1.3 TypeScript Decorators

This section will guide you to:

* Implement decorators
* Use Visual Studio Code
* Push code to Git

This lab has three subsections, namely:

* + 1. Writing a function in TypeScript to implement decorators
    2. Compiling the code
    3. Pushing the code to your GitHub repositories

**Step 1.3.1:** Writing a function in TypeScript to implement decorators

* Open Visual Studio Code, open a folder named **TypeScript Demos**, and create a new file named **main.ts**.
* Write the program in TypeScript.

*enum ValidationType {*

*NotNull*

*}*

*function validate(...types: ValidationType[]) {*

*return function (target: any, propertyKey: string) {*

*Validator2.registerValidators(target, propertyKey, types);*

*}*

*}*

*class Validator2 {*

*private static notNullValidatorMap: Map<any, string[]> = new Map();*

*//todo add more validator maps*

*static registerValidators(target: any, property: any, types: ValidationType[]): void {*

*for (const type of types) {*

*if(type==ValidationType.NotNull){*

*let keys: string[] = this.notNullValidatorMap.get(target);*

*if (!keys) {*

*keys = [];*

*this.notNullValidatorMap.set(target, keys);*

*}*

*keys.push(property);*

*}*

*}//todo add more validators if else*

*}*

*static validate(target: any): boolean {*

*let notNullProps: string[] = this.notNullValidatorMap.get(Object.getPrototypeOf(target));*

*if (!notNullProps) {*

*return true;*

*}*

*let hasErrors: boolean = false;*

*for (const property of notNullProps) {*

*let value = target[property];*

*if (!value) {*

*console.error(property + " value cannot be null");*

*hasErrors = true;*

*}*

*}*

*return hasErrors;*

*}*

*}*

*class Person2 {*

*@validate(ValidationType.NotNull)*

*name: string;*

*constructor(name: string) {*

*this.name = name;*

*}*

*}*

*console.log("-- creating instance --");*

*let person3: Person2 = new Person2(null);*

*console.log(person3);*

*let b2 = Validator2.validate(person3);*

*console.log("validation passed: " + !b2);*

*console.log("-- creating another instance --");*

*let person4: Person2 = new Person2("Tina");*

*console.log(person4);*

*b2 = Validator2.validate(person4);*

*console.log("validation passed: " + !b2);*

**Step 1.3.2:** Compiling the code

* To compile the code, you can open the Integrated Terminal (Ctrl+`) and type the following:

*tsc main.ts*

* This will compile and create a new main.js JavaScript file.
* Type *node main.js* command

Output:

*-- creating instance --*

*Person2 { name: null }*

*name value cannot be null*

*validation passed: false*

*-- creating another instance --*

*Person2 { name: 'Tina' }*

*validation passed: true*

**Step 1.3.3:** Pushing the code to your GitHub repositories

Open your command prompt and navigate to the folder where you have created your files

cd <folder path>

Initialize your repository using the following command:

git init

Add all the files to your git repository using the following command:

git add . 

Commit the changes using the following command:

git commit . -m “Changes have been committed.”

Push the files to the folder you initially created using the following command:

git push -u origin master