08/06 - 18:00

Comienzo del proyecto por la página principal (index.html) utilizando como base la página principal de mi página personal desarrollada durante el curso.

Basándome en el .html y .css de la página https://negativefeedback.co.uk/ creo mi menú de navegación adaptando parte del código que me interesa.

09/06 - 20:00

Comienzo a desarrollar la presentación básica de la página como los colores y tipografías a utilizar.

Después de muchas pruebas con linear-gradients como color de fondo y combinaciones de colores compatibles sugeridas por https://htmlcolorcodes.com/es/ opto por utilizar granate (#ab2a3e) como color principal combinado con blanco.

Para crear un logotipo utilizo un <div> con un ancho determinado en el que va contenido el h1 de la página con la información de aquí: https://www.w3schools.com/tags/tag\_div.asp

09/06 - 23:00

Comienzo a darle estructura a toda la página creando el about.html con la información correspondiente a la página y a su autor, con la imagen mía que tenía en OneDrive utilizando el link público para integrarla y centrándola con la técnica aquí explicada: https://www.w3schools.com/howto/howto\_css\_image\_center.asp

También integro un máximo de px de ancho para la página en el .css principal.

10/06 - 01:00

Creación de juego.html, juego.css y juego.js utilizando código desarrollado durante el curso con un juego simple de la serpiente para el uso del API canvas.

Dándole un borde al canvas y un tamaño siendo el mayor posible sin que llegue a quedar parte de la pantalla oculta de modo que al mover la serpiente con las flechas no interfiera en el scroll de la página.

Por otra parte, también desarrollo una clase 'selected' en el css por el cual cada página html tendrá asignada dicha clase al elemento del menú correspondiente con dicha página, quedando así coloreado en granate para indicar en cuál se está.

10/06 - 19:00

Modifico el código ECMAScript de la serpiente para que se mueva con las teclas WASD no con las flechas para no interferir en el scroll.

Comienzo con la creación de un xml conteniendo las noticias utilizando como base el desarrollado durante el curso y modificándolo con noticias relacionadas a la fotografía, modificando el dtd para cambiar algún tipo de datos. Una vez acabado el xml creo un svg a partir de él. Después genero un xsd utilizando Visual Studio. Modifico el xsd manualmente cambiando el tipo de año de string a integer. Genero un svg a partir de este xsd.

19/06 - 15:30

A partir del xml bien formado comienzo con las transformaciones xslt. Utilizando como base el desarrollado durante el curso adapto el xsl para que muestre la información ahora manejada. Consulto esta página (https://stackoverflow.com/questions/2563021/setting-xslvalue-of-into-an-href-attribute-and-the-text-field-of-a-link-in-an-x) para saber cómo escribir un valor obtenido mediante xsl del xml dentro del atributo href de la etiqueta <a> y así tener enlaces a las noticias originales.

Tengo problemas con el xsl puesto que se puede visualizar en todos los navegadores (Edge, Firefox, IE) pero en Chrome aparece vacía la página.

Llego a la conclusión que en Chrome no funciona porque están en local ambos archivos, ejecutándolo desde Ritchie funciona.

20/06 - 23:00

Para la parte de la geolocalización utilizo el código desarrollado durante el curso que muestra la localización utilizando la API de Google Maps y, una vez obtenida la localización, con las coordenadas muestra el tiempo que hace en ese lugar a través de la API de OpenWeatherMap. Para obtener esto reutilizo parte del código que mostraba el tiempo al pasarle el nombre de la ciudad para que funcione con las coordenadas obtenidas.

Utilizando como base el js que mostraba la localización con una <section> que contiene el botón y en la que luego se muestra el mapa creo otra <section> para mostrar los detalles relacionados con el tiempo escribiendo en ella con innerHTML.

Con la información de esta página (https://openweathermap.org/current) formulo la petición necesaria para obtener la información.

21/06 - 19:00

Comienzo con la parte de las búsquedas en la web 500px utilizando su API. La consulta que hago es buscar imágenes asociadas a una palabra.

21/06 - 21:00

La aplicación que tenía registrada en la API de 500px ya no funciona ni tampoco la consumer key asociada. El servicio de API de esta página ya no permite registrar aplicaciones libremente y por ello voy a modificar mi plan original y utilizar la API de otro sitio relacionado con la fotografía.

21/06 - 23:00

Después de hacer varias pruebas con la de Instagram opto por utilizar la de Flickr. Modifico el archivo js para que funcione con la API de Flickr. A través de un formulario se le pasa el contenido de un textfield a la función del objeto creado, la cual hace una petición a Flickr y devuelve 6 fotos, relacionadas con dicha palabra, dentro de un <div> que es vaciado en cada llamada.

22/06 01:00

Comienzo con la base de datos en php. Me baso en el ejercicio que hice durante el curso que crea la base, las tablas e introduce información en ellas a partir de archivos php dedicados a cada tarea, con links y forms para introducir la información. Lo modifico con la base de datos que tengo diseñada.

22/06 - 15:00

Comienzo a validar las páginas html y css con los validadores del W3C y modificar lo necesario para que todas sean válidas. Después de modificar todo para que sea completamente válido estoy probando la página en los diferentes navegadores.

Por un lado, en local ni Edge ni Chrome interpretan la transformación xsl. Desde el servidor si que lo interpretan todos, pero en ninguno es posible ver la localización utilizando la API de Google Maps mientras que en local si es posible.

22/06 - 20:00

Haciendo una prueba con una persona llego a la conclusión de que el botón de 'Sitúame', para detectar la posición, situarte en el mapa y buscar el tiempo de la zona, no es muy visible. Decido aumentar el tamaño del botón e incluir la temperatura en la medida también.

22/06 - 23:00

Intento arreglar el que se muestre la imagen cuando la página está en Ritchie, pero no parece funcionar de ninguna manera. Hay un ejemplo entre los de JavaScript que, si funciona desde la página de Cueva mostrando el mapa, pero cuando copio el código y lo ejecuto desde Ritchie no funciona, por tanto, no sé cómo corregirlo.

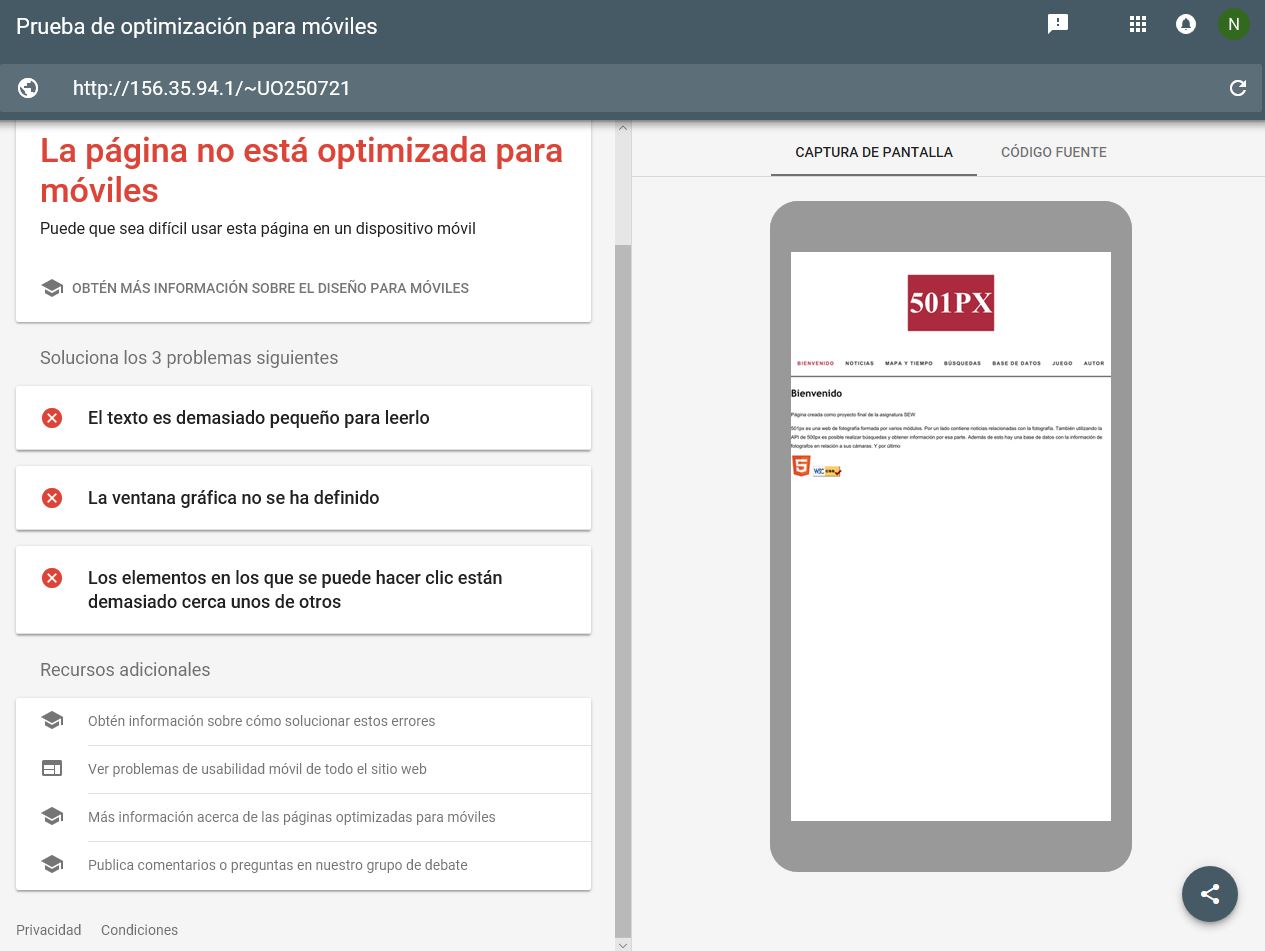
24/06 – 11:00

Antes de comenzar a comprobar la adaptabilidad compruebo que mi código utilice las técnicas descritas en el pdf del seminario de adaptabilidad, como utilizar porcentajes, en oposición a píxeles, cuando sea posible, y añadiendo una max-width del 100% con la altura variable en función del ancho para las imágenes.

Comienzo a probar la aplicación con las herramientas para comprobar la adaptabilidad:

Herramienta de Google:

Al principio me salen estos 3 errores.

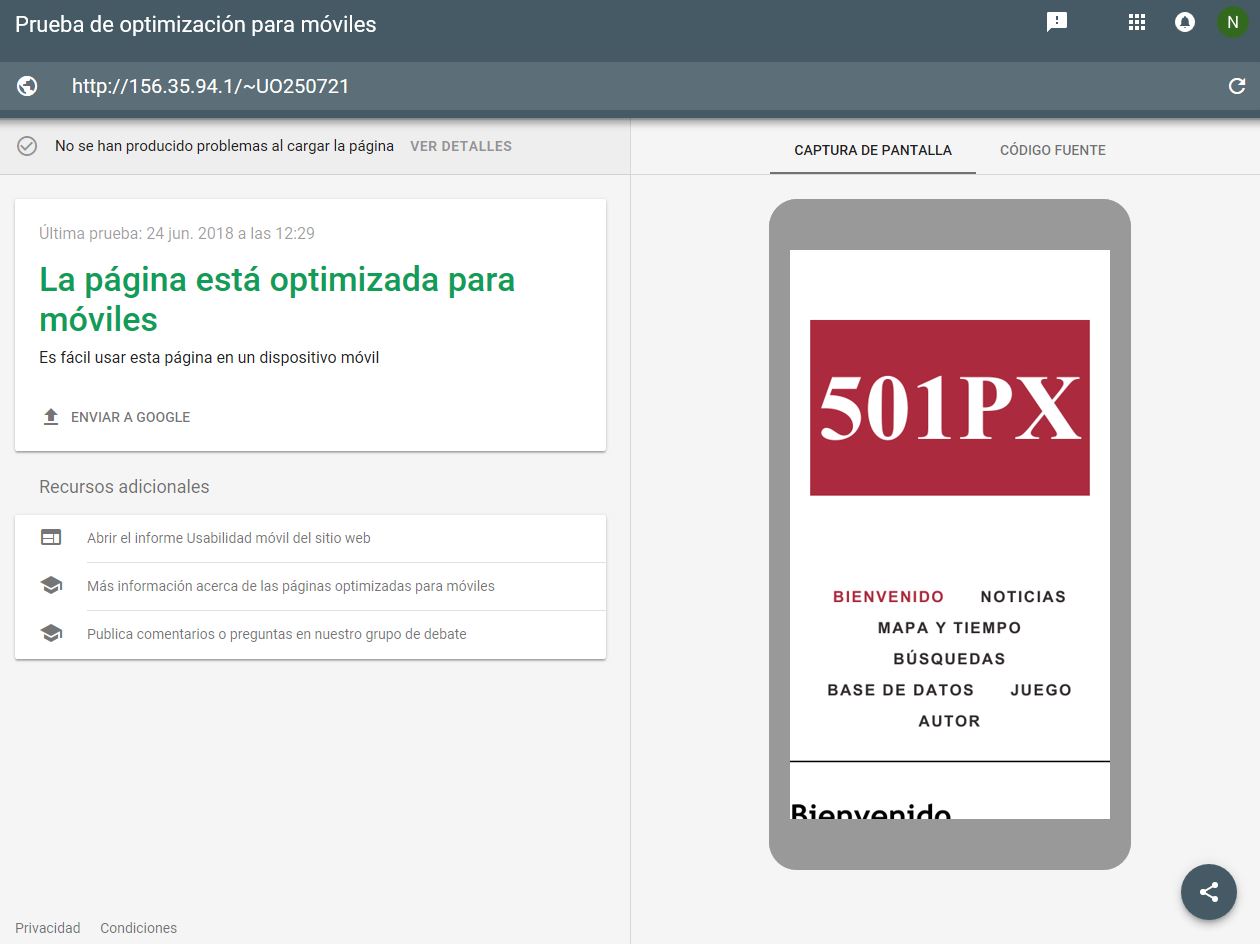


Para el problema del texto siendo demasiado pequeño le añado 5px a las fuentes más pequeñas para que se lea bien.

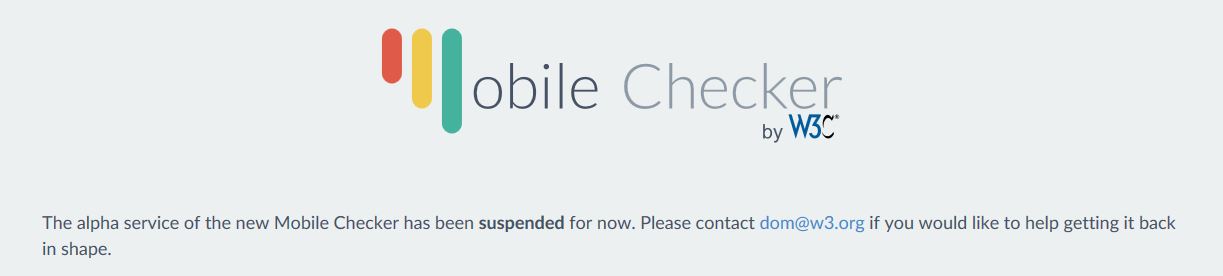
Añado padding de 10px a las imágenes con links dentro y a los elementos del menú.

Añado la etiqueta <meta name="viewport" content="width=device-width, initial-scale=1"> tal y como se explica en esta página (<https://developers.google.com/web/fundamentals/design-and-ux/responsive/>) para que el tamaño de la ventana de visualización del navegador sea adaptable .

Una vez corregidos ya muestra que la página está optimizada para móviles



Herramienta de diseño adaptable del W3C

Al entrar en el enlace aparece esto:  


Por tanto, no puedo usar esta herramienta para las comprobaciones.

Quirktools

Es completamente funcional todos los dispositivos actuales, incluso en tabletas con resoluciones muy bajas de hace años como esta de 800x480:

