13、Class 的基本语法

- ES6 提供了更接近传统语言的写法，引入了 Class（类）这个概念，作为对象的模板。通过class关键字，可以定义类。
- 基本上，ES6 的class可以看作只是一个语法糖，它的绝大部分功能，ES5 都可以做到，
- 新的class写法只是让对象原型的写法更加清晰、更像面向对象编程的语法而已。

```
class Point {
  constructor(x, y) {
    this.x = x;
    this.y = y;
  }

  toString() {
    return '(' + this.x + ', ' + this.y + ')';
  }
}
```

- constructor方法是类的默认方法，通过new命令生成对象实例时，自动调用该方法。

```
//定义类
class Point {

  constructor(x, y) {
    this.x = x;
    this.y = y;
  }

  toString() {
    return '(' + this.x + ', ' + this.y + ')';
  }

}

var point = new Point(2, 3);

point.toString() // (2, 3)
```

- Class 表达式
- 与函数一样，类也可以使用表达式的形式定义。

```
const MyClass = class Me {
  getClassName() {
    return Me.name;
  }
};
```

- 上面代码使用表达式定义了一个类。需要注意的是，这个类的名字是Me，
- 但是Me只在 Class 的内部可用，指代当前类。在 Class 外部，这个类只能用MyClass引用。

```
let inst = new MyClass();
inst.getClassName() // Me
Me.name // ReferenceError: Me is not defined
```

- 静态方法
- 类相当于实例的原型，所有在类中定义的方法，都会被实例继承。
- 如果在一个方法前，加上static关键字，就表示该方法不会被实例继承，而是直接通过类来调用，这就称为“静态方法”。

```
class Foo {
  static classMethod() {
    return 'hello';
  }
}

Foo.classMethod() // 'hello'

var foo = new Foo();
foo.classMethod()
// TypeError: foo.classMethod is not a function
```