

P1

Autor: **Néstor Bethencourt Santana**
Asignatura: **Herramientas HTML y CSS II**
Fecha: **3/11/2024**

Boilerplate.....	3
Entorno de desarrollo y ejecución del proyecto.....	3
Despliegue a producción.....	4
Metodología.....	7
Dependencias.....	9
Acceso a repositorio de GitHub y URL pública de Netlify.....	9

Boilerplate

El comienzo de la práctica ha consistido en clonar el UOC Boilerplate.
Para el sistema operativo Windows, mediante Git bash he ejecutado:

```
git clone  
https://github.com/uoc-advanced-html-css/uoc-boilerplate.git
```

Seguidamente renombro la carpeta como PEC1-HTML-CCS-II y configuro Git para subir los cambios a un repositorio propio.

```
git remote remove origin  
git remote add origin  
git@github.com:nbethencourts/herramientas-html-css-2-pec-1.git  
git branch -M main  
git push -u origin main
```

Con esto ya está el UOC Boilerplate en GitHub listo para realizar la práctica.

Entorno de desarrollo y ejecución del proyecto

Como entorno de desarrollo he optado por utilizar el Visual Studio Code.
Para la instalación de todas las dependencias del proyecto basta con ejecutar en el directorio del proyecto:

```
npm install
```

La compilación para producción se realiza con:

```
npm run build
```

De esta forma se crea la carpeta `dist` que contiene la aplicación lista para ponerla en producción.

A la hora de desarrollar he utilizado el comando `npm run dev`. Este comando permite entre otras cosas la actualización en caliente de la aplicación web.

En `http://localhost:8123` se puede ver la aplicación y a la vez que se realizan cambios en el código la aplicación se actualiza automáticamente en el navegador.

Despliegue a producción

Para desplegar a producción he utilizado las cuentas que ya tenía creadas para otras asignaturas en GitHub y Netlify. Una vez autenticado en las dos plataformas, he dado permisos a Netlify para que pueda acceder al proyecto de esta PEC en GitHub

Desde Netlify he elegido desplegar el proyecto de GitHub

1. Connect to Git provider


2. Select repository


3. Configure site

Let's deploy your project with...

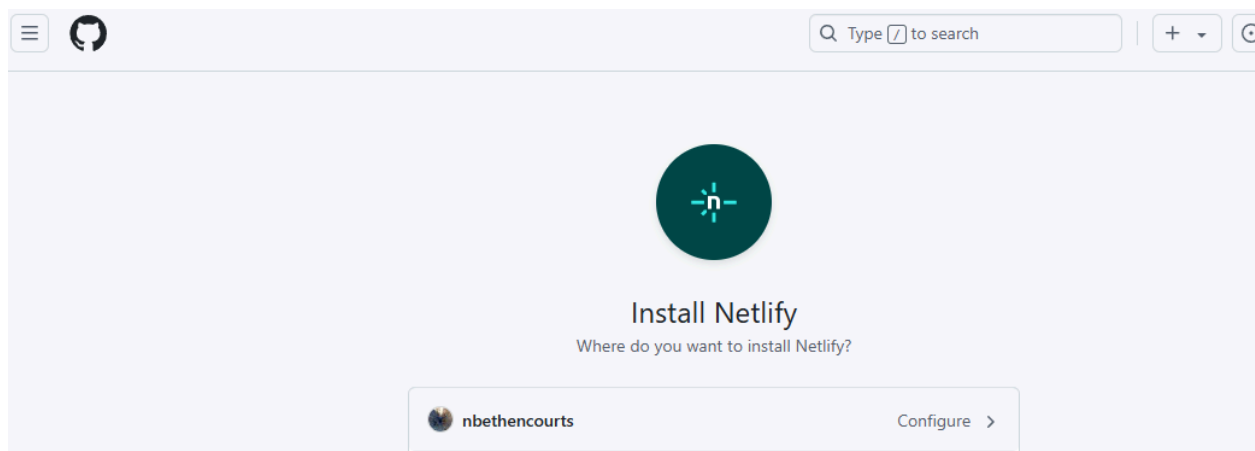
 GitHub

 GitLab

 Bitbucket

 Azure DevOps

En GitHub he dado permisos para que Netlify acceda al repositorio:



Your site will automatically be secured with a free TLS certificate from Let's Encrypt.

Deploy Previews

Netlify's Deploy Previews streamlined your workflow by giving you a unique URL for every change. Changes will be like in production whenever you submit a pull request.

And a lot more!

Netlify includes solutions for forms, identity, and even custom functions powered by serverless. All functionality can be configured directly in your repo.

Permissions

✓ **Read** access to code and metadata

✓ **Read and write** access to checks, commit statuses, issues, and pull requests

Repository access

☐ **All repositories**




This applies to all current *and* future repositories owned by the resource owner.
Also includes public repositories (read-only).

☒ **Only select repositories**

Select at least one repository.
Also includes public repositories (read-only).

 **Select repositories ▼**

Selected 3 repositories.

 nbethencourts/herramientas-html-P1	×
 nbethencourts/herramientas-html-css-2-pec-01	×
 nbethencourts/herramientas-html-css-2-pec-1	×

Se configura el build y finalmente se pulsa en Deploy:

Build settings

Specify how Netlify will build your site.

[Learn more in the docs](#) ↗

Branch to deploy

main



Base directory

The directory where Netlify installs dependencies and runs your build command.

Build command

npm run parcel:build

Examples: jekyll build, gulp build, make all

Publish directory

dist

Examples: _site, dist, public

Functions directory

netlify/functions

Example: my_functions

Environment variables

Define environment variables for more control and flexibility over your build.

Add environment variables

Deploy herramientas-html-css-2-pec-1

En <https://pec1html-css-ii.netlify.app> se puede acceder a la página.

Metodología

He escogido la metodología BEM porque para el desarrollo de una web de una sola página se adapta perfectamente debido a la sencillez de la metodología, a la vez que permite obtener un código modular, reusable y estructurado.

Para la estructura de ficheros he elegido la aproximación [Flex](#) de las diferentes que hay para aplicar BEM, ya que permite crear en un fichero tanto la definición de un bloque, como de sus elementos y modificadores. Al ser un proyecto pequeño, no hace falta crear tantas carpetas y ficheros para definir los estilos.

También he aplicado lo indicado por Mark Otto en su guía de estilo en lo referente al [orden de las declaraciones](#), ya que es una forma de mantener el conjunto de declaraciones ordenado y consistente.

Además, he seguido un enfoque mobile first para realizar la implementación del diseño, es decir, he partido de un diseño que se ajusta a un dispositivo móvil, y he ido modificando las páginas para ir aprovechando el espacio, adaptando el diseño a medida que el tamaño de pantalla es más grande.

Si bien he podido utilizar en algunos casos grid, he resuelto los problemas planteados utilizando flexbox porque encajaba bien con el diseño que quería implementar.

La estructura de archivos, siguiendo la metodología indicada es la siguiente:

- **styles/base:** estilos de inicialización (`_reset.scss`).
- **styles/components:** cada bloque de la página web (carrusel, container, footer, header, map, etc) lo he definido en su propio archivo, cumpliendo con el principio de modularidad de BEM.

En estos mismos archivos he definido los elementos que componen el bloque, así como sus modificadores. Por ejemplo, para el caso del bloque carrusel, he definido elementos como `slide` (`carrusel__slide`), `img` (`carrusel__img`) o `button` (`carrusel__button`).

Además, si he necesitado modificar el estilo de algún elemento o bloque, he definido un modificador. Por ejemplo, para el caso del contenedor (`container`), el elemento `card` (`container_card`) necesité modificarlo para tratar a uno de los elementos con unas características diferentes, por lo que modifiqué su estilo con un modificador (`container_card--max`).

- **styles/layouts**: en esta carpeta se organizan los estilos de las páginas o vistas completas. En este caso, como la web se trata de una sola página, sólo se encuentran los estilos de la página home.
- **styles/_variables.scss**: aquí he definido como variables los diferentes colores utilizados en la web, los diferentes tamaños o breakpoints a tener en cuenta para el diseño responsive o las fuentes a utilizar.
- **views/footer**: aquí se encuentra el código Html que define el pie de la página.
- **index.html**: código Html de la página completa, excepto el pie.

Configuración Stylelint

Mediante `npm install --save-dev stylelint` se instala Stylelint.

Como se usa la sintaxis SCSS de Sass, también he instalado `stylelint-scss` y su configuración base recomendada, `stylelint-config-recommended-scss`.

```
npm install --save-dev stylelint-scss
stylelint-config-recommended-scss
```

Para el orden de las declaraciones:

```
npm install --save-dev stylelint-config-recess-order
```

Para seguir la metodología BEM:

```
npm install --save-dev stylelint-selector-bem-pattern
```

El plugin `stylelint-selector-bem-pattern` lo he configurado en `.stylelintrc` de la siguiente forma:

```
"rules": {
  "plugin/selector-bem-pattern": {
    "implicitComponents": true,
    "preset": "bem"
  }
}
```

Esta configuración permite que se compruebe que se cumplan las siguientes reglas de nomenclatura:

- **B (bloques)**: debe estar compuesto por letras minúsculas y números.
- **E (elementos)**: los bloques podrán ir seguidos de dos guiones bajos con letras minúsculas y números, para indicar el elemento.
- **M (modificadores)**: los bloques y elementos podrán ir seguidos por dos guiones medios con letras minúsculas, números y un guión medio, que corresponderá al modificador.

Dependencias

- Librería de iconos **FontAwesome**:
`npm install --save @fortawesome/fontawesome-free`
- La librería que implementa un carrusel de imágenes denominada **Swiper**:
`npm install --save swiper`

Acceso a repositorio de GitHub y URL pública de Netlify

La url pública de la web es <https://pec1html-css-ii.netlify.app/> y el repositorio en Github <https://github.com/nbethencourts/herramientas-html-css-2-pec-1>