Formular und Login
Daten senden und auswerten

Inhalt:

1)PHP-Formulare verwenden, um Daten einzugeben

- Formulare sind klassische Methoden, um Input vom Benutzer zu erhalten.
- Formulare selbst erstellt man in HTML. Das HTML5 bietet dazu besonders gute Möglichkeiten. Hier wird das Formular gestaltet und festgelegt, auf welche Weise die Daten an den Server übermittelt und wie sie dort ausgewertet werden sollen.
- Die Weiterverarbeitung der eingegebenen Daten geht über PHP. Das Script, das die Auswertung übernehmen soll, muss im Format „.php“ abgespeichert werden.
- Dank des dort vordefinierten Arrays geht das auch sehr komfortabel

Damit ein PHP-Skript auf dem Server ausgeführt wird muss man das Formular und PHP-Skript über das action-Attribut des Formulars verknüpft werden.

Formulare werden unter Verwendung des HTML-<form>-Tags erstellt. Das <form>-Tag für jedes sinnvolle Formular sollte den Wert des action-Attributs setzen, das angibt, was der Browser mit dem Formular machen soll, wenn der Benutzer auf den Absenden-Button klickt. **Gibt man im action-Attribut einen Dateinamen an**, führt der Browser beim Absenden des Formulars stattdessen auf dem Server eine Anfrage nach dieser Ressource durch und schickt dabei die in das Formular eingetragenen Daten mit.
Hat das Skript, das die Formulardaten verarbeiten soll, den Namen form21.php, muss das <form>-Tag, das es aus dem Formular adressiert, folgende Form haben:

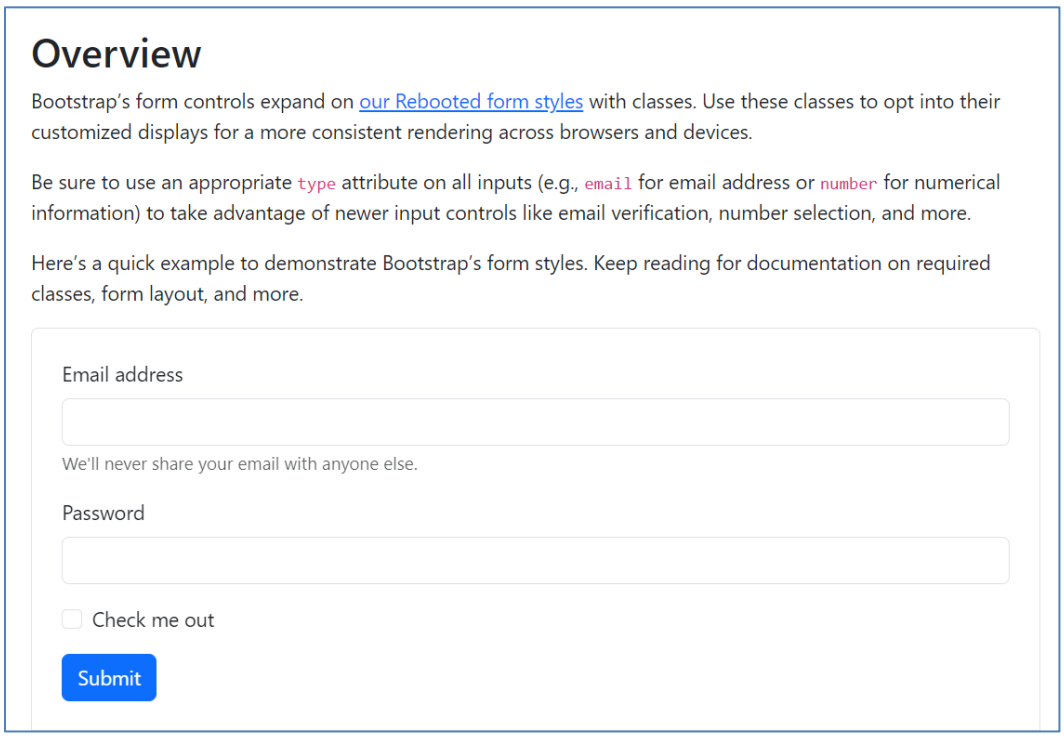
<form action = "formular2.php" method = "post">

INFO:
Innerhalb des HTML-Dokuments befindet sich ein <form>-Container. Dieser beinhaltet verschiedene Elemente: Formulare beginnen und enden mit dem Tag <form>

- Mit **action=““** werden die Daten weitergeleitet, wenn jemand auf den submit-Button drückt. Das Attribut <action> verweist auf das PHP-Auswertungsprogramm mit dem Namen „verarbeitung.php“.
- Das **Attribut <method>** verweist auf die Übermittlungsmethode <post>
- Eine **Schaltfläche zum Absenden** (englisch: submit), die beim Anklicken die eingetragenen Daten an den Server sendet und damit das genannte PHP-Auswertungsprogramm anfordert.
- Hier nicht vorhanden: Die **Schaltfläche zum Zurücksetzten** (englisch: reset) setzt bei Anklick das Formular wieder in den Anfangszustand zurück.

2)Erstelle ein neues html-Dokument „formular.php“

Nutze den Code eines typischen Formulars aus „getbootstrap.com“. Kopiere diesen und füge ihn ein.



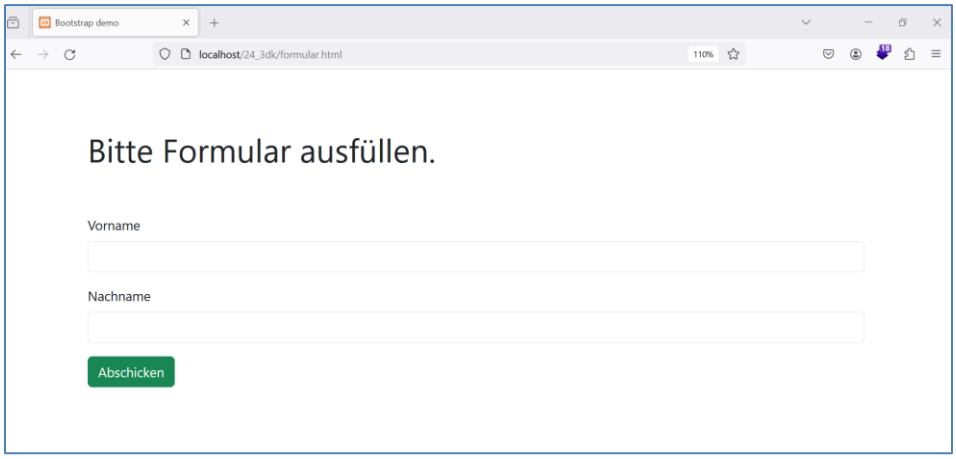
Nutze natürlich die Verbindung zur passenden CSS:

```
    <title>Login</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sh
</head>

```

Ändere den typischen Code ein wenig, damit er besser zu uns passt.

```
<body>
  <div class="container">
      <br><br><br>
  <h1>Bitte Formular ausfüllen.</h1>
      <br><br>
  <form action="formular2.php" method="post">
      <div class="mb-3">
        <label for="vn" class="form-label">Vorname</label>
        <input type="text" name="vorname" class="form-control" id="vn">
      </div>
      <div class="mb-3">
        <label for="nn" class="form-label">Nachname</label>
        <input type="text" name="nachname" class="form-control" id="nn">
      </div>

      <button type="submit" class="btn btn-success">Abschicken</button>
    </form>
  </div>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle
</body>
```

Ergebnis:



Erstelle die PHP-Datei „formular2.php“

Die Daten aus dem Formular sollen nun übernommen und ausgegeben werden.

- Zuerst werden die Daten in neue Variablen gespeichert.
- Dabei wird mit $_POST das entsprechende input-Feld aus der „form.php“-Datei angesprochen und den genau passenden Namen in die rechteckige Klammer geschrieben. Das muss genau ident sein:
  -
```
21          <input type="text" name="vn"  cla
```
  -
```
5   $vn = $_POST['vn'];
```
- In der Ausgabe werden dann die eben erstellten Variablen verwendet.

error_reporting(-1);
ini_set('display_errors','On');

functions > form2.php

```php
<?php
error_reporting(-1);
ini_set('display_errors','On');

$vn = $_POST['vn'];
$nn = $_POST['nn'];
```

Darunter kann man einen sinnvollen Satz ausgeben lassen, der sich mit den Variablen der übergebenen Daten zusammensetzt:

```php
echo "Servus $vn $nn. Schön, dass du dich meldest .<br>";
echo "<br>";
```

Ergebnis:

functions > form2.php

```php
<?php
error_reporting(-1);
ini_set('display_errors','On');

$vn = $_POST['vn'];
$nn = $_POST['nn'];

echo "Servus $vn $nn. Schön, dass du dich meldest .<br>";
echo "<br>";
```

3)Error Meldung

Zur besseren Information bei Fehlern ist es sinnvoll eine gute Error-Reporting Meldung automatisch einzubauen. Dazu sind nur ganz wenige Zeilen nötig, nämlich folgende beiden, welche gleich am Beginn in „verarbeiten.php“ nach dem Start von PHP eingefügt werden soll.

error_reporting(-1);
ini_set('display_errors','On');

```php
<?php
error_reporting(-1);
ini_set('display_errors','On');
```

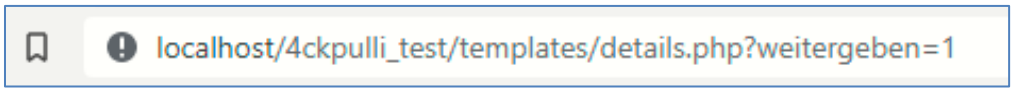
4)Erklärungen - Info

Man kann in PHP mit 2 Möglichkeiten auf Inhalte zugreifen. Dafür gibt es

- POST
- GET

Bei **GET** werden die Formulardaten über die URL übertragen. Dabei folgt hinter der URL des Skripts ein Fragezeichen, und dann der Name des Formularfeldes und nach einem = der eingetragene Wert. Das kann aber eventuell zu einem Sicherheitsproblem führen.

Beispiel:



Hier wird die Zahl „1“ übergeben.

Vorteil:

- Da es in der URL inkludiert ist, kann man damit viel machen wie z.B. als Lesezeichen verwenden, weil die Parameter mit aufgenommen sind – ist es praktisch für Suchanfragen. Google verwendet es.

- Die URL samt Parameter kann auch in der Browser-History gespeichert werden

Nachteil:

- Die Daten sind sichtbar und daher zur Versendung von Log-in Daten samt Passwort nicht geeignet.
- Die Menge der Daten, die übertragen werden können, ist begrenzt.

$_POST:

Die vordefinierte Variable „$_POST“ übernimmt die Elemente zur Auswertung aus dem HTML-Formular genau dann, wenn in der eckigen Klammer genau der Name des Formularelements angesprochen wird:

$_POST[“vorname“]

$_POST ist eine spezielle Variable, eine **sogenannte superglobale Variable**, die an jedem Punkt eines PHP-Skripts verfügbar ist.
Die superglobale Variable $_POST ist unmittelbar mit der Formularübermittlungsmethode verknüpft, die vom HTML-Formular verwendet wird. Ist das method-Attribut des Formulars auf post gesetzt, stellt PHP dem Skript die Formulardaten in der superglobalen Variablen $_POST zur Verfügung, aus der sie nach Bedarf abgerufen werden können.

Beispiel:

$_POST["name"];	die Variable „name“ erhält den Inhalt vom Attribut „name“ aus dem <input>-Tag für das Formularfeld.

$_POST ist eine besondere Art von Speicherbehälter, ein **sogenanntes Array**, das eine Sammlung von Daten unter einem einzigen Namen speichert. Die vom Server empfangenen Formulardaten werden von PHP also im Array $_POST zur Verfügung gestellt. Jedes Element im Array $_POST entspricht den Daten eines Formularfelds. Alle Formulardaten sind über das Array $_POST zugreifbar.

Die Elemente eines Arrays werden angesprochen, indem in eckigen Klammern der Index, ein Name oder eine Zahl, dieses Elements angegeben wird. Um auf die Elemente des Arrays $_POST zuzugreifen, verwendet man den Namen des entsprechenden Formularfelds, unter dem PHP die Daten in $_POST abgelegt hat.

Info: Die Übergabe an eine Variable ermöglicht eine leichtere Bearbeitung

Damit man mit den Variablen leichter weiterarbeiten kann, übergibt man lieber die durch die Methode POST ausgelesenen Elemente an eine passende Variable:

```
$vorname= $_POST["vorname"];
$nachname = $_POST["nachname"];
```

Danach kann man diese Variable ansprechen:
Beispiel:

```
echo "Guten Tag, $vorname  $nachname";
```

Ende Info

5)Radio Buttons und Auswahllisten

Radio Buttons

Aus einer Gruppe von Radiobuttons kann ein Benutzer IMMER NUR EINEN auswählen.

- Damit das funktioniert müssen die zusammengehörigen Buttons denselben NAMEN erhalten.
- Was übertragen wird steht im „value“
- Soll ein Radiobutton zu Anfang schon aktiviert sein, muss er folgendes haben: checked=“checked“

Auswahllisten

- Benötigen ein „select“ und „option“
- select muss einen Namen erhalten
- Die einzelnen Möglichkeiten werden in option-Elemente geschrieben
- Standardmäßig ist der erste Punkt der Liste vorausgewählt, soll es ein anderer sein, muss man dem geben: selected=“selected“

Beispiel Radio-Button

Erstelle eine Anrede in der Datei „form.php“, nutze dazu den Code aus „getbootstrap.com“.

Füge den Code nach den vorher schon angelegten Nachnamen ein:

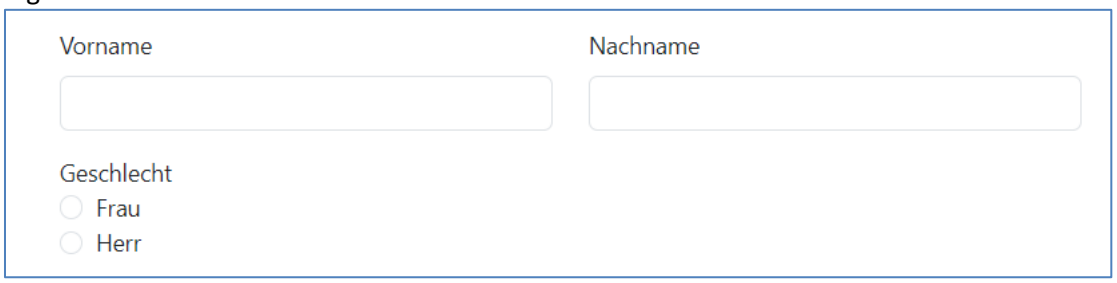
```
29  <!-- radio buttons -->
30  <div class="mb-3">  <!--margin-bottom -->
31    <label class="form-check-label">Geschlecht</label><br>
32    <input class="form-check-input" type="radio" name="sex" value="Frau" >   Frau
33    <br>
34    <input class="form-check-input" type="radio" name="sex" value="Herr" >   Herr
35  </div>
```


```
<!-- radio buttons -->
<div class="mb-3">  <!--margin-bottom -->
  <label class="form-check-label">Geschlecht</label><br>
  <input class="form-check-input" type="radio" name="sex" value="Frau" >   Frau
  <br>
  <input class="form-check-input" type="radio" name="sex" value="Herr" >   Herr
</div>
```


Info:

- Dabei ist es wichtig, dass der „name“ bei allen inputs derselbe ist, damit die Auswahl nur auf einen Radio-Button beschränkt ist. Würde der „name“ unterschiedlich lauten, könnte man mehrere Radio-Buttons auswählen und das wäre hier kontraproduktiv.
-
  steht für ein geschütztes Leerzeichen, damit man einen Abstand machen kann

Ergebnis:



In der Datei „form2.php“ muss nun dieser Value übernommen werden. Dann kann er auch ausgeben werden:

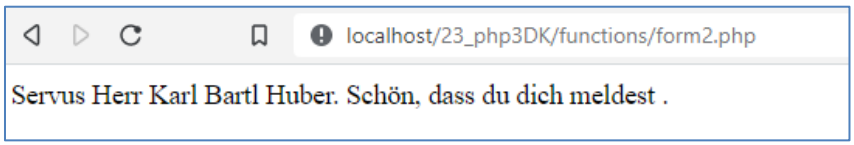
Zuerst die neue Variable abhängig von $_POST und dann im „echo“:

```
5   $vn = $_POST['vn'];
6   $nn = $_POST['nn'];
7   $sex = $_POST['sex'];
```

Echo-Ausgabe:

```
echo "Servus $sex $vn $nn. Schön, dass du dich meldest .<br>";
echo "<br>";
```

Ergebnis:



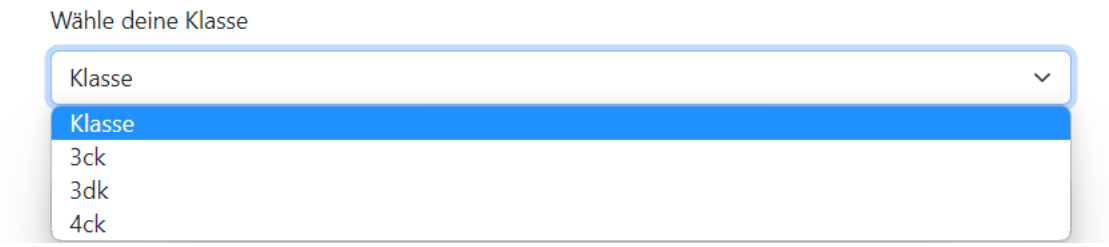
<u>Beispiel Auswahllisten:</u>

Ändere in der Datei „form.php“ auch noch dahingehend, dass eine Auswahlliste eingebaut wird:

In welche Klasse gehst du?

```
36  <!-- Auswahlliste - select und options -->
37  <div class="mb-3">
38  <label class="form-label">Wähle deine Klasse</label>
39  <select class="form-select" name="klasse">
40    <option selected>Klasse</option>
41    <option value="3ck">3ck</option>
42    <option value="3dk">3dk</option>
43    <option value="4ck">4ck</option>
44  </select>
45  </div>
```

Ergebnis mit 2
 als Abstand:

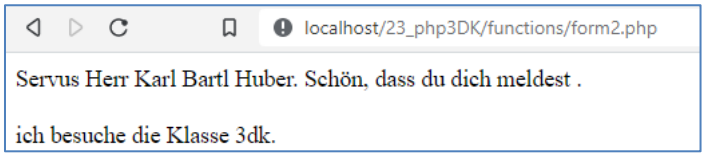


Auswertung in „form2.php“ mit Aufnahme der Variablen und dann als Ausgabe in „echo“:

```
$sex = $_POST['sex'];
$meineKlasse = $_POST['klasse'];
```

```
echo "Servus $sex $vn $nn. Schön, dass du dich meldest .<br>";
echo "<br>";
echo "ich besuche die Klasse $meineKlasse. ";
echo "<br>";
```

Ergebnis:



Aufgabe:

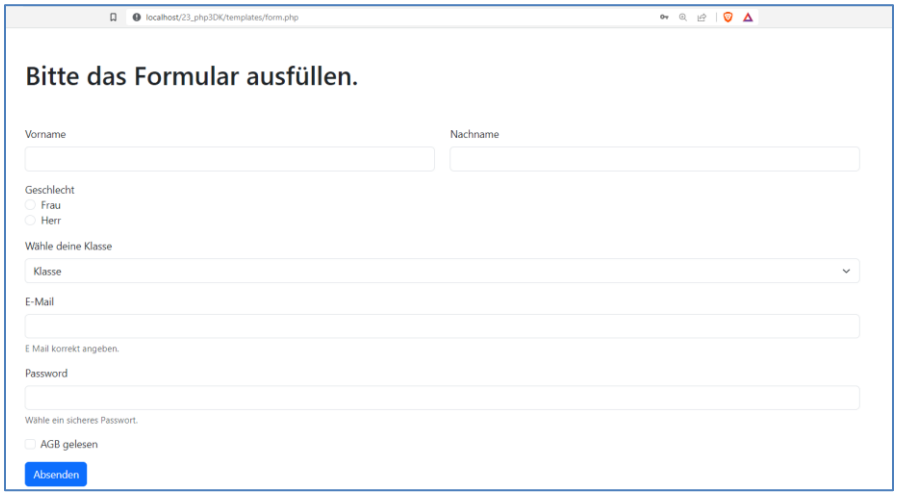
Füge noch die Felder

- „E-Mail“ und
- „Passwort“ und
- AGB gelesen darunter hinzu.

Damit können wir später dieses Dokument in „registrieren“ umbenennen.

```
<div class="mb-3">
  <label for="em" class="form-label">E-Mail</label>
  <input type="email" name="email"  class="form-control" id="em">
  <div id="emailHelp" class="form-text">E Mail korrekt angeben.</div>
</div>
<div class="mb-3">
  <label for="pw" class="form-label">Password</label>
  <input type="password" name="pwd"  class="form-control" id="pw">
  <div id="emailHelp" class="form-text"> Wähle ein sicheres Passwort.</div>
</div>
<div class="mb-3 form-check">
  <input type="checkbox" class="form-check-input" id="exampleCheck1">
  <label class="form-check-label" for="exampleCheck1">AGB gelesen</label>
</div>
<button type="submit" class="btn btn-primary">Absenden</button>
</form>
</div> <!-- ende container -->
```

Ergebnis:

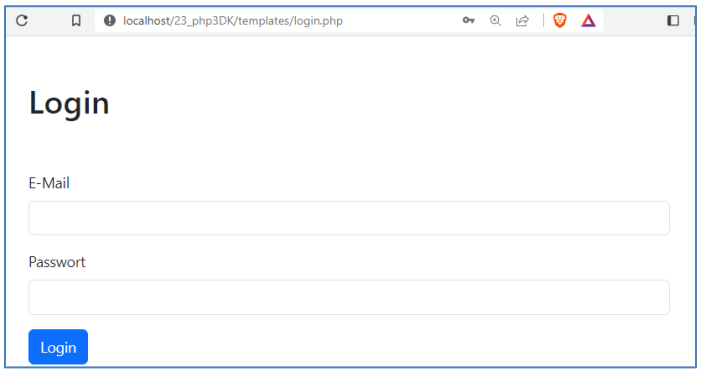


6)Login - mit if-else die Eingabe überprüfen

6a)Erstelle eine login-Datei.

In der „login.php“ wird die Eingabe des Kunden übernommen und mit festgelegten Daten verglichen. In der entsprechenden Funktion müssen beide Felder damit übereinstimmen, da sie mit einem UND (&&) verknüpft sind, damit man auf die „geheime“ Seite weitergeleitet wird. Ansonsten wird eine Meldung „Leider ist der Zugang falsch“ ausgegeben.

Erstelle im selben templates-Ordner eine login.php, in der du eventuell einiges aus der „form.php“ übernimmst und kopierst oder es mit Hilfe von getbootstrap.com neu aufbaust.




```
16  <form action="../functions/login2.php" method="post">
17  <div class="row">
18      <div class="mb-3 col">  <!--mb für margin-bottom als Abstand -->
19          <label for="em" class="form-label">E-Mail</label>
20          <input type="email" name="email"  class="form-control" id="em">
21      </div>
22  </div>
23  <div class="row">
24      <div class="mb-3 col">
25          <label for="pw" class="form-label">Passwort</label>
26          <input type="password" name="passwort"  class="form-control" id="pw">
27      </div>
28  </div>  <!-- ende row -->
29
30  <button type="submit" class="btn btn-primary">Login</button>
31  </form>
```

Da wir noch keine Verbindung zu einer Tabelle in einer Datenbank haben, müssen wir die Eingaben für die E-Mail und das Passwort vorgeben. Um es nicht zu vergessen kann man es ausnahmsweise auf der login.php sogar anführen:

```
31  </form>
32
33  <br><br>
34  E-Mail: andi@hak.it
35  PSW: 1234
36
37  </div> <!-- ende container -->
38  </body>
39  </html>
```

6b)login2.php in functions erstellen

Dann erstelle im Ordner „functions“ die dazu passende „login2.php“. Diese soll die beiden Werte der Inputfelder übernehmen, in eine Variable speichern und dann in einer IF-Funktion überprüfen:

login.php ● login2.php ×

functions > login2.php

```
1  <?php
2  error_reporting(-1);
3  ini_set('display_errors','On');
4
5  $email = $_POST['email'];
6  $pwd = $_POST['passwort'];
```

Die IF soll den Weg zur Datei (hier index.php“) freigeben, wenn die beiden Felder korrekt ausgefüllt werden. Hier verwenden wir eine UND Verknüpfung, d.h. dass beide Felder passen müssen:

```
7
8  if($email == "andi@hak.it" && $pwd == "1234")
9  {}
```

Ist die IF erfolgreich, soll mit Hilfe der PHP-Weiterleitung die „index.php“ aufgerufen werden. Ansonsten soll ausgegeben werden: „Zugang verweigert“

```
8  if($email == "andi@hak.it" && $pwd == "1234")
9  {
10     header("Location: ../index.php");
11     exit;
12 } else
13 {
14     echo "Zugang verweigert.";
15 }
```

Dabei wird der Zugang mit festgelegtem E-Mail und Passwort direkt in den Code geschrieben:

```
$email = $_POST['email'];
$pwd = $_POST['passwort'];

if($email == "andi@hak.it" && $pwd == "1234")
{
   header("Location: ../index.php");
   exit;
} else
{
   echo "Zugang verweigert.";
```