* Crear carpeta
* Abrir con VSCode (code .)
* Crear index.html
* Comandos de consola a ejecutar en el directorio principal del proyecto:
  + npm init :
    - npm init es un comando que se utiliza para crear un nuevo archivo package.json en el directorio actual. package.json es un archivo clave en cualquier aplicación basada en Node.js. Contiene metadatos sobre la aplicación, como el nombre, la versión y las dependencias (otros paquetes de Node.js que tu aplicación necesita para funcionar).

Aquí está el proceso paso a paso que npm init sigue:

* + - * Pide el nombre de tu paquete.
      * Pide la versión de tu paquete.
      * Pide la descripción de tu paquete.
      * Pide el punto de entrada de tu paquete (normalmente index.js).
      * Pide el comando de prueba de tu paquete.
      * Pide el repositorio git de tu paquete.
      * Pide las palabras clave de tu paquete.
      * Pide el autor de tu paquete.
      * Pide la licencia de tu paquete.
      * Crea el archivo package.json con la información proporcionada.
      * Si no quieres responder a estas preguntas una por una, puedes usar npm init -y para generar automáticamente un archivo package.json con valores predeterminados.
  + npm install -D live-server:
    - El comando npm install -D live-server instala el paquete live-server como una dependencia de desarrollo en tu proyecto Node.js.

live-server es un pequeño servidor de desarrollo con capacidad de recarga en vivo. Se utiliza principalmente para proyectos de front-end donde se necesita recargar automáticamente la página web cuando se realizan cambios en los archivos del proyecto.

* + - Aquí está lo que hace el comando paso a paso:
    - npm install: Este es el comando que le dice a npm (el administrador de paquetes de Node.js) que instale un paquete.
    - -D: Esta es una bandera que indica que el paquete debe ser instalado como una dependencia de desarrollo. Las dependencias de desarrollo son paquetes que no son necesarios para que tu aplicación funcione en producción, pero son necesarios para el desarrollo. Por ejemplo, herramientas de prueba o, en este caso, un servidor de desarrollo.
    - live-server: Este es el nombre del paquete que quieres instalar.

Después de ejecutar este comando, live-server se añadirá a la sección devDependencies de tu archivo package.json.

Los archivos package.json y package-lock.json son fundamentales en cualquier proyecto de Node.js. Aquí te explico para qué sirve cada uno:

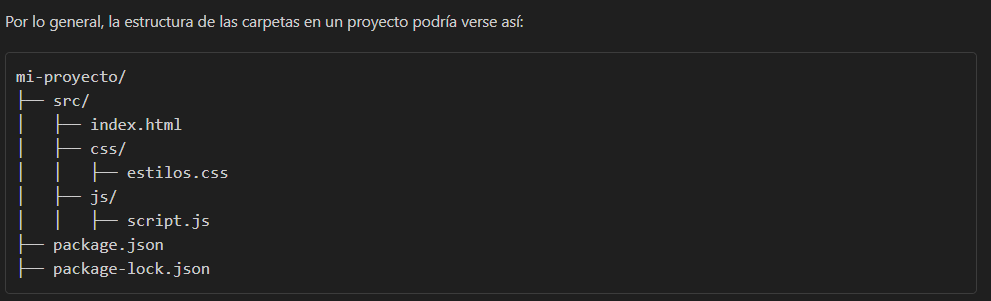
* **package.json**: Este archivo contiene metadatos sobre tu proyecto, incluyendo el nombre, la versión, la descripción, los scripts que se pueden ejecutar, y las dependencias que tu proyecto necesita para funcionar. Las dependencias son otros paquetes de Node.js que tu proyecto utiliza. Cuando ejecutas npm install <nombre-del-paquete>, npm añade ese paquete a la lista de dependencias en tu package.json.
* **package-lock.json**: Este archivo se genera automáticamente cuando instalamos paquetes con npm. Contiene información exacta sobre qué versiones de cada dependencia se instalaron. Esto asegura que cuando otra persona instala tu proyecto, obtiene exactamente las mismas versiones de las dependencias que tú tienes instaladas, lo que ayuda a evitar problemas de incompatibilidad entre versiones. También acelera el proceso de instalación de dependencias, ya que npm puede usar la información en package-lock.json para evitar tener que buscar la última versión de cada paquete.

En resumen, package.json es para los humanos y package-lock.json es para las máquinas. Como desarrollador, normalmente interactúas con package.json para gestionar tus dependencias, mientras que package-lock.json será manejado automáticamente por npm.

* Crear carpeta src en raíz del proyecto.
* Meter index.html dentro de la carpeta src.
* Abrir package.json:
  + Añadir en “scripts” :"start": "live-server src/"
* En terminal ejecutar servidor con npm start
  + El comando npm start ejecuta el script "start" definido en tu archivo package.json. En este caso, el script "start" está configurado para ejecutar live-server src/.
  + live-server src/ inicia el paquete live-server en el directorio src/ de tu proyecto. Esto servirá los archivos en ese directorio en un servidor local y recargará automáticamente la página en tu navegador cada vez que se realice un cambio en esos archivos.

Por lo tanto, al ejecutar npm start, estás iniciando un servidor de desarrollo local que sirve los archivos en tu directorio src/ y recarga la página automáticamente cuando se realizan cambios en esos archivos.

* Crear contenido del html con la siguiente línea: main>div.container>div.pixel\*5
* Crear carpeta css dentro de src:



* Vincular archivo CSS en el archivo HTML.
* Dar estilos a los div para verlos.
* Crear carpeta js y archivo index.js.
* Vincular archivo js en el archivo HTML.
* En index.js:
  + window.onload = function () { console.log("loaded"); };

Este fragmento de código JavaScript se ejecuta cuando la página web ha terminado de cargar.

window.onload es un evento que se dispara cuando todo el contenido de la página (incluyendo imágenes, scripts, CSS, etc.) ha terminado de cargar.

En este caso, cuando se dispara el evento window.onload, se ejecuta la función que imprime el mensaje "loaded" en la consola del navegador.

Por lo tanto, si ves el mensaje "loaded" en la consola, puedes estar seguro de que todo el contenido de la página se ha cargado completamente.

* Añadir eventos…