15.Symbol 类型和属性

学习要点：

1.Symbol 类型

2.Symbol 属性

本节课我们来开始学习 ES6 新增的 Symbol 类型和作为属性的方法。

一. Symbol 类型

1. ES6 之前基础数据类型有：字符串、数值、布尔、对象、null 和 undefined；
2. ES6 新增了一种叫做 Symbol 的基础数据类型，表示独一无二的值，类似 ID；
3. 创建 Symobl 通过函数 Symbol()完整，可以传递参数，也可以为空；

```
//注意，不支持 new Symbol()
let s = Symbol();
console.log(s);         //输出 Symbol()
console.log(typeof s);  //输出 symbol 类型
```

4. 在 Symbol()函数中参数，是对变量的描述，程序无法访问，只能日志打印；

```
//为了输出测试时，便于区分
Symbol('s!');
```

5. 创建两个 Symobel 类型的变量，来验证他们独一无二的特性；

```
//有没有参数，或参数是否相同都不恒等
//因为 Symbol 是唯一存在的
let s1 = Symbol(),
    s2 = Symbol();
console.log(s1 === s2);
```

6. Symobol 类型变量无法进行隐式转换，需要提前显示转换匹配的类型；

```
//Symbol 类型无法隐式转换，可显式
let s = Symbol();
console.log(s.toString() + '变量');
console.log(String(s) + '变量');
console.log(!s);        //布尔值
```

二. Symbol 属性

1. Symbol 类型有哪些应用场景？解决了哪些问题？最常用的一种就是对象属性；
2. 由于 Symbol 类型是独一无二的值，作为对象属性就具有唯一性不出现重名；
3. 对于多模块、多人开发或者拼装属性名的情况下，有可能会出现属性名重复；
4. 首先，先故意设置一个相同的对象属性名，看看会出现什么问题？

```
//重名的属性名不报错，被覆盖
let obj = {
    name : 'Mr.Lee',
    name : 'Mr.Wang'
};

console.log(obj);

//拼装的属性名，也被覆盖
let x = 'Name',
    y = 'Name';

let obj = {
    ['user' + x] : 'Mr.Lee',
    ['user' + y] : 'Mr.Wang'
};

console.log(obj);
```

5. 那么，Symbol 作为对象属性名，该如何使用呢？具体如下：

```
let name = Symbol('name');
let obj = {
    [name] : 'Mr.Lee'
};
console.log(obj);       //结果：{ [Symbol(name)]: 'Mr.Lee' }
```

6. 强调：上面的例子中，属性名不是 name，而是[Symbol(name)]；
7. 其中：参数 name ,要不要都无所谓，主要是为了看上去清晰；

```
let x = Symbol('name');
let y = Symbol('name');
let obj = {
    [x] : 'Mr.Lee',
    [y] : 'Mr.Wang'
};
//结果：{ [Symbol(name)]: 'Mr.Lee', [Symbol(name)]: 'Mr.Wang' }
console.log(obj);
```

8. 方法名也可以使用 Symbol 类型；

```
let fn = Symbol('fn');
let obj = {
    [fn]() {
        return 'fn';
    }
};
```

```
        console.log(obj);
        console.log(obj[fn]());
```