2017.6.10

一:

|---常见的编程思想/软件开发思想:

1.面向对象

特点: 以对象为中心的编程思想.实际为 将需要研究的事物都当做对象,然后抽象对象到程序中,通过系统模拟现实对象,操作模拟的对象实现需求.

2.面向过程

特点: 以过程为中心的编程思想. 实际为 分析解决需求需要哪些步骤,然后通过函数/方法实现这些步骤,最后按照步骤先后顺序执行这些函数.

面向对象中通常 类[包含方法]作为基本的功能单元,面向过程中通常以函数/方法作为基本的功能单元.

|---JS:

JS是一种基于对象和事件驱动的客户端脚本语言.[动态 ,弱类型]

JS属于面向对象开发思想.

|---JS类:

JS类的编写实际与 方法/函数 格式相同.

|---方法:

方法是由一段代码组成,完成特定的功能.

|----参数:

参数是由 被调用方法外 向 方法内 传入数据 一种方式

JS参数格式: 参数名1,参数名2,...

Java参数格式: 参数类型 参数名,参数类型 参数名,...

参数分类:

形参: 通常指方法声明时 参数的编写. 特点: 无法确定参数值

实参: 通常指调用方法时 传入的数据. 特点: 可以确定参数值

|---字符串/字符:

字符: 指有且只有1个符号.

字符串: 指 含有1个或多个符号.

在JS语言中,无论是字符串/字符都可以使用 单引号或双引号包裹

<script type="text/javascript">

/\* JS方法格式:

\* function 方法名([参名1,参名2,..]){

\* 方法体

\* }

\* 说明:

\* 1.JS是一种动态的,弱类型语言. 参数不需要说明数据类型,当传入数据时系统自动处理.

\* 2.JS相对其他语言,方法声明中不能编写返回值类型,但是可以在方法体中返回数据.

\* 延伸:

\* Java方法格式:

\* 权限 [static/native/abstract等等] 返回值类型 方法名([参数类型 参名,参数类型 参名,..]){

\* 方法体

\* }

\*/

function test1(){

//弹出警告窗 ,并显示信息

alert("确定1 被点击了!");

}

function test2(name){//形参

alert("确定2 被点击了! 姓名:"+name);//name引用参数name的具体数据

}

</script>

</head>

<body>

<!-- 最简单的函数 -->

<input type="button" value="确定1" onclick="test1()"><br><!-- 指定点击事件产生后,执行的js代码 -->

<!-- 带有参数的函数 -->

<input type="button" value="确定2" onclick="test2('admin123')"><br>

</body>

</html>

二:

处理原则:

Java参数处理原则: java中调用含有参数的方法时,必须按照参数的个数,顺序,类型传入数据.

JavaScript参数处理原则: 若传递参数多了,多余的参数将自动忽略.若传递参数少了,且方法体中没有使用,将没有影响,否则少传递的参数值为undefined.

|----返回值:

返回值是 由 被调用方法内 向 方法外传递数据 的 一种方式

返回值格式:

return 数据;

<script type="text/javascript">

function test1(){

//弹出警告窗 ,并显示信息

alert("确定1 被点击了!");

}

function test2(name,age){//形参

//alert("确定2 被点击了! "+name);//name引用参数name的具体数据

alert("你好--test2()");

alert(name);

alert(age);

}

//计算语文和数学的总成绩

function test3(num1,num2){

var result= num1+num2;

return result;//将数据返回到方法调用处

}

//根据总成绩判断学生是否需要重考

function temp(){

var r= test3(60,70);//调用test3()

if(r>120){

alert("恭喜您,考试通过!");

}else{

alert("恭喜您,重考!");

}

}

</script>

</head>

<body>

<!-- 最简单的函数 -->

<input type="button" value="确定1" onclick="test1()"><br><!-- 指定点击事件产生后,执行的js代码 -->

<!-- 带有参数的函数 -->

<input type="button" value="确定2" onclick="test2('admin123')"><br>

<!-- 带有返回值的函数 -->

<input type="button" value="确定3" onclick="test3()"> <input type="button" value="temp" onclick="temp()"><br>

</body>

</html>

三:

<title>数据拼接.html</title>

<script type="text/javascript">

function test(){

/\*

\* alert("警告信息"): 弹出一个窗口,并显示指定的警告信息.

\*

\*/

//弹出一条信息

alert('你好');

//弹出变量的值

var i=23;

alert(i);//引用变量i的值,放入到 小括号中

//弹出一条信息及变量值: + 进行拼接

alert("警告框 " + i);//警告框 23

//弹出多组信息

alert("你好"+" how are you"+i+"");

}

//折行:”\n”

</script>

</head>

<body>

数据拼接 <br>

<input type="button" value="确定" onclick="test()">

</body>

</html>

四:

\* 函数直接量语法:

\* var 变量名=function [方法名]([参名1,参名2,..]){

\* 方法体

\* }

var test4=function (){

alert("函数直接量: test4()");

}

/\* 方法调用:

\* 通常一个复杂的功能,需要多个方法支持,共同实现或完成该功能.

\*

\* 方式:

\* a. 方法名([值1,值2,..]); //直接调用: 两个方法必须在同一个HTML文件中

\* b. new 方法名([值1,值2,..]);//将方法/函数 看做特殊对象使用. new中文意思: 新建 ,创建. 通常用于创建对象.

\*/

function test5(){

//test1();

alert("test5()");

//调用 test1()

//test1();

// test1();

new test1();

//test();

}

//只能将函数直接量语法的函数 放在普通函数中,不能反向存放.

function test6(){

alert("test6()");

var test=function(num){

//弹出警告窗 ,并显示信息

alert("test 被点击了! "+num);

}(3);

//test(23);

}

</script>

</head>

<body>

<!-- 函数直接量 -->

<input type="button" value="确定4" onclick="test4()"><br>

<!-- 函数的调用 -->

<input type="button" value="确定5" onclick="test5()" ><br>

<!-- 函数直接量与普通函数[函数嵌套] -->

<input type="button" value="确定6" onclick="test6(66)" ><br>

</body>

</html>

五:

HTML注释: <!-- HTML注释 -->

CSS注释: /\* CSS注释 \*/

JS注释: //单行注释 /\* 多行注释 \*/

HTML标签不区分大小写,属性值区分大小.

CSS在HTML中不区分大小写,XHTML中区分大小写.

JS严格区分大小写.

<title>typeof.html</title>

<script type="text/javascript">

function test1(){

/\*var i=34;

alert(typeof(i));\*/

/\*var i="zhangsan";

alert(typeof(i));\*/

/\*var i=true;

alert(typeof(i));\*/

/\*var i=null;

alert(typeof(i));\*/

var i=new String("ABC");

alert(i+" "+typeof(i));

}

function test2(){

//获得文本框对象

var user=document.getElementById("username");

//获得文本框的内容

var userVal=user.value;

//测试类型

alert(userVal+" "+typeof(userVal));

// 文本框中的数据类型始终为string,若不写任何内容,文本框值为"".

}

function test3(num){

//测试数据类型

var i;

alert("i:"+typeof(i));

alert("j:"+typeof(j));

//访问数据

alert("num:"+num);

alert("j:"+j);

}

</script>

</head>

<body>

<!--

JS中的数据划分为:

数字(number):[整数,小数]

字符串 (string String对象)[所有数据都必须被引号包裹]

布尔(boolean): 逻辑值[true false]

null: 表示没有对象,由历史原因JS中将null划分为object.

object: 对象

undefined: 声明的变量没有赋值时 或 没有声明的变量

-->

测试js中的数据类型 <br>

<input type="text" id="username"><br>

<!-- 测试数据类型 -->

<input type="button" onclick="test1()" value="确定1">

<!-- 测试文本框内容的数据类型 -->

<input type="button" onclick="test2()" value="确定2">

<!-- 说明引用未声明的变量,为何报错 -->

<input type="button" onclick="test3()" value="确定3">

</body>

</html>

六:

<script type="text/javascript">

function test1() {

var i=23;//将23 存放到 容器名为 i的容器[变量]中

alert("i="+i);

alert("i+3="+(i+3));

var j;

//j=-1;

alert("j="+j);

}

function test2() {

var i;//创建变量

//变量属于容器,对于容器操作分为: 查看 修改 赋值 删除

//赋值: 变量名=值;

i="张三";

//查看: 通过变量名,访问容器中的数据

alert("i="+i);

//修改: 变量名=值;

i="小丽";

i=23;

alert("i="+i);

//删除: 变量名=null; null表示没有数据 ,没有对象

i=null;

alert("i="+i);

}

function t() {

var t\_var0 = "我是t中的变量t\_var0";

t\_var1 = "我是t中的变量t\_var1";

alert(t\_var0)

alert(t\_var1)

}

function test3() {

//alert("引用t\_var0:"+t\_var0);

//alert("引用t\_var1:"+t\_var1);

alert("num:"+num);

}

var num=23;//全局代码,在加载网页时就会执行

/\*

\* 局部变量:

\* 全局变量:

\*/

function test4() {

const i=1;

alert("i="+i);

/\*i=-1;

alert("i="+i);\*/

}

</script>

</head>

<body>

<!--

为什么使用变量?

通常程序中会含有多个模块,不同模块之间会有相同数据使用. 为了方便数据的使用,要求将数据存放到容器中.

常见容器?

变量,数组,对象

变量: 变量是JS中最基本的存储单元,可以存储一个数据.

变量的使用:

A. var 变量名=值;

说明:

1. var是JS中的关键字,表示"变量". 用于创建变量[声明变量]

2. JS是弱类型语言,同一个变量中可以存放多种类型的数据.

3. 以上代码表示创建容器,并将指定数据放入容器中

B. var 变量名; //创建变量[容器]

变量名=值; //将指定数据 存放到 容器中

说明:

1. 以上方式先创建变量,然后根据需求 存放数据.

延伸:

java属于强制类型: 变量类型 变量名=值;

变量分类:

根据变量的有效范围,分为: 全局变量[成员变量] 和 局部变量

全局变量: 变量可以在任何方法体中被访问/引用

要求: 变量必须编写在方法体之外 或 变量不使用var修饰

局部变量: 变量只能在当前方法体中被访问/引用

要求: 变量必须编写在方法体之内,且使用var修饰

常量: 可以存放一个数据,数据确定以后将不能更改.

格式: const 常量名=值;

说明: JS有常量的支持,但是不同浏览器兼容不同.

-->

<!-- 编写一个简单的变量 -->

<input type="button" value="确定1" onclick="test1()">

<br>

<!-- 操作变量 -->

<input type="button" value="确定2" onclick="test2()">

<br>

<!-- 编写全局变量和局部变量 -->

<input type="button" value="确定3" onclick="test3()">

<br>

<!-- 辅助按钮 -->

<input type="button" value="确定4" onclick="t()">

<br>

<!-- 编写常量,常量的特性和兼容性 -->

<input type="button" value="常量" onclick="test4()">

<br>

</body>

</html>