Conceptos de Accesibilidad



ÍNDICE

[Desarrollando para Accesibilidad Web 3](#_Toc94724773)

[Asociar una etiqueta con cada control de formulario 3](#_Toc94724774)

[Incluir texto alternativo para las imágenes 3](#_Toc94724775)

[Identificar el idioma de la página y los cambios de idioma 3](#_Toc94724776)

[Utilice el marcado para transmitir significado y estructura 3](#_Toc94724777)

[Ayudar a los usuarios a evitar y corregir errores 4](#_Toc94724778)

[Refleja el orden de lectura en el orden del código. 5](#_Toc94724779)

[Escribir código que se adapte a la tecnología del usuario 6](#_Toc94724780)

[Proporcionar significado para elementos interactivos no estándar. 6](#_Toc94724781)

[Asegúrese de que todos los elementos interactivos sean accesibles desde el teclado 7](#_Toc94724782)

[Evite CAPTCHA siempre que sea posible 8](#_Toc94724783)

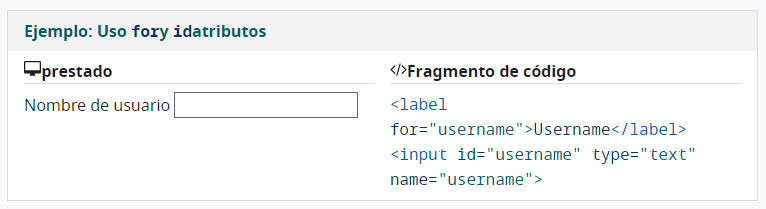
# Desarrollando para Accesibilidad Web

En este documento se presentarán ciertas consideraciones para tener en cuenta a la hora de desarrollar una web más accesible.

Estos consejos son buenas prácticas para ayudarlo a cumplir con los requisitos de las Pautas de accesibilidad al contenido web (WCAG).

## Asociar una etiqueta con cada control de formulario

Use un foratributo en el <label>elemento vinculado al idatributo del elemento de formulario, o use atributos WAI-ARIA . En situaciones específicas, puede ser aceptable ocultar <label>elementos visualmente, pero en la mayoría de los casos se necesitan etiquetas para ayudar a todos los lectores a comprender la entrada requerida.



## Incluir texto alternativo para las imágenes

Use texto alternativo vacío, alt=""para las imágenes, o inclúyelos en el CSS en su lugar. Las alternativas de texto suelen ser proporcionadas por los responsables del contenido escrito.

## Identificar el idioma de la página y los cambios de idioma

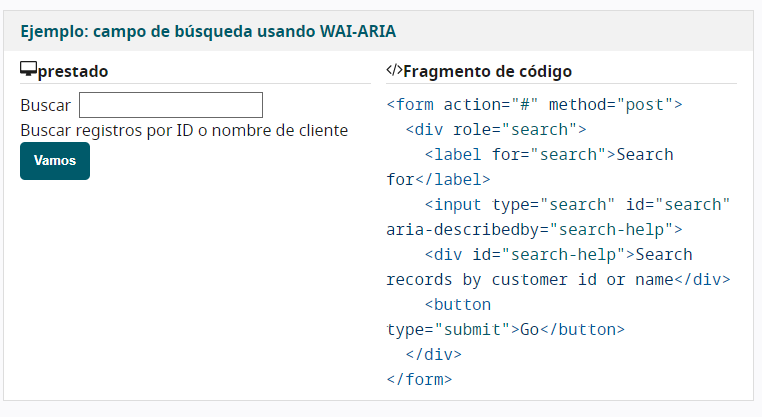
Indicar el idioma principal de cada página mediante el langatributo en la htmletiqueta, por ejemplo <html lang="en">.

## Utilice el marcado para transmitir significado y estructura

Use el marcado apropiado para encabezados, listas, tablas, etc. HTML5 proporciona elementos adicionales, como <nav>y <aside>, para estructurar mejor su contenido.

 Utilizar role="search"para identificar la funcionalidad de búsqueda.





## Ayudar a los usuarios a evitar y corregir errores

Proporcione instrucciones claras, mensajes de error y notificaciones para ayudar a los usuarios a completar formularios en su sitio. Cuando ocurre un error:

* Ayudar a los usuarios a encontrar dónde está el problema
* Proporcionar explicaciones específicas y comprensibles.
* Sugerir correcciones



## Refleja el orden de lectura en el orden del código.

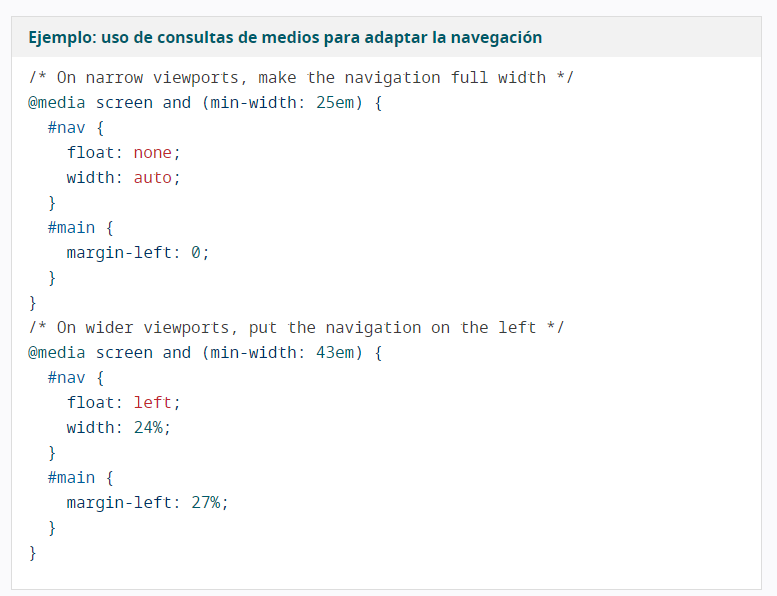
El orden en el que aparezcan los elementos en el código deben coincidir con el orden cronológico de la información que aparece.

Una forma en la que podemos verificarlo es eliminar el estilo CSS y revisar que el orden del contenido tenga sentido.



## Escribir código que se adapte a la tecnología del usuario

Utilice un diseño receptivo para adaptar la pantalla a diferentes estados de zoom y tamaños de ventana gráfica. Cuando el tamaño de fuente aumenta al menos en un 200 %, evite el desplazamiento horizontal y evite cualquier recorte de contenido. Utilice la mejora progresiva para ayudar a garantizar que la funcionalidad y el contenido principal estén disponibles independientemente de la tecnología que se utilice.



## Proporcionar significado para elementos interactivos no estándar.

Utilice WAI-ARIA para proporcionar información sobre la función y el estado de los widgets personalizados, como acordeones y botones personalizados.

Por ejemplo, role="navigation"y aria-expanded="true". Se requiere código adicional para implementar el comportamiento de dichos widgets, como expandir y contraer contenido o cómo responde el widget a los eventos del teclado.



## Asegúrese de que todos los elementos interactivos sean accesibles desde el teclado

Piense en el acceso al teclado, especialmente cuando desarrolle elementos interactivos, como menús, información sobre el mouseover, acordeones plegables o reproductores multimedia.

Úselo tabindex="0"para agregar un elemento que normalmente no recibe el foco, como <div>o <span>, en el orden de navegación cuando se usa para la interacción. Use secuencias de comandos para capturar y responder a los eventos del teclado.



## Evite CAPTCHA siempre que sea posible

Existen otros medios para verificar que la entrada del usuario fue generada por un ser humano que son más fáciles de usar, como la detección automática o las interacciones de la interfaz.

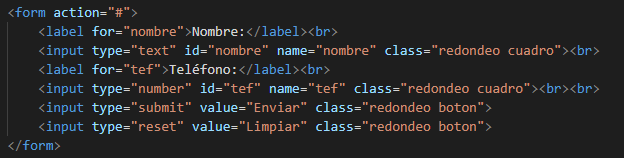
Si realmente necesita incluir CAPTCHA, debe seguir las siguientes pautas para que sea lo más accesible posible:

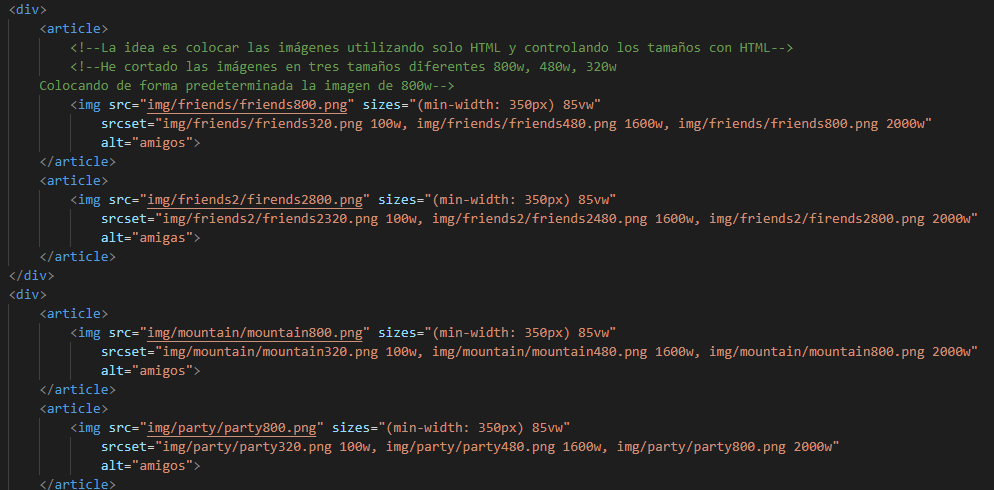
* Proporcionando más de dos formas de resolver los CAPTCHA
* Proporcionar acceso a un representante humano que puede omitir CAPTCHA
* No requiere CAPTCHA para usuarios autorizados.

# IMPLEMENTACIÓN EN ALGUNAS PÁGINAS QUE YA HEMOS CREADO

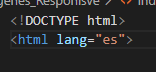
El primer ejemplo lo desempeñamos en un formulario que creamos específicamente para probar la accesibilidad.

Vemos como está asociado a etiquetas <label>.

Para mejorar la accesibilidad en las imágenes lo que tenemos que hacer es incluir un texto alternativo que especifique que hay en ellas.

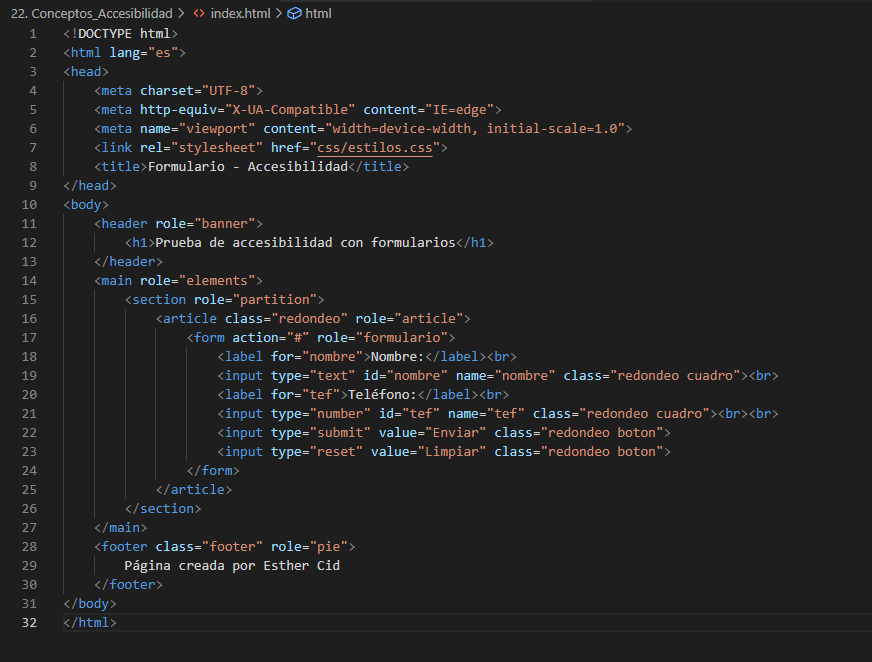


Debemos especificar el idioma en el que se encuentra la página que vamos a desarrollar.



Debemos utilizar las etiquetas semánticas que nos proporciona HTML5 para que quede clara la parte de la página en la que se encuentra cada elemento.

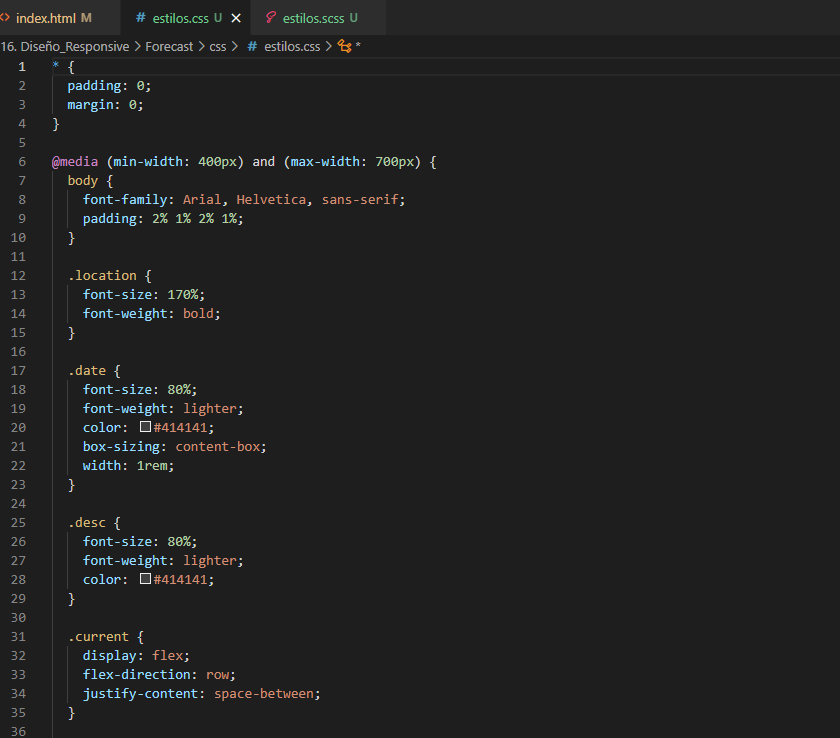
Podemos utilizar el atributo “role” para ayudar a la compresión de la utilización de cada etiqueta.



Los formularios pueden tener un patrón que limite el envío de datos que no se ajusten a él. Otra de las cosas que tenemos que hacer es ayudar al usuario señalándole donde se encuentra el error.



El código debe adaptar el contenido en función de la pantalla en la que se vea, de esta forma lo podemos hacer más accesible en caso de que se acceda mediante un dispositivo con la pantalla más pequeña como una Tablet o un teléfono móvil.





También podemos utilizar una serie de etiquetas que son capaces de, utilizando una serie de widgets personalizados, indicar si ese elemento transmite el estado actual del componente o si, por ejemplo, el botón funciona como una casilla de verificación.



# PRUEBAS ACCESIBILIDAD CON WAVE

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Escala de tiempo

Descripción generada automáticamente con confianza media

# HERRAMIENTAS DE ACCESIBILIDAD WEB

## HERRAMIENTAS ONLINE

### TAW

### HERA

### WEB ACCESIBILITY CHECKER

### CYNHTIA SAYS

### ACCESS COLORS

### WAVE

### TENON.IO

## PLUG INS

### ACCESIBILITY EVALUATION TOOLS (FIREFOX)

### ACCESIBILITY DEVELOPER TOOLS (CHROME)

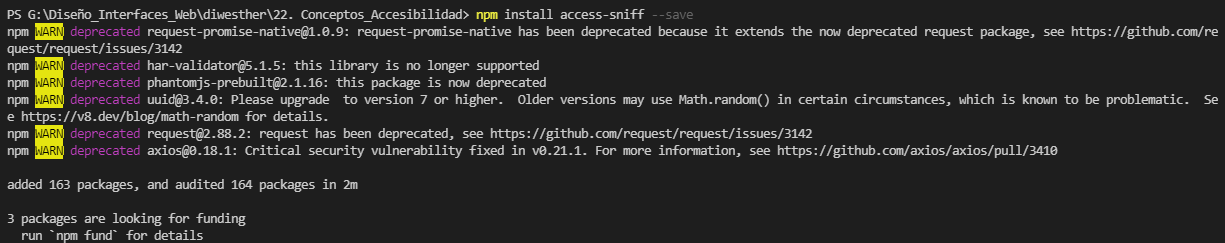
### WEB ACCESIBILITY TOOLBAG (OPERA E INTERNET EXPLORER)

## LOCALES

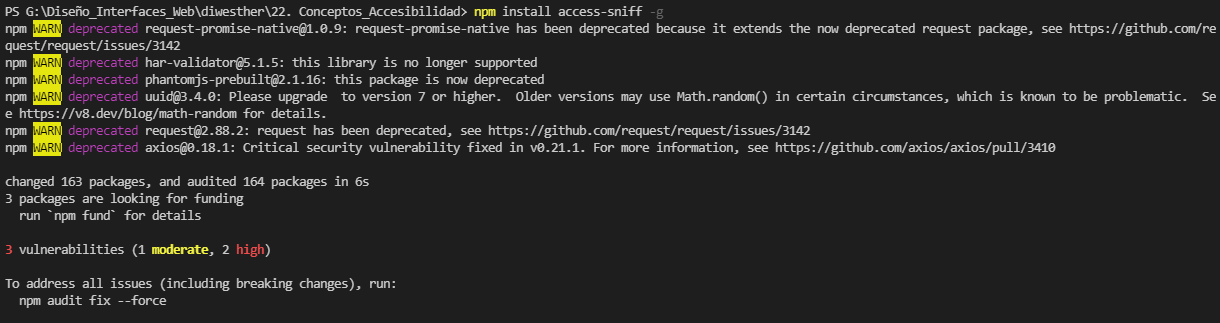
### ACCESS SNIFF

Se instala mediante un paquete de nodejs. “*npm install access-sniff –save*”.

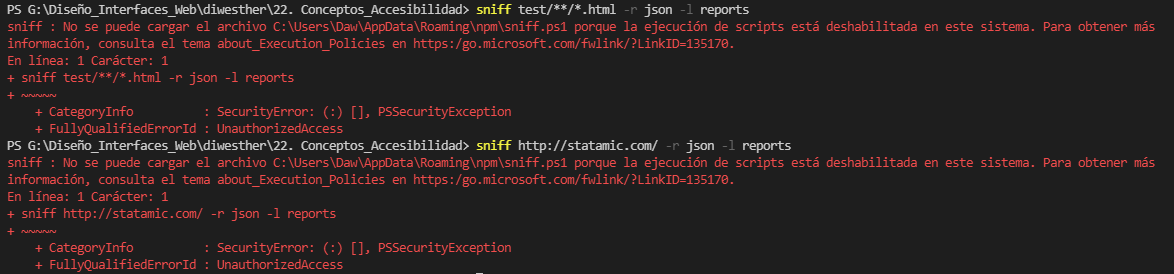
Lo instalamos dentro del documento del que vayamos a comprobar su accesibilidad.



Después instalamos el acceso con el siguiente paquete: “npm install access-sniff -g”.

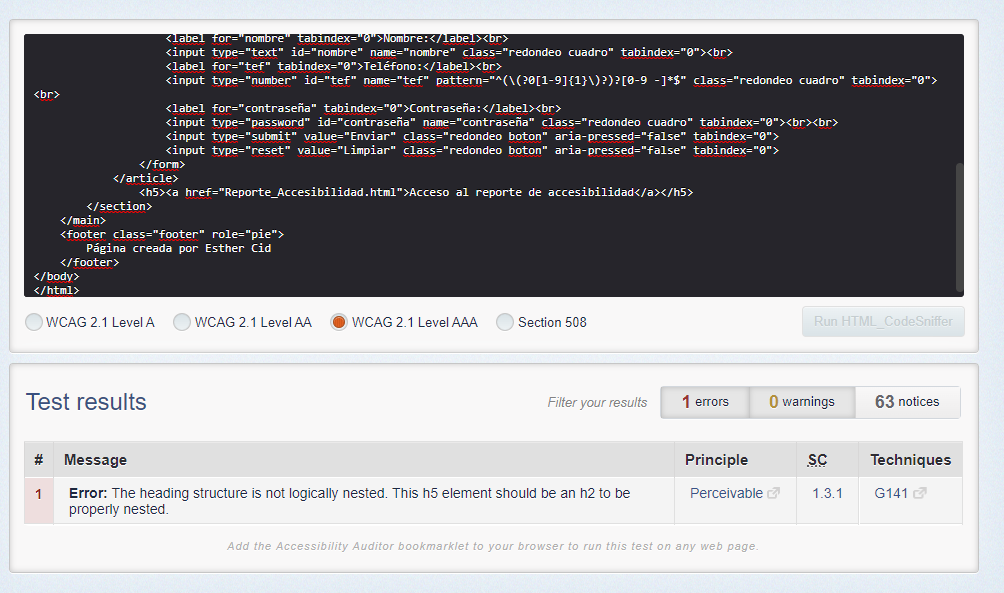


Despues de instalarlo no podemos acceder a la aplicaciónporque da un error de acceso a uno de los scripts que necesita para poder ejecutarse.



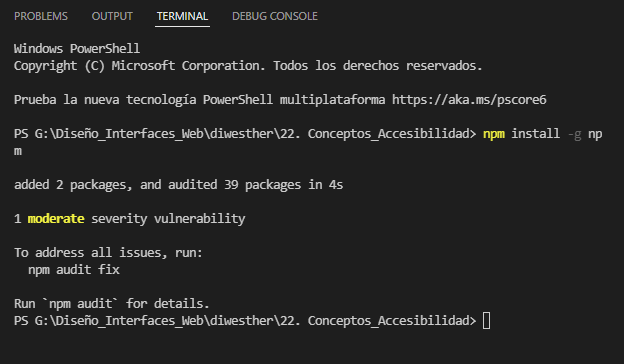
### HTML CODE SNIFFER

Podemos acceder a él a través de su GitHub, en el mismo GitHub nos permite usar la aplicación de forma OnLine, sin necesidad de descargarlo.

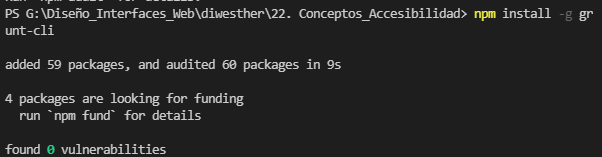


Podemos ver los errores que nos devuelve, nos dice que el “<h5>” que tenemos en el código, estructuralmente debería ser un “<h2>”.

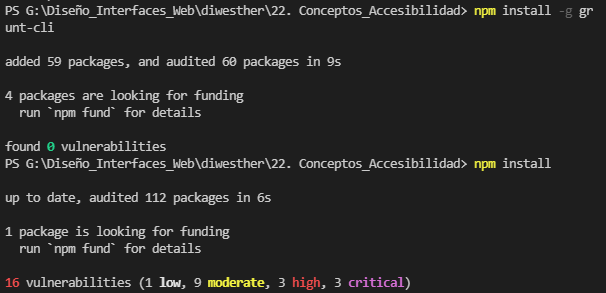
Lo primero que nos indica su GitHub es que debemos actualizar node.js.



Instalamos grunt cli.



Instalamos las dependencias de node.js que necesita grunt.



### GRUNT ACCESIBILITY

### GRUNT HTML VALIDATOR