Aplicación sort

Primero vamos a crear el directorio.

* Sort.
  + Src
    - Index.ts
  + Build

Creamos el archivo de configuración

**Tsc --init** con este archivo especificamos las configuraciones deseadas.

Dentro una de las opciones importantes son:

outFile: especifica el nombre del archivo

outDir: “./build”especifica el directorio relativo(desde donde se ejecuta este archivo) donde pondrá todos los archivos compilados (archivos js)

rootDir: “./src”Especifica la ruta de donde obtendremos los archivos, aunque si no especificamos busca en todo el directorio del proyecto (archivos ts) al hacer tsc, pero si queremos que dentro del directorio root de forma automática, hacemos: **tsc -**w y automáticamente compilará los cambios.

Aunque compila los cambios no los ejecuta de forma automática, para esto podemos instalar nodemon y concurrently:

**npm install nodemon concurrently**

como ya tengo nodemon, solo concurrently (permite correr varios scripts al mismo tiempo):

**npm install concurrently**

Configuremos el archivo package.json para que corra de forma automática con los comandos, en scripts:

“start:build”:”tsc -w”, //Este comando corre tsc -w

“start:run”: “nodemon build/index.js” //Este comando el nodemon

“start”: “concurrently npm:start:\*” //Este comando ejecuta los dos anteriores y ya no tenemos que abrir dos terminales, solo ejecutar este comando.

Construyendo la APP

Usaremos bubble sort – ordenamiento burbuja

Se hace el clase, problema, al realizar el método makeSort() para ordenar solo funciona con number pero si queremos que funcione con otros tipos como string podemos hacer super(public collecition: number[] | string) el problema es que no podemos tener el mismo algoritmo para string y number[], además que tsc solo permitirá usar los métodos compartidos, que en este caso no es más que el acceso de solo lectura para los índices. Para evitar esto usaremos ***type guard:***

***If(this.collection instanceof Array )***

***{}***

Indica que el tipo que se manejará de los ya recibidos es Array, de esta manera devuelve las propiedades y métodos propios de tipo.

Para comprobar el tipo usamos **if(algo instaceof typoalgo)** cuando no es un valor primitivo, que pueden ser array, clases Sorter(ejemplos), objetos ect. y cuando es primitivo como **string, number, boolean, symbol** usamos **if(typeof algo===’tipoacomprobar’)**