Parte 2: Documentación

Parte 2: Documentación

Para el desarrollo de esta practica utilizaremos el editor de texto de Visual Studio Code y seguiremos los pasos del modulo 2 para la creación del boilerplate.

1. Creación de la carpeta de proyecto y los archivos:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente



En este punto no tienen contenido, solo creo la estructura.

1. Instalación de NodeJS

Para la gestión de paquetes (scripts de Javascript) emplearemos npm, para ello instalaremos NodeJS. Para instalarlo, iremos a la pagina del fabricante y descargaremos la última versión LTS. En este caso ya tenia Node instalado en el equipo por lo que la versión es 16.17.1.

Aplicaremos las siguientes lineas para la configuración y añadir el autor y la licencia:

npm set init-author-email "cbalarezo@uoc.edu"

npm set init-author-name "Claudia Anahí Balarezo Quiroz"

npm set init-license "MIT"

1. Package.JSON

Package.json es un archivo que contiene información importante del proyecto, esta alojada en la raíz del proyecto. Para crearlo ejecutaremos la sentencia:

npm init –yes

Y modificaremos los datos de source, por nuestra raíz:

"source":"src/portada.html",

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente



Que seria como nuestro “index”.

1. Empaquetador de modulo

Como module bundler utilizaremos Parcel. Parcel procesa todo el código de la aplicación y los empaqueta generando en una carpeta “Dist” archivos compatibles con el navegador. En primer lugar, utilizaremos Parcel en el entorno de desarrollo para lanzar un servidor y monitorizar los cambios. Para instalarlo, ejecutaremos lo siguiente:

npm install --save-dev parcel

npm install --save-dev rimraf npm-run-all

Y modificaremos el package.json para añadir las siguientes líneas:

"scripts":

"parcel:dev": "parcel",  
"parcel:build": "parcel build",  
"clean": "rimraf dist .parcel-cache",

"start": "npm-run-all clean parcel:dev",  
"build": "npm-run-all clean parcel:build"

Para que cuando compilemos, la carpeta dist donde se genera el empaquetado de código, se elimine y se vuelva a crear, ya que si no, tendríamos varios archivos.

1. Versión de navegador

Modificaremos el package.json para añadir una línea en la que indicaremos las versiones de navegador compatibles con nuestra aplicación web.

"browserslist":"> 0.5%, last 2 versions, not dead",

Finalmente el archivo package.json quedaría de la siguiente manera:

Texto

Descripción generada automáticamente

1. Autoprefixer

Utilizaremos el autoprefixer que ya incluye Parcel para que de manera automática agregue los prefijos.

1. Ejecutaremos npm run build y podremos ver en <http://localhost:1234> nuestra aplicación.