es, Optional Properties, Static Properties and Methods in TypeScript

Class

- A class is a blueprint for creating objects with properties and methods.

Example:

```
class Person {
  name: string;
  constructor(name: string) {
    this.name = name;
  }
 greet() {
    console.log(`Hello, ${this.name}`);
  }
}
const p1 = new Person("Pavan");
p1.greet(); // Output: Hello, Pavan
```

Read-only Properties

- Read-only properties cannot be changed after initialization.
- Use the **readonly** keyword.
- Useful for constants or values that shouldn't be modified.

Example:

```
class Car {
  readonly brand: string;
  constructor(brand: string) {
    this.brand = brand;
  }
}
```

```
const car = new Car("Toyota");

// car.brand = "Honda"; // ❌ Error: Cannot assign to 'brand' because it is a read-only property.
```

Optional Properties

- Optional properties are not required when creating an object.
- Defined using the **?** symbol.
- Useful when a property is not always needed.

Example:

```
class Student {
  name: string;
  age?: number;

  constructor(name: string, age?: number) {
    this.name = name;
    if (age) this.age = age;
  }
}

const s1 = new Student("Pavan");
const s2 = new Student("Kiran", 22);
```

Static Properties and Methods

- Static members belong to the class, not the instances.
- Accessed using the class name, not through objects.
- Useful for utility methods or shared data.
- **Example:**

```
class MathUtils {
  static PI = 3.14;
  static square(num: number): number {
    return num * num;
  }
}
console.log(MathUtils.PI); // 3.14
console.log(MathUtils.square(5)); // 25
```