



ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Maintaining state through multiple forms

HTTP – stateless protocol

HTTP is a **stateless protocol**

→ Once a web server completes a client's request for a web page, the connection between the two goes away.

→ There is no way for a server to recognize that a sequence of requests all originate from the same client.

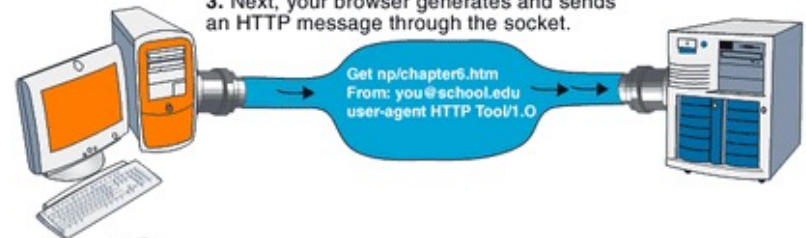
FIGURE 6-9

HTTP messages flow between a browser and a Web server.

1. The URL in the browser's Address bar contains the domain name of the Web server that your browser contacts.

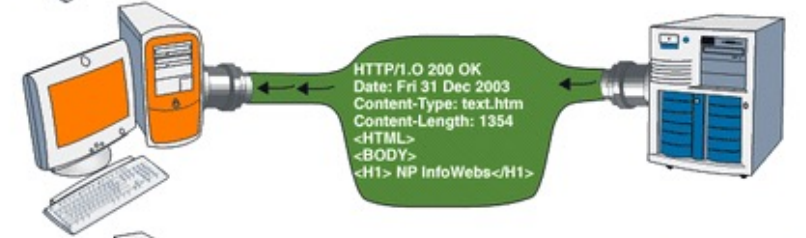
Address

3. Next, your browser generates and sends an HTTP message through the socket.

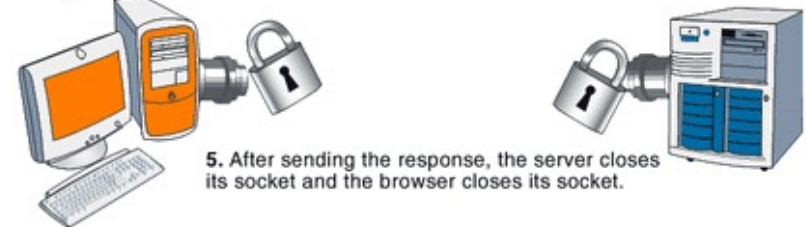


2. Your browser opens a socket and connects to a similar open socket at the Web server.

4. The server sends back the requested HTML document through the open sockets.



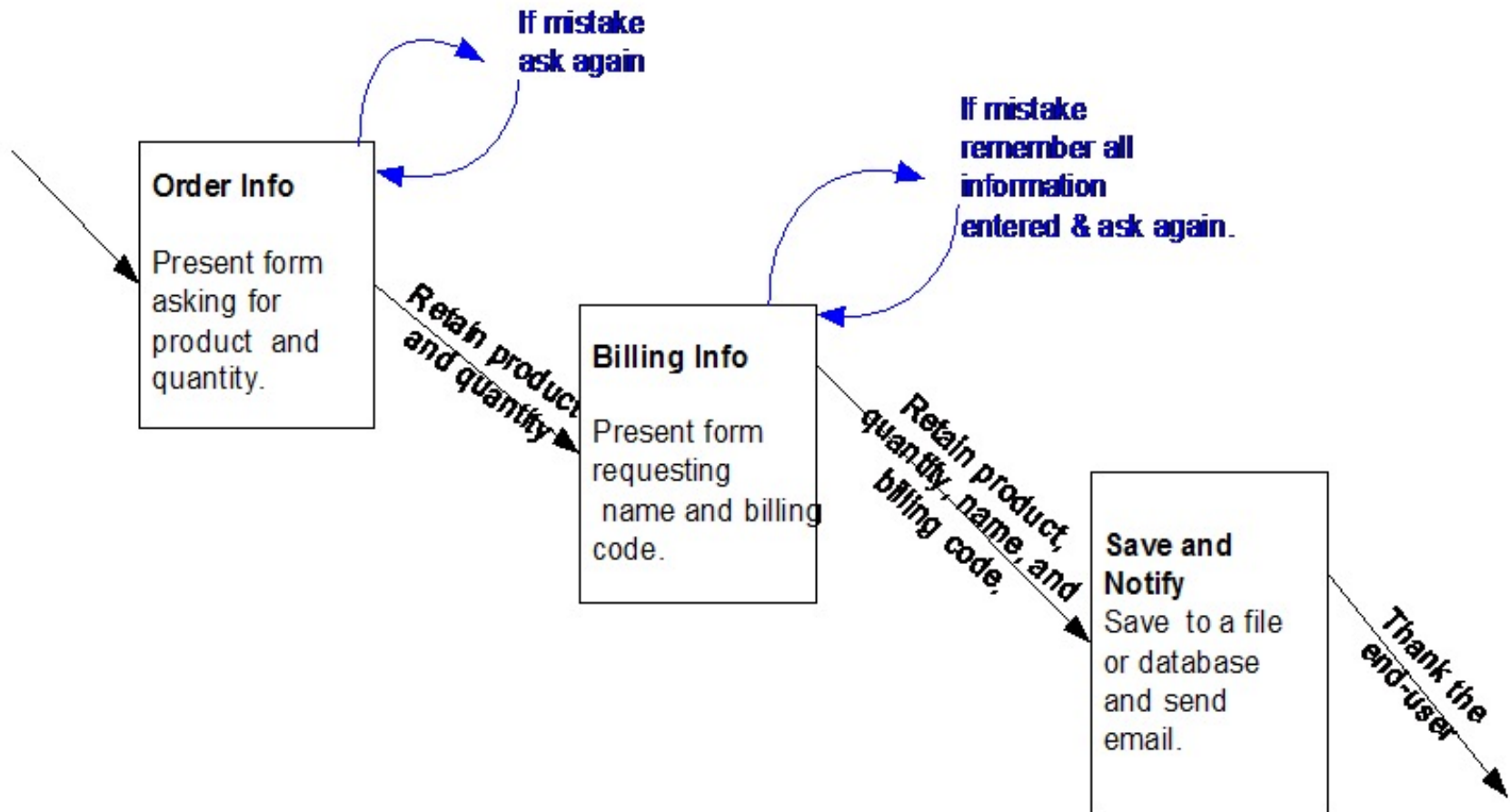
5. After sending the response, the server closes its socket and the browser closes its socket.



What Are Multiple-Form Web Sessions?

- A multiple-form Web session leads the user through a series of HTML forms that work together and pass data from form to form.
- E.g.
 - To build a shopping cart or on-line survey.
 - To save user authentication information from page to page
 - To store persistent user preferences on a site

Example Multiple Screen Session



How to maintain the state through multiform?

- Use tricks to keep track of state information between requests (session tracking)
 - Using hidden form fields
 - URL rewriting: every local URL on which the user might click is dynamically modified to include extra information
 - <http://www.example.com/catalog.php?userid=123>
 - Using cookies: a bit of information that the server give to a client → depends on the client
 - Using session

Content



1. Hidden fields

2. User browser cookies

3. PHP session

1. Hidden fields

- Hidden fields are part of HTML forms
 - Not displayed but value can be accessed in receiving script like any other variable.

```
<input type="hidden" name="preference" value="Likes Power Tools">
```

**Variable name
that will be available
to your PHP program**



**Variable's value in the
PHP program**



A Full Script Example

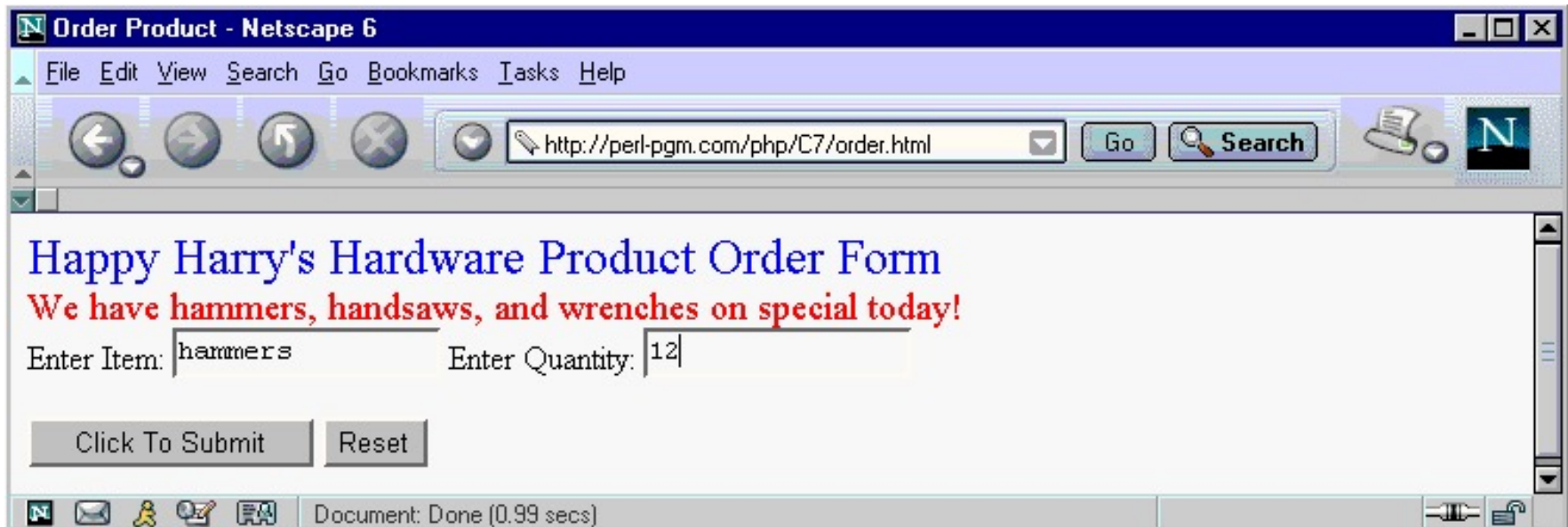
- Consider an example script sets a hidden field
 - Implements the Order Info form
 - on submit sends data to order2.php

PHP Script – order.html

```
1. <html><head><title>Order Product</title></head><body>
2. <form action="order2.php" method="post">
3. <h1> Hardware Product Order Form</h1>
4. <br><p class="highlight">
5. We have hammers, handsaws, and wrenches on special
   today!
6. </p>
7. <input type="hidden" name="sample_hidden"
   value="Welcome!">
8. <br>Enter Item: <input text type="text" size="15"
   maxlength="20" name="product">
9. Enter Quantity: <input text type="text" size="15"
   maxlength="20" name="quantity"><br>
10. <br><input type="submit" value="Click To Submit">
11. <input type = "reset" value="Reset">
12. </form></body></html>
```

The Output ...

The previous code can be executed at
<http://webwizard.aw.com/~phppgm/C7/order.html>



Receiving Hidden Fields in Web Sessions

- Your scripts can receive data from hidden fields like any other data.

- Suppose the following is stored at: order2.php

```
1. <html><head><title> Order Product 2 </title> </head>
2. <body>
3. <form action="order3.php" method="post">
4. <?php $sample_hidden = $_POST["sample_hidden"];
5. $product = $_POST["product"]; $quantity =
   $_POST["quantity"];
6. print "<p class='highlight'>";
7. print "Hidden value=$sample_hidden </p><br>";
8. print "You selected product=$product and
   quantity=$quantity";
```

Receiving PHP Script

```
9. print "<br><br><input type=\"hidden\" name=\"product\"
    value=\"\$product\"> ";
10. print "<input type=\"hidden\" name=\"quantity\"
    value=\"\$quantity\">";
11. print "<input type=\"hidden\"
    name=\"sample_hidden\" value=\"\$sample_hidden\">";
12. print 'Please enter your name: ';
13. print '<input type="text" size="15" maxlength="20"
    name="name">';
14. print ' and billing code: (5 digits)';
15. print '<input type="text" size="5" maxlength="5"
    name="code">';
16. print '<br> <input type=submit value="Process Order">';
17. print '<input type=reset>';
18. ?></form></body></html>
```

Sending email from PHP scripts

- Sometimes it is useful to send email from a PHP script:
 - PHP uses mail() that by default sends e-mail via the Simple Mail Transfer Protocol (SMTP).

`mail(to_address, subject, message, extra_headers);`

Specify the destination email address.

Specify the subject line of the e-mail.

Specify the Text of the email

Specify additional email headers.

Consider the following example ...

```
1. $dest='orders@hardwareville.com';  
2. $subject = 'New Hardware Order';  
3. $message = 'Enclosed is a new order for 12  
   hammers.\n Thanks.';  
4. $extra = 'From: harry@hardwareville.com';  
5. mail( $dest, $subject, $message, $extra );
```

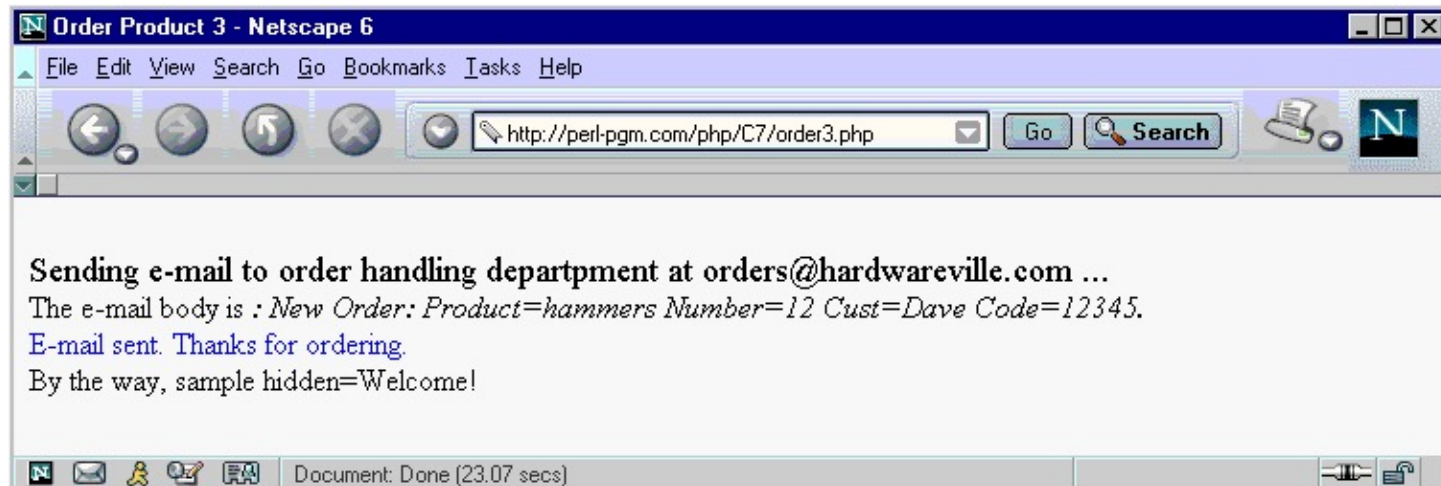
Consider the following full example ...

- Implements save and notify
- Called from order2.php and saved at order3.php
- Can access variables \$product, \$quantity, and \$sample_hidden sent as hidden fields from the Billing Info form.

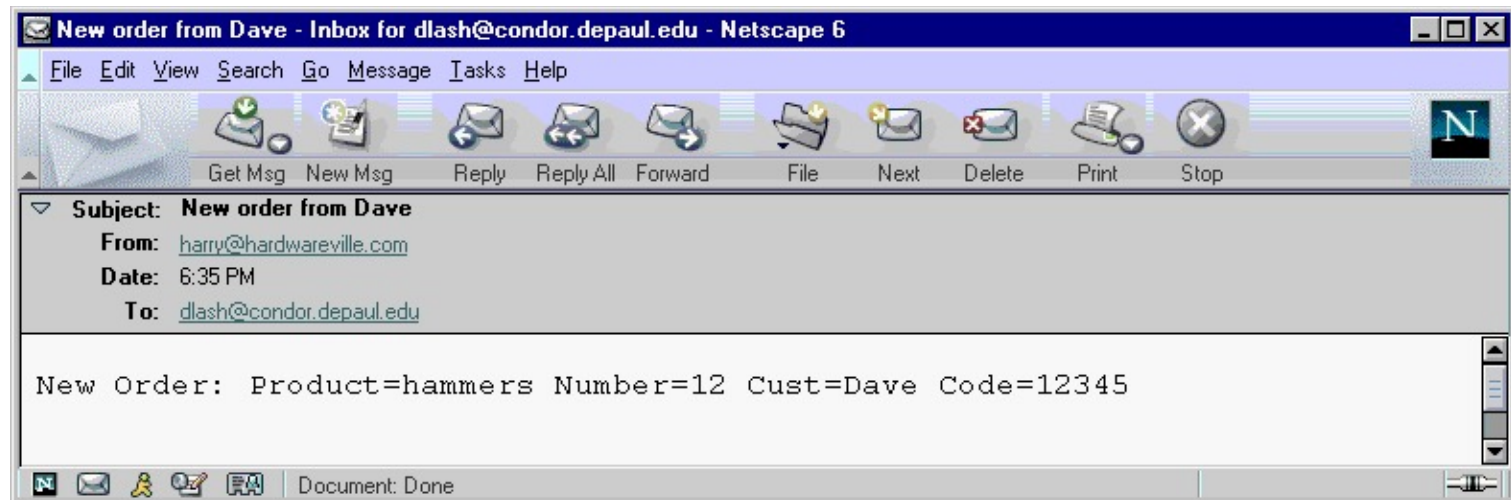
The following PHP Script ...

```
1. <html><head><title>Order Product 3</title> </head><body>
2. <?php
3. $sample_hidden = $ _POST["sample_hidden"];
   quantity=$ _POST["$quantity"];
4. $product = $ _POST["$product"]; $name=$ _POST["$name"];
5. $email='orders@hardwareville.com';
6. $body = "New Order: Product=$product Number=$quantity Cust=$name
   Code=$code";
7. print '<font size=4>';
8. print "<br>Sending e-mail to order handling department at $email
   ... </font>";
9. print "<br>The e-mail body is <i>: $body. </i>";
10. $from = 'harry@hardwareville.com';
11. $subject = "New order from $name";
12. mail($email, $subject, $body, "From: $from");
13. print '<br><font color="blue"> E-mail sent. Thanks for
   ordering. </font>';
14. print "<br>By the way, sample_hidden=$sample_hidden";
15. ?></body></html>
```


Would have the following output ...



Would have the following output ...



Content

1. Hidden fields

→ 2. User browser cookies

3. PHP session

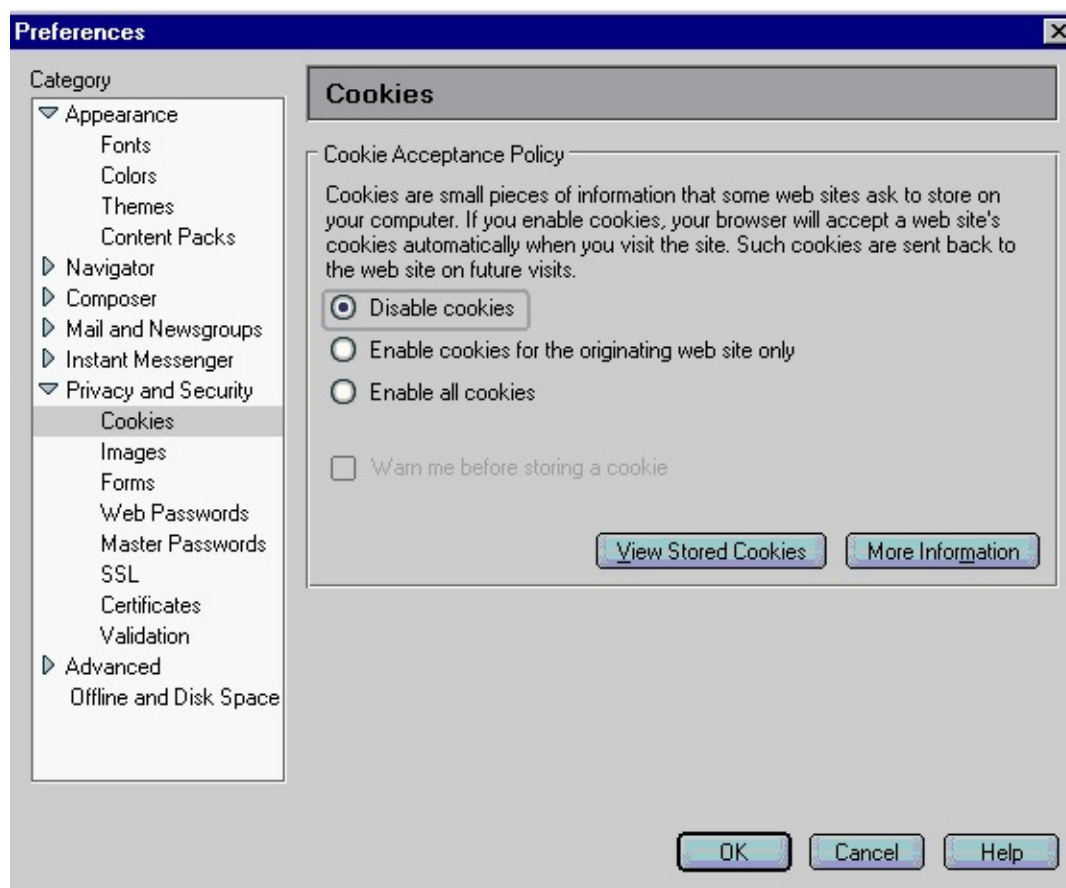
Using Browser Cookies ...

- Cookies are small pieces of data that a Web application can save when a user visits the Web page.
 - Stored on the visitor's hard drive
 - a Web page script can read the previously stored browser cookie data

Understanding Cookie Limitations

- Users can easily disable the cookies feature.
- People move around.
- Users may delete cookies.
- PHP sets limit on cookies

The disable cookie screen in Netscape



Setting and Reading Cookies

- Cookies can be set in memory or on hard disk
 - Set on hard disk are deleted when browser closes
 - Can use the `setcookie()` script
- `setcookie('Customer_name', 'Denise');`

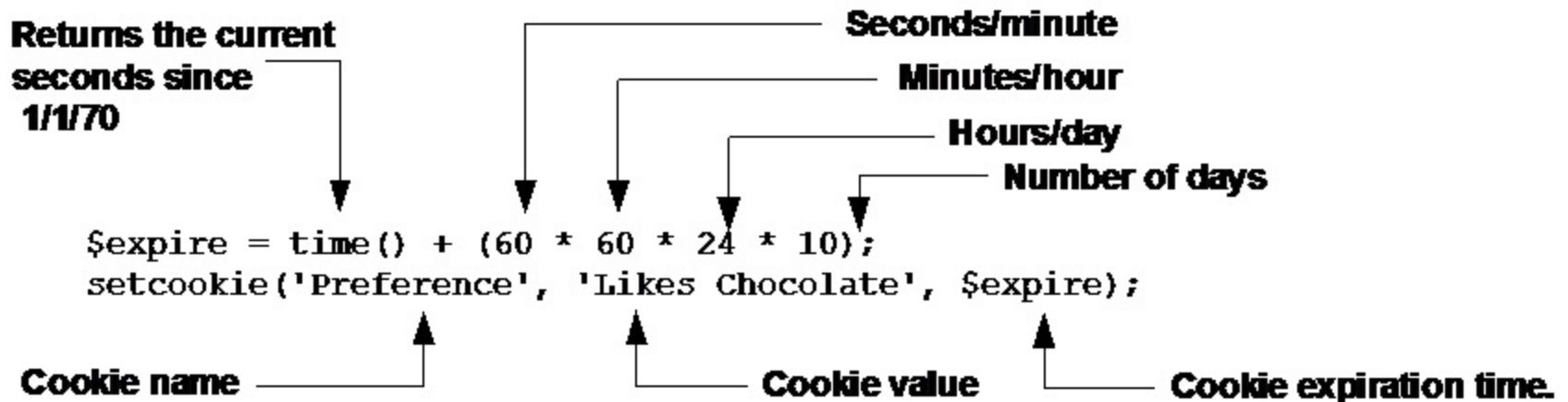
Directs browser
to create a cookie

Specify the
cookie's name

Specify the
cookie's value

Setting A Cookie on a Hard Drive

- You need to use the time() function when want to set a cookie on a hard drive.



A full example of setting a cookie....

- Suppose a front-end web page asks for some survey information:

```
<input type="text" size="15" maxlength="20"  
      name="custname">
```

```
<input type="radio" name="prefers" value="power  
      tools"
```

checked > Power Tools?

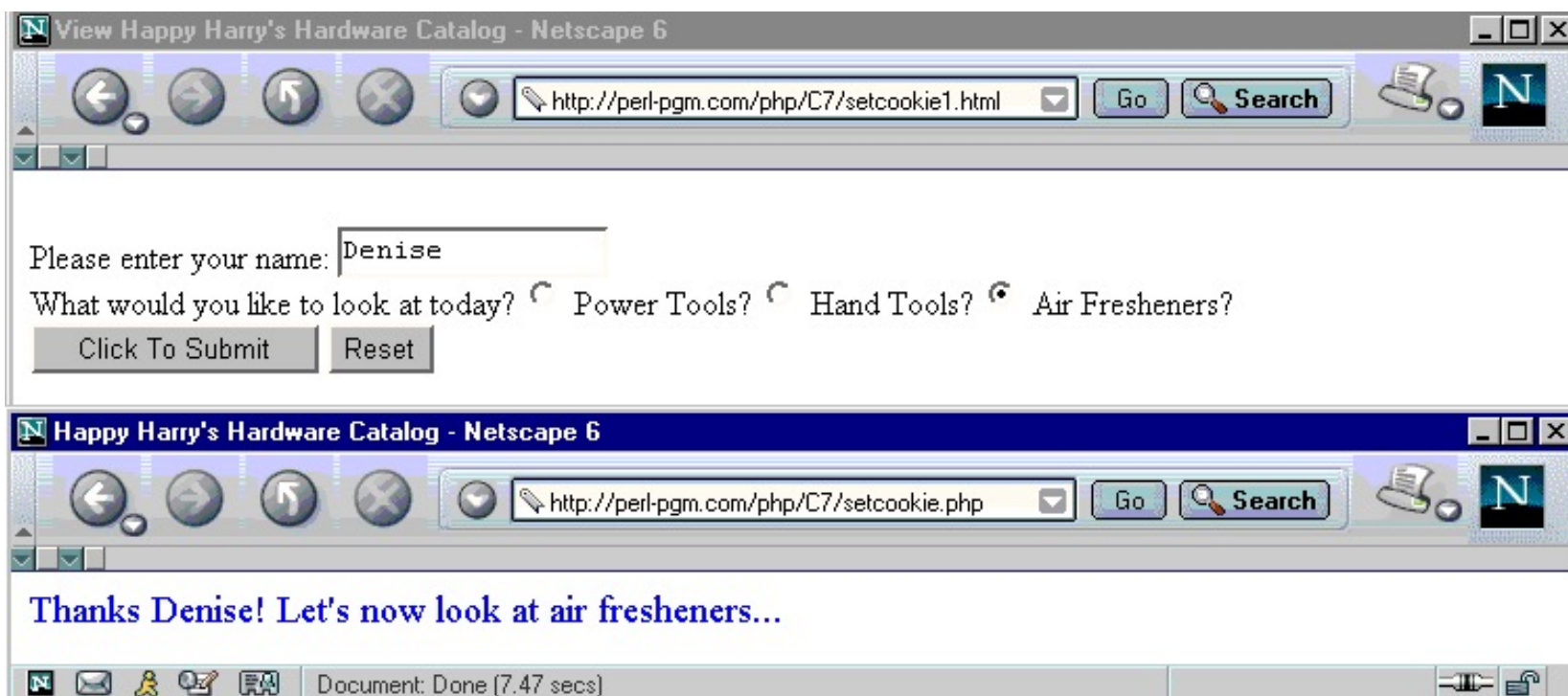
```
<input type="radio" name="prefers"  
      value="hand tools"> Hand Tools?
```

```
<input type="radio" name="prefers" value="air  
fresheners"> Air Fresheners?
```

The following script runs when submitted – setcookie.php

```
1. <?php $prefers = $_POST["prefers"];  
   $expire=$_POST["expire"];  
   $custname=$_POST["custname"];  
2.     $expire = time() + (60 * 60 * 24 * 30);  
3.     setcookie("name", $custname, $expire);  
4.     setcookie("preference", $prefers, $expire);  
5. ?>  
6. <html>  
7. <head><title>Happy Harry's Hardware Catalog  
   </title></head>  
8. <body><font size=4 color="blue">  
9. <?php  
10.  print "Thanks $custname! ";  
11.  print "Let's now look at $prefers... ";  
12. ?> </font></body></html>
```

Would output:



Reading Cookies

- You can read a cookie by using a variable name with the same name as a cookie:
 - print “\$cust_name”;

Reading Cookies with REGISTER_GLOBALS Off

- To read a cookie value use the `$_COOKIE[]` associative array to get the cookie function
- `$cust_name= $_COOKIE[“cust_name”];`

Example Script that read a cookie – readcookie.php

```
1.<html>
2.<head><title>Happy Harry's Hardware Catalog</title>
3.</head><body>
4.<?php
5.  print '<font color="blue" size=4>';
6.  if (isset($name)){
7.    print "Welcome back to our humble hardware site, $name.";
8.  } else {
9.    print '<font color="red">';
10.    print 'Welcome to our humble hardware site.</font>';
11.  }
12.  if ($preference == 'hand tools'){
13.    print '<br> We have hammers on sale for 5 dollars!';
14.  } elseif ($preference == 'power tools'){
15.    print '<br> We have power drills on sale for 25 dollars!';
16.  } elseif ($preference == 'air fresheners'){
17.    print '<br> We now carry extra-strength air fresheners!';
18.  } else {
19.    print '<br> <font color="red">';
20.    print 'We have drills and hammers on special today!';
21.  }
22. ?></font></html>
```

Example Script that read a cookie

```
1.<html>
2.<head><title>Happy Harry's Hardware Catalog</title>
3.</head><body>
4.<?php $name = $ COOKIE["name"]; $preference =
   $_COOKIE["preference"];
5.  print '<font color="blue" size=4>';
6.  if (isset($name)){
7.    print "Welcome back to our humble hardware site, $name.";
8.  } else {
9.    print '<font color="red">';
10.    print 'Welcome to our humble hardware site.</font>';
11.  }
12.  if ($preference == 'hand tools'){
13.    print '<br> We have hammers on sale for 5 dollars!';
14.  } elseif ($preference == 'power tools'){
15.    print '<br> We have power drills on sale for 25 dollars!';
16.  } elseif ( $preference == 'air fresheners'){
17.    print '<br> We now carry extra-strength air fresheners!';
18.  } else {
19.    print '<br> <font color="red">';
20.    print 'We have drills and hammers on special today!';
21.  }
22.  ?></font></html>
```

Content

1. Hidden fields
2. User browser cookies
- 3. PHP session

PHP Sessions

- PHP supports two functions that enable you to retain data between forms
 - **`session_start()`** – either starts a new session or resumes one if a session exists
 - Run at the start of every script
 - By default creates a unique session ID stored as a cookie
 - **`session_register()`** – registers one or more variables as session variables

```
$name = 'Matthew';
```

```
$preference = 'Soccer Equipment';
```

```
session_register('name', 'preference');
```

Example PHP Code

```
1. <?php session_start(); ?>
2. <html><head><title>Order Product</title>
3. </head><body>
4. <form action="sessions2order.php" method="post">
5. <font color=blue size=5> Hardware Product Order Form
   </font>
6. <br> We have hammers, handsaws, and wrenches.
7. <br>Enter Item: <input text type="text" size="15"
   maxlength="20" name="product">
8. Enter Quantity: <input text type="text" size="15"
   maxlength="20" name="quantity"><br>
9. <?php
10.     $sample_hidden='Welcome Again!';
11.     session_register('sample_hidden');
12. ?>
13. <br><input type="submit" value="Click To Submit">
14. <input type = "reset" value = "Reset" >
15. </body></html>
```

Since PHP 5.4

`$ SESSION['sample_hidden ']=$sample_hidden;`

Use the following script to read the session data - sessions2order.php

```
1.  <?php session_start() ?>
2.  <html><head><title> Order Product 2 </title> </head>
3.  <body>
4.  <form action="sessions3order.php" method="post">
5.  <?php $sample_hidden = $_SESSION['sample_hidden'];
6.      print "<h1> Sample hidden= $sample_hidden</h1>";
7.      print "<br>You selected product=$product and
               quantity=$quantity";
8.      session_register('product');    session_register('quantity');
9.      print '<br>Please enter your name';
10.  print '<input text type="text" size="15" maxlength="20"
           name="name">';
11.  print ' and Billing Code: (5 digits)';
12.  print '<input text type="text" size="5" maxlength="5"
           name="code">';
13.  print '<br> <input type=submit value="Process Order">';
14.  print '<input type=reset>';
15.  print '</form></body></html>';
16.  ?>
```

Example output

This script can be executed at:

<http://webwizard.aw.com/~phppgm/C7/ordersession.php>

The image shows two overlapping Netscape 6 browser windows. The top window, titled "Order Product - Netscape 6", displays a form titled "Hardware Product Order Form". The form text says "We have hammers, handsaws, and wrenches." and contains two input fields: "Enter Item:" with the value "hammers" and "Enter Quantity:" with the value "12". Below the fields are "Click To Submit" and "Reset" buttons. The bottom window, titled "Order Product 2 - Netscape 6", displays a confirmation page titled "Sample hidden= Welcome Again!". The text on the page says "You selected product=hammers and quantity=12" and "Please enter your name" followed by an input field, and "and Billing Code: (5 digits)" followed by another input field. Below these fields are "Process Order" and "Reset" buttons. Both windows show the Netscape 6 interface with a toolbar and a menu bar.

Order Product - Netscape 6

http://perl-pgm.com/php/C7/ordersession.php

Go Search

Home My Netscape Search Shop Net2Phone Instant Message Instant Internet MyPresario Search Shop

Hardware Product Order Form

We have hammers, handsaws, and wrenches.

Enter Item: Enter Quantity:

Order Product 2 - Netscape 6

http://perl-pgm.com/php/C7/sessions2order.php

Go Search

Sample hidden= Welcome Again!

You selected product=hammers and quantity=12

Please enter your name and Billing Code: (5 digits)

Document: Done (1.15 secs)

Some session extras

- `session_is_registered()` - can be used to determine if a variable comes from a session:

```
if (session_is_registered('name')) {  
    print "got name=$name from session";  
} else {  
    print "name=$name not set from session";  
}
```

PHP 5.4

`if(isset($_SESSION[$name]))`

Session Extras - \$_SESSION

- Use \$_SESSION Associative array when REGISTER_GLOBALS are off in php.ini
 - Do not need to use session_register()
`session_start();`
`$_SESSION['sample_hidden'] = 'Welcome!';`

Summary

- Hidden fields are HTML form fields you can use to set a variable name and variable value without displaying them on a form.
- Cookies provide a way for Web server applications to store small pieces of data on the user's hard disk.
- PHP session functions provide a convenient way to retain data between PHP scripts.
 - Use `session_start()` and `session_register()` functions to start sessions and define session variables, respectively

Question?

