2017-12-03ES6的几个使用的小技巧

ES6 的几个实用的小技巧

1 . 交换元素

利用数组解构来实现交换元素

```
let a = 'world' , b = 'Hello';
[a,b] = [b,a];
console.log(a)       // Hello
console.log(b)       //world
```

2 . 单条语句

ES6操作数组的语句更加紧凑

```
//寻找数组中最大的元素
const max = (arr) => Math.max(...arr);
max([1,2,4,7,85,210])        //210
//计算数组的总和
const sum = (arr) => arr.reduce((a,b) => (a + b) ,0);
sum([1,2,3,4,5,6,7,8,9,10])      //55
```

3 . 数组拼接

展开运算符代替concat

```
let arr1 = [1,2,3];
let arr2 = [4,5,6];
let arr3 = [7,8,9];
console.log([...arr1,...arr2,...arr3])       //[1,2,3,4,5,6,7,8,9];
```

4 . 制作副本

我们很容易实现数组和对象的浅拷贝

```
let obj = {...oldObj};
let arr = [...oldArr];
```

5 . 类与继承（源码详见源码文件夹里面的类与继承）

JavaScript 中的类继承是基于原型链和构造函数的，ES6 没有退出 class 类的时候，我们是这样写的：

```
    function Person(name){
        this.name = name;
    }
```

```
    Person.prototype.Hello = function(){
        alert('Hello' + this.name);
    }
    var p = new Person('小明');
```

如果用 ES6 的语法 class 来写，我们可以这样写：

```
    class Person{
        constructor(name){
            this.name = name;
        }
        Hello(){
            alert('name' + this.name);
        }
    }
```

比较一下就发现，class 的定义包含了构造函数 constructor 和定义在原型对象上的函数 Hello()(没有关键字 function)，这样就避免了 Person.prototype.Hello = function(){} 这样分散的代码；

class 继承：用 class 定义对象的一个好处就是继承更加方便了，想一想我们从 Person 派生一个 Student 需要编写的代码量。现在，原型继承的中间对象，原型对象的构造函数等等都不需要考虑了，直接通过 extends 来实现：

```
class Student extends Person{
    constructor(name,age){
        super(name);                //super 关键字可以直接调用父类的构造方法
        this.age = age;
    }
    Say(){
        alert('name:' + this.name + 'age' + this.age);
    }
}
```