TypeScript - Classes

- In object-oriented programming languages like Java and C#. classes are the fundamental entities used to create reusable components.
- In terms of OOPs, a class is a **template** or **blueprint** for creating objects.

A class can contain the following properties:

- **Fields:** It is a variable declared in a class.
- **Methods:** It represents an action for the object.
- **Constructors:** It is responsible for initializing the object in memory.
- **Nested class and interface:** It means a class can contain another class.

- ➔ TypeScript is an Object-Oriented JavaScript language, so it supports object-oriented programming features like classes, interfaces, polymorphism, data-binding, etc.
- ➔ JavaScript **ES5** or **earlier versions** did not support classes.
- ➔ TypeScript supports this feature from **ES6** and **later versions**.
- ➔ TypeScript has **built-in** support for using classes because it is based on the ES6 version of JavaScript.
- ➔ Today, many developers use class-based object-oriented programming languages and compile them into JavaScript, which works across all major browsers and platforms.

Syntax to declare a class

A class keyword is used to declare a class in TypeScript.

```
class <class_name>{
   field;
   method;
}
```

JavaScript

```
class Employee {
    empCode: number;
    empName: string;

    constructor(code: number, name: string) {
            this.empName = name;
            this.empCode = code;
    }

    getSalary() : number {
        return 10000;
    }
}
```

Constructor

- The constructor is a special type of method which is called when creating an object.
- In TypeScript, the constructor method is always defined with the name "constructor".

JavaScript

```
class Employee {

    empCode: number;
    empName: string;

    constructor(empcode: number, name: string ) {
        this.empCode = empcode;
        this.name = name;
    }
}
```

- In the above example, the `Employee` class includes a constructor with the parameters `empcode` and `name`. In the constructor, members of the class can be accessed using `this` keyword e.g. `this.empCode` or `this.name`.
- It is not necessary for a class to have a constructor.

Creating an Object of Class

- An object of the class can be created using the new keyword.

```JavaScript
class Employee {
    empCode: number;
    empName: string;
}

let emp = new Employee();
```

- Here, we create an object called emp of type Employee using let emp = new Employee();.
- The above class does not include any parameterized constructor so we cannot pass values while creating an object.
- If the class includes a parameterized constructor, then we can pass the values while creating the object.