## ES6学习

函数的扩展

1. 与解构赋值默认值结合使用。
2. 参数默认值的位置：**通常情况下，定义了默认值的参数，应该是函数的尾参数。因为这样比较容易看出来，到底省略了哪些参数。如果非尾部的参数设置默认值，实际上这个参数是没法省略的。**
3. 函数的length属性：**指定了默认值以后，函数的length属性，将返回没有指定默认值的参数个数。也就是说，指定了默认值后，length属性将失真。**
4. 作用域：**一旦设置了参数的默认值，函数进行声明初始化时，参数会形成一个单独的作用域（context）。等到初始化结束，这个作用域就会消失。这种语法行为，在不设置参数默认值时，是不会出现的**。
5. Name属性：**函数的name属性，返回该函数的函数名。需要注意的是，ES6 对这个属性的行为做出了一些修改。如果将一个匿名函数赋值给一个变量，ES5 的name属性，会返回空字符串，而 ES6 的name属性会返回实际的函数名。**
6. 箭头函数：ES6 允许使用“箭头”（=>）定义函数。如果箭头函数不需要参数或需要多个参数，就使用一个圆括号代表参数部分。如果箭头函数的代码块部分多于一条语句，就要使用大括号将它们括起来，并且使用return语句返回。由于大括号被解释为代码块，所以如果箭头函数直接返回一个对象，必须在对象外面加上括号，否则会报错。

箭头函数有几个使用注意点。

（1）函数体内的this对象，就是定义时所在的对象，而不是使用时所在的对象。

（2）不可以当作构造函数，也就是说，不可以使用new命令，否则会抛出一个错误。

（3）不可以使用arguments对象，该对象在函数体内不存在。如果要用，可以用 rest 参数代替。

（4）不可以使用yield命令，因此箭头函数不能用作 Generator 函数。

上面四点中，第一点尤其值得注意。this对象的指向是可变的，但是在箭头函数中，它是固定的。

数组的扩展

含义：扩展运算符（spread）是三个点（...）。它好比 rest 参数的逆运算，将一个数组转为用逗号分隔的参数序列。

**扩展运算符与正常的函数参数可以结合使用，非常灵活。**

**扩展运算符后面还可以放置表达式。**

**如果扩展运算符后面是一个空数组，则不产生任何效果。**

**注意，只有函数调用时，扩展运算符才可以放在圆括号中，否则会报错。**

**由于扩展运算符可以展开数组，所以不再需要apply方法，将数组转为函数的参数了。**

运算符的应用：

1. 复制数组
2. 合并数组
3. 与解构赋值结合
4. 将字符串转换为真正的数组
5. 实习了Iterator借口的对象
6. Map和Set结构，Genertor函数。

Array.from()方法用于将两类对象转为真正的数组：类似数组的对象（array-like object）和可遍历（iterable）的对象（包括 ES6 新增的数据结构 Set 和 Map）。

Array.of()方法用于将一组值，转换为数组。这个方法的主要目的，是弥补数组构造函数Array()的不足。因为参数个数的不同，会导致Array()的行为有差异。Array.of基本上可以用来替代Array()或new Array()，并且不存在由于参数不同而导致的重载。它的行为非常统一。

copyWithin()方法数组实例的copyWithin()方法，在当前数组内部，将指定位置的成员复制到其他位置（会覆盖原有成员），然后返回当前数组。也就是说，使用这个方法，会修改当前数组。