

## 第19学时 复杂窗体

Web上的窗体不只是简单的单页面窗体。有时窗体要跨越若干页。这些复杂的窗体以调查、查询和购物车等应用程序的形式出现。

这些比较复杂的窗体需要使用某些不同的编程技巧，本学时你将要学习这些技巧。

在本学时中，你将要学习：

- 如何创建多页窗体。

### 19.1 复杂的多页窗体

使用CGI程序来编写复杂的多页窗体时，你会遇到一个特殊的编程难题。Web浏览器与Web服务器之间的连接根本不是一个持久的连接。Web浏览器与服务器建立连接，检查Web页，然后便断开与Web服务器之间的连接。在服务器与你的Web浏览器之间并不保持不间断的连接。

更为复杂的是：浏览器每次与Web服务器连接时，Web服务器并不认为该浏览器预先访问过该站点。服务器并不每次都能很容易地识别该浏览器。

类似的一种情况是：图书馆的读者与没有记忆力的图书馆管理员之间进行谈话，读者每次只能向管理员提出一个问题。

读者向图书管理员借阅一本书，比如关于亚利桑那州的一本书，图书管理员可以检索这本书。图书管理员之所以能够检索这本书，是因为这个请求很容易满足。但是读者不能要求借阅同一个专题的另一本书。图书管理员不能记住上一个借书请求，因此他无法借给你同一个专题的另一本书。如果借书的请求改为“给我另一本关于亚利桑那州的书”，图书管理员仍然无法满足读者的要求，因为他检索的书可能与第一次检索的这本书一样。

若要检索同一专题的第二本书，唯一的办法是说：“我需要另一本关于亚利桑那州的书，我已经有了一本名叫《在亚利桑那州定居》的书”。这个借阅请求带有足够的能够说明问题的信息，使图书管理员能够知道什么应答是不适当的。

为Web页编写多页窗体，也可以使用同样的解决办法。每个问题 / 答复会话必须包含足够的信息，使Web服务器能够知道它需要做什么。你可以用几种不同的方法来创建这样的会话，其中的一种方法，即使用隐藏的HTML域，将在本学时中介绍。

### 19.2 隐藏域

要使Web窗体能够“记住”信息，最容易的方法是使用隐藏域，将以前的信息嵌入Web窗体。隐藏域是HTML窗体的组成部分，它使域和值成为HTML的组成部分，但是在显示窗体时，窗体中并不出现这些域和值。在HTML中，这些域和值编写为下面的形式：

```
<INPUT type="hidden" name="fullname" value="Pink Floyd">
```

如果将上面的HTML代码放入一个窗体，新的名字（“fullname”）和值（“Pink Floyd”）将成为窗体的组成部分。如果该窗体被提交给一个Perl CGI程序，param函数将返回一个关键字和隐藏域的值。

## 在线商店

如果要举一个如何使用隐藏域的例子，可以看一看在线商店，它使用一系列的 Web页，使人们能够根据在线目录来选购商品。目前，我们只是向你介绍复杂窗体的运行情况，在本学时后面部分的内容中，要介绍另一个复杂的窗体，它包含用于创建一个在线调查的代码。



如果不能实现某种形式的安全 Web事务处理，那么请不要使用这个在线商店的例子，请注意，这个例子并不包含任何真实的个人信息，如电话号码或信用卡号码等，因为隐藏域就像正规的 HTML窗体，它根本不具备任何安全性。

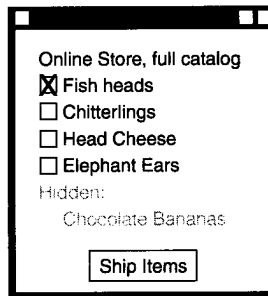
图19-1所示的在线商店第一页显示了该商店的商品清单。

图19-1 在线商店的第一页



当用户单击Go to Store（去商店）按钮时，CGI程序接收来自窗体的值，然后显示完整的目录，如图19-2所示。

图19-2 显示在线商店的商品目录



第二页显示完整的目录。当第一页（带有商店拥有商品的目录）提交时，CGI程序接收各个值，然后当它为完整的目录输出HTML时，它将商品的指定数量作为隐藏域放入新窗体。

每当CGI程序接收来自HTML窗体的值时，新页将包含隐藏域中的旧值，以及普通窗体元素中的新值。

采用这个方法，你可以避免“健忘的图书管理员”存在的问题，当提交完整目录的窗体时，窗体中的隐藏域便提醒CGI程序从第一个窗体中选定哪些项目以及从当前窗体中选择哪些项目。

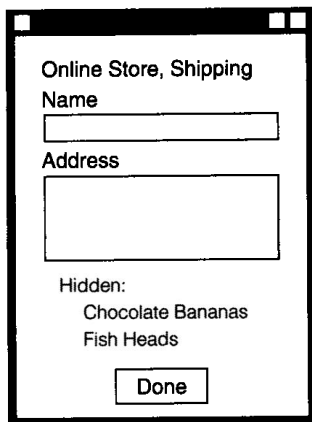
如果需要第三页，前两页中的值可以作为隐藏域存放在第三页上，如图19-3所示。

关于HTML页上的隐藏域，有几个问题应该加以说明。首先，隐藏域中的值是任何人都能够查看的。若要查看这些值，用户只需要查看该页的HTML源代码。大多数Web浏览器都配有一个选项，可以用于查看HTML源代码。

其次，隐藏域中的值可以由远程用户进行修改，如果他们确实想要这样做的话。若要修

改隐藏域的值，可以使用修改后的 Web浏览器，或者使用 HTTP人工提交该窗体。例如，在线商店不应该将价格存放在隐藏域中，它只能存放数量。CGI程序应该在需要显示价格时才查看价格。

图19-3 在线商店的发货信息



Online Store, Shipping

Name

Address

Hidden:

Chocolate Bananas  
Fish Heads



当你设计窗体时，看一看别人是如何设计窗体的，这将会对你有所帮助。这样你也会对他们是否使用隐藏域来保存信息这个问题有所了解。大多数 Web浏览器都有一个 View Page Source（查看页源）选项。你应该将这个选项用在任意窗体上，以了解它是如何形成一个整体的。但是不要拷贝这个窗体，大多数时候，拷贝会侵犯窗体的原开发人员的版权。

### 19.3 多页调查窗体

调查窗体是查找跨越若干不同的 Web页窗体的常见地方。有时这些窗体太长，一个 Web页放不下，它们通常可以分成不同的类别。

接着，简单的多页 Web调查窗体可以用于查找关于你的个人信息的各个方面。这个调查窗体可以展示4个不同的Web页，并且可以改为支持你需要的任何数目的Web页。这4个Web页是：

- 第一页用于提出一系列的一般问题，有时它们可以用来查找你拥有哪些种类的个人信息。
- 第二页用于提出一些关于你的习惯爱好的特殊问题，还有一个根据第一个调查页提出的问题。
- 第三页是供你输入你的名字和对调查的说明的 Web页。
- 在调查完成后输出的一条感谢你的消息。

相同的CGI程序可以用来执行所有这4个功能。它决定了哪一页用来打开下一页。这是根据刚才显示的这一页来决定的。程序清单 19-1显示了调查程序的核心。



通过包含代码 `use CGI::Carp qw (fatalsToBrowser)`，你的CGI程序的die()消息（它通常被写入 Web服务器的日志文件）将作为 Web页的组成部分来输出。当你编写更长的CGI程序时，它将有助于程序的调试。

调查的结果保存在一个文本文件中，但是该程序根本不显示该结果。该程序只不过进行调查结果的收集和存储。你必须编写另一个 CGI 程序，以便显示调查的结果。

#### 程序清单 19-1 调查程序的第一部分

```

1:  #!/usr/bin/perl -w
2:  use Fcntl qw(:flock);
3:  use CGI qw(:all);
4:  use CGI::Carp qw(fatalsToBrowser);
5:  use strict;
6:  my $surveyfile="/tmp/survey.txt";
7:  my @survey_answers=qw(pettytype daytype clothes
8:      castaway travel risky owmpet
9:      realname comments);
10: my $semaphore_file="/tmp/survey.sem";
11: print header;
12: if (! param ) {
13:     page_one();      # Survey just started
14: } elsif (defined param('pageone')) {
15:     page_two();      # Answered one page, print the second
16: } elsif (defined param('pagetwo')) {
17:     page_three();    # Print the last page.
18: } else {
19:     survey_done();   # Print a thank-you note, and save
20: }
```

第6~8行：在调查过程中，每个 HTML 窗体都包含输入域。每个输入域的名字都出现在这个数组中。save ( ) 函数和repeat\_hidden ( ) 函数将在以后使用这个数组。

第12~13行：如果不将任何参数传递给该 CGI 程序，也就是说，它没有作为窗体发送的结果来加载，那么就调用page\_one ( ) 函数来输出调查窗体的第一页。

第14~17行：如果名叫pageone的HTML窗体参数被传递给这个 CGI 程序，便调用page\_two ( ) 函数。如果传递的参数是 pagetwo，便调用函数page\_three。

第19行：如果 HTML 窗体参数传递给这个 CGI 程序，但是传递的参数不是 pageone 或者 pagetwo，则调查完成，其结果被保存，并在 survey\_done ( ) 函数中输出感谢你的消息。

每个 Web 页上的 submit 按钮都提供了一个关于下面应该加载哪一页的线索，你可以在图 19-4 中看到这个情况。由于 submit 按钮的名字作为一个参数被传递给 CGI 程序，因此它可以用来自显示刚刚提交给程序的是 Web 页的哪个版本。

程序清单 19-2 是调查程序的第二部分。

#### 程序清单 19-2 调查程序的第二部分

```

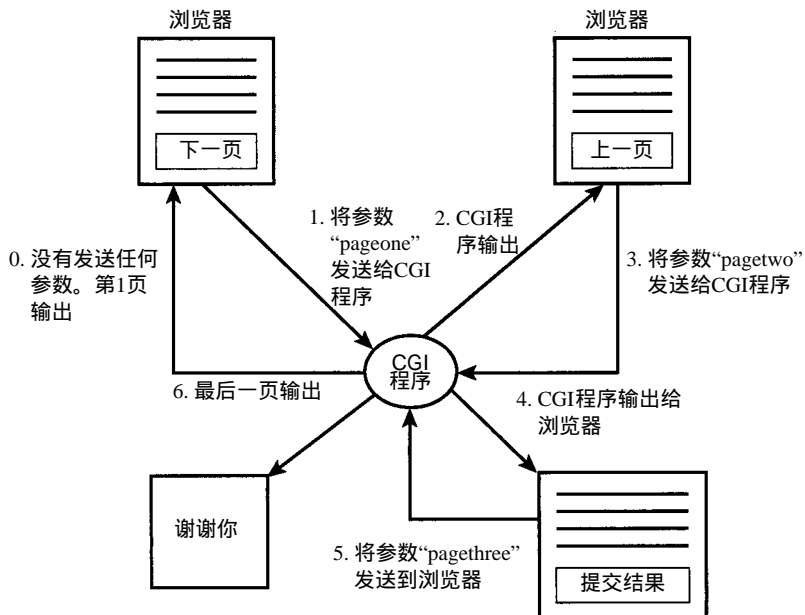
21: sub page_one {
22:     print<<END_PAGE_ONE;
23:     <FORM>
24:     Are you a "cat person" or a "dog person"?<BR>
25:     <INPUT type=radio name=pettytype value=dog>Dog<BR>
26:     <INPUT type=radio name=pettytype value=cat>Cat<BR>
27:     <P>
28:     Are you more of an early-riser or a night owl?<BR>
29:     <INPUT type=radio name=daytype value=early>Early riser<BR>
30:     <INPUT type=radio name=daytype value=late>Night Owl<BR>
31:     <P>
32:     At work, if you had a choice on how to dress....<BR>
```

```

33: <INPUT type=radio name=clothes value=casual>Casual<BR>
34: <INPUT type=radio name=clothes value=business>Business<BR>
35: <P>
36: If stranded on a desert island,
37: who would you rather be stuck with?<BR>
38: <INPUT type=radio name=castaway value=ginger>Ginger<BR>
39: <INPUT type=radio name=castaway value=marya>Mary-Anne<BR>
40: <INPUT type=radio name=castaway value=prof>Professor<BR>
41: <INPUT type=radio name=castaway value=skipper>Skipper<BR>
42: <INPUT type=submit name=pageone value="Next page">
43: </FORM>
44: END_PAGE_ONE
45: }

```

图19-4 哪个按钮用于执行  
哪个操作的示意图



第22~24行是个 Perl 新结构，你以前没有看到过，它称为“here document”。here document 使你设定一个跨越若干行的字符串，它包含其他的引号，可以像一个普通的双引号那样来运行。若要开始编写一个 here document，你可以使用 <<，后随一个单词。这个引号的内容就继续下去，直到在一行的开头再次出现该单词为止，如下例所示：

```

$a=<<END_OF_QUOTE;
This is included as part of the string.
END_OF_QUOTE

```

用于标识 here document 开始的单词，即上面这个代码段中的 END\_OF\_QUOTE，或者程序清单 19-2 中的 END\_PAGE\_ONE，后面必须跟一个分号。在 here document 的结尾，该单词必须出现在第一列的开头，并且后面不能有任何字符，如空格或分号。在 here document 内，变量像它们在普通双引号字符串 (“ ”) 中那样展开，因此在 here document 中，必须慎重使用 \$ 和 @ 字符。

使用 here document 时，可以将大量的 HTML 代码嵌入你的 Perl 程序，而不会夹杂许多引号和多个 print 语句，从而造成混乱。

程序清单 19-2 中的函数只是用来输出一个 HTML 窗体。<FORM> 标记并不包含动作和方法。

当没有设定<FORM>的action属性时,当前的CGI程序(即产生窗体的CGI程序)将在提交窗体时重新加载。当不提交method属性时,便使用默认方法GET。

请注意,窗体上的提交按钮的名字是pageone。当该窗体被提交时,一个称为pageone的参数将被发送到该CGI程序,它的值并不重要。被提交的这个参数将提示CGI程序加载第二个Web页。

程序清单19-3是CGI程序的第三部分。

程序清单19-3 调查程序的第三部分

```
46: # Print out any of the responses so far as hidden fields
47: sub repeat_hidden {
48:     foreach my $answer ( @survey_answers ) {
49:         if (defined param($answer)) {
50:             print "<INPUT TYPE=hidden";
51:             print " name=$answer ";
52:             print " value=\"", param($answer), "\">\n";
53:         }
54:     }
55: }
56: sub page_two {
57:     my $pet=param('pettype');
58:     if (! defined $pet) {
59:         $pet="goldfish";
60:     }
61:     print<<END_PAGE_TWO;
62: <FORM>
63: Would you rather...<BR>
64: <INPUT type=radio name=travel value=travel>Travel<BR>
65: <INPUT type=radio name=travel value=home>Stay at home<BR>
66: <P>
67: Do you consider yourself...<BR>
68: <INPUT type=radio name=risky value=yes>A daredevil<BR>
69: <INPUT type=radio name=risky value=no>Cautious<BR>
70: <P>
71: Do you own a $pet?<BR>
72: <INPUT type=radio name=ownpet value=$pet>Yes<BR>
73: <INPUT type=radio name=ownpet value=no>No<BR>
74: <P>
75: <INPUT TYPE=submit name=pagetwo value="Last Page">
76: END_PAGE_TWO
77:     repeat_hidden();
78:     print "</FORM>";
79: }
```

第47行:正如第46行中的注释所表示的那样,这一行中的函数用于输出作为隐藏域的该窗体的所有域的值。数据@survey\_answers包含HTML窗体上所有可能的“name=”值。当第一次运行时,大多数域将不存在,因为调查的这些部分尚未填入相应的值。

第48~49行:@survey\_answers中可能的每个参数均被检查,每个参数均被定义。输出HTML标号<INPUT TYPE=hidden>,用于存放当前窗体上的值。

第56行:这个函数用于输出调查的第二页。

第57~60行:这个函数在调查的第二页中调用。如果调查的第一页填入了正确的值,那么param('pettype')将保存dog或cat,这个值将存放在\$pet中。如果被调查人跳过了这个问题,同时param('pettype')没有定义,那么就改用goldfish。

第61~76行：来自第一页的HTML窗体参数均转入该窗体，作为其隐藏域。

如果你在这时查看调查窗体，即它的第二页，那么第一页的所有答案均作为隐藏域存放在第二页的结尾处。程序清单 19-4显示了第三页的代码。

程序清单 19-4 调查程序的第四部分

```
80: sub page_three {
81:     print<<END_PAGE_THREE;
82: <FORM>
83: Last page! This information is optional!<BR>
84: Your name:
85: <INPUT TYPE=text name="realname"><BR>
86: Any comments about this survey:<BR>
87: <TEXTAREA NAME=comments cols=40 rows=10>
88: </TEXTAREA>
89: <P>
90: <INPUT TYPE=submit name=pagethree
91:     value="Submit survey results">
92: END_PAGE_THREE
93:     repeat_hidden();
94:     print "</FORM>";
95: }
```

函数page\_three ( ) 是非常明了的。它只是输出窗体中的一个文本框和一个文本区域。在结尾处，它再次调用repeat\_hidden ( ) 函数，以便将所有隐藏域放入调查窗体的第三页。程序清单 19-5显示了CGI调查程序的结尾部分。

程序清单 19-5 调查程序的最后部分

```
96: sub survey_done {
97:     save();
98:     print "Thank You!";
99: }
100: #
101: # Save all of the survey results to $surveyfile
102: #
103: sub save {
104:     get_lock();
105:     open(SF, ">>$surveyfile") || die "Cannot open $surveyfile: $!";
106:     foreach my $answer (@survey_answers) {
107:         if (defined param ($answer) ) {
108:             print SF $answer, "=", param($answer), "\n";
109:         }
110:     }
111:     close(SF);
112:     release_lock();
113: }
114: #
115: # Locks and Unlocks the survey file so that multiple survey-takers
116: # Don't clash and write at the same time.
117: #
118:
119: # Function to lock (waits indefinitely)
120: sub get_lock {
121:     open(SEM, ">$semaphore_file")
122:     || die "Cannot create semaphore: $!";
123:     flock SEM, LOCK_EX;
```



```
124: }
125:
126: # Function to unlock
127: sub release_lock {
128:     close(SEM);
129: }
```

第96行：调用这个函数只是为了输出感谢你的消息。当某人遍历调查窗体的 3个页面后，这样做总是一件很好的事情。然后调用 save ( ) 函数。

第103行：这里的save ( ) 函数几乎是第18学时中的save函数的复制品。它用 get\_lock ( ) 将调用文件锁定，再使用类似 repeat\_hidden ( ) 中的方法写入对问题的答案，然后用 release\_lock ( ) 函数对文件解锁。

你可以随意修改这个调查程序，以适应你自己的需要。它的设计非常灵活，并且可以用于许多不同的目的。

## 19.4 课时小结

在本学时中，你学习了如何创建多页 Web窗体的方法。当你进行这项操作时，了解到程序需要解决的几个问题，最重要的是要记住从一页转到另一页时会出现的一些情况。你还学会了如何使用隐藏域将信息存放在服务器无法记住的 Web页上，然后就可以使用隐藏域来创建框架调查窗体了。

## 19.5 课外作业

### 19.5.1 专家答疑

问题：HTML窗体难道一定是如此不顺眼吗？

解答：本书中介绍的这些窗体是简单的、缺乏特色的框架式的窗体，有的人称它们是不顺眼的窗体。本书的目的是教你进行 Perl和CGI编程，而不是教你如何使用 HTML。实际上，本书中讲到的大多数 HTML程序与标准无关，并且它是不完整的，它没有使用 <HEAD>标记，和<HTML>标记，也没有DTD标题。通过提供基本的 HTML，我想你能够对它进行修改，使之符合你的需要。

前面讲过，给窗体增色的好办法是查找 Web，寻找你喜欢的窗体。通过查看源代码，你就会对如何将这 Web页组合在一起有个大致的了解。

问题：我看到这样一个出错消息：Can't find string terminator " xxxx " anywhere before EOF at ...。这是什么意思？

解答：这个错误是因为在程序的某个位置上有一个左引号，但是没有匹配的右引号而造成的。当你使用 " here document " 时，这意味着无法找到你给 " here document " 做上结尾标记的单词。它的格式如下：

```
print <<MARK;
text
text
text
MARK
```

在上面这个例子中，“ here document ” 开头和结尾的单词 MARK必须完全相同。结尾的单



词这一行上，它的前面不能有任何东西，后面也不能有任何东西。MS-DOS和Windows的文本编辑器有时并不在程序的最后一行的后面放上行尾符。如果你的“ here document ”以文件的结尾为结束，请在它的后面放上一个空行。

### 19.5.2 思考题

- 1) 为了使你的程序能够记住很长的多页 Web 事务处理，你需要使用
  - a. 数据库和 cookie。
  - b. 隐藏的 HTML 窗体域。
  - c. 隐藏的 HTML 窗体域、cookie 和数据库的某种组合。
- 2) 使用 HTML 的 <FORM> 标记时，如果不带 action 属性，那么它将
  - a. 无法运行。
  - b. 导致 submit 按钮使用原先生成 Web 页的 CGI 程序。
  - c. 导致 submit 按钮重新加载当前页。
- 3) 上面介绍的调查程序有一个小错误，是什么错误？
  - a. print<<EOP；它在程序清单 19-2 中是个无效语句。
  - b. HTML 不完整，因为它不带 <HEAD> 标记和类似的标记。
  - c. 调查程序没有输出结果。

### 19.5.3 解答

1) 答案可以是 b 或 c。你可以只使用隐藏的 HTML 域，也可以只使用 cookie。如果只使用数据库，那是不行的。

2) 答案是 b。重新加载当前页将会删除当前窗体的所有答案。如果没有 action 属性，<FORM> 标记将把当前页的 URL 用作替换 URL。

3) 答案是 b。print<<EOP；肯定是个有效的语句，它称为“ here document”。选择 c 是不正确的，因为程序不是使用该方法设计的（参见“实习”这一节）。

### 19.5.4 实习

- 编写一段 CGI 程序，用于显示调查的结果。也可以创建一个表格，以下面的形式显示这些结果：

猫/狗	拥有一只	夜间活动	服装	被谁丢弃	旅行者	有危险吗
猫	否	是	普通	教授	是	是
两者之一	金鱼	否	专用	船长	否	否
狗	是	否	普通	玛丽-安尼	是	是

- 补充问题，编写一个 CGI 程序，将调查结果汇总成下面的形式：

猫/狗的比例	猫 40%	狗 45%	其他	15%
拥有该宠物的人	猫 20%	狗 15%	金鱼 30%	无 35%

的比例：

夜行者：	是 35%	否 40%
------	-------	-------