

# Usabilidad y Accesibilidad

## Práctica 5: CSS (1)

### 1. Objetivos

- Aprender el lenguaje de estilos CSS.
- Conocer algunas herramientas que ayudan a escribir CSS correcto y compatible con distintos navegadores.
- Aprender a maquetar una página web con CSS.
- Conocer los conceptos básicos del diseño adaptativo.
- Aprender a crear un estilo adaptativo.
- Aprender a utilizar una biblioteca de iconos.
- Conocer los principales criterios de conformidad de Web Content Accessibility Guidelines (WCAG) 2.x que se aplican al crear páginas web accesibles.

### 2. Recursos

¿Cómo se escribe una regla en CSS? ¿Qué propiedades y valores existen en CSS? ¿Cómo se selecciona un elemento de una página web para aplicarle una regla de CSS?

- **W3Schools**<sup>1</sup>: cursos de aprendizaje y guías de referencia de diversas tecnologías empleadas en la programación web.
- **W3C**<sup>2</sup>: organismo internacional que desarrolla las especificaciones de las principales tecnologías que se emplean en la programación web.
- **Cascading Style Sheets (CSS) Quick Reference**<sup>3</sup>: resumen en una página de lo más importante de CSS.
- **CSS2 Cheat Sheet**<sup>4</sup>: resumen en una sola cara de lo más importante de CSS2. También está traducida al español<sup>5</sup>.
- **CSS 2.1 and CSS 3 Help Cheat Sheets**<sup>6</sup>: resumen de CSS 2.1 y de las nuevas características de CSS 3 realizado para de la famosa revista Smashing Magazine. Es del año 2010, así que seguramente faltan algunas de las últimas novedades.

¿Cómo puedo saber que el código CSS que he escrito es correcto?

- **W3C CSS Validation Service**<sup>7</sup>: servicio de validación de CSS. Se puede indicar una página web para que valide los CSS que contiene o directamente un CSS.

¿Cómo se representan los colores en CSS? Para trabajar con los colores necesitas conocer la codificación RGB (red, green, blue):

---

<sup>1</sup><https://www.w3schools.com>

<sup>2</sup><https://www.w3.org>

<sup>3</sup><https://web.archive.org/web/20170606140443/http://www.digilife.be/quickreferences/QRC/Cascading%20Style%20Sheets%201.0.pdf>

<sup>4</sup><https://cheatography.com/davechild/cheat-sheets/css2/>

<sup>5</sup><https://web.archive.org/web/20111105110957/http://www.webtutoriales.com/tutoriales/html-css/hoja-referencia-css.26.html>

<sup>6</sup><https://www.smashingmagazine.com/2010/05/13/css-2-1-and-css-3-help-cheat-sheets-pdf/>

<sup>7</sup><https://jigsaw.w3.org/css-validator/>

- **Colores HTML**<sup>8</sup>: explica la codificación RGB empleada en HTML y CSS.
- **Anexo: Colores HTML**<sup>9</sup>: contiene una tabla con los nombres y los códigos hexadecimales de diferentes colores.

¿Qué combinaciones de colores son correctas? Los colores armónicos son aquellos que funcionan bien juntos, es decir, que producen un esquema de color atractivo a la vista. Algunas páginas donde puedes encontrar combinaciones de colores armónicos:

- **El uso de los colores en la Web**<sup>10</sup>: analiza el número de colores que debe emplearse en una página web, explica las diferentes relaciones que pueden existir entre los colores (análogos, complementarios, triada, etc.) y define el contraste entre dos colores.
- **ColorCombos**<sup>11</sup>: posee una librería de combinaciones de colores con diferente número de colores. Permite buscar una combinación que utilice un color deseado. También permite obtener, mediante la introducción de una URL, los colores empleados en una página web.

¿Existen plantillas de CSS que pueda emplear en mi sitio web? Existen multitud de sitios web donde se pueden encontrar plantillas gratuitas y de pago. Algunos sitios que ofrecen plantillas gratuitas son:

- **Free CSS**<sup>12</sup>: cientos de plantillas CSS gratuitas.
- **Free HTML CSS Template**<sup>13</sup>: cientos de plantillas CSS gratuitas.

¿Existe alguna herramienta que me pueda ayudar a escribir el código CSS?

- **Notepad++**<sup>14</sup>: editor gratuito de código fuente que soporta varios lenguajes de programación. Entre otras características, posee sintaxis coloreada, envoltura de sintaxis y autocompletado.
- **Sublime Text**<sup>15</sup>: excelente editor compatible con múltiples lenguajes de programación y disponible para Windows, OS X y Linux; se puede probar de forma gratuita, pero para un uso continuo hay que adquirir una licencia.
- **CSSMate**<sup>16</sup>: editor de CSS en página web.

¿Cómo se maqueta con CSS?

- **Porqué el diseñar con tablas es estúpido: problemas definidos, soluciones ofrecidas**<sup>17</sup>: explica porque no hay que diseñar con tablas y las ventajas de emplear CSS.
- **Simple 2 column CSS layout**<sup>18</sup>: explica paso por paso cómo se crea una página web para que tenga un diseño en 2 columnas.
- **CSS Layout cookbook**<sup>19</sup>: patrones de diseño para diferentes elementos de una página web.

¿Cómo se maqueta un formulario con CSS?

- **Accessible CSS Forms: Using CSS to Create a Two-Column Layout**<sup>20</sup>: explica paso a paso cómo maquetar un formulario con CSS.
- **How to Style Forms in CSS**<sup>21</sup>: explica paso a paso cómo maquetar un formulario con CSS.

<sup>8</sup>[https://es.wikipedia.org/wiki/Colores\\_HTML](https://es.wikipedia.org/wiki/Colores_HTML)

<sup>9</sup>[https://es.wikipedia.org/wiki/Anexo:Colores\\_HTML](https://es.wikipedia.org/wiki/Anexo:Colores_HTML)

<sup>10</sup><https://www.webusable.com/coloursMix.htm>

<sup>11</sup><https://www.colorcombos.com/>

<sup>12</sup><https://www.free-css.com/>

<sup>13</sup><https://templatemo.com/>

<sup>14</sup><https://notepad-plus-plus.org/>

<sup>15</sup><https://www.sublimetext.com/>

<sup>16</sup><http://cssmate.com/csseditor.html>

<sup>17</sup><https://www.hotdesign.com/seibold/spanish/>

<sup>18</sup>[https://www.456bereastreet.com/lab/developing\\_with\\_web\\_standards/csslayout/2-col/](https://www.456bereastreet.com/lab/developing_with_web_standards/csslayout/2-col/)

<sup>19</sup>[https://developer.mozilla.org/en-US/docs/Web/CSS/Layout\\_cookbook](https://developer.mozilla.org/en-US/docs/Web/CSS/Layout_cookbook)

<sup>20</sup><https://www.websiteoptimization.com/speed/tweak/forms/>

<sup>21</sup><https://www.peachpit.com/articles/article.aspx?p=456144>

- **Practical CSS Layout Tips, Tricks, & Techniques**<sup>22</sup>: algunos trucos de maquetación con CSS, explica un método de alinear los controles de un formulario.

¿Cómo se maqueta una tabla con CSS?

- **Styling tables**<sup>23</sup>: cómo definir el estilo de una tabla con CSS.
- **Styling Tables the Modern CSS Way**<sup>24</sup>: cómo definir el estilo de una tabla con técnicas modernas de CSS.
- **A Guide to Styling Tables**<sup>25</sup>: cómo definir el estilo de una tabla con CSS.

¿Existe alguna herramienta que me pueda ayudar a escribir el código CSS necesario para maquetar una página web?

- **CSS Layout Generator**<sup>26</sup>: generador del código CSS para maquetar.
- **CSS Layout Generator**<sup>27</sup>: generador del código CSS para maquetar.

¿Qué es el diseño adaptativo?

- **Diseño web responsivo, sensible, adaptable, ...**<sup>28</sup>: ¿cómo se dice *responsive web design* en español?
- **Responsive Web Design**<sup>29</sup>: artículo seminal en el que Ethan Marcotte acuñó el término *responsive web design*.
- **Multi-Device Web Design: An Evolution**<sup>30</sup>: análisis del diseño adaptativo actual y futuro.
- **Responsive Web Design: Missing the Point**<sup>31</sup>: discusión sobre la importancia actual y futura del diseño adaptativo.

¿Cómo se realiza un diseño adaptativo?

- **CSS Media Queries & Using Available Space**<sup>32</sup>: técnicas básicas para el diseño adaptativo.
- **Responsive Web Design Techniques, Tools and Design Strategies**<sup>33</sup>: colección de técnicas que se aplican en el diseño adaptativo.
- **Using media queries**<sup>34</sup>: sintaxis y operadores que se pueden emplear en las *media queries*.
- **CSS3 @media Rule**<sup>35</sup>: referencia de la sintaxis de las *media queries*.

### 3. ¿Qué tengo que hacer?

En esta práctica tienes que crear el estilo visual de las páginas web de la práctica anterior y del resto de páginas que harás en las próximas prácticas. Como mínimo, tienes que emplear las siguientes propiedades de CSS:

- **color**: para definir el color del texto.
- **font**: para definir las propiedades del texto.

<sup>22</sup><https://alistapart.com/article/practicalcss/>

<sup>23</sup>[https://developer.mozilla.org/en-US/docs/Learn/CSS/Building\\_blocks/Styling\\_tables](https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Styling_tables)

<sup>24</sup><https://piccalil.li/blog/styling-tables-the-modern-css-way/>

<sup>25</sup><https://dev.to/madsstoumann/a-guide-to-styling-tables-28d2>

<sup>26</sup><https://www.webdesignrankings.com/resources/csslayoutgenerator/>

<sup>27</sup><https://www.cssportal.com/layout-generator/>

<sup>28</sup><https://blogs.ua.es/pi/2012/04/17/diseño-web-responsivo-sensible-adaptable/>

<sup>29</sup><https://alistapart.com/article/responsive-web-design>

<sup>30</sup><https://www.lukew.com/ff/entry.asp?1436>

<sup>31</sup><https://bradfrost.com/blog/web/responsive-web-design-missing-the-point/>

<sup>32</sup><https://css-tricks.com/css-media-queries/>

<sup>33</sup><https://www.smashingmagazine.com/2011/07/responsive-web-design-techniques-tools-and-design-strategies/>

<sup>34</sup>[https://developer.mozilla.org/en-US/docs/Web/CSS/Media\\_Queries/Using\\_media\\_queries](https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries)

<sup>35</sup>[https://www.w3schools.com/cssref/css3\\_pr\\_mediaquery.php](https://www.w3schools.com/cssref/css3_pr_mediaquery.php)

- **background**: para definir las propiedades del fondo de un elemento.
- **border**: para definir las propiedades del borde de un elemento.
- **margin**: para definir la distancia que debe existir entre un elemento y los elementos que lo rodean.
- **padding**: para definir la distancia que debe existir entre el borde de un elemento y su contenido.

Además, define al menos un par de clases para definir el estilo particular de algunos elementos, como por ejemplo, `.anuncio` para aplicarlo a un párrafo que define un anuncio o `.conBorde` para aplicarlo a aquellos elementos que tienen que aparecer con un borde.

También debes añadir en cada página web al menos un enlace “saltar a” que dirija al contenido principal de cada página. Este enlace puede estar siempre visible o lo puedes ocultar con CSS para que se muestre únicamente cuando el enlace reciba el foco del teclado.

En esta práctica debes realizar la maquetación (*layout*) de la página. En el desarrollo de las páginas web, se entiende por maquetación a la acción de distribuir o posicionar los distintos elementos que queremos que aparezcan en una página web. En especial, haz una correcta maquetación de los controles de los formularios para que aparezcan alineados.

En la maquetación de tu página web, debes contemplar un diseño en varias columnas en al menos una parte de la página cuando la página se visualice en un dispositivo con una pantalla de alta resolución.

Hoy en día no se accede a la Web exclusivamente desde un ordenador con una pantalla grande con una alta resolución. Es muy normal acceder a la Web desde un dispositivo móvil como un teléfono inteligente (*smartphone*). Por tanto, tienes que **crear una hoja de estilo que incluya un diseño adaptativo** (*responsive design*), que tu sitio web se visualice de forma adecuada en un dispositivo con una pantalla de baja resolución (480px de ancho) o de alta resolución (1920px de ancho o más). Puedes definir todos los puntos de ruptura (*breakpoints*) que consideres necesarios.

Por último, utiliza alguna biblioteca de iconos, como Font Awesome<sup>36</sup>, Fontello<sup>37</sup> o Material Symbols & Icons<sup>38</sup> para añadir iconos a tu sitio web y así mejorar su usabilidad, su accesibilidad (si se hace bien) y su apariencia visual. Los puedes añadir a los formularios, a las barras de navegación o a otros elementos de las páginas de tu sitio web que consideres apropiado. Pero asegúrate de que todo lo que hagas no perjudica la usabilidad y la accesibilidad de las páginas web.

Además de usar la biblioteca de iconos, emplea también algún carácter Unicode para representar un icono.

**Importante:** en la siguiente práctica tendrás que hacer varias variantes sobre el CSS de esta práctica, así que desarrolla el CSS de forma modular para poder hacer esas variantes. Para ello, emplea variables y divide el CSS en varios ficheros.

---

<sup>36</sup><https://fontawesome.com/>

<sup>37</sup><https://fontello.com/>

<sup>38</sup><https://fonts.google.com/icons>

## 4. ¿Cómo lo hago?

En las prácticas anteriores aprendiste a evitar 6 de los 10 errores de accesibilidad más comunes que se explican en “Top Ten Most Common Web Accessibility Issues”<sup>a</sup>. En esta práctica vas a aprender a evitar otros 3 errores:

- **Not providing a visual indication of the current focus**

Keyboard-only users, screen magnification users, and speech recognition users, rely on knowing where the current input focus is placed within the page. All too often, a page author fails to take this into consideration and deliberately or inadvertently hides this visual indication, typically by suppressing the outline entirely (e.g. outline: none;).

- **Link areas are too small**

A closely related issue is that web pages sometimes include clickable target areas that are too small to allow users with dexterity issues to successfully interact with those types of elements with a mouse.

- **Color contrast**

Using color combinations within the content that fall outside the WCAG 2.1 allowable contrast ratios (4.5:1 for text content, or 3:1 for interface elements) is a major problem for color blind or low-vision individuals. We also see the mistake of using color as the only means of conveying information, for instance, adding emphasis to a word or section of a website.

Y no te olvides de los que ya has aprendido en las prácticas anteriores:

- Image alt text errors
- Failure to use proper labels
- Non-descriptive text for hyperlinks
- Tables markup
- Improper use of heading elements
- Same descriptive text for different resources

---

<sup>a</sup><https://www.tpgi.com/ten-common-web-accessibility-issues/>

### 4.1. Cómo empezar con CSS

Para definir el estilo visual tienes que emplear el lenguaje de estilos CSS. No tienes que crear un fichero distinto para cada página, sino que el mismo fichero lo utilizarás en todas las páginas, a no ser que existan algunas páginas con estilos visuales muy diferentes (por ejemplo, la página de inicio, la página principal y el resto de páginas).

Tienes varias formas de crear una hoja de estilo CSS:

1. Crear una hoja CSS desde cero: esto lo podrás hacer cuando ya tengas un conocimiento amplio del lenguaje CSS.
2. Tomar un CSS como “inspiración” para crear un CSS propio. En este caso, es una buena práctica citar al autor de la o de las hojas CSS en que te has basado, pero si existen grandes diferencias no es del todo necesario. La cita la puedes indicar mediante un comentario en la misma hoja CSS.
3. Adaptar una hoja CSS a la estructura de tu página o, al revés, adaptar la estructura de tu página a una hoja CSS. Cuando usas directamente el CSS de otra persona, primero tienes que consultar si es posible hacerlo, es decir, si el autor de la hoja CSS te da permiso. Lo normal es que te pida (y si no lo hace, hazlo) que cites claramente al autor del CSS en la página web, pero a veces con un simple comentario en la misma hoja CSS puede ser suficiente.

Si quieres escribir directamente el código CSS, cualquier editor de texto sencillo como el **Bloc de notas** es suficiente. Sin embargo, puedes emplear un programa como **Notepad++**<sup>39</sup>, que posee algunas características que ayudan a escribir el código, como la sintaxis coloreada o la función de autocompletado (pulsa Ctrl+Espacio).

También existen herramientas online como **CodePen**<sup>40</sup>.

Para indicar que se está usando un fichero CSS en una página web se emplea la etiqueta `<link />`. En una misma página web se pueden indicar varios ficheros CSS que se combinan todos ellos o también se puede indicar de forma que definan estilos alternativos. Por ejemplo, en el siguiente código se define un estilo CSS que se debe emplear cuando se imprima una página web (**print**) y un estilo que se debe emplear en el resto de casos (cuando no se indica el atributo **media** de forma explícita, este toma el valor **all** que significa que se aplica a cualquier dispositivo):

```
<link rel="stylesheet" type="text/css" href="home.css" />
<link rel="stylesheet" type="text/css" href="print.css" media="print" />
```

En el ejemplo anterior aparece el atributo `type="text/css"` que ya no es necesario ponerlo en HTML5, pero que lo puedes encontrar en muchos ejemplos.

## 4.2. Destacar los elementos de interacción

Las Pautas de accesibilidad para el contenido web (WCAG) 2.1 tienen unos criterios que se refieren a la accesibilidad mediante el teclado:

**2.1.1 Teclado:** Toda la funcionalidad del contenido es operable a través de una interfaz de teclado sin que se requiera una determinada velocidad para cada pulsación individual de las teclas, excepto cuando la función interna requiere de una entrada que depende del trayecto de los movimientos del usuario y no sólo de los puntos inicial y final. (Nivel A)

**2.4.3 Orden del foco:** Si se puede navegar secuencialmente por una página web y la secuencia de navegación afecta su significado o su operación, los componentes que pueden recibir el foco lo hacen en un orden que preserva su significado y operabilidad. (Nivel A)

**2.4.7 Foco visible:** Cualquier interfaz de usuario operable por teclado tiene una forma de operar en la cuál el indicador del foco del teclado resulta visible. (Nivel AA)

En CSS, una pseudoclase es una palabra clave que se añade a los selectores y que especifica un estado especial del elemento seleccionado.

La pseudoclase `:focus`<sup>41</sup> representa un elemento (por ejemplo, un enlace, un botón o un cuadro de texto de un formulario) que ha recibido el foco. Generalmente, se activa cuando el usuario hace clic, toca un elemento o lo selecciona con la tecla tabulador del teclado. Por ejemplo:

```
/* Selecciona cualquier control <input> y cambia su color de fondo cuando reciba el foco */
input:focus {
    background: red;
}
```

La pseudoclase `:focus-within`<sup>42</sup> representa un elemento que ha recibido el foco o que contiene un elemento que ha recibido el foco. Por ejemplo, se puede aplicar a todo un formulario y se activa cuando el usuario haga clic, toque o seleccione con la tecla tabulador del teclado cualquier elemento contenido dentro del formulario. Por ejemplo:

```
/* Selecciona cualquier formulario <form> y cambia su color de fondo cuando cualquier elemento
que contenga reciba el foco */
form:focus-within {
    background: yellow;
}
```

Además del indicador del foco, también es interesante destacar los elementos de interacción cuando el puntero del ratón se sitúa encima de ellos. Esto se consigue con la pseudoclase `:hover`<sup>43</sup>. Por ejemplo:

---

<sup>39</sup><https://notepad-plus-plus.org/>

<sup>40</sup><https://codepen.io/>

<sup>41</sup><https://developer.mozilla.org/en-US/docs/Web/CSS/:focus>

<sup>42</sup><https://developer.mozilla.org/en-US/docs/Web/CSS/:focus-within>

<sup>43</sup><https://developer.mozilla.org/en-US/docs/Web/CSS/:hover>

```

/* Selecciona cualquier enlace <a> y cambia su color cuando el puntero del ratón se sitúa encima
*/
a:hover {
    color: orange;
}

```

### 4.3. Variables CSS

Las variables CSS, oficialmente conocidas como *CSS Custom Properties*<sup>44</sup>, son un mecanismo que permite escribir hojas de estilo más fáciles de mantener. El primer borrador de la especificación oficial se publicó en el año 2012<sup>45</sup>, pero no fue hasta el año 2016 que la mayoría de los navegadores web implementaron las variables CSS. En la Figura 1 podemos ver el soporte actual (octubre 2019) de las variables CSS en los principales navegadores web. Solo Microsoft Internet Explorer y algunos navegadores minoritarios no soportan esta característica de CSS.

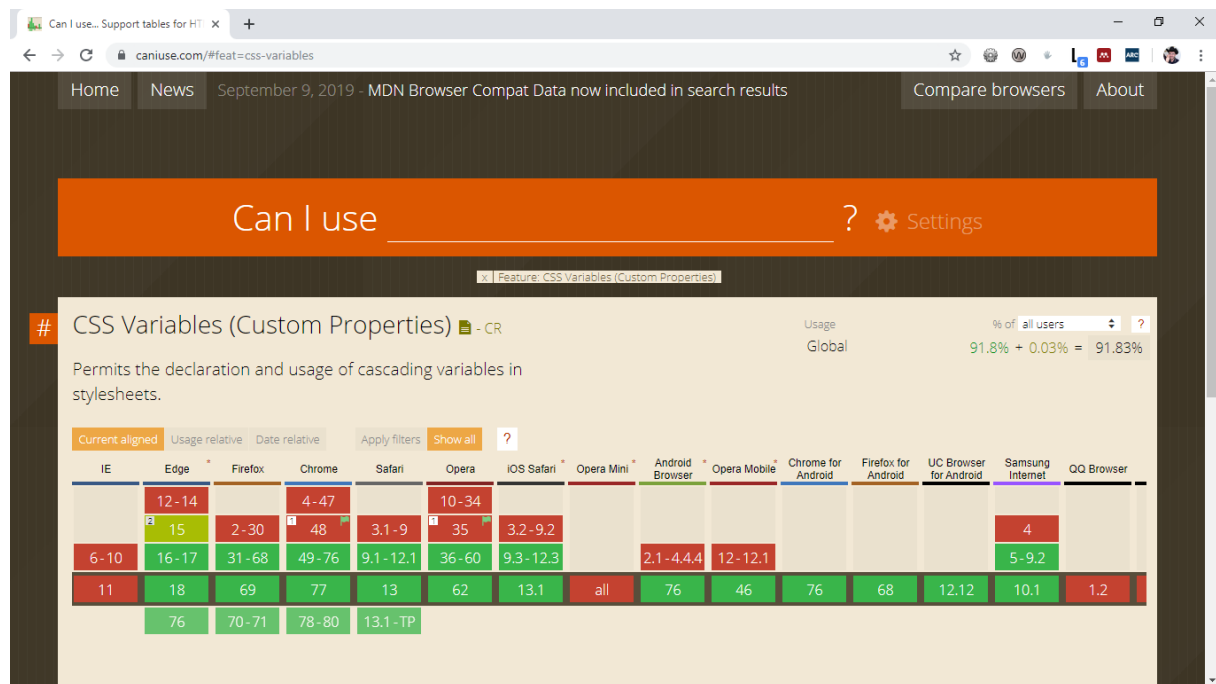


Figura 1: Can I use CSS Variables?

Las variables se declaran con el prefijo `--` (dos guiones) dentro de un selector. El selector `:root` permite declarar las variables en un ámbito global para que estén disponibles dentro de cualquier otro selector. Para utilizar una variable se emplea `var()` y entre paréntesis se escribe el nombre de la variable que se quiere utilizar.

Por ejemplo, el siguiente código define las variables `color1` y `color2` que se emplean para definir el color de dos párrafos:

```

<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Variables CSS</title>
<style>
:root {
    --color1: blue;
    --color2: yellow;
}

```

<sup>44</sup><https://www.w3.org/TR/css-variables-1/>

<sup>45</sup><https://www.w3.org/TR/2012/WD-css-variables-20120410/>

```
#par1 {
    background-color: var(--color1);
    color: var(--color2);
}

#par2 {
    background-color: var(--color2);
    color: var(--color1);
}
</style>
</head>
<body>
<p id="par1">Este párrafo debería tener texto amarillo sobre fondo azul.</p>
<p id="par2">Este párrafo debería tener texto azul sobre fondo amarillo.</p>
</body>
</html>
```

Los nombres de variables de este ejemplo (`--color1`, `--color2`) no son muy buenos, porque no identifican la función del color, ¿qué representa `--color1`? Los nombres de las variables deben ser simples, cortos, sencillos de recordar y deben identificar su función.

Aunque en el ejemplo anterior se han escrito las reglas de CSS directamente en la página web con la etiqueta `<style>`, recuerda que lo correcto es escribirlas en un fichero separado y enlazarlo con la etiqueta `<link />`.

En la Figura 2 se puede ver el resultado del código anterior en un navegador web.

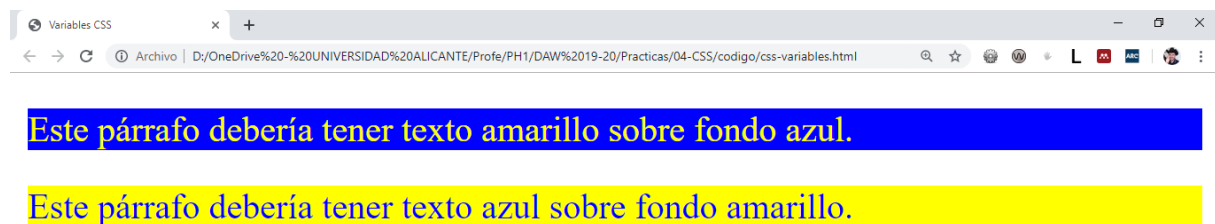


Figura 2: Ejemplo de uso de las variables CSS

En el siguiente ejemplo, dentro del selector `#contenedor` se redefine la variable `color1` que ha sido definido en el ámbito global. Como resultado de este cambio, los dos últimos párrafos no muestran los mismos colores que los dos primeros párrafos, aunque compartan las mismas reglas de CSS. Este ejemplo muestra que los mecanismos de la herencia y la cascada siguen funcionando con las variables.

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Variables CSS</title>
```



```

<style>
:root {
    --color1: blue;
    --color2: yellow;
}

#par1, #par3 {
    background-color: var(--color1);
    color: var(--color2);
}

#par2, #par4 {
    background-color: var(--color2);
    color: var(--color1);
}

#contenedor {
    --color1: green;
}
</style>
</head>
<body>
<p id="par1">Este párrafo debería tener texto amarillo sobre fondo azul.</p>
<p id="par2">Este párrafo debería tener texto azul sobre fondo amarillo.</p>
<div id="contenedor">
    <p id="par3">Este párrafo debería tener texto amarillo sobre fondo verde.</p>
    <p id="par4">Este párrafo debería tener texto verde sobre fondo amarillo.</p>
</div>
</body>
</html>

```

En la Figura 3 se puede ver el resultado del código anterior en un navegador web.

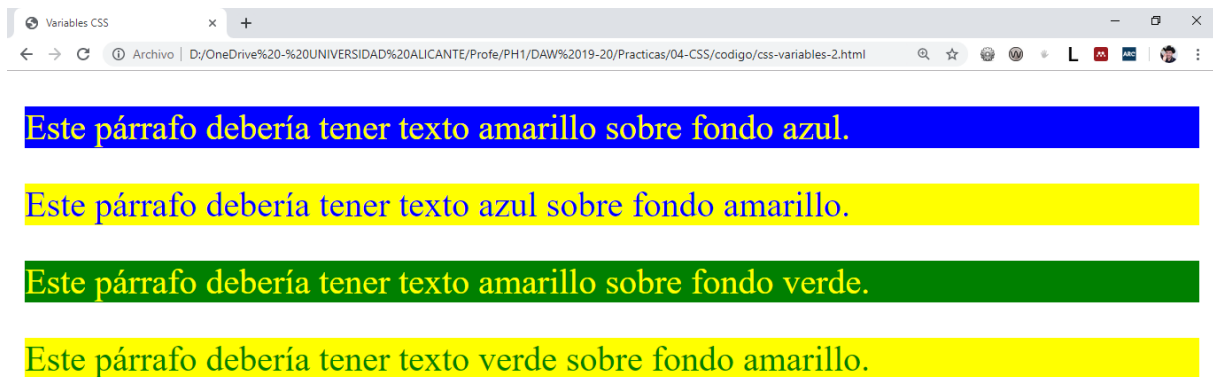


Figura 3: Ejemplo de uso de las variables CSS

## 4.4. Maquetación de una página web

Según el **Diccionario de la Lengua Española** de la Real Academia Española, maquetar<sup>46</sup> es “Hacer la maqueta de una publicación que se va a imprimir”. Si buscamos maqueta, obtenemos varias acepciones y una de ellas dice “Boceto previo de la composición de un texto que se va a publicar, usado para determinar sus características definitivas”.

La maquetación de una página web es el proceso de organizar, estructurar y presentar los elementos visuales y de contenido dentro de una página, siguiendo un diseño previamente definido. Su propósito es traducir un diseño gráfico o una idea conceptual en una estructura web funcional, comprensible y accesible para los usuarios.

En términos más concretos:

- Estructura: Se definen las secciones principales (encabezado, menú de navegación, contenido principal, barra lateral, pie de página, etc.).
- Jerarquía visual: Se organiza el contenido de manera que el usuario entienda qué es lo más importante y cómo debe recorrer la página.
- Estilos y diseño: Se aplican colores, tipografías, espaciados, imágenes y otros recursos visuales para dar coherencia estética.
- Adaptabilidad: Se asegura que la página se vea bien en distintos dispositivos y tamaños de pantalla (*responsive design*).
- Accesibilidad: Una buena maquetación incluye prácticas que faciliten la interacción de personas con diferentes capacidades, siguiendo estándares como las Pautas de accesibilidad para el contenido web (WCAG).

Técnicamente, la maquetación web se lleva a cabo con HTML para la estructura del contenido y CSS para los estilos y la disposición visual.

## 4.5. Maquetación de una tabla

En el siguiente ejemplo se muestran las principales propiedades de CSS que se deben considerar a la hora de maquetar una tabla. El código HTML de la tabla es:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Una tabla</title>
<link href="estilo.css" rel="stylesheet" />
</style>
</head>
<body>
<table>
<caption>Provincias de España</caption>
<thead>
<tr>
<th scope="col">Provincia</th>
<th scope="col">Capital</th>
<th scope="col">Código postal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alicante</td>
<td>Alicante</td>
<td>03</td>
</tr>

<tr>
```

---

<sup>46</sup><https://dle.rae.es/maquetar>

Provincias de España		
Provincia	Capital	Código postal
Alicante	Alicante	03
Barcelona	Barcelona	08
Castellón	Castellón de la Plana	12
Madrid	Madrid	28
Valencia	Valencia	46
Número total de provincias		3

Figura 4: Tabla de ejemplo sin maquetación

```

<td>Barcelona</td>
<td>Barcelona</td>
<td>08</td>
</tr>

<tr>
<td>Castellón</td>
<td>Castellón de la Plana</td>
<td>12</td>
</tr>

<tr>
<td>Madrid</td>
<td>Madrid</td>
<td>28</td>
</tr>

<tr>
<td>Valencia</td>
<td>Valencia</td>
<td>46</td>
</tr>
</tbody>
<tfoot>
<tr>
<th scope="row" colspan="2">Número total de provincias</th>
<td>3</td>
</tr>
</tfoot>
</table>
</body>
</html>

```

Sin aplicar ningún estilo, la tabla se visualiza en un navegador tal como se puede ver en la Figura 4.

Para maquetar el aspecto visual de la tabla, en primer lugar, se define el espaciado y distribución del contenido de la tabla:

Provincias de España		
Provincia	Capital	Código postal
Alicante	Alicante	03
Barcelona	Barcelona	08
Castellón	Castellón de la Plana	12
Madrid	Madrid	28
Valencia	Valencia	46
Número total de provincias		3

Figura 5: Tabla de ejemplo con definición del espaciado y distribución del contenido

```
/* Espaciado y distribución */
table {
    table-layout: fixed;
    width: 100%;
}

thead th:nth-child(1) {
    width: 40%;
}

thead th:nth-child(2) {
    width: 40%;
}

thead th:nth-child(3) {
    width: 20%;
}

th, td {
    padding: 20px;
}
```

El resultado de este fragmento de CSS se muestra en la Figura 5. A continuación, se definen los bordes:

```
/* Bordes */
table, tr, th, td {
    border-collapse: collapse;
    border: 3px solid cyan;
}
```

El resultado de este fragmento de CSS se muestra en la Figura 6. Después se define la tipografía del texto:

```
/* Tipografía */
thead th, tfoot th, caption {
    font-family: Arial, Helvetica, sans-serif;
```

Provincias de España		
Provincia	Capital	Código postal
Alicante	Alicante	03
Barcelona	Barcelona	08
Castellón	Castellón de la Plana	12
Madrid	Madrid	28
Valencia	Valencia	46
Número total de provincias		3

Figura 6: Tabla de ejemplo con definición de los bordes

```

}

th {
    letter-spacing: 2px;
}

td {
    letter-spacing: 1px;
}

tbody td {
    text-align: center;
}

tfoot th {
    text-align: right;
}

caption {
    font-variant: small-caps;
}

```

Y el resultado se muestra en la Figura 7.

A continuación, se define el fondo de la tabla:

```

/* Fondo*/
table {
    background-color: #33ccff;
}

thead, tfoot {
    color: white;
    text-shadow: 1px 1px 1px black;
}

thead th, tfoot th, tfoot td {

```

PROVINCIAS DE ESPAÑA		
Provincia	Capital	Código postal
Alicante	Alicante	03
Barcelona	Barcelona	08
Castellón	Castellón de la Plana	12
Madrid	Madrid	28
Valencia	Valencia	46
Número total de provincias		3

Figura 7: Tabla de ejemplo con la definición de la tipografía

```
background: linear-gradient(to bottom, rgba(0,0,0,0.1), rgba(0,0,0,0.5));
}
```

El resultado de este fragmento de CSS se muestra en la Figura 8. Para las celdas de la cabecera **thead** y el pie **tfoot** de la tabla se ha definido un fondo degradado. Un fondo degradado puede causar problemas de accesibilidad si no se emplea con cuidado. Que se use en este ejemplo no significa que lo tengas que usar en tu práctica.

A continuación se define el estilo cebra para las filas que forman el cuerpo de la tabla:

```
/* Estilo cebra */
tbody tr:nth-child(odd) {
    background-color: #33ccff;
}

tbody tr:nth-child(even) {
    background-color: #c7f2ff;
}
```

Y el resultado se muestra en la Figura 9.

Por último, se define un estilo **hover** para cuando el ratón se sitúa en alguna fila del cuerpo de la tabla:

```
/* Hover */
tbody tr:hover {
    background-color: white;
}
```

Y el resultado se muestra en la Figura 10, en la que se puede apreciar que una fila tiene un color de fondo distinto ya que sobre ella se ha situado el cursor del ratón.

Por último, el CSS completo:

```
/* Espaciado y distribución */
table {
    table-layout: fixed;
    width: 100%;
}
```

PROVINCIAS DE ESPAÑA		
Provincia	Capital	Código postal
Alicante	Alicante	03
Barcelona	Barcelona	08
Castellón	Castellón de la Plana	12
Madrid	Madrid	28
Valencia	Valencia	46
Número total de provincias		3

Figura 8: Tabla de ejemplo con la definición del fondo

PROVINCIAS DE ESPAÑA		
Provincia	Capital	Código postal
Alicante	Alicante	03
Barcelona	Barcelona	08
Castellón	Castellón de la Plana	12
Madrid	Madrid	28
Valencia	Valencia	46
Número total de provincias		3

Figura 9: Tabla de ejemplo con la definición del estilo cebra

PROVINCIAS DE ESPAÑA		
Provincia	Capital	Código postal
Alicante	Alicante	03
Barcelona	Barcelona	08
Castellón	Castellón de la Plana	12
Madrid	Madrid	28
Valencia	Valencia	46
Número total de provincias		3

Figura 10: Tabla de ejemplo con la definición del estilo hover

```
thead th:nth-child(1) {
    width: 40%;
}

thead th:nth-child(2) {
    width: 40%;
}

thead th:nth-child(3) {
    width: 20%;
}

th, td {
    padding: 20px;
}

/* Bordes */
table, tr, th, td {
    border-collapse: collapse;
    border: 3px solid cyan;
}

/* Tipografía */
thead th, tfoot th, caption {
    font-family: Arial, Helvetica, sans-serif;
}

th {
    letter-spacing: 2px;
}

td {
    letter-spacing: 1px;
}
```



```
tbody td {
    text-align: center;
}

tfoot th {
    text-align: right;
}

caption {
    font-variant: small-caps;
}

/* Fondo*/
table {
    background-color: #33ccff;
}

thead, tfoot {
    color: white;
    text-shadow: 1px 1px 1px black;
}

thead th, tfoot th, tfoot td {
    background: linear-gradient(to bottom, rgba(0,0,0,0.1), rgba(0,0,0,0.5));
}

/* Estilo cebra */
tbody tr:nth-child(odd) {
    background-color: #33ccff;
}

tbody tr:nth-child(even) {
    background-color: #c7f2ff;
}

/* Hover */
tbody tr:hover {
    background-color: white;
}
```

## 4.6. Maquetación de un formulario

Para maquetar con CSS un formulario existen varias técnicas. Una sencilla consiste en definir un tamaño fijo para las etiquetas de los controles. Por ejemplo:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Formulario alineado</title>
<style>
fieldset {width: 300px; padding: 10px;}
legend {font-weight: bold;}
label {float: left; display: block; width: 100px;}
input {float: none; display: block;}
.centrado {margin: 10px auto;}
</style>
</head>
<body>
<form id="form1" action="" method="get">
<fieldset>
<legend>Datos personales</legend>
<label for="nombre">Nombre:</label>
```

```

<input type="text" id="nombre" />
<br />
<label for="apellidos">Apellidos:</label>
<input type="text" id="apellidos" />
<input type="submit" value="Enviar" class="centrado" />
</fieldset>
</form>
</body>
</html>

```

Aunque en el ejemplo anterior se han escrito las reglas de CSS directamente en la página web con la etiqueta `<style>`, recuerda que lo correcto es escribirlas en un fichero separado y enlazarlo con la etiqueta `<link />`.

En la Figura 11 podemos observar como se visualiza esta página en un navegador web:

Figura 11: Formulario de ejemplo

Y si se añade una propiedad `text-align: right` se puede lograr otro diseño, tal como podemos ver en la Figura 12:

```

<style>
fieldset {width: 300px; padding: 10px;}
legend {font-weight: bold;}
label {float: left; display: block; width: 100px; text-align: right;}
input {float: none; display: block;}
.centrado {margin: 10px auto;}
</style>

```

Figura 12: Formulario de ejemplo con alineación de las etiquetas a la derecha

Sin embargo, un tamaño fijo no es la mejor opción. Otra opción un poco mejor es basar el tamaño no en píxeles, sino en el tamaño del texto:

```

<style>
fieldset {width: 20em; padding: 0.7em;}
legend {font-weight: bold;}
label {float: left; display: block; width: 6em;}
input {float: none; display: block;}
.centrado {margin: 10px auto;}
</style>

```

Pero todavía se puede hacer mejor, esta solución no es adaptativa. En **How to structure an HTML form**<sup>47</sup> y **Styling HTML forms**<sup>48</sup> se explican algunas técnicas básicas de maquetación de formularios.

<sup>47</sup>[https://developer.mozilla.org/en-US/docs/Learn/Forms/How\\_to\\_structure\\_a\\_web\\_form](https://developer.mozilla.org/en-US/docs/Learn/Forms/How_to_structure_a_web_form)

<sup>48</sup>[https://developer.mozilla.org/en-US/docs/Learn/Forms/Styling\\_web\\_forms](https://developer.mozilla.org/en-US/docs/Learn/Forms/Styling_web_forms)

## Desktop vs Mobile vs Tablet Market Share Worldwide

Q1 2014 - Q3 2024

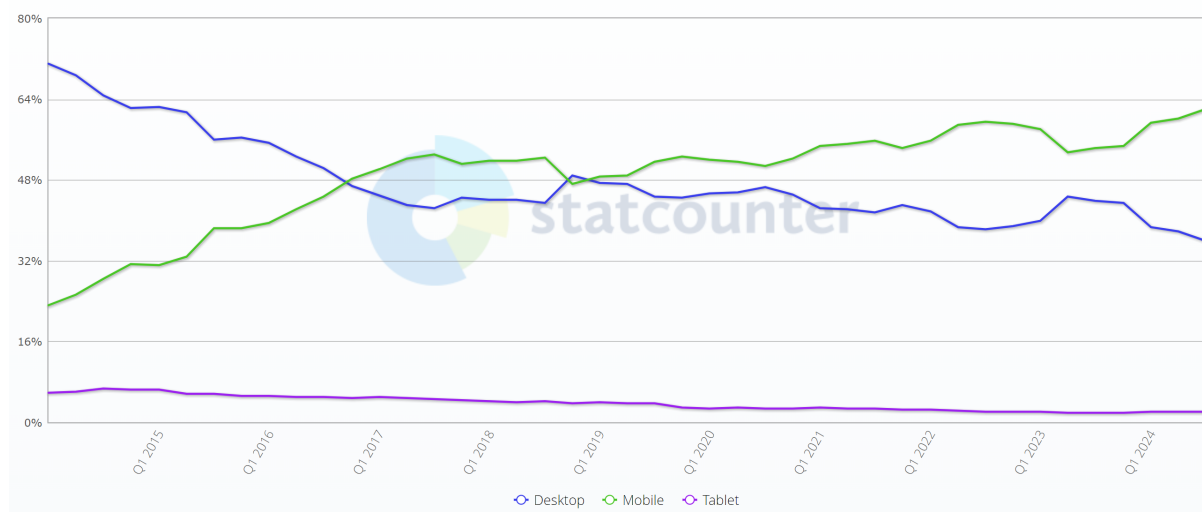


Figura 13: Estadísticas de acceso a la Web por dispositivo entre enero 2014 y septiembre 2024 según StatCounter

Y en **10 CSS HTML Form Designs**<sup>49</sup> se proporcionan 10 estilos visuales de formularios, con el código HTML y CSS.

### 4.7. Estilo adaptativo

¿Hoy en día se accede a la Web exclusivamente desde un ordenador? No, es muy normal acceder a la Web desde un dispositivo móvil como un teléfono inteligente (*smartphone*) y es muy probable que en el futuro se acceda también desde otros tipos de dispositivos que en la actualidad no contemplamos. En realidad, según StatCounter, entre octubre y noviembre de 2016<sup>50</sup>, el acceso a la Web mediante teléfono móvil superó al acceso mediante ordenador, tal como se puede observar en la Figura 13, donde la línea azul representa el acceso mediante ordenador y la línea verde representa el acceso mediante teléfono móvil. Este cambio de tendencia se mantiene hasta la actualidad.

Un buen diseño y una buena maquetación es aquella que está preparada para adaptarse a diferentes dispositivos. ¿Cómo se puede lograr? Una solución que se aplica en la actualidad es el diseño adaptativo, adaptable o flexible (*responsive design*).

El diseño adaptativo se basa principalmente en el uso de **Media Queries**<sup>51</sup>, un estándar del W3C. Una *media query* es una expresión que implica la evaluación de una o varias *media features*, que da como resultado un valor booleano de verdadero o falso.

La *media query* se puede utilizar desde HTML en la etiqueta `<link />` con el atributo `media`:

```
<link href="sans-serif.css" rel="stylesheet" type="text/css" media="screen" />
```

```
<link href="serif.css" rel="stylesheet" type="text/css" media="print" />
```

En el ejemplo anterior, si el dispositivo de visualización es una pantalla, se usa el estilo `sans-serif.css`. Pero si el dispositivo de visualización es la impresora, se usa el estilo `serif.css`.

También se puede utilizar directamente desde CSS con la declaración `@media`:

```
* {font-family: serif;}
```

```
@media screen {  
* {font-family: sans-serif;}  
}
```

<sup>49</sup><https://www.sanwebe.com/2014/08/css-html-forms-designs>

<sup>50</sup><https://gs.statcounter.com/>

<sup>51</sup><https://www.w3.org/TR/mediaqueries-3/>

En el ejemplo anterior, se define un tipo de letra **serif** para todos los dispositivos de visualización, pero si el dispositivo de visualización es una pantalla, el tipo de letra se redefine como **sans-serif** para todos los elementos de la página web.

Una *media query* suele evaluar alguna de estos atributos:

- Tipo de dispositivo.
- Anchura y altura del dispositivo de visualización.
- Orientación del dispositivo (vertical o apaisado).
- Resolución del dispositivo.
- Número de colores del dispositivo.

Por ejemplo, en el siguiente fragmento de código se definen tres hojas de estilo CSS que se aplicarán según la anchura que tenga el dispositivo en el que se visualice la página web:

```
<link href="minimum.css" rel="stylesheet" type="text/css" media="screen and (max-width: 480px)"
/>
<link href="medium.css" rel="stylesheet" type="text/css" media="screen and (min-width: 481px)
and (max-width: 1024px)" />
<link href="maximum.css" rel="stylesheet" type="text/css" media="screen and (min-width: 1025px)"
/>
```

También se puede emplear la instrucción `@import`<sup>52</sup>, que permite importar un CSS desde otro CSS. Esta instrucción es muy útil cuando se quiera modularizar y reutilizar ciertas partes de un CSS.

El ejemplo anterior con tres etiquetas `<link>` en el código HTML, se puede convertir en una única etiqueta:

```
<link href="estilos.css" rel="stylesheet" type="text/css" />
```

Y en el fichero `estilos.css` se importan las tres hojas de estilo:

```
@import url("minimum.css") screen and (max-width: 480px);
@import url("medium.css") screen and (min-width: 481px) and (max-width: 1024px);
@import url("maximum.css") screen and (min-width: 1025px);
```

Para que un diseño adaptativo funcione correctamente, se debe añadir la siguiente instrucción<sup>53</sup> en la sección `<head>` de la página web:

```
<meta name="viewport" content="width=device-width,initial-scale=1.0" />
```

Los siguientes vídeos te pueden ayudar a realizar tu diseño adaptativo:

- **CSS: creación de un diseño adaptable, adaptativo o flexible (parte 1)**<sup>54</sup>: explica el diseño adaptativo, `@media`, patrones de diseño adaptativo (casi fluido, caída de columna, cambio de la estructura, pequeños cambios, fuera de la pantalla).
- **CSS: creación de un diseño adaptable, adaptativo o flexible (parte 2)**<sup>55</sup>: muestra el desarrollo de un ejemplo completo con diseño adaptativo desde cero y algunas herramientas (Window Resizer, Opera Mobile Emulator).
- **CSS: creación de un diseño adaptable, adaptativo o flexible (parte 3)**<sup>56</sup>: proporciona referencias y fuentes de información para aprender más cosas sobre diseño adaptativo.

## 4.8. Biblioteca de iconos

El empleo de iconos en una página web ayuda a mejorar la usabilidad y la accesibilidad, ya que permite una identificación visual de los elementos de la página. Por ejemplo, en la Figura 14 se puede ver un formulario que tiene un icono de una persona para representar el nombre de usuario y un icono de un candado para representar la contraseña. Y a la derecha hay un panel para identificarse mediante redes sociales, donde cada opción tiene su icono correspondiente.

<sup>52</sup><https://developer.mozilla.org/en-US/docs/Web/CSS/@import>

<sup>53</sup>[https://developer.mozilla.org/en-US/docs/Web/HTML/Viewport\\_meta\\_tag](https://developer.mozilla.org/en-US/docs/Web/HTML/Viewport_meta_tag)

<sup>54</sup><https://youtu.be/ktGF7Dda0wI>

<sup>55</sup><https://youtu.be/96U4zPbyq9c>

<sup>56</sup><https://youtu.be/cbtJ4udg7sM>

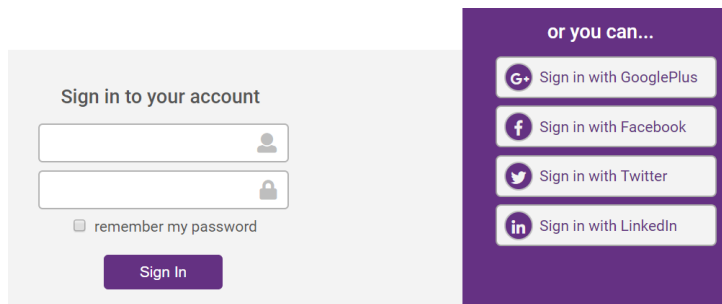


Figura 14: Formulario de acceso con iconos

Sin embargo, si se eliminan los iconos (por ejemplo, porque el usuario decide no cargar las imágenes), aparece un problema importante de usabilidad porque no se sabe qué escribir en los dos cuadros de texto del formulario, tal como se puede ver en la Figura 15. Además, este problema de usabilidad evidencia que también hay un problema de accesibilidad: los usuarios que utilicen un lector de pantalla (*screen reader*) no sabrán qué escribir en cada cuadro de texto, aunque aparezcan los iconos.

Este problema de usabilidad y accesibilidad no surge en el panel de la derecha porque, además de los iconos, existe la explicación textual de cada opción (“Sign in with...”).

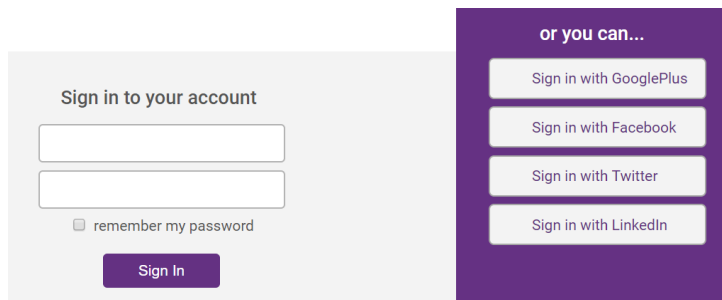


Figura 15: Formulario de acceso sin iconos

Font Awesome<sup>57</sup> es una biblioteca de iconos ampliamente utilizada en el desarrollo web. Funciona como una colección de símbolos (iconos vectoriales) que se pueden integrar fácilmente en páginas y aplicaciones mediante código. Para hacer uso de Font Awesome es necesario crear una cuenta gratuita y configurar un proyecto. El proyecto proporciona un código personalizado que se debe incluir en las páginas web en las que se quiera usar Font Awesome.

En la Figura 16 se muestra el buscador de iconos de Font Awesome. Al localizar un icono, se puede copiar el código HTML necesario para usar un icono.

Por ejemplo, el siguiente código muestra en una página web los iconos de usuario, cadandado, GooglePlus, Facebook, Twitter y LinkedIn (los mismos iconos que aparecen en la Figura 14):

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Prueba de iconos de Font Awesome</title>
<script src="https://kit.fontawesome.com/xxxxxxxx.js" crossorigin="anonymous"></script>
</head>
<body>
<p>Algunos iconos de Font Awesome:
<i class="fa-solid fa-user"></i> <!-- Icono de usuario -->
<i class="fa-solid fa-lock"></i> <!-- Icono de candado -->
</p>

<p>Iconos de marcas:
```

<sup>57</sup><https://fontawesome.com/>

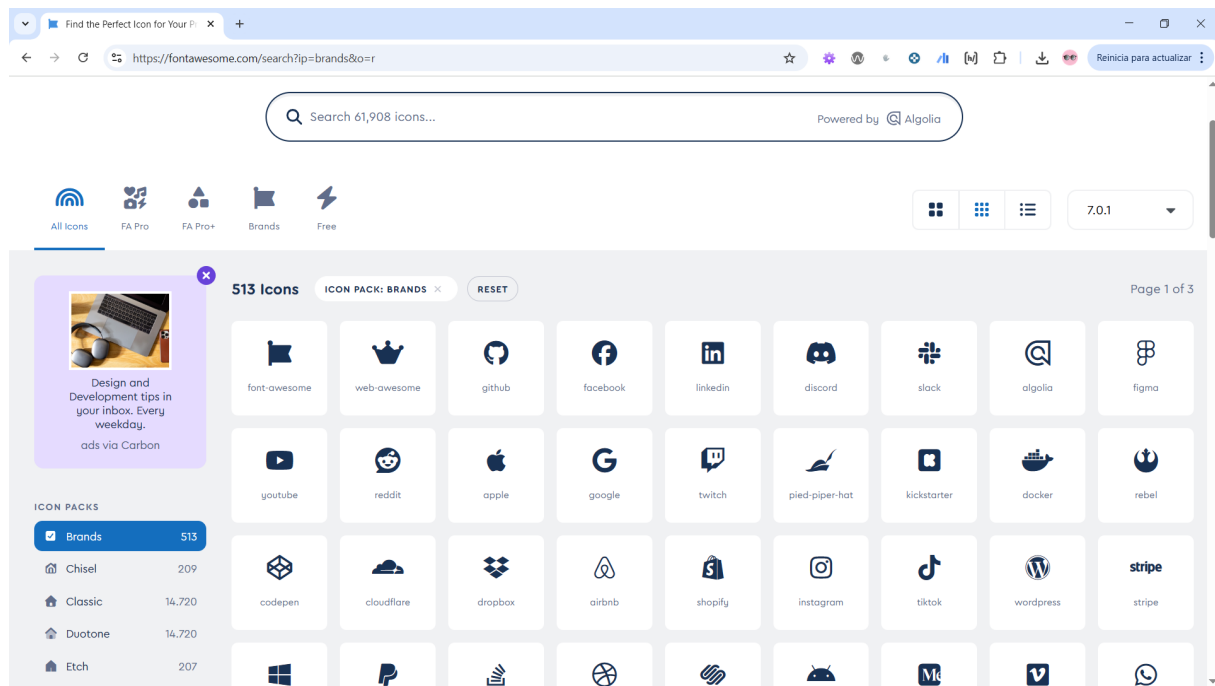


Figura 16: Buscador de iconos de Font Awesome

Algunos iconos de Font Awesome:  





Iconos de marcas:    

Figura 17: Ejemplo de uso de Font Awesome

```
<i class="fa-brands fa-google-plus"></i> <!-- Icono de GooglePlus -->
<i class="fa-brands fa-facebook"></i> <!-- Icono de Facebook -->
<i class="fa-brands fa-twitter"></i> <!-- Icono de Twitter -->
<i class="fa-brands fa-linkedin"></i> <!-- Icono de LinkedIn -->
</p>
</body>
</html>
```

En el código anterior, `xxxxxxxx.js` se tiene que sustituir por el código que proporciona Font Awesome al crear un proyecto. En la Figura 17 se puede ver cómo se visualiza el código anterior en un navegador web.

En la página Accessibility<sup>58</sup> se explica cómo hacer que los iconos no sean meramente decorativos (el comportamiento por defecto) y un lector de pantalla los pueda anunciar. Simplemente se tiene que añadir el código `role="img"` para que el icono se comporte como una imagen y el atributo `aria-label` con el texto alternativo de la imagen.

Por ejemplo, el siguiente es el código del ejemplo anterior que se ha modificado para que sea accesible:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Prueba de iconos de Font Awesome</title>
<script src="https://kit.fontawesome.com/xxxxxxxx.js" crossorigin="anonymous"></script>
</head>
<body>
<p>Algunos iconos de Font Awesome:
```

<sup>58</sup><https://docs.fontawesome.com/web/dig-deeper/accessibility>

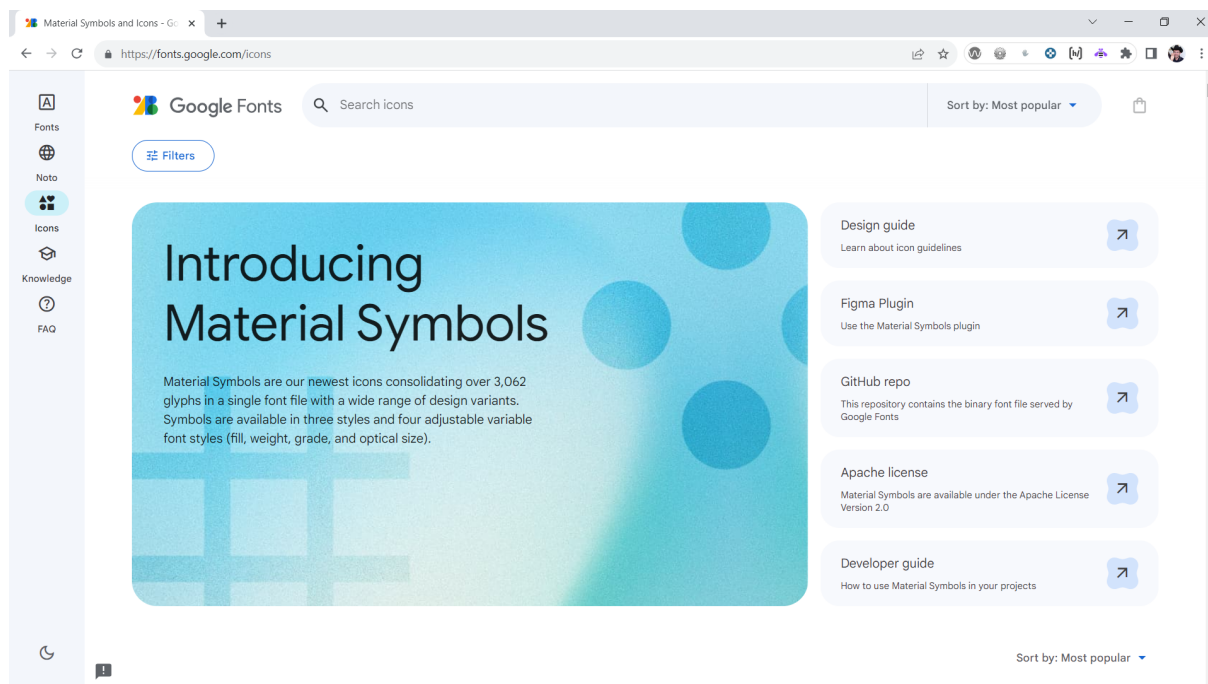


Figura 18: Página principal de Material Symbols de Google

```
<i class="fa-solid fa-user" aria-label="Usuario" role="img"></i> <!-- Icono de usuario -->
<i class="fa-solid fa-lock" aria-label="Candado" role="img"></i> <!-- Icono de candado -->
</p>

<p>Iconos de marcas:
<i class="fa-brands fa-google-plus" aria-label="GooglePlus" role="img"></i> <!-- Icono de
  GooglePlus -->
<i class="fa-brands fa-facebook" aria-label="Facebook" role="img"></i> <!-- Icono de Facebook
  -->
<i class="fa-brands fa-twitter" aria-label="Twitter" role="img"></i> <!-- Icono de Twitter -->
<i class="fa-brands fa-linkedin" aria-label="LinkedIn" role="img"></i> <!-- Icono de LinkedIn
  -->
</p>
</body>
</html>
```

Material Symbols & Icons<sup>59</sup> es otra biblioteca de iconos muy popular en la actualidad.

No obstante, en la actualidad no es imprescindible el uso de una biblioteca de iconos si solo se quiere mostrar unos pocos iconos y son iconos comunes. Unicode tiene miles de caracteres para representar todo tipo de iconos. En la Web, existen muchas páginas web<sup>60</sup><sup>61</sup><sup>62</sup> con tablas de caracteres Unicode que se pueden emplear para buscar un carácter concreto.

Por ejemplo, en el siguiente código se incluyen tres caracteres Unicode que representan el nombre de usuario, el teléfono y la dirección de correo electrónico. Los caracteres se han insertado de dos formas: primero directamente, si así lo permite el editor de código que se esté utilizando, y después con su código Unicode decimal mediante la notación `&#nnnn`;. En este código, los caracteres `***` representan el carácter Unicode directamente insertado en el código<sup>63</sup>. En la Figura 21 se puede ver cómo los caracteres Unicode sí que aparecen directamente en el editor Notepad++.

```
<!doctype html>
<html lang="es">
```

<sup>59</sup><https://fonts.google.com/icons>

<sup>60</sup><https://symbl.cc/en/unicode-table/>

<sup>61</sup>[https://en.wikipedia.org/wiki/List\\_of\\_Unicode\\_characters](https://en.wikipedia.org/wiki/List_of_Unicode_characters)

<sup>62</sup><https://www.vertex42.com/ExcelTips/unicode-symbols.html>

<sup>63</sup>El compilador  $\text{\LaTeX}$  que se emplea para crear este documento tiene problemas con la codificación UTF-8, así que no se pueden insertar y visualizar en este documento.

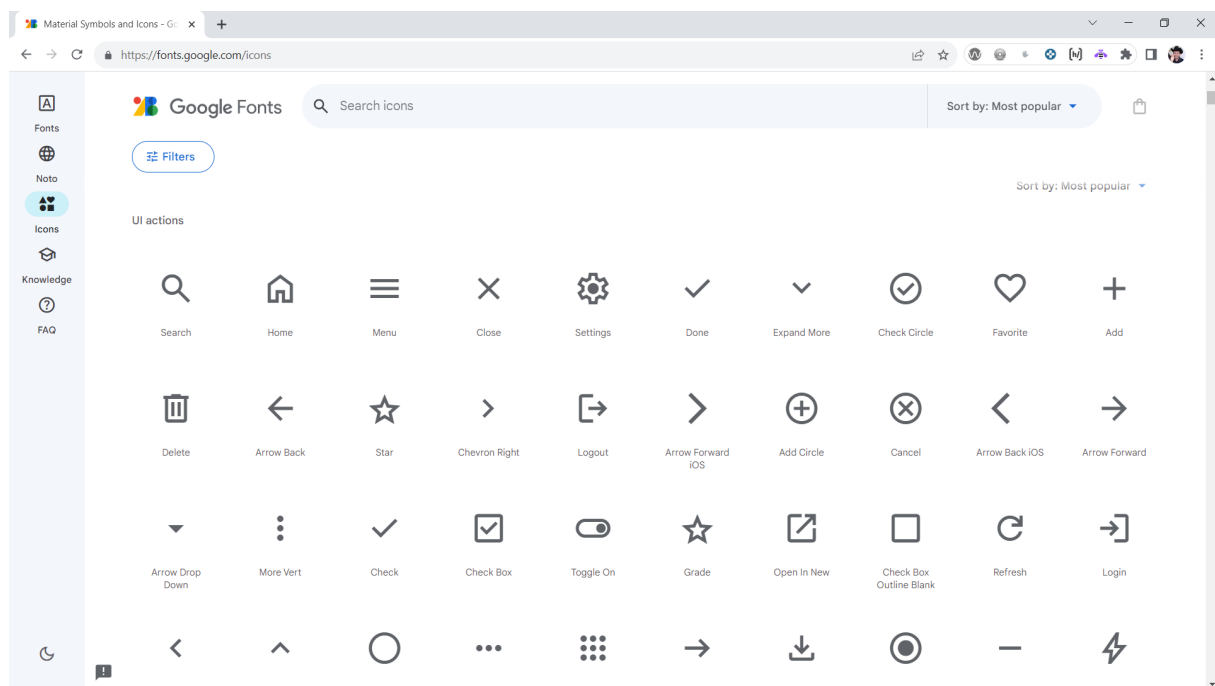


Figura 19: Iconos de Material Symbols de Google

## Material Symbols



Figura 20: Ejemplo de uso de Material Symbols



```

<head>
<meta charset="utf-8">
<title>Prueba de iconos con Unicode</title>
</head>
<body>
<form>
<p>
<label>*** Usuario:
<input type="text"></label>
</p>

<p>
<label>*** Teléfono:
<input type="tel"></label>
</p>

<p>
<label>*** Correo:
<input type="email"></label>
</p>

<p><input type="submit" value="Enviar"></p>
</form>

<form>
<p>
<label>&#128100; Usuario:
<input type="text"></label>
</p>

<p>
<label>&#9742; Teléfono:
<input type="tel"></label>
</p>

<p>
<label>&#9993; Correo:
<input type="email"></label>
</p>

<p><input type="submit" value="Enviar"></p>
</form>
</body>
</html>

```

Y en la Figura 22 se puede observar cómo se visualiza el código anterior en el navegador Google Chrome. Se puede observar que los iconos que se muestran en Notepad++ y en Google Chrome no son exactamente los mismos.

## 4.9. Enlaces “saltar a”

Los enlaces “saltar a” (*skip to*) ayudan a mejorar la accesibilidad para las personas que usan el teclado en vez del ratón como elemento apuntador. Esto es útil para los usuarios ciegos que emplean un lector de pantalla, pero también para otros usuarios no ciegos que tengan movilidad reducida y no puedan emplear un ratón.

Estos enlaces pueden estar siempre visibles, como en el ejemplo que se muestra en la Figura 23, o pueden estar ocultos y mostrarse únicamente cuando reciben el foco, como en el ejemplo que se muestra en la Figura 24.

En “Skip Navigation Links”<sup>64</sup> se explica la utilidad de estos enlaces y en “Invisible Content Just for Screen Reader Users”<sup>65</sup> se explica cómo implementar estos enlaces mediante CSS.

<sup>64</sup><https://webaim.org/techniques/skipnav/>

<sup>65</sup><https://webaim.org/techniques/css/invisiblecontent/>

```
C:\Users\Sergio\Downloads\prueba-unicode.html - Notepad++
Archivo  Editor  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
prueba-unicode.html
4  <meta charset="utf-8">
5  <title>Prueba de iconos con Unicode</title>
6  </head>
7  <body>
8  <form>
9  <p>
10 <label>👤 Usuario:
11 <input type="text"></label>
12 </p>
13
14 <p>
15 <label>☎ Teléfono:
16 <input type="tel"></label>
17 </p>
18
19 <p>
20 <label>✉ Correo:
21 <input type="email"></label>
22 </p>
23
24 <p><input type="submit" value="Enviar"></p>
25 </form>
26
```

Figura 21: Código fuente en Notepad++

Prueba de iconos con Unicode

👤 Usuario:

☎ Teléfono:

✉ Correo:

👤 Usuario:

☎ Teléfono:

✉ Correo:

Figura 22: Ejemplo de página web con caracteres Unicode

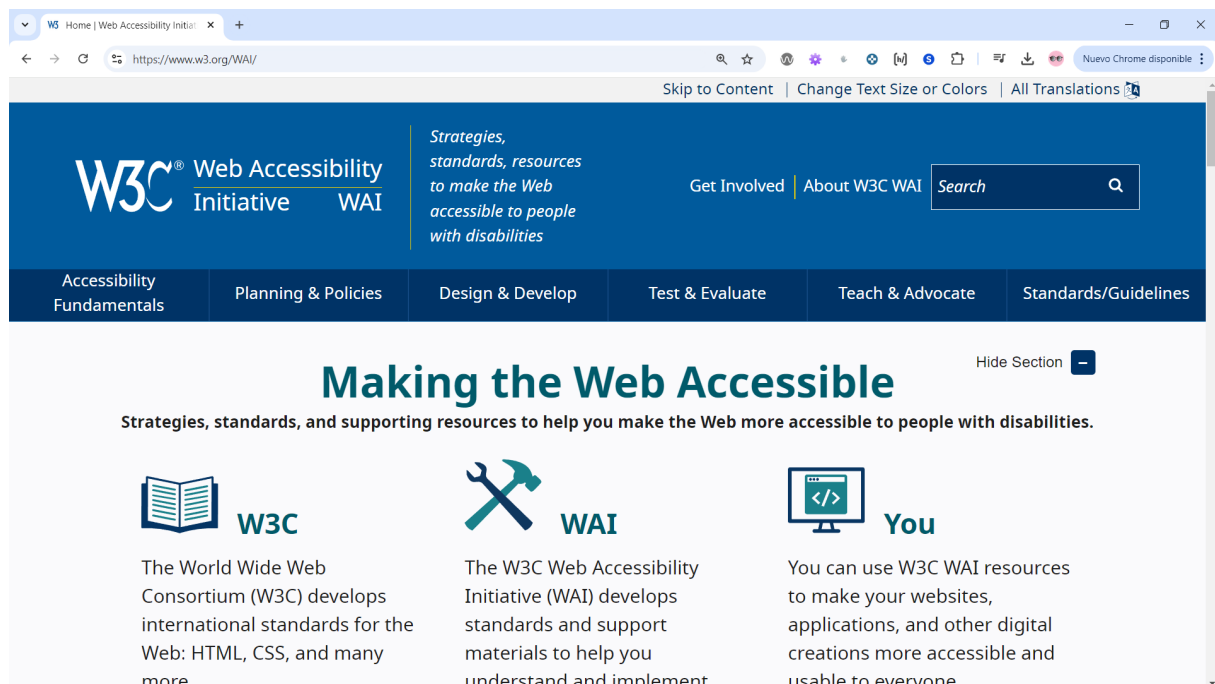


Figura 23: Página web del W3C-WAI con enlace “Skip to content” siempre visible

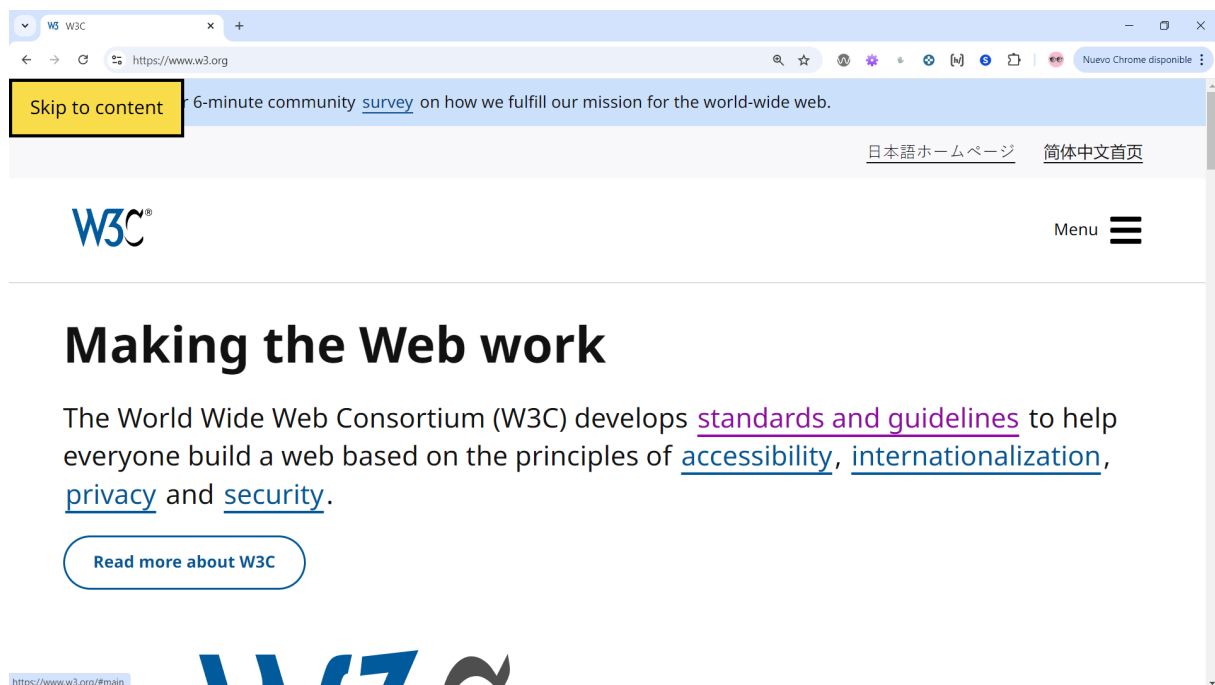


Figura 24: Página web del W3C con enlace “Skip to content” que aparece cuando recibe el foco

## 4.10. Consejos básicos de usabilidad

En “Usability Guidelines for Accessible Web Design”<sup>66</sup> se proporcionan algunos consejos de usabilidad que debes aplicar en tu sitio web. En las prácticas anteriores has tenido que aplicar los consejos para “GRAPHICS AND MULTIMEDIA”, “LINKS AND BUTTONS”, “PAGE ORGANIZATION”, “FORMS AND FIELDS”, “SEARCH” y “TABLES AND FRAMES”, no los olvides. Entre esos consejos, había algunos que hacían referencia a la presentación de las páginas web y hasta ahora no habías podido aplicar:

### ■ LINKS AND BUTTONS

- 18. Avoid very small buttons and tiny text for links.
- 19. Leave space between links and buttons.
- 22. Underline all links

Los consejos 18 y 19 son muy importantes cuando las páginas web se visualizan en los dispositivos móviles en los que se emplea el dedo en vez del ratón como elemento apuntador. El consejo 22 se cumple por defecto, los enlaces siempre aparecen subrayados.

### ■ PAGE ORGANIZATION

- 30. Design pages consistently.
- 31. Carefully consider using “Skip Links” so users can skip links or navigational elements.

### ■ FORMS AND FIELDS

- 37. Do not use only red text or yellow highlighting to indicate form errors.
- 38. Do not rely on only an asterisk (\*) to indicate required fields.
- 39. Make sure tab order is logical.
- 40. Match the tab order to the visual layout when possible.

### ■ TABLES AND FRAMES

- 67. Avoid using tables for aesthetic page design.

Las tablas no se deben emplear para maquetar el contenido de una página web.

En esta práctica también tienes que aplicar:

### ■ PRESENTING TEXT

- 47. Choose text colors for good contrast.
- 48. Do not use very small text for body text.
- 49. Do not use small or subtle text headings and categories.
- 50. Always create good contrast between text and the page background.
- 51. Do not rely on a background image as a page background to create contrast with text.
- 52. Test your site’s text fonts and colors with screen magnifiers.
- 53. Make sure it is possible to magnify your site.
- 54. Write concisely, and remove superfluous text.
- 55. If the company name includes an initialism or acronym, tell screen readers how to pronounce it (you should do this for all abbreviations of this type on the site).
- 56. Rethink how you use parentheses and asterisks.

---

<sup>66</sup>[https://media.nngroup.com/media/reports/free/Usability\\_Guidelines\\_for\\_Accessible\\_Web\\_Design.pdf](https://media.nngroup.com/media/reports/free/Usability_Guidelines_for_Accessible_Web_Design.pdf)

## 5. Recomendaciones

Los mapas de imagen no funcionan con el diseño adaptativo cuando la imagen se hace más pequeña, porque cambian las dimensiones de la imagen y las coordenadas de las zonas sensibles, pero las coordenadas en la etiqueta `<area>` no cambian. ¿Cómo se puede solucionar?

Existen varias soluciones. Una opción es usar JavaScript para cambiar las coordenadas (escalar las coordenadas en función de lo que se haya escalado la imagen), lo cual está fuera de los objetivos de esta asignatura. Otra solución, más moderna, es usar SVG, pero también está fuera de los objetivos de esta asignatura. Por tanto, lo puedes dejar sin solucionar. Pero si quieres intentarlo y aprender algo más, aquí tienes unas páginas web que te pueden ayudar:

- “The Best Way to Make Image Maps (2020)”<sup>67</sup>.
- “How to Implement Responsive Image Maps in your HTML5 Outputs”<sup>68</sup>.
- “Responsive Image Map Generator”<sup>69</sup>.

El estilo que definas no debe ser muy simple (no vale con cambiar el color de fondo de la página y definir un estilo para el texto), pero tampoco es necesario que sea muy complejo. Con unas pocas reglas de CSS bien elegidas se puede lograr un estilo visual impactante. **Y no es necesario que implementes animaciones y transiciones: si lo quieres hacer para aprender más, puedes hacerlo, pero no cuenta para la evaluación de la práctica.**

En general, se recomienda que como mínimo se utilicen tres colores diferentes (y suele ser lo suficiente en la mayoría de los casos) y no más de seis.

En general, se recomienda que se utilicen dos tipos de letra (un tipo de letra para los títulos y otro para el contenido principal) y como máximo tres (un tipo de letra para contenido adicional, como puede ser una barra lateral, o para destacar algo en el texto principal). Si quieres usar tipos de letra especiales, prueba con Google Fonts<sup>70</sup>.

Una mala práctica es definir múltiples clases (`class`) e identificadores (`id`). Siempre que sea posible, emplea selectores simples que hagan referencia a elementos de HTML.

Recuerda que hay propiedades que se heredan, por lo que no es necesario definir las para todos los elementos, sino que si se define en un elemento padre, todos sus hijos la tomarán. Por ejemplo, para aplicar un mismo estilo de texto en toda una página web, lo más sencillo es definirlo en el elemento `html` o `body`. Pero a veces un elemento no hereda una propiedad, aunque otros elementos sí que lo hagan. El ejemplo típico de esto es el enlace, que no hereda el color, aunque otros elementos sí que lo hacen.

Un fichero CSS puede llegar a tener cientos o incluso miles de líneas, por lo que pueden aparecer problemas importantes:

- Reglas no utilizadas.
- Reglas duplicadas, que indican lo mismo o son contradictorias.
- Propiedades heredadas que pasan desapercibidas.
- Por supuesto, el tiempo que se pierde para localizar una regla entre miles de líneas.

Para evitar estos problemas, es necesario organizar un fichero CSS de alguna forma. Cada uno tiene que desarrollar su estilo propio, con el que se sienta más a gusto. Algunas posibilidades son:

- Organizar las reglas por orden alfabético. Por ejemplo: `a`, `div`, `em`, `.importante`, `#principal`, `table`.
- Organizar las reglas por orden de aparición. Por ejemplo: `html`, `body`, `h1`, `h2`, `div`, `#principal`, `p`, `.importante`.
- Separar las reglas según la función que realizan. Por ejemplo: por un lado las reglas que modifican los bordes, por otro lado las reglas que modifican los colores, por otro lado las reglas que definen la maquetación.

---

<sup>67</sup><https://vecta.io/blog/the-best-way-to-make-image-maps>

<sup>68</sup><https://www.madcapsoftware.com/blog/how-to-implement-responsive-image-maps-in-your-html5-outputs/>

<sup>69</sup><https://zaneray.com/responsive-image-map/>

<sup>70</sup><https://fonts.google.com/>

- Organizar las propiedades de una regla por orden alfabético. Por ejemplo: `border`, `font-size`, `margin`, `padding`, `text-align`.
- Organizar las propiedades de una regla del exterior del elemento al interior. Por ejemplo: `margin`, `border`, `padding`, `text-align`, `font-size`.

También existen algunas herramientas que te pueden ayudar a organizar un fichero CSS. **Format CSS Code**<sup>71</sup> es una herramienta online que formatea el aspecto de un CSS: introduce saltos de línea, espacios en blanco, etc. **CSS Optimizer**<sup>72</sup> es otra herramienta online que ayuda a reducir el tamaño de un fichero CSS al aplicar algunas transformaciones, como por ejemplo, cambiar el nombre de algunas propiedades y valores.

Utiliza los comentarios de CSS `/* ... */` para separar las diferentes partes de un CSS y para desactivar ciertas propiedades cuando quieras realizar pruebas.

¿Estás realizando un diseño adaptativo o adaptable? En español nos “faltan palabras”, cuesta distinguir adaptativo (*Perteneciente o relativo a la adaptación o a la capacidad de adaptación*<sup>73</sup>) y adaptable (*Capaz de ser adaptado*<sup>74</sup>), pero en inglés se distinguen dos tipos de diseño, *responsive design* y *adaptive design*. Los siguientes artículos explican las diferencias entre estos dos tipos de diseño:

- Responsive vs Adaptive Design? Which is Best for Mobile Viewing of Your Website?<sup>75</sup>
- The Difference Between Responsive and Adaptive Design<sup>76</sup>
- RWD Is Not AWD, What Is the Difference Between Responsive and Adaptive Design?<sup>77</sup>

**Y recuerda: evita la divitis**<sup>78</sup>, **la spanmania** y **la classitis**<sup>79</sup>.

**Importante:**

- Antes de añadir una etiqueta `<div>` porque quieres aplicar un estilo desde CSS, párate a pensar si puedes usar alguna de las etiquetas que ya tiene tu página.
- Se recomienda usar el juego de caracteres UTF-8. En las hojas de estilo también se puede indicar el juego de caracteres usado con la instrucción `@charset`<sup>80</sup>.
- Ten cuidado con los nombres de los ficheros, utiliza únicamente letras del alfabeto inglés y números, no uses espacios en blanco y emplea únicamente minúsculas.
- Ten cuidado con la caché del navegador, consulta una explicación sobre los problemas y soluciones en el artículo “Ayuda:Cómo limpiar la caché”<sup>81</sup>. Recuerda esto durante toda la asignatura. En concreto, en esa página dice:
  - Firefox: Presiona la combinación de teclas `Ctrl + Mayús + R`. (Alternativamente, presiona la combinación de teclas `Ctrl + F5`).
  - Chrome: Presiona la combinación de teclas `Ctrl + F5`.

Pero en la ayuda oficial de Chrome, en “Combinaciones de teclas de Chrome”<sup>82</sup>, dice:

- Volver a cargar la página actual: `F5` o `Ctrl + R`.
- Volver a cargar la página actual, ignorando el contenido almacenado en caché: `Mayús + F5` o `Ctrl + Mayús + R`.

Así que, prueba tanto con `Ctrl + F5` como `Mayús + F5`.

<sup>71</sup><https://www.cssportal.com/format-css/>

<sup>72</sup><https://www.cssportal.com/css-optimize/>

<sup>73</sup><https://dle.rae.es/adaptativo>

<sup>74</sup><https://dle.rae.es/adaptable>

<sup>75</sup><https://mediumwell.com/responsive-adaptive-mobile/>

<sup>76</sup><https://css-tricks.com/the-difference-between-responsive-and-adaptive-design/>

<sup>77</sup><https://www.mockplus.com/blog/post/difference-between-responsive-and-adaptive>

<sup>78</sup><https://web.archive.org/web/20230602005539/https://csscreator.com/divitis>

<sup>79</sup>[https://web.archive.org/web/20170409230755/https://www.456bereastreet.com/lab/web\\_development\\_mistakes/](https://web.archive.org/web/20170409230755/https://www.456bereastreet.com/lab/web_development_mistakes/)

es/

<sup>80</sup><https://developer.mozilla.org/en-US/docs/Web/CSS/@charset>

<sup>81</sup>[https://es.wikipedia.org/wiki/Ayuda:C%C3%B3mo\\_limpiar\\_la\\_cach%C3%A9](https://es.wikipedia.org/wiki/Ayuda:C%C3%B3mo_limpiar_la_cach%C3%A9)

<sup>82</sup><https://support.google.com/chrome/answer/157179?hl=es>

## 6. ¿Qué debo entregar?

- Todos los ficheros que componen el sitio web.
- Un informe, en formato PDF, a modo de hoja de comprobación (*checklist*), en el que indiques qué has hecho para mejorar la accesibilidad de tus páginas web con CSS y dónde lo has hecho (**no tienes que repetir las características de HTML que se detallaron en el informe previo**).

En la hoja de comprobación debes indicar el o los criterios de conformidad (éxito) de Web Content Accessibility Guidelines (WCAG) 2.x, junto con su nivel (A, AA, AAA), que se corresponde a cada característica de accesibilidad. Para realizar esta parte de la práctica debes consultar la última versión recomendada de WCAG<sup>83</sup>.

Por ejemplo:

Característica	WCAG 2.x	Descripción
...	...	...
<b>Contraste</b>	1.4.3 Contrast (Minimum) (AA)	El color del texto y del fondo tiene un contraste mínimo de 4.5:1 para el texto normal y 3:1 para el texto grande.
...	...	...
...	...	...
<b>Foco</b>	1.4.1 Use of Color (A)  2.4.7 Focus Visible (AA)	El foco se distingue mediante el subrayado del elemento y mediante un cambio en el color de fondo.
...	...	...
...	...	...
<b>Orientación</b>	1.3.4 Orientation (AA)	El diseño adaptativo permite que la página se visualice correctamente tanto en horizontal como en vertical.
...	...	...

- Un fichero de texto con la URL del sitio web publicado.

---

<sup>83</sup><https://www.w3.org/TR/WCAG/>