1. 变量
   1. 变量是什么：容器，用于存储数据。
   2. 变量的作用：用于存储数据，让数据可以复用。
   3. 3 变量的使用方式：
   4. 3.1 变量的声明方式：变量必须声明后再使用 var num;
   5. 变量的赋值操作：
   6. var num;
   7. num = 200;
   8. console.log(num);
   9. num = 300;
   10. console.log(num);
2. 数据类型：

String number Boolean null undefind object(复杂数据类型)

3 . 数据类型转换

String : String() .tostring() +(+号是隐式转换)

特殊点：null和undefined是没有toString这个功能的

Number: Number() parseInt() parseFloat()

Boolean: Boolean()

1. 命名规范

规则 - 必须遵守的，不遵守会报错

* 1. 由字母、数字、下划线、$符号组成，不能以数字开头
  2. 不能是关键字和保留字，例如：for、while。
  3. 区分大小写

规范 - 建议遵守的，不遵守不会报错

* 1. 变量名必须有意义
  2. 遵守驼峰命名法。首字母小写，后面单词的首字母需要大写。例如：userName、userPassword

4 操作符

逻辑运算符 & || ！

关系运算符 < > >= <= == != === !==

算数运算符 + - \* / %

赋值运算符 = += -= \*= /= %=

一元运算符 ++ --

5 循环和判断

1. If….else ……语句

if (/\* 条件表达式 \*/){

// 成立执行语句

} else {

// 否则执行语句

}

if (/\* 条件1 \*/){

// 成立执行语句

} else if (/\* 条件2 \*/){

// 成立执行语句

} else if (/\* 条件3 \*/){

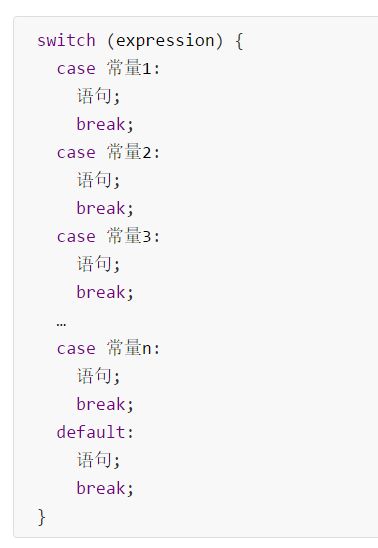
// 成立执行语句

} else {

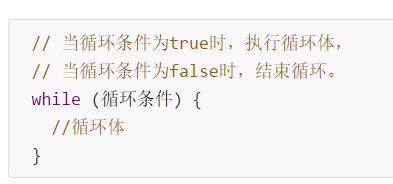
// 最后默认执行语句

}

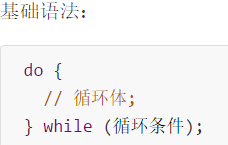
### switch语句



3 while语句

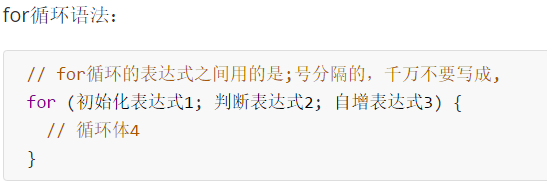


4 do …. While…



do…while 至少执行一次！

### 5 for语句



执行顺序：

初始化表达式

判断表达式

自增表达式

循环体

### 6 continue和break

break:立即跳出整个循环，即循环结束，开始执行循环后面的内容（直接跳到大括号）

continue:立即跳出当前循环，继续下一次循环（跳到i++的地方）

7. 数组

1）数组的作用 ： 用于同时保存多个数据

2）数组的声明方式： var arr=[1,2,3,4,5];

Var arr= new Array(1,2,3,4,5)

3）数组的长度 数组名.length;

4）索引（下标）

数组中的每个元素按照索引值排列，索引从0开始，依次递增为字符串类型

5）基本数据类型和复杂数据类型的区别：

// 1 保存的数据个数不同

// 2 在内存中的保存方式不同：

// - 基本数据类型在内存单元中保存的是具体值。

// - 复杂数据类型在内存单元中保存的是具体指的指针(引用)。

/\*var num = 100;

var arr = [1, 2, 3];\*/

// 3 复制操作后的结果不同：

// 复制操作实际上是复制内存单元中保存的那个东西，

// - 基本数据类型复制前后，两个值没有关联

/\*var num = 100;

var num2 = num;

console.log(num, num2); // 100 100

num++;

console.log(num, num2); // 101 100\*/

// - 复杂数据类型复制前后，两个值实际上是同一个值

/\*var arr = [1, 2, 3];

var arr3 = arr;

console.log(arr, arr3); //[1, 2, 3] [1, 2, 3]

arr[0] = 100;

console.log(arr, arr3); //[100, 2, 3] [100, 2, 3]

\*/

数组练习：数组去重和冒泡排序要求必须掌握；

8 函数

1. 作用：代码重用（让代码可以复用，提高代码利用率）
2. 声明方式：
3. function 函数名（）{

函数体

}

1. 匿名函数；var fun = function(){};
2. 立即执行函数（自调用函数）

效果：再书写位置执行一次

作用：进行作用域的包裹，减少全局变量的使用。

使用位置：再每个功能外，使用立即执行函数包裹即可

注意点：立即执行函数后必须添加分号；

（function （）{

Var num=100;

Console.log（“这是一个立即执行函数”）

}）（）；

1. 回调函数（将函数作为参数使用）

Function fun（num,arr,fn）{

Console.Log(num,arr);

Fn();

}

Function fun2（）{

Console.Log(‘这是会被传入到fun中的函数功能代码’);

}

Fun(100,[1,2,3],fun2);

1. 递归函数：函数内调用当前函数

Function fun（）{

Fun（）；

}

Fun（）；

1. 函数的使用：直接调用函数名，如 sum();
2. 函数的参数：形参和实参
   1. 形参：写在函数的小括号中，用于占位置，多个参数使用逗号隔开.例如：sum(a,b)
   2. 实参：写在函数调用时的小括号中，调用中会把实参传给形参。例如：sum(1,2)
3. 函数的返回值

原因：在操作函数时发现我们无法操作函数内部的变量，得不到函数计算结果，所以需要函数帮我们把值返回来，我们来接收

注意：return还具备结束函数功能，当写return后，return后面的代码不再执行。

9 作用域

1. 全局作用域 不在函数内（所有区域）
2. 局部作用域 任意函数内的区域（局部区域）
3. 全局变量 在全局作用域声明的变量，在任何地方皆可访问。
4. 局部变量 在局部作用域声明的变量，只能在局部区域访问

10 域解析

一句话概括：

会把变量的声明（不包括变量的声明）和函数提升到作用域顶端，其他的依次排列

var a=10;

function aaa(){

alert(a);

};

function bbb(){

var a=20;

aaa();

}

bbb(); //结果为10，因为aaa()函数不能访问到bbb()里面的局部变量，所以访问到的是a=10,这个全局变量。

var a=10;

function aaa(){

alert(a);

};

aaa();

function aaa(){

var a=10;

};

aaa();

alert(a);

function aaa(){

a=10;

}

aaa();

alert(a); //结果为10;

//等价于：

var a;

function aaa(){

a=10;

};

aaa();

alert(a);

function aaa(){

var a=b=10;

}  
 aaa();  
 alert(a);//结果为,无法访问到  
 alert(b);

function aaa(){

alert(a);

var a=20;

}

aaa(); //结果为：undefined

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

var a=10;

function aaa(){

alert(a);

var a=20;

}

aaa(); //结果为：undefined

可以解析为是:

var a=10;

function aaa(){

var a; //声明提前了

alert(a);

a=20; //赋值扔留着原地

}

aaa();

var a=10;

function aaa(a){

alert(a);

var a=20; //因为 a 是形参，优先级高于 var a; 所以 局部变量a的声明其实被忽略了。

}

aaa(a);

var a=[1,2,3];

var b=a;

b.push(4);

alert(a);//结果为[1,2,3,4] 当b改变的时候a也发生了改变

当b重新被赋值的时候 a不会改变.示例：

var a=[1,2,3];

var b=a;

b=[1,2,3,4]

alert(a)//结果为[1,2,3]

11 对象

1对象的作用：用于同时存储多个数据；

2声明方式：var obj =new object( );

Var obj={ };

3对象的属性操作

属性：对象中保存的数据，统称为属性；

方法；方法实际上也是属性，由于内部保存的是函数值，所以称为方法

3.1 对象.属性名

Var obj = new object();

Obj.name=’ jack ’;

Obj.age=’ jack ’;

Obj.sayHi=function ( ) {

Console.log(‘ 这是obj的sayhi方法 ’ )

}

Console.log(obj.name,obj.age);

Obj.sayHi();

3.2 对象[ ‘ 属性名 ’ ]

Var obj = new object();

Obj[‘name’] =’ jack ’;

Obj[‘age’]=’ jack ’;

Obj[‘sayHi’]=function ( ) {

Console.log(‘ 这是obj的sayhi方法 ’ )

}

Console.log(obj[‘name’],obj.age);

Obj.sayHi();

对象的遍历： for(var 变量名 in 对象名){

}

数组去重

Var arr = [2,1,3,2,5,1,2,45,12,45];

|  |  |
| --- | --- |
|  | |
|  | var result = []; | |
|  |  | |
|  | for (var i = 0; i < arr.length; i++) { | |
|  | var flag = true; // 用于标识当前arr[i]是否可以放入，true标识可以放入 | |
|  | for (var j = 0; j < result.length; j++) { | |
|  | if (arr[i] === result[j]) { | |
|  | flag = false; | |
|  | } | |
|  | } | |
|  | // 下面可以进行代码化简(不是必要的化简操作) | |
|  | if (flag) { | |
|  | result[result.length] = arr[i]; | |
|  | } | |
|  | } | |
|  | console.log(resu); | |
|  |  | |

# 函数

|  |
| --- |
|  |
|  | 1 基本使用方式：函数声明语句，函数表达式。 |
|  | function fun () {} |
|  | var fun1 = function () {}; |
|  |  |
|  | 2 匿名函数： |
|  | 注意：匿名函数因为没有函数名，所以不能单独使用 |
|  | var fun = function () {}; |
|  |  |
|  | 3 立即执行函数(自调用函数) |
|  | 效果：在书写位置执行一次。 |
|  | 作用：进行作用域的包裹，减少全局变量的使用。 |
|  | 使用位置：在每个功能外，使用立即执行函数包裹即可。 |
|  | 注意点：立即执行函数后必须添加分号; |
|  | (function () { |
|  | var sum = 0; |
|  | for (var i = 1; i <= 100; i++) { |
|  | sum += i; |
|  | } |
|  | console.log('这是一个立即执行函数。'); |
|  | })(); |
|  | (function () { |
|  | var sum = 0; |
|  | sum = 1 + 2; |
|  | })();  4 构造函数  Function CreateStu(){  1 创建了对象   |  | | --- | |  | |  | var stu1 = new Object(); | |  | // 2 给对象添加功能 | |  | stu1.name = name; | |  | stu1.age = age; | |  | stu1.sayHi = function () { | |  | console.log('这是sayHi方法的功能'); | |  | }; | |  | // 3 返回这个对象 | |  | return stu1; |   }  注意点： 像这种用于进行对象创建的函数，我们称为构造函数  1.调用前必须加new 2 在构造函数中只需要给要创建的对象进行功能设置即可 3 构造函数的函数名首字母大写（建议） |
|  |  |

其他函数还有：回调函数（将函数作为参数去使用），递归函数（在函数内调用当前函数）等，了解即可，在JS高级部分会再讲到并且有案例。

2 arguments: 用于获取函数或者数组中的所有参数。他具有索引和长度.length，是伪数组对象，没有数组的方法

3 this 的使用

在普通函数中 this指向的是window对象（window对象是浏览器环境中的顶级对象）

This在某个对象的方法中使用，指向当前对象（即函数调用者）

构造函数中this指向创建的对象，因为new对函数内的this指向进行了修改

# 内置对象

Math：

Math.max()用于计算多个值中的最大值。

Math.min()用于计算多个值中的最小值。

Math.random() 随机产生0-1（不包含1）的随机数

Math.round() 四舍五入

Math.ceil() 向上取整

Math.floor() 向上取整

Math.abs() 去绝对值

Date：

创建方式: var date = new Date（）；

Date.getFullYear() 获取年份

Date.getMonth() 获取月份

Date.getDay() 获取星期

Date.getDate() 获取天

Date.getHours() 获取小时

Date.getMinutes() 获取分钟

Date.getSeconds() 获取秒

Date.getMilliseconds() 获取毫秒

Array :

Var arr = new Array();

Arr.push(); 像数组的最后位置添加元素

Arr.pop(); 从数组的最后位置删除一个元素。

Arr.unshift() 向数组的最前面位置添加元素。

Arr.shift() 向数组的最前面位置删除一个元素.

Arr.concat() 用于数组

Arr.slice() 用于对数组中的指定部分元素进行拷贝

注意：此方法不会修改原数组。

参数1：拷贝起始位置索引值

参数2：拷贝结束位置索引值（不包含此元素）

console.log(arr.slice()); // 不传参数，默认拷贝所有元素

console.log(arr.slice(2)); // 从参数1索引对应位置拷贝到最后

console.log(arr.slice(1, 4));

arr.splice() 用于数组的增删改操作

参数1：截取的起始位置索引值

参数2：截取个数

返回值：截取后的元素组成的数组

console.log(arr.splice()); // 不传参没用

console.log(arr.splice(2)); // 从指定索引位置截取到数组最后

console.log(arr.splice(1, 3)); // 从指定索引位置截取指定个数元素

console.log(arr.splice(1, 2, 100, 200, 300)); // 修改功能

console.log(arr.splice(1, 0, 100, 200)); // 向数组中指定的索引位置放入元素

arr.indexOf() 用于获取数组中某个元素的索引值

参数1：要检索的元素值

参数2：默认值为0，表示检索的起始位置索引值

返回值：返回从左往右找到的第一个元素的索引值

lastIndexOf() 用于获取某个元素在数组中的索引值，检索顺序与indexOf相反。

reverse() 翻转数组

sort() 排序 - 掌握

如果数值大于9(两位数以及以上值)，也是按照第一个字符的顺序排列

|  |
| --- |
| 参数： |
|  | // - 升序(从小到大排序): function (a, b) { return a - b; } |
|  | // - 降序(从大到小排序): function (a, b) { return b - a; } |

join() 可以使用指定的连接符将每个数组元素连接

console.log(arr.join()); // 如果不传入参数，默认采用逗号连接

|  |
| --- |
| console.log(arr.join('')); // 如果传入空字符串，会直接将每个元素进行连接 |
|  |  |

字符串

length属性:用于获取字符串中的字符个数

concat() - 用于字符串连接操作 - 了解

slice() - 用于拷贝字符串中的某些指定字符 - 掌握A

参数1：起始位置索引值

参数2：结束位置索引值(结束位置的字符取不到)

返回值：拷贝后的字符串

substring() - 用于拷贝字符串中的某些指定字符 - 了解

参数1：起始位置索引值

参数2：结束位置索引值(结束位置的字符取不到)

返回值：拷贝后的字符串

substr() - 用于拷贝字符串中的某些指定字符 - 掌握

参数1：起始位置索引值

参数2：拷贝的字符个数

返回值：拷贝后的字符串

toUpperCase() 将字符串转换为大写

toLowerCase() 将字符串转换为小写

replace() 替换字符串内的某些部分

参数1：要替换的内容

参数2：替换为什么内容

注意：只能替换从左往右找到的第一个满足的字符串

split() - 用于按照指定的分隔符将字符串分隔

webAPI

1 . 入口函数

window.onload = function () {函数体};

2 根据ID获取元素

Var box = document.getElementById('ID值');