

PHP: Einführung

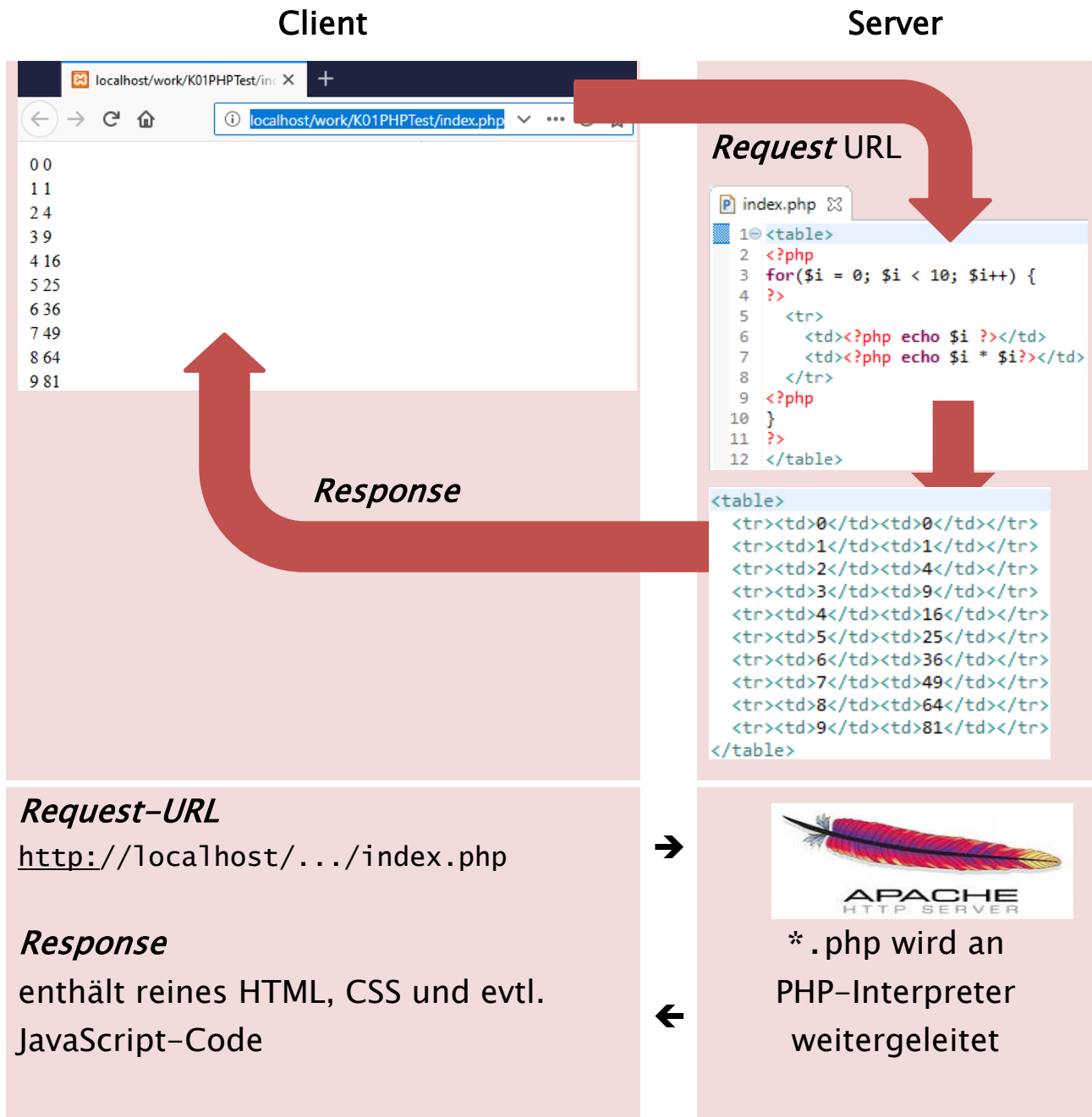
- Verstehen wie der Apache Web-Server PHP-Seiten verarbeitet
- Die Besonderheiten von PHP-Variablen begreifen
- Mit den verschiedenen Datentypen von PHP umgehen können
- Mit numerischen und assoziativen Arrays umgehen können
- Verstehen was Superglobals sind und diese einsetzen können
- Einfache Formulare erstellen können
- Externe Dateien einbinden können
- Das Problem der Mehrfacheinbindung beherrschen können
- Mit Cookies arbeiten können

Literaturhinweis

PHP 7 und MySQL, das umfassende Handbuch, Christian Wenz, Tobias Hauser, Rheinwerk Verlag, ISBN 978-3-8362-4082-6

PHP (ursprünglich Personal Home Page) steht für PHP: Hypertext Preprocessor

„PHP wurde Anfang 2013 auf etwa 244 Millionen Websites eingesetzt und Anfang 2019 von 79 % aller Websites als serverseitige Programmiersprache verwendet. PHP ist die am häufigsten serverseitig verwendete Programmiersprache zum Erstellen von Websites. Zudem ist sie bei den meisten Webhostern vorinstalliert.“¹



WICHTIG: JavaScript wird am Browser ausgeführt

¹ Quelle: Wikipedia

Variablen

```
<?php
$i = 3;
$j = "10 Stück";
echo $i + $j;           // Ausgabe von 13 mit Notice2
echo $i . $j;           // Ausgabe von 310 Stück
echo "$i Stück";        // Ausgabe von 3 Stück

$j = "3";
if ($i == $j)            // Ausgabe von "Inhalte gleich"
    echo "Inhalte gleich";
if ($i === $j)           // Keine Ausgabe
    echo "Auch Typen gleich";

$y = 3.54;
if (is_double($y))
    echo "\$y " . gettype($y); // Ausgabe von "$y double"
echo gettype($_GET["z"]);      // Ausgabe von string3
echo (int)$y;                  // Ausgabe von 3
$k = &$i;                      // Referenzvariable
$k = 7;
echo $i;                       // Ausgabe von 7

$varvar = "i";
echo $$varvar;                 // Ausgabe von 7

$varmeth = "mb_strlen"4;
echo $varmeth($$varvar . " Stück"); // Ausgabe von 7

const KONSTANTE = "wert";
echo KONSTANTE;
?>
```

- Typ kann sich während der Ausführung des Skripts ändern
- Groß- und Kleinschreibung wird berücksichtigt
- === kontrolliert auch ob Typen gleich sind
- GET- und POST-Parameter standardmäßig als Strings übergeben
- & ermöglicht pass by reference (bei Methodenparametern)
- **Variable Variablennamen** und **variable Methodennamen**
- Vielzahl von is_-Methoden zur Typüberprüfung vorhanden
- Konstanten gelten im gesamten Skript

² In Datei php.ini durch error_reporting=E_ALL & ~E_NOTICE ausschaltbar

³ URL könnte beispielsweise lauten: http://localhost/.../index.php?z=3

⁴ mb_strlen ermittelt die Zeichenlänge eines Unicode-Strings (Standard-Zeichensatz für PHP), mit strlen wäre nachfolgender String wegen Umlaut um ein Zeichen länger. Beachte auch mb_strtolower()

Datentypen

Zeichenketten	string	'"Emil\'s" \\Auto'
Ganzzahlen	integer/int	
Fließkommazahlen	float/double/real ⁵	-17.4e-2
Boolesche Werte	boolean/bool	true, false
Arrays	array	[2, 3, 4]
Objekte	object	
Ressourcen	resource	Referenz auf externe Datenquellen (Datenbanken, Dateien)
Null	null	Typ für Variable ohne Wert

`isset()` liefert true, falls Variable existiert

`empty()` liefert true, falls Variable leer ist, "" und 0 liefern true

`unset()` löscht eine Variable

	<code>gettype()</code>	<code>isset()</code>	<code>empty()</code>	<code>if(\$x)...</code>
<code>\$x=""</code> ;	string	true	true	false
<code>\$x=null</code> ;	null	false	true	false
<code>\$x</code> undefiniert	null	false	true	false
<code>\$x=false</code> ;	boolean	true	true	false
<code>\$x=true</code> ;	boolean	true	false	true
<code>\$x=0</code> ;	integer	true	true	false
<code>\$x=1</code> ;	integer	true	false	true
<code>\$x="0"</code> ;	string	true	true	false
<code>\$x="true"</code> ;	string	true	false	true
<code>\$x="false"</code> ;	string	true	false	true

⁵ Synonyme für ein und denselben Datentyp

Numerische Arrays

```
<pre>
<?php
$numArray = array();
$numArray[3] = 3;
$numArray[2] = 3.15;
$numArray[] = "Hallo";
var_dump($numArray);

foreach ($numArray as $key => $value) // Achtung Eingabereihenfolge
    echo "$key => $value<br>";          // ist entscheidend!!!

$numArray = [2, 3.3, "Leute"];
var_dump($numArray);

$numArray = null;
$numArray[7] = "wie geht`s";
$numArray[] = array("7", 9.0, "dir");
$numArray[8][1] = null;                // löscht nur Inhalt an Stelle 1
unset($numArray[8][1]);                // löscht gesamten Eintrag mit Index 1
var_dump($numArray);

echo count($numArray);
echo $numArray[100];                  // wirft Notice

$a = [1, 2, 3];
$b = $a;
$b[0] = 10;
echo $a[0];                          // Ausgabe von 1
?>
</pre>
```

- Die Indizierung beginnt bei 0
- Arrays werden bei Zuweisungen kopiert!
- Es existieren viele Funktionen um Arrays zu manipulieren siehe www.php.net/docs.php

```
array(3) {
  [3]=>
  int(3)
  [2]=>
  float(3.15)
  [4]=>
  string(5) "Hallo"
}
3 => 3
2 => 3.15
4 => Hallo
array(3) {
  [0]=>
  int(2)
  [1]=>
  float(3.3)
  [2]=>
  string(5) "Leute"
}
array(2) {
  [7]=>
  string(10) "wie geht`s"
  [8]=>
  array(2) {
    [0]=>
    string(1) "7"
    [2]=>
    string(3) "dir"
  }
}
2
```

Notice: Undefined offset

Assoziative Arrays

```
<?php
$array = array("Bozen" => 10000, "Meran" => 40000,
    "Brixen" => 19000, 6 => 25000);
$array["Sterzing"] = 5700;
unset($array[6]);
var_dump($array);

ksort($array);
foreach ($array as $key => $value)
    echo "$key ... $value";
foreach ($array as $value)
    echo "$value ";

$array = ["heiß" => 30, "angenehm" => 20, "kalt" => 10];
foreach($array as $key => &$value) // Achtung &
    $value+=5;
unset($value)6;
var_dump($array);
?>
```

`$value` wird normalerweise als Wert-Parameter übergeben. Soll er geändert werden, dann muss er als Referenz übergeben werden

WICHTIG: Nach Verwendung von `$value` muss Variable gelöscht werden (siehe Fußnote)

```
array(4) {
    ["Bozen"]=>
    int(10000)
    ["Meran"]=>
    int(40000)
    ["Brixen"]=>
    int(19000)
    ["Sterzing"]=>
    int(5700)
}
Bozen ... 10000
Brixen ... 19000
Meran ... 40000
Sterzing ... 5700
10000 19000 40000 5700
array(3) {
    ["heiß"]=>
    int(35)
    ["angenehm"]=>
    int(25)
    ["kalt"]=>
    int(15)
}
```

⁶ Siehe <https://stackoverflow.com/questions/3307409/php-pass-by-reference-in-foreach>

Superglobale Arrays

\$_GET	Enthält die über GET aus einem Formular an die URL angehängten Werte (siehe hinten)
\$_POST	Enthält die über POST von einem Formular gesendeten Werte
\$_COOKIE	Enthält die gesetzten Cookies
\$_REQUEST	Enthält \$_GET, \$_POST (, \$_COOKIE je nach Konfiguration)
\$_SESSION	Liefert Daten zur Benutzersitzung
\$_SERVER	Enthält Infos über PHP-Installation und Server HTTP_ACCEPT_LANGUAGE de PHP_SELF /work/K01PHPTTest/superglobals.php PHP_AUTH_USER und PHP_AUTH_PW
\$_ENV	Enthält Informationen über die Umgebung in der PHP läuft. Enthält Umgebungsvariablen
\$_FILES	Enthält hochgeladene Dateien

```

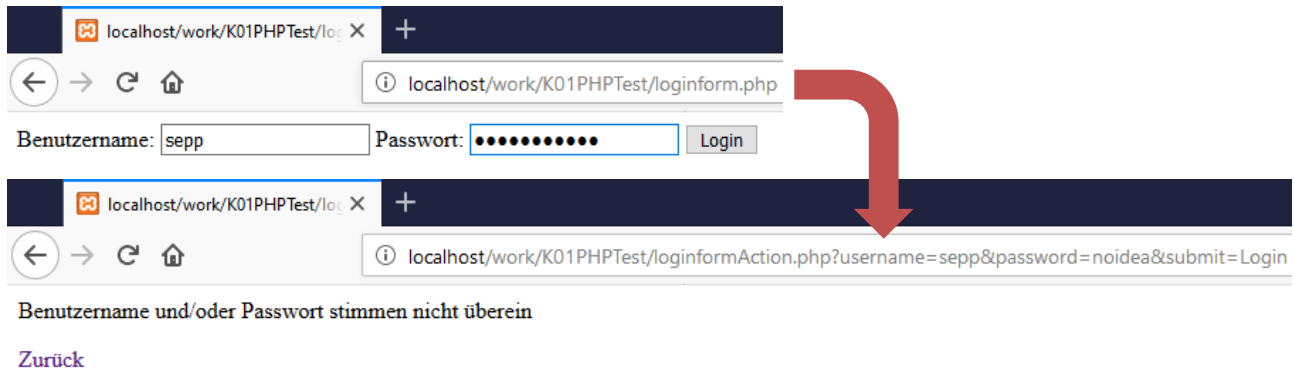
<table><tr><th>Name</th><th>Wert</th></tr>
<?php
ksort($_SERVER);
foreach ($_SERVER as $key => $value) {
    printf("<tr><td>%s</td><td>%s</td></tr>", $key,
        is_array($value) ? implode(" ", $value) : $value);
}
?>
</table>

```

Name	Wert
COMSPEC	C:\WINDOWS\system32\cmd.exe
CONTEXT_DOCUMENT_ROOT	/Benutzer/OEM/Work
CONTEXT_PREFIX	/work
DOCUMENT_ROOT	D:/Benutzer/OEM/Info/Progs/xampp/htdocs
GATEWAY_INTERFACE	CGI/1.1
HTTP_ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apn
HTTP_ACCEPT_ENCODING	gzip, deflate, br
HTTP_ACCEPT_LANGUAGE	de-DE,de;q=0.9,en-US;q=0.8,en;q=0.7
HTTP_CONNECTION	keep-alive
HTTP_HOST	localhost
HTTP_SEC_FETCH_MODE	navigate
HTTP_SEC_FETCH_SITE	none
HTTP_SEC_FETCH_USER	?1
HTTP_UPGRADE_INSECURE_REQUESTS	1
HTTP_USER_AGENT	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, l
MIBDIRS	/Benutzer/OEM/Info/Progs/xampp/php/extras/mibs
MYSQL_HOME	\xampp\mysql\bin

Mit einfachen Formularen arbeiten

```
<form method="get" action="loginformAction.php">
  <label>Benutzername:</label>
  <input type="text" name="username">
  <label>Passwort:</label>
  <input type="password" name="password">
  <input type="submit" name="submit" value="Login">
</form>
```



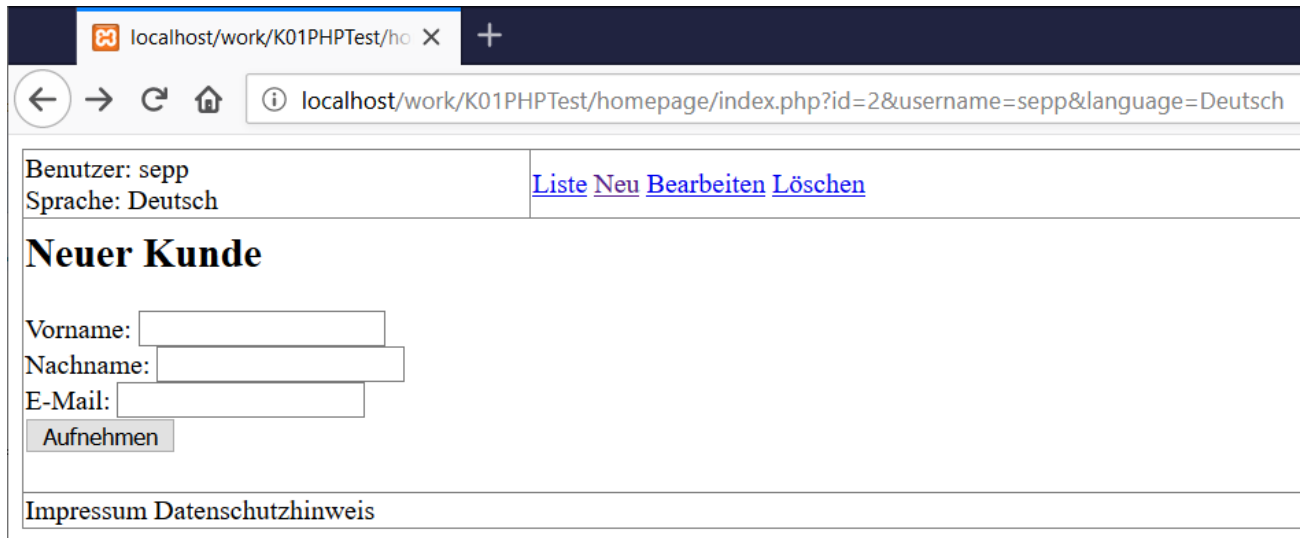
```
<?php
if ($_GET["username"]=="sepp" && $_GET["password"]=="verycomplex")
  header("Location:index.php");
else {
  echo "<p>Benutzername und/oder Passwort nicht korrekt</p>";
  echo "<p><a href='loginform.php'>Zurück</a></p>";
}
?>
```

- loginform.php → action → loginformAction.php
- Bei method="get" Request-Parameter in URL codiert
Alternative method="post"
- HTML bietet eine Vielzahl von Eingabefeldern
- **HINWEIS:** Beim Versand sensibler Daten ist in der Praxis immer HTTPS einzusetzen!

Weiterleitung

- header() muss vor jeglicher HTML-Ausgabe passieren
- In Adressleiste des Browsers erscheint URL der aufgerufenen Seite
- **ACHTUNG:** Skript wird weiter abgearbeitet. Evtl. mit exit() verlassen

Einbinden externer Dateien



localhost/work/K01PHPTest/homepage/index.php?id=2&username=sepp&language=Deutsch

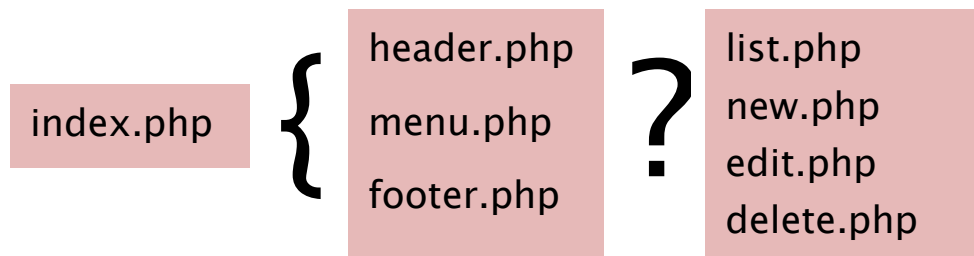
Benutzer: sepp
Sprache: Deutsch

[Liste](#) [Neu](#) [Bearbeiten](#) [Löschen](#)

Neuer Kunde

Vorname:
Nachname:
E-Mail:

[Impressum Datenschutzhinweis](#)



`include()` und `require()`

Genau an der Stelle des Aufrufes wird das einzubindende Skript in das aufrufende Skript **kopiert** und ausgeführt

Ist einzubindendes Skript nicht vorhanden produziert...

`include()` eine *Warnung*, aufrufendes Skript wird fortgesetzt

`require()` eine *Fehlermeldung*, aufrufendes Skript wird abgebrochen

index.php

```
<table>
  <tr>
    <td><?php include "header.php"?></td>
    <td><?php include "menu.php"?></td>
  </tr>
  <tr height="100%">
    <td colspan="2">
      <?php
        switch ($_GET["id"]) {
          case 1: {
            $file = "list.php";
            break;
          }
          case 2: {
            $file = "new.php";
            break;
          }
          ...
        }
        include $file;
      ?>
    </td>
  </tr>
  <tr>
    <td colspan="2">
      <?php include "footer.php"?>
    </td>
  </tr>
</table>
```

header.php

```
Benutzer: <?php echo $_GET["username"]?><br>
Sprache: <?php echo $_GET["language"]?>
```

menu.php

```
<a href="index.php?id=1">Liste</a>
<a href="index.php?id=2">Neu</a>
<a href="index.php?id=3">Bearbeiten</a>
<a href="index.php?id=4">Löschen</a>
```

new.php

```
<h2>Neuer Kunde</h2>
<form method="post" action="newAction.php">
  <label>Vorname:</label>
  <input type="text" name="firstname"><br>
  <label>Nachname:</label>
  <input type="text" name="surname"><br>
  <label>E-Mail:</label>
  <input type="email" name="email"><br>
  <input type="submit" name="submit"
    value="Aufnehmen">
</form>
```

newAction.php

```
<?php
if (empty($_POST["firstname"]) ||
    empty($_POST["surname"]) || empty($_POST["email"])){
  ?>
  <h2>Fehler</h2>
  <p>Vor-, Nachname und E-Mail müssen ... </p>
  <p><a href="index.php?id=2">Zurück</a>
  <?php
  } else {
    header("Location:index.php");
  }
  ?>
```

PROBLEM: Mehrfacheinbindung verursacht Fehlermeldung mit Abbruch

LÖSUNG: `include_once()` oder `require_once()`

`functions.php`

```
<?php
function circumference($r) {
    return 2 * $r * pi();
}
function surface($r) {
    return pow($r, 2) * pi();
}
?>
```

`circle.php`

```
<?php
require("functions.php");
echo $_SERVER["PHP_SELF"];
echo "Umfang Kreis: " .
    circumference(3.1);
?>
```

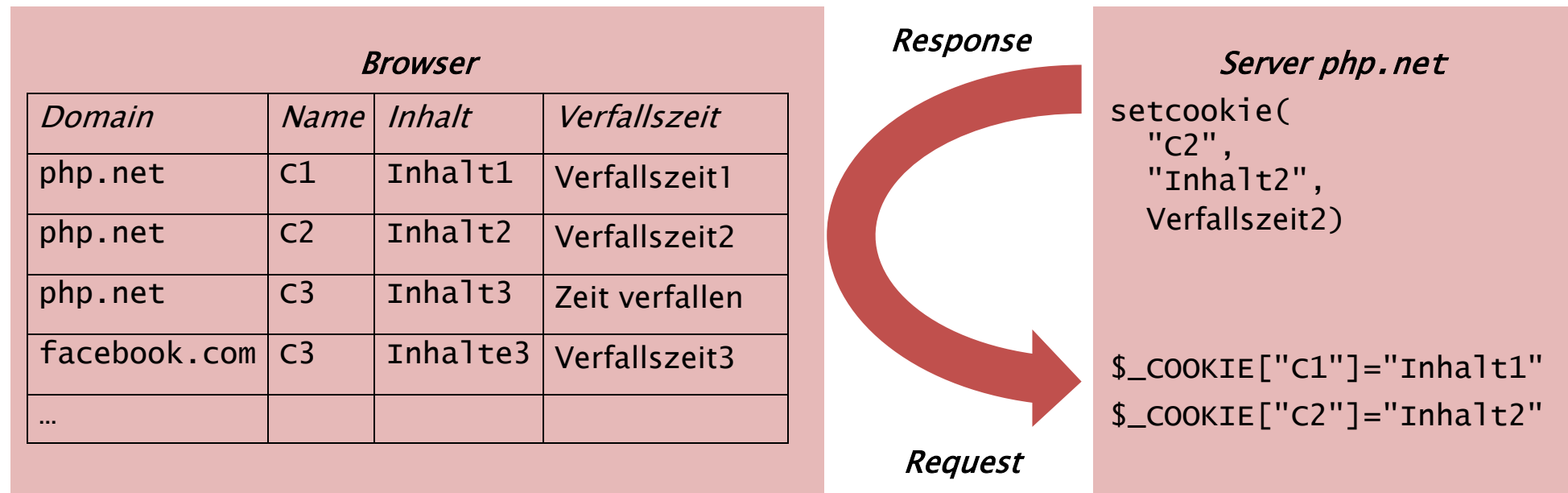
`hemisphere.php`

```
<?php
require_once "circle.php";
require_once "functions.php";
function volume($r) {
    return 2/3*pow($r, 3)* pi();
}
echo "Volumen Halbkugel: " .
    volume(3.1);
echo "Umfang Halbkugel: " .
    circumference(3.1);
?>
```

`/work/K01PHPTest/hemisphere.php`

Umfang Kreis: 19.477874452257
Volumen Halbkugel: 62.394124495396
Umfang Halbkugel: 19.477874452257

Cookies



```
setcookie(
    "BesuchteKategorien",
    "wetter Lokales Inserate Chronik webcams",
    time() + 60 * 60 * 24
);
```

- Ohne Zeitangabe → temporäres Cookie
- Zeit in Vergangenheit → Cookie wird gelöscht

- Cookies werden im Http-Header verschickt
ACHTUNG: vor setCookie() keine HTML-Ausgaben
- Cookies werden mit jeder übermittelten Datei übertragen
- **PROBLEM:** Cookies zur Aktivitätenverfolgung von Drittanbietern

