6、运算符的扩展

- es6之扩展运算符 三个点（...）
  - 对象的扩展运算符
  - 数组的扩展运算符
  - 剩余参数的处理
- ① 对象的扩展运算符

```
//声明一个对象stu1，内有三个属性
let stu1 = {name:'李四',age:22,sex:'男'};
//使用三个点将stu1对象中的属性复制一份赋给stu2(独立的)
let stu2 = {...stu1};
stu1.age = 30;
console.log(stu2); //{name: "李四", age: 22, sex: "男"}
```

- ② 数组的扩展运算符

```
//声明一个数组，并赋初值，使用三个点实现数组的复制
let a = [10,20,30];
let b = [...a]; //复制数组a中的所有元素赋给b
a[2] = 300; //修改数组a中的一个元素值
console.log(b); //数组b没有影响 [10, 20, 30]

//使用三个点实现将数组转为参数序列。
function demo(x,y,z){
    console.log(x,y,z); //10,20,300
}
demo(...a); //使用三个点将数组转换为参数序列
```

- ③ 使用三个点(...)对剩余参数的处理

```
//在以前我们使用arguments来收集函数调用传递的任意数量参数。
function sum(){
    let res = 0;
    //遍历所有参数
    for(let v of arguments){
        res += v; //累加
    }
    return res;
}
console.log(sum(10,20,30,40)); //100 调用上面函数可以传递很多参数

//执行效果同时，使用三点来收集所有参数
function sum2(...numbers){
```

```
    let res = 0;
    for(let v of numbers){
        res += v;
    }
    return res;
}
console.log(sum2(10,20,30,40)); //100
```

- 剩余参数的处理实例

```
//定义一个处理折扣的函数，第一个参数为折率，其余参数为价格
function discount(rate,...prices){
    return prices.map(p => p*rate);
}
//计算0.68参数以外的其他价格，打完6.8折后的值。
console.log(discount(0.68,125,98,246,50));//[85, 66.64, 167.28, 34]


//有一条信息，内容为姓名、编号和数据
const info = ["zhangsan","1002",12,23,45,34];
//使用解构赋值
const [name,id,...params] = info;
console.log(name,id,params); //zhangsan  1002  [12, 23, 45, 34]
```

- 实例：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>扩展运算符实例</title>
    <style>
        body{background-color: darkorange;}
        #hid{color:white;font-size:80px;text-align:center;}
        #hid span{
            margin:5px;
            cursor:pointer;
            display:inline-block;
            transsition:transform 0.25s;
        }
        /*定义鼠标移入的动画效果*/
        #hid span:hover{
            transform:translateY(-20px) rotate(10deg) scale(2);
        }
    </style>
</head>
<body>
```

```
    <h1 id="hid">Hello ES6!</h1>

    <script>
        //获取页面中h1元素节点
        const hid = document.querySelector("#hid");
        //获取节点中间的内容，作为参数调用下面函数，并将返回结果替换原先h1标签之间的内容中
        hid.innerHTML = wrapWithSpan(hid.textContent);
        //自定义函数
        function wrapWithSpan(str){
            //将内容使用map遍历，并未每个字符外添加span标签后合并返回。
            return [...str].map(v=>`<span>${v}</span>`).join("");
        }
    </script>
</body>
</html>
```