ac/le/es/m2/ud6/index.html>
- UOC. [Conceptos básicos de diseño gráfico.](#)
- CÓRCOLES TENDERO, J.E.; MONTERO SIMARRO, F. *Diseño de Interfaces*. Madrid: Ra-Ma, 2012. p. 11-41.
- TIDWELL, J; Designing Interfaces, O'Reilly 2005. Sección 9.2.: "*The basics of visual design*".



# UT 2 - Crear interfaces web en HTML

# Indice

1. Conceptos básicos
  2. !DOCTYPE y metaelementos en HTML5
  3. Secciones
  4. Agrupar contenido
  5. Semántica a nivel de texto
  6. Editar contenido
  7. Contenido embebido
  8. Enlaces
  9. Tablas
  10. Formularios
- Referencias

# 1. Conceptos básicos

## Versiones

### HTML 4.01

- Última versión completa de HTML 4 (1999)

### XHTML 1.0 y 1.1

- Extensible HTML.
- Variante de HTML. Más estricto que HTML. Si una página está escrita en XHTML se puede decir también que está escrita en HTML.
- Versiones:
  - XHTML 1.0 y XHTML 1.1
  - XHTML 2.0: Muy estricto, ha sido abandonado.

### HTML5 (*sin espacio entre HTML y 5*)

- Recomendación del W3C (28 de octubre de 2014).
- Compatible con XHTML, pero menos estricto que XHTML.

### HTML 5.1 2<sup>a</sup> ed. (*con espacio entre HTML y 5*)

- Recomendación del W3C (3 de octubre de 2017).

### HTML 5.2

- Recomendación del W3C (14 de diciembre de 2017).

# Etiquetas

- Es: una palabra clave encerrada entre corchetes angulares `<html>`
- Tipos: etiqueta inicial `<html>` y etiqueta final `</html>`.
  - La etiqueta inicial puede tener **atributos**.

`<html lang="es">`
- **Etiquetas: sintaxis HTML5:**
  - Se pueden escribir en mayúsculas o minúsculas `<html>` o `<HTML>`
  - Se recomienda: **minúsculas** `<html>` (por compatibilidad con versiones anteriores de HTML)

# Elementos

- Es: conjunto formado por la etiqueta inicial, la etiqueta final y lo que haya entre ambas.
  - Ej. `<head>...</head>`
- **Elementos vacíos**
  - Elementos que sólo tienen la etiqueta inicial, no tienen etiqueta final.  
Ej. `<br>`
  - **Elementos vacíos: sintaxis HTML5:** dos formas admitidas:
    - `<br>`
    - `<br/>`

# Elegir las etiquetas semánticamente

- Es importante **elegir las etiquetas semánticamente**, es decir, por su significado, no por cómo se va a visualizar (o se suele visualizar) en el navegador.
  - Ej. Las etiquetas <h1>...<h6> se deben utilizar para texto que es cabecera de un determinado nivel, no por el aspecto que le da por defecto el navegador (este se cambia con CSS).
- Un documento con etiquetas semánticamente correctas
  - Permite que los lectores no-humanos, como los **robots de búsqueda**, indexen correctamente el contenido.
  - Se visualiza correctamente en **más navegadores**.

# Elementos anteriores a HTML5

## ■ Elementos obsoletos

- Ya no forman parte del lenguaje HTML5.
- Ej. `<center>`, `<frame>`, `<frameset>`, `<noframes>`,  
`<font>`

## ■ Elementos que tienen un significado diferente

- Ej. `<i>` o `<b>`

## ■ Elementos nuevos

- Ej. `<mark>`, `<address>`, `<time>`

# Atributos

- Proporcionan información acerca de un elemento HTML.
  - Ej. Indicar el lenguaje de la página: <html lang="es">
- Normalmente tienen la forma
  - **nombre=valor** (ej. lang="es")
- **Atributos: sintaxis HTML5**
  - **Nombre** del atributo: mayúsculas (**LANG**) o minúsculas (**lang**).
    - Se recomienda: **minúsculas** (**lang**)
  - **Valor** del atributos: sólo necesitan comillas si contienen espacios en blanco; las comillas pueden ser simples o dobles.
    - Se recomienda: siempre **comillas** (simples o dobles).

# Atributos booleanos

- Sólo pueden tomar el valor verdadero o falso.
  - Toma el **valor falso**: cuando no aparece en la etiqueta.
  - Toma el **valor verdadero**: cuando aparece en la etiqueta:
    - Sin que se le asigne valor
    - Asignándoles un valor que puede ser el nombre del atributo o la cadena vacía
- Ej. Los siguientes ejemplos son equivalentes:

`<h1 hidden>`

`<h1 hidden="hidden">`

`<h1 hidden="">`

# Atributos globales

■ Son atributos que se pueden especificar para todos los elementos HTML.

- id
- title
- lang
- translate
- dir
- class
- style
- accesskey
- contenteditable
- draggable
- dropzone
- hidden
- spellcheck
- tabindex

# Espacios en blanco

- Al igual que los saltos de línea, en HTML los espacios en blanco no son significativos.
- Uno, dos o diez espacios en blanco se visualizan como un sólo espacio en blanco.

# Referencias de caracteres

- Necesarios para:
  - Caracteres que **no existen en el código de caracteres** utilizado.  
Ej. en ASCII no se pueden representar: @, á,
  - Caracteres que tienen un **significado especial** en HTML.  
Ej. el carácter < se interpreta como el inicio de una etiqueta.
  - Caracteres que **no están disponibles en el teclado**.  
Ej ® >? &>X£=?
- **Referencias de caracteres**
  - **Empiezan con & y terminan con ;**
  - Se pueden escribir de tres formas:
    - **Nombre:** &nombre;
    - **Número decimal:** &#nnn;
    - **Número hexadecimal:** &#Xnnn;

# Referencias de caracteres

Carácter	Nombre	Descripción	Número
&nbsp;	&nbsp;	Espacio en blanco que no se puede romper	&#160;
&	&amp;	Ampersand	&#038;
<	&lt;	Menor que	&#060;
>	&gt;	Mayor que	&#062
©	&copy;	Copyrigth	&#169;
€	&euro;	Euro	&#8364;
£	&pound;	Libra	&#163;
"	&quot;	Dobles comillas	
'	&apos;	Comillas simples	
á, é, í, ó, ú	&aacute; &eacute; &iacute; &oacute; &uacute;	a, e, i, o, u con tilde aguda.	

# Referencias de caracteres: &nbsp;

- *No-break space* (espacio en blanco que no se debe romper).
- Ejemplo: **Carlos V**
- No se debe utilizar para introducir espacios en blanco por una cuestión de estética: la presentación se debe controlar con CSS.

# Validar

- Herramienta: <http://validator.w3.org/>
- **Empezar a corregir los primeros errores encontrados**, no por los últimos, que pueden ser consecuencia de los primeros.
- **Buena práctica** porque los documentos HTML válidos, tienen más posibilidades de **visualizarse correctamente en diferentes navegadores**.

# 2. !DOCTYPE y metalementos en HTML5

## <!DOCTYPE html>

- Es el DOCTYPE que indica que es un documento de HTML5.

Necesario para asegurar que el navegador interprete el documento como HTML5 (por defecto puede interpretarlo de otro modo).
- Debe ser la primera línea del archivo: no debe haber espacios ni líneas antes de ella.

### <html> ... </html>

- Elemento raíz del árbol de elementos.

```
<!DOCTYPE html>
<html lang="es">
</html>
```

- El único atributo que se necesita es **lang**.

## <link rel="..." href="...>

- Permite incorporar, desde archivos externos: estilos, scripts, imágenes, iconos.
- Atributos
  - <**link** **rel**=..." **href**=...">
  - **rel**: relación entre el documento HTML y el archivo
  - **href**: ruta para cargar el archivo
  - **type**: opcional en HTML5
- **Ejemplo**: Incorporar una hoja de estilos:
  - <**link** **rel**=**"stylesheet"** **href**=**"css/estilo.css"**>

## <meta charset="UTF-8">

- **Valor de este atributo:**
  - Juego de caracteres del documento
  - Se puede escribir en mayúsculas o minúsculas.
- **Importante:** Se debe utilizar el mismo juego de caracteres en todos los ficheros de un sitio Web: HTML, PHP, CSS, JS, etc.
- Cuál elegir:
  - **UTF-8 sin BOM.** Permite utilizar texto en diferentes idiomas dentro del documento.
  - Si no se puede el anterior: **ISO-8859-1** (también llamado **Latin1**). Buen juego de caracteres para los idiomas del oeste de Europa.
  - Si se necesita el símbolo del euro: **ISO-8859-15 (Latin9)**
  - Latin1 y Latin9 se diferencian en muy pocos caracteres.

# 3. Secciones

- En HTML5 se definen nuevas etiquetas para definir secciones.
- ¿Por qué se definen nuevas etiquetas?
  - Se utilizaban muchos div del tipo: <div id="header"><div id="nav">....
  - Se han creado etiquetas específicas para esos usos.
  - De este modo, los navegadores y otros programas que rastrean la web pueden entender mejor el contenido del documento HTML.

- Etiquetas que definen “secciones”:
  - **body**
  - **article**, **section**
  - **nav**, **aside**
- Dentro de todas las “secciones” *puede haber*:
  - Títulos y subtítulos: **h1**, **h2**, **h3**, **h4**, **h5**, **h6**
  - Cabecera: **header**
  - Pie: **footer**
- Dentro de las “secciones” **article** y **body** *puede haber*:
  - Información de contacto: **address**

Esta etiqueta puede estar anidada dentro de otra etiqueta (div, header,  
21 footer), pero se refiere a su antecesor article o body.

## **<body> ... </body>**

- Elemento de tipo «sección».
- Sólo puede haber un elemento <body> en la página.
- Contiene la parte visible de la página.

# <article>

- Elemento de tipo «sección».
- Agrupa **contenido independiente**, que se puede extraer individualmente del documento y sindicar (ej. RSS) sin perder su significado.
- El ejemplo típico es una entrada de un blog.

## Ejemplo

```
<article>
  <header>
    <h1>La primera clase</h1>
    <p>02/05/2005</p>
  </header>
  <p>....</p>
  <p>...</p>
  <footer>
    <a href="?comments=1">Mostrar
      comentarios...</a>
  </footer>
</article>
```

# <section>

- Elemento de tipo «sección».
- Representa una **sección genérica de un documento o aplicación**.
- Una sección es un **grupo de contenido del mismo tema**. El tema de cada sección **debería identificarse** mediante un elemento de cabecera (**h1-h6**) que esté **dentro de la sección**.
- Ejemplos de secciones serían las zonas en las que se puede dividir una página web: introducción, news.
- Se aconseja utilizar article en lugar de sección cuando tendría sentido sindicar el contenido del elemento.

## Ejemplo

```
<body>
  <h1>Graduación</h1>
  <section>
    <h2>Programa de actos</h2>
    <ul>
      <li>Palabras de..</li>
      <li>...</li>
    </ul>
  </section>
  <section>
    <h2>Graduados</h2>
    <ul>
      <li>Susana R.</li>
      <li>...</li>
    </ul>
  </section>
</body>
```

# <nav>

- Elemento de tipo «sección».
- Contiene los **enlaces de la navegación principal** (dentro del documento o a otras páginas).
- También se puede utilizar para **navegación secundaria**.
- Puede haber varios elementos <nav> dentro de una página.
- No todos los grupos de enlaces tienen que estar en un elemento nav, sólo los que forman bloques de navegación principal. Por ejemplo, en el footer suele haber una lista corta de enlaces que no es necesario que formen parte de un <nav>.
- El elemento <nav> puede formar parte del <header> o no.
- Si contiene una lista de enlaces, en general se recomienda utilizar el elemento ul.

## Ejemplo

```
<nav>
  <ul>
    <li><a href="/">Home</a></li>
    <li><a href="/events">
      Events</a></li>
    ...
  </ul>
</nav>
```

## <aside>

- Elemento de tipo «sección».
- Contenido: relacionado tangencialmente con el contenido que está a su alrededor, y que podría tener sentido separado de ese contexto.
- Se puede utilizar para efectos tipográficos, anuncios, agrupar elementos nav, etc.

### Ejemplo

Utilizar un elemento <aside> para una cita, dentro de un artículo más largo.

```
<p>Mi trabajo...</p>
<aside>
  <q>La gente me suele preguntar cómo me
  divierto cuando no estoy trabajando... </q>
</aside>
<p>....</p>
```

## <h1> ... <h6>

- Título de un elemento de tipo “sección” (**article**, **section**, **nav**, **aside** o **body**).
- Se recomienda seguir el orden lógico: h1, h2, h3, ...
- En cada elemento de tipo “sección”, se puede empezar desde h1.  
Sin embargo se recomienda que en cada página HTML haya sólo un h1, y que los títulos definan la estructura de contenidos de la página.

# <header>

- Cabecera de un elemento de tipo sección (**section**, **article**, **aside**, **nav** o **body**).
- Contenido introductorio.  
Normalmente contiene un grupo de ayudas introductorias o de navegación.
- No siempre contiene elementos de cabecera como h1-h6. Puede contener, por ejemplo, un formulario de búsqueda, o la tabla de contenidos de la sección.

## Ejemplo

```
<article>
  <header>
    <h1>La primera clase</h1>
    <p>02/05/2005</p>
  </header>
  <p>....</p>
  <p>...</p>
  <footer>
    <a href="?comments=1">Mostrar
      comentarios...</a>
  </footer>
</article>
```

# <footer>

- Pie de un elemento de tipo sección (**section, article, aside, nav o body**).
- Normalmente contiene información sobre la sección, como quién lo escribió, enlaces a otros documentos relacionados, datos de copyright.
- Puede contener otras secciones.

## Ejemplo

```
<footer>
<!-- footer del sitio -->
<nav>
  <p> <a href="#">credits</a> -
    <a href="#">terms</a> -
    <a href="#">index</a>
  </p>
</nav>
<p>copyright © 2009 ...</p>
</footer>
```

# <address>

- Información de contacto de un elemento **article** o de un elemento **body**.
- Se puede incluir en un footer.
- No debe utilizarse para direcciones si no son direcciones de contacto.

## Ejemplo

```
<footer>
  <address>
    Para más detalles, contactar con <a href="mailto:js@example.com">John Smith</a>.
  </address>
  <p><small>© copyright 2038 Example Corp.</small></p>
</footer>
```

# Compatibilidad con navegadores no HTML5

- Las nuevas etiquetas HTML5 no son reconocidas por los navegadores que no soportan HTML5.
- Soluciones:
  - Con estilos:

```
section, article, .. {  
    display: block;  
}
```
  - Con JavaScript

[https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Using\\_HTML\\_sections\\_and\\_outlines#Using\\_HTML5\\_elements\\_in\\_non-HTML5\\_browsers](https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Using_HTML_sections_and_outlines#Using_HTML5_elements_in_non-HTML5_browsers)

# 4. Agrupar contenido

- p
- hr
- pre
- blockquote
- ol
- ul
- li
- dl
- dt
- dd
- figure
- figcaption
- div
- main

<https://www.w3.org/TR/html52/grouping-content.html#grouping-content>

## <ol>

### ol: listas ordenadas

```
<ol>
    <li>Elemento 1</li>
    <li>Elemento 2</li>
    <li>Elemento 3</li>
    <li>Elemento 4</li>
    <li>Elemento 5</li>
</ol>
```

1. Elemento 1
2. Elemento 2
3. Elemento 3
4. Elemento 4
5. Elemento 5

Las listas se pueden **anidar**.

# <ul>

## ul: listas desordenadas

```
<ul>
    <li>Elemento 1</li>
    <li>Elemento 2</li>
    <li>Elemento 3</li>
    <li>Elemento 4</li>
    <li>Elemento 5</li>
</ul>
```

- Elemento 1
- Elemento 2
- Elemento 3
- Elemento 4
- Elemento 5

Las listas se pueden **anidar**.

## <dl>, <dt>, <dd>

### dl: listas de definiciones

```
<dl>
    <dt>Director</dt>
    <dd>Es el responsable de...</dd>
    <dt>Jefe de estudios</dt>
    <dd>Es el responsable de...</dd>
    <dd>Además...</dd>
    <dt>Secretario</dt>
    <dd>Es el responsable de...</dd>
    <dt>Conserjes</dt>
    <dt>Personal administrativo</dt>
    <dd>Son los responsables de...</dd>
</dl>
```

Director

Es el responsable de...

Jefe de estudios

Es el responsable de...

Además...

Secretario

Es el responsable de...

Conserjes

Personal administrativo

Son los responsables de...

Las listas <dl> pueden anidarse dentro de otras, pero no pueden contener otras dentro de ellas.

## <figure> y <figcaption>

Se utilizan para encapsular una figura como un sólo item.

Puede contener un título para la figura.

```
<figure>
<figcaption>
</figcaption>
</figure>
```

# 5. Semántica a nivel de texto

Etiquetas *in-line* que permiten definir semántica:

- a
- em
- strong
- small
- s
- cite
- q
- dfn
- abr
- data
- time
- code
- var
- samp
- kbd
- sub y sup
- i
- b
- u
- mark
- ruby
- rb
- rt
- rtc
- rp
- bdi
- bdo
- span
- br
- wbr

# 6. Editar contenido

- ins
- del

# 7. Contenido embebido

- Permiten introducir en el documento
  - Otro recurso o archivo: ej., imagen, vídeo, página web
  - Otro contenido: ej., svg.
- **Todas estas etiquetas son elementos in-line (*phrasing content*).**
- Etiquetas que permiten introducir “contenido embebido” : **img, iframe, embed, object, audio, video, canvas, math** y **svg**.
- En esta presentación, veremos **img** e **iframe**.

```
<img src="" width="" height="">
```

## Atributos width y height

- Ancho de la imagen en píxeles: `width="número"`
- Alto de la imagen en píxeles: `height="número"`
- Utilizar estos atributos hace que la página se cargue más deprisa.
- Utilizar el tamaño real de la imagen, para que el navegador no tenga que escalarla.

## <img ... alt="**texto alternativo**">

- Texto alternativo:
  - Texto que se muestra si no se puede cargar una imagen.
  - Texto utilizado por los lectores de pantalla.
- Valor del texto alternativo:
  - Una breve descripción de lo que hay en la imagen.
  - Si una imagen no añade ningún significado al texto se recomienda alt=""

# Imágenes decorativas

- Se recomienda mover las imágenes puramente decorativas al CSS (*background-image*).
- Ventajas:
  - Documento más limpio y accesible.
  - Más fácil hacer cambios en el aspecto.

```
<iframe src="" width ="" height ="">  
    contenido alternativo  
</iframe>
```

- Permite embeber en un documento HTML otro documento HTML separado u otro recurso.
- A veces se utiliza para contenido de terceros, como anuncios interactivos u otros widgets, garantizando así que no interfieren con el contenido y scripting del resto de la página.

```
<iframe src="lista.html" width="400" height="200">  
Tu navegador no soporta inline frames.  
Lee la <a href="lista.html">Lista</a>  
</iframe>
```

# 8. Enlaces

## ■ Sintaxis

`<a href="URL">texto o elemento</a>`

## ■ ¿Qué puede ir entre las etiquetas <a> y </a>

- HTML 4.01: Elementos inline.
- HTML5: Cualquier elemento, incluso elementos de bloque.

## ■ Tipos de referencias "URL":

- Cómo indicar el documento.
  - **Referencias absolutas:** en cualquier sitio Web.
  - **Referencias relativas:** en el propio sitio Web.
- Cómo indicar un lugar concreto de un documento.
- Otras referencias (correo electrónico, ficheros para descarga).

## <a href="url absoluta">

- Para referenciar un documento de otro sitio Web
- Empiezan con el protocolo (http://), nombre de dominio y ruta si es necesaria.

<a href="http://www.w3c.org">w3c</a>

## <a href="url relativa">

- Para referenciar un documento de tu propio sitio.
- Se escribe la página web, y delante la ruta relativa.

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

□ En la carpeta padre

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

□ En una subcarpeta

`<a href="fotos/foto.html">foto</a>`

□ Relativo a la raíz del sitio

`<a href="/img/fotos/foto.html">foto</a>`

# <a href="id dentro de una página">

- PÁGINA DESTINO: `id="nombre"`

```
<h1 id="inicio">TITULO</h1>
```

- PÁGINA DONDE SE REFERENCIA: `#nombre`

- Enlace en la misma página Web

```
<a href="#inicio">Ir a inicio</a>
```

- Enlace a otra página del sitio

```
<a href="glosario.html#inicio">Ir al glosario (inicio)</a>
```

- Enlace a otra página de otro sitio

```
<a href="http://w3c.org/glosario.html#inicio">Ir al  
glosario (de w3c, inicio)</a>
```

## <a target="...">

- Atributo target: definir en qué ventana se abre la página enlazada

**target="\_blank"** En una pestaña nueva del navegador o en una nueva ventana.

**target="\_self"** En la misma ventana.

**target="\_parent"** En la ventana padre (marcos).

**target="\_top"** En la ventana superior del conjunto de marcos.

**target="nombre"** En una pestaña nueva del navegador o en una nueva ventana.

- Nombre es cualquier nombre que no empiece por subrayado (\_)
- Todos los documentos que se abran indicando ese target, se abren en la misma ventana.

- **Consejo:** Puede ser mejor no utilizar este atributo.

- El usuario se pierde, es más difícil que vuelva.
- Si accede desde dispositivos móviles, es más incómodo.
- Si accede con dispositivos lectores, puede que no lo sepan interpretar.

## <a href="mailto:....">

- Enlace a un correo

- Utilizar el protocolo mailto:

<a href="mailto:pepe@perez.com">....</a>

- Cuidado:

- Proporcionar un correo electrónico en un documento HTML es arriesgarse a recibir spam.
    - Algunas soluciones: escribirlo de modo que las personas puedan leerlo, pero los robots no:
      - pepe at perez.com
      - imagen

## <a href="tel:....">

### ■ Enlaces a teléfonos

- Aprovechar que los smartphones que se utilizan para acceder a Internet, permiten hacer llamadas telefónicas!

<a href="tel:+18050505">+18050505</a>

- Sólo funciona desde dispositivos móviles (excepto iPad y iPod Touch). No funciona desde equipos de escritorio.
- Consejos
  - Escribir el número internacional
  - Repetir el número entre las etiquetas <a>, por si no funciona el enlace

# 9. Tablas

- Se utilizan para presentar contenidos que se organizan en forma de tabla, como calendarios, estadísticas, etc.

ALUMNO	NOTA
Pepe	7,65
Maria	8,54
Jordi	5,55

```
<table>
    <tr>
        <th>Producto</th>
        <th>Precio</th>
    </tr>
    <tr>
        <td>Aceituna</td>
        <td>2,60</td>
    </tr>
    <tr>
        <td>Almendra</td>
        <td>1,20</td>
    </tr>
    <tr>
        <td>Cacahuete</td>
        <td>1,60</td>
    </tr>
    <tr>
        <td>Total</td>
        <td>5,40</td>
    </tr>
</table>
```

Producto	Precio
Aceituna	2,60
Almendra	1,20
Cacahuete	1,60
Total	5,40

## Etiquetas td y th

- Son las etiquetas en las que va el contenido.
- Contenido de una celda td o th: puede ser cualquiera: texto, gráfico, otra tabla, etc.

## Etiquetas th

- Sustituye a la celda <td> para celdas de cabecera.
- Se pueden utilizar para cualquier fila, aunque se recomienda para las primeras y últimas.
- También puede ser una columna.
- Los navegadores suelen presentar su contenido en **negrita**.

- Combinar celdas: atributos **colspan** y **rowspan** de **<td>** y **<th>**
- A estos atributos se les asigna un valor entero que especifica cuántas columnas o filas ocupa la celda.

```
<table>
    <tr>
        <th colspan="2">Precios de venta</th>
    </tr>
    <tr>
        <th>Producto</th>
        <th>Precio</th>
    </tr>
    <tr>
        <td rowspan="2">Aceituna</td>
        <td>2,60</td>
    </tr>
    <tr>
        <td>1,20</td>
    </tr>
    <tr>
        <td>Cacahuete</td>
        <td>1,60</td>
    </tr>
    <tr>
        <td>Total</td>
        <td>5,40</td>
    </tr>
</table>
```

Precios de venta	
Producto	Precio
Aceituna	2,60
	1,20
Cacahuete	1,60
Total	5,40

# Accesibilidad: atributo scope de la etiqueta <th>

## Syntax

```
<th scope="col|row|colgroup|rowgroup">
```

## Attribute Values

Value	Description
col	Specifies that the cell is a header for a column
row	Specifies that the cell is a header for a row
colgroup	Specifies that the cell is a header for a group of columns
rowgroup	Specifies that the cell is a header for a group of rows

## Etiquetas **thead**, **tbody**, **tfoot**

- Se utilizan para agrupar filas.
- ¿Por qué agrupar filas?
  - Algunos **navegadores especializados** pueden repetir las cabeceras o los pies cuando una tabla ocupa varias páginas.
  - Para dar estilo por grupos de filas.

```
<table>
  <thead>
    <tr>
      <th>Producto</th>
      <th>Precio</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Aceituna</td>
      <td>2,60</td>
    </tr>
    <tr>
      <td>Almendra</td>
      <td>1,20</td>
    </tr>

    <tr>
      <td>Cacahuete</td>
      <td>1,60</td>
    </tr>

  </tbody>
  <tfoot>
    <tr>
      <td>Total</td>
      <td>5,40</td>
    </tr>
  </tfoot>
</table>
```

## Título de las tablas: <caption>

- Utilizar este elemento para dar un título o breve descripción que se muestre junto a la tabla.
  
- Si se utiliza, debe ser el primer elemento dentro del elemento <table>.

## Agrupar columnas con <colgroup> y <col>

- Para **dar estilo** a columnas o grupos de columnas.
- Etiquetas:
  - Grupo de columnas: <**colgroup**> </**colgroup**>
  - Columnas dentro del grupo de columnas: <**col**>
- El atributo **span** (de <colgroup> y <col>) indica el número de columnas.
- Si se utiliza esta etiqueta, tiene que aparecer al principio, después de <table> y <caption> (si existe).

```
<table>
  <colgroup>
    <col span="2" style="background-color:red">
    <col style="background-color:yellow">
  </colgroup>
  <tr>
    <th>ISBN</th>
    <th>Title</th>
    <th>Price</th>
  </tr>
  <tr>
    <td>3476896</td>
    <td>My first HTML</td>
    <td>$53</td>
  </tr>
</table>
```

ISBN	Title	Price
3476896	My first HTML	\$53
5869207	My first CSS	\$49

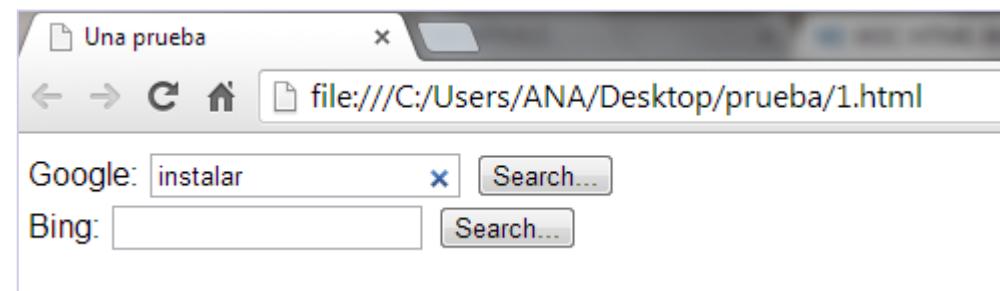
# 10. Formularios

- **<form action="..." method="...> ... </form>**
- Contiene los elementos de control de un formulario.
- ¡No anidar!
- Atributos
  - **action="URL"**
    - URL de la aplicación o script.
  - **method="post|get"**
    - Método utilizado para enviar los datos del formulario.
    - **post**: el navegador envía una petición separada al servidor con cabeceras especiales seguidas de los datos. Más seguro.
    - **get**: el navegador envía los datos al servidor en la URL.
  - **enctype="..."**
    - Método de codificación. Normalmente **application/x-www-form-urlencoded (default)** o **multipart/form-data**

Este ejemplo muestra dos formularios de búsqueda:

```
<form action="http://www.google.com/search" method="get">
<label>Google: <input type="search" name="q"></label>
<input type="submit" value="Search... ">
</form>

<form action="http://www.bing.com/search" method="get">
<label>Bing: <input type="search" name="q"></label>
<input type="submit" value="Search... ">
</form>
```



# Accesibilidad

Diseñar los formularios para los usuarios que no pueden utilizar navegadores visuales.

Tres etiquetas mejoran la accesibilidad de los formularios, haciendo conexiones semánticas entre los componentes de un formulario.

Son: **label**; **fieldset** y **legend**

```
<label for="...>  
</label>
```

- Permite asociar un texto (etiqueta) con un control de formulario.
- Dos formas de hacerlo:
  - Utilizando el atributo **for** en el elemento **label**, y el atributo **id** en el elemento de control

```
<label for="form-nombre">Nombre:</label>  
<input type="text" name=nombre id="form-nombre">
```
  - Colocando el control dentro del elemento **label**.

```
<label>Nombre:<input type="text" name=nombre></label>
```

<fieldset>

<legend>

</legend>

...

</fieldset>

- Con <fieldset> se agrupan elementos de control agrupados bajo un nombre común. Y con <legend> se les da un título.
- Atributos
  - **name="..."**  
Nombre del elemento.
  - **form="nombre\_formulario"**  
Para indicar explícitamente el nombre de su formulario.
  - **disabled**  
Hace que todos los elementos hijos de fieldset, excluidos los hijos del primer elemento legend, estén desactivados
- Los elementos **fieldset** se pueden anidar.

```
<fieldset name="clubfields" disabled>
<legend> <label>
    <input type=checkbox name=club
        onchange="form.clubfields.disabled = !checked">
    Use Club Card </label>
</legend>
<div><label>Name on card: <input name=clubname required>
    </label></div>
<div><label>Card number: <input name=clubnum required
    pattern="[-0-9]+">
    </label></div>
<div><label>Expiry date: <input name=clubexp type=month>
    </label></div>
</fieldset>
```

The screenshot shows a web browser window titled "Una prueba". The address bar indicates the file is located at "file:///C:/Users/ANA/Desktop/prueba/1.html". Inside the browser, there is a form with the following elements:

- A checkbox labeled "Use Club Card" which is checked.
- A text input field labeled "Name on card:".
- A text input field labeled "Card number:".
- A date input field labeled "Expiry date:" with a dropdown arrow for month selection.

The entire form is contained within a `<fieldset name="clubfields" disabled>` block, which is why the "disabled" attribute is reflected in the browser's UI.

# Atributos comunes a la mayoría de los controles

- name
- value
- size
- maxlenght
- disabled
- readonly

# Atributo name

- Es el **nombre de la variable** donde se almacenará el valor recogido en ese elemento de control.
- Debe ser **única en el formulario**.
- Atributo **obligatorio** en **casi todos** los controles de un formulario, **excepto en submit y reset** (que lo pueden llevar, pero no es obligatorio porque tienen una función dentro del formulario diferente de recoger datos).
- **Ejemplo** (de uno de los dos formularios del ejemplo anterior):

```
<label>Google: <input type="search" name="q"></label>
```

## Atributo value

- Valor de la variable.
- En los campos de tipo texto, este valor aparecerá en el campo cuando se carga o cuando se reinicializa el formulario.

**value="valor"**

## Atributo size

- Es el número de caracteres que el navegador permite ver cuando se edita el contenido de un elemento de control.

## Atributo maxlength

- Es el número máximo de caracteres que puede tener la variable.

## Atributo disabled

- Es un atributo booleano.
- Se utiliza para deshabilitar el contenido (la posibilidad de modificarlo).
- Si se utiliza:
  - En un control: deshabilita el control.
  - En un elemento fieldset: deshabilita todos los elementos del grupo (excepto los descendientes del elemento legend, si hay alguno).

## Atributo readonly

- Es un atributo booleano.
- Se utiliza para evitar que un usuario pueda editar el control.

# <input type="..." name="..." >

Algunos usos de la etiqueta <input>

type="..."	descripción
<b>text</b>	Permite introducir un texto, es el tipo por defecto.
<b>password</b>	Funciona como un control de entrada de texto, sólo que no se muestra el texto que vamos escribiendo. En su lugar veremos unos asteriscos u otro símbolo. No ofrece seguridad (no cifra)
<b>file</b>	Muestra un botón Seleccionar o Examinar para poder elegir el archivo que se quiere enviar. Requiere que el formulario envíe los datos mediante el método "post" y utilizando el atributo enctype="multipart/form-data". Permite enviar múltiples archivos, utilizando el atributo multiple.
<b>hidden</b>	Mantiene un dato oculto a la vista pero se pasa al servidor junto con el resto de datos del formulario. Campo adecuado para pasar información de una página a otra. Además, se puede utilizar en los scripts como un dato más.

# <input>: tipos HTML5

## ■ Entradas de texto especiales

- `type="search"`
- `type="email"`
- `type="tel"`
- `type="url"`

## ■ Entradas numéricas

- `type="number"`
- `type="range"`
- atributos `min`, `max` y `step`

## ■ Entradas de fechas y hora

- `type="date"`
- `type="time"`
- `type="datetime"`
- `type="datetime-local"`
- `type="month"`
- `type="week"`

## ■ Entradas para seleccionar un color

- `type="color"`

## <input>: tipos HTML5 y compatibilidad

- ¿Qué ocurre con los navegadores que no reconocen los tipos de HTML5?
  - Los ignora
  - Asigna el tipo por defecto: **type="text"**
- Mejora progresiva: diseñar para que las páginas se puedan ver con navegadores muy avanzados y menos avanzados.

## <input type="radio">

- Un grupo controles de tipo "radio" tienen **asociada la misma variable** (atributo name).

Sólo puede haber uno seleccionado: a la variable se le asignará el valor (value) del elemento seleccionado

- Atributos

- **type="radio"**
- **name="variable"**
- **value="valor"**
- **checked**

```
<p>Selecciona la edad</p>
<ul>
  <li><input type="radio" name="edad" value=15 checked>15</li>
  <li><input type="radio" name="edad" value=20>20</li>
  <li><input type="radio" name="edad" value=30>30</li>
</ul>
```

## <input type="checkbox">

- Control que puede tener dos estados: activado o no activado.
- Atributos
  - **type="checkbox"**
  - **checked**

## <input type="submit", "reset" y "button">

- <**input type="submit" value="..."**>
  - Envía los datos de un formulario al servidor.
- <**input type="reset" value="..."**>
  - Reinicia los controles de un formulario a sus valores por defecto.
- <**input type="button" value="..."**>
  - Crea un botón genérico sin una función concreta. Se le suele dar funcionalidad con JavaScript.
- Atributos
  - **value="..."** Texto del botón

## <textarea> ... </textarea>

- Para introducir varias líneas de texto.
- El texto que se escriba entre las dos etiquetas será el valor por defecto.
- Atributos
  - **rows="n"** número de líneas de texto
  - **cols="n"** número de columnas (caracteres) de texto

## <button>...</button>

- <button> . . . </button>
- Representa un **botón etiquetado con el texto** que hay entre las dos etiquetas.
- Atributos
  - **type="submit|reset|button|menu"**
    - **submit** Envía el formulario (default)
    - **reset** Reinicia el formulario
    - **button** No hace nada
    - **menu** Muestra un menú
  - Si el tipo es "submit", el navegador realizará la **validación de restricciones**.

## <select>... </select>

<select>...</select>

<option>...</option>

<optgroup>...</optgroup>

- Menús desplegables
- Menús con scroll
- Agrupar: <optgroup label="...">
  - Opcional; aparece al principio del grupo.

## <datalist>...</datalist>

- Muestra una lista de valores posibles para un elemento <input>.
- A diferencia de <select>, el usuario puede introducir un valor diferente a los proporcionados en la lista.

```
<input name="estudios" type="text" list="tipos est">
<datalist id="tipos est">...
    <option value="primaria">
    <option value="secundaria">
    <option value="universidad">
</datalist>
```

# Nuevos atributos

- **autocomplete="on" o "off"**
  - mostrar los textos de entradas anteriores como posibles valores
- **novalidate**
  - atributo para la etiqueta form, permite enviar un formulario sin validar
- **formnovalidate**
  - atributo para la etiqueta submit, permite enviar un formulario sin validar
- **placeholder**
  - Muestra un texto para ayudar al usuario a llenar el formulario
- **required**
  - El formulario no se debe enviar si el campo está vacío

# Nuevos atributos (2)

## ■ **multiple**

- Permitir múltiples entradas en un mismo campo. Los valores se escriben separados por comas. Sólo lo permiten algunos controles como **email** y **file**.

## ■ **autofocus**

- coloca el foco

## ■ **pattern**

- Define un patrón para validar una entrada
- Ej. <input pattern="[0-9]{5}" name="cpostal" title="Inserte código postal">

## ■ **form**

- Permite declarar elementos de un formulario fuera de la etiqueta form

# Otras etiquetas HTML5

**progress**

**meter**

**output**

**keygen**

# Referencias

- Especificaciones oficiales de HTML:
  - HTML 4.01: <http://www.w3.org/TR/html4/>
  - XHTML 1.0: <http://www.w3.org/TR/xhtml1/>
  - HTML5: <https://www.w3.org/TR/html52/>    <http://www.w3.org/TR/html5/>
- Libros Web: <http://librosweb.es/>
- Especificaciones de CSS: <https://www.w3.org/Style/CSS/specs.en.html>
  - Especificaciones de CSS2 (en castellano):  
<http://www.sidar.org/recur/desdi/traduc/es/css/cover.html#minitoc>
- Compatibilidad de los navegadores: <http://caniuse.com/>
- Web technology for developers  
<https://developer.mozilla.org/en-US/docs/Web>
- HTML Training de W3C (inglés, algunos materiales en castellano):  
<http://www.w3.org/community/webed/wiki/HTML/Training>.
- Tutoriales de HTML: [www.w3schools.com](http://www.w3schools.com)
- WHATWG: <http://www.whatwg.org/>

# DISEÑO DE INTERFACES WEB

UT3. Aplicar estilos CSS (I):  
Estilos básicos.



CFGS Desarrollo de Aplicaciones Web  
CIFP Juan de Colonia  
Curso 2020-2021

## Tabla de contenido

1.	Conceptos básicos.....	3
1.1.	Sintaxis .....	3
1.1.1.	Comentarios.....	3
1.1.2.	Mayúsculas y minúsculas.....	3
1.1.3.	Identificadores.....	3
1.1.4.	Palabras clave.....	3
1.1.5.	Declaraciones .....	3
1.1.6.	Propiedades short-hand (o propiedades resumidas) .....	5
1.1.7.	Cómo interpretan los navegadores los errores .....	5
1.1.8.	Buenas prácticas de escritura de reglas CSS.....	5
1.2.	Selectores.....	6
1.2.1.	Selectores de tipo, universal, de clase y de identificador .....	6
1.2.2.	Selectores de atributo.....	6
1.2.3.	Pseudoclases.....	7
1.2.4.	Pseudo-elementos (CSS1 y 2) .....	11
1.2.5.	Combinadores de selectores .....	11
1.2.6.	Especificidad de los selectores.....	12
1.3.	Aplicar CSS al HTML: Hojas de estilo en Cascada .....	13
1.3.1.	Estilos en línea .....	13
1.3.2.	Estilos incrustados .....	13
1.3.3.	Hojas de estilos externas .....	13
1.3.4.	Importar hojas de estilos con @import .....	14
1.3.5.	Herencia.....	15
1.3.6.	Cascada .....	17
2.	Dar estilo a los elementos .....	18
2.1.	Unidades.....	18
2.1.1.	Unidades absolutas.....	19
2.1.2.	Unidades relativas.....	19
2.2.	Cadenas .....	20
2.3.	URL.....	20
2.4.	Color .....	21
2.4.1.	Nombres de colores .....	21
2.4.2.	Valores de los colores: RGB, RGBa, HSL, HSLa .....	21
2.5.	Fuentes .....	23
2.5.1.	Fuentes web @font-face .....	24
2.6.	Texto.....	26

2.7.	Fondo.....	27
2.8.	Listas .....	28
2.9.	Tablas .....	29
3.	Modelo de formato visual .....	30
3.1.	El modelo de cajas (box model) .....	30
3.1.1.	Márgenes (margin) .....	30
3.1.2.	Relleno (padding) .....	32
3.1.3.	Bordes (border) .....	32
3.1.4.	Bordes redondeados (border-radius) CSS3.....	33
3.1.5.	Contenido.....	34
3.2.	Bloques de contención.....	34
3.2.1.	El bloque de contención inicial .....	34
3.2.2.	Tipos de elementos: elementos a nivel de bloque y elementos en línea .....	34
3.3.	La propiedad "display".....	35
3.4.	Esquema de posicionamiento .....	36
3.4.1.	Esquema de posicionamiento flotante .....	36
3.4.2.	Esquemas de posicionamiento de flujo normal y posicionamiento absoluto .....	40
3.5.	Efectos visuales .....	42
3.5.1.	Desbordamiento: overflow .....	42
3.5.2.	Visibilidad: visibility .....	43
4.	Estructuras de distribución ( <i>layouts</i> ).....	44
4.1.	Diseños de ancho fijo o fluidos.....	44
4.1.1.	Diseños de ancho fijo .....	44
4.1.2.	Diseños fluidos o líquidos.....	45
5.	Técnicas para crear columnas .....	46
5.1.	Crear columnas utilizando contenedores flotantes .....	46
5.2.	Crear columnas utilizando posicionamiento absoluto .....	47
5.3.	¿Qué técnica aplicar en los diseños fijo y fluido? .....	47
6.	Técnicas para crear barras de navegación.....	47
6.1.	Crear barras de navegación verticales .....	47
6.2.	Crear barras de navegación horizontales.....	48
7.	Referencias y bibliografía .....	49

## 1. Conceptos básicos

Las hojas de estilo en cascada o **CSS** (*Cascading Style Sheets*) permiten separar el contenido y el estilo en un sitio web.

Las **ventajas** de separar el contenido y el estilo son:

- Simplifica el **desarrollo y mantenimiento** de los sitios web.
- Permite **unificar los sitios web**: se pueden aplicar distintos estilos, con el mismo HTML.

### 1.1. Sintaxis

#### 1.1.1. Comentarios

- Un comentario es el texto que esté contenido entre /\* y \*/.
- Un comentario puede ocupar varias líneas y puede aparecer en cualquier lugar.
- Los comentarios no pueden anidarse.

#### 1.1.2. Mayúsculas y minúsculas

- Las hojas de estilo CSS **no diferencian mayúsculas/minúsculas** (*case-insensitive*) en los nombres de las propiedades CSS y de las palabras reservadas (ej. sans-serif). Sin embargo, se aconseja utilizar minúsculas.

*Ejemplo: Las dos declaraciones siguientes son correctas.*

```
color: red;  
COLOR: Red;
```

- **Sí se diferencian mayúsculas/minúsculas** en los nombres de los identificadores o de las clases.

#### 1.1.3. Identificadores

Los identificadores (nombres de elementos HTML, clases, identificadores, etc.):

- Pueden contener solamente los caracteres [A-Za-z0-9], el guion alto (-), el guion bajo (\_) y la apertura de exclamación (!; carácter U+00A1 (161)).
- No pueden comenzar con un número.
- Sólo algunos identificadores pueden comenzar con un guion: propiedades, valores, unidades, pseudo-clases, pseudo-elementos, reglas arroba.

#### 1.1.4. Palabras clave

- Las palabras clave se forman como los identificadores.
- Las palabras clave no se deben poner entre comillas ("..." o '...').

*Ejemplo: La primera declaración es correcta, pero la segunda no lo es.*

```
color: red; /* Correcto: red es una palabra reservada */  
color: "red"; /* Incorrecto: "red" es una cadena de caracteres */
```

#### 1.1.5. Declaraciones

Una hoja de estilo CSS consiste en una lista de declaraciones. Hay dos clases de declaraciones: **Reglas CSS** y **Reglas-arroba**.

## Reglas CSS

Una **regla CSS** está formada por un selector y un bloque de declaraciones.

### Selector

Un selector identifica los **elementos** afectados por la regla. Un selector puede ser el nombre de un elemento HTML, una clase, un identificador, etc. Un selector puede estar formado por varios selectores, **separados por comas**.

### Bloque de declaraciones

Un bloque comienza con una llave de apertura ({}) y termina con la correspondiente llave de cierre (}). Entre ambas llaves, debe haber una lista de cero o más declaraciones **separadas por punto y coma**.

### Declaración

La declaración contiene las instrucciones que indican al navegador cómo mostrar los elementos seleccionados. Está formada por uno o varios pares **propiedad:valor** (separados por dos puntos).

#### Ejemplos:

```
h1 {  
    color: blue;  
}  
  
h2, h3 {  
    color: blue;  
    background-color: red;  
}
```

## Reglas-arroba

Las reglas-arroba comienzan con el carácter arroba, '@', seguido inmediatamente por un identificador (ej. '@import' o '@media'). Después, puede ir una declaración seguida de punto y coma o un bloque de declaraciones.

Ejemplo: La regla '@import' termina en el punto y coma.

```
@import url("texto.css");
```

Ejemplo: La regla '@font-face' va seguida de un bloque de declaraciones.

```
@font-face {  
    font-family: "Vintage";  
    src: url("fuentes/vintage.ttf") format("truetype");  
}
```

### 1.1.6. Propiedades short-hand (o propiedades resumidas)

Algunas propiedades son propiedades resumidas que permiten especificar los valores de varias propiedades con una sola propiedad. Por ejemplo, la propiedad `font` es una propiedad resumida para definir a la vez `font-style`, `font-variant`, `font-weight`, `font-size`, `line-height` y `font-family`. Cuando se omiten algunos valores en la fórmula resumida, a cada propiedad "ausente" se le asigna su valor por defecto.

Ejemplo: Las siguientes reglas son equivalentes:

```
h1 {  
    font-weight: bold;  
    font-size: 12px;  
    line-height: 14px;  
    font-family: Helvetica;  
    font-variant: normal;  
    font-style: normal;  
}  
  
h1 {  
    font: 12px/14px Helvetica;  
}
```

En la segunda regla, las propiedades `font-variant` y `font-style` no se han especificado por lo que toman sus valores por defecto.

### 1.1.7. Cómo interpretan los navegadores los errores

- **Si un navegador encuentra una regla cuya sintaxis no entiende**, ignora la declaración entera y continúa con la siguiente, independientemente de si el error se ha producido en el selector, en el atributo, en el valor o en todo.
- **Si se omite un punto y coma entre dos declaraciones**, ambas declaraciones son ignoradas.
- **Si un navegador encuentra una regla-arroba que no conoce** (ej. '@mio'), ignora todo lo que haya hasta el fin de la regla-arroba.

### 1.1.8. Buenas prácticas de escritura de reglas CSS

- Una declaración en cada línea.
- Terminar todas las declaraciones con punto y coma. El *punto y coma* se utiliza para separar declaraciones por lo que no sería necesario un punto y coma después de la última declaración. Sin embargo, se aconseja escribirlo por dos motivos: no olvidarlo cuando es necesario y facilitar que se puedan añadir declaraciones al bloque de declaraciones.

## 1.2. Selectores

### 1.2.1. Selectores de tipo, universal, de clase y de identificador

Selector	Descripción	Tipo de selector	Nivel CSS
*	cualquier elemento	selector universal	2
e	un elemento de tipo <e>	selector de tipo	1
e#miID	un elemento <e> con ID igual a "miID"	selector de identificador	1
e.clase	un elemento <e> cuya clase es "clase"	selector de clase	1

Ejemplo: Seleccionar...

Para seleccionar	Selector
todos los elementos	*
elementos de tipo <p>	p
elementos <p> con identificador "importante"	p#importante
elementos <p> con clase "importante"	p.importante

### 1.2.2. Selectores de atributo

Seleccionan los elementos que tengan un atributo, o a los que tengan un valor determinado para un atributo.

Selector	Descripción	Nivel CSS
e[miAtr]	<e> si posee el atributo "miAtr"	2
e[miAtr="v"]	<e> si posee el atributo "miAtr" y este tiene el valor "v"	2
e[miAtr~="v"]	<e> si posee el atributo "miAtr" que es una lista de valores separados por espacios, y uno es "v"	2
e[miAtr^="v"]	<e> si posee el atributo "miAtr" y este tiene un valor que comienza con "v"	3
e[miAtr\$="v"]	<e> si posee el atributo "miAtr" y este tiene un valor que termina con "v"	3
e[miAtr*="v"]	<e> si posee el atributo "miAtr" y este tiene un valor que contiene el texto "v"	3
e[miAtr = "v"]	<e> si posee el atributo "miAtr" que es una lista de valores separados por guiones, y el primer valor es "v"	2

### Ejemplo: Seleccionar...

Para seleccionar	Selector
elementos con atributo placeholder	*[placeholder]
elementos input de tipo email	input[type="email"]
elementos cuyo atributo href empiece por http	*[href^="http"]
elementos cuyo atributo href termine con .com	*[href\$=".com"]
elementos cuyo atributo href contenga "elmundo"	*[href*="elmundo"]
elementos de la clase "def" (pueden ser de otras clases también)	*[class~="def"]
elementos cuyo idioma empiece por "es-" y a continuación tenga otro valor	*[lang = "es"]

### 1.2.3. Pseudoclases

Permiten **seleccionar elementos en función de su estado**. Las pseudoclases se definen con el nombre del elemento, el carácter ':' (dos puntos) y el nombre de la pseudoclase, sin dejar ningún espacio en blanco alrededor del carácter ':'.

#### Pseudoclases de enlaces

Hay dos pseudoclases que se aplican a enlaces. Los enlaces pueden estar en dos estados: **link** y **visited**.

Selector	Descripción	Nivel CSS
a:link	Enlaces no visitados	1
a:visited	Enlaces visitados	1

#### Pseudoclases que dependen de las acciones del usuario

Hay tres **pseudoclases** que se aplican a cualquier elemento (no sólo a los enlaces), dependiendo de las acciones del usuario: **active**: "se está haciendo clic sobre él", **hover**: "se está pasando el ratón sobre él" y **focus**: "recibe foco del teclado".

Selector	Descripción	Nivel CSS
e:active	<e> cuando es activado (tiempo entre que se pulsa un botón sobre él y se suelta)	1 y 2
e:hover	<e> cuando se posiciona el cursor del ratón sobre él, pero no se activa	1 y 2
e:focus	<e> cuando el foco del teclado está posicionado en él	1 y 2

### Uso de pseudoclases en los enlaces:

- Cuando se aplican varios selectores sobre un mismo elemento se pueden producir colisiones. Por ejemplo, al pasar el ratón sobre un elemento visitado, se aplican las pseudoclases `:visited` y `:hover`. Por este motivo, es importante el orden en el que se definen las pseudoclases, que debe ser el siguiente: `link`, `visited`, `hover`, `active`.
- Una técnica habitual es desactivar el subrayado por defecto a `a:link` y `a:visited` y activarlo para `a:hover`, `a:focus` y `a:active`. Además, algunos diseñadores tienen a agrupar `link` (estado inicial) y `visited` por un lado y `hover`, `focus` y `active` por otro.
- Se recomienda utilizar un estilo `:focus` para los usuarios que usan el teclado para navegar a través de los enlaces, en lugar de hacerlo con el ratón. Es frecuente que se aplique el mismo estilo que para `:hover`, pero no es obligatorio.

### Ejemplo:

```
a { text-decoration: none; }
a:link { color: #00008b; }
a:visited { color: #4682b4; }
a:hover, a:focus, a:active {text-decoration: underline; }
```

- Si se utilizan imágenes como enlaces (imágenes que están dentro de un enlace), se suele eliminar el borde:

```
a img { border: none; }
```

### **Pseudoclase target**

Si en la URL de la página, hay una referencia a un elemento dentro de la página (ej. `#parrafo`), se puede seleccionar el elemento que tenga ese identificador.

Selector	Descripción	Nivel CSS
<code>e:target</code>	<code>&lt;e&gt;</code> cuando el URI de la página selecciona un elemento dentro de la página, y el elemento <code>&lt;e&gt;</code> es el elemento seleccionado (por medio de su id)	3

### Ejemplo:

```
p:target { background-color: blue; }
```

### **Pseudoclase lang**

Si se ha definido el idioma en un documento, se puede utilizar un selector que representa un elemento en función de su idioma.

Selector	Descripción	Nivel CSS
<code>e:lang(es)</code>	los elementos <code>&lt;e&gt;</code> si están en el idioma es	2

### Pseudoclases de elementos de UI (*user interface*)

Selector	Descripción	Nivel CSS
<b>e:enabled</b>	un elemento del UI <e> que está activo	3
<b>e:disabled</b>	un elemento del UI <e> que está desactivado	3
<b>e:checked</b>	un elemento del UI <e> que está seleccionado (ej. un <i>radio-button</i> o <i>checkbox</i> )	3

### Pseudoclases estructurales

Permiten seleccionar elementos dependiendo de su situación en el HTML.

Selector	Descripción	Nivel CSS
<b>e:root</b>	un elemento <e>, raíz del documento; en HTML 4, siempre es html	3
<b>e:nth-child(n)</b>	un elemento <e>, hijo número "n" de su padre	3
<b>e:nth-last-child(n)</b>	un elemento <e>, hijo número "n" de su padre, empezando a contar desde el último	3
<b>e:nth-of-type(n)</b>	un elemento <e>, hermano número "n" de este tipo	3
<b>e:nth-last-of-type(n)</b>	un elemento <e>, hermano número "n" de su tipo, empezando a contar desde el último	3
<b>e:first-child</b>	un elemento <e>, primer hijo de su padre.	2
<b>e:last-child</b>	un elemento <e>, último hijo de su padre.	3
<b>e:first-of-type</b>	un elemento <e>, primer hermano de su tipo	3
<b>e:last-of-type</b>	un elemento <e>, último hermano de su tipo	3
<b>e:only-child</b>	un elemento <e>, único hijo de su padre	3
<b>e:only-of-type</b>	un elemento <e>, único hermano de su tipo	3
<b>e:empty</b>	un elemento <e>, que no tiene hijos (ni siquiera texto)	3

Las pseudoclases que aparecen con una n entre paréntesis (n), pueden utilizarse para seleccionar uno o varios elementos. En los selectores anteriores aparece (n), pero en realidad podría ser (an + b):

#### **e:nth-child(an + b).**

- 'a' y 'b' deben ser números enteros, positivos, negativos o cero.
- Para seleccionar los elementos, n va tomando valores 0, 1, 2, ...

Hay que tener en cuenta que el primer hijo de un elemento es el 1.

Ejemplo: Elementos p que ocupan un lugar impar entre sus hermanos. Es decir, ocupan los lugares 1, 3, 5, ...

Entonces, a=2 y b=1 (ocupan la posición 2\*0+1, 2\*1+1, 2\*2+1, ...)

```
p:nth-child(2n+1) /* Párrafos que ocupan una posición impar */
```

#### Otras características:

- :nth-child() puede tomar como argumentos 'odd' (equivalente a 2n+1), y 'even' (equivalente a 2n).

Ejemplo: Uso de los argumentos odd y even.

```
tr:nth-child(odd) /* Filas impares de una tabla */  
tr:nth-child(even) /* Filas pares de una tabla */
```

Ejemplo: b con valor negativo.

```
p:nth-child(10n-1) /* párrafos en la posición 9°, 19°, 29°... */
```

- Si 'a' tiene valor 0, se pueden omitir.

```
p:nth-child(0n+5) /* Elemento p, 5° hijo de su padre */  
p:nth-child(5) /* lo mismo */
```

- Si 'a' tiene valor 1, no es necesario escribir el 1 antes de la n.

```
p:nth-child(1n+0) /* todos los elementos p*/  
p:nth-child(n+0) /* lo mismo */  
p:nth-child(n) /* lo mismo */  
p /* lo mismo */
```

- Si 'b'=0, se puede omitir.

```
tr:nth-child(2n+0) /* filas pares de una tabla */  
tr:nth-child(2n) /* lo mismo */
```

#### Pseudoclase de negación (CSS3)

Selector	Descripción	Nivel CSS
e:not(s)	un elemento <e> que no es seleccionado por el selector simple "s"	3

Ejemplo: Seleccionar todos los elementos que no sean de tipo <p>.

```
*:not(p)
```

#### 1.2.4. Pseudo-elementos (CSS1 y 2)

Los pseudo-elementos **permiten acceder a parte de los elementos**, o incluso **crear contenido** que no existen en el documento original. Los pseudo-elementos se definen con el nombre del elemento, dos caracteres ':' (dos puntos) y el nombre del pseudoelemento, sin dejar ningún espacio en blanco alrededor de los caracteres '::'.

La notación :: se introduce en CSS3 para diferenciar pseudo-clases y pseudo-elementos. Sin embargo, en CSS1 y 2, estos pseudo-elementos se escribían con :. Por este motivo, los agentes de usuario (navegadores) aceptan ambas notaciones en los pseudo-elementos de nivel 1 y 2.

Selector	Descripción	Nivel CSS
e::first-line	primera línea de un elemento <e>	1
e::first-letter	primera letra de un elemento <e>	1
::before y ::after	Estos pseudo-elementos se utilizan para insertar contenido generado antes o después del contenido del elemento; para ello se utiliza la propiedad 'content'.	2

Ejemplo: Añadir el texto "(\*EMPIEZA UN PÁRRAFO\*)" al principio de todos los párrafos.

```
p::before { content: "(*EMPIEZA UN PÁRRAFO*)"; }
```

#### 1.2.5. Combinadores de selectores

##### Combinadores de descendientes

Selector	Descripción	Nivel CSS
e f	un elemento <f> que es descendiente de un elemento <e>	1
e > f	un elemento <f> que es hijo de un elemento <e>	2

##### Combinadores de hermanos

Selector	Descripción	Nivel CSS
e + f	un elemento <f> precedido inmediatamente por un elemento <e>	2
e ~ f	un elemento <f> precedido por un elemento <e>	3

Ejemplo: Aplicar color rojo al texto que esté dentro de la etiqueta **em** que está dentro de **strong**, y que a su vez está dentro de **h1**.

```
h1 strong em {color:red}
```

### 1.2.6. Especificidad de los selectores

La especificidad de un selector se calcula del siguiente modo:

- a: número de **selectores de identificador** que contiene.
- b: número de **selectores de clase, pseudo-clase y de atributo** que contiene.
- c: número de **selectores de tipo de elemento y pseudo-elementos** que contiene.

Teniendo en cuenta las siguientes reglas:

- Ignorar el selector universal.
- Los selectores dentro de la pseudo-clase de negación se cuentan como cualquier otro, pero el selector de negación no se cuenta como una pseudo-clase.
- Los combinadores de selectores (como >, + ó el espacio en blanco), no afectan a la especificidad.

Concatenando los tres números (a-b-c) tenemos la especificidad.

Ejemplo: Calcular la especificidad de los siguientes selectores.

Selector	A	B	C	Especificidad
h1	0	0	1	001
.clase	0	1	0	010
#id	1	0	0	100
html > head + body ul#nav *.home a:link	1	2	5	125
html h1	0	0	2	002

Las especificidades se comparan comparando los tres componentes en orden: la especificidad con un valor A mayor es más específica; si los dos valores A son iguales, entonces la especificidad con un valor B mayor es más específica; si los dos valores B también son iguales, entonces la especificidad con un valor C más grande es más específica; si todos los valores son iguales, las dos especificidades son iguales.

## 1.3. Aplicar CSS al HTML: Hojas de estilo en Cascada

### 1.3.1. Estilos en línea

Los estilos en línea se definen en un elemento y sólo se aplican al elemento en el que se definen. Para ello se utiliza el atributo **style** en el elemento, como en el siguiente ejemplo:

```
<h1 style="color:blue; background-color:red">...</h1>
```

#### Notas

- En general, estos estilos se deben evitar, porque no tiene las ventajas de las hojas de estilo (facilitar el mantenimiento) y no llevan asociado el atributo media.
- No obstante, tienen algunas aplicaciones prácticas. Por ejemplo, los correos electrónicos en formato HTML. Los programas de correo no reconocen los estilos embebidos con la etiqueta **<style>** ni las hojas de estilo externas. Los diseñadores se ven forzados a incorporar estilos en línea para dar estilo a sus correos.

### 1.3.2. Estilos incrustados

Las hojas de estilo definidas en el elemento style en la cabecera **<head>** se denominan internas, embebidas o incrustadas. Se pueden utilizar si la página tiene un estilo único, independiente del resto de las páginas. Las reglas se definen en el elemento **style** en el **head**, como en el siguiente ejemplo:

```
<style media="all">
    h1 {color:blue; background-color:red; }
    h2 {color:blue; }
</style>
```

#### Atributos del elemento **style**

<b>media="dispositivo"</b>	Dispositivo de visualización en el que se aplica el estilo (opcional).
<b>type="text/css"</b>	Lenguaje que se utiliza en la hoja de estilo (obligatorio en HTML4.01 y XHTML 1.0/1.1; opcional en HTML5).

### 1.3.3. Hojas de estilos externas

Los estilos se pueden colocar directamente en archivos separados del HTML. Los archivos tendrán extensión **.css**. Este documento sólo debe tener reglas CSS y comentarios. Algunas ventajas de utilizar estos archivos son:

- En el diseño es posible **reutilizar estilos**.
- Es más fácil que el sitio web tenga un **diseño homogéneo**.
- Se facilita el **mantenimiento** del sitio web.
- **Rapidez en la descarga** de las páginas de un sitio: el archivo con los estilos se descarga sólo una vez, y luego se accede a él en la caché.

Para vincular una hoja de estilos externa al archivo HTML, se utiliza el elemento `link`, dentro del `head`, como en el siguiente ejemplo:

```
<link rel="stylesheet" href="miEstilo.css" media="screen">
```

Atributos del elemento `link`:

<code>rel="stylesheet"</code>	Relación entre el documento CSS y el documento enlazado (obligatorio).
<code>href="miEstilo.css"</code>	URL del documento enlazado (obligatorio).

\* Ver los atributos `media` y `type` en el apartado anterior.

Nota: Se pueden utilizar varios elementos `link` para enlazar varias hojas de estilo.

#### 1.3.4. Importar hojas de estilos con `@import`

La regla '`@import`' permite a los usuarios importar hojas de estilo desde otras hojas de estilo. Se pueden importar varias hojas de estilos. La palabra clave '`@import`' debe ir seguida por el URI de la hoja de estilo a incluir. También se permite una cadena; que será interpretada como si contuviera `url(...)` en torno a ella. Cualquier regla '`@import`' debe preceder a todas las otras reglas en una hoja de estilo. Los navegadores ignorarán toda regla '`@import`' que aparezca dentro de un bloque o después de otras reglas CSS.

Ejemplo: Importar varias hojas de estilo dentro del elemento `<style>`.

```
<style>
    @import url("texto.css") print; /* Para impresión */
    @import "imagenes.css" all;      /* Para todos los medios */
    @import "tablas.css";           /* Para todos los medios */
    ...
</style>
```

Ejemplo: En un documento CSS, las reglas '`@import`' aparecen al principio.

```
@import url("miEstilo.css");
...
```

Ejemplo: El navegador ignorará la segunda regla `@import`, porque está después de una declaración CSS.

```
@import url("texto.css") print;
h1 {color: red}
@import "imagenes.css"
```

Ejemplo: El navegador ignorará la `@import` porque aparece dentro de un bloque `@media`.

```
@media print {
    import url("texto.css");
}
```

### 1.3.5. Herencia<sup>1</sup>

La herencia en CSS es el mecanismo mediante el cual ciertas propiedades de un elemento padre se transmiten a sus hijos.

Ejemplo: A continuación, vamos a presentar un ejemplo concreto donde estudiar la herencia.

```
<!DOCTYPE html>
<html lang="es">
    <head>
        <meta charset="utf-8">
        <title>Herencia</title>
        <style>
            html {
                font: 75% Verdana, sans-serif;
                background-color: blue;
                color: white;
            }
        </style>
    </head>
    <body>
        <h1>Título</h1>
        <p>Un párrafo de texto.</p>
    </body>
</html>
```

#### Qué propiedades son heredables

No todas las propiedades CSS se heredan, porque no tendría sentido. Por ejemplo, no sería lógico que el ancho de un elemento fuera heredable. En general, podemos decir que se heredan las propiedades relacionadas con el texto: fuente, tamaño, color, estilo...

No se heredan las propiedades que afectan a la "caja" donde está el elemento: bordes, márgenes, fondo, etc. Normalmente el sentido común dicta qué propiedades se heredan y cuáles no, pero para estar seguros debemos consultar las tablas de propiedades de la especificación CSS: <https://www.w3.org/Style/CSS/all-properties.en.html> (*Inherited: yes/no*).

#### En el ejemplo anterior...

Serían heredables las propiedades `font` y `color`. Sin embargo, la propiedad `background-color`, que se refiere a la caja, no es heredable. Esto nos puede sorprender porque el color de fondo de toda la página es azul. El motivo es que el valor por defecto del color de fondo es `transparent` (transparente).

---

<sup>1</sup> <http://mosaic.uoc.edu/ac/le/es/m6/ud2/>

## Qué elementos heredan de sus padres las propiedades heredables

Todos los elementos de un documento HTML heredan de su padre las propiedades heredables, excepto el elemento raíz (html), que no tiene padre.

### En el ejemplo de herencia anterior...

Para el <html>, font-size es el 75%. Pero, ¿el 75% de qué? Sería el 75% del tamaño heredado de la fuente. Como HTML no hereda (porque no tiene un elemento padre), sería el 75% del tamaño de la fuente por defecto.

## Qué valores se heredan

Se heredan los valores computados. Por ejemplo, si el tamaño de una fuente está definido como "1.2em" para un <div>, el navegador, calcula cuál es el valor en píxeles de ese atributo, y es el que heredan sus hijos.

### En el ejemplo anterior...

El tamaño de la fuente en el <h1> y en el <p>, ¿sería el 75% del tamaño de la fuente del <html>? No, porque en CSS se heredan valores computados. Es decir, se calcula cuál es el 75% del tamaño de la fuente por defecto, y ese resultado es el que heredan los elementos hijos. Por ejemplo, si el tamaño de la fuente por defecto es 16px, el 75% sería 12px, y ese sería el tamaño que tendrían html y los elementos que hereden este valor.

Pero, si visualizamos el documento, nos sorprenderá que el tamaño del <h1> es superior al tamaño del <p>. ¿Por qué, si ambos heredan el tamaño? El motivo se explica a continuación.

## ¿Qué tiene prioridad: la herencia o la definición de propiedades para ese elemento?

El hecho de que las propiedades se hereden no quiere decir que siempre se apliquen. Por ejemplo, si se define el color del texto para un elemento, será ese color el que se aplique, no el heredado.

### En el ejemplo anterior...

El motivo es que existe una regla del navegador que da estilo a h1, y las reglas definidas tienen más prioridad que los valores heredados.

## Forzar la herencia

Mediante la palabra clave `inherit` se puede forzar la herencia, incluso para propiedades que no se heredan normalmente (esto sólo es compatible con IE8 y posteriores).

### En el ejemplo anterior...

Si hubiéramos incluido la siguiente regla, el tamaño de la fuente sería el mismo de su elemento padre.

```
h1 { font-size: inherit; }
```

### 1.3.6. Cascada

CSS significa "hojas de estilo en cascada". La cascada es el mecanismo que controla el resultado final cuando se aplican varias declaraciones CSS contradictorias. Para encontrar el valor para una combinación de elemento/propiedad, las aplicaciones de usuario aplican los siguientes criterios:

- Encontrar todas las declaraciones que se aplican al tipo de medio seleccionado.
- En caso de colisiones, se aplicará la que tenga más "importancia".
- En caso de colisiones, se aplicará la que tenga un selector más "específico".
- Si sigue habiendo colisiones, clasificar por el orden de declaración. Aplicar las últimas que se definen.

A continuación, se explicarán los conceptos de importancia, especificidad y orden.

#### Importancia

La importancia de una declaración depende de donde se haya especificado. El nivel de importancia (de menor a mayor), es el siguiente:

1. Hoja de estilos de agente de usuario (*user-agent-style*).
2. Declaraciones normales en hojas de estilo de usuario (*user-style*).
3. Declaraciones normales en hojas de estilo de autor (*author-style*).
4. Declaraciones importantes en hojas de estilo de autor.
5. Declaraciones importantes en hojas de estilo de usuario.

Una **hoja de estilos de agente de usuario** es la hoja de estilo integrada en el navegador. Cada navegador tiene sus propias reglas sobre cómo mostrar varios elementos de HTML si no se especifica ningún estilo.

Una **hoja de estilos de usuario** es una hoja de estilos que ha especificado el usuario. No todos los navegadores son compatibles con las hojas de usuario, pero pueden ser muy útiles para personas con discapacidades.

Una **hoja de estilos de autor** es una hoja de estilos proporcionada con un documento HTML.

Las **declaraciones normales** son las que utilizamos normalmente. Lo contrario son las **declaraciones importantes**, que son las declaraciones que van seguidas de una directiva **!important**.

Como vemos, las declaraciones importantes en la hoja de estilos del usuario tienen la máxima prioridad. Esto permite, por ejemplo, que el usuario pueda visualizar la página de algún modo concreto, con independencia de la hoja de estilos que haya definido el autor de la página.

Ejemplo: Declaración **!important**.

```
h1 {  
    font-size: 300% !important;  
}
```

## Especificidad

- Las declaraciones en el atributo style tienen la máxima especificidad.
- Para el resto, se calcula la especificidad como se vio antes.

## Orden

- Si dos declaraciones afectan al mismo elemento, tienen la misma importancia y la misma especificidad, se tiene en cuenta el orden en que aparecen. La declaración que aparece más tarde prevalece sobre las anteriores.

Ejemplo: En este ejemplo, el primer párrafo, que tiene identificador #special, tendrá fondo rojo, porque se le aplica la regla más específica. Sin embargo, el segundo párrafo, tendrá fondo cyan, ya que es la última que aparece en el documento.

```
<style>
    #special { background-color: red; }
    p { background-color: yellow; }
    p { background-color: cyan; }
</style>

....
<p id="special">Un párrafo de texto.</p>
<p>Un segundo párrafo de texto.</p>
```

Ejemplo: El texto marcado como strong dentro de h1, será rojo, mientras que el texto marcado como strong que no esté dentro de h1, será azul.

```
h1 strong { color: red; }
strong { color: blue; }

<h1>Una cabecera <strong>importante</strong> de tipo h1</h1>
```

## 2. Dar estilo a los elementos

### 2.1. Unidades

Las unidades de medida se utilizan, entre otras, para definir la altura, anchura y márgenes de los elementos, y para establecer el tamaño de la letra del texto. Las medidas se especifican con un signo – o + (opcional), un número y la unidad de medida, todo sin espacios en blanco (sólo se admite no escribir ninguna medida para el valor 0).

Ejemplo: Las siguientes son ejemplos de medidas de longitud.

```
12px /* Correcto */
-1.2em /* Correcto */
.2em /* Correcto */
0 /* Correcto */
12 /* Incorrecto: falta la unidad de medida */
1 em /* Incorrecto: hay un espacio entre el número y la unidad */
```

Las unidades de medida se clasifican en absolutas y relativas.

### 2.1.1. Unidades absolutas

Las unidades de medida absolutas son útiles solamente cuando las propiedades del medio físico de salida son conocidas.

- **in** Pulgadas (*inches*) (1 inch = 2,54 cm)
- **cm** Centímetros
- **mm** Milímetros
- **pt** Punto (1 pt = 1/72 pulgadas)
- **pc** Pica (1 pc = 12 puntos)

### 2.1.2. Unidades relativas

- **em:** La unidad '**em**' es relativa respecto al tamaño de letra empleado. Su referencia es la altura de la letra 'M' mayúscula del tipo de fuente (font-family) utilizado en el elemento.
  - La unidad em es relativa al valor computado de la propiedad 'font-size' del elemento en el que se usa.
  - Sin embargo, si 'em' aparece en la propiedad 'font-size', será relativa al valor computado de la propiedad 'font-size' del elemento padre.
- **ex:** La unidad '**ex**' es relativa al tamaño de letra empleado. Su referencia es la altura de la letra "x" minúscula. Además, esta unidad está definida también para aquellas fuentes que no contienen la letra "x".

Cuando se utilizan en el elemento raíz, <html>, las unidades **em** y **ex** se refieren al valor por defecto de la propiedad.

- **px:** La unidad '**px**' (píxeles) es relativa respecto a la pantalla del usuario (en el nivel 3 de CSS 1in=96px, el tamaño real de cada píxel en el monitor dependerá de la resolución en la que esté configurado y del escalado, por lo que las unidades de medida absolutas pueden no coincidir con su medida física <sup>2</sup>).

#### Ejemplos con px y em:

```
h1 { line-height: 1.2em; }      /* La altura de línea de los elementos h1  
será un 20% mayor que el tamaño  
de la fuente de los elementos h1  
*/  
  
h1 { font-size: 1.2em; }        /* La propiedad font-size de los  
elementos h1 será un 20% mayor que  
el tamaño de la fuente del  
elemento padre */  
  
p { font-size: 11px;  
    text-indent: 2em; }          /* El texto se indentará 22px */  
  
div { font-size: 15px; }  
  
div p { font-size: 1.2em; }      /* Dentro de este elemento, el  
texto será un 20% mayor que en el  
resto del div */
```

<sup>2</sup> <https://www.mydevice.io/>

## Porcentajes

Es también una medida relativa. El formato de un valor en porcentaje es un <número> seguido inmediatamente por '%'. Los valores expresados en porcentajes son siempre relativos a otra medida.

Para cada propiedad que admite porcentajes, el estándar define a qué valor se refiere el porcentaje: otra propiedad para el mismo elemento, una propiedad para un elemento antepasado u otro valor.

Ejemplo: El tamaño de la fuente será 10px; la altura de la línea será el 120% de 10px, es decir, 12px. Es lo mismo indicarlo con 120% o con 1.2em.

```
p { font-size: 10px }
p { line-height: 120% } /* 120% de 'font-size' */
p { line-height: 1.2em } /* Equivalente a la anterior */
```

Ejemplo: Establecer la anchura de los elementos <div>: el ancho del div#principal será el 60% del ancho del div#wrapper, es decir, 576px.

```
div#wrapper { width: 960px; }
div#principal { width: 60%; }

<div id="wrapper">
    <div id="principal">
        </div>
    </div>
```

## 2.2. Cadenas

Las cadenas pueden escribirse con comillas simples o dobles. Para incluir una comilla dentro de una cadena, se deben escapar con la barra invertida \.

## 2.3. URL

Las direcciones URL se especifican de la siguiente forma:

```
url(dirección)
```

La dirección puede ir entre comillas simples o dobles.

Cuando en la dirección URL aparecen paréntesis, espacios, comas o comillas, deben ir escapadas con una barra invertida (): \(. \). \' \".

Las direcciones relativas se interpretan como relativas respecto al archivo donde aparecen. Por ejemplo, si se utilizan en un archivo de hoja de estilo, serán relativas a ese documento, no al documento HTML que utiliza la hoja.

Ejemplo: En la segunda regla, se supone que la imagen (mi\_imagen.png) está en el mismo directorio que la hoja de estilos donde aparece.

```
background-image: url("http://www.example.com/mi_imagen.png");
background-image: url("mi_imagen.png");
```

## 2.4. Color

Para especificar un color en las hojas de estilo, se puede utilizar un nombre de color predefinido, o un valor.

### 2.4.1. Nombres de colores

- CSS1 y CSS2 adoptaron los 16 nombres de colores estándar de HTML4.01.
- CSS2.1 añadió el color 17 (orange)

maroon #800000	red #ff0000	orange #ffa500	yellow #ffff00	olive #808000
purple #800080	fuchsia #ff00ff	white #ffffff	lime #00ff00	green #008000
navy #000080	blue #0000ff	aqua #00ffff	teal #008080	
black #000000	silver #c0c0c0	gray #808080		

- CSS3 proporciona 147 colores. Estos colores, llamados también nombres de colores de X11, son los proporcionados originalmente con el sistema X Window de Unix. Una lista de estos colores: <http://www.w3.org/TR/2011/REC-css3-color-20110607/#svg-color>.

### 2.4.2. Valores de los colores: RGB, RGBa, HSL, HSLa

#### Colores RGB

- **#RRGGBB** RR, GG, BB: Números hexadecimales.
- **#RGB** Notación simplificada (#RGB = #RRGGBB).
- **rgb(R,G,B)** R, G, B: Números enteros entre 0 y 255, ambos incluidos.
- **rgb(r%,g%,b%)** r%. g% y b%: valores entre 0.0 y 100.0 que indican el porcentaje del cada color.

#### Colores RGBA (CSS3)

El color RGBa permite especificar un color y aplicarle un nivel de transparencia. La "a" de RGBa significa "alpha", que es un canal adicional que controla el nivel de transparencia. Los colores rgba() se definen de forma parecida a los rgb():

- **rgba(R,B,G, a)** R, G, B: Números enteros entre 0 y 255, ambos incluidos.  
La a es un valor decimal, en la escala de 0.0 (completa transparencia) a 1.0 (completa opacidad).
- **rgba(r%,g%,b%, a)** r%. g% y b%: valores entre 0.0% y 100.0%  
La a tiene el mismo valor que en el caso anterior.

Ejemplo: Distintos modos de indicar el color rojo.

```
em { color: rgb(255,0,0) }      /* rango de enteros: 0 y 255 */  
em { color: rgba(255,0,0,1) }    /* lo mismo, con opacidad de 1*/  
em { color: rgb(100%,0%,0%) }    /* rango de reales: 0.0% - 100.0% */  
em { color: rgba(100%,0%,0%,1) } /* lo mismo, con opacidad de 1*/
```

Ejemplo: Color rojo con distintos niveles de transparencia.

```
h1 { color: rgba(255, 0, 0, 1) }    /* Rojo */  
h1 { color: rgba(255, 0, 0, 0.5) }   /* Rojo semi-transparente */  
h1 { color: rgba(255, 0, 0, 0.25) }  /* Rojo más transparente */
```

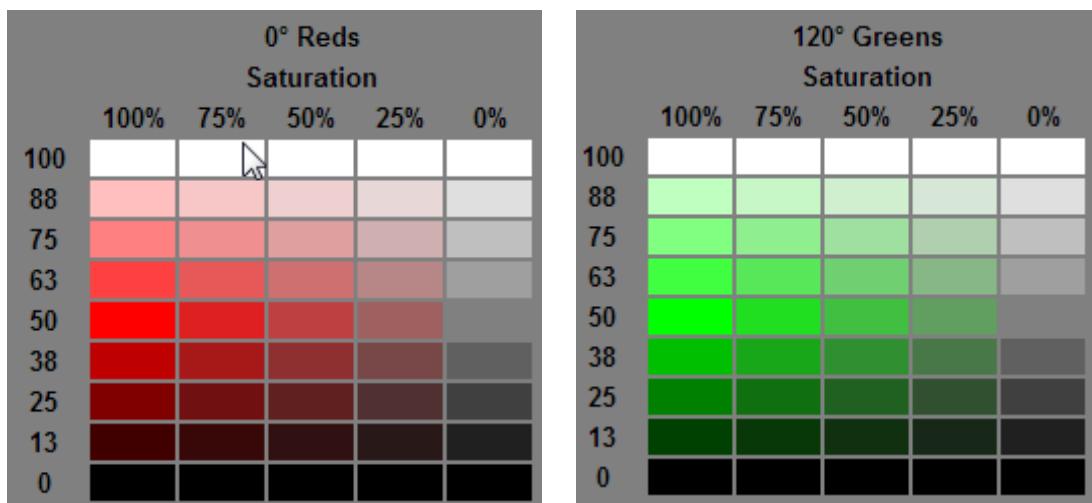
### Colores HSL (CSS3)

CSS3 introduce la posibilidad de especificar los colores con sus valores HSL: H, tono (*hue*); S, saturación, L, luminosidad.

Los colores hsl se definen con **hsl(H, s%, l%)**:

- **H: tono, hue:** Valor numérico en el círculo cromático, entre 0 y 360.  
0=360 es rojo, verde 120 y azul 240.
- **s%: saturación:** 100.0% es total saturación; 0.0% es un tono de gris.
- **l%: luminosidad:** 0.0% es negro, 100.0% es blanco y 50% es normal.

Dos ejemplos de HSL, para los tonos rojo (H=0 o H=360) y verde (H=120) serían los siguientes (tomados de <https://www.w3.org/TR/css-color-3/#hsl-examples>):



Ejemplo: Distintos colores basados en el verde (120), variando la saturación y la luminosidad.

```
* { color: hsl(120, 100%, 50%) } /* lima */
* { color: hsl(120, 100%, 25%) } /* verde oscuro */
* { color: hsl(120, 100%, 75%) } /* verde claro */
* { color: hsl(120, 75%, 75%) } /* verde pastel */
```

### Colores HSLA (CSS3)

La función hsla() permite especificar un color HSL y un nivel de transparencia (de igual modo que rgba).

Ejemplo: Color rojo con distintos niveles de transparencia.

```
* { color: hsl(0, 100%, 50%) }      /* rojo */
* { color: hsla(0, 100%, 50%, 1) } /* lo mismo, con opacidad de 1 */
* { color: hsla(0, 100%, 50%, 0.5) } /* rojo semi-transparente */
```

## Compatibilidad:

Algunos navegadores (ej. Internet Explorer 8 y anteriores) no soportan las funciones RGBa, HSL y HSLa. Una solución es proporcionar dos reglas CSS. La primera asigna un color en RGB que se aproxime al color que deseamos utilizar. La segunda asignará el color CSS3 que queremos utilizar. Los navegadores que no soportan las nuevas funciones ignoran la segunda regla, mientras que en el resto, la segunda regla sobrescribe la primera.

Ejemplo: Para utilizar un gris con transparencia.

```
h1 { color: rgb(120, 120, 120);  
      color: rgba(0, 0, 0, 50%); }
```

## 2.5. Fuentes

Propiedad	Descripción	Valores
font-family	Familias de fuentes	[ [ <nombre-familia>   <familia-genérica> ] [, <nombre-familia>   <familia-genérica> ]* ]
font-style	Estilo de la fuente	[ normal   italic   oblique ]
font-variant	Convierte las minúsculas a mayúsculas pero mantienen un tamaño inferior a las mayúsculas	[ normal   small-caps ]
font-weight	Intensidad de la fuente	[ normal   bold   bolder   lighter   100   200   300   400   500   600   700   800   900 ]
font-size	Tamaño de la fuente	[ xx-small   x-small   small   medium   large   x-large   xx-large ]   [ larger   smaller ]   <longitud>   <porcentaje> ]
font	Atajo para establecer el resto de propiedades sobre las fuentes a la vez	[ [ [ <font-style>    <font-variant>    <font-weight> ]? <font-size> [ / <line-height> ]? <font-family> ]   caption   icon   menu   message-box   small-caption   status-bar ]

### Familia de fuentes

**font-family**. Lista de familias de fuentes posibles separadas por comas. La lista está ordenada: primero se intentará utilizar la primera fuente; si no está disponible, se intentará la segunda, etc.

#### Fuentes genéricas:

- Se recomienda que la última fuente sea una fuente genérica.
- Las fuentes genéricas son: **serif, sans-serif, monospace, fantasy y cursive** (**las dos últimas** apenas se utilizan).
- Los nombres de las fuentes genéricas son palabras reservadas y no se deben escribir entre comillas. Se suelen escribir en minúsculas.

#### Fuentes no genéricas:

- Los nombres de las fuentes (no genéricas) se suelen escribir con la primera letra mayúscula. Se pueden escribir entre comillas o sin comillas, pero si contienen espacios o caracteres especiales, se recomienda escribirlas entre comillas.
- Se debe elegir una familia de fuentes que estén disponibles en Windows, Mac y Linux.

### Estilo de la fuente

**font-style**. Este atributo puede tomar los valores **normal, italic y oblique**.

- El navegador buscará una fuente que sea de ese tipo.
- Si se especifica **italic** y no existe, se buscará una fuente **oblique**.

## Variante de la fuente

**font-variant.** Este atributo puede tomar los valores **normal** y **small-caps** (versalles o versalitas).

- En la fuente **small-caps** el texto aparecerá en mayúsculas, pero en un tamaño algo más pequeño y con diferentes proporciones.

## Grosor de la fuente

**font-weight.** Especifica el peso de la fuente.

- Los valores van de **100** a **900**, donde **100** es el menos grueso.
- El valor normal es **400** y **bold** es igual a **700**.

## Tamaño de la fuente

**font-size.** Tamaño de la fuente. Puede tomar los valores:

- **xx-small, x-small, small, medium, large, x-large, xx-large:** Son tamaños absolutos, definidos por el navegador.
- **larger, smaller:** Son tamaños referidos al tamaño de la fuente del elemento padre. Por ejemplo, si el padre tenía small, larger indica que en este elemento el tamaño será medium.
- **Longitud y porcentaje.**

## Propiedad abreviada

**font.** Permite definir las cinco propiedades anteriores y la propiedad **line-height**.

Los valores **caption, icon, menu, message-box, small-caption** y **status-bar** se corresponden a las fuentes utilizadas por el sistema para rotular ciertos elementos.

Ejemplo: Definir la fuente con la propiedad abreviada Font.

```
font: 12px/1.4em Arial; /* Correcto */  
font: 12px; /* Incorrecto. font-family es obligatorio. */
```

### 2.5.1. Fuentes web @font-face

Las fuentes web son las fuentes que pueden mostrarse en el navegador sin necesidad de que el usuario las tenga instaladas en su ordenador ya que se descargan automáticamente.

#### Sintaxis

La regla @font-face permite definir una nueva font-family si disponemos del archivo donde se está la fuente. Una vez definida, se puede utilizar como cualquier otra fuente. La sintaxis de la regla @font-face es la siguiente:

```
@font-face {  
    font-family: nombre_que_queremos_usar;  
    src: url(...) [, url(...)];  
    ...  
}
```

Donde:

- src: Ruta dónde se encuentra el fichero con la fuente. El fichero puede estar en otro sitio Web. Se pueden definir varias propiedades src o varias rutas en la misma src.

Ejemplo:

```
@font-face {  
    font-family: 'Vintage';  
    src: url("fuentes/vintage.ttf");  
}  
  
h1 { font-family: "Vintage", sans-serif;}
```

### Compatibilidad de la regla @font-face<sup>3</sup>

Esta regla estaba incluida en CSS2, se eliminó en CSS2.1 y se ha incluido de nuevo en CCS3. Está disponible en algunos navegadores, como Internet Explorer 9, Safari 5, Firefox 7. Internet Explorer 8 y anteriores no soportan la regla @font-face. Por este motivo, es necesario utilizar fuentes alternativas.

### Formatos de las fuentes y compatibilidad

Los formatos de fuentes disponibles son:

- **EOT** (*Embedded OpenType fonts*): Fuente desarrollada por Microsoft.
- **WOFF** (*Web Open Font Format*): Recomendado por la recomendación W3C WOFF File Format 1.0 de diciembre de 2012.
- **SVG fonts**: Fuentes definidas como formas SVG. El soporte para este tipo de fuentes se está retirando de los navegadores, porque esta característica se eliminó del SVG 2.0.
- **TTF** (*True Type Fonts*): Formato estándar de tipos de letras escalables utilizado por Apple y Microsoft.
- **OTF** (*OpenType Fonts*): Formato de tipos de letra escalables basado en TTF.
- **WOFF 2.0**: Formato WOFF más comprimido. Está siendo desarrollado por un grupo de trabajo de W3C.

Formatos y compatibilidad con los navegadores:

Fuente	IE	Firefox	Chrome	Safari	Opera
<b>EOT</b>	6+	-	-	-	-
<b>WOFF</b>	9+	3.6+	5+	5.1+	11.5+
<b>TTF/OTF</b>	9+ (soporte parcial)	3.5+	4+	3.1+	10.1+
<b>WOFF 2.0</b>	-	39+	36+	10	23+

<sup>3</sup> La información sobre compatibilidades fue obtenida de <http://caniuse.com>

### Qué formato de fuente elegir

Habrá que comprobar que la fuente que hemos elegido es aceptada por la mayoría de los navegadores (vemos que el formato con más compatibilidad es WOFF). Podríamos utilizar una fuente de ese formato, o proporcionar dos ficheros de fuentes. Por ejemplo, uno .eot y otro .ttf, e incluirlos en la regla @font-face.

#### Ejemplo:

```
@font-face {  
    font-family: 'Vintage';  
    src: url('fuentes/vintage.eot'),  
         url('fuentes/vintage.ttf'));  
}  
  
h1 { font-family: "Vintage", sans-serif;}
```

#### Obtener fuentes

Existen páginas web con fuentes que podemos descargar para nuestro sitio web. Una página con fuentes libres para uso comercial es <http://www.fontsquirrel.com/>. En esta página también se puede subir una fuente TTF y obtener las fuentes WOFF y WOFF2 equivalentes.

#### Utilizar Google Fonts / Google Fonts API

Google Fonts es un servicio de alojamiento de fuentes libres (<https://fonts.googleapis.com/>). Las fuentes pueden utilizarse en nuestras páginas web sin necesidad de alojarlas en nuestro servidor. Esto tiene una ventaja: evitamos tráfico de red a nuestro servidor; y un inconveniente: la fuente puede no estar disponible.

En la página de Google nos indica cómo utilizarlas en nuestra página Web: en este caso, no escribimos la directiva @font-face en nuestro CSS, sino que Google nos proporciona un archivo con el CSS donde está definida.

#### Ejemplo:

```
<link href="https://fonts.googleapis.com/css?family=Shrikhand"  
      rel="stylesheet">  
  
h1 { font-family: font-family: 'Shrikhand', cursive;}
```

## 2.6. Texto

Propiedad	Descripción	Valores
text-indent	Desplazamiento de la primera línea del texto	[ <longitud>   <porcentaje> ]
text-align	Alineamiento del texto	[ left   right   center   justify ]
text-decoration	Efectos de subrayado, tachado, parpadeo	[ none   [ underline    overline    line-through    blink ] ]
letter-spacing	Espacio entre caracteres	[ normal   <longitud> ]
word-spacing	Espacio entre palabras	[ normal   <longitud> ]
text-transform	Transformaciones del texto a mayúsculas/minúsculas	[ capitalize   uppercase   lowercase   none ]
white-space	Comportamiento de los espacios dentro de los elementos	[ normal   pre   nowrap   pre-wrap   pre-line ]

**text-align.** Cómo se alinearán el texto (o las imágenes) dentro de un elemento. Esta propiedad se asigna al elemento contenedor (párrafo, cabecera, div, etc.).

**Alinear imágenes:** Las imágenes son elementos en línea, lo que significa que pueden mezclarse con el texto en la misma línea. Y también significa que la alineación de imágenes básicas se controla a través de la misma propiedad: **text-align**. En el caso de las imágenes, el valor *justify* no tiene sentido.

**letter-spacing.** Si es una longitud, indica el espacio entre caracteres a mayores del espacio por defecto entre los caracteres (definido por la fuente o el agente de usuario).

**word-spacing.** Si es una longitud, este valor indica el espacio entre palabras a mayores del espacio entre las palabras por defecto (definido por la fuente o el agente de usuario).

**white-space.** Declara cómo son tratados los espacios en blanco y los saltos de línea dentro del elemento. Los valores posibles son:

- **pre.** Respeta los espacios y los saltos de línea. No ajusta el texto al espacio disponible.
- **pre-wrap.** Respeta los espacios y los saltos de línea. Ajusta el texto al espacio disponible.
- **nowrap.** Sustituye los espacios y los saltos de línea por un espacio. Escribe todo el texto en una línea. (como la etiqueta `<pre>`).
- **pre-line.** Sustituye los espacios consecutivos, por un espacio. Pero respeta los saltos de línea. Ajusta el texto al espacio disponible.
- **normal** (valor por defecto). Considera los espacios y los saltos de línea como espacios. Ajusta el texto al espacio disponible.

## 2.7. Fondo

Propiedad	Descripción	Valores
<code>color</code>	Color del primer plano	<code>&lt;color&gt;</code>
<code>background-color</code>	Color de fondo	[ <code>&lt;color&gt;</code>   transparent ]
<code>background-image</code>	Imagen de fondo	[ url(...)   none ]
<code>background-repeat</code>	Repetición de la imagen de fondo	[ repeat   repeat-x   repeat-y   no-repeat ]
<code>background-attachment</code>	Desplazamiento de la imagen de fondo	[ scroll   fixed ]
<code>background-position</code>	Posición de la imagen de fondo	[ [ <porcentaje>   <longitud> ] left   center   right ] [ <porcentaje>   <longitud> ] top   center   bottom ]   [ left   center   right ]    [ top   center   bottom ] ]
<code>background</code>	Propiedades individuales relacionadas con el fondo	[ <background-color>    <background-image>    <background-repeat>    <background-attachment>    <background-position> ]

Las siguientes propiedades se utilizan cuando se quiere definir una imagen de fondo:

**background-image.** Imagen de fondo. Si se utiliza una ruta relativa, esta debe ser relativa al archivo donde se define esta propiedad (hoja de estilos o documento HTML).

**background-repeat.** Por defecto, una imagen de fondo rellena el elemento contenedor repitiendo la imagen horizontal y verticalmente cuantas veces sea necesario. Este comportamiento se puede modificar con la propiedad `background-repeat` que puede tomar cuatro valores:

- **repeat:** La imagen se repite horizontal y verticalmente (valor por defecto).
- **repeat-y:** La imagen se repite verticalmente, a lo largo del eje y (vertical).
- **repeat-x:** La imagen se repite horizontalmente, a lo largo del eje x (horizontal).
- **no-repeat:** La imagen no se repite, sólo se muestra una vez.

**background-position.** Si la imagen sólo aparece una vez, esta propiedad define su posición dentro del elemento contenedor: se indica primero la posición horizontal y luego la vertical. La posición puede tomar tres valores: *medidas fijas*, *porcentajes* o *palabras reservadas*.

- La posición horizontal se puede definir con: left, right, center.
- La posición vertical se puede definir con: top, bottom, center.

**background-attachment.** Determina cómo se visualiza la imagen de fondo cuando hacemos scroll:

- scroll: La imagen se desplaza con el texto.
- fixed: La imagen no se mueve.

## Varias imágenes de fondo en CSS3

CSS3 permite definir varias imágenes de fondo.

```
#fondos{  
    background: url(fondo3.png) bottom right no-repeat,  
              url(fondo2.png) center no-repeat,  
              url(fondo1.gif) center repeat;  
  
    width:300px;  
}
```

## 2.8. Listas

En los elementos **ol** y **ul** se pueden aplicar los siguientes estilos.

list-style-type	Estilo aplicable a los marcadores visuales de las listas	[ disc   circle   square   decimal   decimal-leading-zero   lower-roman   upper-roman   lower-greek   lower-latin   upper-latin   armenian   georgian   lower-alpha   upper-alpha   none ]
list-style-image	Imagen aplicable a los elementos de las listas	[ url("http://...")   none ]
list-style-position	Posición dentro de la lista de los elementos marcadores de las listas	[ inside   outside ]
list-style	Permite establecer el estilo de la lista, la imagen y/o la posición	[ <list-style-type>    <list-style-position>    <list-style-image> ]

**list-style-type.** Define qué símbolo (disco, letras, números, ninguno) se verán delante de los elementos de una lista (no se puede cambiar el color, tamaño, etc.). Los valores que puede tomar son:

- **none**: Elimina el símbolo.
- **disc**, **circle**, **square**: Para listas desordenadas **<ul>**.
- Resto de valores: para listas ordenadas.

## @counter-style

CSS3 introduce nuevos elementos utilizando la regla **@counter-style**.

Ejemplo:

```
@counter-style circled-alpha {  
    system: fixed;  
    symbols: □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □;  
    suffix: " ";  
}  
.items {  
    list-style: circled-alpha;  
}
```

**list-style-image.** Permite utilizar una imagen en lugar de los valores predeterminados para las listas desordenadas (*disc*, *circle*, *square*). La imagen de *list-style-image* no admite parámetros *width* y *height*, por lo que habrá que asegurarse de que la imagen tiene el tamaño adecuado.

## 2.9. Tablas

Las siguientes propiedades se aplican a la tabla (*table*):

Propiedad	Descripción	Valores
<code>caption-side</code>	Posición del título de respecto la tabla	[ <code>top</code>   <code>bottom</code> ]
<code>table-layout</code>	Control del algoritmo usado para el formato de las celdas, filas y columnas	[ <code>auto</code>   <code>fixed</code> ]
<code>border-collapse</code>	Selección del modelo de los bordes	[ <code>collapse</code>   <code>separate</code> ]
<code>border-spacing</code>	Espaciado entre los bordes de celdas adyacentes	<longitud> <longitud>?
<code>empty-cells</code>	Visibilidad de los bordes de celdas sin contenido	[ <code>show</code>   <code>hide</code> ]

Para ver los bordes de la tabla habrá que utilizar la propiedad **border** de la tabla (*table*) o de la celda (*th* o *td*). También se pueden utilizar las propiedades *padding* y *margin* para estos elementos.

```
table {
    margin: 20px;
    border: 1px solid black;
}
```

**caption-side.** Determina dónde se verá el título (<caption>), si existe: sobre la tabla (top) o debajo de la tabla (bottom).

**border-collapse.** Define cómo queremos que se vean los bordes de las celdas.

- `separate` (default): Hace que veamos las celdas adyacentes separadas
- `collapse`: Hace que los bordes de celdas adyacentes se unan, y se vean como uno solo.

**border-spacing.** Se establece el espacio que hay entre los bordes de las celdas. Esta propiedad sólo se aplica si los bordes de las celdas están separados (*border-collapse: separate*).

- Se indica primero el espacio horizontal (espacio-x) y después el vertical (espacio-y).
- Si sólo se indica un valor, se aplica a los dos espacios.

### Ejemplo:

```
border-spacing: 5px; /* Establece el espaciado entre bordes a 5 */
border-spacing: 5px 15px; /* espacio-x=5px, espacio-y=15px */
```

**empty-cells.** Permite definir cómo se visualizan las celdas vacías. Esta propiedad sólo se aplica si los bordes de las celdas están separados (*border-collapse: separate*).

- `show`: Se visualizan el background y los bordes.
- `hide`: No se visualiza nada.

## Alinear tablas

La alineación de las tablas se realiza con los márgenes, como en los elementos de bloque. Por defecto, están alineadas a la izquierda.

Ejemplo: Centrar en su contenedor.

```
table {  
    margin: 0 auto;  
}
```

## 3. Modelo de formato visual

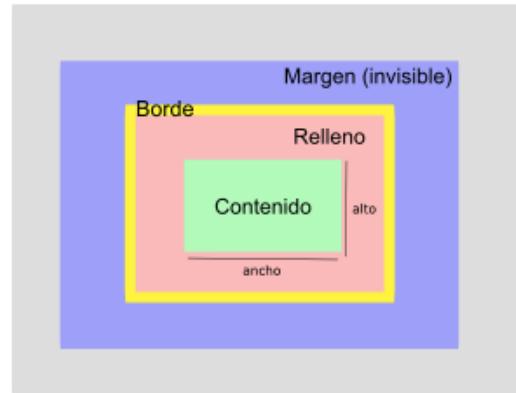
El objetivo de este apartado es describir cómo las aplicaciones de usuario procesan la estructura del documento para los medios visuales. Los medios visuales son los que proporcionan una salida visual, por ejemplo, la pantalla y la impresora.

### 3.1. El modelo de cajas (box model)

El navegador crea una "caja" para cada uno de los elementos html que componen una página web (`<h1>`, `<p>`, `<strong>`, `<div>`...). Mediante CSS se pueden configurar las características de esas "cajas". Este comportamiento de las páginas Web se conoce como "modelo de cajas".

Estas cajas se organizan de la siguiente forma:

- **Contenido:** Es lo que se quiere mostrar, texto, imagen, etc.
- **Relleno (padding):** Espacio que se puede crear alrededor del contenido.
- **Borde (border):** Línea alrededor del contenido y el relleno (si existe).
- **Margen (margin):** Espacio que se puede crear alrededor del borde. Separa esta caja de las cajas adyacentes o de la caja del elemento padre. Los márgenes son siempre transparentes.



### Cómo se aplica el `background` a las cajas

- Las imágenes de fondo (`background-image`) y el color de fondo (`background-color`) están debajo del contenido y del relleno o padding.
- Si se definen un contenido, una imagen de fondo y un color de fondo, el orden de presentación es el siguiente: se muestra el contenido sobre la imagen de fondo y esta sobre el color del fondo. Es decir, el color de fondo se verá si el contenido y la imagen de fondo lo permiten: porque no ocupen completamente el espacio del contenido o porque tengan zonas transparentes.

#### 3.1.1. Márgenes (margin)

Establecen la distancia entre el borde y las cajas adyacentes o de la caja del elemento padre.

Los márgenes verticales (margin-top y margin-bottom) sólo se pueden aplicar a los elementos de bloque y las imágenes, mientras que los márgenes laterales (margin-left y margin-right) se pueden aplicar a cualquier elemento.

Propiedad	Descripción	Valores
margin-top		
margin-right	Tamaño del margen superior, derecho, inferior e izquierdo	[ <longitud>   <porcentaje>   auto ]
margin-bottom		
margin-left		
margin	Ancho para varios márgenes individuales	[ <longitud>   <porcentaje>   auto ]{1,4}

### Valores que se pueden utilizar para especificar los márgenes

- longitud: normalmente en píxeles o em
- porcentaje
- auto: Calcula automáticamente la distancia mínima según la relación con otros elementos.

### Puede tomar 1, 2, 3 o 4 valores

- Si toma 1 valor, se aplica el mismo margen a los cuatro lados.
- Si toma 2 valores, se aplica el primer valor a los márgenes superior e inferior, y el segundo a los márgenes izquierdo y derecho.
- Si toma 3 valores, el primero se aplica al margen superior, el segundo a los márgenes izquierdo y derecho, y el tercero al margen inferior.
- Si toma 4 valores, se aplican a los márgenes superior, derecho, inferior e izquierdo (empezando arriba, y en el sentido de las agujas del reloj).

Ejemplo: Establecer los márgenes superior e inferior como 15px y los márgenes izquierdo y derecho como 30px.

```
margin-top: 15px;  
margin-right: 30px;  
margin-bottom: 15px;  
margin-left: 30px;
```

El ejemplo anterior se podía haber escrito así:

```
margin: 15px 30px;
```

### Márgenes negativos

Es posible especificar valores negativos para los márgenes. Cuando se aplica un valor negativo, el contenido, el relleno y el borde de la caja se desplazan en dirección contraria de donde se colocarían con un valor positivo.

### Fusión de márgenes verticales

Si se juntan los márgenes verticales de cajas adyacentes o anidadas, los márgenes se fusionan de forma automática para formar un solo margen. No puede haber entre ellos rellenos o bordes. Sólo se fusionan en el caso de cajas de bloques en el flujo normal. El nuevo margen será igual a la altura del margen más alto de los que se han fusionado.

### 3.1.2. Relleno (padding)

- Establece la distancia entre el contenido del elemento y el borde.
- Como margin, puede tomar 1, 2, 3 ó 4 valores.

Propiedad	Descripción	Valores
padding-top padding-right padding-bottom padding-left padding	Ancho del relleno superior, derecho, inferior e izquierdo [ <longitud>   <porcentaje> ] Tamaños para varios rellenos individuales [ <longitud>   <porcentaje> ] {1,4}	

### 3.1.3. Bordes (border)

Se pueden definir tres propiedades: el ancho (border-width), el color (border-color) y el estilo (border-style).

Propiedad	Descripción	Valores
border-top-width border-right-width border-bottom-width border-left-width border-width	Anchura del borde superior, derecho, inferior o izquierdo [ thin   medium   thick   <longitud> ] Anchos de varios bordes individuales [ thin   medium   thick   <longitud> ] {1,4}	
border-top-color border-right-color border-bottom-color border-left-color border-color	Color del borde superior, derecho, inferior o izquierdo [ <color>   transparent ] Colores de varios bordes individuales [ <color>   transparent ] {1,4}	
border-top-style border-right-style border-bottom-style border-left-style border-style	Estilo del borde superior, derecho, inferior o izquierdo [ none   hidden   dotted   dashed   solid   double   groove   ridge   inset   outset ] Estilos de varios bordes individuales [ none   hidden   dotted   dashed   solid   double   groove   ridge   inset   outset ] {1,4}	
border-top border-right border-bottom border-left border	Ancho, estilo y el color para el borde superior, derecho, inferior o izquierdo [ <border-top-width>    <border-top-style>    <border-top-color> ] Ancho, el estilo y el color para los 4 bordes [ <border-top-width>    <border-top-style>    <border-top-color> ]	

#### border-color

- El valor inicial es el que tiene la propiedad color.

#### border-style

- Los bordes más utilizados en los diseños habituales son solid y dashed, seguidos de double y dotted.
- Los estilos none y hidden son idénticos visualmente, pero se diferencian en la forma que los navegadores resuelven los conflictos entre los bordes de las celdas adyacentes en las tablas.
- El valor por defecto de border-style es none.

Como margin y padding, las propiedades border-width, border-color y border-style pueden tomar 1, 2, 3 ó 4 valores.

### 3.1.4. Bordes redondeados (border-radius) CSS3

Se utiliza para hacer bordes redondeados en lugar de cuadrados.

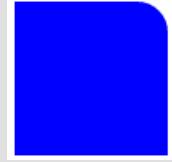
Propiedad	Descripción	Valores
border-top-left-radius border-top-right-radius border-bottom-left-radius border-bottom-right-radius	Radio de los bordes superior-izquierdo, superior-derecho, inferior-izquierdo e inferior-derecho.	[<longitud>   <porcentaje>]{1,2}
border-radius	Radio de los cuatro bordes redondeados.	[<longitud>   <porcentaje>]{1,4}

**border-top-left-radius, border-top-right-radius, border-bottom-left-radius, border-bottom-right-radius**

Se pueden especificar uno o dos valores. El primero es el radio horizontal. El segundo es el radio vertical. Si no se especifica, toma como valor el radio vertical.

Ejemplo:

```
div {
    width: 100px;
    height: 100px;
    background-color: blue;
    border-top-right-radius: 20px;
}
```



Cada esquina se definen dos valores: el radio horizontal seguido del radio vertical.

```
div {
    width: 100px;
    height: 100px;
    background-color: blue;
    border-top-right-radius: 20px 40px;
}
```



#### border-radius

- Las propiedad border-radius puede tomar 1, 2, 3 o 4 valores.
- Si se quieren especificar los radios horizontal y vertical, se escriben del siguiente modo: radio-horizontal/radio-vertical.
- Si se omite el radio vertical, toma el mismo valor que el radio horizontal.

Ejemplo: los bordes superior-izquierdo e inferior derecho tienen un radio de 20px, mientras que los bordes superior-derecho e inferior-izquierdo tienen un radio de 40px.

```
div {
    width: 100px;
    height: 100px;
    background-color: blue;
    border-radius: 20px 40px;
}
```



### 3.1.5. Contenido

Propiedad	Descripción	Valores
<code>width</code>	Ancho	[ <code>&lt;longitud&gt;</code>   <code>&lt;porcentaje&gt;</code>   <code>auto</code> ]
<code>min-width</code>	Ancho mínimo	[ <code>&lt;longitud&gt;</code>   <code>&lt;porcentaje&gt;</code> ]
<code>max-width</code>	Ancho máximo	[ <code>&lt;longitud&gt;</code>   <code>&lt;porcentaje&gt;</code>   <code>none</code> ]
<code>height</code>	Alto	[ <code>&lt;longitud&gt;</code>   <code>&lt;porcentaje&gt;</code>   <code>auto</code> ]
<code>min-height</code>	Alto mínimo	[ <code>&lt;longitud&gt;</code>   <code>&lt;porcentaje&gt;</code> ]
<code>max-height</code>	Alto máximo	[ <code>&lt;longitud&gt;</code>   <code>&lt;porcentaje&gt;</code>   <code>none</code> ]

Las propiedades `width` y `height` permiten establecer el ancho y el alto del contenido.

Las propiedades `min-width`, `min-height`, `max-width` y `max-height` permiten establecer unos valores mínimos y máximos para el ancho y el alto respectivamente.

#### Valores

- Longitud: No admite valores negativos.
- Porcentajes: Se calculan a partir de la anchura o altura de su elemento padre.
- `auto`: el navegador debe calcular automáticamente la anchura o altura del elemento, teniendo en cuenta sus contenidos y el sitio disponible en la página.

### 3.2. Bloques de contención

En general, una caja "establece" el bloque de contención para sus descendientes. El "bloque de contención de una caja" es la caja dentro de la cual se ha creado esa caja. Cada caja tiene una posición dada con respecto a su bloque de contención, pero lo puede desbordar.

#### 3.2.1. El bloque de contención inicial

Es el que se genera para el elemento raíz (`html`).

Ancho del bloque de contención inicial:

- Se puede especificar con la propiedad `width`.
- El valor "`auto`" significa que la aplicación de usuario calcula el ancho inicial

Alto del bloque de contención inicial:

- Se puede especificar con la propiedad `height`.
- El valor "`auto`" significa que la aplicación de usuario calcula el alto inicial

Este bloque no puede ser posicionado o flotar. Las aplicaciones de usuario ignoran las propiedades "`position`" y "`float`" para el elemento raíz.

#### 3.2.2. Tipos de elementos: elementos a nivel de bloque y elementos en línea

##### Elementos a nivel de bloque

- Los *elementos de bloque* empiezan en una línea nueva y ocupan todo el ancho del elemento contenedor. Cuando se redimensiona la ventana o un elemento contenedor, los elementos de bloque se ajustan al nuevo ancho.
- Sólo pueden aparecer dentro de otros elementos de bloque: no pueden insertarse dentro de elementos en línea.
- Pueden contener cajas de bloque o cajas en línea.

- Los siguientes valores de la propiedad *display* conforman un elemento a nivel de bloque: *block*, *list-item*, *run-in* (a veces) y *table*.
- Por ejemplo, son elementos de bloque: *h1*, *h2*, *h3*, *h4*, *h5*, *h6*, *p*, *div*, *li*, *ul*, *ol*, *table*.

### Elementos a nivel de línea

- Los *elementos en línea* se visualizan en la posición en la que aparecen y ocupan sólo el espacio que necesitan. Cuando se redimensiona el navegador, el contenido se reajusta.
- Pueden aparecer tanto dentro de un elemento de bloque como dentro de un elemento en línea.
- Los siguientes valores de la propiedad *display* conforman un elemento en línea: *inline*, *inline-table* y *run-in* (a veces).
- Por ejemplo, son elementos en línea: *a*, *br*, *em*, *strong*, *sub*, *sup*.

### 3.3. La propiedad "display"

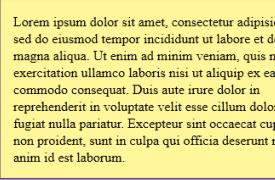
Todos los elementos tienen una propiedad *display* que determina el tipo de caja que se generará. El tipo de caja predeterminado es *inline*, pero la hoja de estilo del navegador establece otros estilos predeterminados. Por ejemplo, el estilo predeterminado para un elemento *div* es *block*.

MODELO DE FORMATO VISUAL		
Propiedad	Descripción	Valores
<i>display</i>	Comportamiento del contenedor	[ <i>inline</i>   <i>block</i>   <i>list-item</i>   <i>run-in</i>   <i>inline-block</i>   <i>table</i>   <i>inline-table</i>   <i>table-row-group</i>   <i>table-header-group</i>   <i>table-footer-group</i>   <i>table-row</i>   <i>table-column-group</i>   <i>table-column</i>   <i>table-cell</i>   <i>table-caption</i>   <i>none</i> ]

Algunos de los valores más utilizados son:

- block**: El elemento genera una caja de bloque
- inline-block**: El elemento genera una caja de bloque, que fluye como una caja en línea, similar a un elemento reemplazado (*img* o *iframe*).
- inline**: El elemento genera una caja en línea.
- none**: El elemento no genera cajas en la estructura de formato. Los descendientes tampoco generan ninguna caja. El resto de elementos se visualizan como si no existiera este elemento.

Ejemplo: Aplicar la propiedad *display* a la imagen.

 <i>display: inline (default)</i>	 <i>display: none</i>	 <i>display: block</i>
---	---	--

### 3.4. Esquema de posicionamiento

En CSS se definen tres esquemas de posicionamiento:<sup>4</sup>

- **Flujo normal:** El objeto se coloca en función del lugar que ocupa en el documento (esto significa que el lugar que ocupa en el árbol de renderización es similar al lugar que ocupa en el árbol de DOM) y se diseña de acuerdo con sus dimensiones y con el tipo de caja.
- **Flotante:** El objeto se diseña primero según el flujo normal y, posteriormente, se mueve hacia la derecha o hacia la izquierda todo lo posible.
- **Posicionamiento absoluto:** El objeto se coloca en el árbol de renderización de una forma diferente a la que se utiliza para colocarlo en el árbol de DOM. El diseño se define con exactitud independientemente del flujo normal. El objeto no participa en el flujo normal.

Las propiedades **position** y **float** determinan el esquema de posicionamiento.

- Si se utiliza **position**, con valores "**static**" o "**relative**", se genera un flujo normal.
- Si se utiliza **float**, se genera un esquema de posicionamiento flotante.
- Si se utiliza **position**, con valores "**absolute**" y "**fixed**", se produce un posicionamiento absoluto.

**Nota:** Si **display** vale **none**, se ignoran las propiedades **float** y **position**.

#### 3.4.1. Esquema de posicionamiento flotante

##### Propiedades float y clear

La distribución de elementos en una página con elementos flotantes se controla con las propiedades **float** y **clear**.

<b>float</b>	Posicionamiento flotante	[ left   right   none ]
<b>clear</b>	Control de cajas adyacentes a los <b>float</b>	[ none   left   right   both ]

**float.** Define el comportamiento del objeto que flota.

El objeto se coloca en su posición normal y después se sitúa todo a la izquierda o a la derecha que se puede en su línea (dentro de su elemento padre).

El resto de cajas fluye a su alrededor.

**clear.** Define qué lados de un elemento no pueden estar al lado de elementos flotantes.

- **left:** Impide que el lado izquierdo del elemento se sitúe junto a un elemento flotante.
- **right:** Impide que el lado derecho del elemento se sitúe junto a un elemento flotante.
- **both:** Impide que el elemento se sitúe junto a elementos flotantes.
- **none:** No hay restricciones sobre la posición de la caja respecto a elementos flotantes.

<sup>4</sup> <http://www.html5rocks.com/es/tutorials/internals/howbrowserswork/#css>

### Notas:

- Se debe especificar el ancho de los elementos flotantes (sólo se podría omitir en las imágenes porque el navegador lo calcula).
- Los elementos flotantes se colocan cuando aparecen en el flujo del documento. Por ello, siempre estarán dentro de la línea en la que aparecen o más abajo, *nunca más arriba*.
- Los elementos flotantes tienen que aparecer antes que los elementos que fluyen alrededor.
- Los elementos flotantes permanecen siempre dentro del área de contenido de su elemento padre (no invaden el padding).

### Ejemplo: Imagen flotando a la derecha

```
img {  
    float: right;  
}  
  
p {  
    padding: 15px;  
    background-color: #FFF799;  
    border: 2px solid #6C4788;  
}  
  
<p>Lorem ipsum dolo... </p>
```

Lorem ipsum dolor sit amet, consectetur adipisicing 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.



### Elementos en línea como elementos flotantes

Si se define la propiedad float para un elemento, cambia la propiedad display: display=block. Al comportarse como elementos de bloque, se pueden especificar los márgenes superior e inferior que normalmente no se tienen en cuenta para los elementos en línea.

### Ejemplo: span flotando a la derecha.

```
span.aclaracion {  
    float: right;  
    width: 150px;  
    padding: 10px;  
    margin: 5px;  
    background-color: gray;  
    color: white;  
}
```

```
<p><span class="aclaracion">Texto latino</span>Lorem ipsum  
dolo... </p>
```

Text latino

Texto latino

## Relleno, bordes y márgenes del elemento flotante

Un elemento flotante conserva siempre sus márgenes. Es decir, flota toda la caja (contenido, rellenos, bordes y márgenes), y los márgenes no se colapsan con los superiores o inferiores.

Ejemplo: Se especifica relleno, borde y margen para la imagen.

```
img {  
    float: right;  
    padding: 10px;  
    border: 1px solid red;  
    margin: 25px;  
}
```

Etiam tempore 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.



El margen de la imagen se añade al margen del elemento contenedor. Si el margen superior fuera 0, el borde de la imagen (o la imagen, si no tiene borde) quedaría alineado con el resto del contenido del párrafo (`margin: 0 25px;`).

**Excerpt from the Declaration of Independence:**

“We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable Rights, that among these are Life, Liberty and the Pursuit of Happiness.”

**Conclusion:**

The Declaration of Independence is a powerful document that has inspired generations of Americans to stand up for their rights and freedoms. It serves as a reminder that we must always be vigilant in protecting our democracy and upholding the principles of equality, freedom, and justice for all.



## Varios elementos flotando dentro de un contenedor

Supongamos que varios elementos dentro de un contenedor flotan a la izquierda. ¿Dónde se colocarán en este caso el resto de los elementos flotantes?

Cada uno se colocará todo a la izquierda que pueda, en la línea en la que está o más abajo, teniendo como límite el borde del elemento padre y otros elementos flotantes que ya estén colocados en ese lado, si existen.

¿Qué ocurre si un elemento flotante no cabe en la línea (porque se haya cambiado la ventana del navegador, por ejemplo)?

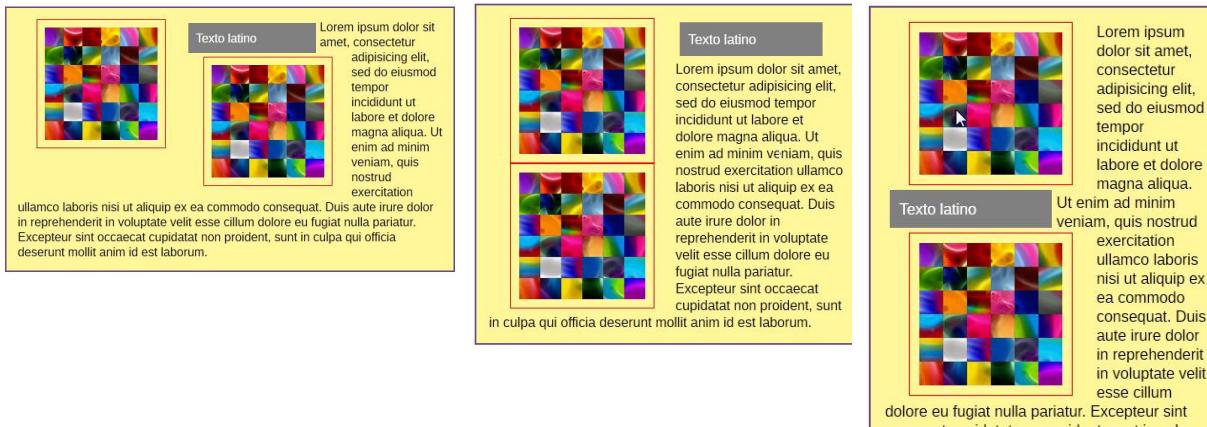
En este caso, se desplazará hacia abajo, hasta que encuentre una línea en la que quepa.

Ejemplo: Los estilos del ejemplo anterior se aplican al siguiente código HTML (varios flotan a la izquierda):

```
<p> 
    <span class="aclaracion">Texto latino</span>
    Lorem ipsum...</p>
```



Si se redimensiona la ventana del navegador, lo podríamos ver así:



## Elementos contenedores de elementos flotantes

El elemento padre no tiene en cuenta a los elementos flotantes para calcular su altura.

Ejemplo:

El siguiente código HTML crea dos divs flotantes y un div general que los contiene:



Ejemplo: En la primera captura tenemos los dos div sin flotar. En la segunda captura hemos hecho flotar los dos div: observamos que del div general sólo se ven los bordes.

El siguiente código HTML crea dos divs flotantes y un div general que los contiene:

El resultado es que el div general sólo muestra sus bordes, ya que no tiene en cuenta a los elementos flotantes para calcular su altura.

Hay varias técnicas con las que se puede conseguir este efecto.

- Una es hacer que el elemento contenedor sea flotante y darle un ancho del 100%.
- Otra solución frecuente es beneficiarnos del comportamiento de la propiedad `overflow`. Dando el valor `auto` o `hidden` a esta propiedad del contenedor, este tendrá en cuenta a los elementos flotantes para calcular su altura.

Ejemplo: En el ejemplo anterior podemos añadir la siguiente propiedad al elemento <p>.

```
p {  
    float: left;  
    ...
```

Lorem ipsum dolor sit amet, consectetur adipisicing 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.



Se podría conseguir un efecto parecido con la siguiente propiedad (en lugar de las anteriores):

```
p {  
    overflow: auto;  
    ...
```

### 3.4.2. Esquemas de posicionamiento de flujo normal y posicionamiento absoluto

Como se dijo en el punto 3.4, además del esquema flotante, hay dos esquemas de posicionamiento:

- **flujo normal:** Si `position` es `static` o `relative`.
- **posicionamiento absoluto:** Si `position` es `absolute` o `fixed`.

El tipo de esquema de define con la propiedad `position`, y la posición del elemento se define con las propiedades `top`, `bottom`, `left` y `right`.

position	Esquema de posicionamiento	[ static   relative   absolute   fixed ]
top		
right		
bottom	Desplazamiento de la caja (respecto al límite superior, derecho, inferior o izquierdo del contenedor)	[ <longitud>   <porcentaje>   auto ]
left		

La propiedad `position` puede tener cuatro valores:

- **static:** Posicionamiento normal o estático.
- **relative:** Posicionamiento relativo.
- **absolute:** Posicionamiento absoluto.
- **fixed:** Posicionamiento fijo.

**Nota:** Si *position = absolute* o *fixed*, se asigna: *float=none* y *display=block*.

Las propiedades *top*, *bottom*, *left* y *right*, indican el desplazamiento del elemento respecto a un elemento u objeto de referencia. Especifican el desplazamiento desde un borde de dicho elemento y hacia el centro del elemento.

- **top:** Desplazamiento respecto al borde superior: es un desplazamiento hacia abajo.
- **right:** Desplazamiento respecto al borde derecho: es un desplazamiento hacia la izquierda.
- **bottom:** Desplazamiento respecto al borde inferior: es un desplazamiento hacia arriba.
- **left:** Desplazamiento respecto al borde izquierdo: es un desplazamiento hacia la derecha.

### Valores como porcentajes

Si el desplazamiento se indica en porcentaje, se refiere al porcentaje respecto a la anchura (propiedades *right* y *left*) o respecto a la altura (propiedades *top* y *bottom*) del elemento padre.

### Valores negativos

Se pueden utilizar valores negativos, que mueven los elementos en dirección contraria a los valores positivos.

## Flujo normal

### Posicionamiento estático

En el tipo de posicionamiento por defecto En este posicionamiento, las propiedades *top*, *right*, *bottom* y *left* son ignoradas.

### Posicionamiento relativo

Es una variante del posicionamiento estático. El navegador coloca todos los elementos de la página según el posicionamiento estático. Después, desplaza la caja **respecto a su posición original**, teniendo en cuenta los valores de las propiedades *top*, *bottom*, *left* y *right*. El resto de las cajas permanecen como si este elemento no se hubiera movido, y quedará vacío el espacio que ocuparía el elemento. Se pueden producir solapamientos entre cajas.

## Posicionamiento absoluto

### Posicionamiento absoluto

El navegador coloca el elemento en la posición que se determine con las propiedades *top*, *right*, *bottom* y *left*, que indican el desplazamiento **respecto a su primer elemento antecesor posicionado**.

Un elemento posicionado es un elemento para el que se ha asignado al atributo **position** un valor **distinto de static**, aunque no se haya desplazado, es decir, que puede aparecer en su posición original (ej. *position=relative*, pero no se modifican las propiedades de desplazamiento).

Si no hubiera ninguno, se posiciona respecto al elemento body. Ese elemento queda fuera del flujo de elementos de la página: el resto de elementos se colocan ignorando la posición de ese elemento: es como si no existiera. Se producen solapamientos.

### **Posicionamiento fijo**

Es una variante del posicionamiento absoluto. Se colocan los elementos de la página según el posicionamiento absoluto, pero siempre **respecto a la ventana del navegador**. Después, su lugar en la pantalla es fijo, y no varía, aunque se haga scroll.

### **Superposición de elementos, z-index**

Los métodos de posicionamiento pueden provocar la superposición de cajas. Por defecto, los elementos se apilan en el orden en que aparecen. Se puede cambiar este comportamiento con la propiedad *z-index*.

La propiedad *z-index* permite definir el nivel de profundidad de una caja. Su valor es un número entero: aunque se permiten números negativos, 0 se suele tomar como el valor más bajo. Cuanto más alto sea el valor, "más cerca" del usuario se muestra la caja. La propiedad *z-index* sólo tiene efecto sobre elementos posicionados (position: relative, absolute o fixed).

## **3.5. Efectos visuales**

EFFECTOS VISUALES		
Propiedad	Descripción	Valores
<code>overflow</code>	Comportamiento del contenido si se desborda en la caja	[ <code>visible</code>   <code>hidden</code>   <code>scroll</code>   <code>auto</code> ]
<code>clip</code>	Especifica la región visible del elemento	[ <code>rect (&lt;longitud&gt;, &lt;longitud&gt;, &lt;longitud&gt;, &lt;longitud&gt;)</code>   <code>auto</code> ]
<code>visibility</code>	Visibilidad de las cajas	[ <code>visible</code>   <code>hidden</code>   <code>collapse</code> ]

### **3.5.1. Desbordamiento: overflow**

Generalmente, el contenido de una caja de bloque está dentro de los límites del contenido de la caja. En ciertos casos, el contenido se puede "desbordar". Esta propiedad especifica si el contenido de un elemento a nivel de bloque se recorta cuando desborda la caja del elemento.

Puede tomar los siguientes valores:

- **visible**: El contenido no se recorta, es decir, puede ser procesado fuera de la caja de bloque. Es el valor por defecto.
- **hidden**: El contenido se recorta y no se proporcionan barras de desplazamiento para acceder a la zona recortada.
- **scroll**: El contenido se recorta y, si la aplicación de usuario dispone de un mecanismo de desplazamiento (como las barras de desplazamiento), este mecanismo se muestra aunque el contenido quepa dentro de su contenedor.
- **auto**: Se permite que el navegador decida cómo manejar el overflow. En la mayoría de los casos, las barras de scroll se añaden sólo si son necesarias.

### Ejemplo:

<p>lorem ipsum</p>	<p>lorem ipsum</p>	<p>lorem ipsum</p>	<p>lorem ipsum</p>
<p>lorem ipsum</p>	<p>lorem ipsum</p>	<p>lorem ipsum</p>	<p>lorem ipsum</p>

### 3.5.2. Visibilidad: visibility

La propiedad *visibility* se utiliza para ocultar elementos. A diferencia de *display:none*, el elemento es invisible, pero el espacio que ocupa se mantiene dejando un hueco.

Puede tomar los siguientes valores:

- **visible:** El elemento es visible. Es el valor por defecto
  - **hidden:** El elemento no es visible. El resto de los elementos se visualizan como si el elemento fuera visible.
  - **collapse:**
    - Si se aplica sobre filas, grupos filas, columnas o grupos de columnas de una tabla, esta propiedad hace que el elemento se colapse y el resto de los elementos se visualicen como si este elemento no estuviera en la pantalla.
    - Si se aplica sobre otros elementos, su efecto es idéntico a la propiedad **hidden**.

Ejemplo: Aplicar la propiedad visibility a la imagen.

 <p data-bbox="403 1563 743 1610">Lorem ipsum dolor sit amet, consectetur adipisciing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ac minim veniam, quis nostrud exercitation ullamco labor</p>	<p data-bbox="903 1563 1181 1610">Lorem ipsum dolor sit amet, consectetur adipisciing 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</p>
<p data-bbox="339 1628 743 1650"><b><i>img { visibility: visible (default);}</i></b></p>	<p data-bbox="790 1628 1181 1650"><b><i>img { visibility: hidden (o collapse);}</i></b></p>

## 4. Estructuras de distribución (*layouts*)

Existe un número prácticamente ilimitado de formas de estructurar una página web. En este apartado, se verán sólo algunos de ellos.

### 4.1. Diseños de ancho fijo o fluidos

Los diseñadores utilizan varios enfoques para la distribución del contenido de una página Web; los principales son:

- **Diseños de ancho fijo.**
- **Diseños fluidos o líquidos.**
- **Diseños híbridos:** Contienen áreas de ancho fijo y áreas de ancho variable.

Los diseños se diferencian por las unidades de medida que utilizan para especificar las dimensiones horizontales de los elementos (ancho).

#### 4.1.1. Diseños de ancho fijo

La página tiene siempre el mismo tamaño, independientemente de la ventana del navegador.

#### Características

- En este diseño, el ancho de los elementos se determina en píxeles.
- **Ancho de la página.** La mayoría de los sitios se diseñan para un ancho de 960 píxeles de ancho aproximadamente, para que se ajusten bien en pantallas con una resolución de 1024x768.

#### Uso

- Apropiado para páginas que se van a acceder desde un equipo de escritorio (no dispositivos móviles, *tablets*, etc.).
- **Inconveniente:** Si la ventana del navegador es más pequeña que la página, aparece la barra de *scroll* horizontal.

#### Ejemplo



#### 4.1.2. Diseños fluidos o líquidos

Ajusta la página proporcionalmente cuando la ventana del navegador cambia de tamaño.

##### Características

- En este diseño, el ancho de los elementos se determina en porcentajes.
- El diseño fluido es uno de los pilares de la técnica de **diseño web adaptativo** (*responsive web design*).
- Este tipo de diseños está tomando cada vez más importancia. ¿Los motivos? La gran variedad de pantallas de distintos tamaños desde las que actualmente se accede a Internet: obviamente, no es posible crear un diseño fijo para cada tamaño de pantalla.

Un ejemplo: [www.w3.org](http://www.w3.org).

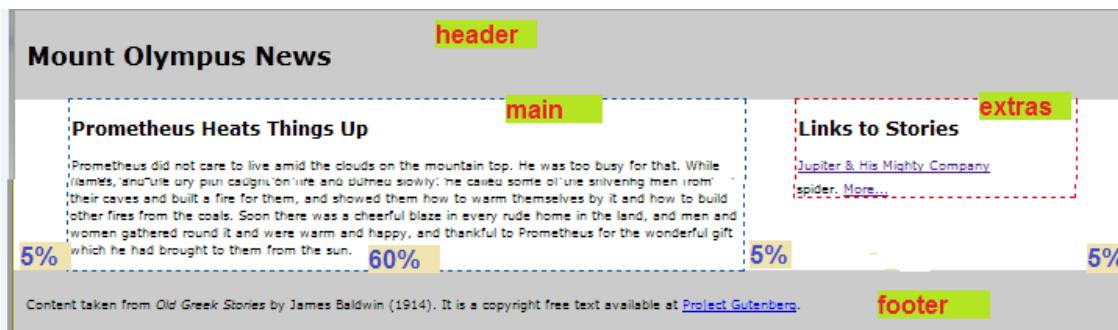
##### Ventajas

- Se adaptan a distintos medios.
- El texto ocupa toda la pantalla.
- No hay barras de scroll horizontales.

##### Inconvenientes

- Si la pantalla es muy grande, la línea de texto puede ser demasiado larga.
- Menos predecible.
- Más cálculos para determinar las medidas.

##### Ejemplo



## 5. Técnicas para crear columnas

### 5.1. Crear columnas utilizando contenedores flotantes

Este método consiste en utilizar la propiedad `float` para crear columnas. Es el método más utilizado. La ventaja de utilizar la propiedad `float` es que es más fácil que no existan solapamientos, aunque el ancho de las columnas sea fijo. La desventaja es que el diseño puede ser más complicado, ya que el orden en el que los elementos aparecen en el código influye en la visualización de la página.

Se pueden crear dos columnas dentro de un contenedor de distintos modos:

- Hacer flotar un **div** a un lado y dejar un margen al otro.
- Hacer flotar los dos **div**, ambos a la izquierda o a la derecha.
- Hacer flotar un **div** a la izquierda y el otro a la derecha (o al revés).

Para crear más columnas, se haría de forma parecida.

Ejemplo: Se crean dos **div** (de clase "columna") dentro de un **div**, #contenedor:

```
#contenedor {  
    overflow: auto;  
    width: 98%;  
    background-color: #EEE;  
    border: 2px solid #CCC;  
    margin: 0 auto;  
}  
.columna {  
    float: left;  
    width: 46%;  
    margin: 0 2% 1.2em;  
}  
  
<div id="contenedor">  
<div class="columna"><p><strong>Lorem ipsum dolor sit amet,</strong> ... </p>  
</div>  
<div class="columna"><p><strong>Duis aute irure dolor</strong> in ... </p>  
</div>  
</div>
```

Se verá:

**Lorem ipsum dolor sit amet,** consectetur adipisicing 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.

- Es importante calcular bien los anchos de cada columna, teniendo en cuenta los rellenos, bordes y márgenes. Dependerán de la propiedad **box-sizing**: content-box (width/height incluye solo contenido) o border-box (incluye contenido, rellenos y bordes).

- Es frecuente mezclar porcentajes y em para crear diseños fluidos que se adapten al tamaño de la ventana del dispositivo. Algunos diseñadores utilizan **porcentajes** en distancias horizontales y **em** en las verticales, ya que hace referencia a las líneas de texto.

## 5.2. Crear columnas utilizando posicionamiento absoluto

El segundo modo de crear columnas es utilizar posicionamiento absoluto. La desventaja de utilizar este método es que no podemos posicionar un pie de página, a no ser que las columnas tengan un alto fijo. Además, con esta técnica, pueden existir solapamientos, si el ancho de las columnas es fijo.

## 5.3. ¿Qué técnica aplicar en los diseños fijo y fluido?

Ambas técnicas se pueden aplicar a diseño fijo o fluido. La diferencia está en la unidad de medida que utilicemos para los anchos de las columnas, así como los rellenos y márgenes:

- *Diseño fijo*: Las unidades de medida serán **píxeles**.
- *Diseño fluido*: Las unidades de medida serán **porcentajes**.

# 6. Técnicas para crear barras de navegación

Una barra de navegación normalmente consiste en una serie de enlaces agrupados en un área vertical u horizontal. La mayoría de los diseñadores Web actuales utilizan listas desordenadas para crear barras de navegación para sus sitios.

## ¿Por qué se utilizan listas desordenadas?

- Los enlaces de la barra de navegación son una colección de *elementos similares*, como los elementos de las listas.
- Si el navegador no es capaz de mostrar bien los estilos, la barra de navegación será una lista de enlaces operativa.
- Los elementos del submenú de una barra de navegación, que se pueden mostrar cuando se hace clic o se pasa sobre el elemento de menú del nivel principal, encuentran su paralelo exacto en los elementos de las listas anidadas.

## 6.1. Crear barras de navegación verticales

1. Crear la lista, normalmente dentro de un elemento nav.

```
<nav>
  <ul>
    <li><a href="inicio.html">Inicio</a></li>
    <li><a href="productos.html">Productos</a></li>
    <li><a href="servicios.html">Servicios</a></li>
    <li><a href="contacto.html">Contacto</a></li>
  </ul>
</nav>
```

2. Quitar las viñetas, y establecer los márgenes y el relleno a cero.

```
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
}
```

Inicio
Productos
Servicios
Contacto

3. Definir las anclas como elementos de bloque. Así se pueden establecer sus dimensiones, *paddings*, márgenes y otros efectos visuales.

```
ul li a {  
    display: block;  
    width: 10em;  
    /* más estilos */  
}
```

Inicio  
Productos  
Servicios  
Contacto

## 6.2. Crear barras de navegación horizontales

1. y 2. Para crear una barra de navegación horizontal, primero creamos la lista y quitamos las viñetas, como en los pasos 1 y 2 de la creación de barras de navegación verticales.

3. A continuación, tenemos que hacer que los ítems de lista se coloquen uno al lado de otro. Esto se puede hacer de varios modos:

- Con float:left: Flotar cada ítem de la lista a la izquierda y definir las anclas como elementos de bloque.

```
ul li {  
    float: left;  
}  
  
ul li a {  
    display: block;  
    width: 10em;  
    /* más estilos */  
}
```

- Con display: inline-block: Se puede conseguir el mismo efecto utilizando display:inline-block en lugar de float:left.

```
ul li {  
    display: inline-block;  
}  
  
ul li a {  
    display: block;  
    width: 10em;  
    /* más estilos */  
}
```

- Con display: inline: También se puede utilizar display: inline. Este método hace más difícil controlar el espacio entre ítems de navegación porque el navegador calcula el espacio en blanco entre los ítems teniendo en cuenta el font-size del contenedor.

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

## 7. Referencias y bibliografía

### REFERENCIAS WEB

#### CSS

- Validador CSS: <http://jigsaw.w3.org/css-validator/>.
- Guía de referencia CSS2.1, traducción al castellano: <http://www.w3c.es/Divulgacion/GuiasReferencia/CSS21/>
- Recomendación de W3C sobre el módulo de color en CSS3: <http://www.w3.org/TR/css3-color/>
- Información sobre CSS 3: <http://www.css3.info/preview/>
- W3SCHOOLS. Referencia CSS3: <http://www.w3schools.com/cssref/default.asp>
- Libros Web. Introducción a CSS: <http://www.librosweb.es/css/index.html>
- Libros Web. Referencia CSS 2.1: <http://www.librosweb.es/referencia/css/index.html>
- Libros Web. CSS avanzado: [http://www.librosweb.es/css\\_avanzado/](http://www.librosweb.es/css_avanzado/)
- Herencia y cascada: <http://mosaic.uoc.edu/ac/le/es/m6/ud2/index.html>

#### Fuentes Web

- Fuentes: <https://fonts.google.com/>
- Fuentes Web: [http://www.mclibre.org/consultar/htmlcss/css/css\\_fuentes\\_web.html](http://www.mclibre.org/consultar/htmlcss/css/css_fuentes_web.html)

### BIBLIOGRAFÍA

- CÓRCOLES TENDERÓN, J.E.; MONTERO SIMARRO, F. Diseño de Interfaces. Madrid: Ra-Ma, 2012. p. 43-93.
- NIEDERST ROBBINS, J.; Learning Web Design. O'Reilly, 2012.