第21学时　cookie

在第19学时中，我们讲述了如何使用HTML中的隐藏域使你的Web浏览器记住各个Web之间的信息。你必须理解这个进程，因为从CGI程序的一个实例到另一个实例，有时需要在它们之间传递信息。进行这项操作的唯一方法是将一些信息存储在浏览器中。

将信息存储在浏览器中的另一个方法是使用HTTP的cookie。正如它的名字所表示的那样，HTTP Cookie是指在HTTP连接期间浏览器与CGI程序之间传递的信息。使用Cookie，可以比使用HTML隐藏域更加灵活地用浏览器来存储信息。

在本学时中，你将要学习

- 什么是cookie。
- 如何编写和检索cookie。
- 如何处理和避免cookie的常见问题。

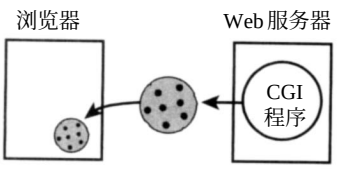
21.1　什么是cookie

可以将cookie视为电影院的入场券。你可以到电影院购买一张入场券，以便在以后的某个时间拿着入场券到电影院去看电影。看完电影你就可以离开电影院，往回家路上走，买一点爆玉米，并做你喜欢做的任何事情。当你准备看电影时，你向电影院的收票员出示电影票。收票员并不知道你如何、何时和为何购买电影票，但是，只要你持有电影票，收票员就允许你进入电影院。电影票使持票人有权在以后进入电影院去看电影。

HTTP cookie只不过是CGI程序要求浏览器持有的一个信息包。这个信息包可以由另一个CGI程序或原来的程序在任何时候回收。当有人要检索正常的HTML Web页时，cookie甚至可以重新传回给服务器。cookie可以包含任何种类的信息，比如关于多页Web窗体的信息、访问信息、用户喜欢的信息等。

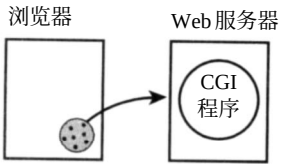
每当CGI程序要求创建cookie时，cookie可以从服务器传送到浏览器（见图21-1），这个进程称为安装cookie。

图21-1　cookie从CGI程序传送到浏览器



CGI程序可以在晚些时候回收，以便检索存储在cookie中的信息，如图21-2所示。

图21-2　浏览器将cookie送回到服务器



cookie因何而得名

在计算机界，cookie是个非常老的术语。它是指例程或程序之间传递的任何一组信息，它使cookie的持有者能够执行某项操作。某些类型的 cookie称为神秘的cookie，因为它们包含的数据非常神秘，只有cookie的发送者和接收者才能理解其含义。 CGI cookie并不神秘。

21.1.1 如何创建cookie

若要创建cookie，你可以使用CGI函数cookie。cookie函数的句法如下：

```
$cookie_object=cookie( -name => cookie_name,          # Optional
                       -value => cookie_value,
                       -expires => expiration_date,   # Optional
                       -path  => path_info,           # Optional
                       -domain => domain_info,        # Optional
                       -secure => true/false          # Optional
);
```

cookie函数以一种特殊的方式使用参数。调用 cookie时使用的每个参数都带有名字。实际上，在Perl中以这种方法将参数传递给函数是非常方便的，因为你不必记住参数的顺序，它们将按你使用它们时的顺序进行命名。

当你用这个句法来调用 cookie函数时，该函数便返回一个 cookie（该cookie应该存放在一个标量变量中），然后该cookie可以被赋予CGI模块的header函数，以便发送给浏览器。创建cookie时必须要的唯一参数是-value。-name参数允许同时将若干个cookie发送给浏览器，而检索时则可以单个检索，也可以成组检索。其他参数如 -expires、-path、-domain和-secure等，将在下一节介绍。

CGI模块中的header函数负责管理将cookie发送给浏览器的实际操作。这意味着必须使用cookie函数来创建cookie，然后紧接着就调用 header函数。在cookie和标题发送之前，不应该将任何其他种类的数据发送给浏览器。

若要使用CGI程序创建一个cookie并将它发送给浏览器，你可以使用类似下面这样的CGI程序：

```
#!/usr/bin/perl -w
use CGI qw(:all);
use strict;

my $cookie=cookie(-name => 'Sample',
    -value => 'This cookie contains no MSG');

# Transmit the cookie to the browser
print header(-cookie => $cookie);
```

当上面这个代码段运行之后，浏览器上就安装了一个称为 sample的cookie。该cookie包含了“ This cookie contains no MSG（该cookie不包含任何消息）”这样一个信息。

实际上该cookie并没有安装。浏览器可以因为许多原因而拒绝接受某个cookie。请参见本学时后面部分中的“ cookie存在的问题 ”这一节。

若要在你的CGI程序中从浏览器中检索 cookie，可以使用相同的cookie函数。如下面的例子所示，如果不带任何参数， cookie函数返回浏览器拥有的服务器的一个 cookie列表：

```
@cookie_list=cookie();   # Returns names of all cookies set

--or--

# Returns the value for a particular cookie
$cookie_value=cookie($cookie_name);
```

按照默认设置，当cookie安装在浏览器上之后，它将返回给驻留在同一个服务器上的任何CGI程序。也就是说，只有安装 cookie的服务器才能检索这些 cookie。若要查看以前创建的Sample cookie，可以使用另一个CGI程序：

```
#!/usr/bin/perl -wT
use CGI qw(:all);
use strict;

print header();  # Print out the standard header

print "Sample's  value: ", cookie('Sample'), "<P>";
```

上面的代码段使用带有一个参数的 cookie函数，这个参数就是你想查看其值的 cookie的名字。该值被检索并输出。

cookie应该被浏览器保留到浏览器运行终止。当浏览器重新启动时， cookie sample将不复存在。如果你想创建一个比较永久的 cookie，请参见本学时后面部分中的“ 设置 cookie终止运行的时间 ”这一节。

大多数浏览器都配有一个选项，用于在 cookie被安装时查看这些 cookie。在Netscape中，你可以在 Advanced选项卡上的 Preferences选项下找到查看cookie的各个选项。在 Internet Explorer中，这个选项出现在 Internet Options对话框的 Advanced选项卡上，还有一个单选按钮可用于控制你是否可以在安装cookie时查看它们。

21.1.2 举例：使用cookie

使用这个例子，你可以创建一个小程序，让用户可以使用 Web浏览器来设置他查看的 Web页的颜色。该程序实际上能够同时执行若干项操作：

1) 通过查看程序的各个参数，以便观察默认背景色的变化。
2) 用正确的背景色在浏览器上设置 cookie。
3) 将Web页的背景色设置为正确的颜色。
4) 显示一个CGI窗体，使你能够改变其颜色。

程序清单21-1包含改变颜色的程序。

程序清单21-1 ColorChanger程序的完整清单

```
1:   #!/usr/bin/perl -w
2:   use strict;
3:   use CGI qw(:all);
4:   use CGI::Carp qw(fatalsToBrowser);
5:   my($requested_color, $old_color, $color_cookie)=("","");
6:   $old_color="blue";  # Default value
7:   # Is there a new color requested?
```

```
8:   if (defined param('color')) {
9:           $requested_color=param('color');
10:  }
11:  # What was the old color, if any?
12:  if (defined cookie('bgcolor')) {
13:          $old_color=cookie('bgcolor');
14:  }
15:  if ($requested_color and ($old_color ne $requested_color)) {
16:          # Set the cookie in the browser
17:          $color_cookie=cookie(-name => 'bgcolor',
18:                               -value => $requested_color);
19:          print header(-cookie => $color_cookie);
20:  } else {
21:          # Nothing's changed, no need to set the cookie
22:          $requested_color=$old_color;
23:          print header;
24:  }
25:  print<<END_OF_HTML;
26:  <HTML>
27:  <HEAD>
28:  <TITLE>Set your background color</TITLE>
29:  </HEAD>
30:  <BODY BGCOLOR="$requested_color">
31:  <FORM>
32:  <SELECT NAME="color">
33:          <OPTION value='red'>Red
34:          <OPTION value='blue'>Blue
35:          <OPTION value='yellow'>Yellow
36:          <OPTION value='white'>White
37:  </SELECT>
38:  <INPUT TYPE=SUBMIT VALUE="Set the color">
39:  </FORM>
40:  </BODY>
41:  </HTML>
42:  END_OF_HTML
```

第7~10行：如果该程序作为CGI窗体的目标程序来调用，那么param（‘color’）函数值返回一个定义的值，即一个新颜色。否则，它不返回任何值，$requested_color则保持未设定状态。

第12~14行：这些行用于检索名叫bgcolor的cookie。它可能存在，也可能不存在。如果它不存在，那么它存放在$old_color中，这是上次保存到cookie中的屏幕颜色值。

第15~19行：如果颜色已经改变（即cookie的值与新值不一致），那么新cookie必须用新值进行设置。

第20~24行：否则，输出一个纯标题，不带 cookie。请记住，浏览器将无限期保留以前的cookie。

第25~42行：这些代码行用于创建一个标准HTML窗体。不过请注意第30行，在这一行上，被取代的颜色被送入HTML输出。

21.1.3 另一个例子：cookie查看器

程序清单21-2中列出的一个非常短的程序是个 cookie查看器，它用于帮助你调试使用cookie的CGI程序。它列出了存储在Web浏览器上的所有cookie，这些cookie恰好来自同一个Web服务器。

程序清单21-2 Cookie查看器

```
1:   #!/usr/bin/perl -w
2:
3:   use strict;
4:   use CGI qw(:all);
5:
6:   print header();
7:
8:   print "Cookies set that can be seen:<P>";
9:
10:  foreach my $cookie (cookie()) {
11:          print "Cookie name: $cookie <BR>";
12:          print qq{Cookie value: "}, cookie($cookie), qq{"<P><HR>};
13:  }
```

第10行：用cookie函数检查所有cookie的名字，并赋予$cookie，每次检索1个cookie。

第11~12行：输出每个cookie的名字和值。

该cookie查看器运行时，可以获取使用 cookie（）函数能够得到的所有 cookie的列表，然后对这些名字迭代运行cookie，输出每个cookie的名字和值。

21.2 高级cookie特性

cookie的基本概念简单明了，你将 cookie赋予浏览器，过一会儿浏览器又将它送回给服务器。不过cookie的基本特性并不止此。可以将 cookie设置为可以存在较长的时间，这种 cookie称为永久性cookie。你可以让这些cookie只返回到另一个特定的URL，它们能够指明关于你的连接的安全程度之类的信息。

21.2.1 设置cookie终止运行的时间

到现在为止，你在浏览器上安装的 cookie都是临时的。一旦浏览器关闭， cookie就消失。当你使用cookie将值保存在窗体上的多个页中（而不是保存隐藏的 HTML值）时，使用临时cookie是完全合适的。当一个新浏览器启动时，你不想让 cookie返回给服务器，因为用户不会从中间开始填写窗体，他将再次从头开始时填写。

在有些情况下，你可能希望cookie能够保留更长的时间。也许你想在浏览器关闭和重新启动之后使cookie持续数天、数周、数月时间。用Perl的CGI模块来创建这种cookie是非常容易的。

若要为cookie设置一个终止日期，可以在创建 cookie时使用-expires选项。-expires选项必须后随一个想使cookie终止运行的日期。可以如表21-1所示用若干种格式设置这个日期。

表21-1 cookie的终止日期格式

格 式	示 例	含 义
秒数	+30s	从现在起30秒后终止
分钟数	+15m	从现在起15分钟后终止
小时数	+12h	从现在起12小时后终止
月数	+6M	从现在起6个月后终止
年数	+1Y	从现在起1年后终止
	now	cookie立即终止运行
任何负时间值	-10m	cookie立即终止运行
一个特定时间	Saturday,28-Aug-1999 22:51:05 GMT	

当设定一个特定时间时，必须完全使用表 21-1中列出的时间格式。所有其他的各种设置值都是指从当前时间起的时间偏移量。系统将为你计算出完全合格的时间值，然后发送给浏览器。

下面这个小程序用于在浏览器上安装一个将在 8天后终止运行的cookie：

```
#!/usr/bin/perl -w
use CGI qw(:all);
use strict;

my $cookie=cookie(-name => 'Favorite',
    -value => 'soft oatmeal raisin cookies',
     -expires => '+8d' );

# Transmit the cookie to the browser
print header(-cookie => $cookie);
```

21.2.2 cookie的局限性

要使cookie能够永久运行是做不到的。这就是说，如果将一个 cookie发送给浏览器，希望从现在起该cookie能够在数周、数月或者数年内保持运行，那么你一定会大失所望的。

当你读到后面的“ cookie存在的问题 ”这一节内容时，就会知道浏览器并不是必须将cookie存储起来的。实际上，它们根本不必接受你的 cookie，它们并不通知你这些 cookie并没有保留起来。

浏览器可以随时清除它们的 cookie，以便为来自其他站点的新 cookie腾出地方，或者根本毫无理由就这样做了。有些浏览器允许用户编辑 cookie，或者添加新的cookie。

用户可能不小心删除 cookie，也可能故意将cookie删除掉。如果用户安装了浏览器或操作系统的新版本， cookie就会被清除，或者放到别的什么地方。只要改用另一种浏览器， cookie就会“ 不知去向 ”。当浏览器尚未激活时， cookie通常存放在一个文件中，该文件可以供用户编辑，删除，或者遭到损坏。

如果你有兴趣的话，我们可以告诉你，大多数浏览器是在没有激活时将cookie存放在文件中的，这些文件通常是文本文件，你可以使用编辑器查看这些文件。Netscape将cookie存放在用户主目录下的 cookies.txt文件中（不同的系统下该目录将各不相同）。Internet Explorer将cookie存放在\Windows\Cookies下。

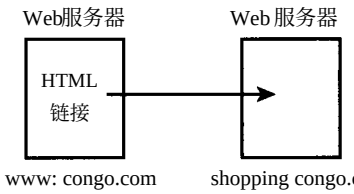
因此，将重要信息存放在一个 HTTP cookie中真的不是个好主意。你想永久存放在 cookie中的任何信息不应该被轻易改变位置，这些信息包括用户喜欢的信息，输入指定 Web页的可替换项目关键字，上次刚刚访问的信息等。

21.2.3 将cookie发送到其他地方

按照默认设置， cookie只能送回到曾经发出 cookie的服务器。有时，你希望将 cookie送回到服务器，但有时你并不希望如此。以神秘的 Web站点Congo.com为例，这个销售书籍的 Web站点拥有两个Web服务器，即www.congo.com和shopping.congo.com，如图21-3所示。主要的Web站点（www.congo.com）包含公司的所有信息，可以连接到其他站点，并且最重要的是可

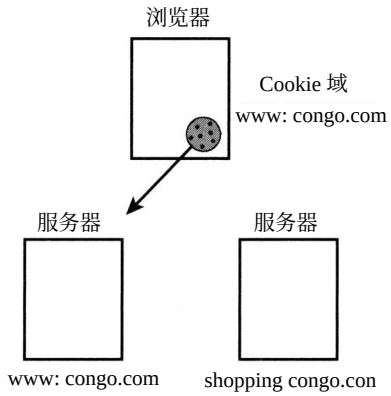
以连接到在线书店。

图21-3 两个互相连接的Web站点



www.congo.com包含一个注册用的HTML窗体／CGI程序，使用户可以将他们的名字添加到电子邮件的地址列表，设置他们喜欢什么类型的书籍。以后，当用户浏览 www.congo.com时，他就可以阅读关于他感兴趣的新书的信息。用户浏览器上的 cookie负责告诉 www.congo.com，应该向他介绍哪些书籍的情况（见图 21-4）。

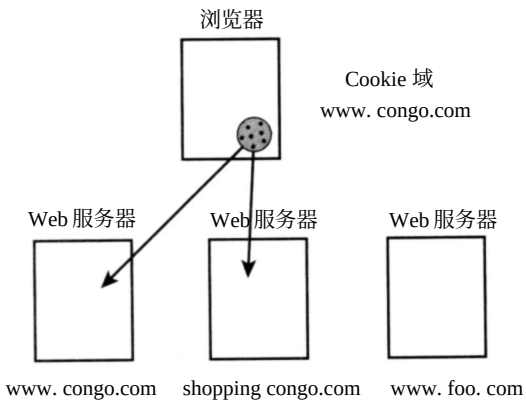
图21-4 只返回给单个 Web站点的cookie



问题是当用户从 www.congo.com转到位于 shopping.congo.com站点上的在线书店时，cookie没有被发送到shopping.congo.com服务器。HTTP cookie只返回给原先发送cookie的这个服务器。如果www.congo.com发送了该cookie，它并不发回给shopping.congo. com。

那么你应该怎么办呢？如果让用户填写另一个首选项窗体，并且从 shopping.congo.com给他发送一个新cookie，那将是不切实际的。更好的办法是限制这个 cookie只能使用一个特定的域名。例如，当原始 cookie从www.congo.com发送出来时，可以将该 cookie送回给任何 congo.com Web站点，如图21-5所示。

图21-5 返回给两个 Web站点的cookie



若要进行上述操作，可以在创建 cookie时使用带有 -domain参数的cookie函数：

```
$cookie=cookie( -name => 'preferences',
        -value => 'mysteries, horror',
        -domain => 'congo.com');
print header(-cookie => $cookie);
```

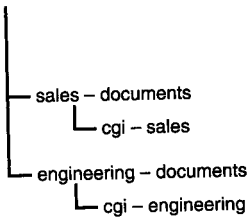
在上面这个代码段中， cookie $cookie得以创建，并且限制为congo.com域。任何Web服务器，如果其主机名以congo.com为结尾，将使它的cookie由浏览器返回给该服务器。

域的参数至少必须由两个部分组成，并且不能是绝对的顶层域，即 .com或.net。这样，就可以避免浏览器将 cookie从一个.com域移植到另一个 .com域。

21.2.4 限制cookie返回到的位置

将cookie限制为只能返回到某个服务器，这也是可能的。当你创建一个 cookie时，按照默认设置，该cookie可以返回到Web站点上的任何URL，包括非CGI URL。例如，如图21-6所示的汽车销售Web站点的组织结构。

图21-6 一个多用户 Web站点的目录树结构



让sales（销售）CGI程序和engineering（工程设计）CGI程序驻留在不同的目录中，是有意义的。如果sales CGI程序准备建立一个cookie，那么engineering CGI程序便将它接收过来，反过来也一样。这样的结果是人们所不希望的，为两个站点同时编写 CGI程序的开发人员必须采取协调措施，以确保不会重复使用对方的 cookie名字。

为了解决这个问题，可以使用 cookie函数的-path选项。该选项用于指明cookie应该返回到的路径名（相对于URL顶层的路径名）。例如，若要发送只返回到sales CGI程序的一个cookie，可以使用下面的代码段：

```
# Cookie only visible to sales CGI programs
$cookie=cookie( -name => 'profile',
        -value => 'sedan,luxury,2-door',
        -path => '/cgi-sales');
print header(-cookie => $cookie);
```

按照默认设置， cookie返回到服务器上的每个站点，就像已经使用了选项 -path=>‘／’一样。若要限制只能返回到一个CGI程序，可以在-path选项中使用CGI程序的URL：

```
# Return the cookie only to this program
$cookie=cookie( -name => 'profile',
        -value => 'sedan,luxury,2-door',
        -path => script_name() );
print header(-cookie => $cookie);
```

上个学时中我们讲过， CGI模块中的script_name函数能够返回当前CGI程序的部分URL。这可以有效地创建这样一个cookie，它只返回到在浏览器上安装该cookie的程序。

21.2.5 带有安全性的cookie

有些cookie你可能只想在一条安全的连接上传输它们。使用 cookie函数的-secure参数，就可以只在连接是安全的时候从浏览器发送 cookie。下面的代码用于将一个包含账号的 cookie发送给浏览器。包含此类敏感信息的 cookie只能在安全的连接上发送。

```
# Caution!  Send this only over an https connection
$cookie=cookie( -name => 'account',
        -value => '00-12-3-122-1313',
        -secure => 1);
print header(-cookie => $cookie);
```

以后，如果你要检索该cookie，只要像平常那样使用cookie函数即可。如果连接是安全的，而且该cookie是在该浏览器上，那么该浏览器就可以在需要时将 cookie发回给服务器。

```
# Fetch the account number from the browser.
$account_number=cookie('account');
```

你不应该依赖这个方法来检查连接是否安全，也不应该依赖账号的准确性。请记住，用户负责控制Web浏览器及其cookie文件。cookie可以在不安全的连接上发回给服务器，甚至可以有一个无效号码。

21.3 cookie存在的问题

在你将cookie投入应用之前，应该知道与cookie相关的一些问题。由于这些原因和将来可能产生的其他原因，你应该认真设计你的Web页和CGI程序，使得cookie完全成为可以选择的选项。

例如，如果你使用 cookie来存放用户的首选项，那么倘若 cookie无法使用，你应该使用一组默认首选项。编码时请采取相应的防范措施。

21.3.1 cookie的生存期很短

本学时中多次讲到， cookie的寿命很短。Cookie可以从用户的系统中删除，可以由用户编辑，也可以毫无理由地被浏览器甩掉。

浏览器可以接受cookie，将它使用一会儿，然后毫无理由就将它忘掉。如果你使用 -expire选项安装了一个永久性cookie，浏览器仍然可以甩掉这个cookie，并且根本不通知用户。

21.3.2 并非所有浏览器都支持cookie

并非所有浏览器都支持 HTTP cookie，这是千真万确的事实。适用于 HTTP和Web信息传输的Internet标准并不能保证浏览器必须支持 cookie。

并不是说大多数浏览器都不支持 cookie，大多数浏览器是支持 cookie的。Netscape（自从1.1版以来），Internet Explorer（所有版本），Lynx,Opera，以及大多数流行的 Web浏览器都支持cookie。在大多数浏览器中，有一个选项可供用户关闭对 cookie的支持。

即使你使用CGI模块的user_agent函数，确定你想使用的浏览器应能支持 cookie，也不要完全指望它。

21.3.3 有些人不喜欢cookie

这一节的标题也许很难理解，为什么世界上竟然有人不喜欢 cookie呢？

在Web上冲浪实际上是一种匿名活动。正如你在上一学时中看到的那样，当浏览器要求检索一个Web页时，这个检索请求是在真空中发生的。服务器不一定知道浏览器所在的位置，也不知道该浏览器上次曾经要求检索过该站点上的一个 Web页。

请记住，一个浏览器不一定代表一个用户，一个浏览器可以被一个家庭、网吧、Internet网吧或公共访问点（如图书馆）中的许多人共享。为一个人安装（或修改）一个cookie，实际上也为若干人安装了cookie。

cookie可以用来跟踪人们曾经访问过某个站点的哪个位置以及他们曾经点击过什么。如果你非常在乎隐私问题，那么这个情况你应该注意。

例如，前面的“将 cookie发送到其他地方”这一节中我们提到的一个虚构在线书店congo.com能够跟踪Web冲浪者点击了哪些书籍以便了解其详细信息，并使用该信息编写符合读者需要的书目，提供给Web冲浪者。

从表面上看，这些特性很好。但是对于那些想要维护隐私权的人来说，这会带来两个问题。首先，现在有一个机构负责跟踪 Web冲浪者感兴趣的是什么种类的书籍。如果这些信息与Web冲浪者的名字和地址有关（也许这些信息是从 congo.com共享信息的另一个站点的填写式窗体中获得的），那么Web冲浪者将会收到与他选购书籍相关的垃圾邮件。与 cookie搜集站点共享的信息越多，就能获得关于Web冲浪者更详细的信息。

除了隐私问题外，如果Web冲浪者查看的头两本书属于“计算机”书籍， Web站点就会停止向Web冲浪者提供“传奇”类和“烹饪”类书籍。 Web站点将把Web冲浪者“转移”到他们想要的书籍类别。

你会惊奇地发现 cookie是多么频繁地用来在你的浏览器上搜集和存储信息。请打开你的浏览器上的cookie确认特性，以便访问流行的Web站点。

为了避开对cookie的使用，人们想了多办法。支持 cookie的Web浏览器均配有关闭cookie的特性，有些浏览器在安装cookie时允许你查看这些cookie。可以使用某些辅助软件包对发送到浏览器和浏览器返回的cookie进行筛选，还可以对它们进行编辑。 Web站点的设计使你可以对其他Web站点进行匿名冲浪，而cookie不会搜集关于你的信息。

总之，有些人将HTTP cookie视为侵犯隐私权的一个特性，因此你在使用cookie时应该慎重。

21.4 课时小结

在本学时中，我们全面介绍了如何使用HTTP cookie在浏览器上存储信息，供别的CGI程序在以后使用。还介绍了按照预定时间使cookie终止运行，仅为特定Web服务器激活，或者为特定目录激活cookie等特性。最后，讲述了不使用cookie的许多理由以及使用cookie会带来的问题。

21.5 课外作业

21.5.1 专家答疑

问题：我应该如何将多个项目放入一个 HTTP cookie？

解答：最容易的方法是将多个项目组合在单个 cookie中，用域分隔符将各个项分开，如下例所示：

```
$cookie=cookie(-name => 'preferences',
    -value => 'bgcolor=blue,fgcolor=red,banners=no,java=no');
```

然后，当你检索cookie时，可以使用Split将各个项目分开：

```
    $cookie=cookie('preferences');
    @options=split(/,/,  $cookie);
# Now, make a hash with the option as the key,
# and the option's value as the hash value
foreach $option (@options) {
    ($key,$value)=split(/=/, $option)
    $Options{$key}=$value;
}
```

问题：如何使用cookie来跟踪用户在Web页上点击了哪些链接？

解答：在解答这个问题之前，必须指出，有些人将这种跟踪视为是侵犯他人的隐私权。说明这一情况后，再来说明跟踪的一般方法：

1) 编写你的＜A HREF＞链接，将它们纳入一个CGI程序，将真实的目标URL作为参数来传递：

```
<A
HREF="http://server/cgi/redirect.pl?target=http://www.congo.com">Congo
</A>
```

2) 上例中的redirect.pl程序应该使用CGI模块的param函数，以便从参数target中获得真实的URL（http://www. congo.com）：

```
target:

$target_url=param('target');
```

3) 然后使用该值中的目标URL创建一个cookie，其名字你可以在以后查看，如下所示：

```
$tracking_cookie=cookie(-name => 'tracker',
            -value => $target_url,
            -expires => '+1w');
```

4) 然后将重定向的项目与cookie一同发送给浏览器：

```
print redirect(-uri => $target_url,
        -cookie => $tracking_cookie);
```

以后，当浏览器返回到你的 Web站点时，你就可以查找名字为 tracker的cookie，它包含了用户退出你的站点时访问过的 URL。

问题：我在传送cookie时能够将浏览器重定向到另一个 Web页吗？

解答：当然可以。CGI模块的redirect函数也能像header函数那样带有一个-cookie参数。

```
my $cookie=cookie(-name => 'target',
            -value => 'redirected to foo.html');
print redirect(-uri => "http://www.server.com/foo.html",
            -cookie => $cookie);
```

21.5.2 思考题

1) 用cookie来长期存储信息，为什么有时会失败？

a.浏览器会“甩掉”cookie的信息。

b.软件更新时cookie可能丢失。

c.用户可能关闭浏览器对cookie的支持。

2) 若要使cookie在一周后终止运行， cookie函数的-expire选项应该使用什么参数？

a. +7d

b. +1w

c. +10080m

3) 为什么有些人认为cookie会侵犯隐私权？

a. cookie可以用来跟踪用户点击的链接。

b. 被跟踪的cookie信息可以共享，以便建立关于用户的档案资料。

c. cookie信息可用于将某些类别的信息“传递”给用户。

21.5.3 解答

1) 3个答案均成立。

2) a和c均成立。参数+1w无效。

3) 3个答案均成立。

21.5.4 实习

- 扩展背景色修改程序，以便设置前景色和字体，并随机选定一个图形，以便显示在 Web 页上，方法是编辑＜IMG＞标记的目标对象。