Memoria PEC-2 Herramientas HTML y CSS II

1. Entorno de desarrollo

IDE para realizar el proyecto web: **VisualStudioCode**

El gestor de paquetes utilizado es NPM con la versión:

```
$npm -v = v8.6.0
```

Node con la versión:

```
$node -v = v16.14.2
```

2. Descargar/clonar el repositorio

A partir del enlace que nos brinda el módulo 1 de Herramientas HTML y CSS II he descargado del repositorio de GitHub el código del Boilerplate de la UOC mediante el botón verde CODE, que nos da la opción. A partir de aquí el .zip obtenido lo he descomprimido.

3. Instalación de dependencias de la PEC2 también en la PEC1.

1. **FontAwesome**
2. **htmlnano**
3. **cssnano**
4. **Sharp: optimización de imágenes.**
5. **Babel**

Para saber cómo instalar las dependencias anteriores consultar la **documentación de la PEC1** de Herramientas HTML i CSS II.

4. Instalación de la dependencia bootstrap.

```
$npm install @popperjs/core bootstrap –save
```

Para realizar la importación de dependencias para bootstrap dentro de **'src/assets/scripts/main.js'** añadimos una nueva línea con la importación correspondiente:

```
import * as bootstrap from 'bootstrap';
```

A continuación, en el fichero **src/assets/styles/main.scss** incluiremos los estilos, dentro de la sección **"Import npm dependencies"**:

```
@import "bootstrap/scss/bootstrap.scss";
```

***** Podemos importar solamente las partes de Bootstrap que nos interesen simplemente comentando la dependencia que no necesitemos.**

Mediante variables podemos personalizar Bootstrap (colores, tipografías,…). Como veremos más adelante en este documento.

5. Dependencia de iconos Bootstrap

```
$ npm install bootstrap-icons
```

Estos iconos tienen la ventaja de que han sido pensados para facilitar la **accesibilidad** a gente incapacitada a la hora de crear un web inteligible mediante el atributo **aria-label** cuando implemento el icono en el código.

Hay muchas maneras de implementar los iconos mediante bootstrap según las necesidades.

El inconveniente que tiene esta depedencia de bootstrap es que no hay muchos iconos donde escoger y por ese motivo también he instalado la dependencia de Fontawesome para implementar el de Google+.

La implementación de los iconos bootstrap la he realizado mediante SVG para mejorar el peso de la página al cargar.

6. Dependencia de Youtube Lite

Para mejora del rendimiento en la carga de la página de videos de youtube:

```
$npm install lite-youtube-embed
```

Esta dependencia nos sirve para cargar el recurso de vídeo de manera inteligente **minimizando** el impacto **en tiempo de carga** para este recurso tan pesado.

7. Instalación de Stylelint y personalización de la configuración.

La instalación de este **linter** con la **configuración** escogida mediante plugins y ficheros de configuración **es la misma** que para la PEC1 (**consultar memoria PEC1**) de esta asignatura:

– El estilo de guía escogido para implementar la PEC2 es la de **@mdo.** La justificación es parecida que en la PEC1: Aún y que este proyecto es más grande que el de la PEC1, el orden de las propiedades en las reglas y también de los atributos en el HTML de los elementos me empujan a escoger @mdo porque facilitan muy bien la lectura del documento y el mantenimiento.

– La metodología que he utilizado para codificar las hojas de estilo es ITCSS con nomenclatura BEM (**ITCSS + BEM = BEMIT**). En este proyecto no hay gran cantidad de componentes ni tampoco es un SPA como la PEC1, pero el hecho de dividir los estilos sobretodo **conceptualmente** en genéricos, objetos, elementos, componentes, settings, ...y junto con la nomenclatura BEM no resulta difícil encontrar la clase que estamos buscando de entre el código y modificarla, ni tampoco conocer donde está utilizándose solamente viendo la App visualmente desde el navegador. **He hecho hincapié en no implementar estilos de elemento siguiendo as recomendaciones de la corrección de la PEC1**; sino más bien implementar estilos de clases para ligar con ITCSS.

8. Desarrollo de la página HOME con Grid.

La página de **Inicio de la Web está diseñada** mediante la propiedad CSS ‘**display: grid’.**

La estructura de la página es una parrilla o retícula formada por **3 columnas y 5 filas**, que entrelazándose conforman un total de **15 áreas**. La primera fila pertenece al Header y la última al Footer. Las de en medio para fotos y texto.

Utilizamos la consulta de **@supports de CSS** de la propiedad Grid en el navegador que va a cargar la página para preguntar si éste soporta Grid o no. Si no lo soporta entonces aplicamos reglas más convencionales que aproximen el comportamiento de la propiedad Grid.

Para simular Grid en navegadores antiguos, he utilizado la propiedad CSS **float** aplicada a todos los hijos del elemento padre que contiene la clase **.c-container-inicio,** donde voy colocando a la izquierda y en el orden que determina el documento HTML todos los elementos hijos (con **float:left**). Los elementos flotantes se van colocando uno debajo del otro según su ancho y según el ancho de la pantalla del dispositivo.

Si el navegador encuentra una propiedad que no conociera, la ignora y continua con la ejecución de la hoja de estilos. En el caso del elemento Footer de nuestro proyecto, en la implementación, encontramos la propiedad **grid-column:1/-1 que el navegador que no la reconozca simplemente ignorará** y por tanto no hace falta realizar otra consulta del tipo @supports de si soporta o no grid para este elemento.

Para saber **si debemos implementar reglas CSS más convencionales** todo depende de la **lista de navegadores** que hayamos predefinido (e.g en nuestro package.json) que ha de soportar nuestra App. Los **polyfills** que emulan a un **display:grid** no son del todo efectivos y por eso es más eficiente implementar código nuevo con reglas CSS más convencionales o antiguas para navegadores antiguos para esta propiedad.

Para saber si una propiedad está soportada por según qué versión de navegador, por ejemplo, podemos averiguarlo mediante el sitio web **how can i use CSS** (buscar en Google) donde nos muestra la compatibilidad de los navegadores con la propiedad CSS a implementar.

9. Problemas encontrados.

1. He intentado parametrizar el color de los iconos sociales del tipo bootstrap en el footer mediante **variables scss** sin éxito (debido a a que los he embebido mediante código SVG). El icono de Fontawesome de Google + no se encuentra en los iconos disponibles en Bootstrap.

2. Al utilizar la clase predefinidas de bootstrap **.row** he observado que por defecto tienen un margen derecho e izquierdo para la **fila** en cuestión afectando al diseño web produciendo la barra de **scroll horizontal en el navegador** no deseada. He solucionado esta incidencia poniendo las propiedades **de margen derecho e izquierdo** de la clase .row **a cero en nuestro fichero de estilos de la App**.

3. El incluir bootstrap al proyecto ha alterado el aspecto de los enlaces de la App porque al pasar por encima el ratón cambian de color, efecto que yo no he implementado en mi código. Lo he dejado tal cual porque me sirve, pero hay que tenerlo en cuenta.

10. Componentes de bootstrap en nuestro proyecto web.

1. **Botón:** utilizamos clases predefinidas para darle aspecto.

2. **Lista desordenada** predefinida de Bootstrap: He modificado el componente bootstrap para que se asemeje a la lista que encontramos en el wireframe del enunciado.

3. **Componente Alert** de bootstrap: El componente se muestra cuando el usuario clica sobre el botón enviar formulario y éste es correctamente cumplimentado.

4. **Dropdown de bootstrap**: He personalizado el componente mediante las variables Sass disponibles.

11. Personalización de un componente Bootstrap.

Para personalizar un componente bootstrap que hayamos utilizado en nuestro proyecto se realiza mediante el fichero de variables scss llamado **variables.scss**

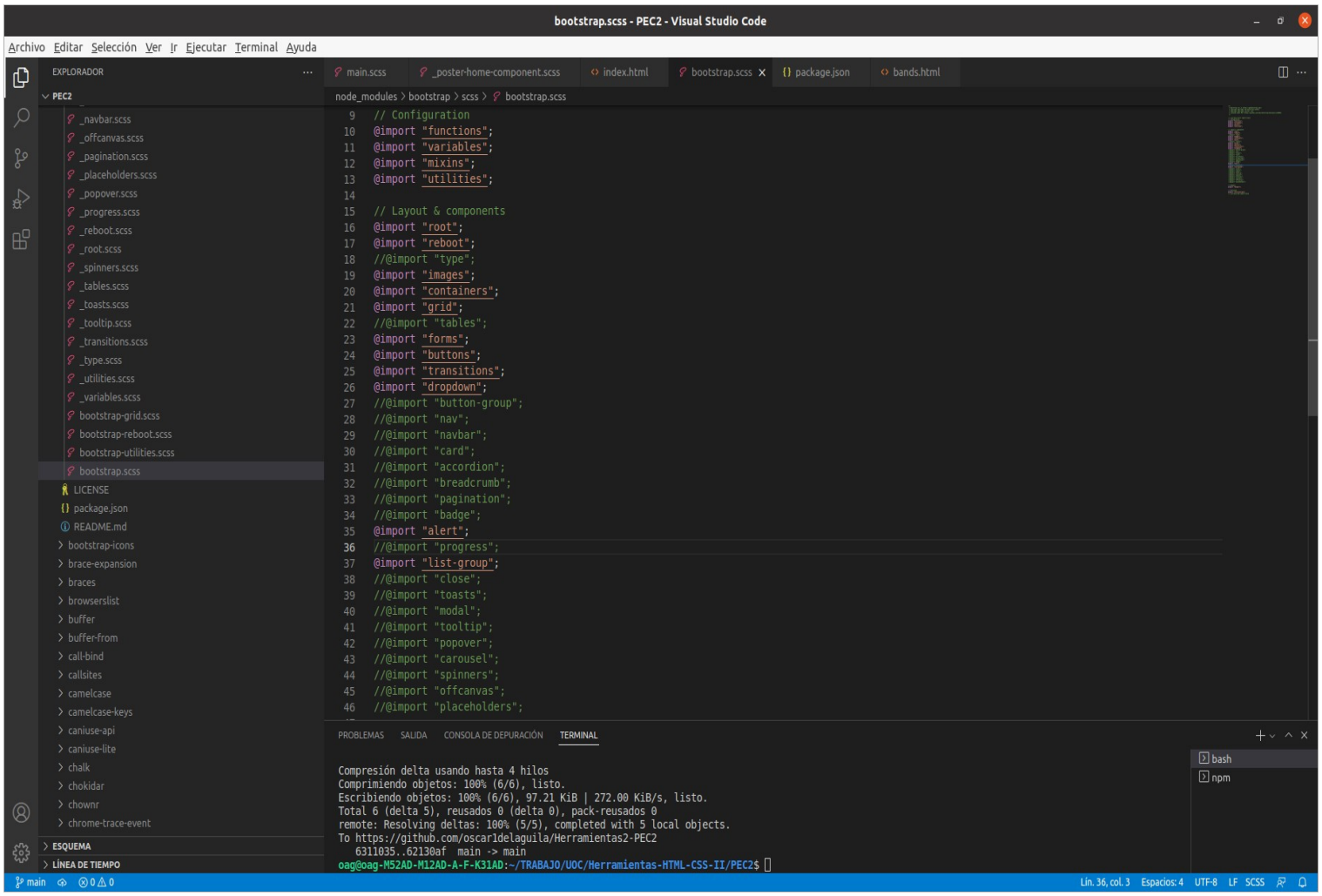
ubicado en el directorio **node_modules/bootstrap/scss/_variables.scss** de nuestro proyecto**.** En este fichero bootstrap guarda las variables que hacen referencia a los componentes y que podemos cambiar su valor para personalizar el aspecto del componente para que se muestre en la web. Así pues el componente **dropdown** lo he personalizado y he cambiado el valor de **dos variables referentes a este componente** (las vemos en amarillo en el siguiente código):

```
// Dropdowns
//
// Dropdown menu container and contents.

// scss-docs-start dropdown-variables

$dropdown-min-width: 10rem !default;
$dropdown-padding-x: 0 !default;
$dropdown-padding-y: .5rem !default;
$dropdown-spacer: .125rem !default;
$dropdown-font-size: $font-size-base !default;
$dropdown-color: $body-color !default;
$dropdown-bg: $white !default;
$dropdown-border-color: rgba($black, .15) !default;
$dropdown-border-radius: $border-radius !default;
$dropdown-border-width: $border-width !default;
$dropdown-inner-border-radius: subtract($dropdown-border-radius, $dropdown-border-width) !default;
$dropdown-divider-bg: $dropdown-border-color !default;
$dropdown-divider-margin-y: $spacer * .5 !default;
$dropdown-box-shadow: $box-shadow !default;

$dropdown-link-color: $gray-900 !default;
$dropdown-link-hover-color:   white;      //shade-color($dropdown-link-color, 10%) !default;
$dropdown-link-hover-bg:      $primary;    //$gray-200 !default;
```

Ponemos de color blanco la fuente del enlace al pasar por encima el ratóny el fondo del color primario de bootstrap (azul).

12. Comentar ‘aquello’ que no utilizamos de bootstrap

Del fichero bootstrap.scss comentamos en la sección de **layouts y componentes**:



Desde este fichero **accedemos a las variables de personalización** de los componentes de bootstrap clicando sobre **@import “variables”**.

Comentando lo que no utilizamos de bootstrap ahorramos mucho tiempo en desarrollo cada vez que construye el proyecto (‘npm run dev’).

13. Crear repositorio mediante GitHub

Subir el código al repositorio GitHub

1. Creamos un repositorio que contenga todos los ficheros del proyecto en GitHub desde este sitio web.

Desde el terminal realizamos las siguientes acciones y/o comandos:

1. No situamos en la carpeta ‘PEC2’ donde se encuentran todos los ficheros que queremos subir al repositorio.

2. Ejecutamos **git init** (Comando que inicializa un repositorio local de git).
He copiado el fichero .gitignore del proyecto de la UOC proporcionado en el enunciado y he eliminado la entrada dist de este fichero para que al realizar el comando del siguiente punto 3 el directorio dist se añada al repositorio.

3. **git add .** (Añade todos los ficheros del directorio actual ‘PEC1’ al repositorio local y los prepara para poder realizar el commit).

4. **git commit -m “version1.0.0”** (Hacemos commit de los ficheros añadidos al repositorio local). (Puede ser que nos pida autentificación en local, seguid los pasos).

5. **git branch -m main** (El commit anterior ha provocado la creación de una rama llamada ‘master’, pero en nuestro repositorio remoto de GitHub la rama principal se llama ‘main’ así que cambiamos el nombre de la rama en local para que coincidan y no haya conflicto con este comando).

6. **git remote add origin <remote url>**
(Asignamos la dirección de nuestro repositorio remoto que en nuestro caso es “https://github.com/oscar1delguila/Herramientas2-PEC2.git” a ‘origin’).

7. **git remote -v** (Comprobamos que realmente se ha asignado bien la dirección remota).

8. **git push origin main** (Subimos a nuestro repositorio remoto los ficheros desde nuestro repositorio local, en la rama main).

14. Publicación de la web en Netlify

1. Realizo una vinculación entre el repositorio remoto en Github y Netlify.

2. En nuestro caso queremos que Netlify construya nuestro proyecto con nuestro **package.json** desde cero (**deploy**). A la hora de deployar (**Netlify** ejecuta **npm run build**) **quitamos** la ejecución del script de **stylelint** para que no de error.

3. Configuramos Netlify para que detecte cambios en el repositorio GitHub y automáticamente deploye el proyecto mediante un botón.

15. Incluir los enlaces al repositorio de GitHub y la URL pública de Netlify.

– URL Curriculum vitae

https://bandas-musica-oag.netlify.app

– Repositorio github:

https://github.com/oscar1delaguila/herramientas2-PEC2.git