

目录

目录

作业讲评

学习目标

遍历对象的属性

JSON数据格式

什么是JSON数据格式

JSON数据格式的应用场景

JSON格式的定义

JSON与JS对象的区别

定义JSON对象

访问JSON对象的属性

遍历JSON对象的属性

JSON数组

JSON格式的字符串与JSON对象的转换

内置对象

内置对象分类

通过 MDN/W3C/手册 查询需要的内容

如何学习对象中的方法？

String对象

如何创建String对象

String对象的属性

String对象的方法

1) 根据位置获取字符

2) 字符串操作方法

3) 获取字符串位置方法

4) replace() 替换

5) 转换大小写

6) split() 切割字符串

7) trim() 去除字符串的头尾空格

String对象练习

Math对象

Math对象的属性

Math对象的方法

Math.random() 生成随机数 难点

Math案例

今日总结

今日作业

作业讲评

作业1:

```
1 // 1.用{}方式,创建一个电脑对象,有颜色,有重量,有品牌,有型号,可以看电影,可以听音乐,可以打游戏,可以敲代码
2 var computer = {};
3 computer.color = "black";
4 computer.weight = "2.5kg";
5 computer.bland = "华硕";
6 computer.type = "飞行堡垒6";
7 computer.movie = function(){
8 console.log( this.type + ",可以看电影");
9 }
```

```
10  computer.music = function(){
11  console.log( this.type +",可以听音乐");
12  }
13  computer.game = function(){
14  console.log( this.type +",可以打游戏");
15  }
16  computer.code = function(){
17  console.log( this.type +",可以敲代码");
18  }
19  console.log( computer.color );
20  console.log( computer["weight"] );
21  computer.code();
22  computer.movie();
23  computer.music();
24  computer.game();
```

作业2:

```
1  // 2.用new Object()方式,创建一个按钮对象,宽,高,背景颜色,可以鼠标点击
2  var button = new Object();
3  button.width = "100px";
4  button.height = "100px";
5  button.bgcolor = "pink";
6  button.dian = function(){
7  console.log("我是"+this.width+"宽度,"+this.height+"高
8  度,"+this.bgcolor+"背景颜色的按钮,我可以被点击");
9  }
10  button.dian();
```

作业3:

```
1  function Car(weight,color,bland){
2  // 属性
3  this.weight = weight;
4  this.color = color;
5  this.bland = bland;
6
7  // 方法
8  this.manned = function(){
9  console.log( this.bland + "可以载人");
10  }
11
12  this.pickUp = function(){
```

```

13 console.log( this.bland +"可以拉货");
14 }
15 }
16
17 // 构造函数定义的是一个大类,还需要通过new关键字实例化得到新的对象
18 var car1 = new Car("1000kg","white","宝马");
19 car1.manned();
20 car1.pickUp();
21
22 var car2 = new Car("2000kg","pink","奔驰");
23 console.log( car2.color );
24 console.log( car2.weight );
25 car2.manned();
26 car2.pickUp();

```

练习1:

```

1 // 王可可 是一条 阿拉斯加犬(type), 今年5岁了, 颜色是 红色。 它会 汪汪汪(cry) 它会 演电影 《后会无期》(showFilm)
2 var dog = {
3   dogname : "王可可",
4   type : "阿拉斯加犬",
5   age : 5,
6   color : 'red',
7   cry:function(){
8     console.log("汪汪汪");
9   },
10  showFilm:function(){
11    console.log("演电影 《后会无期》");
12  }
13 };
14
15 console.log("狗狗名字:"+dog.dogname);
16 console.log("狗狗类型:"+dog.type);
17 console.log("狗狗年龄:"+dog.age);
18 console.log("狗狗颜色:"+dog.color);
19 dog.cry();
20 dog.showFilm();

```

练习2:

```

1  // 使用new Object方式,创建一个 手机对象 型号(type) iPhoneX 颜色是 黑色 大小
   (size) 5.5寸 可以 发短信 (sendMessage) 输出 吃大肘子 可以 打电话 (call) 说 全
   聚德吃烤鸭
2
3  var mobilePhone = new Object();
4  mobilePhone.type = 'iPhoneX';
5  mobilePhone.color = 'black';
6  mobilePhone.size = '5.5寸';
7  mobilePhone.sendMessage=function(){
8  console.log("吃大肘子");
9  }
10 mobilePhone.call=function(){
11 console.log("全聚德吃烤鸭");
12 }
13 console.log("手机型号是"+mobilePhone.type+",颜色是"+mobilePhone.color+",
   大小是"+mobilePhone.size);
14 mobilePhone.sendMessage();
15 mobilePhone.call();

```

练习3:

```

1  // 利用构造函数 创建一个英雄Hero 对象。
2  // 里面有 姓名属性(username),类型属性(type), 血量属性(blood), 具有 攻击方法
   (attack)
3
4  // 使用的英雄:
5  // 安琪拉 法师型 80血量 法术攻击
6  // 后羿 射手型 100血量 物理攻击
7
8  function Hero(username,type,blood,attack){
9  this.username = username;
10 this.type = type;
11 this.blood = blood;
12 this.attack = function(){
13 console.log("攻击方法:"+attack);
14 };
15 this.showInfo = function(){
16 console.log(this.username+"是一个"+this.type+"英雄,有"+this.blood);
17 }
18 }
19 var anqila = new Hero("安琪拉","法师型","80血量","法术攻击");
20 anqila.showInfo();

```

```
21  anqila.attack();
22
23  var houyi = new Hero("后羿","射手型","100血量","物理攻击");
24  houyi.showInfo();
25  houyi.attack();
```

学习目标

- 能够遍历对象的属性
- 能够了解并使用JSON数据格式
- 能够遍历JSON对象的属性
- 能够掌握string对象的属性和方法
- 能够理解Math对象的属性和方法

遍历对象的属性

for...in 语句用于对数组或者对象的属性进行循环操作

格式:

```
1  for (var 变量名 in 对象) {
2      在此执行代码
3  }
4  或者
5  for(var 变量名 in 数组变量名){
6      console.log ( 数组变量名[变量名] );
7  }
```

- 这个变量是自定义 符合命名规范 但是一般我们都写为 k 或则 key 或者 attr
- in后面的可以是对象 也可以是数组 因为 数组也属于对象

我们可以使用for in 遍历 对象

举例:

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  <meta charset="UTF-8">
```

```
5 <title>Document</title>
6 <script type="text/javascript">
7 var arr = ["张三",23,"男"];
8 // console.log( arr[0] );
9 // console.log( arr[1] );
10 // console.log( arr[2] );
11 /*for(var i=0;i<arr.length;i++){
12 console.log( arr[i] );
13 }
14 console.log("");
15
16 var obj = {
17 "username":"李四",
18 "age":24,
19 "sex":"女"
20 };*/
21 // 遍历对象属性就是访问对象中每个属性
22 // console.log( obj.username );
23 // console.log( obj.age );
24 // console.log( obj.sex );
25
26 // 对象的属性是不可以使用for来遍历的,首先因为对象没有length这个属性,而且属性
  名没有规律
27
28 // console.log( obj.length );// undefined
29 /*for(var j=0;j<obj.length;j++){
30 console.log( obj[j] );
31 }*/
32
33 // 遍历对象的属性我们需要使用for...in,为什么不使用for呢,因为for循环的东西需要
  有规律
34
35 // for...in这是一个新的语法结构,for...in可以遍历对象也可以遍历数组,因为数组
  也是对象中的一种,叫数组对象
36 // for...in语法结构
37
38 /*
39 for (var 变量名 in 对象) {
40 在此执行代码
41 }
42
```

```
43  或者
44
45  for(var 变量名 in 数组变量名){
46      console.log ( 数组变量名[变量名] );
47  }*/
48
49  var obj = {
50      "username":"李四",
51      "age":24,
52      "sex":"女"
53  };
54
55  // 使用for...in遍历对象属性
56  for(var i in obj){
57      // 我们在for...in输出i, 这个时候i就是属性名
58      // console.log( i );
59      // 我们在for...in输出obj,其实就是我们那个对象
60      // console.log( obj );
61
62      // 访问对象属性有几种方法
63      // console.log( obj.i );// undefined,在for...in结构不能使用对象名.属性名访问属性
64
65      console.log( obj[i] );// 只能使用对象名["属性名"]访问属性
66  }
67
68  console.log("");
69  console.log("");
70  console.log("");
71  console.log("");
72
73  // 我们还可以使用for...in遍历数组
74  var arr = [10,20,30,40,50,60,70];
75  for(var j in arr){
76      // 这里的j变量代表"数组下标"
77      // console.log( j );
78      // 这里的arr变量代表当前数组
79      // console.log( arr );
80      // console.log( "" );
81
82      // console.log( arr.j );// undefined
```



```

83
84 console.log( arr[j] );
85 }
86
87 // for...in虽然也可以遍历数组,但是平时遍历数组,我们还是习惯用for,for...in一般留给遍历对象使用的
88 </script>
89 </head>
90 <body>
91
92 </body>
93 </html>

```

JSON数据格式

什么是JSON数据格式

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。由于其利于阅读和编写、同时也方便机器进行解析和生成。这使得JSON成为最理想的数据交换语言。

JSON数据格式的应用场景

只要有数据交换的地方，都会有JSON数据格式的身影。

① 手机号码归属地查询

The screenshot shows a web browser with the Baidu search engine. The search query is "手机号码归属地查询" (Mobile Number Location Query). The search results show a link to "手机号码归属地查询" (Mobile Number Location Query) with a brief description. The network tab is open, showing a request to "api.php?cb=jsonQuery1102000...". The response is a JSON object containing location information for the mobile number "13123456789".

Search Results:

- 手机号码"13123456789" 玉林 中国联通
- 13123456789
- 号码归属地数据由百度手机卫士提供
- 手机号码归属地查询
- 3天前 - ip138为你提供全面的手机号码归属地查询服务,电话号码归属地查询,包括移动手机号码,联通手机号码,固定电话归属地查询,电信手机号码等等。
- www.ip138.com/sj/ - 百度快照
- 手机号码归属地查询 手机在网站 手机号码定位 始创于2001年
- 手机号码归属地查询官方版,手机在网站拥有最全的手机号码数据库,手机定位查询归属地就上手机在线www.showji.com
- https://www.showji.com/ - 百度快照 - 30条评价
- 手机号码归属地查询 百度百科
- 应用为一款查询手机号码归属地的生活助手类应用,查询的手机号码涵盖中国移动、中国联通和中国电信的手机号码 (不包括固定电话号码)。查询结果包括归属省份、归属城市和卡类型。
- 新版特性 系统要求
- https://baike.baidu.com/ -
- 电话号码查询_号码认证_号码举报中心
- 本站原创的电话号码归属地查询功能,可以通过输入固定电话号码,查询到该号码所在的地理位置,最精确可以到村镇,还可查询中国移动、联通和电信的所有手机号码的归属...

Network Request Details:

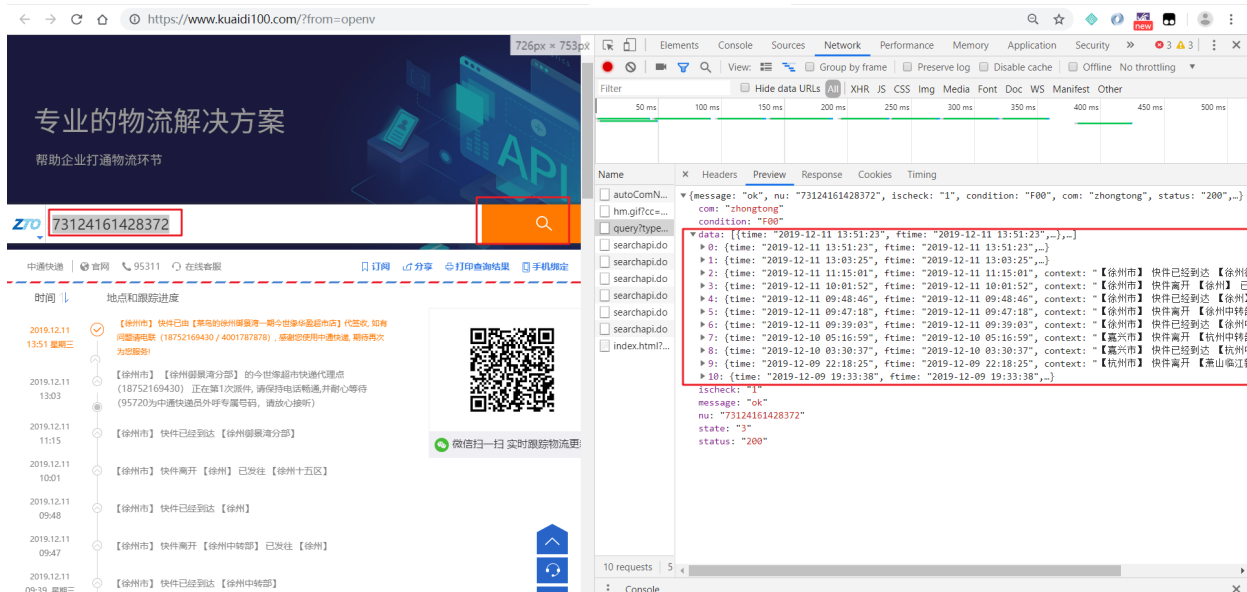
- Name: w.gif?q=%BA%CS%C2%EB...
- URL: api.php?cb=jsonQuery1102000...
- Method: GET
- Headers:
 - Accept: */*
 - Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.6,en;q=0.5
 - Cache-Control: no-cache
 - Content-Type: application/javascript
 - Host: www.baidu.com
 - Referer: https://www.baidu.com/s?ie=1&wd=手机号码归属地查询
- Response:


```

{
  "status": "0",
  "t": "",
  "set_cache_time": "1572262016",
  "data": {
    "StdStg": "6004",
    "StdStl": "8",
    "update_time": "1572262016",
    "cambrian_appid": "0",
    "ExtendedLocation": "",
    "OriginQuery": "13123456789",
    "SiteId": "2002696",
    "StdStg": "6004",
    "StdStl": "8",
    "appid": "",
    "cambrian_appid": "0",
    "city": "玉林",
    "clickneed": "0",
    "disp_type": "0",
    "fetchkey": "6004_1312345",
    "key": "1312345",
    "loc": "https://ssl.baidu.com/8a0QcnSm2Q511B6InVG/q?r=2002696&k=1312345",
    "origphoneno": "13123456789",
    "phoneinfo": "手机号码"13123456789"",
    "phoneno": "13123456789",
    "prov": "广西",
    "querytype": "手机号码",
    "resourceid": "6004",
    "role_id": "1",
    "showlamp": "1",
    "showurl": "https://anquan.baidu.com/page/41"
  }
}

```

② 快递查询



JSON格式的定义

基本语法：

```
1 {  
2   "name": "Mark",  
3   "age": 18,  
4   "mobile": "13586007445"  
5 }
```

注意:

JSON格式的数据，其键名必须使用**"双引号"**引起来

JSON格式的数据，**"不能出现成员方法，只能是成员属性"**

但是在JS中，如果以上格式赋值给某个变量，这个变量也是**JS对象**

```
1 var jsonObj = {  
2   "name": "Mark",  
3   "age": 18,  
4   "mobile": "13586007445"  
5 };
```

JSON与JS对象的区别

区别	JSON	javascript对象
含义	仅仅是一种数据格式	表示类的实例
传输	可以跨平台数据传输,速度快	不能传输
表现	①键值对方式,键必须加 双引号 ②值不能是方法函数,不能是undefined或者NaN	①键值对方式,键不加引号②值可以是函数、对象、字符串、数字、布尔等
相互转换	JSON转换为JS对象:①JSON.parse(JSONString)(不兼容	JS对象转换为

ie7)@var jsobj=eval("(" + JSONString + ")") (兼容所有浏览器,但不安全,会执行json里面的表达式	JSON:JSON.stringify(jsobj);(不兼容ie7)
---	-------------------------------------

其他:调用JSON官网的js,实现parse和stringify在各个浏览器的兼

容:<https://github.com/douglascrockford/JSON-js/blob/master/json2.js>

定义JSON对象

```
1 // 定义JSON对象
2 var jsonObj = {
3   "name": "Allen",
4   "age": 23,
5   "mobile": "13123456789"
6 }
```

访问JSON对象的属性

基本语法:

```
1 JSON对象.属性
2 或
3 JSON对象[属性]
```

```
1 // 定义JSON对象
2 var jsonObj = {
3   "name": "zhangsan",
4   "age": 23,
5   "mobile": "13123456789"
6 }
7 console.log(jsonObj.name);
8 console.log(jsonObj["age"]);
```

遍历JSON对象的属性

for...in 语句用于对数组或者对象的属性进行循环操作

格式:

```
1 for (attr in JSON对象) {
2   console.log( attr ); // attr是属性名
```

```
3 console.log( JSON对象[attr] ); // JSON对象[attr]可以得到属性名对应的属性值
4 }
```

- 这个变量是自定义 符合命名规范 但是一般我们都写为 k 或则 key 或者 attr
- 后面的可以是对象 也可以是数组 因为 数组也属于对象

注意: 遍历的时候使用 JSON对象. 属性名 得到的是undefined , 所以只能使用 JSON对象[属性名]

举例:

```
1 // 定义JSON对象
2 var jsonObj = {
3     "name": "zhangsan",
4     "age": 23,
5     "mobile": "13123456789"
6 }
7 // 遍历JSON对象中的所有属性
8 for(var attr in jsonObj){
9     console.log(attr); // 属性名
10    console.log(jsonObj[attr]); // 属性值
11    console.log(jsonObj.attr); // undefined
12    console.log("");
13 }
```

JSON数组

问题：在JS中，我们可以通过JSON格式的数据来保存一个人的信息，如果我们想保存多个人的信息如何进行操作呢？答：使用JSON数组，把多个json对象放在一个数组中

基本语法:

```
1 var 变量名 = [
2     {键值对},
3     {键值对},
4     {键值对},
5     ...
6 ];
```

举例:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6   <script type="text/javascript">
7
8   // JSON数组
9   // 问题: 在JS中, 我们可以通过JSON格式的数据来保存一个人的信息;
10
11   // 如果我们想保存多个人的信息如何进行操作呢? 答: 使用JSON数组, "把多个json对象放在一个数组中"
12
13   /*基本语法:
14   var 变量名 = [
15     {键值对},
16     {键值对},
17     {键值对},
18     ...
19   ];
20   */
21
22   // 保存一个人的信息
23   var jsonObj = {
24     "username" : "张三",
25     "age" : 23,
26     "sex" : "男"
27   };
28
29   console.log( jsonObj.username )
30   console.log( jsonObj.age )
31   console.log( jsonObj.sex )
32   console.log("");
33   console.log("");
34
35   // 如果需要保存多个人的个人信息, 可以就可以使用JSON数组, 说白了, 就是把多个JSON对象放在一个数组里面
36   // 我们之前说过一个东西, 数组中的元素类型可以是任意, 可以是字符串, 数值, 布尔, null, undefined, 数组, 对象
37
```

```
38  var arrObjs = [  
39  {  
40    "username" : "李四",  
41    "age" : 24,  
42    "sex" : "女"  
43  },  
44  {  
45    "username" : "小明",  
46    "age" : 25,  
47    "sex" : "男"  
48  },  
49  {  
50    "username" : "小红",  
51    "age" : 28,  
52    "sex" : "女"  
53  }  
54  ];  
55  console.log( arrObjs );  
56  
57  // 我们想直接输出李四,该怎么写  
58  console.log( arrObjs[0] );  
59  console.log( arrObjs[0]["username"] );  
60  console.log( arrObjs[0].username );  
61  console.log("");  
62  
63  
64  // 如果遍历里面所有属性值出来  
65  for( var i=0;i < arrObjs.length; i++){  
66    // 因为对象没有数组下标0,1,2,3...所以不能使用for循环  
67    for(var j in arrObjs[i] ){  
68      console.log( arrObjs[i][j] );  
69    }  
70  }  
71  </script>  
72  </head>  
73  <body>  
74  
75  </body>  
76  </html>
```

JSON格式的字符串与JSON对象的转换

把JSON对象转换为JSON格式的字符串

语法如下

```
1 JSON.stringify(JS对象);
```

把JSON字符串转换为JSON对象

语法如下

```
1 JSON.parse(JSON格式的字符串);
```

举例:

```
1 <script type="text/javascript">
2 // JSON格式的字符串与JS对象的转换
3 // JSON格式的字符串,就是把json对象的内容,全部放一个字符串里面,为什么要放字符串
  里面,因为数据在传输的时候,比如数据库中只能保存字符串,保存不了对象
4
5 // json对象,因为字符串不能得到我们想要的属性值,所以又要转回json对象
6
7
8 // 把JSON对象转换为JSON格式的字符串 语法如下
9 // JSON.stringify(JS对象);
10
11 // 定义了一个json对象
12 var jsonObj = {
13   "username" : "张三",
14   "age" : 23,
15   "sex" : "男"
16 };
17 // 我们使用document.write直接输出对象
18 // document.write( jsonObj );
19
20 // 把JSON对象转换为JSON格式的字符串 语法如下
21 var jsonStr = JSON.stringify( jsonObj );
22 // 输出JSON格式的字符串
23 document.write( jsonStr );
24
25
26
27 // 把JSON字符串转换为JSON对象 语法如下
```

```
28 // JSON.parse(JSON格式的字符串);
29 // JSON字符串需要用引号在最外层包裹
30 var str = '{"username":"张三","age":23,"sex":"男"}';
31 console.log( str );
32 console.log( typeof(str) );
33 console.log("");
34
35 var obj = JSON.parse( str );
36 console.log( obj );
37 console.log( typeof(obj) );
38 </script>
```

内置对象

JavaScript中的对象分为3种: **内置对象**、**自定义对象**、**浏览器对象**

内置对象就是指**JS语言自带的一些对象**，供开发者使用，这些对象提供了一些常用的或是最基本而必要的功能

JavaScript 提供多个内置对象: **Math/Array/Number/String/Boolean...**

对象只是带有**属性**和**方法**的特殊数据类型。

学习一个内置对象的使用，只要学会其常用的成员的使用(通过查文档学习)

可以通过MDN/W3C来查询

内置对象的方法很多，我们只需要知道内置对象提供的常用方法，使用的时候查询文档。

内置对象分类



















- String对象:提供了处理字符串的属性与方法。
- Math对象:提供了一些操作数学方面属性与方法
- Array对象:提供了一些操作数组的属性与方法
- Date对象:提供了一些对时间日期操作的方法

- Number对象:它主要是提供了一个操作数值的方法
- Events对象:提供对JavaScript事件的处理信息。

通过 MDN/W3C/手册 查询需要的内容

Mozilla 开发者网络(MDN)提供有关开放网络技术(Open Web)的信息，包括 HTML、CSS 和万维网及 HTML5 应用的 API。

- MDN : <https://developer.mozilla.org/zh-CN/>
- W3C: <http://www.w3school.com.cn/jsref/index.asp>
- 手册:

电脑 > 软件 (D:) > 软件安装包 > 手册					搜索
名称	修改日期	类型	大小		
 CSS2.1中文完全参考手册.chm	2013/8/23 21:12	编译的 HTML 帮...	256 KB		
 CSS2.1中文完全参考手册.chw	2013/8/23 21:12	CHW 文件	29 KB		
 HTML4 参考手册.chm	2013/8/23 21:12	编译的 HTML 帮...	969 KB		
 HTML4 参考手册.chw	2013/9/17 14:26	CHW 文件	25 KB		
 Javascript语言手册(jscript).chm	2013/8/23 21:12	编译的 HTML 帮...	603 KB		
 Javascript语言手册(jscript).chw	2014/6/3 12:08	CHW 文件	185 KB		
 JavaScript核心参考手册.chm	2018/1/22 21:13	编译的 HTML 帮...	301 KB		
 JavaScript核心参考手册.chw	2018/3/27 16:38	CHW 文件	9 KB		
 MySQL5.1中文参考手册.chm	2013/8/23 15:25	编译的 HTML 帮...	4,508 KB		
 MySQL5.1中文参考手册.chw	2013/8/23 21:12	CHW 文件	200 KB		
 MySql参考手册-新.CHM	2017/9/6 14:55	编译的 HTML 帮...	201 KB		
 php_enhanced_zh00.chm	2016/5/2 16:24	编译的 HTML 帮...	29,770 KB		
 php_enhanced_zh00.chw	2016/5/2 16:24	CHW 文件	885 KB		
 php_manual_zh.chm	2016/11/24 17:05	编译的 HTML 帮...	15,465 KB		
 php_manual_zh.chw	2016/12/12 16:04	CHW 文件	953 KB		
 php7.1_enhanced_en.chm	2018/4/26 23:55	编译的 HTML 帮...	31,794 KB		
 php7.1_enhanced_en.chw	2019/5/7 11:24	CHW 文件	1,125 KB		
 php7.1_enhanced_zh.chm	2018/4/26 23:49	编译的 HTML 帮...	32,580 KB		

隐藏 查找 上一步 下一步 上一步 前进 停止 刷新 主页 打印

目录(C) 索引(N) 搜索(S) 键入关键字进行查找(W):

运算符
! 运算符
!= 运算符
!== 运算符
\$'
\$&
\$`
\$+
\$1...\$9 属性
% 运算符
%= 运算符
& 运算符
&& 运算符
&= 运算符
* 运算符
*= 运算符
/ 运算符
/= 运算符
?: 运算符
@cc_on 语句
@if
@if 语句
@set 语句
^ 运算符
^= 运算符
| 运算符
|= 运算符
“非”
“非贪婪”匹配
“贪婪”匹配
正则表达式
+ 运算符
++
+= 运算符
+运算符
< 按位左移

JScript

JScript 语言参考

- 特性信息
- 字母顺序的关键字列表
- 错误
- 函数
- 方法
- 对象**
- 运算符
- 属性
- 语句

隐藏 查找 上一步 下一步 上一步 前进 停止 刷新 主页 打印

目录(C) 索引(N) 搜索(S) 键入关键字进行查找(W):

运算符
! 运算符
!= 运算符
!== 运算符
\$'
\$&
\$`
\$+
\$1...\$9 属性
% 运算符
%= 运算符
& 运算符
&& 运算符
&= 运算符
* 运算符
*= 运算符
/ 运算符
/= 运算符
?: 运算符
@cc_on 语句
@if
@if 语句
@set 语句
^ 运算符
^= 运算符
| 运算符
|= 运算符
“非”
“非贪婪”匹配
“贪婪”匹配
正则表达式
+ 运算符
++
+= 运算符
+运算符
< 按位左移

JScript

JScript 语言参考

- 特性信息
- 字母顺序的关键字列表
- 错误
- 函数
- 方法
- 对象**
- 运算符
- 属性
- 语句

语言元素

[ActiveXObject 对象](#)
[Array 对象](#)
[Boolean 对象](#)
[Date 对象](#)
[Dictionary 对象](#)
[Enumerator 对象](#)
[Error 对象](#)
[FileSystemObject 对象](#)
[Function 对象](#)
[Global 对象](#)
[Math 对象](#)
[Number 对象](#)
[Object 对象](#)
[RegExp 对象](#)
[正则表达式对象](#)
[String 对象](#)
[VBArray 对象](#)

描述

启用并返回一个 Automation 对象的引用。
提供对创建任何数据类型的数组的支持。
创建一个新的 Boolean 值。
提供日期和时间的基本存储和检索。
存储数据键、值的对象。
提供集合中的项的枚举。
包含在运行 JScript 代码时发生的错误的有关信息。
提供对计算机文件系统的访问。
创建一个新的函数。
是一个内部对象，目的是将全局方法集中在一个对象中。
是一个内部对象，提供基本的数学函数和常数。
表示数值数据类型和提供数值常数的对象。
提供所有的 JScript 对象的公共功能。
存储有关正则表达式模式查找的信息。
包含一个正则表达式模式。
提供对文本字符串的操作和格式处理，判定在字符串中是否存在某个子字符串及确定其位置。
提供对 Visual Basic 安全数组的访问。



如何学习对象中的方法?

1. 先了解这个方法的功能是什么(通过看手册了解)
2. 每个参数的意义和**变量类型**
3. 返回值意义和**变量类型**
4. demo代码(示例代码)演示进行测试效果

String对象

如何创建String对象

- **定义的字符串变量名**其实就是一个字符串对象 这种方式称之为隐式创建

举例:

```
1 var str = "abc";
```

- 使用new关键字和String()方法来创建! 这种方式称之为显示创建

举例:

```
1 var str = new String("abc");
```

String对象的属性

属性名	機能
-----	----

StringObject.length	得到字符串对象的长度 注意 ：这里的长度指的 字符 的个数！
---------------------	--

举例：

String对象的方法

注意:字符串所有的方法，**都不会修改字符串本身，操作完成会返回一个新的字符串**

注意:字符串方法中的索引都是**从0开始**!

1) 根据位置获取字符

方法名	说明	使用
charAt(index)	返回指定位置的字符(index 字符串的索引号)	str.charAt(0)
charCodeAt(index)	获取指定位置处字符的ASCII码 (index索引号)	str.charCodeAt(0)
str[index]	获取指定位置处字符	HTML5, IE8+ 支持 和charAt()等效

ASCII 字符代码表 一																								
高四位 低四位		ASCII非打印控制字符												ASCII 打印字符										
		0000				0001				0010	0011	0100	0101	0110	0111									
		十进制	字符	ctrl	代码	字符解释	十进制	字符	ctrl	代码	字符解释	十进制	字符	十进制	字符	十进制	字符	十进制	字符	十进制	字符	ctrl		
0000	0	0	BLANK NULL	^@	NUL	空	16	▶	^P	DLE	数据链路转意	32		48	0	64	@	80	P	96	`	112	p	
0001	1	1	☺	^A	SOH	头标开始	17	◀	^Q	DC1	设备控制 1	33	!	49	1	65	A	81	Q	97	a	113	q	
0010	2	2	☹	^B	STX	正文开始	18	↕	^R	DC2	设备控制 2	34	"	50	2	66	B	82	R	98	b	114	r	
0011	3	3	♥	^C	ETX	正文结束	19	!!	^S	DC3	设备控制 3	35	#	51	3	67	C	83	S	99	c	115	s	
0100	4	4	♦	^D	EOT	传输结束	20	¶	^T	DC4	设备控制 4	36	\$	52	4	68	D	84	T	100	d	116	t	
0101	5	5	♣	^E	ENQ	查询	21	¢	^U	NAK	反确认	37	%	53	5	69	E	85	U	101	e	117	u	
0110	6	6	♠	^F	ACK	确认	22	■	^V	SYN	同步空闲	38	&	54	6	70	F	86	V	102	f	118	v	
0111	7	7	●	^G	BEL	震铃	23	↑	^W	ETB	传输块结束	39	'	55	7	71	G	87	w	103	g	119	w	
1000	8	8	□	^H	BS	退格	24	↑	^X	CAN	取消	40	(56	8	72	H	88	X	104	h	120	x	
1001	9	9	○	^I	TAB	水平制表符	25	↓	^Y	EM	媒体结束	41)	57	9	73	I	89	Y	105	i	121	y	
1010	A	10	▣	^J	LF	换行/新行	26	→	^Z	SUB	替换	42	*	58	:	74	J	90	Z	106	j	122	z	
1011	B	11	♂	^K	VT	垂直制表符	27	←	^[ESC	转意	43	+	59	;	75	K	91	[107	k	123	{	
1100	C	12	♀	^L	FF	换页/新页	28	└	^\	FS	文件分隔符	44	,	60	<	76	L	92	\	108	l	124		
1101	D	13	♪	^M	CR	回车	29	↔	^]	GS	组分隔符	45	-	61	=	77	M	93]	109	m	125	}	
1110	E	14	🎵	^N	SO	移出	30	▲	^G	RS	记录分隔符	46	.	62	>	78	N	94	^	110	n	126	~	
1111	F	15	☼	^O	SI	移入	31	▼	^-	US	单元分隔符	47	/	63	?	79	O	95	_	111	o	127	Δ	Back space

注：表中的ASCII字符可以用:ALT + “小键盘上的数字键” 输入

注：表中的ASCII字符可以用:ALT + “小键盘上的数字键” 输入

举例：

```
1 <!DOCTYPE html>
2 <html lang="en">
```

```
3 <head>
4 <meta charset="UTF-8">
5 <title>Document</title>
6 <script type="text/javascript">
7 // 如果你想使用String对象的方法,你首先要有String
8 // 如何创建String对象
9 // 方法一:通过字面量 "" 或者 '' 字面量就是编程语言提供给程序员一种比较方便的写法
10 var str1 = "abc";
11 var str2 = '456789';
12
13 // 方法二:通过new String("字符串") 构造函数得到实例化对象
14 var str3 = new String();
15 var str4 = new String("你好,hello");
16
17 console.log( str1, str2, str3, str4);
18 document.write( str1 + "<br/>" );
19 document.write( str2 + "<br/>" );
20 document.write( str3 + "<br/>" );
21 document.write( str4 + "<br/>" );
22
23 // String对象的属性
24 // 字符串长度 String对象.length 注意:指的是字符个数的长度
25
26 console.log( str1.length );// 3
27 console.log( str4.length );// 8
28 console.log("");
29
30 // 1) 根据位置获取字符
31 // charAt(index) 返回指定位置的字符(index 字符串的索引号) index是从0开始的,
32 // 0代表取第一个字符串, 比如:str.charAt(0)
33 var str5 = "我爱我的祖国";
34 console.log( str5.charAt(0) );// 我
35 console.log( str5.charAt(3) );// 的
36
37 // 更常用 str[index] 获取指定位置处字符 HTML5, IE8+支持 和charAt()等效
38 console.log( str5[5] );// 国
39 console.log( str5[ str5.length-1 ] );// 国
40 console.log("");
41
```

```

42 // charCodeAt(index) 获取指定位置处字符的ASCII码 (index索引号) str.charCodeAt(0)
43 var str6 = "abc123";
44 console.log( str6.charCodeAt(0) );// 97
45 console.log( str6.charCodeAt(3) );// 49
46
47
48 </script>
49 </head>
50 <body>
51
52 </body>
53 </html>

```

2) 字符串操作方法

方法名	说明
concat(str1,str2,str3...)	concat() 方法用于连接两个或多个字符串。拼接字符串，等效于+，+更常用
substr(start,length)	从start位置开始(索引号)，length 取的个数 重点记住这个 如果length (长度) 没有书写 表示一直截取到字符串的末尾 如果有写则表示截取的长度; 另外,start还可以写负值,表示从倒数第一个开始截取,比如 "abc".substr(-2,1) 表示从倒数第二个字符串开始截取一个长度的字符
substring(start, end)	截取字符串 从start(开始下标)处开始截取;如果end(结束下标) 没有书写 表示一直截取到字符串的末尾 如果有写则表示截取到end下标的前一个位置

举例:

```

1  /*String对象的方法,必须由String对象来调用*/
2
3  // 字符串操作方法
4  // concat(str1,str2,str3...) concat() 方法用于连接两个或多个字符串。拼接字符串, "等效于+, +更常用"
5  var str = "abc";
6  var str1 = "456";
7  var str2 = "zxv";
8  var str3 = "789";
9  console.log( str.concat(str1,str2,str3) );
10 // 拼接字符串, "等效于+, +更常用"
11 console.log( str+str1+str2+str3 );
12
13 /*var str4 = 123;

```



```
14  var str5 = 456;
15  console.log( str4.concat( str5 ) );// str4不是string对象,所以不能使用concat方法*/
16
17
18  // substr(start,length) 从start位置开始(索引号) start也是从0开始的, length
19  h  截取的字符个数 重点记住这个 如果length (长度)没有书写 表示一直截取到字符串的末尾
20  如果有写则表示截取的长度
21
22  var str6 = "快到中午了,大家饿了,老师饿了,想吃饭了";
23
24  // 如果length (长度)没有书写 表示一直截取到字符串的末尾
25  console.log( str6.substr(2) );
26  console.log( str6.substr(0) );
27
28  // 如果length有写则表示截取的长度
29  console.log( str6.substr(2,2) );// 从第2个字符串开始,截取两个字符,包括第2个
30  字符
31  console.log( str6.substr(6,4) );// 从第6个字符串开始,截取四个字符,包括第6个
32  字符
33  console.log("");
34
35  // substring(start, end) 截取字符串 从start(开始下标)处开始截取 ; 如果end
36  (结束下标) 没有书写 表示一直截取到字符串的末尾 如果有写则表示截取到end下标的前
37  一个位置
38  var str7 = "野火烧不尽,春风吹又生";
39
40  // 如果end(结束下标) 没有书写 表示一直截取到字符串的末尾 如果有写则表示截取
41  到end下标的前一个位置
42  console.log( str7.substring( 0 , 5 ) );
43  console.log( str7.substring( 6 ) );
44  console.log( str7.substring( 2 , 8 ) );
45  console.log("");
46
47  // 扩展一下: substr(start,length) 其中start的取值还可以是负数,如果是负数,代
48  表倒数第几个字符串开始截取指定length长度个字符串,如果length不设置代表截取到字符串
49  的末尾
50  var str8 = "各位靓仔靓女,大家下午好啊";
51  console.log( str8.substr(2,4) );
52  console.log( str8.substr(0,2) );
53
54  // 如果length不设置代表截取到字符串的末尾
55  console.log( str8.substr(-4) );
56  console.log( str8.substr(-6) );
57  console.log( str8.substr(-6,5) );
```

3) 获取字符串位置方法

方法名	说明
<code>indexOf('要查找的字符或字符串')</code>	返回指定内容在原字符串中第一次出现的位置，如果找不到就返回 -1; 如果查找的是多个字符, 找到的就返回第一个字符的下标
<code>lastIndexOf('要查找的字符或字符串')</code>	返回指定内容在原字符串中最后一次出现的位置，如果找不到就返回 -1; 如果查找的是多个字符, 找到的就返回第一个字符的下标

举例:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6   <script type="text/javascript">
7     // 获取字符串位置的方法,用于从字符串中查找指定字符或者字符串的位置
8
9     // indexOf('要查找的字符或字符串') 返回指定内容在原字符串中第一次出现的位置,
10    如果找不到就返回 -1; 如果查找的是多个字符 ,找到的就返回第一个字符的下标
11
12    var str1 = "1789h23hello456hello789efg";
13    // 查找h字符在str1中第一次出现的位置
14    console.log( str1.indexOf("h") );// 4
15
16    // 查找hello字符串在str中第一次出现的位置
17    console.log( str1.indexOf("hello") );// 3
18
19    // 查找一个不存在的字符,会返回-1
20    console.log( str1.indexOf("a") );// -1
21
22    console.log("");
23
24
25    // lastIndexOf('要查找的字符或字符串') 返回指定内容在原字符串中最后一次出现的
26    位置, 如果找不到就返回 -1; 如果查找的是多个字符 ,找到的就返回第一个字符的下标
27    var str2 = "1789h23hello456hello789efgh";
```



```

28 console.log( str2.indexOf("hello") );// 7
29
30 // 返回指定内容在原字符串中最后一次出现的位置
31 console.log( str2.lastIndexOf("hello") );// 15
32
33 console.log( str2.lastIndexOf("h") );// 26
34
35 // 如果找不到,也是返回-1
36 console.log( str2.lastIndexOf("a") );// -1
37 </script>
38 </head>
39 <body>
40
41 </body>
42 </html>

```

4) replace() 替换

replace() 方法用于在字符串中用一些字符替换另一些字符 **默认只替换一次,不会全部替换**

格式如下:

- 1 **replace**(被替换的字符串,要替换为的字符串);
- 2 默认只能进行一次替换,不会全部替换

- 1 如果需要全部替换,可以使用正则表达式(这个后面会学)
- 2 在js里面通过//来定义正则表达式
- 3 **g=>**global全局的意思
- 4 **i=>**ignore表示不区分大小写

举例:

```

1
2 // replace()方法中还可以放正则表达式
3 // replace() 方法用于在字符串中用一些字符替换另一些字符
4 // 默认只替换一次,不会全部替换
5
6 // 格式如下:
7 // replace(被替换的字符串, 要替换为的字符串);
8 // 默认只能进行一次替换,不会全部替换

```

```

9
10 var str1 = "我去你妹的,你妹的逗比";
11 console.log( str1.replace("你妹","*") );
12
13 /*
14 如果需要全部替换,可以使用正则表达式(这个后面会学)
15 在js里面通过//来定义正则表达式
16 g=>global全局的意思
17 i=>ignore表示不区分大小写
18 */
19
20 var str2 ="hello帅哥,Hello美女,hEllo世界";
21
22 // 定义一个正则表达式, 正则是在注释//里面的
23 var reg = /hello/gi;
24
25 // replace(正则表达式, 要替换为的字符串);
26 console.log( str2.replace( reg ,"你好") );

```

5) 转换大小写

toUpperCase()

英文字母全部转换成大写

toLowerCase()

转换成小写

举例:

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6   <script type="text/javascript">
7     // 转换大小写,比如:输入验证码的时候,
8     // 有些网页的验证码不区分大小写的
9     var str1 = "hEllo";
10    // toUpperCase()
11    // 英文字母全部转换成大写
12    console.log( str1.toUpperCase() );
13
14    var str2 = "aBc";
15    // toLowerCase()
16    // 英文字母全部转换成小写
17    console.log( str2.toLowerCase() );
18  </script>
19 </head>
20 <body>
21
22 </body>
23 </html>

```

HELLO
abc

6) split() 切割字符串

- 1 `split(sep)` 使用`sep`参数将字符串分隔为一个数组
- 2 `sep`就是代码根据什么字符来分割字符串
- 3 `sep`如果是`""`代表,分割每一个字符

举例:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6   <script type="text/javascript">
7     // split() 切割字符串
8     // split(sep分隔符) 使用sep分隔符参数
      将"字符串"分隔为一个"数组"
9
10    // sep就是代码根据什么字符来分割字符串
11
12    var str1 = "张三-李四-王五-赵六";
13    console.log( str1 );
14    console.log( str1.split("-") );
15
16    // sep如果是""代表,分割每一个字符
17    console.log( str1.split("") );
18
19    // 如果分割符在字符串中不存在,那么就会
20    把整个字符串当做一个数组元素放进数组中
21    console.log( str1.split(" ") );
22  </script>
23 </head>
```

10-split()切割字符串.html:13
张三-李四-王五-赵六
10-split()切割字符串.html:14
(4) ["张三", "李四", "王五", "赵六"]
10-split()切割字符串.html:17
(11) ["张", "三", "-", "李", "四", "-", "王", "五", "-", "赵", "六"]
10-split()切割字符串.html:20
["张三-李四-王五-赵六"]

7) trim() 去除字符串的头尾空格

```
4 <meta charset="UTF-8">
5 <title>Document</title>
6 <script type="text/javascript">
7   // trim() 去除字符串的"头尾"空格
8   var str = "  a b c  ";
9   console.log( str );
10  console.log( str.length );
11  console.log("");
12
13  // 去除字符串头尾空格
14  console.log( str.trim() );
15  console.log( str.trim().length );
16  console.log("");
17
18  // 如果想取出字符串中间的空格,
19  又需要用正则表达式了
20  // 正则表达式后续课程会学
21  var str2 = "  a b c  ";
22  var reg = /\s/gi;
23  console.log( str2.replace(reg,"") );
24 </script>
25 </head>
26 <body>
27 </body>
```

11-trim()去除字符串的头尾空格.html:9
a b c
11-trim()去除字符串的头尾空格.html:10
11-trim()去除字符串的头尾空格.html:11
a b c
11-trim()去除字符串的头尾空格.html:14
5
11-trim()去除字符串的头尾空格.html:15
abc
11-trim()去除字符串的头尾空格.html:22

String对象练习

- 1 练习: 截取字符串"我爱中华人民共和国", 中的"中华"
- 2 练习: 把字符串"abcd123456"每个字符倒过来输出变成6 5 4 3 2 1 d c b a

- 3 练习：获取文件的扩展名 "C:/Users/Administrator/Desktop/images/dog.png"
- 4 练习：把字符串"abc~~oe~~foxy~~o~~zzopp"中所有的o替换成*
- 5
- 6 练习：给定一个字符串如:"~~a~~bcdfg~~a~~aa123411156"问题如下：
- 7 1、 求该字符串的长度
- 8 2、 取出指定位置对应的字符，0,3,5,9
- 9 3、 查找指定字符是否在以上字符串中存在 i,c,b
- 10 4、 替换指定的字符 所有a替换为A,所有1替换为一
- 11 5、 截取指定开始位置到结束位置的字符串如:取得下标从1到5之间的所有字符串

代码:

```
1 <script type="text/javascript">
2 // 练习：截取字符串"我爱中华人民共和国"，中的"中华"
3 var str1 = "我爱中华人民共和国";
4 console.log( str1.substr(2,2) );
5 console.log("");
6
7
8 // 练习：把字符串"abcd123456"每个字符倒过来输出变成6 5 4 3 2 1 d c b a
9 var str2 = "abcd123456";
10 for(var i=str2.length-1;i>=0;i--){
11     console.log( str2[i] );
12     // console.log( str2.charAt(i) );
13 }
14
15 console.log("");
16
17 // 练习：获取文件的扩展名 "C:/Users/Administrator/Desktop/images/dog.png"
18 var str3 = "C:/Users/Administrator/Desktop/images/dog.png";
19 // 目标是获取png
20
21 // 根据/正斜杠分割字符串
22 var arr = str3.split("/");
23
24 // 获取文件名
25 var filename = arr[ arr.length-1 ];// dog.png
26
27 // 获取.的位置
28 var pos = filename.indexOf(".");// 3
29
```

```
30 // console.log( filename );
31
32 // 最后截取文件名
33 console.log( filename.substr( pos+1 ) );
34 console.log("");
35
36 // 练习：把字符串"abcfoxyozzopp"中所有的o替换成*
37 var str4 = "abcfoxyozzopp";
38 var reg = /o/gi;
39 console.log( str4.replace( reg , "*" ) );
40 console.log("");
41
42 // 练习：给定一个字符串如："abcfgaaa123411156"问题如下：
43 var str5 = "abcfgaaa123411156";
44 // 1、 求该字符串的长度
45 console.log( str5.length );
46
47 // 2、 取出指定位置对应的字符，0,3,5,9
48 console.log( str5[0] );
49 console.log( str5[3] );
50 console.log( str5[5] );
51 console.log( str5[9] );
52 console.log("");
53
54 // 3、 查找指定字符是否在以上字符串中存在 i,c,b
55 console.log( str5.indexOf("i") );
56 console.log( str5.indexOf("c") );
57 console.log( str5.indexOf("b") );
58
59 // 4、 替换指定的字符 所有a替换为A,所有1替换为一
60 var reg1 = /a/gi;
61 var newStr = str5.replace( reg1, "A" );
62
63 var reg2 = /1/gi;
64 console.log( newStr.replace( reg2, "一" ) );
65 console.log("");
66
67 // 5、 截取指定开始位置到结束位置的字符串如：取得下标从1到5之间的所有字符串
68 // substring(start,end)
69 console.log( str5.substring(1,5) );
70 </script>
```

Math对象

Math对象不需要创建 也就是不需要new Math() 直接使用即可！

Math关键字就是对象名

跟数学相关的运算就可以使用Math对象提供的方法(比如求绝对值，取整,四舍五入)

Math对象的属性

属性名	功能
Math.PI	求圆周率

Math对象的方法

方法名	功能
Math.ceil(x)	对x进行向上取整 得到一个比当前数要大的最小整数
Math.floor(x)	对x进行向下取整 得到一个比当前数要小的最大整数
Math.round(x)	对x进行四舍五入
Math.abs(x)	返回x的绝对值
Math.max(x1,x2,x3...)	求x1,x2,x3...这些数中的 最大值
Math.min(x1,x2,x3...)	求x1,x2,x3...这些数中的 最小值
Math.pow(x,y)	求x的y次方 也就是y个x相乘
Math.sqrt(x)	求算术平方根 比如4的算术平方根是2
Math.random()	生成一个随机小数，取值范围是范围[0, 1) 左闭右开 $0 \leq \text{随机小数} < 1$

举例:

```
1 <script type="text/javascript">
2 // Math对象中的Math是数学的意思
3 // 跟数学相关的运算就可以使用Math对象提供的方法(比如求绝对值，取整,四舍五入)
4 // Math对象不需要创建 也就是不需要new Math() 直接使用即可！
5 // Math关键字就是对象名
6
7 // Math的属性
8 console.log("圆周率:" + Math.PI );
```

```
9  console.log("");
10
11  // Math的方法
12  // Math.ceil(x) 对x进行向上取整 得到一个比当前数要大的最小整数
13  console.log( Math.ceil(5.6) );// 6
14  console.log( Math.ceil(5.0) );// 5
15  console.log( Math.ceil(-5.5) );// -5
16  console.log("");
17
18  // Math.floor(x) 对x进行向下取整 得到一个比当前数要小的最大整数
19  console.log( Math.floor(5.1) );// 5
20  console.log( Math.floor(5.9) );// 5
21  console.log( Math.floor(-5.5) );// -6
22  console.log("");
23
24  // Math.round(x) 对x进行四舍五入
25  console.log( Math.round(5.1) );// 5
26  console.log( Math.round(5.9) );// 6
27  console.log( Math.round(-5.5) );// -5
28  console.log("");
29
30  // Math.abs(x) 返回x的绝对值 绝对值只有正数,没有负数,0还是得到0
31  console.log( Math.abs(5) );
32  console.log( Math.abs(-6) );
33  console.log( Math.abs(0) );
34  console.log("");
35
36  // Math.max(x1,x2,x3...) 求x1,x2,x3...这些数中的最大值
37  console.log( Math.max( 10,5,7,8,9 ) );
38
39  // Math.min(x1,x2,x3...) 求x1,x2,x3...这些数中的最小值
40  console.log( Math.min( 10,5,7,8,9 ) );
41  console.log("");
42
43  // Math.pow(x,y) 求x的y次方 也就是y个x相乘
44  console.log( Math.pow(5,2) );
45  console.log( Math.pow(5,-2) );
46  console.log( Math.pow(3,3) );
47  console.log("");
48
49  // Math.sqrt(x) 求算术平方根 比如4的平方根是2,比如9的平方根是3
```

```

50 console.log( Math.sqrt(9) );
51 console.log( Math.sqrt(25) );
52 console.log( Math.sqrt(3) );
53 console.log( Math.sqrt(6) );
54 console.log( Math.sqrt(16) );
55 </script>

```

Math.random() 生成随机数 难点

随机返回一个小数，取值范围是范围[0, 1) 左闭区间右开区间 $0 \leq x < 1$

如果想要得到10~20之间的随机整数 或者是20~30之间的随机整数!

如何通过上面的方法可以得到我们想要的指定区间的随机数!

原始值		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
10~20	*10+10	10	11	12	13	14	15	16	17	18	19
20~30	*10+20	20	21	22	23	24	25	26	27	28	29
公式											
Math.floor(Math.random()*(最大值-最小值)+最小值) 包含最小值但不包含最大值											
Math.floor(Math.random()*(最大值-最小值+1)+最小值) 包含最小值也包含最大值											

Math案例

课堂案例 求10-20(包含10和20)之间的随机整数

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <title>Document</title>
6 <script type="text/javascript">
7 // Math.random() 生成介于0和1之间(含 0, 不含 1)所有小数
8 // [0,1) 左闭区间,右开区间
9 // 开区间指的是区间边界的两个值不包括在内: (a,b)
10 // 闭区间指的是区间边界的两个值包括在内。[a,b]
11 // 半开半闭区间:开区间一边的边界值不包括在内,而闭区间一边的边界值包括在内。
12 [a,b)、(a,b]

```



```

13 // console.log( Math.random() );
14
15 // 如果我们想随机得到10~20之间的所有整数,包括10跟20
16 /*
17 Math.random() => [0,1)
18 Math.random()*10 => [0,10)
19 Math.random()*10+10 => [10,20)
20
21 Math.random()*11 => [0,11)
22 Math.random()*11+10 => [10,21)
23
24 但是我们要的是整数,不是小数 Math.floor()向下取整,得到比自己小的整数
25 Math.floor( Math.random()*11+10 ) => 得到10~20之间所有整数
26 */
27
28 // console.log( Math.random()*10+10 );
29 // console.log( Math.floor(19.8789) );// 19
30
31 // console.log( Math.floor( Math.random()*11+10 ) );
32
33
34 // 我们可以使用以下代码测试
35 /*while(true){
36 var num = Math.floor( Math.random()*11+10 );
37 console.log( num );
38 if(num == 10 || num == 20){
39 break;
40 }
41 }*/
42
43 // 公式,要求记住
44 // Math.floor( Math.random()*(最大值-最小值)+最小值 ) 包含最小值但不包含最大值
45 // Math.floor( Math.random()*(最大值-最小值+1)+最小值 ) 包含最小值也包含最大值
46
47 console.log( Math.floor(Math.random()*(20-10+1)+10) );
48 </script>
49 </head>
50 <body>
51

```

```
52 </body>
53 </html>
```

练习: 随机生成rgb颜色值 可以实现刷新浏览器改变网页背景颜色

使用以下代码可以设置body的背景颜色

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6 </head>
7 <body>
8   <!-- 刷新随机生成rgb颜色值,script暂时需要写在body中或者body下面 -->
9   <!-- rgb的颜色取值范围rgb(0~255,0~255,0~255) -->
10  <script type="text/javascript">
11    document.body.style.backgroundColor = "rgb(40,50,7)";
12  </script>
13 </body>
14 </html>
```

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6 </head>
7 <body>
8   <!-- 刷新随机生成rgb颜色值,script暂时需要写在body中或者body下面 -->
9   <!-- rgb的颜色取值范围rgb(0~255,0~255,0~255) -->
10  <script type="text/javascript">
11    // 一般得到指定返回的随机数这个功能是非常常见,所以我们会写在一个函数里面,专门
    得到一个指定范围的随机整数
12    function getRandom(min,max){
13      return Math.floor( Math.random()*(max-min+1) + min )
14    }
15
16    // 我们的rgb取值范围是0~255
17    var r = getRandom(0,255);
18    var g = getRandom(0,255);
19    var b = getRandom(0,255);
```

```
20 // console.log( r,g,b);  
21  
22 document.body.style.backgroundColor = "rgb("+r+", "+g+", "+b+")";  
23 </script>  
24 </body>  
25 </html>
```

今日总结

xmind要做

今日作业

在作业文件夹中