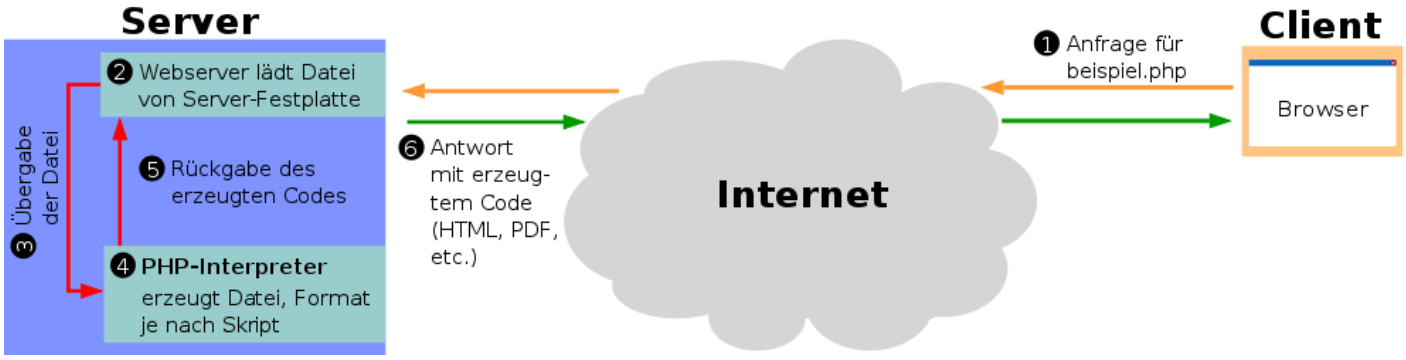
Universität zu Köln
Historisch-Kulturwissenschaftliche Informationsverarbeitung
IT-Zertifikat der Phil.-Fak.: Advanced Web Basics
Dr. Jan G. Wieners - jan.wieners@uni-koeln.de

IT-Zertifikat: Advanced Web Basics – PHP

PHP („PHP: Hypertext Preprocessor“) ist eine serverseitige Skriptsprache: Der PHP-Code wird nicht – wie bei JavaScript – auf dem Clientrechner ausgeführt, sondern auf dem Server.



Bildquelle: http://commons.wikimedia.org/wiki/File:PHP_funktionsweise.svg

Die Kommunikation von Webserver und Webclient erfolgt bei PHP in mehreren Schritten:

1. Der Clientrechner fordert über den Browser (Eingabe einer URL) eine Website über das Internet von einem Server an.
2. Die angeforderte Datei hat die Dateiendung .php – der Server interpretiert die angeforderte Datei folglich als ein PHP-Skript und leitet die Seite nicht direkt weiter an den Client, sondern an den PHP-Interpreter.
3. Der PHP-Interpreter interpretiert die PHP-Befehle und erzeugt daraus eine neue HTML-Seite.
4. Die vom Interpreter generierte Datei wird vom Server über das Internet an den Client gesendet. Zwar hat die im Browser angezeigte Seite noch die Dateiendung .php, sie enthält jedoch keinen PHP-Code mehr.

➔ Der entscheidende Unterschied zu statischen Seiten (i.e. eine HTML-Datei auf der Festplatte) besteht bei **dynamischen Webseiten** darin, dass die Seite erst vom Server generiert wird.

Grundlagen; PHP in HTML-Dokumente einbinden

PHP-Code wird direkt in HTML-Dokumente eingebunden. Um dem PHP-Interpreter verständlich zu machen, dass Teile der HTML-Datei PHP-Befehle sind, wird PHP-Code innerhalb von <?php und ?> notiert:

```
<html><head><title>PHP-Website</title><head>
  <body>
       <?php
            print 'Mein erstes PHP-Dokument';
       ?>
  </body>
</html>
```

Achtung: Die Dateiendung muss auf .**php** gesetzt sein, z.B. meineWebsite.php.

PHP-Befehle lassen sich an beliebigen Stellen des HTML-Dokumentes einbinden, nicht nur im <body> des HTML-Codes:

```
<html><head><title><?php echo date('j.n.Y'); ?></title><head>
  <body><?php print 'Mein erstes PHP-Dokument'; ?></body>
</html>
```

So gibt die Zeile <?php echo date('j.n.Y'); ?> im <title>-Bereich des HTML-Dokumentes als Titel das aktuelle Datum aus.

PHP und HTML-Tags

Wir können PHP-Befehle dazu verwenden, um (u.a.) HTML- und CSS-Code auszugeben:

```
<html><head><title>PHP-Website</title><head>
  <style type="text/css">
       body { background-color: <?php print 'yellow'; ?>; }
  </style>
  <body><?php print '<h1>Mein erstes PHP-Dokument</h1>';?></body></html>
```

Kommentare

```
// dies ist ein einzeiliger Kommentar

/* dies ist
ein mehrzeiliger
Kommentar
*/
```

Variablen

- Variablen werden in PHP gekennzeichnet durch das Dollarzeichen: **$**variablenName;
- PHP unterscheidet Groß- und Kleinschreibung von Variablen. So ist die Variable $meineVariable eine andere Variable als $Meinevariable.
- Auf das Dollarzeichen darf nicht unmittelbar eine Zahl folgen: $23Variable ist kein korrekter Name für die Variable.
- Leerzeichen, Punkte, Ausrufezeichen oder Bindestriche sind in Variablennamen nicht gestattet.
- Um einer Variable einen Wert zuzuweisen, wird der Zuweisungsoperator verwendet
  $meineVariable = 23;
  $summe = 23 + 15;
- Der Inhalt von Variablen lässt sich über die PHP-Funktion print ausgeben:
  print $summe;
  print "Eine Zeichenkette";
- Textinhalt lässt sich mit dem Inhalt von Variablen kombinieren (Variableninterpolation):

```
<?php

  $vorname = "Max";
  $nachname = "Mustermann";

  print "Vorname: $vorname, Nachname: $nachname";

?>
```

Zeichenketten, Sonderzeichen, Anführungszeichen

Soll innerhalb von doppelten Anführungszeichen ein Dollarzeichen ausgegeben werden, so muss das Dollarzeichen über das Zeichen „****“ „maskiert“ werden:

```
print "Der Preis beträgt 25 \$";
```

Auch doppelte Anführungszeichen müssen maskiert werden:

```
print "<img src=\"meinbild.jpg\" alt=\"Bild\">";
```

Anstatt der doppelten Anführungszeichen empfiehlt sich hier die Verwendung einfacher Anführungszeichen:

```
print 'Der Preis beträgt 25 $';
print '<img src="meinbild.jpg" alt="Bild">';
```

➔ Empfehlung: Einfache Anführungszeichen bei HTML-Tags (z.B. bei Attributwerten) verwenden, ansonsten doppelte Anführungszeichen.

Zeichenketten Verknüpfen

Um Zeichenketten zu verknüpfen wird der Verknüpfungsoperator (“**.**“) verwendet:

```
<?php

  print "Hello" . "World";

  $variable1 = "Hello";
  $variable2 = "World";
  $variable3 = $variable1 . " " . $variable2;

  print "$variable1 verknüpft mit $variable2 ergibt $variable3";

?>
```

Operatoren

Operator	Operation	Beispiel
+	Addition	$i = 13 + 25;
-	Subtraktion	$i = 13 – 25;
*****	Multiplikation	$i = 13 * 25;
/	Division	$i = 13 / 25;

Arrays

Um ein Array zu erstellen, wird die Funktion array() verwendet:

Leeres Array: $meinArray = **array()**;
Array initialisieren: $meinArray = **array(**"15", 15, "Hello"**)**;

Elemente an das Array hängen über "[]":
$meinArray**[ ]** = „World";
Hierbei wird die Zeichenkette „World" an das Array gehangen. Der Zugriff auf die Elemente des Arrays gestaltet sich analog JavaScript durch Angabe der Speicherzelle:
print $meinArray**[0]**;

Elemente des Arrays ausgeben

Zu Testzwecken lassen sich alle Elemente des Arrays über die PHP-Funktion print_r( ) ausgeben. So gibt

```
print_r($meinArray);
```

die Information `Array ( [0] => 15 [1] => 15 [2] => Hello )` aus.

Arrays mit foreach( ) durchlaufen

Bei der PHP-Funktion foreach werden nacheinander die einzelnen Elemente des Arays durchlaufen:

```
<?php

   $meinArray = array("Hello", "World");
   $elementCounter = 0;

   foreach($meinArray as $einzelnesElement)
   {
   print "meinArray an Position $elementCounter => $einzelnesElement<br>";
   $elementCounter++;
   }

?>
```

Assoziative Arrays

Wir können Schlüssel-/Wertepaare definieren:

```
$farben = array("rot" => "#ff0000",
                "gruen" => "#00FF00",
                "blau" => "0000FF");

  $farben["schwarz"] = "#000000";
  $farben["Weiss"] = "#ffffff";
```

Hierbei wird in der ersten Zeile der Arrayschlüssel „rot" angelegt und dem Schlüssel der Wert „#ff0000" zugewiesen.

Die Ausgabe des Wertes eines Schlüssels erfolgt über die print-Funktion:

```
  print $farben["gruen"];
```

Über foreach lassen sich alle Werte-/Schlüsselpaare ausgeben:

```
  foreach ($farben as $key => $value)
  {
       print "Schluessel: $key, Wert: $value<br>";
  }
```

PHP-Quellcode auslagern und einbinden

Selbstdefinierte Funktionen lassen sich in externe PHP-Dateien auslagern und über die Funktion **include** einbinden:

```
<?php

  include 'externeDatei.php';

?>
```

Anweisungen: if, elseif, while, for

if, elseif und else: Über die if-Anweisung lassen sich Entscheidungen treffen bzw. Anweisungen je nach gegebenen Umständen auszuführen:

```
<?php
  if(Ausdruck)
  {
       Anweisung
  }
  elseif(Ausdruck)
  {
       Anweisung
  }
  else
  {
       Anweisung
  }
?>
```

while: Mit der while-Anweisung lassen sich wiederholt Aktionen ausführen, solange eine Bedingung zutrifft:

```
while(Ausdruck)
{
  Anweisung
}
```

for

```
for(Initialisierung; Bedingung; Veränderung der Variablen)
  {
       Anweisung
  }
```

Beispiel:

```
for($i = 0; $i < 5; $i++)
{
  print "$i <br>";
}
```

Vergleichsoperatoren

Operator	Operation	Beispiel
==	Auf Gleichheit prüfen	variable1 == variable2
!=	Auf Ungleichheit prüfen	variable1 != variable2
&&	Logisches UND: Zwei Bedingungen prüfen	if((a==b) && (b==c))
||	Logisches ODER	if((a==b) || (b==c))
<	kleiner	if(a < b)
<=	kleiner gleich	if(a <= b)
>	größer	if(a > b)
>=	größer gleich	if(a >= b)

Funktionen definieren

```
function stringAusgeben($ausgabeString)
{
  print $ausgabeString;
}
```

Aufruf:

```
stringAusgeben("Hello World");
```