

## **HITO 2 INDIVIDUAL SEGUNDO TRIMESTRE**

Carlos Parro Pérez

CAMPUS FP – GETAFE

2º DAW – DESARROLLO DE INTERFACES WEB

Carmelo Escribano

8 de Marzo de 2022

## Índice

ELEMENTOS Y/O ETIQUETAS HTML USAD@S EN EL PROYECTO .....	5
DTD .....	5
Idioma .....	5
Viewport.....	5
Título.....	6
Secciones .....	6
Encabezados .....	7
Párrafos .....	7
Listas. ....	8
Marcas estructurales vs. Marcas de formato.....	8
Enlaces. ....	8
Imágenes.....	9
SVG.....	9
Font Awesome. ....	9
Tablas. ....	10
Tablas multinivel.....	10
Formularios.....	11
Etiquetado.....	11
Multimedia. ....	13
CSS .....	14
Diseño Adaptativo vs Diseño Responsive.....	14
Grid vs Flex .....	15
Mejoras de Legibilidad, Contraste y Foco.....	16
Legibilidad.....	16
Contraste.....	17
Foco.....	20
USO DE JAVASCRIPT PARA AÑADIR ACCESIBILIDAD .....	21
Navegación mediante teclado .....	22
Saltar al contenido principal .....	22
Componentes de Usuario.....	22
Cambios de contexto .....	23
Componentes de interfaz de usuario .....	23

Menú de navegación.....	23
Patrones de diseño.....	23
Presentación del menú.....	23
Navegación mediante el teclado .....	27
Implementación mediante JavaScript.....	28
Conclusión .....	29
BIBLIOGRAFÍA .....	30
Affde .....	30
Anna Monus.....	30
Programación Tecnología.....	30

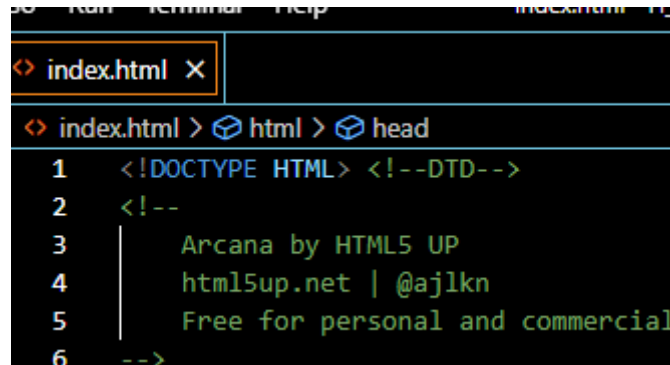
## Índice de Figuras

Ilustración 1 .....	5
Ilustración 2 .....	5
Ilustración 3 .....	5
Ilustración 4 .....	6
Ilustración 5 .....	6
Ilustración 6 .....	6
Ilustración 7 .....	7
Ilustración 8 .....	7
Ilustración 9 .....	7
Ilustración 10 .....	8
Ilustración 11 .....	8
Ilustración 12 .....	8
Ilustración 13 .....	9
Ilustración 14 .....	9
Ilustración 15 .....	9
Ilustración 16 .....	10
Ilustración 17 .....	12
Ilustración 18 .....	13
Ilustración 19 .....	14
Ilustración 20 .....	15
Ilustración 21 .....	16
Ilustración 22 .....	17
Ilustración 23 .....	18
Ilustración 24 .....	18
Ilustración 25 .....	19
Ilustración 26 .....	20
Ilustración 27 .....	20
Ilustración 28 .....	20
Ilustración 29 .....	22
Ilustración 30 .....	24
Ilustración 31 .....	24
Ilustración 32 .....	25
Ilustración 33 .....	25
Ilustración 34 .....	25
Ilustración 35 .....	27
Ilustración 36 .....	27
Ilustración 37 .....	28

## ELEMENTOS Y/O ETIQUETAS HTML USAD@S EN EL PROYECTO

### DTD.

Especifica las marcas que se pueden emplear en un lenguaje de marcas, así como las marcas que pueden ser hijas de otras marcas. Está altamente recomendado su uso en HTML5 ya que incluye mejoras, relacionadas con la accesibilidad, con respecto a versiones anteriores.

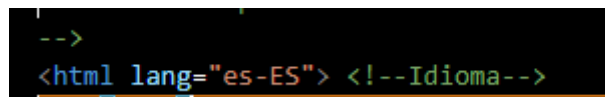
A screenshot of a code editor with a dark theme. The top bar shows 'index.html' with a close button. Below it, a breadcrumb trail reads 'index.html > html > head'. The main code area shows a DTD declaration: line 1: <!DOCTYPE HTML> <!--DTD-->, line 2: <!--, line 3: | Arcana by HTML5 UP, line 4: | html5up.net | @ajlkn, line 5: | Free for personal and commercial, line 6: -->.

```
<!--DTD-->
<!--
| Arcana by HTML5 UP
| html5up.net | @ajlkn
| Free for personal and commercial
-->
```

Ilustración 1

### Idioma.

Especifica el idioma principal de las páginas.

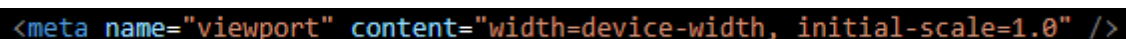
A screenshot of a code editor with a dark theme. It shows a code snippet for setting the language attribute on the HTML tag: <html lang="es-ES"> <!--Idioma-->.

```
<html lang="es-ES"> <!--Idioma-->
```

Ilustración 2

### Viewport.

Se especifica cómo se comporta la web con respecto a resoluciones menores a las empleadas habitualmente en equipos de escritorio, haciendo coincidir el ancho del viewport con la anchura física del dispositivo donde se esté visualizando la web, y que el nivel de zoom sea el predeterminado. Todo ello para que no se pierda información en diferentes dispositivos y configuraciones.

A screenshot of a code editor with a dark theme. It shows the viewport meta tag: <meta name="viewport" content="width=device-width, initial-scale=1.0" />.

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

Ilustración 3

## Título.

Título descriptivo del contenido de cada página que informa al usuario y a las tecnologías de asistencia dónde se sitúa dentro de un sitio web.

```
<title>HITO INDIVIDUAL DIW</title>
```

Ilustración 4

## Secciones.

Emplea las marcas de sección de HTML5 (<header>, <nav>, <main>, <footer>, etc.) para delimitar e identificar las diferentes secciones empleadas en la página. Permite la ubicación sencilla de la estructuración del contenido.

```
<main>
  <!-- Highlights -->
  <section class="wrapper style1">
    <div class="container">
      <div class="row gtr-200">
        <section class="col-4 col-12-narrower">
          <div class="box highlight">
            <i class="icon solid major fa-crow"></i>
            <h3>Por el aire</h3>
            <p>Las aves son animales vertebrados, de sangre
          </div>
        </section>
        <section class="col-4 col-12-narrower">
          <div class="box highlight">
            <i class="icon solid major fa-hippo"></i>
            <h3>Por tierra</h3>
            <p>Son animales que viven predominante o totalm
          </div>
        </section>
        <section class="col-4 col-12-narrower">
```

Ilustración 5

```
<!-- CTA -->
<section id="cta" class="wrapper style3">
  <div class="container">
    <header>
      <h2>Disfrute de la mejor experiencia con animales</h2>
      <a href="galeria.html" class="button">Galería</a>
    </header>
  </div>
</section>
```

Ilustración 6

```

<!-- Footer -->
<div id="footer">
  <div class="container">
    <div class="row">
      <section class="col-3 col-6-narrower col-12-mobilep">
        <h3>Inicio</h3>
        <ul class="links">
          <li><a href="index.html">Página principal</a></li>
        </ul>
      </section>
      <section class="col-3 col-6-narrower col-12-mobilep">
        <h3>Imágenes</h3>
        <ul class="links">
          <li><a href="galeria.html">Galería</a></li>
        </ul>
      </section>
      <section class="col-3 col-6-narrower col-12-mobilep">
        <h3>Contáctanos</h3>
        <ul class="links">
          <li><a href="contacto.html">Formulario de contacto</a></li>
        </ul>
      </section>
      <section class="col-3 col-6-narrower col-12-mobilep">
        <h3>Horarios</h3>
        <ul class="links">
          <li><a href="horarios.html">Cuándo visitarnos</a></li>
        </ul>
      </section>
    </div>
  </div>
</div>

```

Ilustración 7

## Encabezados.

Una vez estructurada las diferentes áreas de la página con las secciones, debemos estructurar los contenidos de cada sección mediante encabezados.

Las secciones sólo deben incluir un encabezado de primer nivel, no se deben saltar niveles y deben tener contenido.

```

<!-- Gigantic Heading -->
<section class="wrapper style2">
  <div class="container">
    <header class="major">
      <h2>Safari para todos los públicos</h2>
      <p>Ven a descubrir a los animales más increíbles del mundo</p>
    </header>
  </div>
</section>

```

Ilustración 8

## Párrafos.

Etiquetas <p>. Definen un bloque de texto conformado por oraciones y especifica el cambio de idioma que se pueda producir. No es buena práctica el usar div para añadir texto de ésta manera ya que nuestro objetivo es tratar de identificar adecuadamente un tipo de contenido en específico. Mejora la legibilidad sobre el texto.

```

<h3>Por tierra</h3>
<p>Son animales que viven predominante o totalmente en la tierra, a difer
</div>

```

Ilustración 9

## Listas.

Permiten definir términos (<dl>, <dt>, <dd>) y la descripción de enumeraciones desordenadas y ordenadas de elementos, para dar semántica adecuada al contenido que se pretende comunicar (secuencias de elementos con orden no relevante, pasos con orden relevante o secuencias de instrucciones).

```
<!-- Icons -->
<ul class="icons">
  <li><a href="https://www.facebook.com/safarimadrid/" class="icon brands fa-facebook">
  <li><a href="https://github.com/Chaarly99/" class="icon brands fa-github">
  <li><a href="https://www.linkedin.com/in/carlos-parro-p%C3%A9rez-5907aa">
  <li><a href="https://www.instagram.com/safarimadridoficial/" class="icon brands fa-instagram">
</ul>
```

Ilustración 10

## Marcas estructurales vs. Marcas de formato.

Debido a la ausencia en los orígenes del HTML de un lenguaje de diseño (Css), se introdujeron marcas de formato como <b> o <i> para aplicar formatos específicos a los textos.

Hoy en día debemos evitar el uso de marcas de formato y sólo emplear marcas estructurales como <strong> y <em>.

```
>SAFARI <em>CAMPUS</em>
```

Ilustración 11

## Enlaces.

HTML además de un lenguaje de marcado es un lenguaje de hipertexto en la cual los contenidos de unos documentos se referencian/relacionan con los contenidos de otros mediante hipervínculos o enlaces.

Si el enlace emplea una imagen, ésta deberá incluir un texto alternativo mediante el atributo **alt** el cual puede ser comunicado a una tecnología de asistencia a la accesibilidad, que puede ejecutar los enlaces mediante comandos de voz.

Si incluyen texto e imagen los textos deben de ser diferentes. (Enlaces gráficos)

```
<a href="galería.html">Galería</a>
```

Ilustración 12



## Imágenes.

Las imágenes ofrecen información visual al usuario pero deben de suministrar información de la misma manera que otros mecanismos alternativos que lo suministren.

```
<div class="image">
  
  <p id="tigrepeq">Cachorro de tigre</p>
  
  <p id="dromedarios">Dromedarios</p>
  
  <p id="ave">Ave rapaz</p>
</div>
```

Ilustración 13

El atributo **alt** incluye el texto alternativo a la imagen que se mostrará en el caso que la imagen no se pueda cargar.

Si la imagen es meramente decorativa no hace falta comunicar su propósito a una tecnología de asistencia a la accesibilidad, por lo cual, bastará con asignar la cadena vacía al atributo alt.

Si la imagen es difícilmente expresable por su texto alternativo tendremos que aportar una descripción por separado de la imagen.

## SVG.

Si se incluyen gráficos vectoriales mediante svg (Scalable Vector Graphics), se deben de identificar como imágenes, mediante el atributo role de WAI-ARIA, y añadirle un texto alternativo mediante el atributo aria-label de WAI-ARIA.

```
<svg width="100" height="100">
  <defs>
    <pattern id="image3" patternUnits="userSpaceOnUse" height="100" width="100">
      <image x="0" y="0" height="100" width="100" xlink:href="media/dromedarios.webp"></image>
    </pattern>
  </defs>
  <circle id='top' cx="50" cy="50" r="50" fill="url(#image3)"/>
</svg>
```

Ilustración 14

## Font Awesome.

Si se incluyen iconos de Font Awesome se deben ocultar los iconos mediante el atributo aria-hidden de WAI-ARIA y proporciona una descripción alternativa para los lectores de pantalla, mediante algún elemento de HTML, cuya clase no muestre en pantalla, dicho texto alternativo.

```
<i class="icon solid major fa-crow" aria-hidden="true" title="Cuervo"></i>
```

Ilustración 15

## Tablas.

Debemos utilizar las tablas para la presentación de información categorizada mediante encabezados.

Es conveniente y no obligatorio el uso de <caption> para identificar el propósito de la tabla, para que a la hora de identificar las tablas de una página, mediante tecnologías de asistencia a la accesibilidad sean capaces de identificarlas y que el usuario pueda decidir si está interesado, o no, en su lectura.

### Tablas multinivel

Si la tabla incluye múltiples niveles de encabezado es necesario identificarlos mediante su id con el atributo **id** y referenciarlos desde cada celda mediante el atributo headers para que las tecnologías de asistencia a la accesibilidad puedan diferenciarlos sin dar lugar a equívoco.

```
<table class="default" aria-describedby="descripción">
  <caption>Disponibilidad</caption>
  <tr>
    <td></td>
    <th id="d1">Lunes</th>
    <th id="d2">Martes</th>
    <th id="d3">Miércoles</th>
    <th id="d4">Jueves</th>
    <th id="d5">Viernes</th>
    <th id="d6">Sábado</th>
    <th id="d7">Domingo</th>
  </tr>
  <tr>
    <th id="t1"></th>
    <td headers="d1 t1">10:00 - 14:00</td>
    <td headers="d2 t1">10:00 - 14:00</td>
    <td headers="d3 t1">10:00 - 14:00</td>
    <td headers="d4 t1">10:00 - 14:00</td>
    <td headers="d5 t1">10:00 - 14:00</td>
    <td headers="d6 t1">10:00 - 14:00</td>
    <td headers="d7 t1" style="text-align: center;">-</td>
  </tr>
  <tr>
    <th id="t2"></th>
    <td headers="d1 t2">16:00 - 20:00</td>
    <td headers="d2 t2">16:00 - 20:00</td>
    <td headers="d3 t2">16:00 - 20:00</td>
    <td headers="d4 t2">16:00 - 20:00</td>
    <td headers="d5 t2">16:00 - 20:00</td>
    <td headers="d6 t2" style="text-align: center;">-</td>
    <td headers="d7 t2" style="text-align: center;">-</td>
  </tr>
</table>
```

Ilustración 16

## Formularios.

Nos permiten recabar información del usuario para procesarla.

Se pueden usar tanto campos de texto, como casillas de verificación, como botones de opción, listas desplegables...

Debemos de organizar los controles de los formularios en áreas específicas mediante la marca **<fieldset>**, esto permite la agrupación de los controles para su correcta comprensión y cumplimentación.

## Etiquetado

Los controles de formularios deben ser correctamente etiquetados con los atributos correspondientes ayudando así a las tecnologías de asistencia a la accesibilidad a interactuar con ellos.

Para los campos de texto debemos identificar el campo mediante el atributo **id** y referenciarlo con el atributo **for** de la marca **<label>**.

Para los botones de opción y casillas de verificación, los identificaremos mediante **<fieldset>** y **<legend>**.

Las listas desplegables también emplearán el etiquetado **<label>**. Si sus valores pueden ser agrupados es conveniente hacerlo mediante la marca **<optgroup>**.

Es conveniente incluir el atributo **placeholder** para los campos de texto, campos de tipo number, o textarea.

Los formularios disponen, además, de otros tipos de controles que les permiten el envío de los datos (**<input type=submit>**) y el restablecimiento del formulario a sus valores originales (**<input type=reset>**).

Un problema es que las tecnologías de asistencia a la accesibilidad no son capaces de establecer relación entre un marcado y otro. Una de las soluciones posibles es el uso del atributo **aria-label** de WAI-ARIA, mediante el cual se pueden etiquetar elementos interactivos que carecen de un etiquetado visual.

El criterio de accesibilidad 2.5.3. nos dice que en estos casos el nombre accesible (el valor del atributo **aria-label**, en nuestro ejemplo: **Buscar**) debe ser igual o contener al texto empleado como etiquetado visible (en nuestro ejemplo, el texto del botón **Buscar**).

Evita las repeticiones visuales innecesarias de información, sin dejar de comunicar el propósito de los controles a las tecnologías de asistencia a la accesibilidad.

Una vez formulado y etiquetados sus controles tenemos en cuenta las **instrucciones** que nos ayudan a cumplimentar el formulario, dirigido especialmente a gente mayor con problemas de comprensión de la información a cumplimentar; las **validaciones** que informen a los usuarios que campos del formulario son obligatorios o para sujetar los campos a algún tipo de formato en

concreto, informando así de los posibles errores que se puedan cometer al proporcionar la información; las **notificaciones** para informar si la información ha sido enviada o no de manera que el usuario tenga constancia de ello(feedback); **información personal**, el criterio 1.3.5. requiere la identificación del propósito de cada campo de texto que solicite información personal sobre el usuario mediante la utilización del atributo **autocomplete** y un valor que identifique dicho tipo de información.

Aquí tienes alguno de los valores más frecuentemente usados:

- Nombre: given-name
- Apellidos: family-name
- Puesto de trabajo: organization-title
- Empresa: organization
- Sexo: sex
- Email: email

```

<section class="col-6 col-12-narrower">
  <h3>Pongámonos en contacto</h3>
  <form>
    <fieldset>
      <legend>Formulario de contacto</legend>
      <div class="row gtr-50">
        <div class="col-6 col-12-mobilep">
          <label for="email"><abbr title="campo obligatorio">*</abbr>
            Email
            <input type="email" name="email" id="email" placeholder="Email" aria-describedby="Email" required="required" autocomplete="email"/>
            <span id="formatoemail">Formato Email: pepito@gmail.com</span>
          </label>
        </div>
      </div>
    </fieldset>
    <br>
    <!-- Casillas de verificación -->
    <fieldset>
      <legend>Seleccione las semanas en las que se encuentra
        interesado</legend>
      <input id="Semana1" type="checkbox" name="Semana1"
        value="Semana1" />
      <label for="Semana1">Semana1</label>
      <input id="Semana2" type="checkbox" name="Semana2"
        value="Semana2" />
      <label for="Semana2">Semana2</label>
    </fieldset>
    <br>
    <label for="Dia">Seleccione un día de la semana</label>
    <select id="Dia" name="Dia">
      <option value="">Sin seleccionar</option>
      <optgroup label="Laborales">
        <option value="Lunes">Lunes</option>
        <option value="Martes">Martes</option>
        <option value="Miércoles">Miércoles</option>
        <option value="Jueves">Jueves</option>
        <option value="Viernes">Viernes</option>
      </optgroup>
      <optgroup label="Festivos">
        <option value="Sabado">Sabado</option>
        <option value="Domingo">Domingo</option>
      </optgroup>
    </select>
    <br>
    <br>
    <div class="col-12">
      <textarea name="mensaje" id="mensaje" placeholder="Mensaje" rows="5"></textarea>
    </div>
    <br>
    <div class="col-12">
      <ul class="actions">
        <li><input type="submit" class="button alt" value="Enviar" /></li>
        <li><input type="reset" class="button alt" value="Borrar" /></li>
      </ul>
    </div>
    <!-- Mensaje de envío correcto
    <p class="StatusOK">El formulario ha sido enviado con éxito.</p>
    Mensaje de envío incorrecto
    <p class="StatusERROR">Se ha producido un error en el envío del
    formulario.</p>-->
  </form>
</section>

```

Ilustración 17

## Multimedia.

Los contenidos audiovisuales o multimedia representan una barrera para muchos usuarios que no pueden escucharlos, visualizarlos o entenderlos (algunas personas procesan mejor la información escrita que audiovisual).

A la hora de facilitar el acceso a los contenidos multimedia debemos seguir las instrucciones siguientes como Subtítulos en los videos, Descripción de audio o Transcripciones.

Los subtítulos y las descripciones de audio se incluyen mediante la marca **<track>**.

```
<!-- Video -->
<video src="media/Mini Spot Safari Madrid.mp4" poster="media/SafariCamp.png" controls="controls">
  <!-- <source src="fichero.mp4" type="video/mp4" />
  <source src="fichero.webm" type="video/webm" /> -->
  <!-- Subtitulos y descripciones de audio -->
  <track src="fichero.vtt" kind="captions" srclang="es"
    label="español" />
  <track src="fichero.vtt" kind="descriptions" srclang="es" />
</video>
<!-- Audio -->
<audio src="media/zoo.mp3" controls="controls">
  <source src="fichero.mp3" type="audio/mpeg" />
  <source src="fichero.wav" type="audio/wav" />
</audio>
```

Ilustración 18

## CSS

### Diseño Adaptativo vs Diseño Responsive

Vamos a establecer un diseño **Responsive** el cual nos permitirá reestructurar los elementos de la web optimizando todo el espacio disponible para ofrecer una buena experiencia al usuario (UX). Para ello se necesita utilizar **media queries** en las hojas de estilo css y fijar las medidas con respecto a la resolución del dispositivo que visualiza el sitio web.

```
@media screen and (max-width: 1680px) {  
    .container {  
        width: 1200px;  
    }  
}  
  
@media screen and (max-width: 1280px) {  
    .container {  
        width: 960px;  
    }  
}  
  
@media screen and (max-width: 980px) {  
    .container {  
        width: 95%;  
    }  
}  
  
@media screen and (max-width: 840px) {
```

Ilustración 19

## Grid vs Flex

Ambos nos permiten generar diseños complejos. **Flexbox** es unidimensional, mientras que **Grid** es bidimensional. Flexbox establece elementos a lo largo del eje horizontal o vertical, lo que te obliga a decidir si emplearás un diseño basado en filas o en columnas.

En la teoría está bien pero no es una teoría basada en la realidad ya que ambos se pueden utilizar para los mismos objetivos, aunque uno lo trate con más precisión que otro, por ejemplo, el diseño 2D está enfocado en grid aunque flex con su capacidad de envoltura puede realizar las mismas funciones.

```
.container {  
  display: grid;  
  grid-template-columns: repeat(auto-fill, minmax(15rem, 1fr));  
  grid-gap: 1rem;  
}
```

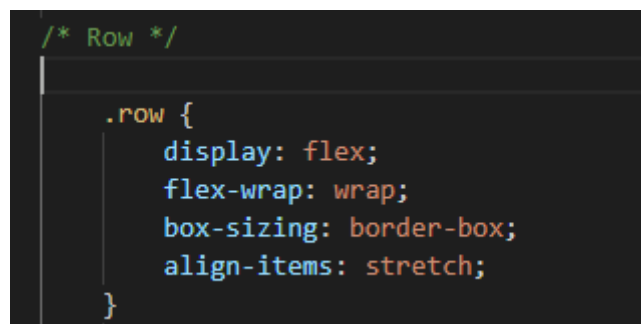
Flexbox nos permite alinear elementos y asignar espacios de una forma menos precisa pero nos da más flexibilidad a la hora de colocar los elementos.

```
.container {  
  display: flex;  
  flex-wrap: wrap;  
  margin: -0.5rem;  
}  
.item {  
  margin: 0.5rem;  
}
```

Aquí no existen columnas ni cuadrículas, sino diseño 1D.

Otra característica importante de flexbox es que nos da buena compatibilidad con los navegadores ya que Grid no es compatible con IE11 ni Edge 15.

En este proyecto utilizamos Flexbox ya que nos da más juego a la hora de establecer los elementos en nuestra web y aporta más flexibilidad.

A screenshot of a code editor showing CSS code for a flexbox container. The code is as follows:

```
/* Row */  
  
.row {  
  display: flex;  
  flex-wrap: wrap;  
  box-sizing: border-box;  
  align-items: stretch;  
}
```

Ilustración 20

## Mejoras de Legibilidad, Contraste y Foco

### Legibilidad

A la hora de establecer una mejora de Legibilidad nos adentramos en hacer que la tipografía sea más fácil de leer. Para ello existe una propiedad que controla el renderizado del texto `text-rendering` y puede tomar los siguientes valores:

- **optimizeLegibility** – Que da prioridad a la legibilidad del texto. Habilita el kerning y la ligadura de la tipografía.
- **auto** – por defecto usa `optimizeLegibility` para texto de menor tamaño que 20px y `optimizeSpeed` para los de mayor tamaño.
- **optimizeSpeed** – Que da prioridad a la velocidad con que se muestra un texto.
- **geometricPrecision** – Da prioridad a la geometría del texto.

Para saber si tu sitio web tiene buena legibilidad, debemos de fijarnos, gracias a la ayuda de herramientas de accesibilidad y usabilidad, en cuanto **tasa de rebote**, lo que significa que los lectores interactúen o no con otra página de su sitio antes de abandonarla.

Punto de referencia general para la tasa de rebote:

26-40% = excelente

41-55% = aproximadamente promedio

56-70% = más alto que el promedio

70%> = decepcionante

A la hora de realizar su página, debe preguntarse si los lectores querrán salir de su página antes de haber tenido la oportunidad de leer la mayor parte del contenido.

Debido a que un gran porcentaje de lectores acceden a las páginas web desde dispositivos móviles, es necesario que la legibilidad sea óptima y, por lo tanto, la web sea atractiva para el lector.

Para mejorarla también existe la posibilidad de uso de imágenes, fotos y presentaciones visuales de texto, ser consciente del tipo y color de la fuente, utilizar viñetas o números, encabezados y subtítulo o evitar grandes bloques de texto entre otras...

```
body {  
  line-height: 1;  
  text-rendering: optimizeLegibility;  
}
```

Ilustración 21



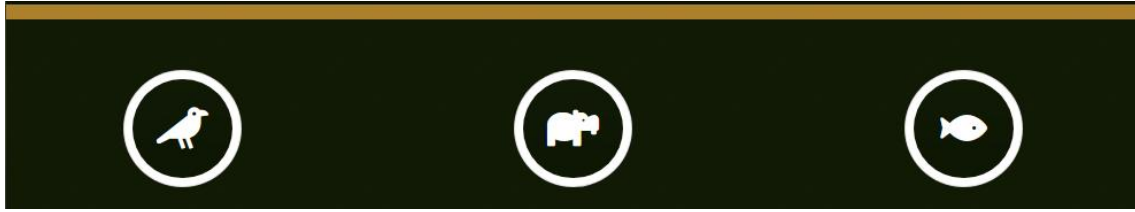


Ilustración 22

## Contraste

Si buscas que una web obtenga ese toque especial y consiga el considerado **efecto wow**, se debe tener en cuenta el contraste de los elementos para diferenciarlos.

Ello se puede conseguir a través de tipografías con peso visual, añadir tamaño a la tipografía, añadiendo textura al sitio web para un mayor enfoque visual, evitar que el sitio se sienta plano, añadir elementos inesperados y/o elegir una paleta de color que llame la atención teniendo en cuenta los siguientes en cuanto a rueda de color:

- **Complementarios:** Los colores de alta intensidad encajan con sus opuestos en la rueda de color.
- **Triádicos:** Colores espaciados en un tercio se incrementan en la rueda.
- **Divididos y complementarios:** Un color y los dos colores al lado de un matiz complementario.

Mezclar colores cálidos, fríos y neutrales, claros y oscuros, elementos sin color (en blanco negro).

**Según WCAG 2.0 para el contraste de colores, para el cumplimiento del nivel AA, el texto debe tener una relación de contraste de 4.5:1**

**Para el nivel AAA, el texto debe tener una relación de contraste de 7:1. Sin embargo, si su texto es grande (más de 18 puntos o 14 puntos si el texto está en negrita), entonces su relación puede ser de 3:1 para el nivel AA y de 4,5:1 para el nivel AAA.**

**Con la combinación de colores elegida conseguimos la siguiente puntuación en cuanto a Ratio de contraste y niveles de Accesibilidad:**

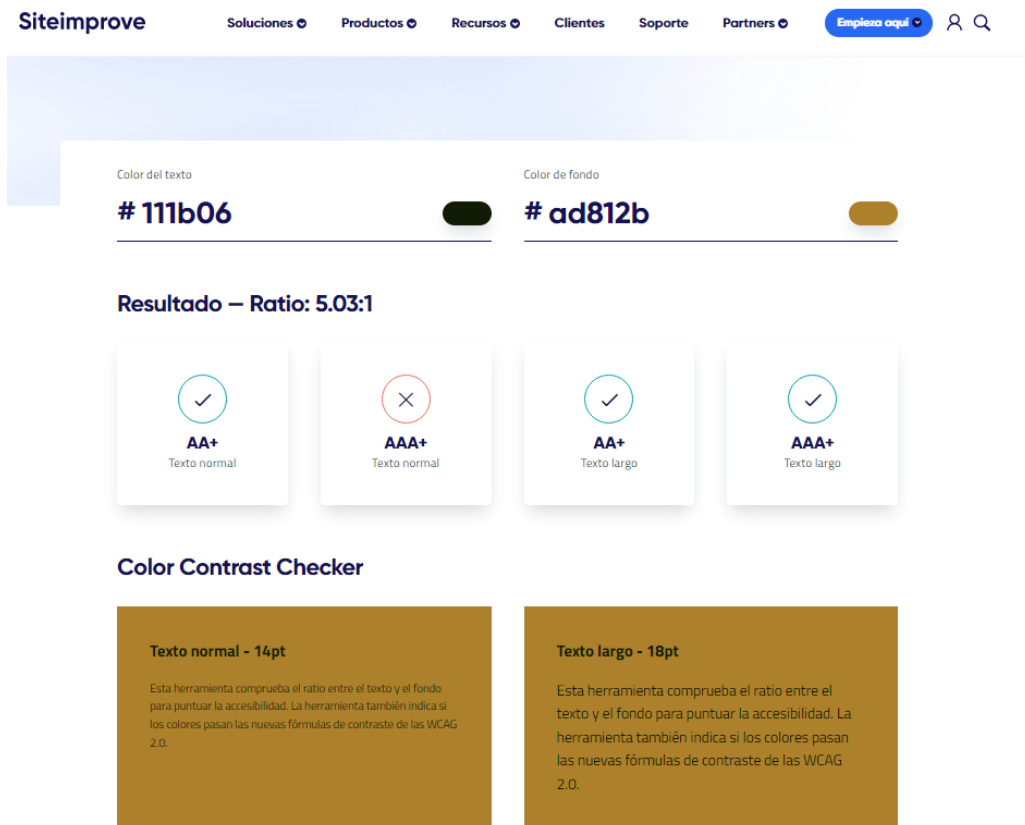


Ilustración 23

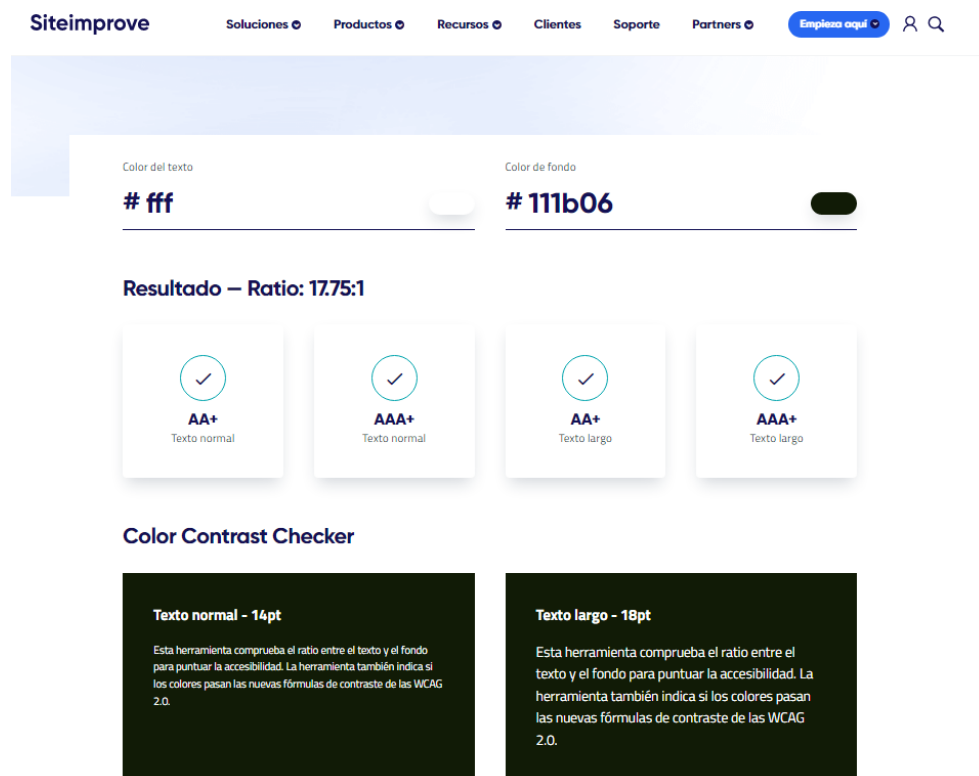


Ilustración 24

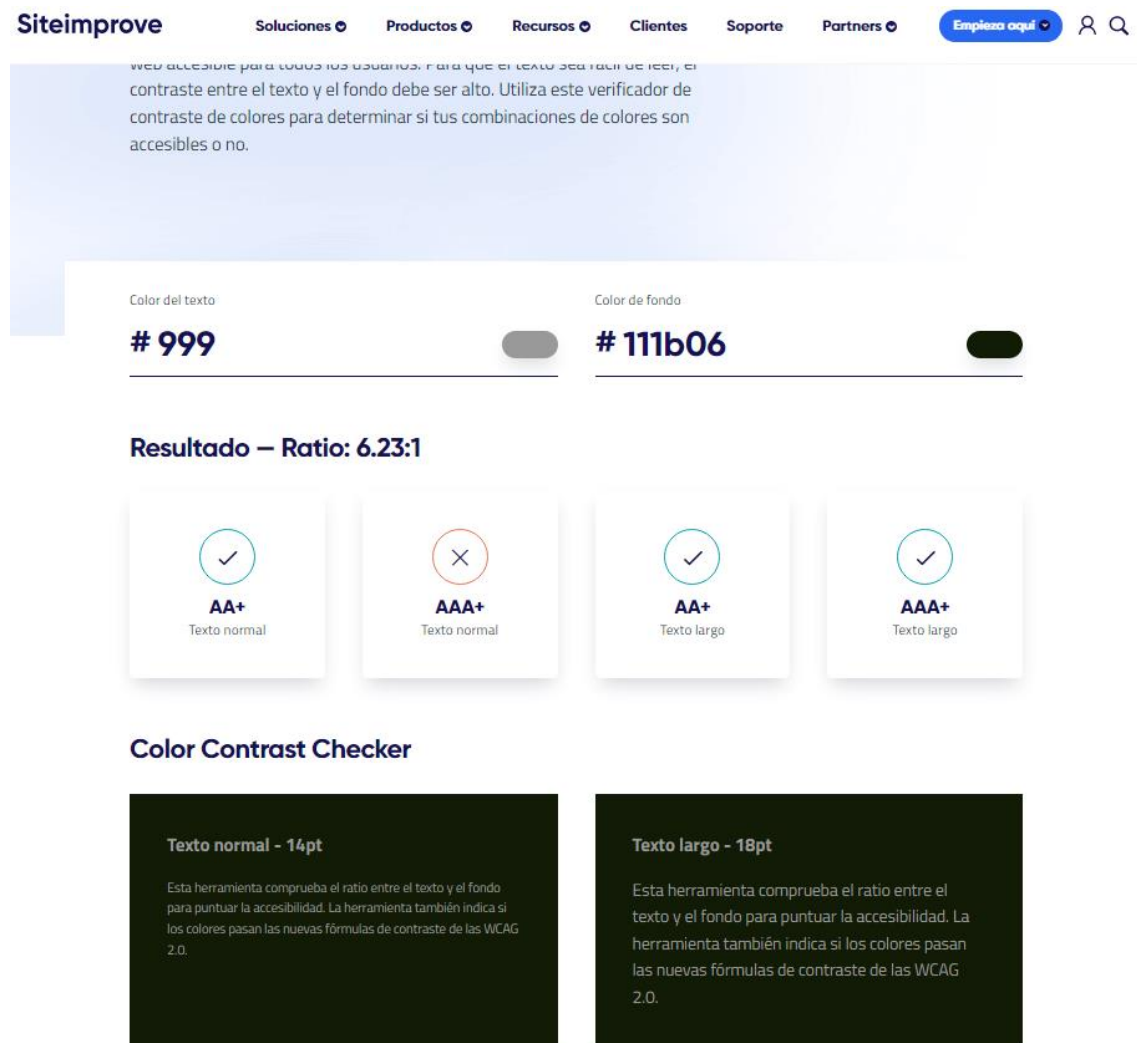


Ilustración 25

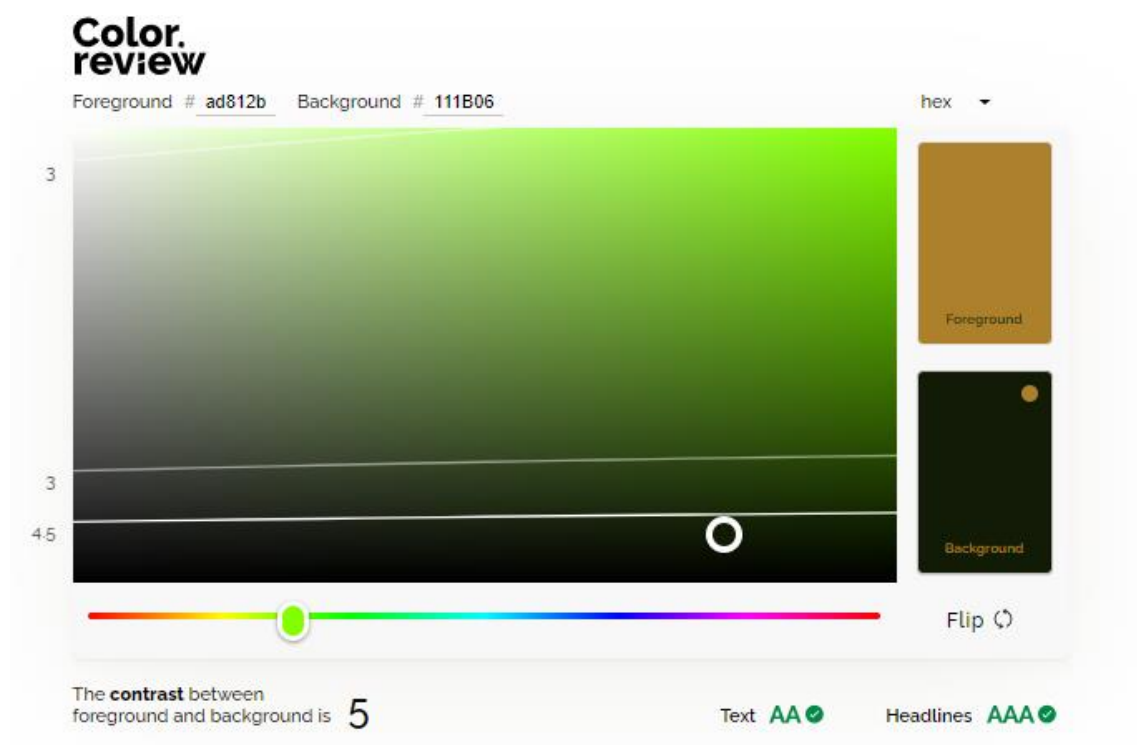


Ilustración 26

## Foco

El foco determina en qué parte de la página tienen lugar los eventos del teclado en un momento determinado.

En toda web se debe permitir el acceso a todas las funcionalidades de la página por medio de un teclado, a menos que sea algo que no se pueda hacer con teclado, como dibujar a mano. Esto se puede controlar mediante el uso de la tecla *tab* del teclado. Se debe de asegurar que el orden de tabulación sea lógico. Elementos como campos de texto, botones y listas se insertan automáticamente en el orden de tabulación. Sin embargo, no todos los elementos pueden enfocarse como párrafos o divs, ya que no es algo con lo que puedas interactuar.

**Pongámonos en contacto**

No dude en contactar con nosotros, estamos para ayudarle.

Formulario de contacto

\* Email

Email

Formato Email: pepito@gmail.com

Ilustración 27

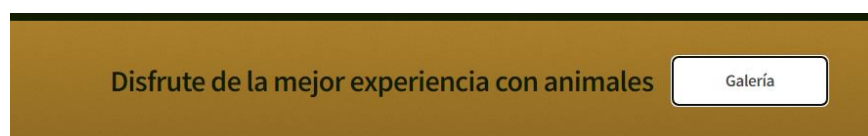


Ilustración 28

## USO DE JAVASCRIPT PARA AÑADIR ACCESIBILIDAD

Hasta la llegada de WCAG 2.0 el uso de JavaScript se encontraba desaconsejado, siempre y cuando no se aportara una alternativa que no hiciera uso de él.

Con la aparición de la versión 2.0 de las pautas de accesibilidad web apareció el concepto de tecnología compatible con la accesibilidad, por lo que JavaScript lo cumpliría y sigue cumpliéndolo, siendo estándar de facto en el desarrollo web.

HTML:

```
<p><output id="output1" aria-live="polite">&nbsp;</output></p>
```

```
<p><a id="anchor1" href="#">Salúdame</a></p>
```

Js:

```
window.addEventListener("load",function(e){  
    var anchor1 = document.querySelector("#anchor1");  
    var output1 = document.querySelector("#output1");  
  
    function hSaludo(e){  
        output1.innerHTML = "!Hola!";  
    }  
  
    anchor1.addEventListener("mouseover",hSaludo);  
    anchor1.addEventListener("focus",hSaludo);  
});
```

En el ejemplo anterior podemos observar que existe un problema de accesibilidad ya que, aunque el programa deba realizar un saludo al situarse sobre el enlace, al ser tabulado no se realizará el saludo, por lo que, se deben evitar eventos dependientes de un dispositivo concreto, como ratón o teclado exclusivamente.

El uso de el atributo **aria-live** de WAI-ARIA, como en el ejemplo, permite que tecnologías de acceso a la accesibilidad puedan percibir los cambios que se producen en los elementos y lo anunciarán rápidamente si se establece el valor **assertive** o tras una pausa si se asigna **polite**.

## Navegación mediante teclado

Consiste en ir tabulando por los elementos de la página interactivos mediante teclado.

Para ello el orden de navegación se debe de tener en cuenta, respetando el orden lógico mediante la estructuración adecuada de nuestras páginas mediante secciones, evitar el uso de estilos que puedan modificar el orden lógico de navegación (flexbox -> order) y evitar el uso de tabindex (atributo de HTML).

## Saltar al contenido principal

Podemos proporcionar enlaces para permitir al usuario saltar las secciones de navegación para acceder a la sección principal.

```
<a id="skip" href="#main">Saltar navegación</a>
```

```
<header>
```

```
<!-- Contenido -->
```

```
</header>
```

```
<!-- Contenido -->
```

```
<main id="main">
```

```
<!-- Contenido -->
```

```
</main>
```

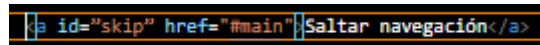
A screenshot of a code editor showing the HTML code for a skip link. The code is: `<a id="skip" href="#main">Saltar navegación</a>`. The text is highlighted in blue, and the code is displayed on a dark background with a light border.

Ilustración 29

Y ocultarlos hasta que sean necesarios, mediante atributos diferentes a display o visibility ya que no se encontrarían, por lo tanto, disponibles para los lectores de pantalla.

## Componentes de Usuario

Para asegurar una navegación mediante teclado lógica, rápida y sin trampas, se pueden utilizar componentes de usuario (campo de texto, botón, menú). Se puede comprobar a través de la tecla TAB que se obtiene el foco dentro del control y que se puede seguir navegando, y que se tiene en cuenta la navegación a través de otros tipos de dispositivos.

## Cambios de contexto

Se debe evitar desorientar al usuario con cambios de contexto una vez que el mismo interactúa con elementos de la web, es decir, que se advierta explícitamente sobre los cambios que se producirán una vez el usuario interactúe con la página, ya sea cuando obtengan el foco los elementos o cuando su valor cambie.

## Componentes de interfaz de usuario

### Menú de navegación

Deben de ser reconocidos por una tecnología de asistencia a la accesibilidad, mediante el uso de el atributo **aria-label**, identificación de los tipos de los elementos mediante **role**, especificar que elementos de menú contienen menús desplegables mediante **aria-haspopup**, especificar el estado de despliegue de los menús, mediante el atributo **aria-expanded**, e identificar el elemento de menú que apunte a la página actual en la que se encuentra el usuario, mediante **aria-current**. Todo ello gracias a WAI-ARIA que se introdujo en HTML para dicho fin, ya que es algo que HTML por sí sólo no puede llegar a lograr.

### Patrones de diseño

El W3C tiene publicado, en el documento WAI-ARIA Authoring Practices 1.2, una serie de patrones de diseño para cada uno de los componentes de usuario que un desarrollador pueda necesitar implementar.

Nos ofrecen ejemplos para cada tipo de elementos y se nos indica los atributos WAI-ARIA que se deben implementar y el comportamiento mediante JavaScript para navegar mediante el teclado.

Se introducirán los atributos WAI-ARIA mediante JavaScript en tiempo de ejecución y cambiaremos sus valores según sea necesario.

### Presentación del menú

Se debe de emplear hoja de estilos en cascada, con secciones para teléfonos móviles, otra para tabletas y otra para dispositivos con una resolución mayor.

Aprovechando los atributos de WAI-ARIA, que vamos a emplear en el menú, pintaremos una flecha hacia abajo en los elementos de menú superior que se puedan desplegar, y hacia la derecha, para los elementos de niveles inferiores que también contengan un submenú:

```

nav > ul > li > a[aria-haspopup]::after {

    content:url("\25BC");

}

nav ul li ul a[aria-haspopup]::after {

    content:url("\25B6");

}

```



Ilustración 30

Aplicaremos a nuestros elementos de menú una visualización de bloque para que ocupen la totalidad del ancho de nuestros dispositivos móviles:

```

nav ul li a {

    display:block;

}

```

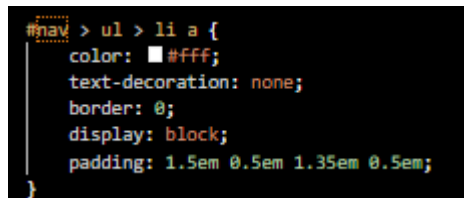


Ilustración 31

Aplicaremos estilos diferenciados para informar al usuario: cuando se encuentra sobre un elemento de menú (en dispositivos táctiles, dicho evento no suele reconocerse de una forma adecuada); cuando está pulsando sobre él; y para indicarle, cuál es el elemento de menú que apunta a la página actual en la que se encuentra:

```

nav ul li a:hover, nav ul li a:focus {

/* Propiedades*/

}

nav ul li a:active {

/* Propiedades*/

}

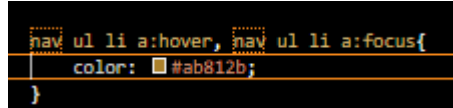
```



```
nav ul li a[aria-current] {

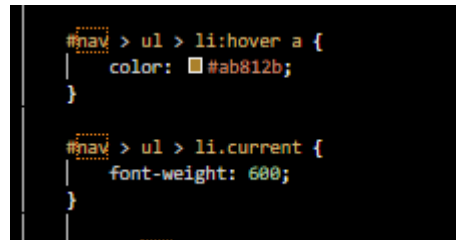
/* Propiedades*/

}
```



```
nav ul li a:hover, nav ul li a:focus {
  color: #ab812b;
}
```

Ilustración 32



```
#nav > ul > li:hover a {
  color: #ab812b;
}

#nav > ul > li.current {
  font-weight: 600;
}
```

Ilustración 33

De nuevo, emplearemos un atributo de WAI-ARIA, `aria-expanded`, para ocultar o mostrar los menús, en base al valor de dicho atributo:

```
nav ul li a[aria-expanded="false"] ~ ul {

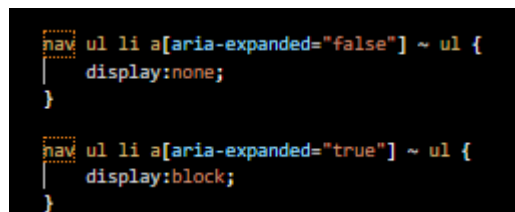
  display:none;

}

nav ul li a[aria-expanded="true"] ~ ul {

  display:block;

}
```



```
nav ul li a[aria-expanded="false"] ~ ul {
  display:none;
}

nav ul li a[aria-expanded="true"] ~ ul {
  display:block;
}
```

Ilustración 34

En modo tableta, definiremos el elemento `<ul>` raíz del menú, como un contenedor de tipo Flexbox. Y especificaremos, para cada elemento flexible `<li>`, una anchura mínima de 100 píxeles, a partir de la cual, podrá expandirse hasta ocupar horizontalmente el espacio que se encuentre disponible en su contenedor:

**En este caso utilizamos el diseño responsive de más a menos, por lo que utilizaremos `max-width` en vez de `min-width` para establecer el diseño para dispositivos con resoluciones más pequeñas.**

```
/* ***** Reglas para tabletas ***** */
```

```
@media only screen and (min-width: 768px) {
```

```
nav > ul {
```

```
    display:flex;
```

```
}
```

```
nav > ul > li {
```

```
    width:100px;
```

```
    flex-grow:1;
```

```
    position:relative;
```

```
}
```

Por último, aplicaremos diferentes estilos para la visualización de los menús y submenús desplegables:

```
nav ul li ul {
```

```
    position:absolute;
```

```
    z-index:1;
```

```
    width:100%;
```

```
}
```

```
nav ul li ul li {
```

```
    position:relative;
```

```
}
```

```
nav ul li ul li ul {
```

```
    top:0px;
```

```
    left:100%;
```

```
}
```

```
nav ul li:last-child ul li ul {
```

```
    left:-100%;
```

```
}}
```

```

}

@media screen and (max-width: 980px) {
  .container {
    width: 95%;
  }
}

@media screen and (max-width: 840px) {
  .container {
    width: 95%;
  }
}

@media screen and (max-width: 736px) {
  .container {
    width: 90%;
  }
}

@media screen and (max-width: 480px) {
  .container {
    width: 100%;
  }
}

```

Ilustración 35

## Navegación mediante el teclado

Para facilitar la comprensión de los comportamientos es conveniente confeccionar un árbol invertido, con el marcado del menú.

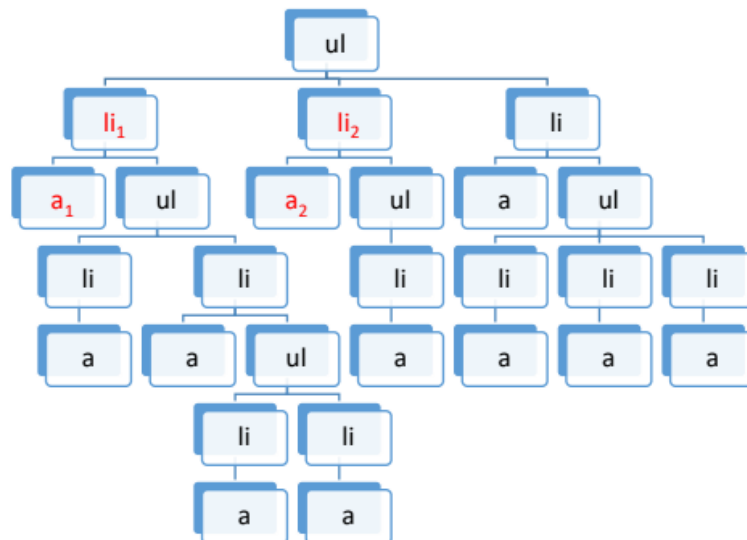


Ilustración 36

## Implementación mediante JavaScript

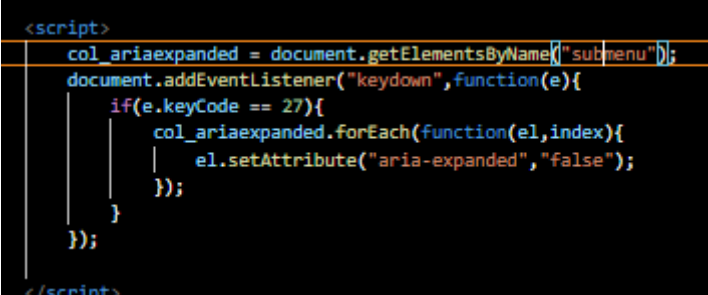
Para realizar las asignaciones WAI-ARIA se establecen mediante JavaScript, mediante el acceso al elemento a través de una variable a la que le estableceremos el atributo mediante el método `setAttribute("atributo aria", "valor")`.

También se puede establecer el atributo `tabIndex` de la misma forma.

El criterio 1.4.13. de las pautas de accesibilidad, Content on Hover or Focus, nos indica que, si el usuario despliega por error un menú, deberemos proporcionarle un mecanismo para que pueda replegarlo, sin necesidad de tener que desplazarse fuera de él. Por ejemplo, pulsando la tecla ESC:

//WCAG 1.4.13

```
document.addEventListener("keydown",function(e){  
    if(e.keyCode == 27){  
        col_ariaexpanded.forEach(function(el,index){  
            el.setAttribute("aria-expanded","false");  
        });  
    }  
});
```



```
<script>  
    col_ariaexpanded = document.getElementsByName("submenu");  
    document.addEventListener("keydown",function(e){  
        if(e.keyCode == 27){  
            col_ariaexpanded.forEach(function(el,index){  
                el.setAttribute("aria-expanded","false");  
            });  
        }  
    });  
</script>
```

Ilustración 37

Validación de email del formulario de contacto:

```
<script>
    function ValidateEmail(inputEmail)
    {
        var mailformat = /^w+([.-]?w+)*@w+([.-]?w+)*(.w{2,3})+$/;

        if(!inputEmail.value.match(mailformat))
        {
            alert("Ha introducido un email inválido");
            document.form1.email.focus();
            return false;
        }
    }
</script>
```

Ilustración 38

## Conclusión

Primera regla del documento Using ARIA:

“If you can use a native HTML element or attribute with the semantics and behaviour you require already built in, instead of re-purposing an element and adding an ARIA role, state or property to make it accessible, then do so.”

Esto es, si HTML dispone de un marcado adecuado para identificar un elemento o componente no emplees atributos de WAI-ARIA. Estos atributos susciben un compromiso de comportamiento asociado, que a menudo no se implementa o se implementa de forma incorrecta.

## BIBLIOGRAFÍA

## 1. PÁGINAS WEB

Autor	Fecha	Título	Fuente	
<b>Affde</b>	04/03/2021	Cómo mejorar la legibilidad del contenido de su blog y sitio web (¡y por qué es imprescindible!)	<b>Sitio.</b>  www.affde.com/	Consultado el 08 de Marzo de 2022  <a href="https://www.affde.com/es/how-to-improve-readability-for-your-blog-website-content-and-why-its-a-must-2.html">https://www.affde.com/es/how-to-improve-readability-for-your-blog-website-content-and-why-its-a-must-2.html</a>
<b>Anna Monus</b>	18/07/2021	Flexbox frente a CSS Grid: ¿Cuál debes usar y cuándo?	webdesign.tutsplus.com	Consultado el 08 de Marzo de 2022  <a href="https://webdesign.tutsplus.com/es/articles/flexbox-vs-css-grid-which-should-you-use--cms-30184">https://webdesign.tutsplus.com/es/articles/flexbox-vs-css-grid-which-should-you-use--cms-30184</a>
<b>Programación Tecnología</b>	29/04/2015	Diseño adaptativo vs. responsive: ¿son lo mismo?	deustoformacion.com	Consultado el 08 de Marzo de 2022  <a href="https://www.deustoformacion.com/blog/programacion-tic/disenio-adaptativo-vs-responsive-son-lo-mismo">https://www.deustoformacion.com/blog/programacion-tic/disenio-adaptativo-vs-responsive-son-lo-mismo</a>
<b>Luiggi Santa María</b>	13/03/2016	5 formas de mejorar el contraste en tus páginas web	staffdigital.pe	Consultado el 08 de Marzo de 2022  <a href="https://www.staffdigital.pe/blog/5-formas-mejorar-contraste-paginas-web/">https://www.staffdigital.pe/blog/5-formas-mejorar-contraste-paginas-web/</a>
<b>Meggin Kearney Dave Gash Rob Dodson</b>	No presenta	Introducción al foco   Web   Google Developers	developers.google.com	Consultado el 08 de Marzo de 2022  <a href="https://developers.google.com/web/fundamentals/accessibility/focus?hl=es">https://developers.google.com/web/fundamentals/accessibility/focus?hl=es</a>