

Vorlesung



University of Applied Sciences

DBSP

Unit PHP I

Programmierparadigma

Grundlegende Elemente eines PHP Programms (I)

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

1



University of Applied Sciences



Prof. Dr. rer. nat. Nane Kratzke

*Praktische Informatik und
betriebliche Informationssysteme*

- Raum: 17-0.10
- Tel.: 0451 300 5549
- Email: kratzke@fh-luebeck.de



@NaneKratzke

Updates der Handouts auch über Twitter #dbsp

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

2

Übergreifende Ziele der Lehrveranstaltung



University of Applied Sciences

Client- und Serverseitige Entwicklung

PHP (Serverseitig)

JavaScript (Clientseitig)

„Hosten“ von Apps

Framework Erfahrungen

CMS (Drupal)

WebServices (Google-Maps)

jQuery

Datenbank-Integration

Berücksichtigung von Sicherheitsaspekten

HTML-Injections

SQL-Injections

Session Hijacking

Login-Systeme

Um sich weitere Web-Technologien autodidaktisch erarbeiten zu können.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

3

Units



University of Applied Sciences

Unit 1
Cloud Computing
IaaS

Unit 2
CMS Drupal

Unit 3
HTML und CSS

Unit 4 - 7
PHP I - IV

Unit 8
Sessions, Cookies,
Formulare und
Login-System

Unit 9
JavaScript

Unit 10
Drupal Module
Development

Unit 11
Datenmodellierung

Unit 12 - 13
Datenbanken und SQL
Vom Datenmodell zur
Datenbank

Unit 14
Datenbank-gestützte
Web-Anwendungen

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

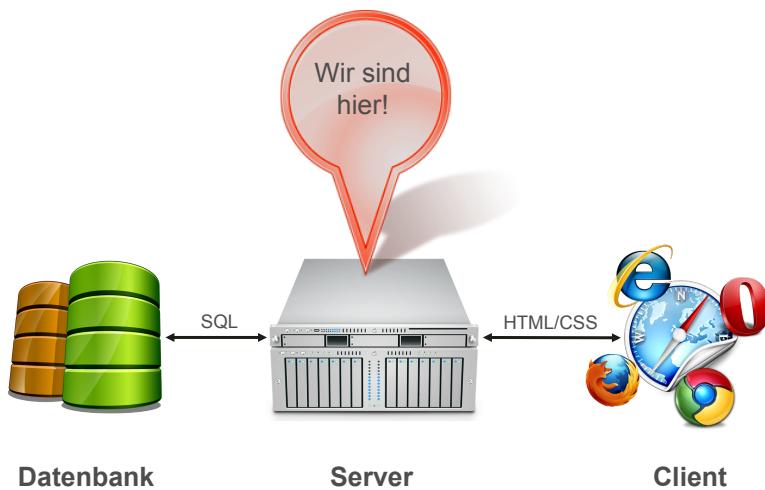
4

Datenbank – Server – Client

Wo waren wir nochmal?



University of Applied Sciences



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

5

Zum Nachlesen ...



University of Applied Sciences



Kapitel 4

PHP-Basics

Kapitel 4.3 Variablen definieren und ausgeben

Kapitel 2

Erste Schritte

Kapitel 3

Variablen und ihre Datentypen

Abschnitte 3.1 bis 3.7



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

6

Zum Nachlesen ...



University of Applied Sciences



Bereitgestellte Skripte:

PHP Programmierung

- **Kapitel 1:** Einleitung
- **Kapitel 2:** Serverseitige Programmierung
- **Kapitel 3:** Variablen

<http://praktische-informatik.fh-luebeck.de/node/39>

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

7

Inhalte der PHP Units



University of Applied Sciences

1

- Laufzeitmodelle und Programmierparadigma

2

- Elemente eines PHP-Programms

3

- Datentypen

4

- Operatoren

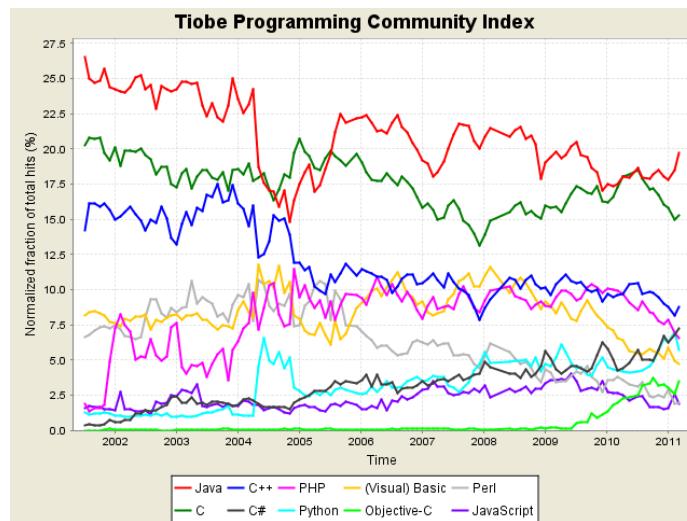
5

- Kontrollstrukturen

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

8

Verbreitung von Programmiersprachen

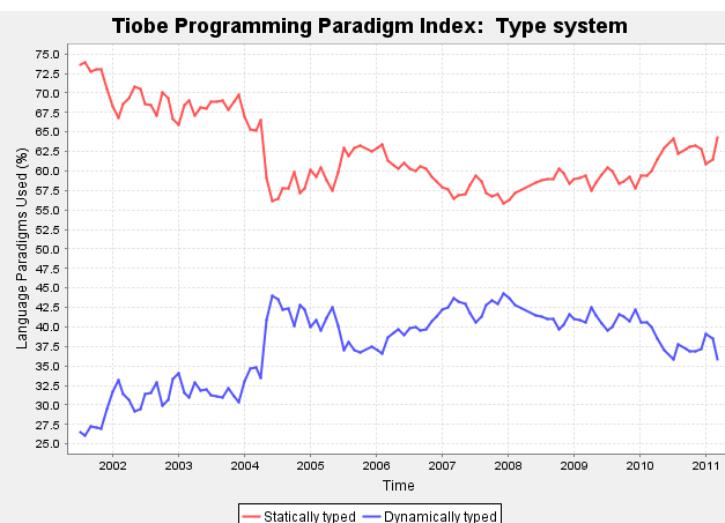


<http://www.tiobe.com>,
Stand: März 2011

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

9

Verbreitung von Programmiersprachen



<http://www.tiobe.com>,
Stand: März 2011

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

10

Die drei großen Paradigma



University of Applied Sciences

z.B.

Smalltalk, JAVA,
C++, C#,
ADA-95,
Eiffel, PHP



z.B.

C, PASCAL,
COBOL,
FORTRAN,
Assembler, PHP

z.B.

Prolog, Haskell,
SQL

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

11

Imperative Programmierung



University of Applied Sciences

Bei allen imperativen Programmiersprachen versteht man ein Computerprogramm als

- lineare Folge von Befehlen, die der Rechner in einer definierten Reihenfolge abarbeitet.
- Daten werden häufig in Variablen gespeichert. Die Werte in Variablen können sich im Programmablauf durch Befehlsabarbeitung ändern.
- Daher kann man sie auch als zustandsorientierte Programmierung bezeichnen.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

12

Deklarative Programmierung



University of Applied Sciences

In der deklarativen Programmierung wird formuliert, welches Ergebnis gewünscht ist.

- Bei deklarativen Paradigmen gibt es keine Nebeneffekte.
- Beweise (zum Beispiel Korrektheitsbeweis, Beweise über Programmeigenschaften) sind dank mathematischer Basis durchführbar.
- Aufgrund dessen jedoch teilweise geringe Akzeptanz (man spricht gern von sogenannten Akademiersprachen).

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

13

Objektorientierte Programmierung



University of Applied Sciences

Unter Objektorientierung versteht man eine Sichtweise auf komplexe Systeme, bei der ein System durch das Zusammenspiel kooperierender Objekte beschrieben wird.

- Ein Objekt hat
 - Attribute (Eigenschaften)
 - Methoden (Verhalten) und
 - kann Nachrichten empfangen und senden.
- Das Konzept der Objektorientierung wurde entwickelt, um die Komplexität von SW-Programme besser zu beherrschen.
- Das objektorientierte Programmierparadigma fasst Daten und zugehörige Programmteile zu einer Einheit zusammenzufassen, um Konzepte der realen Welt besser nachbilden zu können.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

14

Die drei gängigen Laufzeitmodelle von Programmiersprachen



University of Applied Sciences



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

15

Compiler



University of Applied Sciences

Ein Compiler ist ein Computerprogramm, das ein in einer Quellsprache geschriebenes Programm – genannt **Quellprogramm** – in ein semantisch äquivalentes Programm einer Zielsprache (**Zielprogramm**) umwandelt.

Üblicherweise handelt es sich dabei um die Übersetzung eines Quelltextes in direkt auf einem Rechner ausführbares Programm in Maschinensprache.

Das Resultat ist also ein nur auf einer spezifischen Rechnerarchitektur lauffähiges Programm. Vorteile liegen vor allem in der **Ausführungsgeschwindigkeit** der Programme.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

16

Interpreter



University of Applied Sciences

Interpreter **lesen** und **analysieren** den Quellcode eines Programmes und **führen** dann die entsprechenden **Aktionen aus**.

Dies ist im Vergleich zu Compilersprachen, bei denen das Programm vor seiner Ausführung in Maschinencode übersetzt wird, der dann vom Prozessor direkt ausgeführt wird, sehr **zeitaufwändig**.

Der Vorteil liegt darin, dass interpretierte Programmiersprachen auf jeder Rechnerarchitektur lauffähig sind, sofern es Interpreter für die Rechnerarchitektur gibt (**Portabilität**).

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

17

Byte Code



University of Applied Sciences

Bytecode ist eine Sammlung von Befehlen für eine **virtuelle Maschine**.

Bei Kompilierung eines Quelltextes mancher Programmiersprachen – wie beispielsweise **Java** – wird nicht direkt Maschinencode, sondern ein **Zwischenencode**, der Bytecode, erstellt.

Dieser Code ist in der Regel unabhängig von realer Hardware und im Vergleich zum Quelltext oft relativ kompakt. Dieser Ansatz verbindet Vorteile von Compilern (**Geschwindigkeit**) und Interpretern (**Portabilität**) in einem **Mittelweg**.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

18

Einordnung der Sprache PHP



University of Applied Sciences

Laufzeitmodell

| Programmierparadigma | Interpreter | Compiler | Byte-Code |
|----------------------|-------------------|--|-----------|
| | Imperativ | PHP | z.B. C |
| | Deklarativ | PHP (funktionale Prog-Anteile) | |
| | Objekt-orientiert | PHP | z.B. C++ |

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

19

Der PHP Programm-Run Zyklus



University of Applied Sciences



- Der Entwickler erzeugt **.php** Source Code-Dateien, die in einem **PHP Interpreter** oder **Virtual Machine** ausgeführt werden.
- Es gibt keinen Link-Lauf. Die **.php** Files werden **zur Laufzeit angezogen**.
- Die **.php** Dateien können auf unterschiedlichen serverseitigen Plattformen in PHP Interpretern oder PHP Virtual Machines ausgeführt werden.
- Die **.php** Dateien müssen hierzu nur in einem Verzeichnis des Servers abgelegt werden.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

20

Inhalte der PHP Units



University of Applied Sciences

- 1 • Laufzeitmodelle und Programmierparadigma

- 2 • Elemente eines PHP-Programms

- 3 • Datentypen

- 4 • Operatoren

- 5 • Kontrollstrukturen

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

21

Das klassische „Hello World“



University of Applied Sciences

```
<?php echo „Hello World“; ?>
```



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

22

Unser „Hello World“.php



University of Applied Sciences

```
<html>
  <head></head>
  <body>

    <?php
      $woerter = array("Hello", " ", "World");
      foreach ($woerter as $w) {
        echo $w;
      }
    ?>

  </body>
</html>
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

23

Unser „Hello World“.html



University of Applied Sciences

```
<html>
  <head></head>
  <body>

    Hello World

  </body>
</html>
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

24

Elemente eines PHP-Programms



University of Applied Sciences

Programmelemente

- **Anweisungen**
 - Variablen
 - Funktionen
 - Objekte

Anwendungen

- PHP in HTML einbinden
- Bibliotheken und Archive

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

25

Elementarste Sprachelemente



University of Applied Sciences

Elementarste Sprachelemente

Deklaration
von
Variablen

Auswertung
von
Ausdrücken
Anweisung

Steuerung
eines
Programm-
ablaufs

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

26

Anweisungen Beispiele



University of Applied Sciences

```
// Deklaration von Variablen
$zaehler = 5;

// Auswertung eines Ausdrucks
$a = $b + $c;

// Steuerung des Programmablaufs
if ($a > 0) {
    // Block 1
} else {
    // Block 2
}
```

Nutze die Variable *zaehler* mit dem Wert 5

Addiere den Wert von *b* zum Wert von *c* und weise das Ergebnis *a* zu

Wenn der Wert von *a* größer als 0 ist führe die Anweisungen des **Blocks**

Block 2

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

27

Wo finden wir es in unserem Beispiel?



University of Applied Sciences

```
<?php
$woerter = array("Hello", " ", "World");
foreach ($woerter as $w) {
    echo $w;
}
?>
```

Variablen

Auswertung von
Ausdrücken
Anweisungen

Steuerungen

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

28

Wo finden wir es in unserem Beispiel?



University of Applied Sciences

```
<?php
    $woerter = array("Hello", " ", "World");
    foreach ($woerter as $w) {
        echo $w;
    }
?>
```

Variablen

Auswertung von
Ausdrücken
Anweisungen

Steuerungen

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

29

Wo finden wir es in unserem Beispiel?



University of Applied Sciences

```
<?php
    $woerter = array("Hello", " ", "World");
    foreach ($woerter as $w) {
        echo $w;
    }
?>
```

Variablen

Auswertung von
Ausdrücken
Anweisungen

Steuerungen

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

30

Wo finden wir es in unserem Beispiel?



University of Applied Sciences

```
<?php
    $woerter = array("Hello", " ", "World");
    foreach ($woerter as $w) {
        echo $w;
    }
?>
```

Variablen

Auswertung von
Ausdrücken
Anweisungen

Steuerungen

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

31

Elemente eines PHP-Programms



University of Applied Sciences

Programmelemente

- Anweisungen
- **Variablen**
- Funktionen
- Objekte

Anwendungen

- PHP in HTML einbinden
- Bibliotheken und Archive

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

32

Variablen definieren (\$variable)



University of Applied Sciences

Variablen sind
Platzhalter für
Daten

Bezeichner eines
Speicherbereichs

Variablennamen
beginnen mit
einem \$-Zeichen

Regeln zur Benennung

- Unterscheidung von Groß- und Kleinschreibung
- Keine Zahl am Anfang einer Variable
- Keine Sonderzeichen bis auf _

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

33

Variablen definieren Beispiele



University of Applied Sciences

Groß- und
Kleinschreibung

- `$variable` ≠
`$Variable`

Ungültige
Variable

- `$0variable`

Ungültige
Variable

- `$variable!`

Gültige Variable

- `$wichtige_variable`

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

34

Wertzuweisung an Variablen mit dem Zuweisungsoperator =



University of Applied Sciences

Zuweisung von Werten

- Die Werte werden der Variablen zugewiesen
- `$name = "Tjorben";`
- `$alter = 2;`

Zuweisung eines Ausdrucks

- Der Ausdruck wird ausgewertet
- Das Ergebnis wird der Variablen zugewiesen
- `$erg = 17 + 4;`

Zuweisung einer Referenz (&\$variable)

- Es wird nicht der Inhalt, sondern nur der Zeiger auf die Variable zugewiesen. Beide Variablen referenzieren dieselbe Speicherstelle.
- `$a = "Hello World";`
- `$b = &$a;`

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

35

Initialisierung von Variablen



University of Applied Sciences

Initialisierung von Variablen ist nicht erforderlich

- es wird aber empfohlen dies zu tun.

Nicht initialisierte Variablen haben einen Vorgabewert der vom Typ abhängt

- Bool : FALSE;
- Zahlentypen: 0
- String: leerer String „“
- Array: leeres Array

Abhängig von den `error_reporting` Vorgaben wird bei Nutzung nicht initialisierter Variablen eine Notice ausgegeben.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

36

Variable Variablen (`$$variable`) Variable Funktionen (`$var=func`)



University of Applied Sciences

Zuweisung von Variablennamen an Variablen

- `$varname = „beispiel“;`
- `$$varname = „crazy php“;`
- `echo $beispiel;`
- Erzeugt die Ausgabe: crazy php

Zuweisung von Funktionen an Variablen

- `function id($x) { return $x; }`
- `$f = id;`
- `echo $f(„more crazy php“);`
- Erzeugt die Ausgabe: more crazy php

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

37

Variablen ausgeben Variableninterpolation



University of Applied Sciences

- Der Inhalt von Variablen kann mittels `echo` ausgeben werden.

```
$name = „Tjorben“;  
$alter = 2;  
echo $name;  
echo $alter;
```

Tjorben2

- Zur kombinierten Ausgabe von Text und Variablen bietet sich die Variableninterpolation.

```
$name = „Tjorben“;  
$alter = 2;  
echo „$name ist $alter Jahre  
alt“.
```

Tjorben ist 2
Jahre alt.

Wegen diesem Feature ist PHP so populär in der
Webprogrammierung.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

38

“ oder ““



University of Applied Sciences

- Zeichenketten können in PHP
 - in einfachen Hochkommata oder
 - doppelten Hochkommata ausgegeben werden.

```
$name = 'Tjorben';  
echo $name;
```

Tjorben

```
$name = "Tjorben";  
echo $name;
```

Tjorben

Jedoch ergibt:

```
$name = „Tjorben“;  
$alter = 2;  
echo '$name' . ' Variableninterpolation funktioniert nur in  
alt';
```

Zeichenketten mit doppelten Hochkommata „“ Jahre alt.

Prof. Dr. rer. nat. Nane Kratzke | 39
Praktische Informatik und betriebliche Informationssysteme

Variablennamen in der Variableninterpolation kennzeichnen



University of Applied Sciences

- Gegeben ist die Variable **\$name**.
- Sie wollen “**\$names** Geburtstag“ ausgeben, d.h. ein s direkt an den Namen hängen
 - Peter => Peters Geburtstag
 - Marie => Maries Geburtstag
 - ...

Wie lösen Sie dieses Problem, wenn Sie Variableninterpolation nutzen?

```
echo "$names Geburtstag";
```

Geburtstag

```
echo "$name s Geburtstag";
```

Peter s Geburtstag

```
echo "{$name}s Geburtstag";
```

Peters Geburtstag

Prof. Dr. rer. nat. Nane Kratzke | 40
Praktische Informatik und betriebliche Informationssysteme

Konstanten definieren (**define**)



University of Applied Sciences

- Der Wert von Konstanten ändert sich nicht
- Feststehende Werte können mit der Funktion **define** festgelegt werden
- Auf Konstanten wird ohne \$-Zeichen zugegriffen.
- Konstanten können keine Werte zugewiesen werden.
- Konstanten können nicht über Variableninterpolation ausgegeben werden.

```
define("MAXWERT", 100);  
echo MAXWERT;
```

100

```
define("MAXWERT", 100);  
echo "Max. Wert: MAXWERT";
```

Max. Wert: MAXWERT

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

41

Mini-Übung: Variablen (I)



University of Applied Sciences

- Welche Variablennamen sind korrekt?

var

Nicht korrekt

\$var7

korrekt

\$7var

Nicht korrekt

\$Var

korrekt

\$_Var

korrekt

\$var!

Nicht korrekt

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

42

Mini-Übung: Variablen (II)



University of Applied Sciences

- Welche Ausgabe erfolgt nach folgenden Wertzuweisungen?

```
$var = "Hello World"; echo $var;
```

Hello World

```
$var = 5 + 13; echo $var;
```

18

```
$x = $y + 13; echo $x;
```

13

```
$x = "Original";
$y = &$x;
$x = "Modified";
echo $y;
```

Modified

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

43

Mini-Übung: Variablen (II)



University of Applied Sciences

- Welche Ausgabe erfolgt nach folgenden Wertzuweisungen?

```
$var = "hello";
$$var = "World";
echo $hello;
```

World

```
$var = "variable";
$$var = "value";
echo $Variable;
```

Leerer String

```
$x = "x";
$$x = "value";
echo $x;
```

value

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

44

Mini-Übung: Variablen (III)



University of Applied Sciences

- Welche Ausgabe erfolgt nach folgenden Wertzuweisungen?

```
function func(x) {  
    return x * x;  
}  
$n = func;  
$var = 10;  
echo $n($var / 2);  
  
function func($a, $b) {  
    $x = $a + $b;  
    return $a;  
}  
$n = func;  
$x = 10; $y = 30;  
echo $n($x, $y);
```

25

10

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

45

Mini-Übung: Variablen (IV)



University of Applied Sciences

- Welche Ausgabe erfolgt nach folgenden Wertzuweisungen?

```
$name = "Peter";  
$alter = 34;  
echo "$name ist $alter Jahre alt.;"
```

Peter ist 34
Jahre alt.

```
define("MINALTER", 18);  
echo "Minimales Alter: MINALTER";
```

Minimales
Alter:
MINALTER

```
$name = "Sabine";  
$alter=23;  
echo '$name ist $alter Jahre alt.';
```

\$name ist
\$alter Jahre alt.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

46

Zusammenfassung



University of Applied Sciences

- **Programmierparadigmen**
 - PHP ist eine imperative und objektorientierte Programmiersprache
 - PHP ist eine interpretierte Programmiersprache (es gibt byte code Compiler)
 - PHP ist dynamisch typisiert
- **Elemente eines PHP-Programms (I)**
 - Anweisungen
 - Variablen
 - Konstanten
 - Variablen zum „Speichern“ von Funktionen
 - Variable Variablen
 - Variableninterpolation