Jawahar Education Societys Annasaheb Chudaman Patil College of Engineering, Kharghar, Navi Mumbai

DOP: / /2023 **DOS: / /2023**

Experiment No: 2.3

Title: Access Modifiers example using TypeScript.

Theory:

- **Access Modifiers:**

Typescript allows us to use access modifiers at the class level. It gives direct access control to the class member. These class members are functions and properties. We can use class members inside its own class, anywhere outside the class, or within its child or derived class.

The access modifier increases the security of the class members and prevents them from invalid use. We can also use it to control the visibility of data members of a class. If the class does not have to be set any access modifier, TypeScript automatically sets public access modifier to all class members.

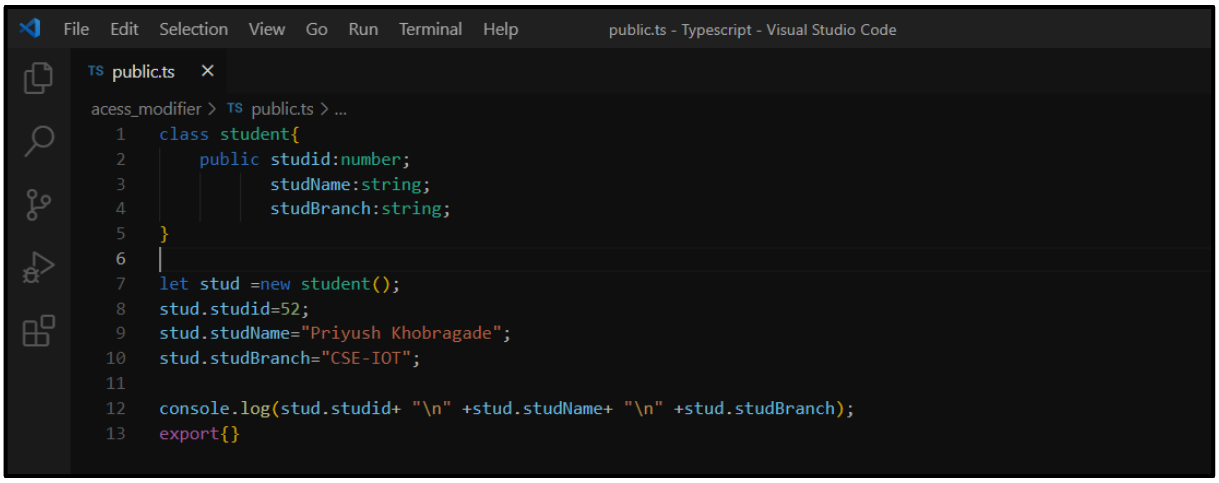
The TypeScript access modifiers are of three types. These are:

1. Public
2. Private
3. Protected.

1.Public

In TypeScript by default, all the members (properties and methods) of a class are public. So, there is no need to prefix members with this keyword. We can access this data member anywhere without any restriction.

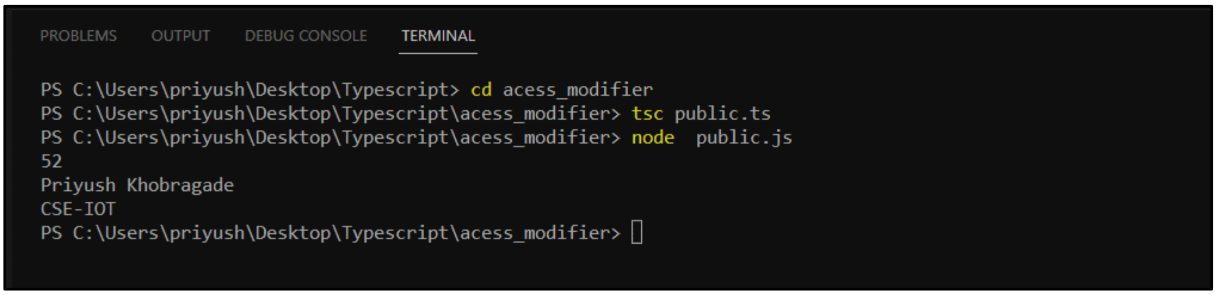
Input:



File Edit Selection View Go Run Terminal Help
public.ts - Typescript - Visual Studio Code
TS public.ts ×
acess_modifier > TS public.ts > ...

```
class student{
    public studid:number;
           studName:string;
           studBranch:string;
}

let stud =new student();
stud.studid=52;
stud.studName="Priyush Khobragade";
stud.studBranch="CSE-IOT";

console.log(stud.studid+ "\n" +stud.studName+ "\n" +stud.studBranch);
export{}
```

Output:

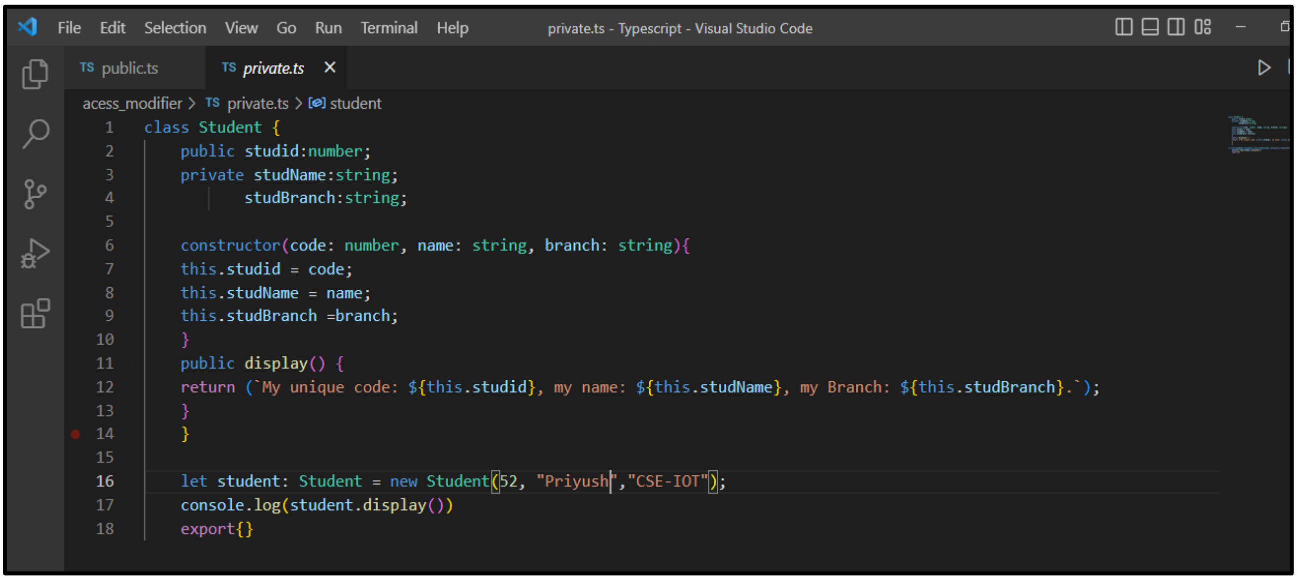


```
PS C:\Users\priyush\Desktop\Typescript> cd acess_modifier
PS C:\Users\priyush\Desktop\Typescript\acess_modifier> tsc public.ts
PS C:\Users\priyush\Desktop\Typescript\acess_modifier> node  public.js
52
Priyush Khobragade
CSE-IOT
PS C:\Users\priyush\Desktop\Typescript\acess_modifier> 
```

Private

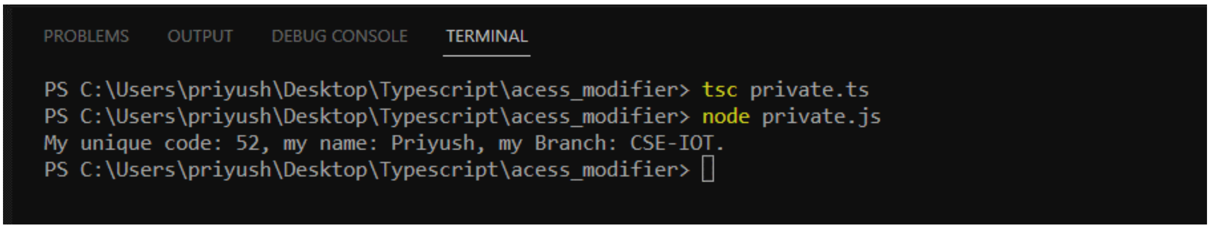
The private access modifier cannot be accessible outside of its containing class. It ensures that the class members are visible only to that class in which it is containing.

Input:


File
Edit
Selection
View
Go
Run
Terminal
Help
private.ts - Typescript - Visual Studio Code
TS public.ts
TS private.ts
acess_modifier > TS private.ts > student

```
class Student {
    public studid:number;
    private studName:string;
        studBranch:string;

    constructor(code: number, name: string, branch: string){
    this.studid = code;
    this.studName = name;
    this.studBranch =branch;
    }
    public display() {
    return (`My unique code: ${this.studid}, my name: ${this.studName}, my Branch: ${this.studBranch}.`);
    }
    }

    let student: Student = new Student(52, "Priyush","CSE-IOT");
    console.log(student.display())
    export{}
```

Output:

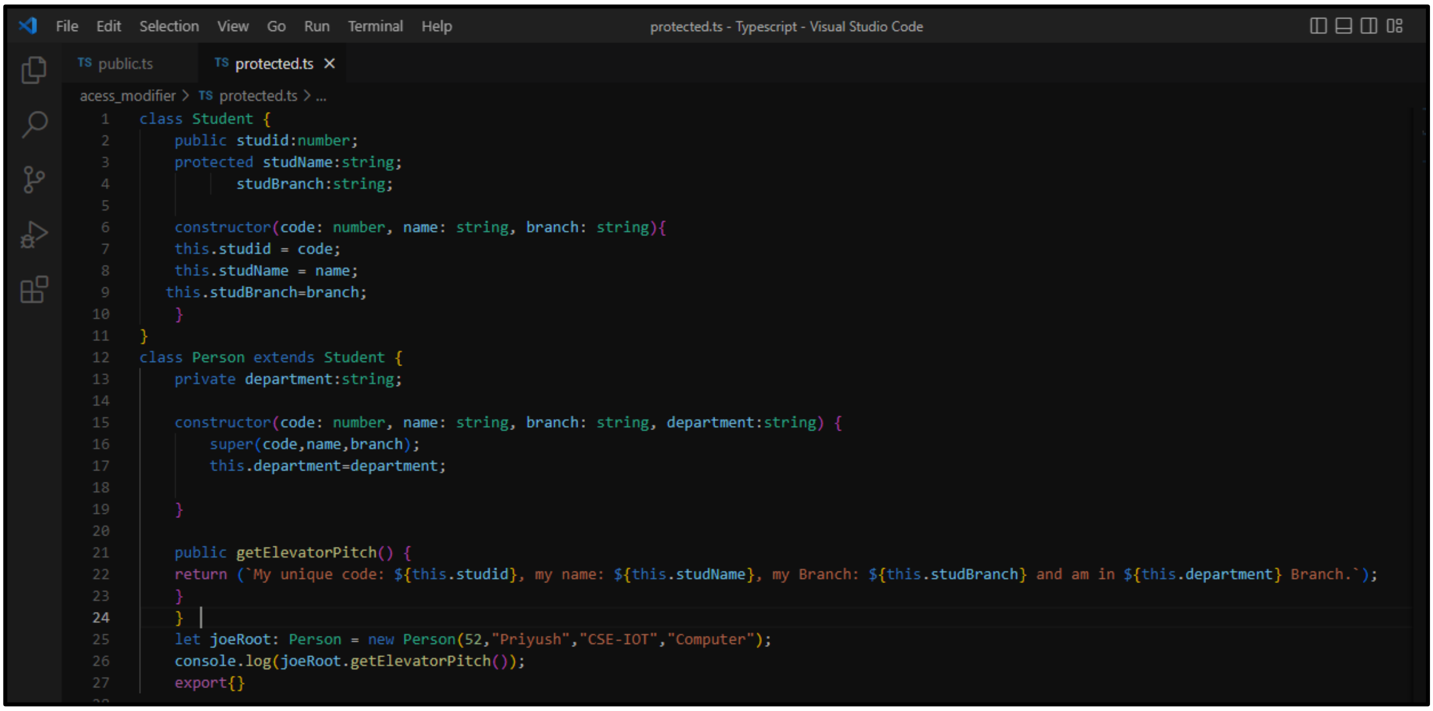


```
PS C:\Users\priyush\Desktop\Typescript\acess_modifier> tsc private.ts
PS C:\Users\priyush\Desktop\Typescript\acess_modifier> node private.js
My unique code: 52, my name: Priyush, my Branch: CSE-IOT.
PS C:\Users\priyush\Desktop\Typescript\acess_modifier> 
```

Protected:

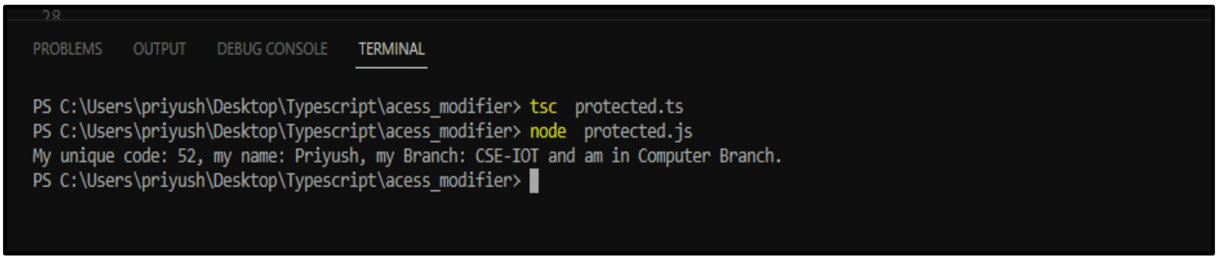
A Protected access modifier can be accessed only within the class and its subclass. We cannot access it from the outside of a class in which it is containing.

Input:


File
Edit
Selection
View
Go
Run
Terminal
Help
protected.ts - Typescript - Visual Studio Code
TS public.ts
TS protected.ts
acess_modifier > TS protected.ts > ...

```
class Student {
    public studid:number;
    protected studName:string;
        studBranch:string;

    constructor(code: number, name: string, branch: string){
    this.studid = code;
    this.studName = name;
   this.studBranch=branch;
    }
}
class Person extends Student {
    private department:string;

    constructor(code: number, name: string, branch: string, department:string) {
        super(code,name,branch);
        this.department=department;

    }

    public getElevatorPitch() {
    return (`My unique code: ${this.studid}, my name: ${this.studName}, my Branch: ${this.studBranch} and am in ${this.department} Branch.`);
    }
    }
    let joeRoot: Person = new Person(52,"Priyush","CSE-IOT","Computer");
    console.log(joeRoot.getElevatorPitch());
    export{}
```

Output:



```
PS C:\Users\priyush\Desktop\Typescript\acess_modifier> tsc  protected.ts
PS C:\Users\priyush\Desktop\Typescript\acess_modifier> node  protected.js
My unique code: 52, my name: Priyush, my Branch: CSE-IOT and am in Computer Branch.
PS C:\Users\priyush\Desktop\Typescript\acess_modifier>
```

Conclusion: -

Thus, we have done Access Modifiers example using TypeScript.