Inheritance, super(), Overriding, and Access Modifiers

Inheritance in TypeScript

- Inheritance allows a **class (child)** to acquire properties and methods from another **class (parent)**.
- Use the **extends** keyword.

```
class Animal {
  eat() {
    console.log("Animal eats food");
  }
}

class Dog extends Animal {
  bark() {
    console.log("Dog barks");
  }
}

const d = new Dog();
d.eat(); // Inherited from Animal
d.bark(); // Own method
```

super()

- **super()** is used in the **child class constructor** to call the **parent class constructor**.

```
class Animal {
  constructor(public name: string) {}
}

class Dog extends Animal {
  constructor(name: string, public breed: string) {
```

```
    super(name); // Call parent constructor
  }
}
```

Method Overriding

- A **child class can redefine (override)** a method from the parent class.
- The method in the child class should have the **same name and signature**.

```
class Animal {
  makeSound() {
    console.log("Animal makes sound");
  }
}

class Dog extends Animal {
  makeSound() {
    console.log("Dog barks");
  }
}

const dog = new Dog();
dog.makeSound(); // Output: Dog barks
```

Access Modifiers

Used to control the visibility of class members (properties and methods):

Modifier	Description
public	Accessible from anywhere (default)
private	Accessible only within the same class
protected	Accessible in the class and its subclasses

```
class Person {
  public name: string;
  private age: number;
  protected city: string;

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

class Student extends Person {
  showCity() {
    console.log(this.city); // ✅ allowed (protected)
    // console.log(this.age); // ❌ Error (private)
  }
}
```