●

P10, 1.3例子

<script>

        function Checking (amount) {

            this.balance = amount

            this.deposit = deposit

            this.withdraw = withdraw

            this.toString = toString

        }

        function deposit (amount) {

            this.balance += amount

        }

        function withdraw (amount) {

            if (amount <= this.balance) {

                this.balance -= amount

            } else if (amount > this.balance) {

                console.log('Insufficient funds')

            }

        }

        function toString () {

            return 'Balance:' + this.balance

        }

        var account = new Checking(500)

        account.deposit(1000)

        console.log(account.toString())

        account.withdraw(750)

        console.log(account.toString())

        account.withdraw(800)

        console.log(account.toString())

    </script>

█数组

●p16, 2.2.4 数组的浅复制和深复制

<script>

        // 浅复制的例子

        var ce1 = [1, 3, 5]

        var ce2 = ce1

        // 由于将ce2的值只是对ce1的引用，是浅复制，所以改变ce2的值，同时会影响ce1

        ce2[0] = 7

        console.log(ce1) //[7, 3, 5]

        console.log(ce2) //[7, 3, 5]

        // 深复制的例子

        // 用于深复制数组的函数

        function copy (arr1, arr2) {

            for (var i = 0; i < arr1.length; ++i) {

                arr2[i] = arr1[i]

            }

        }

        var a = [1, 3, 5, 7, 9]

        var b = []

        copy(a, b)

        console.log(b) //[1, 3, 5, 7, 9]

        // 深复制后，修改b不会影响a

        b[0] = 99

        console.log(a) //[1, 3, 5, 7, 9]

        console.log(b) //[99, 3, 5, 7, 9]

    </script>

●p18, 2.3.2 数组转字符串

<script>

        // 数组转字符串

        var a = [1, 3, 'haha', 5, 'jojo']

        // join()方法如果不输入参数，默认是以逗号分隔，所以效果和toString()方法一样，如下：

        console.log(a.join()) //1,3,haha,5,jojo

        console.log(a.toString()) //1,3,haha,5,jojo

        // join()方法可以输入一个参数，则以这个参数分割

        console.log(a.join('')) //13haha5jojo

        console.log(a.join('\*')) //1\*3\*haha\*5\*jojo

    </script>

* P21, 2.4.4 排序sort

Sort默认是对字符串排序，根据数字，字母，汉字来排序

<script>

        var a = [100, 'ab', 3, 0, 2, 'bc', '我', '啊', '祖国', 'c', 1000]

        // sort默认是字符串排序，即数字到字母到汉字拼音顺序

        console.log(a.sort()) //[0, 100, 1000, 2, 3, "ab", "bc", "c", "啊", "我", "祖国"]

    </script>

如果要给数字安装大小排序，则需要给sort传一个参数

<script>

        // 如果需要从数学意义上对数字的大小进行排序，就要给sort传参，参数是以下函数

        // 以下两种函数写法都可以

        // function compare (num1, num2) {

        //  return num1 - num2

        // }

        function compare(value1, value2) {

            if (value1 < value2) {

                return -1;

            } else if (value1 > value2) {

                return 1;

            } else {

                return 0;

            }

        }

        var a = [100, 3, 0, -1, 2, -88, 1000]

        console.log(a.sort(compare)) //[-88, -1, 0, 2, 3, 100, 1000]

    </script>

●p22, 2.5 迭代器

forEach的应用

<script>

        var a = [1, 3, 'jojo', 5, 'haha', 7, 9]

        var b = []

        function fe (n) {

            if (typeof n === "number") {

                console.log(n + 1)

                b.push (n + 1)

            } else if (typeof n === "string") {

                console.log(n + 'oo')

                b.push (n + 'oo')

            }

        }

        a.forEach(fe)

        console.log(b) //[2, 4, "jojooo", 6, "hahaoo", 8, 10]

    </script>

Every 和some的应用

<script>

        // 判断数组里是否全部是数字

        var a = [1, 3, 5, '', 8]

        function e (n) {

            return typeof n === 'number'

        }

        console.log(a.every(e)) //false

        function s (n) {

            return typeof n === 'string'

        }

        console.log(a.some(s)) //true

    </script>

Reduce应用

<script>

        var a = [1, 3, 5, 7, 9]

        function rd (total, current) {

            return current + total

        }

        console.log(a.reduce(rd)) //25

    </script>