```
like 2
maybe 1
or 2
oranges 1
than 1
you 3
```

9.10 Form Handling

One common way for a browser user to interact with a Web server is through forms. A form is presented to the user, who is invited to fill in the text boxes and click the buttons of the form. The user submits the form to the server by clicking the form's *Submit* button. The contents of the form are encoded and transmitted to the server, which must use a program to decode the contents, perform whatever computation is necessary on the data, and produce output in the form of a markup document that is returned to the client. When PHP is used to process form data, it implicitly decodes the data.

It may seem strange, but when PHP is used for form handling, the PHP script is embedded in an HTML document, as it is with other uses of PHP. Although it is possible to have a PHP script handle form data in the same HTML document that defines the form, it is perhaps clearer to use two separate documents. For this latter case, the document that defines the form specifies the document that handles the form data in the `action` attribute of its `<form>` tag.

PHP can be configured so that form data values are directly available as implicit variables whose names match the names of the corresponding form elements. However, this implicit access is not allowed in many Web servers (through the configuration of PHP), because it creates a security risk. The recommended approach is to use the implicit arrays `$_POST` and `$_GET` for form values. These arrays have keys that match the form element names and the values that were input by the client. For example, if a form has a text box named `phone` and the form method is POST, the value of that element is available in the PHP script as follows:

```
$_POST["phone"]
```

The following is an HTML document that presents a form for popcorn sales:

```
<!DOCTYPE html>
<!-- popcorn3.html - This describes the popcorn sales form -->
<html lang = "en">
  <head>
    <title> Popcorn Sales - for PHP handling </title>
    <meta charset = "utf-8" />
    <style type = "text/css">
```

```
      td, th, table {border: thin solid black;}
    </style>
  </head>
  <body>
    <form action = "http://localhost/popcorn3.php"
          method = "post">
     <h2> Welcome to Millennium Gymnastics Booster Club Popcorn
          Sales </h2>
      <table>

<!-- Text widgets for the customer's name and address -->
        <tr>
          <td> Buyer's Name: </td>
          <td> <input type = "text" name = "name"
                      size = "30" /></td>
        </tr>
        <tr>
          <td> Street Address: </td>
          <td> <input type = "text" name = "street"
                      size = "30" /></td>
        </tr>
        <tr>
          <td> City, State, Zip: </td>
          <td> <input type = "text" name = "city"
                      size = "30" /></td>
        </tr>
      </table>
      <p />
      <table>

<!-- First, the column headings -->
        <tr>
          <th> Product </th>
          <th> Price </th>
          <th> Quantity </th>
        </tr>

<!-- Now, the table data entries -->
        <tr>
          <td> Unpopped Popcorn (1 lb.) </td>
          <td> $3.00 </td>
          <td>
            <input type = "text" name = "unpop"
                   size = "3" /></td>
        </tr>
```

```
          <tr>
            <td> Caramel Popcorn (2 lb. canister) </td>
            <td> $3.50 </td>
            <td>
              <input type = "text" name = "caramel"
                     size = "3" /> </td>
          </tr>
          <tr>
            <td> Caramel Nut Popcorn (2 lb. canister) </td>
            <td> $4.50 </td>
            <td>
              <input type = "text" name = "caramelnut"
                     size = "3" /> </td>
          </tr>
          <tr>
            <td> Toffey Nut Popcorn (2 lb. canister) </td>
            <td> $5.00 </td>
            <td>
              <input type = "text" name = "toffeynut"
                     size = "3" /> </td>
          </tr>
        </table>
        <p />

<!-- The radio buttons for the payment method -->
        <h3> Payment Method </h3>
        <p>
          <input type = "radio" name = "payment" value = "visa"
                 checked = "checked" />
            Visa <br />
          <input type = "radio" name = "payment" value = "mc" />
            Master Card <br />
          <input type = "radio" name = "payment"
                  value = "discover" />
            Discover <br />
          <input type = "radio" name = "payment" value = "check" />
            Check <br /> <br />

<!-- The submit and reset buttons -->
          <input type = "submit" value = "Submit Order" />
          <input type = "reset" value = "Clear Order Form" />
        </p>
      </form>
    </body>
</html>
```

Figure 9.5 displays the output of `popcorn3.html`, after it has been filled out.

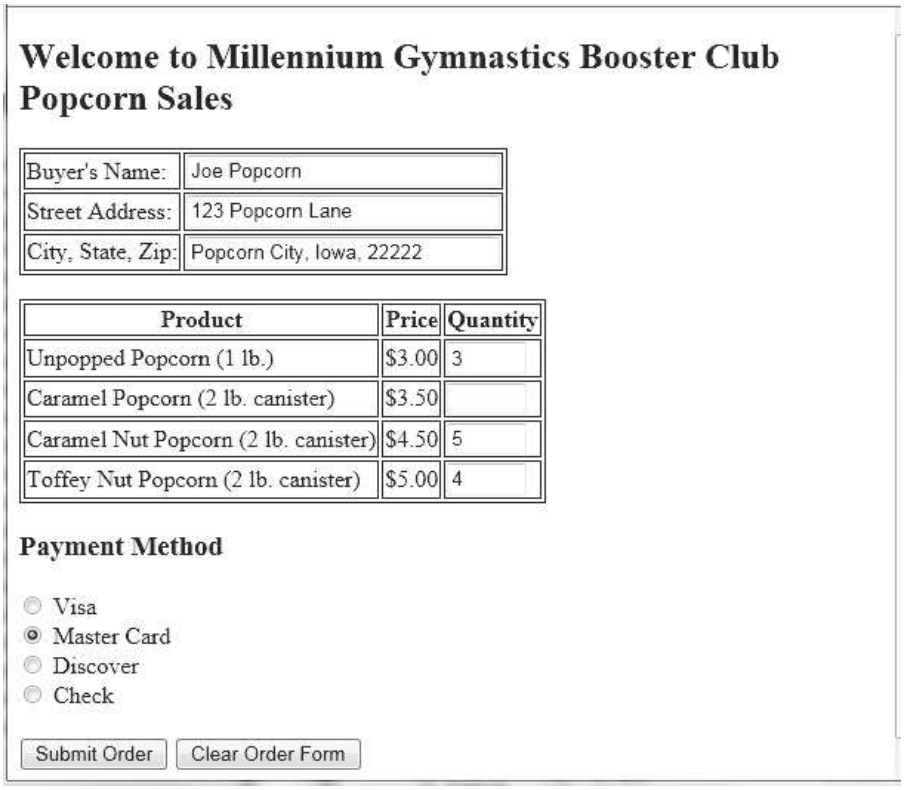


Figure 9.5 Display of the output of `popcorn3.html`

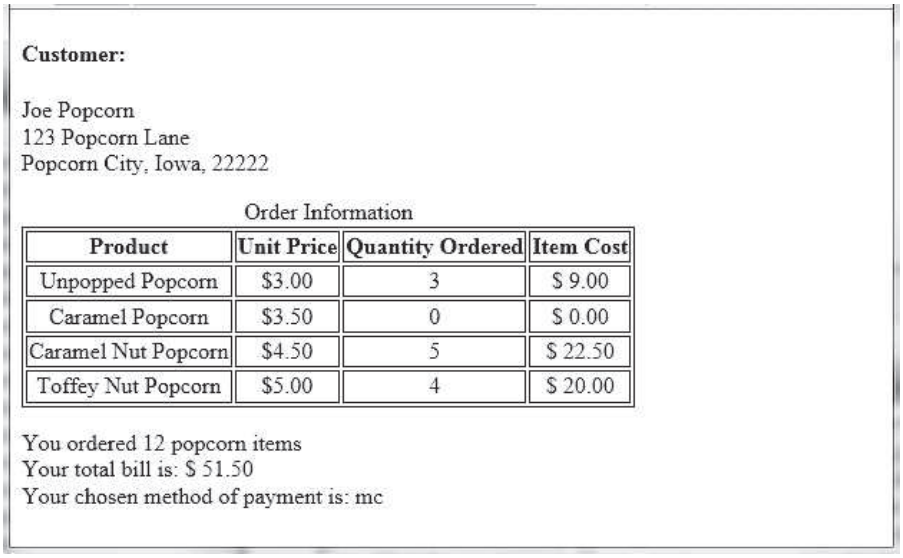
The PHP script that handles the data from the form described in `popcorn3.html` follows. It uses the form data to compute the cost of each product, the total cost of the order, and the total number of items ordered. The name, unit price, number ordered, and total cost for each product are presented to the client in a table defined with interwoven HTML markup and PHP script. The table structure is described with HTML, but the contents of some of the data cells are defined with PHP. Here is the document:

```
<!DOCTYPE html>
<!-- popcorn3.php - Processes the form described in
     popcorn3.html
   -->
<html lang = "en">
  <head>
    <title> Process the popcorn3.html form </title>
```

```
    <meta charset = "utf-8" />
    <style type = "text/css">
      td, th, table {border: thin solid black;}
    </style>

  </head>

  <body>
    <?php

// Get form data values
      $unpop = $_POST["unpop"];
      $caramel = $_POST["caramel"];
      $caramelnut = $_POST["caramelnut"];
      $toffeynut = $_POST["toffeynut"];
      $name = $_POST["name"];
      $street = $_POST["street"];
      $city = $_POST["city"];
      $payment = $_POST["payment"];

// If any of the quantities are blank, set them to zero
      if ($unpop == "") $unpop = 0;
      if ($caramel == "") $caramel = 0;
      if ($caramelnut == "") $caramelnut = 0;
      if ($toffeynut == "") $toffeynut = 0;

// Compute the item costs and total cost
      $unpop_cost = 3.0 * $unpop;
      $caramel_cost = 3.5 * $caramel;
      $caramelnut_cost = 4.5 * $caramelnut;
      $toffeynut_cost = 5.0 * $toffeynut;
      $total_price = $unpop_cost + $caramel_cost +
                     $caramelnut_cost + $toffeynut_cost;
      $total_items = $unpop + $caramel + $caramelnut + $toffeynut;

// Return the results to the browser in a table
    ?>
    <h4> Customer: </h4>
    <?php
      print ("$name <br /> $street <br /> $city <br />");
    ?>
    <p /> <p />
    <table>
      <caption> Order Information </caption>
```

```
        <tr>
          <th> Product </th>
          <th> Unit Price </th>
          <th> Quantity Ordered </th>
          <th> Item Cost </th>
        </tr>
        <tr>
          <td> Unpopped Popcorn </td>
          <td> $3.00 </td>
          <td> <?php print ("$unpop"); ?> </td>
          <td> <?php printf ("$ %4.2f", $unpop_cost); ?>
          </td>
        </tr>
        <tr>
          <td> Caramel Popcorn </td>
          <td> $3.50 </td>
          <td> <?php print ("$caramel"); ?> </td>
          <td> <?php printf ("$ %4.2f", $caramel_cost); ?>
          </td>
          </tr>
        <tr>
          <td> Caramel Nut Popcorn </td>
          <td> $4.50 </td>
          <td> <?php print ("$caramelnut"); ?> </td>
          <td> <?php printf ("$ %4.2f", $caramelnut_cost); ?>
          </td>
        </tr>
        <tr>
          <td> Toffey Nut Popcorn </td>
          <td> $5.00 </td>
          <td> <?php print ("$toffeynut"); ?> </td>
          <td> <?php printf ("$ %4.2f", $toffeynut_cost); ?>
          </td>
        </tr>
      </table>
      <p /> <p />

      <?php
          print "You ordered $total_items popcorn items <br />";
          printf ("Your total bill is: $ %5.2f <br />", $total_price);
          print "Your chosen method of payment is: $payment <br />";
      ?>
    </body>
  </html>
```

Notice that the `printf` function is used to implement the numbers that represent money, so exactly two digits appear to the right of the decimal points. Figure 9.6 displays the output of `popcorn3.php`.



Customer:

Joe Popcorn
123 Popcorn Lane
Popcorn City, Iowa, 22222

Order Information

Product	Unit Price	Quantity Ordered	Item Cost
Unpopped Popcorn	$3.00	3	$ 9.00
Caramel Popcorn	$3.50	0	$ 0.00
Caramel Nut Popcorn	$4.50	5	$ 22.50
Toffey Nut Popcorn	$5.00	4	$ 20.00

You ordered 12 popcorn items
Your total bill is: $ 51.50
Your chosen method of payment is: mc

Figure 9.6 Display of the output of `popcorn3.php`

9.11 Cookies

PHP includes convenient support for creating and using cookies.

9.11.1 Introduction to Cookies

A *session* is the time span during which a browser interacts with a particular server. A session begins when a browser connects to the server. That session ends either when the browser is terminated or because the server terminated the session because of client inactivity. The length of time a server uses as the maximum time of inactivity is set in the configuration of the server. For example, the default maximum for some servers is 30 minutes.

The HTTP protocol is essentially stateless: It includes no means for storing information about a session that is available to a subsequent session. However, there are a number of different reasons why it is useful for the server to be capable of connecting a request made during a session to the other requests made by the same client during that session, as well as to requests made during previous and subsequent sessions.

One of the most common needs for information about a session is to implement shopping carts on Web sites. An e-commerce site can have any number of simultaneous online customers. At any time, any customer can add an item to or remove an item from his or her cart. Each user's shopping cart is identified by