window.open()支持环境： JavaScript1.0+/JScript1.0+/Nav2+/IE3+/Opera3+

基本语法：

window.open(pageURL,name,parameters)

其中：

pageURL 为子窗口路径

name 为子窗口句柄

parameters 为窗口参数(各参数用逗号分隔)

示例：

<SCRIPT>

<!--

window.open ('page.html','newwindow','height=100,width=400,top=0,left=0,toolbar=no,menubar=no,scrollbars=no, resizable=no,location=no, status=no')

//写成一行

-->

</SCRIPT>

脚本运行后，page.html将在新窗体newwindow中打开，宽为100，高为400，距屏顶0象素，屏左0象素，无工具条，无菜单条，无滚动条，不可调整大小，无地址栏，无状态栏。

上例中涉及的为常用的几个参数，除此以外还有很多其他参数，如下所示：

各项参数

其中yes/no也可使用1/0；pixel value为具体的数值，单位象素。

|  |  |  |
| --- | --- | --- |
| 参数 | 取值范围 | 说明 |
| alwaysLowered | yes/no | 指定窗口隐藏在所有窗口之后 |
| alwaysRaised | yes/no | 指定窗口悬浮在所有窗口之上 |
| depended | yes/no | 是否和父窗口同时关闭 |
| directories | yes/no | Nav2和3的目录栏是否可见 |
| height | pixel value | 窗口高度 |
| hotkeys | yes/no | 在没菜单栏的窗口中设安全退出热键 |
| innerHeight | pixel value | 窗口中文档的像素高度 |
| innerWidth | pixel value | 窗口中文档的像素宽度 |
| location | yes/no | 位置栏是否可见 |
| menubar | yes/no | 菜单栏是否可见 |
| outerHeight | pixel value | 设定窗口(包括装饰边框)的像素高度 |
| outerWidth | pixel value | 设定窗口(包括装饰边框)的像素宽度 |
| resizable | yes/no | 窗口大小是否可调整 |
| screenX | pixel value | 窗口距屏幕左边界的像素长度 |
| screenY | pixel value | 窗口距屏幕上边界的像素长度 |
| scrollbars | yes/no | 窗口是否可有滚动栏 |
| titlebar | yes/no | 窗口题目栏是否可见 |
| toolbar | yes/no | 窗口工具栏是否可见 |
| Width | pixel value | 窗口的像素宽度 |
| z-look | yes/no | 窗口被激活后是否浮在其它窗口之上 |

用函数控制弹出窗口

下面是一个完整的代码。

<html>

<head>

<script LANGUAGE="JavaScript">

function openwin() {

window.open ("page.html", "newwindow", "height=100, width=400, toolbar =no, menubar=no, scrollbars=no, resizable=no, location=no, status=no") //写成一行

}

</script>

</head>

<body onload="openwin()">

任意的页面内容...

</body>

</html>

这里定义了一个函数openwin(),函数内容就是打开一个窗口。怎么调用呢？

方法一：

<body onload="openwin()">

浏览器读页面时弹出窗口；

方法二：

<body onunload="openwin()">

浏览器离开页面时弹出窗口；

方法三：

用一个连接调用：

<a href="#" onclick="openwin()">打开一个窗口</a>

注意：使用的“#”是虚连接。

方法四：

用一个按钮调用：

<input type="button" onclick="openwin()" value="打开窗口">

如何实现在不使用window.showModalDialog 的情况下用 window.open方式 向父窗口返回值。

例如： 页面AAA.htm 用 window.open方式弹出页面 BBB.htm 。

在页面BBB.htm上选择一个值，确定关闭窗口后将选择的这个值返回到父窗口AAA.htm。

AAA.htm得到返回的值后，给本页面上的文本框赋值。

BBB.htm页面中加入下面代码：

window.opener.document.getElementById("theTextAreaId").value = document.getElemnetById("theSelectId").value ;

window.opener 的用法

window.opener 返回的是创建当前窗口的那个父窗口的引用，比如点击了a.htm上的一个链接而打开了b.htm，然后我们打算在b.htm上输入一个值然后赋予a.htm上的一个id为“name”的textbox中，就可以写为：

window.opener.document.getElementById("name").value = "输入的数据";

对于javascript中的window.opener没有很好的理解。

为什么框架中不能使用，弹出窗口的父窗口不能在框架里面的某个页面呢？那怎样通过弹出窗口操作框架中的父窗口呢？

opener.parent.frames['frameName'].document.all.input1.value

即opener这个对象为前一个窗口，可以使用window.opener.document...调用document的相关方法，例如下面的例子，插入一些table行到前一个窗口：

function taletoTb(itemStr) {

newRow = opener.document.all.itemTb.insertRow(opener.document.all.itemTb.rows.length);

rowCnt = opener.document.all.itemTb.rows.length;

newCell = newRow.insertCell();

newCell.insertAdjacentHTML('BeforeEnd','<div align="center">'+itemCode+'</div>');

newCell = newRow.insertCell();

newCell.insertAdjacentHTML('BeforeEnd','<div align="center">'+itemName+'</div>');

newCell = newRow.insertCell();

newCell.insertAdjacentHTML('BeforeEnd','<div align="center"><img src="../../../images/delete2.gif" width="12" height="12" onClick="userRightDel(this);"><input type="hidden" name="itembox" value='+itemCode+'></div>');

}

缩略显示confirm用法和例子

一般用于弹出对话框（确定／否）

确定：就执行其嵌套的内容；否：则反之

<script language="javascript">

//验证时间格式 YYYY-MM-DD/YYYY,MM,DD

function isDate(date){

var regu = "^[0-9]{4}-([0-1]?)[0-9]{1}-([0-3]?)[0-9]{1}$";

var re = new RegExp(regu);

if (date.search(re) != -1)

return true;

else

return false;

}

function sureButton(){

if(!confirm('真的要删除吗？删除后将无法恢复！')){

return;

}

//验证时间格式 YYYY-MM-DD

var startDate=document.getElementById("startDate").value;

var endDate=document.getElementById("endDate").value;

if(!isDate(startDate)){

alert(startDate+"请输入正确的开始日期格式!如:(YYYY-MM-DD)2008-01-01");

return document.getElementById("startDate").focus();

}

if(!isDate(endDate)){

alert("请输入正确的结束日期格式!如:(YYYY-MM-DD)2008-01-01");

return document.getElementById("endDate").focus();

}

if(startDate==""){

alert("请输入开始日期");

return document.getElementById("startDate").focus();

}

if(endDate==""){

alert("请输入结束日期");

return document.getElementById("endDate").focus();

}

startDate=startDate.replace(new RegExp('-', 'g'),'/');

alert(startDate);

endDate=endDate.replace(new RegExp('-', 'g'),'/');

var startTime=new Date(startDate).getTime();

alert(new Date(startDate).getTime());

var endTime=new Date(endDate).getTime();

if((endTime-startTime)<0){

alert("结束日期必须大于开始日期");

return document.getElementById("endDate").focus();

}

}

</script>

<html>

<head>

<meta http-equiv="Content-Language" content="zh-cn">

<meta name="generator" content="Bluefish 1.0.7">

<meta name="ProgId" content="FrontPage.Editor.Document">

<meta http-equiv="Content-Type" content="text/html; charset=gb2312">

<title></title>

<style type="text/css">

table{color: #000000; font-family: 宋体; font-size: 12px; height:12 }

t1{color:#008000;align:center}

</style>

</head>

<body topmargin="0" leftmargin="0">

<div align="left">

<table border="0" cellpadding="0" cellspacing="0" style="border-collapse: collapse" bordercolor="#111111" width="461" height="8" id="AutoNumber1" background="images/kabg.gif">

<tr>

<td colspan="3">

<table cellpadding="0" cellspacing="0" style="border-collapse: collapse" bordercolor="#111111" width="100%" height="17">

<tr>

<td width="19%" height="25" align="center" style="border:1px solid #000080;"><font color="#008000">开始日期</font></td>

<td width="31%" style="border:1px solid #000080;">

<input name="startDate" type="text" id="startDate" size="15"></td>

<td width="22%" align="center" style="border:1px solid #000080;"><font color="#008000">结束日期</font></td>

<td width="28%" style="border:1px solid #000080;">

<input name="endDate" type="text" id="endDate" size="15"></td>

</tr>

</table>

</td>

</tr>

<tr>

<td height="28" align="center" style="border:1px solid #000080; " colspan="3">

<table border="0" cellpadding="0" cellspacing="0" style="border-collapse: collapse" bordercolor="#111111" width="100%" id="AutoNumber3" height="17">

<tr>

<td width="14%" align="center" height="25">

<font color="#008000">当前状态</font></td>

<td width="18%" align="center" height="25">

<select size="1" name="display">

<option value="0">无效</option>

<option value="1">等待</option>

<option value="2" selected>显示中</option>

</select></td>

<td width="15%" align="center" height="25" style="border-left: 1px solid #000080; border-right-width: 1; border-top-width: 1; border-bottom-width: 1">

<font color="#008000">Logo行宽</font></td>

<td width="10%" align="center" height="25">

<select size="1" name="colValue">

<option>1</option>

<option>2</option>

<option>3</option>

<option>4</option>

<option>5</option>

<option>6</option>

</select></td>

<td width="12%" align="center" style="border-left: 1px solid #000080; border-right-width: 1; border-top-width: 1; border-bottom-width: 1"><font color="#008000">显示顺序</font></td>

<td width="5%" >

<input name="displayOrder" type="text" id="displayOrder" size="3" value="1">

</td>

<td width="43%" align="center" height="25" style="border-left-style: solid; border-left-width: 1; border-right-width: 1; border-top-width: 1; border-bottom-width: 1">

<input onclick="cancelButton()" type="reset" value="关闭" name="B2" style="border-style: outset; border-width: 1; color:#0000FF">

<input onclick="sureButton()" type="submit" value="确定" name="B1" style="border-style: outset; border-width: 1; color:#0000FF"></td>

</tr>

</table>

</td>

</tr>

<tr>

<td height="20" colspan="3" align="center" style="border:1px solid #000080; ">

<marquee behavior="slide" style="color: #808080">::日期格式为年-月-日,直接填入图片和点击路径全名时应仔细查对是否正确::</marquee></td>

</tr>

</table>

</div>

</body>

</html>

可以批量删除表格记录行，通过checkbox选择删除的行；可以批量增加记录行，通过输入框指定行数。

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=shift-js">

<title>The page of append rows to Table</title>

<script language="JavaScript">

// 新增行

function addRow(){

var textNum = document.getElementById("rownum");

// 得到新增行记录的行数

var index = textNum.value;

if(!checknum(index)){

alert("You can only input number in the TEXT!");

textNum.focus();

textNum.select();

}

for(var i = 0; i < index; i++){

// 得到表格对象

var tableObj = document.getElementById("mainTb");

// 得到tbody对象

var tableBodyObj = document.getElementById("mainBody");

var newRowObj = document.createElement("tr");

var newCheckBox = document.createElement("td");

var newtext1 = document.createElement("td");

var newtext2 = document.createElement("td");

var newtext3 = document.createElement("td");

newCheckBox.innerHTML = '<center><input type="Checkbox" name="checkbox" onclick = "checkBoxSel()"></center>';

newtext1.innerHTML = '<input type="text" name="newCarName" size="9">';

newtext2.innerHTML = '<input type="text" name="newCarName" size="9">';

newtext3.innerHTML = '<input type="text" name="newCarName" size="9">';

// 新增的tr节点下增加td节点

newRowObj.appendChild(newCheckBox);

newRowObj.appendChild(newtext1);

newRowObj.appendChild(newtext2);

newRowObj.appendChild(newtext3);

// tbody节点下增加tr节点

tableBodyObj.appendChild(newRowObj);

}

}

// 新增行数选择框检查输入必须是数字

function checknum(strVal){

if (strVal.length != 0){

var r = strVal.match(new RegExp(/^[0-9]+$/));

if (r == null){

return false;

}else{

return true;

}

}

return true;

}

// 批量删除指定的行

function deleteRow(){

var Tblen;

// 得到所有 checkbox 对象

var checkbox = document.getElementsByName("checkbox");

//var checkboxlen = document.all.checkbox.length;

// 得到所有提交的checkbox个数

var checkboxlen = checkbox.length;

var ischecked;

// 得到删除按钮对象

var delbutton = document.getElementById("delete");

for (var i=0; i < checkboxlen; i++){

// 得到表格行数

Tblen = this.mainTb.rows.length;

// 最终保留一行记录

if (Tblen == 1)

{

document.getElementsByName("checkbox")[0].checked = false;

alert("A line have to be saved in the ID of mainTb!");

// 删除按钮不可用

delbutton.disabled = true;

return false;

}

ischecked = checkbox[i].checked;

// 如果当前的checkbox选中：删除当前节点，由于删除节点后，下面的节点上移，游标（记录行指针）也应该上移

if (ischecked)

{

// document.all("mainTb").deleteRow(i);

document.getElementById("mainTb").deleteRow(i);

// 游标（记录行指针）上移

i--;

}

// 重新统计checkbox个数

checkboxlen = checkbox.length;

}

// 删除操作后按钮状态是不可用

delbutton.disabled = true;

}

// checkbox按钮单击事件处理函数：是否至少选中一行记录，是可以删除操作；如果没有选择，删除按钮不可用

function checkBoxSel(){

// 得到删除按钮对象

var delbutton = document.getElementById("delete");

if(checkselect()){

delbutton.disabled = false;

}else{

delbutton.disabled = true;

}

}

function checkselect(){

// 得到所有 checkbox 对象

var checkbox = document.getElementsByName("checkbox");

// 得到所有提交的checkbox个数

var chklength = checkbox.length;

var flag = false;

for(var i = 0; i < chklength; i++)

{

if(checkbox[i].checked == true)

{

flag = true;

break;

}

}

if(flag == true)

{

return true;

} else {

return false;

}

}

// 页面加载body内容时执行

function loadpage(){

var delbutton = document.getElementById("delete");

var rownum = document.getElementById("rownum");

// 初始“删除”按钮不可用

delbutton.disabled = true;

// 初始行数输入框内容为空

rownum.value = "";

}

</script>

动态添加、删除行，分别通过insertRow，deleteRow方法实现，显示行号，通过rowIndex属性获得，基本可以动态实现相关功能。

<Script Language="Javascript">

var cGetRow=-99999;

var lineNo = 1;

function \_(id) {

return document.getElementById(id);

}

// The index of the row to be deleted.

// This index starts from 0 and is relative to the logical order (not document order)

// of all the rows contained inside the table.

// If the index is -1 the last row in the table is deleted.

function AddRow(){

//添加一行

var newTr = \_("tab1").insertRow();

//添加两列

var newTd0 = newTr.insertCell();

var newTd1 = newTr.insertCell();

//设置列内容和属性

newTd0.innerHTML = '<input type=checkbox id="box' + lineNo + '" onClick="GetRow()">';

// 测试动态改变innerHTML中的checkbox的id

alert(\_("box" + lineNo).id);

newTd1.innerText= '新增加行' + lineNo;

lineNo++;

}

function DelRow(iIndex){

//删除一行

if(iIndex==-99999){

alert("系统提示：没有选中行号!");

}else{

iIndex = cGetRow;

\_("tab1").deleteRow(iIndex);

}

}

function GetRow(){

//获得行索引

//两个parentElement分别是TD和TR哟，rowIndex是TR的属性

//this.parentElement.parentElement.rowIndex

cGetRow=window.event.srcElement.parentElement.parentElement.rowIndex;

}

function ShowRow(){

//显示行号

alert(cGetRow);

//alert(document.getElementsByTagName("tr").length);

}

</script>

<table id="tab1" border=1>

<tr id="tr1">

<td width=6%><input type=checkbox id="box1" onClick="GetRow()"></td>

<td id="a">第一行</td>

</tr>

<tr id="tr2">

<td width=6%><input type=checkbox id="box2" onClick="GetRow()"></td>

<td id="b">第二行</td>

</tr>

<tr id="tr3">

<td width=6%><input type=checkbox id="box3" onClick="GetRow()"></td>

<td id="c">第三行</td>

</tr>

</table>

<table id="tab1" border=1>

<tr id="tr1">

<td width=6%><input type=checkbox id="box1" onClick="GetRow()"></td>

<td id="a">统计</td>

</tr>

</table>

<input type="submit" name="Submit" value="AddRow" onclick="javascript:AddRow();">

<input type="submit" name="Submit" value="DelRow" onclick="javascript:DelRow(cGetRow);">

<input type="submit" name="Submit" value="ShowRow" onclick="javascript:ShowRow();"> 表格动态增加行

window.opener 实际上就是通过window.open打开的窗体的父窗体。

比如在父窗体parentForm里面 通过 window.open("subForm.html"),那么在subform.html中 window.opener

就代表parentForm,可以通过这种方式设置父窗体的值或者调用js方法。

如：1,window.opener.test(); ---调用父窗体中的test()方法

2,如果window.opener存在,设置parentForm中stockBox的值。

if (window.opener && !window.opener.closed) {

window.opener.document.parentForm.stockBox.value = symbol;

}

1>window.opener 的用法

在一般的用法中，只是用来解决关闭窗口时不提示弹出窗口， 而对它更深层的了解一般比较少。其 实 window.opener是指调用window.open方法的窗口。

在工作中主要是用来解决部分提交的。这种跨页操作对工作是非常有帮助的。

如果你在主窗口打开了一个页面，并且希望主窗口刷新就用这个，打开页面的window.opener就相当于

主窗口的window。

主窗口的刷新你可以用

window.opener.location.reload()；

如果你用虚拟的目录：如struts的\*.do会提示你重试

你可以改成这样 window.opener.yourformname.submit()

就好了

2〉

在应用中有这样一个情况，

在A窗口中打开B窗口，在B窗口中操作完以后关闭B窗口，同时自动刷新A窗口

function closeWin(){

hasClosed = true;

window.opener.location="javascript:reloadPage();";

window.close();

}

function window.onbeforeunload(){

if(!hasClosed){

window.opener.location="javascript:reloadPage();";

}

}

</script>

上面的代码在关闭B窗口的时候会提示错误，说缺少Object,正确的代码如下：

function closeWin(){

hasClosed = true;

window.opener.location="javascript:reloadPage();";

window.opener=null;

window.close();

}

function window.onbeforeunload(){

if(!hasClosed){//如果已经执行了closeWin方法，则不执行本方法

window.opener.location="javascript:reloadPage();";

}

}

</script>

reloadPage方法如下:

function reloadPage() {

history.go(0);

document.execCommand("refresh")

document.location = document.location;

document.location.reload();

}

PS:由于需要支持正常关闭和强制关闭窗口时能捕捉到事件，用了全局变量hasClosed

==============================================

补充，在父窗口是frame的时候在刷新父窗口的时候会出现问题:

The page cannot be refreshed without resending the information.

后修改如下:

window.opener.parent.document.frames.item('mainFrame').location.href = window.opener.location.href;

不需要执行自带的reload()方法,注意，不要再画蛇添足加上这一句:

window.opener.parent.document.frames.item('mainFrame').location.reload();

========================================================================================

最后，为了同时支持刷新普通父窗口和frame父窗口，代码如下:

function closeWin() {

hasClosed = true;

<%if(null != frame){%>

window.opener.parent.document.frames.item('mainFrame').location.href = window.opener.location.href;

<%}else{%>

window.opener.location = "javascript:reloadPage();";

<%}%>

//window.opener.top.mainFrame.location="javascript:reloadPage();";

//self.opener.frames.mainFrame.location.reload(true);

window.opener = null;

window.close();

}

function window.onbeforeunload(){

if (!hasClosed) {

<%if(null != frame){%>

window.opener.parent.document.frames.item('mainFrame').location.href = window.opener.location.href;

<%}else{%>

window.opener.location = "javascript:reloadPage();";

<%}%>

window.opener = null;

}

}

关于window.opener

window.opener 的用法

window.opener 返回的是创建当前窗口的那个窗口的引用，比如点击了a.htm上的一个链接而打开了b.htm，然后我们打算在b.htm上输入一个值然后赋予a.htm上的一个id为“name”的textbox中，就可以写为：

window.opener.document.getElementById("name").value = "输入的数据";

对于javascrīpt中的window.opener没有很好的理解。

为什么框架中不能使用，弹出窗口的父窗口不能在框架里面的某个页面呢？那怎样通过弹出窗口操作框架中的父窗口呢？

opener.parent.frames['frameName'].document.all.input1.value 试试这个：）

正确使用window.open返回对象的opener

众所周知JavaScript中:

var win = window.open(url,windowName,...); 的使用,

而win.opener则是指向父窗口的引用

然而,有种情况却比较特别,

假如有两个窗口window1和window2

按下列步骤执行:

var win = window.open(url,windowName,...);// (window1)

var win = window.open(url,windowName,...);//(window2)

其中先后这两次打开的子窗口的windowName一样

此时你会发现在window2中的win.opener却不是指向window2的,却是指向window1.

如果你想在子窗口关闭父窗口的话,就不正确了,因此可以修改上面的执行方法为:

var win = window.open(url,windowName,...);? (window1)

win.opener = window;

var win = window.open(url,windowName,...);? (window2)

win.opener = window;

只有这样修改才OK

通过window.showModalDialog或者.showModelessDialog弹出的页面

这种情况需要两个步骤：1 在父窗口.showModalDialog或.showModelessDialog方法的第二个参数传递window对象比如: window.showModelessDialog('a.htm',window);2 在a.htm中就可以通过window.dialogArguments获取该参数比如: window.dialogArguments.fun1();PS:子窗口可以通过设置window.returnValue设置页面返回值

比如: window.returnValue=’OK’;window.close();

strRtn=window.showModalDialog(......)

回答者： 啵若波罗蜜 - 六级 2008-7-22 17:02

window.opener 的用法

opener 返回的是创建当前窗口的那个窗口的引用，比如点击了a.htm上的一个链接而打开了b.htm，然后我们打算在b.htm上输入一个值然后赋予a.htm上的一个id为“name”的textbox中，就可以写为：

opener.getElementById("name"). = "输入的数据";

对于java中的opener没有很好的理解。

那怎样通过弹出窗口操作框架中的父窗口呢？

opener.parent.s['Name'].all.input1. 试试这个

==========

parent指父窗口，在SET中，的PARENT就是SET窗口。

opener指用OPEN等方式创建的新窗口对应的原窗口。

==============

top：

该变更永远指分割窗口最高层次的浏览器窗口。如果计划从分割窗口的最高层次开始执行命令，就可以用top变量。

parent：

该变量指的是包含当前分割窗口的父窗口。如果在一个窗口内有分割窗口，而在其中一个分割窗口中又包含着分割窗口，则第2层的分割窗口可以用parent变量引用包含它的父分割窗口。

附：Windwo对象、Parent对象、对象、Document对象和Form对象的阶层关系

Windwo对象→Parent对象→对象→Document对象→Form对象，如下：

parent.1.forms[0].elements[0].;