PHP 02

Forms

Embedding PHP code in HTML

Processing Forms

while

for

Based on Rasmus Lerdorf& Kevin Tatroe: Programming PHP. Sebastopol: O'Reilly, 2002.

W05/161B/Daniel Sauter

Forms (HTML)

Forms provide a method for true interaction between users and the publisher of a website.

Can be used to provide a customized response on-the-fly or to collect data. E.g. surveys, guestbooks, e-commerce systems

```
<form action="admin.php">
   <!-- form elements -->
</form>
```

Attributes

`action` = URL

Specifies the url that will process the form (required). Default is the current URL.

`method` = get / post

Specifies HTTP method used to submit the form data. get is the default method. Use post if you send the form data to an email address. W3C recommends the use of post method which puts the data in a separate part in the body while processing the HTTP request. The get method appends the data to the URL itself (this method is limited to 256 characters).

Types of input elements

```
<input type=button>
<input type=checkbox>
<input type=file>
<input type=hidden>
<input type=image>
<input type=password>
<input type=radio>
<input type=text>
<input type=submit>
```

Example: Text input fields

```
<html>
<head>
<title>Forms</title>
</head>
<body>
  <form action="input_text.htm">
    <p>Firstname:<br>
    <input name="firstname" type="text" size="30">
    </p>
    <p>Lastname:<br>
    <input name="lastname" type="text" size="30">
    </p>
  </form>
</body>
</html>
```

Textarea

```
<html>
<head>
<title>HTML page with a text area</title>
</head>
<body>
  <form action="comments.php">
    <p>Input your comment:<br>
    <textarea name="user_eingabe" cols="50" rows="10">

    </textarea>
    </p>
  </form>
</body>
</html>
```

Select

`<select> ... </select>`

Defines a multiple-choice menu

```
<html>
<head>
<title>lists</title>
</head>
<body>
  <form action="select.php">
      <select name="top5" size="1">
      <option value="1">Michael Jackson</option>
      <option value="2">Tom Waits</option>
      <option value="3">Nina Hagen</option>
      <option value="4">Luziano Pavarotti</option>
    </select>
  </form>
</body>
</html>
```

Radio Buttons and Checkboxes

```
<html>
<head>
<title>Gender</title>
</head>
<body>
  <form action="survey.php">
    <input type="radio" name="gender" value="male">
    <input type="radio" name="gender" value="female">
    <input type="radio" name="gender" value="other">

    <input type="checkbox" name="married" value="yes">

  </form>
</body>
</html>
```

name Attribute [E: resources/PHP/processing_variables]

If a name has been assigned to a form element, PHP stores the value of this form element in a variable with the identical name which has been given to the form element.

E.g. this is an html page called input.html. It will send the data from the form to process.php (pay attention to the endings!):

input.html:

```
<html>
<body>
  <form action="process.php" method="post">
    <input type="text" name="birthdate">
    <input type="submit" value="send birthdate">
  </form>
</body>
```

Embedding PHP in HTML

process.php:

```
<html>
<head>
<title>lists</title>
</head>
<body>
 <?php
    echo "The birthday you entered is:";
    echo "<br>";
    echo $birthdate;
 ?>
</body>
</html>
```

Appending data to the URL

As mentioned earlier talking about the send methods post and get, data can be appended to the URL directly.

Sending the birth date “1980” to process.php using a text input field with the name birthdate, is equivalent to calling the URL:

process.php?birthdate=1980

the data is seperated from the PHP page with a “?”.

Multiple variables and values can be attached to the URL as follows:

process.php?birthdate=1980&gender=female&funny=yes

PHP ‘for’ loops

Controls a sequence of repetitions. A for() structure has three parts: init, test, and update. Each part must be separated by a semi-colon ";". The loop continues until the test evaluates to false.

When a for() structure is executed, the following sequence of events occurs:

1. The init statement is executed
2. The test is evaluated to be true or false
3. If the test is true, jump to step 4. If the test is False, jump to step 6
4. Execute the statements within the block
5. Execute the update statement and jump to step 2
6. Exit the loop.

```
for ($i = 1; $i <= 10; $i++) {
    echo $i;
}
```

PHP ‘while’ loops

The while structure executes a series of statements continuously while the expression is true. The expression must be updated during the repetitions or the program will never "break out" of while().

```
i = 1;
while ($i <= 10) {
    echo $i++;
}
```

Workshop Exercises

Using PHP:

1. Write ‘hello world’
2. Make a form with one input field and a submit button. After submitting, write the content of this input field on the screen, and, the input field as mentioned before.
3. Make a form including a popup menu, a checkbox, a text area, and a input field, and a radio button. Write all contents/ values of these forms onto the screen.