

3.7.5 关联函数 web_reg_save_param 详解

上面说了常见的3种关联应用的方式，可以看到所谓关联都是使用 web_reg_save_param() 函数将服务器返回的内容进行收集过滤的过程，接着我们来仔细研究一下关联函数提供的选项。

首先介绍一个函数 web_set_max_html_param_len(), 当关联出错的时候 VuGen 都会提示以下内容：

```
Action.c(20): Error -26377: No match found for the requested parameter "WCSPParam2". Check whether the requested boundaries exist in the response data. Also, if the data you want to save exceeds 1024 bytes, use web_set_max_html_param_len to increase the parameter size [MsgId: MERR-26377]
```

很多朋友看到这个错误就会头皮发麻，完全不知所措。这种错误99%都是由于关联的边界设置不合理导致没有关联到所需要的内容。系统提示使用 web_set_max_html_param_len() 函数的目的是提醒如果被关联内容超出了默认的1024字节就会导致存放数据溢出，就会产生参数值为空、关联失败的情况（做附件下载的脚本就可能会遇到这个问题），但通常都不会关联到如此巨大的内容。

web_set_max_html_param_len() 函数可以自定义关联返回值存放的参数的最大长度。

打开 Insert/Add Step 窗口，找到对应的 web_set_max_html_param_len 函数，如图 3.152 所示。

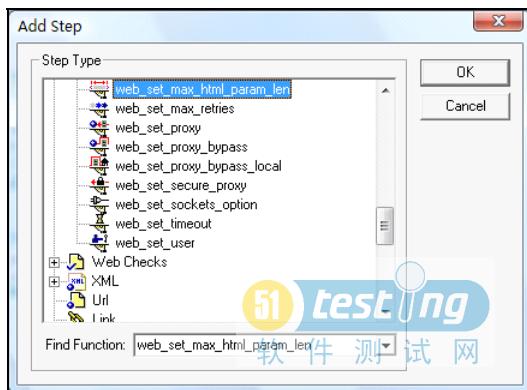


图 3.152 添加 web_set_max_html_param_len 函数

设置最大长度为 9999999，如图 3.153 所示。

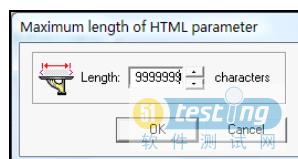


图 3.153 设置 web_set_max_html_param_len 函数长度

得到以下脚本：

```
web_set_max_html_param_len("9999999");
```

通过这个函数可以确保不会出现参数内容过长而无法存放的错误，不过这是以开销系统资源为代价的。

接着来看看 web_reg_save_param() 函数的选项，由于关联出来的内容存放在参数中，所

以还是建议打开日志中的 Parameter substitution 选项，以方便调试跟踪。

打开 Add Step 添加步骤，选择 web_reg_save_param 函数，打开关联函数设置窗口，如图 3.154 所示。

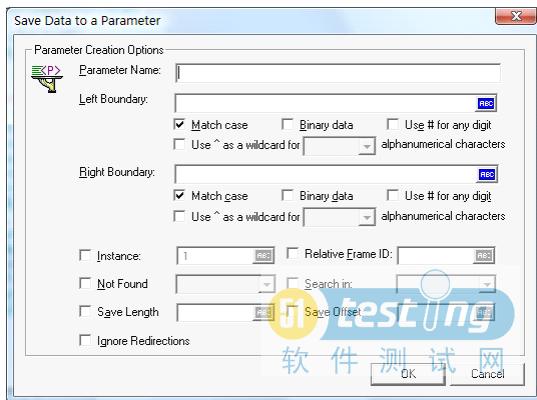


图 3.154 web_reg_save_param 函数设置窗口

Parameter Name

此处设置存放参数的名称，关联出来的内容将会存放在该参数中。这里受到 Instance 选项的影响。

例如：

设置 Parameter Name 为 temp，当对应的 Instance 选项是任意一个数字的时候，只会关联一个匹配的记录，关联值将会存放在 temp 这个参数中。当 Instance 是 All 的时候，关联成功后的值将会依次存放在“temp_数字”这样的参数数组中，并且还会添加一个 temp_count 的参数存放关联出来的记录条数。

Left Boundary

此处设置左边界，这里是用来填写关联对于数据处理的左匹配内容规则。

在左边界中存放的是一个字符串，例如填写的内容为“左边界”会被转换成以下形式：

```
web_reg_save_param("Param",
    "LB=左边界",
    "RB=",
    LAST);
```



注意

如果输入的内容里面有双引号，那么需要通过转义符来进行处理，转义符为 \。例如： web_reg_save_param("Param",

```
"LB=\\"左边界",
"RB=",
LAST);
```

Match case

默认情况下边界是 Match case 的，也就是检查大小写的，可以取消下面的选项来忽略大小写检查，会看到函数变为以下形式：

```
web_reg_save_param("Param",
    "LB/IC=左边界",
    "RB=",
    LAST);
```

Binary data

如果需要关联的内容是非 ASCII 字符的，那么需要使用该选项。选中该选项后可以看到函数变为以下形式：

```
web_reg_save_param("p", "LB/BIN=\x3F\xDD", "RB/BIN=\xCCb", LAST);
```

Use # for any digit

有些时候需要关联的边界中有些变动的数字，并且由于这个数字导致关联非常难于设置边界，可以使用该选项，例如需要关联处理的内容是：

```
<span style="white-space:nowrap;"><a class="lastViewLink" title="[Alt+1]" accessKey="1" href="index.php?module=ACLRoles&action=DetailView&record=e2ff0695-ebfd-0b34-7947-48cf557dcc0d">&ampnbspbackup</a></span>
<span style="white-space:nowrap;"><a class="lastViewLink" title="[Alt+2]" accessKey="2" href="index.php?module=Users&action=DetailView&record=59002589-79d0-7037-0ad5-48cf5cd80d2e"><img src='themes/default/images/Users.gif' width='16' height='16' border="0" align="absmiddle" alt="chen">&ampnbspchen</a></span>
<span style="white-space:nowrap;"><a class="lastViewLink" title="[Alt+3]" accessKey="3" href="index.php?module=Users&action=DetailView&record=1"><img src='themes/default/images/Users.gif' width='16' height='16' border="0" align="absmiddle" alt="Administrator">&ampnbspAdministrator</a></span>
<span style="white-space:nowrap;"><a class="lastViewLink" title="[Alt+4]" accessKey="4" href="index.php?module=Contacts&action=DetailView&record=df820bd4-e8ee-6df5-9725-48cf52ebabff"><img src='themes/default/images/Contacts.gif' width='16' height='16' border="0" align="absmiddle" alt="云层">&ampnbsp云层</a></span>
```

现在需要从这个服务器返回中获得 href 对应的地址，应该怎么写边界呢？可以跳过变动的数字直接设置 href=为左边界好了，但这样做可能会出现一个问题，如果服务器返回中间有很多 href=这样的超链接，那么关联出来的数据可能就无法满足我们的需要，所以这里可以使用#来解决这个问题。

在这里关联的左边界内容应该为：

title="[Alt+2]" accessKey="2" href="#"，然后将这个内容写入关联函数，添加转义符，函数结果应该为：

```
web_reg_save_param("Param",
    "LB/DIG=title=\\"[Alt+#]\\"
    accessKey=\#\"
    href=\"\",
    "RB=",
    LAST);
```

使用这样的左边界可以得到准确范围内的关联值，如果在项目中遇到一些很难确定关联边界的情况时，可以请求程序员编写一些便于区分的标识来方便进行关联的边界设置。

Use ^ as a wildcard for xxx alphanumerical characters

对比上面的功能，这里可以使用^符号来代替任何常用的字符。该功能有 3 个选项：All、LowerCase、UpperCase，使用^支持任意的字符，可以通过该选项来支持大写字母、小写字母或者不区分大小写。

```
web_reg_save_param("time",
    "LB/ALNUMIC=Server response ti^^:",
    "RB=seconds",
    "Ord=1",
    "Search=Body",
    LAST);
```

该选项在某些时候能够帮助跳过边界中变动的内容完成抓取操作，但是注意这里使用的^符号是不支持长度变化的，也就是说一个^代表了一个字符的位置。如果需要关联的边界本身就是变动的，就不能通过普通的方法来解决了。

Right Boundary

此处设置右边界，这里是用来填写关联时对于数据处理的右匹配内容规则，选项同左边界。

Instance

这个关键字在很多函数里面都有应用，在这里可以填写任意一个整数，也可以填 All。如果填写数字，那么说明从返回的记录中取出对应顺序的值，而填写 All 的话将会返回所有的内容。

当使用 Ord=All 时，关联函数会把所有匹配过滤策略的记录都抓出来，由于参数只能存放一条记录，所以关联函数会生成一个参数数组。被关联的记录会以{关联参数名_关联 id}的形式生成参数列表，并且在最后会有一个{关联参数名_count}的参数来存放被关联到的记录条数。

例如：上面写过的一个关联函数如下所示：

```
web_reg_save_param("Param",
    "LB/DIG=title=\\"[Alt+#]\\"           accessKey=\\"#\\""
    href="",
    "RB=\\">",
    "ORD=ALL",
    LAST);
```

当 Instance 设置为 All 时，关联将会返回所有匹配左右边界的内容，结果如下：

```
Param_1= index.php?module=ACLRoles&action=DetailView&
record=e2ff0695-ebfd-0b34-7947-48cf557dcc0d
Param_2= index.php?module=Users&action=DetailView&
record=59002589-79d0-7037-0ad5-48cf5cd80d2e
Param_3= index.php?module=Users&action=DetailView&
record=1
Param_4= index.php?module=Contacts&action=DetailView&
record=df820bd4-e8ee-6df5-9725-48cf52ebabff
Param_count=4
```

Relative Frame ID

这个选项是专门针对框架结构的网站设计的，有些时候需要关联的内容是在某个框架中的，这个时候就需要说明所关联的页面是框架中的哪一个了。比如自动关联 Web Tour 页面就必须使用这个属性：

```
web_reg_save_param("WCSPParam_Diff1",
    "LB=userSession value=",
    "RB=>",
    "Ord=1",
    "RelFrameId=1.2.1",
    "Search=Body",
    "IgnoreRedirections=Yes",
    LAST);
```

这里的 1.2.1 说明 Web Tour 网站是基于在第一个大的框架中的第二个框架，第二个框架中的第一个页面这样的两次嵌套框架，所以如果想读取左下侧的页面，这个框架所属的编号就为 1.2.1。



Not Found

如果关联的对象不存在，又该如何进行处理呢？默认值为 ERROR，默认情况下如果没有关联到任何内容则提示错误。

```
Action.c(20): Error -26377: No match found for the requested parameter "time".
Check whether the requested boundaries exist in the response data. Also, if the
data you want to save exceeds 256 bytes, use web_set_max_html_param_len to increase
the parameter size      [MsgId: MERR-26377]

Action.c(20): Notify: Saving Parameter "time = "
Action.c(20): web_url("Mspetshop") highest severity level was "ERROR", 452988
body bytes, 11060 header bytes, 13 chunking overhead bytes      [MsgId:
MMSG-26387]
```

而选择 WARNING，则只会简单提示没有抓到内容，不会产生错误。

```
Action.c(20): Warning -26377: No match found for the requested parameter
"time". Check whether the requested boundaries exist in the response data. Also,
if the data you want to save exceeds 256 bytes, use web_set_max_html_param_len
to increase the parameter size      [MsgId: MWAR-26377]
```

在很多时候关联不到内容也是正常的。例如需要关联一个板块中的帖子编号，就存在没

有帖子的情况。为了确保关联的正确性，在某些情况下可以修改 Not Found 的提示方式为 WARNING，并且为了确保没有帖子时不会对后面的操作产生影响，需要编写一个 if 语句进行判断，如下所示：

```
Action()
{
    web_reg_save_param("topicid",
        "LB=<a href=\"showtopic-",
        "RB=.aspx\" target=\"_blank"><img
src=\"templates/default/images/ t_top\",
        "Ord=1",
        "NotFound=WARNING",
        "Search=NoResource",
        LAST);

    web_url("Topic",
        "URL=http://127.0.0.1/showforum-1.aspx",
        "Resource=0",
        "Referer=",
        LAST);

    if (strcmp(lr_eval_string("{topicid}"), "")==0) {
        return 0;
    }
    else
    {
        web_url("topic",
            "URL=http://127.0.0.1/showtopic-{topicid}.aspx",
            "Resource=0",
            "Referer=",
            LAST);
    }
    return 0;
}
```

这样如果该板块没有帖子，就不会执行后面请求帖子页面的操作了。

 **注意** 如果关联选项 Ord 为 All 时，这里 if 语句检查的对象应该修改为 topicid_count 参数。

Search in

该项设置关联查询的范围，这里 VuGen 提供了 4 个选项：Header、Body、Noresource、All。我们将这 4 个选项划分两个大类。

- Noresource

Noresource 是从服务器返回的内容类别来考虑的，Noresource 就是指只从资源文件中关

联内容，也就是只从 HTML 文件格式中抓内容。

- Header/Body/All

这 3 个选项都是从请求返回的所有内容进行关联处理，包括图片、JavaScript 脚本等。区别在于对返回信息的分隔方式。在前面介绍 HTTP 的时候介绍过 HTTP 返回的内容其实是由 Header (HTTP 信息头) 和 Body (HTTP 内容) 组成的。

➤ Header

指所关联的内容是所有服务器返回请求的 HTTP 头部分内容。可以通过查看服务器返回内容来了解，Body 之前的内容都属于 Header，例如：

Header:

```
-----  
HTTP/1.1 200 OK  
Date: Thu, 25 Sep 2008 06:27:37 GMT  
Server: Apache/2.2.8 (Win32) PHP/5.2.5  
X-Powered-By: PHP/5.2.5  
Expires: Thu, 19 Nov 1981 08:52:00 GMT  
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0  
Pragma: no-cache  
Keep-Alive: timeout=5, max=98  
Connection: Keep-Alive  
Transfer-Encoding: chunked  
Content-Type: text/html; charset=UTF-8
```

➤ Body

就是服务器返回在 Body 以后的内容，例如：

Body:

```
-----  
<html><head><link rel="stylesheet" type="text/css" media="all" href="themes/  
Sugar/calendar-win2k-cold-1.css?s=5.0.0c&c=>  
<script>jscal_today = 1222324058000; if(typeof app_strings == "undefined")  
app_strings = new Array();  
</script><script type="text/javascript" src="include/javascript/sugar_grpl_<br/>  
yui.js?s=5.0.0c&c=></script>  
<script type="text/javascript" src="include/javascript/sugar_grpl.js?s=5.0.0c&c=></script><script type="text/javascript" src="jcalendar/lang/calendar-en.js?s=5.0.0c&c=></script>  
<script type="text/javascript">var asynchronous_key = "603a973f-7bd6-8e22-  
c753-48db2fb41273";</script><script type="text/javascript">  
    var time_reg_format ='  
        ([0-9]{1,2}):([0-9]{1,2}) ([ap]m)';  
    var date_reg_format ='  
        ([0-9]{1,2})/([0-9]{1,2})/([0-9]{4})';  
    var date_reg_positions = {'m': 1,'d': 2,'Y': 3};  
    var time_separator = ':';  
    var cal_date_format = '%m/%d/%Y';
```

```
var time_offset = -0;  
</script>  
<script type="text/javascript" src="cache/jsLanguage/en_us.js?s=5.0.0c&c=&j=1"></script>  
<script type="text/javascript" src="cache/jsLanguage/Users/en_us.js?s=5.0.0c&c=&j=1"></script>  
<!DOCTYPE html PUBLIC "-//W3C//DTD html 4.01 Transitional//EN">  
<link REL="SHORTCUT ICON" HREF="include/images/sugar_icon.ico">  
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  
<title>SugarCRM</title>
```

➤ All

指服务器返回的所有内容。

关于 Search in 这个选项，一般使用得比较多的是 Noresource，因为需要关联的内容一般都存放在 HTML 页面中，并且使用 Noresource 被关联到的内容又比较少（只返回一个 HTML 页面）比较适合常用处理，如果某些信息是放在 HTTP 头内的，那么只能用 Header 了。

Save Length

关联出来的内容所需要保存的长度。

例如：通过左右边界关联出来的内容是“sessionid=123456&action=work”，那么如何获得需要的 sessionid 信息呢？可以使用 Save Length 来实现，注意这里必须确保被关联内容的长度恒定。将 Save Length 设置为 16，关联出来的结果就变成“sessionid=123456”了，如果想得到后面的 sessionid 值，就要靠 Save Offset 选项了。

Save Offset

设置关联的内容偏移量，从第几位开始进行关联操作。继续上面的例子，如果需要获得 123456 这个字符串，则需要设置 Save Offset 为 10，同时设置 Save Length 为 6 即可。

通过 Save Length 和 Save Offset 的设置，我们就可以方便地抓取服务器返回内容的任意一个部分了。

关联可以调整偏移量和长度，那么参数能做到吗？当然可以，如果需要对一个参数值进行偏移和长度设置，则需要使用 lr_save_var() 函数，例如下面的代码：

```
lr_save_string("I come from shanghai","city");  
lr_save_var(lr_eval_string("{city}"),6,0,"result");  
//从 city 这个参数中取 6 位长度的内容保存到 result 参数中  
lr_save_var(lr_eval_string("{city}")+7,4,0,"result");  
//从 city 这个参数的第 7 位开始取 4 个长度的内容保存到 result 参数中
```

可以看到运行的结果是：

```
Action.c(3): Notify: Saving Parameter "city = I come from shanghai"  
Action.c(4): Notify: Saving Parameter "result = I come"  
Action.c(5): Notify: Saving Parameter "result = from"
```

问题：

前面关联的左右边界都是静态的。如果左右边界是动态的，并且系统返回的 id 是不定长度的，那么如何使用关联函数将该 id 取出呢？这个问题在现实情况中会经常遇到，仅仅

通过一个关联函数是无法处理的，这个时候还需要使用一个函数 strtok()来进行字符串内容切割（类似于正则表达式）。

strtok()函数的作用是通过某个分隔符来切分内容。

例如：

```
char city[1000];
char * token;
extern char * strtok(char * string, const char * delimiters ); //这个函数是
扩展的要声明
strcpy(city,"this is shanghai!");
token = (char *)strtok(city," ");
lr_error_message(token);
token = (char *)strtok(NULL," ");
lr_error_message(token);
token = (char *)strtok(NULL," ");
lr_error_message(token);
```

通过这个函数可以得到三个字符：this、is、shanghai。通过空格来分隔字符串，可以得到第一个符合该条件的内容，如果需要继续分隔就使用 strtok(NULL, " ");语句。如果关联出来的内容 sessionid 是变动长度的，如"sessionid=54321123&action=work"，则如何获得这个变动长度的 sessionid 呢？使用下面的代码即可解决：

```
char temp[100];
char * token;
extern char * strtok(char * string, const char * delimiters );
lr_save_string("sessionid=54321123&action=work","param");
strcpy(temp,lr_eval_string("{param}")); //取出参数值，并且赋值给变量 temp
token = (char *)strtok(temp,"&"); //使用&符号作为分隔符
```

这个时候 token="sessionid=54321123"，并且是根据&符号分隔的，所以 id 的长度可以任意变化，而 token 中的 sessionid 可以通过关联的时候 Save Offset 进行处理，或者使用 strtok() 函数对等号再次进行处理。