ELEMENTOS DE NUESTRA APLICACIÓN

Tomé Maseda Dorado

Adrián Lage Gil

Todos los elementos serán referenciados por su id, en caso de no tener id, se especificará otra forma de identificarlos

Imágenes y tablas

instagram

No contiene un texto alternativo.

Corrección: Se le proporciona el texto alternativo “Logo Instagram”.

facebook

No contiene un texto alternativo.

Corrección: Se le proporciona el texto alternativo “Logo Facebook”.

twitter

No contiene un texto alternativo.

Corrección: Se le proporciona el texto alternativo “Logo Twitter”.

imagei

Este es un elemento dinámico, la imagen con el número i, corresponderá con el pokémon número i (Ej.: *image1* corresponde al pokémon 1).

No contiene un texto alternativo.

Corrección: Le pondremos un texto alternativo distinto en función del valor de i, es decir, para *imagei* se le proporcionará el texto “Logo Pokémon i”.

Contenidos repetitivos

El único bloque de contenido que podríamos saltar es la lista completa de pokémons, no disponemos de ningún mecanismo de navegación para esto, por tanto, introducimos un botón para ir al final de la lista (*end*) y otro para volver al inicio (*top*).

Color

Para el color nos ajustamos a una paleta de colores generada aleatoriamente en la web uicoach.io/.

Se cumple con los estándares de accesibilidad WCAG 2 y la Sección 508 (existe contraste entre el color de fondo y el primer plano y no se usa SOLO colores para marcar acciones, campos obligatorios, errores...).

Formularios

pokemoname

El formulario solo contiene un elemento (input), sin embargo, este no tiene un label correspondiente.

Corrección: Introducimos el párrafo que pone "Search by pokémon name" dentro de un label para seguir una estructura sintáctica correcta, a pesar de que en cuanto a accesibilidad el formulario ya cumplía con los estándares.

Fields

Todos los elementos del formulario están relacionados (no hace falta agruparlos por fieldsets), sin embargo, cada elemento no tiene su label correspondiente.

Corrección: Introducimos un label por cada elemento del formulario. Además, aprovechamos los placeholders de cada elemento, que antes se usaban para especificar el dato a introducir (ahora está función la realiza el label), para proporcionar instrucciones sobre la introducción de datos.

Enlaces y botones

(Los enlaces para saltar al final de la lista o volver al principio ya han sido explicados previamente).

stats ability y *type*

Los botones poseen una descripción clara del objetivo, obtener las estadísticas, habilidades o tipos de la lista de pokémons.

send

El botón *send* posee una descripción clara del objetivo, enviar los datos del formulario rellenado previamente.

instagram, facebook y *twitter*

El objetivo de los enlaces se describe mediante una imagen con el logo de la red social a la que se quiere acceder.

info

Los botones *info* son botones dinámicos que se generan al mostrar la lista completa de pokémons, el texto que ofrecen es "more info", el objetivo de los botones se puede inferir del contexto, pero no queda claro del todo.

Problemas: Estamos usando el mismo texto para varios botones en la misma página, estamos etiquetando los botones con textos genéricos (pulsa aquí, leer más...), además, todos los botones

tienen el id info (IDs repetidos), tampoco cumple el estándar W3C, a pesar de que al ser botones dinámicos no da problemas al pasar el html por el validador.

Corrección: Para cada pokémon se mostrará un botón con texto “Info of <nombre_pokemon>”, estamos dando una descripción más clara del objetivo y no estamos repitiendo nombres para los botones (el id de cada botón será el id del pokémon precedido de la letra b correspondiente también).

Teclado y focus

La lista de filtros (*dropdown*) no es accesible mediante teclado.

Corrección: Añadimos tabindex=”0” para especificar que el elemento pueda ser leído por el tabulador.

Hojas de estilos

No incluimos imágenes con contenido como *background-images.*

No incluimos contenido con significado en CSS (::before, ::after).

No incluimos botones con posicionamiento absoluto.

Títulos y cabeceras

La página web incluye un título que describe el propósito de esta.

Salto de nivel en la etiqueta:

```
<h3>The Pokémon Company</h3>
```

Solución: Cambio de <h3> por <h1> y adecuación del formato y estilo deseado en el fichero .css.

Mal uso de la etiqueta de cabecera <h6>:

```
<h6>By Tomé Maseda & Adrián Lage</h6>
```

Solución: Cambio de la etiqueta <h6> por <p> y adecuación del formato y estilo en el fichero .css.

Elementos multimedia

No incluimos elementos multimedia (sonidos videos…).

Otros

La página está en inglés y tiene como idioma general el español.

Corrección: Cambiamos el atributo lang de la etiqueta html de “es” a “en”.

No incluimos ningún contenido que produzcan parpadeos, destellos, autoscroll…

No incluimos CAPTCHA porque no tenemos un sistema de autenticación.

El código HTML está validado (estándar W3C).

WAI-ARIA

Se asignan los siguientes roles a nuestros elementos:

- A la lista de pokémons (*pokemonslist*) se le asigna el rol *list.*
- En consecuencia, a cada pokémon se le asigna el rol *listitem*.
- Al formulario de búsqueda (*pokemoname*) se le asigna el rol *search*.
- Para los enlaces de navegación (*top* y *end*) cambiamos el div en el que estaban por un nav, para darles el rol *navigation*.
- Para controlar los errores añadimos un elemento *dialog* (con rol *dialog*) y el div que contiene el texto interior tendrá el rol *document*, este rol se utiliza para elementos que tengan modo lectura, con el fin de que se pongan en ese modo y utilicen el elemento con rol *document* para leer ese texto.
- Le damos el rol *region* a los filtros y especificamos que está etiquetada por el div de debajo (*aria-labelledby=”filtersheading”)* que es el que contiene el texto “Filters”. Le ponemos el id *filtersheading* al div de debajo.
- Marcamos la sección *show* como live-region ya que ahí es donde va a aparecer la lista cuando hagamos peticiones, por tanto, el contenido de este elemento es dinámico y estará variando continuamente. La marcamos como *polite* ya que no queremos que nuestra región sea asertiva, el lector de pantalla terminará de leer lo que esté leyendo y después leerá el contenido de la sección.
- Marcamos también el div (*infoi*, siendo i el número del pokémon al que le corresponde la información de ese div) como live-region, el contenido de este elemento es dinámico, variará cuando pulsamos los botones *info* y *close*. Lo marcamos de nuevo como una región *polite*.
- Mientras modificamos las dos live-regions siempre marcamos el atributo *aria-busy* como true y al finalizar le devolvemos su valor por defecto (false).