An instance member function declaration declares a property in the class instance type and assigns a function object to a property on the prototype object of the class. In the body of an instance member function declaration, `this` is of the class instance type.

A static member function declaration declares a property in the constructor function type and assigns a function object to a property on the constructor function object. In the body of a static member function declaration, the type of `this` is the constructor function type.

A member function can access overridden base class members using a super property access (section 4.8.2). For example

```
class Point {
    constructor(public x: number, public y: number) { }
    public toString() {
        return "x=" + this.x + " y=" + this.y;
    }
}

class ColoredPoint extends Point {
    constructor(x: number, y: number, public color: string) {
        super(x, y);
    }
    public toString() {
        return super.toString() + " color=" + this.color;
    }
}
```

In a static member function, `this` represents the constructor function object on which the static member function was invoked. Thus, a call to 'new this()' may actually invoke a derived class constructor:

```
class A {
    a = 1;
    static create() {
        return new this();
    }
}

class B extends A {
    b = 2;
}

var x = A.create();  // new A()
var y = B.create();  // new B()
```

Note that TypeScript doesn't require or verify that derived constructor functions are subtypes of base constructor functions. In other words, changing the declaration of 'B' to