**Lektion 11: Verarbeitung von Formular-Variablen**

Interaktive Webseiten benötigen Eingaben oder auch Angaben vom Nutzer. Eine der gebräuchlichsten Möglichkeiten ist hier diese Eingaben über Formulare (Forms) zu erhalten.

In dieser Lektion werden wir uns anschauen, wie man Formulare erstellt und die Eingaben auf dem Server verarbeitet.

**<form>**

Wenn Sie ein Formular erstellen, gibt es zwei wichtige Attribute: **action** und **method**.

**action**

Wird für die Angabe der URL benötigt, an die die Formulardaten übermittelt werden. Hier ist dies jetzt die PHP-Datei, die die Eingaben verarbeiten soll.

**method**

Kann entweder den Wert "post" oder "get" haben, welches zwei verschiedene Arten sind, Daten weiterzugeben. Zur Zeit brauchen Sie noch nicht viel über die Unterschiede zu wissen. Nur so viel: mit "get" werden die Daten über die URL weitergegeben und mit "post" als Datenblock über den 'standard input service' (STDIN). In der vorigen Lektion haben wir uns angeschaut, wie Daten aus der URL über [$\_GET](http://php.net/manual/en/reserved.variables.get.php) abgefragt werden können. In dieser Lektion werden wir schauen, wie Daten, die über ein Formular per "post" übermittelt wurden, weiterverarbeitet werden können.

**Eine HTML-Seite mit einem Formular**

Die Seite, die das Formular enthält, muss keine PHP-Datei sein (kann aber). Es muss noch nicht mal die gleiche Seite sein, die später die Daten weiterverarbeitet.

In unserem ersten Beispiel, werden wir mit einem sehr einfachen Formular mit nur einem Textfeld arbeiten:

 <html>

 <head>

 <title>Forms</title>

 </head>

 <body>

 <h1>Bitte geben Sie Ihren Namen ein!</h1>

 <form method="post" action="handler.php">

 <input type="text" name="username">

 <input type="submit">

 </form>

 </body>

 </html>

Das Ergebnis im Browser ist ein Formular:



Jetzt kommen wir zum 'lustigen' Teil: Abfrage und Verarbeitung der Daten mit PHP.

**Abfrage der Formular-Daten mit PHP**

Wenn Sie Daten, die mit einem Formular (post-Methode) übermitteln wurden, abfragen möchten, benutzen Sie [$\_POST](http://www.php.net/manual/en/reserved.variables.post.php):

 $\_POST["Feldname"];

Dies enthält den Wert eines Feldes im Formular. Lassen Sie und dies an einem Beispiel zeigen.

Zuerst erstellen Sie eine Seite mit dem Formular wie es oben zu sehen ist. Danach erstellen Sie eine PHP-Seite mit dem Namen "handler.php" (Beachten Sie, dass dies der Name der Seite ist, die wir in unserem <form> im action-Attribut stehen haben).

Die Datei "handler.php" sollte den folgenden Inhalt haben:

 <html>

 <head>

 <title>Forms</title>

 </head>

 <body>

 <?php

 echo "<h1>Hallo " . $\_POST["username"] . "</h1>";

 ?>

 </body>

 </html>

[Beispiel anzeigen](http://de.html.net/tutorials/php/lesson11_ex1.php)

**Nutzereingaben und Bedingungen**

Im nächsten Beispiel werden wir den Nutzereingaben in Bedingungen (Conditions) benutzen. Zuerst brauchen wir wieder ein Formular:

 <html>

 <head>

 <title>Forms</title>

 </head>

 <body>

 <form method="post" action="handler.php">

 <p>Wie ist Ihr Name?:</p>

 <input type="text" name="username"></p>

 <p>Welches ist Ihre Lieblingsfarbe?:

 <input type="radio" name="favoritecolor" value="r" /> Rot

 <input type="radio" name="favoritecolor" value="g" /> Grün

 <input type="radio" name="favoritecolor" value="b" /> Blau </p>

 <input type="submit" value="Submit" />

 </form>

 </body>

 </html>

Im Browser sollte dies in etwa so aussehen:



Jetzt werden wir diese Eingaben verwenden, um eine Seite zu erstellen, die automatisch die Hintergrundfarbe zu der Farbe ändert, die der Nutzer als Lieblingsfarbe festgelegt hat. Wir können dies mit Hilfe einer Bedingung (Condition) (siehe [Lektion 6](http://de.html.net/tutorials/php/lesson6.php)), die die Daten des Nutzers aus dem Formular verwendet, bewerkstelligen.

 <?php

 $strHeading = "<h1>Hallo " . $\_POST["username"] . "</h1>";

 switch ($\_POST["favoritecolor"]) {

 case "r":

 $strBackgroundColor = "rgb(255,0,0)";

 break;

 case "g";

 $strBackgroundColor = "rgb(0,255,0)";

 break;

 case "b":

 $strBackgroundColor = "rgb(0,0,255)";

 break;

 default:

 $strBackgroundColor = "rgb(255,255,255)";

 break;

 }

 ?>

 <html>

 <head>

 <title>Forms</title>

 </head>

 <body style="background: <? echo $strBackgroundColor; ?>;">

 <? echo $strHeading; ?>

 </body>

 </html>

Die Hintergrundfarbe ist weiß, wenn der Nutzer im Formular keine Angaben zu seiner Lieblingsfarbe gemacht hat. Dies passiert mit der **default**-Option, die ein Verhalten festlegt, wenn keine der vorherigen Bedingungen erfüllt ist.

Aber was passiert, wenn der Nutzer seinen Namen nicht ausfüllt? Dann ist da nur "Hallo" zu lesen. Wir fügen jetzt aber eine zusätzliche Bedingung hinzu, die das ändert.

 <?php

 $strUsername = $\_POST["username"];

 If ($strUsername != "") {

 $strHeading = "<h1>Hallo " . $\_POST["username"] . "</h1>";

 }

 else {

 $strHeading = "<h1>Hallo Fremder!</h1> ";

 }

 switch ($\_POST["favoritecolor"]) {

 case "r":

 $strBackgroundColor = "rgb(255,0,0)";

 break;

 case "g";

 $strBackgroundColor = "rgb(0,255,0)";

 break;

 case "b":

 $strBackgroundColor = "rgb(0,0,255)";

 break;

 default:

 $strBackgroundColor = "rgb(255,255,255)";

 break;

 }

 ?>

 <html>

 <head>

 <title>Form</title>

 </head>

 <body style="background: <? echo $strBackgroundColor; ?>;">

 <? echo $strHeading; ?>

 </body>

 </html>

[Beispiel anzeigen](http://de.html.net/tutorials/php/lesson11_ex2.php)

Im obigen Beispiel verwenden wir eine Bedingung um die Eingabe des Nutzers zu **validieren**. In diesem Fall wird es wahrscheinlich nicht so wichtig sein, wenn der Nutzer seinen Namen nicht einträgt. Aber wenn Sie später anspruchsvollere Sachen programmieren, ist es unerlässlich in Betracht zu ziehen, dass ein Nutzer Formularfelder nicht immer so ausfüllt, wie Sie es sich gedacht haben.

**Beispiel: Kontaktformular**

Mit Ihrem Wissen über PHP und Formulare können Sie nun schon ein Kontaktformular, unter Verwendung der Function [mail](http://php.net/manual/en/function.mail.php), die die folgende Syntax hat, erstellen:

 mail*(Empfänger, Betreff, Nachricht);*

Zuerst brauchen wir wieder ein einfaches HTML-Formular:

 <html>

 <head>

 <title>Kontaktformular</title>

 </head>

 <body>

 <h1>Kontaktformular</h1>

 <form method="post" action="handler.php">

 <p>Betreff:<br /><input type="text" name="subject" /></p>

 <p>Nachricht:<br /><textarea name="message"></textarea></p>

 <input type="submit">

 </form>

 </body>

 </html>

Dann brauchen wir ein PHP-Skript, um die Eingaben des Nutzers zu versenden:

 <html>

 <head>

 <title>Send Mail</title>

 </head>

 <body>

 <?php

 // Empfänger (tragen Sie hier Ihre eMail-Adresse ein)

 $strEmail = "name@meinedomain.de";

 // Eingaben des Nutzers auslesen

 $strSubject = $\_POST["subject"];

 $strMessage = $\_POST["message"];

 mail($strEmail,$strSubject,$strMessage);

 echo "Nachricht versandt.";

 ?>

 </body>

 </html>

Bitte beachten Sie, dass dieses Beispiel nur funktionieren wird, wenn Sie Zugang zu einem Mailserver haben. Bei XAMPP und den meisten Anbietern von freiem Speicherplatz ist dies nicht der Fall. Manche Hoster können auch verlangen, dass Sie eine "from"-Information in den eMail-Header einfügen, die wird über einen zusätzlichen Parameter bewerkstelligt:

 mail*("sie@ihredomain.de", "Test", "Das ist eine Testmail", "From: ich@meinedomain.de");*