JS之基础

1. 如何插入js代码：写在<script></script>之间。

文件中引入：<script src="script.js"></script>

放在<head>部分——最常用的方式是在页面中head部分放置<script>元素，浏览器解析head部分就会执行这个代码，然后才解析页面的其余部分。

放在<body>部分——JavaScript代码在网页读取到该语句的时候就会执行。

1. 变量：变量必须使用字母、下划线(\_)或者美元符($)开始；然后可以使用任意多个英文字母、数字、下划线(\_)或者美元符($)组成。在JS中区分大小写。声明变量var 变量名
2. 数据类型：number、string、boolean、null、undefined、object（函数、数组。。。）

隐式转换：null==undefined、 number==string转number 、boolean==？转number、object==number|string尝试对象转为基本类型。其他为false。

Typeof xx 返回类型字符串：如typeof true 🡪”boolean” 。 比较适合基本类型及function检查，遇到Null失败

Instanceof 判断对象是属于什么类型。比较适合自定义对象，在不同的iframe和window间检测失效。

Object.prototype.toString比较适合内置对象和基元类型，遇到null和undefine失效

1. 数组类型：元素可以是不同类型

创建数组：var array = [元素一,元素二,…..]

var array = new Array();array[0]=1;

var array = new Array(1)

var array = new Array(元素一,元素二，元素三,…..)

访问数组元素：array[n]

数组元素增加：动态数组，添加时可以通过array.push(元素)或者array[n]=元素，在尾部添加了一个元素。arr.unshift(0)，在头部添加一个元素。

删除数组元素：delete array[n]:数组长度不变，只是相应的元素变成undefine，arr.pop()在尾部删除一个元素，返回这个元素。arr.shift()在头部删除一个元素。

数组遍历：for(;;)；或者for(下标 in arr){};

数组方法:都是Array.prototype.menthod的方法

Join如：var a = [1,2,3];a.join()//1,2,3;a.join(“\_”)//1\_2\_3合并元素为串

Reverse如var a =[1,2,3];a.reverse()//[3,2,1]倒序

Sort如arr.sort()排序

Concat拼接

Slice(firstIndex,lastIndex)(左闭右开)取片段

以下都是EC5的方法

ForEach(function)遍历

Map(function)计算

Filter()筛选

Every()判断每个元素是否符合一定条件

some()判断一些元素是否符合一定条件

reduce()计算

indexOf()；lastIndexOf()查找元素

1. In运算符：属性 in 对象 ，返回true表示有。
2. Delete运算符：delete 对象.属性。
3. 运算符：算法运算、比较运算、逻辑运算、
4. 判断语句：

if(条件)

{ 条件成立时执行的代码 }

else if (条件)

{代码}

else

{ 条件不成立时执行的代码 }

Switch语句

1. 循环语句：for循环 while循环
2. Try {}catch()finally{}:异常捕获语句
3. 函数

函数声明：

function 函数名()

{

函数代码;

}

函数表达式：

var add = function(){};

函数调用：函数名();

函数也是一种对象；

12、this：全局this，相当于window。this.a=window.a

一般函数的this仍然指的是window。

Function a(){

Return this;

}

a()===window

作为对象方法的函数中的this指的是这个包含这个函数的对象。

call方法和apply方法：调用call或apply方法的第一个参数为传入的代表this的对象

function add (c,d){

return this.a+this.b+c+d

}

Var o = {a:1,b:2}

Add.call(o,2,3)//1+2+2+3

Add.apply(o,[10,20]);//1+2+10+20

Bind方法：1）绑定this

Function foo(){

This.b = 100;

Return this.a

}

Var func = foo.bind({a:1});

Func();//1

2）科里化：公共一部分

Function getAttribute(age,sex,name){

}

Var defaultGetAttribute = getAttribute.bind(null,13,’woman’);

defaultGetAttribute（‘张三’）

JS之基本对象

1. 定义：js中所有事物都是对象，每个对象带有属性和方法。访问属性-对象名.属性名或对象名[‘属性名’]；访问方法-对象名.方法名()；
2. Date日期对象：new Date()；
3. String字符串对象
4. Math数学对象
5. Array对象

JS之自定义对象

1. 创建对象

1）var name={};

name.属性=value;

name[‘属性’]=value;

2）var name = new 对象名()；会继承原型链上的属性。并在访问时会去原型链查找

原型链：原型对象组成的链，最上面的为Object.prototype,。给原型对象赋值：原型对象.prototype.属性名=value。

3）var 对象 = Object.create(对象)；其中里面的对象里面的属性均为原型对象属性。

对象标签

Proto：指向prototype属性，原型链一直到Object.prototype

Class：Object.prototype.toString可以查看到类型

Extensible:对象是否可扩展。默认为可扩展Object.isExtensible(对象)为True。Object.preventExtensible。

对象序列化（转为string）：JSON.stringify(对象)；反序列化（转为对象）JSON.parse(字符串)

属性读写：

name.属性名

name[‘属性名’]

get/set方法：get 属性名(){….};set 属性名(参数){…..}。可以定义在设置或者读取属性值时的执行逻辑。name.属性名访问时会调用get方法；name.属性名=value会调用set方法

属性删除：

Delete name.属性名

属性检测

属性 in 对象变量

对象名.hasOwnProperty(属性名)：属性是否为自有属性

属性可枚举（为true可以遍历出这个属性，false就不可遍历出这个属性）

对象变量.propertyIsEnumerable（属性）

创建枚举或不可枚举属性

Object.defineProperty(对象变量,’属性名’,{enumerable:false,value:1000})

属性标签

Writale：属性值能否修改；enumerable属性是否可被遍历；configurable属性是否可以被修改被删除等

查看属性标签：Object.getOwnPropertyDescriptor({对象：属性值}，‘属性名’)

创建带标签的属性

Object.defineProperty(对象名，’属性名’,{configurable:,writable:,enumerable:,value:})

Object.defineProperties(对象名，{属性名:{configurable:,writable:,enumerable:,value:}, 属性名:{configurable:,writable:,enumerable:,value:},..)

1. 的

JS之浏览器对象

1. window对象是bom的核心，指当前的浏览器窗口。



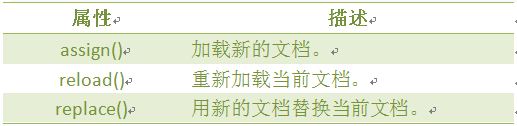
1. 计时器：

setTimeout在指定的延迟时间之后来执行代码；setTimeout(代码，交互时间)

setInterval每隔指定的时间执行代码。setInterval(代码，交互时间)

1. Window.History对象记录用户曾经浏览过的页面（URL），并可以实现浏览器前进与后退相似导航的功能。History.back()后退；history.forward()前进，相当于history.go(1)；
2. Location对象用于获取或设置窗体的URL，并且可以用于解析URL。Location.[属性或方法]





1. Navigator对象包含有关浏览器的信息，用于检测浏览器
2. Screen对象用于获取用户的屏幕信息。Screen.height高度；screen.width宽度；screen.availWidth可用宽度；screen.availHeight可用高度；

JS之操作

1. 输出内容：document.write() 可用于直接向 HTML 输出流写内容。简单的说就是直接在网页中输出内容。
2. 消息对话框：alert(字符串或变量);
3. confirm 消息对话框通常用于允许用户做选择的动作，如：“你对吗？”等。弹出对话框(包括一个确定按钮和一个取消按钮)。

confirm(str);

参数说明:

str：在消息对话框中要显示的文本

返回值: Boolean值

返回值:当用户点击"确定"按钮时，返回true；当用户点击"取消"按钮时，返回false

1. prompt弹出消息对话框,通常用于询问一些需要与用户交互的信息。弹出消息对话框（包含一个确定按钮、取消按钮与一个文本输入框）。
2. prompt(str1, str2);

参数说明：

str1: 要显示在消息对话框中的文本，不可修改

str2：文本框中的内容，可以修改

返回值:1）点击确定按钮，文本框中的内容将作为函数返回值；2）点击取消按钮，将返回null

1. window.open([URL], [窗口名称], [参数字符串])方法打开新窗口,可以查找一个已经存在或者新建的浏览器窗口。

URL：可选参数，在窗口中要显示网页的网址或路径。如果省略这个参数，或者它的值是空字符串，那么窗口就不显示任何文档。

窗口名称：\_blank：在新窗口显示目标网页；\_self：在当前窗口显示目标网页；\_top：框架网页中在上部窗口中显示目标网页

参数字符串：可选参数，设置窗口参数，各参数用逗号隔开。

1. 事件：用户操作，可以被js侦测到的行为，调用相关的函数或者程序处理。
2. 鼠标单击事件(onclick)：通常用于按钮。
3. 鼠标经过事件（onmouseover）：鼠标移到一个对象时，就触发。
4. 鼠标移开事件（onmouseout）：鼠标移开一个对象时，就触发。
5. 光标聚焦事件（onfocus）：光标移动文本框内，就触发onfocus事件。
6. 光标失焦事件（onblur）：失去焦点时触发。
7. 内容选择事件（onselect）：文本框内的文字被选中时，触发。
8. 内容改变事件（onchange）：改变文本框内容来触发onchange()
9. 加载事件 (onload)：加载页面时，立即发生，同时执行被调用的程序。注意：1）加载页面时，触发onload事件，事件写在<body>标签内；2）此节的加载页面，可理解为打开一个新页面时。

一般在页面js中，window.onload()=function{…..}写在开始，表示在页面加载完成时会触发这些操作；原因，防止在其他函数调用时，页面还没加载到而报错。

JS之DOM

1. 文档对象模型DOM（Document Object Model）定义访问和处理HTML文档的标准方法。DOM 将HTML文档呈现为带有元素、属性和文本的树结构（节点树）



1. HTML文档可以说由节点构成的集合，三种常见的DOM节点:

元素节点：上图中<html>、<body>、<p>等都是元素节点，即标签。

文本节点:向用户展示的内容，如<li>...</li>中的JavaScript、DOM、CSS等文本。

属性节点:元素属性，如<a>标签的链接属性href="http://www.imooc.com"。

节点属性：在文档对象模型(DOM)中，每个节点都是一个对象。DOM 节点有三个重要的属性 ：

1.nodeName:节点的名称

2.nodeValue:节点的值

3.nodeType:节点的类型

一、nodeName属性:节点的名称，是只读的。

1.元素节点的nodeName与标签名相同

2.属性节点的nodeName是属性的名称

3.文本节点的nodeName永远是#text

4.文档节点的nodeName永远是#document

二、nodeValue属性:节点的值

1.元素节点的nodeValue是 undefined 或 null

2.文本节点的nodeValue是文本自身

3.属性节点的nodeValue是属性的值

三、nodeType属性:节点的类型，是只读的。以下常用的几种结点类型:

元素类型 节点类型

元素 1

属性 2

文本 3

注释 8

文档 9

访问子节点：elementNode.childNodes返回子节点列表数组，也有length属性；

node.firstChild访问数组的第一个节点；node. lastChild访问数组的最后一个节点

访问父节点：elementNode.parentNode返回父节点对象；

访问兄弟节点：nodeObject.nextSibling访问紧跟的节点；nodeObject.previousSibling访问之前的节点；

创建元素节点：document.createElement(tagName)方法可创建元素节点。此方法可返回一个Element 对象。

创建文本节点：document.createTextNode(data) 方法创建新的文本节点，返回新创建的 Text 节点。

添加节点：

在指定节点的最后一个子节点列表之后添加一个新的子节点：node.appendChild(newnode)

在已有的子节点前插入一个新的子节点：insertBefore(newnode,node); newnode- 要插入的新节点。

删除子节点：removeChild()方法从子节点列表中删除某个节点。如删除成功，此方法可返回被删除的节点，如失败，则返回NULL。

替换子节点：node.replaceChild(newnode,oldnew)实现子节点(对象)的替换。返回被替换对象的引用。

1. 通过ID获取元素document.getElementById(“id”);或者elementNode. getElementById(“id”)­——表示在这个元素下面找。
2. 通过name获取元素document.getElementByName(“name”);
3. 通过TagName获取元素document.getElementByTagName(“tagname”);
4. 元素属性：通过元素节点的属性名获取属性值elementNode.getAttribute(name)；

elementNode.setAttribute(name,value)设置属性值；

1. innerHTML属性用于获取或替换HTML元素的内容。Object.innerHTML

var a = document.getElementById(“id”);a.innerHTML;a.innerHTML=”内容”;

1. 改变HTML样式：Object.style.属性名=属性值;

elementNode.style[name]=value

style只是控制行间样式（即显示设置在标签的style属性上的样式）；

1. 显示和隐藏（display属性）：Object.style.display = value；value取值:none不显示，block显示为块级元素。
2. className 属性设置或返回元素的class 属性：object.className = classname

获取元素的class 属性；为网页内的某个元素指定一个css样式来更改该元素的外观