Class

Example 1:

class Bulb

{

printBrand():void{

console.log("Philips");

}

}

function eg1(){

let b:Bulb; // data type of b is Bulb, it can store address of Bulb type object

// b is pointer.

b = new Bulb();

b.printBrand();

}

eg1();

Construtor Example

class Bulb

{

constructor()

{

console.log("Instance of Bulb Instantiated.");

}

printBrand():void

{

console.log("Philips");

}

}

function eg1(){

let b:Bulb;

b = new Bulb();

b.printBrand();

}

eg1();

class Bulb

{

wattage:number;

constructor()

{

console.log(`Bulb Instantiated `);

}

printBrand():void

{

console.log(`Philips and wattage is ${this.wattage}`);

}

}

function eg1(){

let b:Bulb;

b = new Bulb();

b.printBrand();

}

eg1();

/\*

Philips and wattage is undefined

\*/

Getting undeinfed in js response.

We we can do 2 options

One TSC which will propile to code with strict mode

Or we can do

> tsc –strict eg1.ts

See now:

class Bulb

{

wattage:number;

constructor()

{

console.log(`Bulb Instantiated `);

}

printBrand():void

{

console.log(`Philips and wattage is ${this.wattage}`);

}

}

function eg1(){

let b:Bulb;

b = new Bulb();

b.printBrand();

}

eg1();

/\*

eg4.ts:3:5 - error TS2564: Property 'wattage' has no initializer and is not definitely assigned in the constructor.

3 wattage:number;

~~~~~~~

Found 1 error in eg4.ts:3

\*/

Even if we put setter in the class still same response.

setWatteage(wattage:number):void

{

this.wattage = wattage;

}

Compiler needs in constructor, this is method, its not required everyone will call this.

Same story for calling setter without initialising in construtor.

In case if we have given any property and not used it will also give error to us.

Then how to add property in a way that if we will not use it will not give Error

price : number; // Error

price ! : number // allowed

With, how to use.

// for compiling use tsc --strict name.ts

class Bulb

{

wattage:number;

price!:number

constructor()

{

this.wattage = 0;

console.log(`Bulb Instantiated `);

}

setWattage(wattage:number)

{

this.wattage = wattage;

}

setPrice(price:number)

{

this.price = price;

}

printInfo():void

{

console.log(`Philips and wattage is ${this.wattage}`);

if(this.price)

console.log(`Price is ${this.price}`);

}

}

function eg9(){

let b:Bulb;

b = new Bulb();

b.setWattage(60);

b.setPrice(10);

b.printInfo();

}

eg9();

We can use ? optional property also

price?:number; // change here // optional property

But better we should use

Price ! :number;