JavaScript中的数组是动态地，即长度是可以发生变化的。





1. 创建数组对象（new Array()）

创建一个数组对象，3种方式：

new Array(); 创建长度为0的数组

new Array(5); 创建长度为5的数组，但是其每一个元素都是undefined

new Array(3,1,4,1,5,9,2,6); 根据参数创建数组

<script>

function p(s, v) {

document.write(s + " " + v);

document.write("<br />");

}

var x = new Array();

p("创建一个空数组：", x);

x = new Array(5);

p("创建一个长度是5的数组：", x);

p("像new Array(5)这样没有赋初值的方式创建的数组，每个元素的值都是：", x[0]);

x = new Array(3, 1, 4, 1, 5, 9, 2, 6);

p("创建有初值的数组：", x);

</script>

1. 数组长度（length）

属性length获取一个数组的长度。

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

var x = new Array(3, 1, 4, 1, 5, 9, 2, 6);

p("当前数组是：" + x);

p("通过.length获取当前数组的长度：" + x.length);

</script>

1. 遍历一个数组（for循环、增强for in循环）

遍历有两种方式：

结合for循环使用，通过下标遍历；

使用增强for in循环遍历。

注意：在增强for in中，i是下标的意思。

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

var x = new Array(3, 1, 4);

p("当前数组是：" + x);

p("使用普通的for循环遍历数组");

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

p(x[i]);

}

p("使用增强for循环遍历数组");

for(i in x) {

p(x[i]);

}

</script>

1. 连接数组（concat）

方法concat连接两个数组。

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

var x = new Array(3, 1, 4);

var y = new Array(1, 5, 9, 2, 6);

p("数组x是：" + x);

p("数组y是：" + y);

var z = x.concat(y);

p("使用concat连接数组x和y");

p("数组z是：" + z);

</script>

1. 通过指定分隔符，返回一个数组的字符串表达（join）

方法join通过指定分隔符，返回一个数组的字符串表达。

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

var x = new Array(3, 1, 4);

p("数组x是：" + x);

var y = x.join();

p("y是数组x的字符串表达，其值是" + y + "其类型是：" + (typeof y));

var z = x.join("@");

p("z是x的字符串表达，不过分隔符不是默认的，而是@：" + z);

</script>

1. 分别在最后的位置插入数据和获取数据（push、pop）

方法push、pop分别在最后的位置插入数据和获取数据（获取后删除）

就像先入后出的栈一样。

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

var x = new Array(3, 1, 4);

p("数组x是：" + x);

x.push(5);

p("向x中push 5，得到" + x);

var e = x.pop();

p("从x中pop一个值出来，其值是：" + e);

p("pop之后，x数组的值是：" + x);

</script>

1. 分别在最开始的位置插入数据和获取数据（unshift、shift）

方法unshift，shift，分别在最开始的位置插入数据和获取数据（获取后删除）

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

var x = new Array(3, 1, 4);

p("数组x是：" + x);

x.unshift(5);

p("对数组unshift(5)，最前面加，数组变为：" + x);

var e = x.shift();

p("从数组中shift一个数(从最前面取)，其值是：" + e);

p("shift之后，数组变为：" + x);

</script>

1. 对数组的内容进行排序（sort）

方法sort对数组的内容进行排序。正排序，从小到大。

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

var x = new Array(3, 1, 4, 1, 5, 9, 2, 6);

p("数组x是：" + x);

x.sort();

p("使用sort排序后的数组x是：" + x);

</script>

1. 自定义排序算法（sort(comparator)）

sort()默认采用正排序，即小的数排在前面。如果需要采用自定义排序的算法，就把比较器函数作为参数传递给sort()。

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

//比较器函数

function comparator(v1, v2) {

//v2-v1表示大的放前面，小的放后面。

return v2 - v1;

}

var x = new Array(3, 1, 4, 1, 5, 9, 2, 6);

p("数组x是：" + x);

x.sort(comparator);

p("使用sort进行自定义倒排序后的数组x是：" + x);

</script>

1. 对数组的内容进行反转（reverse）

方法reverse，对数组的内容进行反转。

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

var x = new Array(3, 1, 4, 1, 5, 9, 2, 6);

p("数组x是：" + x);

x.reverse();

p("使用reverse函数进行反转后的值是：" + x);

</script>

1. 获取子数组（slice）

方法slice获取子数组

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

var x = new Array(3, 1, 4, 1, 5, 9, 2, 6);

p("数组x是：" + x);

var y = x.slice(1);

p("x.slice(1)获取的子数组是：" + y);

var z = x.slice(1, 3);

p("x.slice(1,3)获取的子数组是：" + z);

p("第二个参数获取不到");

</script>

1. 删除和插入元素（splice）

方法splice用于删除数组中的元素，还能用于向数组中插入元素。

<script>

function p(s) {

document.write(s);

document.write("<br />");

}

var x = new Array(3, 1, 4, 1, 5, 9, 2, 6);

p("数组x是：" + x);

x.splice(3, 2);

p("x.splice(3,2)表示从位置3开始，删除2个元素：" + x);

x.splice(3, 0, 1, 5);

p("x.splice(3,0,1,5)从位置3开始，删除0个元素，但是插入1和5，最后得到：" + x);

</script>