目录列表：

$()

DWRUtil.getText(id)

DWRUtil.getValue(id)

DWRUtil.setValue(id, value)

DWRUtil.getValues()

DWRUtil.setValues()

DWRUtil.addOptions and DWRUtil.removeAllOptions

DWRUtil.addRows and DWRUtil.removeAllRows

DWRUtil.onReturn

DWRUtil.toDescriptiveString

DWRUtil.useLoadingMessage

修补浏览器事件

util.js包含了一些工具函数来帮助你用javascript数据(例如从服务器返回的数据)来更新你的web页面。

你可以在DWR以外使用它，因为它不依赖于DWR的其他部分。

4个基本的操作页面的函数：getValue[s]()和setValue[s]()可以操作大部分HTML元素除了table，list和image。 getText()可以操作select list。

要修改table可以用addRows()和removeAllRows()。要修改列表(select列表和ul,ol列表)可以用 addOptions()和removeAllOptions()。

还有一些其他功能不是DWRUtil的一部分。但它们也很有用，它们可以用来解决一些小问题，但是它们不是对于所有任都通用的。

1、$()

$() 函数(它是合法的Javascript名字) 是从Protoype偷来的主意。大略上的讲： $ = document.getElementById。 因为在Ajax程序中，你会需要写很多这样的语句，所以使用 $() 会更简洁。

通过指定的id来查找当前HTML文档中的元素，如果传递给它多个参数，它会返回找到的元素的数组。所有非String类型的参数会被原封不动的返回。这个函数的灵感来至于prototype库，但是它可以在更多的浏览器上运行。

从技术角度来讲他在IE5.0中是不能使用的，因为它使用了Array.push，尽管如此通常它只是用来同engine.js一起工作。如果你不想要 engine.js并且在IE5.0中使用，那么你最好为Array.push找个替代品。

2、DWRUtil.getText(id)

getText(id)和getValue(id)很相似。它是为select列表设计的。你可能需要取得显示的文字，而不是当前选项的值。

例子：getahead.ltd.uk/dwr/browser/util/gettext

3、DWRUtil.getValue(id)

DWRUtil.getValue(id)是 setValue()对应的"读版本"。它可以从HTML元素中取出其中的值，而你不用管这个元素是select列表还是一个div。

这个函数能操作大多数HTML元素包括select(获取当前选项的值而不是文字)、input元素(包括textarea)、div和span。

4、DWRUtil.setValue(id, value)

DWRUtil.setValue(id, value)根据第一个参数中指定的id找到相应元素，并根据第二个参数改变其中的值。

这个函数能操作大多数HTML元素包括select(设置当前选项的值而不是文字)、input元素(包括textarea)、div和span。

5、DWRUtil.getValues()

getValues()和getValue()非常相似，除了输入的是包含name/value对的javascript对象。name是HTML元素的 ID，value会被更改为这些ID对象元素的内容。这个函数不会返回对象，它只更改传递给它的值。

【基于Form的getValues()】

从DWR1.1开始getValues()可以传入一个HTML元素(一个DOM对象或者id字符串)，然后从它生成一个reply对象。例子：getahead.ltd.uk/dwr/browser/util/getvalues

6、DWRUtil.setValues()

setValues()和setValue()非常相似，除了输入的是包含name/value对的javascript对象。name是HTML元素的 ID，value是你想要设置给相应的元素的值。

7、DWRUtil.addOptions and DWRUtil.removeAllOptions

【生成列表】

DWR的一个常遇到的任务就是根据选项填充选择列表。下面的例子就是根据输入填充列表。

下面将介绍 DWRUtil.addOptions() 的几种是用方法。

如果你希望在你更新了select以后，它仍然保持运来的选择，你要像下面这样做：

var sel = DWRUtil.getValue(id);

DWRUtil.removeAllOptions(id);

DWRUtil.addOptions(id, ...);

DWRUtil.setValue(id, sel);如果你想加入一个初始的"Please select..." 选项那么你可以直接加入下面的语句：

DWRUtil.addOptions(id, \["Please select ..."]);然后再下面紧接着加入你真正的选项数据。

【DWRUtil.addOptions有5种模式】

简单数组: DWRUtil.addOptions(selectid, array) 会创建一堆option，每个option的文字和值都是数组元素中的值。

简单对象数组 (指定text): DWRUtil.addOptions(selectid, data, prop) 用每个数组元素创造一个option，option的值和文字都是在prop中指定的对象的属性。

高级对象数组 (指定text和value值): DWRUtil.addOptions(selectid, array, valueprop, textprop) 用每个数组元素创造一个option，option的值是对象的valueprop属性，option的文字是对象的textprop属性。

对象: DWRUtil.addOptions(selectid, map, reverse)用每个属性创建一个option。对象属性名用来作为option的值，对象属性值用来作为属性的文字，这听上去有些不对。但是事实上却是正确的方式。如果reverse参数被设置为true，那么对象属性值用来作为选项的值。

对象的Map: DWRUtil.addOptions(selectid, map, valueprop, textprop) 用map中的每一个对象创建一个option。用对象的valueprop属性做为option的value，用对象的textprop属性做为 option的文字。

ol 或 ul 列表: DWRUtil.addOptions(ulid, array) 用数组中的元素创建一堆li元素，他们的innerHTML是数组元素中的值。这种模式可以用来创建ul和ol列表。

例子：getahead.ltd.uk/dwr/browser/lists

8、DWRUtil.addRows and DWRUtil.removeAllRows

【生成Table】

DWR通过这两个函数来帮你操作table： DWRUtil.addRows() 和 DWRUtil.removeAllRows() 。这个函数的第一个参数都是table、tbody、thead、tfoot的id。一般来说最好使用tbody，因为这样可以保持你的header和 footer行不变，并且可以防止Internet Explorer的bug。

【DWRUtil.removeAllRows()】

语法:

DWRUtil.removeAllRows(id);

描述:

通过id删除table中所有行。

参数:

id: table元素的id(最好是tbody元素的id)

【DWRUtil.addRows()】

语法:

DWRUtil.addRows(id, array, cellfuncs, [options]);

描述:

向指定id的table元素添加行。它使用数组中的每一个元素在table中创建一行。然后用cellfuncs数组中的没有函数创建一个列。单元格是依次用cellfunc根据没有数组中的元素创建出来的。

DWR1.1开始，addRows()也可以用对象做为数据。如果你用一个对象代替一个数组来创建单元格，这个对象会被传递给cell函数。

你可以写一些像这样的伪代码：

for each member in array

for each function in cellfuncs

create cell from cellfunc(array[i])

参数:

id: table元素的id(最好是tbody元素的id)

array: 数组(DWR1.1以后可以是对象)，做为更新表格数据。

cellfuncs: 函数数组，从传递过来的行数据中提取单元格数据。

options: 一个包含选项的对象(见下面)

高级选项:

rowCreator: 一个用来创建行的函数(例如，你希望tr加个css). 默认是返回一个document.createElement("tr")

cellCreator: 一个用来创建单元格的函数(例如，用th代替td). 默认返回一个document.createElement("td")

rowData: the element value from the array (the same for all cells in a row)

rowIndex: the key (if map) or index (if array) from the collection

rowNum: The row number counting from 0 in this section (so if you are using tbody, it counts rows in the tbody and not the whole table)

data: The 'computed' data value for the cell (cellCreators only)

cellNum: The cell number that we are altering counting from 0 (cellCreators only

例子：getahead.ltd.uk/dwr/browser/tables

【动态编辑表格(Dynamically Editing a Table)】

例子：getahead.org/dwr/examples/table

9、DWRUtil.onReturn

当按下return键时，得到通知。

当表单中有input元素，触发return键会导致表单被提交。当使用Ajax时，这往往不是你想要的。而通常你需要的触发一些Javscript。

不幸的是不同的浏览器处理这个事件的方式不一样。所以DWRUtil.onReturn修复了这个差异。如果你需要一个同表单元素中按回车相同的特性，你可以用这样代码实现：

js 代码

&lt;input type="text" onkeypress="DWRUtil.onReturn(event, submitFunction)">

&lt;input type="button" onclick="submitFunction()">

你也可以使用onkeypress事件或者onkeydown事件，他们做同样的事情。

一般来说DWR不是一个Javascript类库，所以它应该试图满足这个需求。不管怎样，这是在使用Ajax过程中一个很有用函数。

【onSubmit】

这个函数的工作原理是onSubmit()事件只存在于form元素上。

例子：getahead.ltd.uk/dwr/browser/util/onreturn

10、DWRUtil.toDescriptiveString

DWRUtil.toDescriptiveString()函数比默认的toString()更好。第一个参数是要调试的对象，第二个参数是可选的，用来指定内容深入的层次：

0: 单行调试

1: 多行调试，但不深入到子对象。

2: 多行调试，深入到第二层子对象

以此类推。一般调试到第二级是最佳的。

还有第三个参数，定义初始缩进。这个函数不应该被用于调式程序之外，因为以后可能会有变化。

11、DWRUtil.useLoadingMessage

设置一个Gmail风格的加载信息。所有演示页面

● dynamic text getahead.ltd.uk/dwr/examples/text

● selection lists getahead.ltd.uk/dwr/examples/lists

● live tables getahead.ltd.uk/dwr/examples/table

● live forms getahead.ltd.uk/dwr/examples/form

● dynamic validation getahead.ltd.uk/dwr/examples/validation

● address entry

都使用了GMail风格的加载消息。

这个方法将来可能被废弃，因为这个实现实在太专断了。为什么是红色，为什么在右上角，等等。唯一的真正答案就是：抄袭GMail。这里的建议是以本页面中的代码为模板，根据你的需求自定义。

你必须在页面加载以后调用这个方法(例如，不要在onload()事件触发之前调用)，因为它要创建一个隐藏的div来容纳消息。

最简单的做法时在onload事件中调用DWRUtil.useLoadingMessage，像这样：

js 代码

function init() {

DWRUtil.useLoadingMessage();

}

可能有些情况下你是不能容易的编辑header和body标签(如果你在使用CMS，这很正常)，在这样的情况下你可以这样做：

js 代码

function init() {

DWRUtil.useLoadingMessage();

}

if (window.addEventListener) {

window.addEventListener("load", init, false);

}

else if (window.attachEvent) {

window.attachEvent("onload", init);

}

else {

window.onload = init;

}

下面这些是这个函数的代码，它对于你要实现自己的加载消息很有用。这个函数的主要内容是动态创建一个div(id是disabledZone)来容纳消息。重要的代码是当远程调用时使它显示和隐藏：

js 代码

DWREngine.setPreHook(function() {

$('disabledZone').style.visibility = 'visible';

});

DWREngine.setPostHook(function() {

$('disabledZone').style.visibility = 'hidden';

});

This is fairly simple and makes it quite easy to implement your own "loading" message.

function useLoadingMessage(message) {

var loadingMessage;

if (message) loadingMessage = message;

else loadingMessage = "Loading";

DWREngine.setPreHook(function() {

var disabledZone = $('disabledZone');

if (!disabledZone) {

disabledZone = document.createElement('div');

disabledZone.setAttribute('id', 'disabledZone');

disabledZone.style.position = "absolute";

disabledZone.style.zIndex = "1000";

disabledZone.style.left = "0px";

disabledZone.style.top = "0px";

disabledZone.style.width = "100%";

disabledZone.style.height = "100%";

document.body.appendChild(disabledZone);

var messageZone = document.createElement('div');

messageZone.setAttribute('id', 'messageZone');

messageZone.style.position = "absolute";

messageZone.style.top = "0px";

messageZone.style.right = "0px";

messageZone.style.background = "red";

messageZone.style.color = "white";

messageZone.style.fontFamily = "Arial,Helvetica,sans-serif";

messageZone.style.padding = "4px";

disabledZone.appendChild(messageZone);

var text = document.createTextNode(loadingMessage);

messageZone.appendChild(text);

}

else {

$('messageZone').innerHTML = loadingMessage;

disabledZone.style.visibility = 'visible';

}

});

DWREngine.setPostHook(function() {

$('disabledZone').style.visibility = 'hidden';

});

}

下面的做法能简单的使用有加载消息图片：

js 代码

function useLoadingImage(imageSrc) {

var loadingImage;

if (imageSrc) loadingImage = imageSrc;

else loadingImage = "ajax-loader.gif";

DWREngine.setPreHook(function() {

var disabledImageZone = $('disabledImageZone');

if (!disabledImageZone) {

disabledImageZone = document.createElement('div');

disabledImageZone.setAttribute('id', 'disabledImageZone');

disabledImageZone.style.position = "absolute";

disabledImageZone.style.zIndex = "1000";

disabledImageZone.style.left = "0px";

disabledImageZone.style.top = "0px";

disabledImageZone.style.width = "100%";

disabledImageZone.style.height = "100%";

var imageZone = document.createElement('img');

imageZone.setAttribute('id','imageZone');

imageZone.setAttribute('src',imageSrc);

imageZone.style.position = "absolute";

imageZone.style.top = "0px";

imageZone.style.right = "0px";

disabledImageZone.appendChild(imageZone);

document.body.appendChild(disabledImageZone);

}

else {

$('imageZone').src = imageSrc;

disabledImageZone.style.visibility = 'visible';

}

});

DWREngine.setPostHook(function() {

$('disabledImageZone').style.visibility = 'hidden';

});

}

然后你就可以这样使用:useLoadingImage("images/loader.gif");

12、修补浏览器事件

如果你创建了一个DOM元素，然后用addAttribute在这个元素上创建了一个事件，那么他们不能被正常的触发。你可以使用下面的脚本来遍历一个 DOM树，并重新为他们绑定事件，这样他们就能正常的触发了。

把'click'改成你希望的事件。

js 代码

DWREngine.\_fixExplorerEvents = function(obj) {

for (var i = 0; i < obj.childNodes.length; i++) {

var childObj = obj.childNodes [i];

if (childObj.nodeValue == null) {

var onclickHandler = childObj.getAttribute('onclick');

if (onclickHandler != null) {

childObj.removeAttribute('onclick');

// If using prototype:

// Event.observe(childObj, 'click', new Function(onclickHandler));

// Otherwise (but watch out for memory leaks):

if (element.attachEvent) {

element.attachEvent("onclick", onclickHandler);

}

else {

element.addEventListener("click", onclickHandler, useCapture);

}

}

DWREngine.\_fixExplorerEvents(childObj);

}

}

}

13.

DWRUtil.selectRange("sel-test", , )

现有输入框 <input type=text id="sel-test" value="0123456789">, 则点击按钮时4567被选中: 0123456789