Dynamische Webprogrammierung

Skript

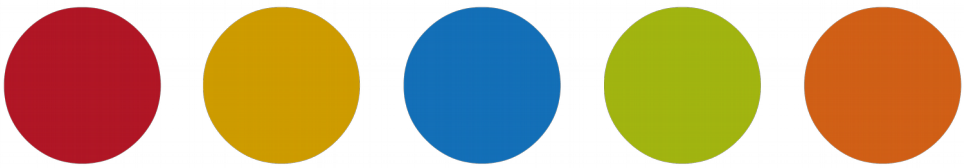
Materialsammlung

Schulung:	Informatik und Wirtschaftsinformatik

Unterrichtsbegleitendes E-Learning:

→ E-Learning OOP

Stand: 19. Apr 2020



https://edublog.emotionalspirit.de/

Inhaltsverzeichnis

1 Allgemeines

Das Skript schildert den Umgang mit der Programmiersprache PHP (objektorientiert) anhand von konkreten Beispielen die sich auch in den Unterricht im Fachbereich Wirtschaftsinformatik respektive im Fachbereich Informatik einbetten lassen. Alle Oberstufenlehrpläne der Sekundarstufe 2 (Jahrgang 2 in Klasse 13 Berufliches Gymnasium, Berufskolleg Jahrgang 1, Jahrgang 2 und im Berufskolleg Wirtschaftsinformatik Jahrgang 1) enthalten eine Einheit „Webprogrammierung".

Aktuelle Versionen des Skriptes selbst und die im Skript behandelten Quellcodes können Sie online herunterladen und testen:

Skript & Sources:

Aktuelle Unterrichtssoftware und Materialien

Webseiten mit PHP (objektorientiert) programmieren

Skript & Sources:

Webprogrammierung im Unterricht

E-Learning OOP

Alle Quellangaben wurden nach bestem Gewissen genannt und aufgeführt. Permanent begleitende Literatur waren:

[PHP01]

Achour, Mehdi , Betz, Friedhelm, et al, "PHP-Handbuch", 2001-2015, https://secure.php.net/manual/de/index.php, zuletzt geprüft am 23.11.15

[PHP02]

w3schools, "PHP 5 Tutorial", 1999-2015, http://www.w3schools.com/php/, zuletzt geprüft am 23.11.15

[WEB01]

w3schools, "HTML(5) Tutorial", 1999-2015, http://www.w3schools.com/html/default.asp, zuletzt geprüft am 23.11.15

[HRE01]

Landesinstitut für Schulentwicklung Baden-Württemberg, "Handreichung zur Lehrplaneinheit 'Objektorientierte Systementwicklung'", 1999-2015, http://www.ls-bw.de/Handreichungen/reihe_beruflich/spezifisch/bg/Informatik/HR_JG1_OO%20vorlaeufig.pdf , zuletzt geprüft am 23.11.15

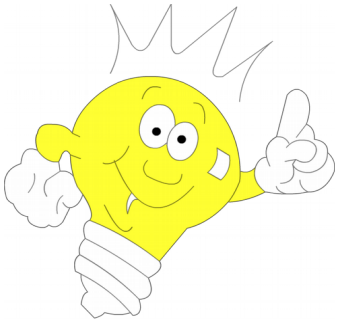
[HRE02]

Landesinstitut für Schulentwicklung Baden-Württemberg, "Handreichung zur Lehrplaneinheit 'Design und Realisierung von Internetseiten'", 1999-2015,http://www.ls-bw.de/Handreichungen/reihe_beruflich/spezifisch/bg/Informatik/hr_inf_j2_bg.pdf , zuletzt geprüft am 23.11.15

[PBR01]

Vortrag ZAG Jahrestagung Informatik, "Ich-Kann-Liste", 2015, Pia Brunner

2 Einführung in PHP



Gut zu Wissen! Voraussetzung für die Entwicklung ist eine geeignete Entwicklungsumgebung. Für den Unterricht ist es für die Schüler und die Lehrkraft von Vorteil, wenn Sie die Digitale Tasche (Informatikstick) nutzen. Diese wird vom Land Baden-Württemberg bereit gestellt und enthält u.a. anderem Eclipse mit den notwendigen Erweiterungen und das Softwarepaket Xampp.

Objektorientierte Softwareentwicklung. Wie in allen objektorientierten Sprachen stehen unter anderem die Grundprinzipien der Abstraktion , Wiederverwendbarkeit, Zerlegung , Vererbung und Kapselung im Vordergrund. Vor ein paar Jahren hat man noch unter den Fachkundigen eifrig diskutiert, ob die Objektorientierung ein fundamentaler Aspekt der Softwareentwicklung werden wird. Zwischenzeitlich können wir vermutlich guten Gewissens behaupten, dass objektorientiert entwickelte Software Systeme mehr Aussicht auf langfristigen Erfolg haben. Ein absoluter Erfolgsfaktor also. Die Mehrheit der Programmiersprachen besitzen zwischenzeitlich objektorientierte Nachfolger. So sind C++ und auch Java objektorientierte Nachfolger der Programmiersprache C. PHP5 ist dagegen der objektorientierte Nachfolger von PHP4. Andere Programmiersprachen sind noch relativ jung und sind schon von Beginn an objektorientiert. Dazu gehören u.a. die seit 1991 und 1995 existierenden Sprachen Python und Ruby. Alle genannten objektorientierten Sprachen sind auch imperativ und setzen ein weitere grundsätzliche Denkweise (Programmierparadigma) um. Imperative Sprachen enthalten vorgefertigte Strukturen, die für die Abarbeitung von Alternativen und Wiederholungen genutzt werden können, außerdem sehen diese Sprachen vor Programmeinheiten (Module, Prozeduren) zu schaffen die in anderen Zusammenhängen wiederverwendet werden können.

Warum ist das so? Nach dem Motto „Ordnung ist das halbe Leben" wird Quellcode konkreten Objekten zugeordnet, organisiert. Der Programmcode enthält die Beschreibung (Eigenschaften und Verhaltensweisen) dieser Objekte. Der Programmierer entscheidet dann, wie die Objekte untereinander beliebig interagieren sollen.

Objekte deren Eigenschaften und Verhaltensweisen die gleichen oder ähnliche Ausprägungen besitzen packt man in einem Klassengrundgerüst zusammen. Mit einer Klasse schafft sich der Programmierer Muster, also eine Art Vorlage, für eine ganze Menge von Objekten. Das Prinzip nennt sich Abstraktion.

Wenn sich Eigenschaften oder Verhaltensweisen von Objekten ändern, verändert oder erweitert der Programmierer den Quellcode. Bestenfalls hat jede Eigenschaft und Verhaltensweise seinen festen Platz. Um Redundanzen (Widersprüche) zu vermeiden, ist es zielführend Wiederholungen im Quellcode zu vermeiden. Damit wird die Wartbarkeit eines Systems langfristig sichergestellt.

Da wir Menschen im Prinzip von Kind auf intuitiv objektorientiert denken, fällt es uns in der Regel leicht dieses Prinzip in der Programmierung einzusetzen.

Alles auf dieser Welt sind Objekte (Dinge), Dinge die man getrennt betrachten oder aber auch Dinge die wir gruppieren oder zusammenfassen können, da sie gleich oder ähnlich sind.

Wir organisieren unseren Alltag ständig nach diesem Prinzip, stecken alles zusammen, was zusammen gehört! Damit fällt es uns relativ leicht den Überblick zu behalten. Stellen Sie sich doch die folgenden Fragen:

Woran erkennen Sie ein Auto ?

Was haben alle Fahrzeuge gemeinsam?

Was unterscheidet ein Mensch vom Tier ?

Wann ist ein Stück Land eine Insel?

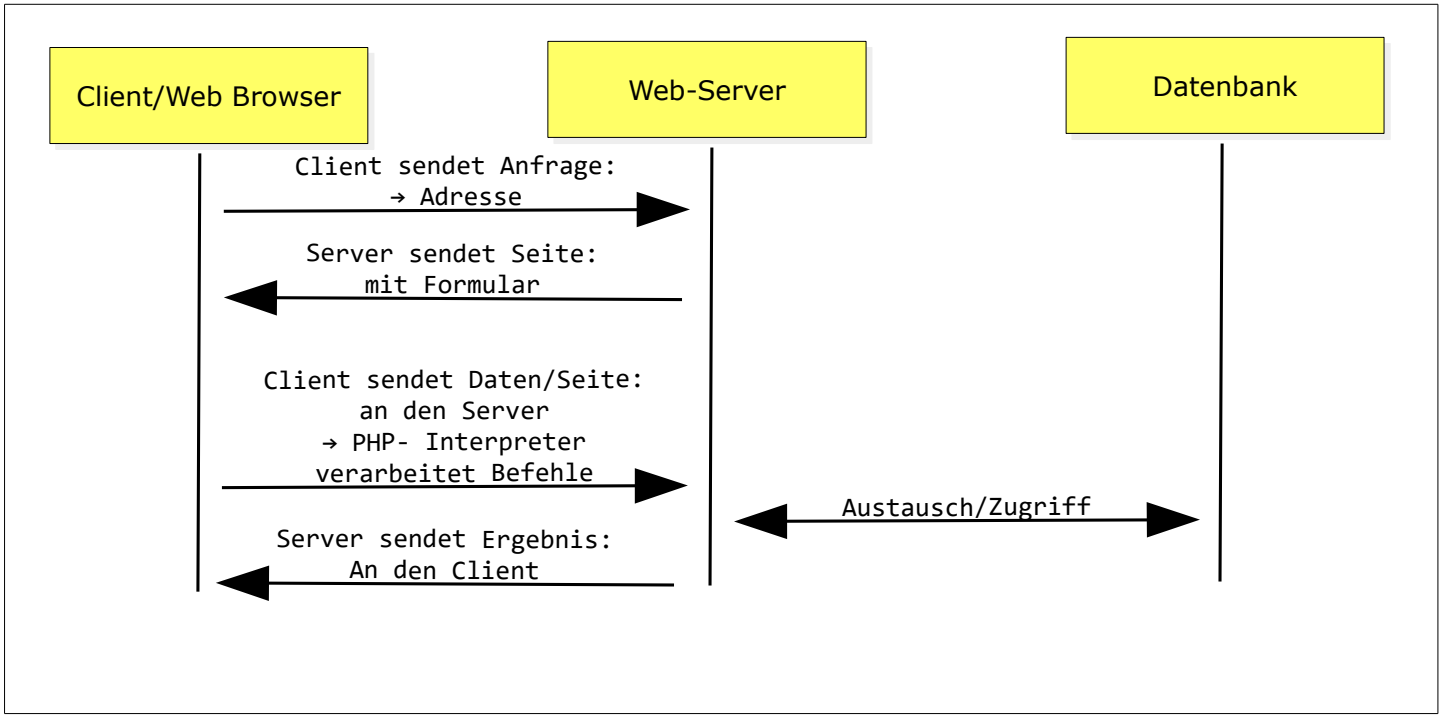
Was macht eine Uhr zur Uhr?

Wie viele unterschiedliche Automaten kennen Sie?

Die Objektorientierung ist eine Art Managementprinzip das sicherstellen soll, dass die Software erweiterbar, wartbar und sicher ist und über die Zeit hinweg auch bleibt.

2.1 Arbeits- und Informationsmaterial: Einführung in PHP

Thema:	Einführung in PHP Autor: Christine Janischek Client-Server-Prinzip

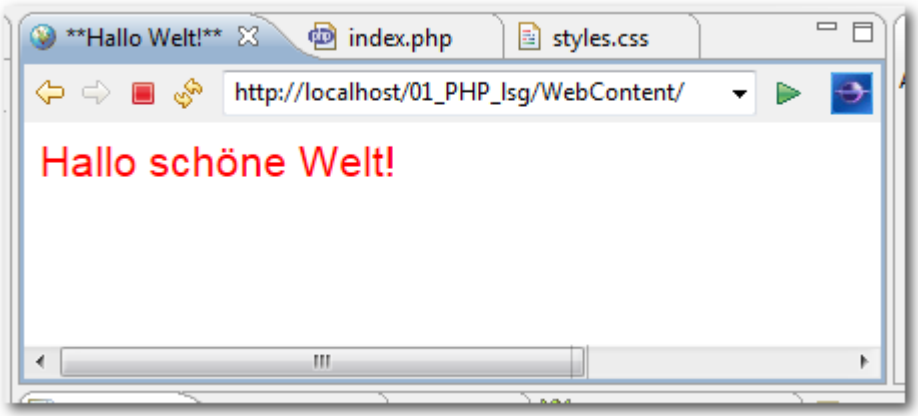


Grundkonzept und Technik[1]

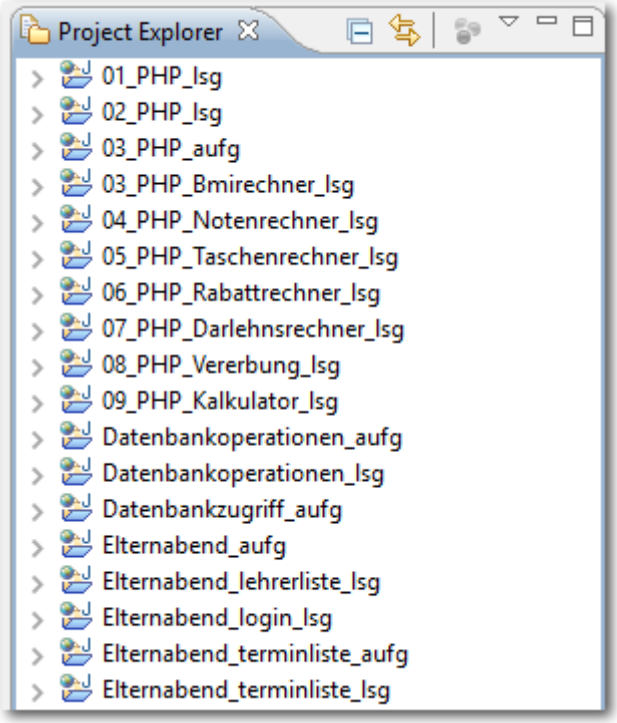
- ✗ Webseiten werden durch den Zusatz von speziellen Tags programmierbar
- ✗ Seitenbestandteile werden erst beim Aufruf der Seite durch den Client, quasi in Echtzeit zusammengebaut. Die Seite wird also dynamisch erzeugt!
- ✗ Der Benutzer kann auf der Benutzeroberfläche interagieren (Eingaben tätigen und Schaltflächen bedienen). Die Anwendung verarbeitet die Daten, erzeugt und präsentiert ein Ergebnis. Durch PHP wird die HTML-Anwendung also erst richtig interaktiv und in gewisser Weise intelligent.
- ✗ absolute Browser-Unabhängigkeit, da die Anwendungen auf dem Server ablaufen. Der Benutzer kann also keine Einschränkungen durch spezielle Einstellungen im Browser verursachen (z.B. deaktivieren von Javaskript).
- ✗ einfache Datenbankzugriffe auf eine MySQL-Datenbank, über die Schnittstelle ODBC auf Access-Datenbanken und sqllite-Datenbank (Mobile-Apps)
- ✗ Grundlage z.B. für die meisten Content-Management-Systeme (CMS).

1 Auszug in Anlehnung an das Skript von Johannes Kaiser

2.2 Leittext: Einführung in PHP



View: Ergebnis der Übung (index.php)



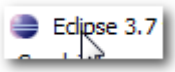
Arbeitsverzeichnis ausfindig machen.

Öffnen Sie dazu Ihren USB-Stick und wählen Sie die Verzeichnisse → Informatikstick → EigeneDateien → myPHP.

Dieses Verzeichnis nutzen Sie künftig als Workspace für Ihre PHP-Projekte in Eclipse.

Hinweis:
Merken Sie sich unbedingt den Ort dieses Verzeichnisses:

\Informatikstick\EigeneDateien\myPHP

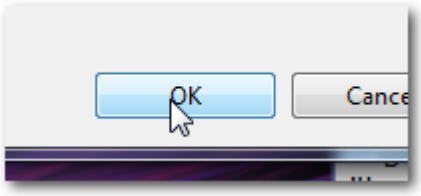
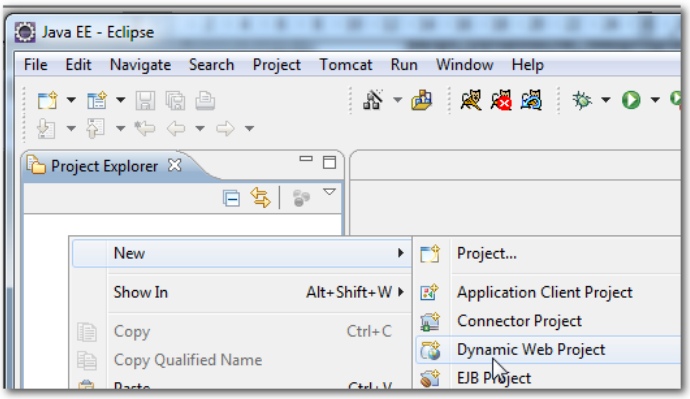
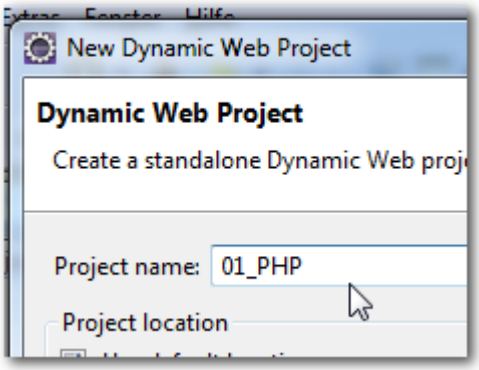


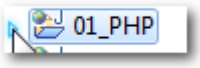
Eclipse starten.

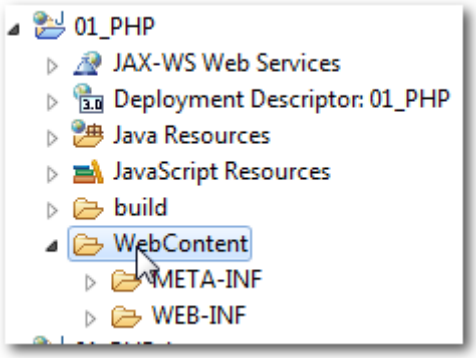
Klicken Sie dazu auf die Datei → eclipse.exe im Programmverzeichnis oder starten Sie alternativ Eclipse über das Start-Mnü Ihrer Digitalen Tasche (Informatikstick).

Workspace öffnen.

Wählen Sie über die Schaltfläche → Browse das

	gerade erzeugte Verzeichnis → \Informatikstick\EigeneDateien\myPHP aus.  Mit einem abschließenden Klick auf die Schaltfläche → OK öffnen Sie den Workspace.
 	*Neues Projekt erstellen.* Für die Dynamische Webseite benötigen wir ein neues Projektverzeichnis vom Typ → Dynamic Web Project Klicken Sie dazu im linken Fenster von Eclipse für das Kontext-Menü (rechte Maustaste) und wählen Sie die Option → New → Dynamic Web Project aus. Für den Fall, dass der Eintag im Kontext-Menü nicht erscheint wählen Sie die Option → Other → Web → Dynamic Web Project aus.
 	*Projektname festlegen.* Geben Sie den Namen für Ihr Projekt an und belassen Sie alle anderen Einstellungen. Schließen Sie den Vorgang mit einem Klick auf die Schaltfläche → Finish ab. 



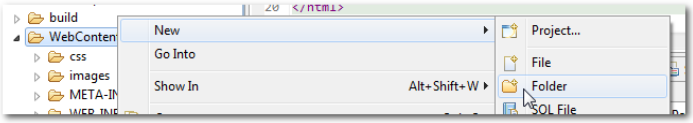


Öffnen Sie das Projektverzeichnis.

Klicken Sie dazu auf den kleinen blauen Pfeil links neben dem Projektname und wählen Sie mit einem Klick das WebContent-Verzeichnis aus.

→ WebContent

Hinweis:
In diesem Verzeichnis werden wir die Inhalte des Projektes platzieren.

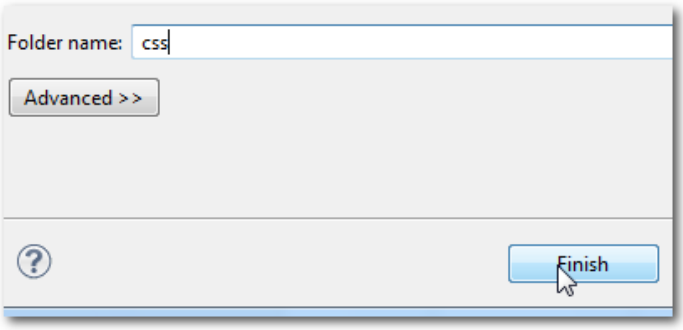


Arbeitsplatz organisieren.

Erzeugen Sie dazu im WebContent-Verzeichnis die zwei neuen zusätzlichen Unterverzeichnisse:

Klicken Sie dazu das WebContent-Verzeichnis an und wählen Sie im Kontext-Menü (rechte Maustaste) die Optionen → New → Folder.

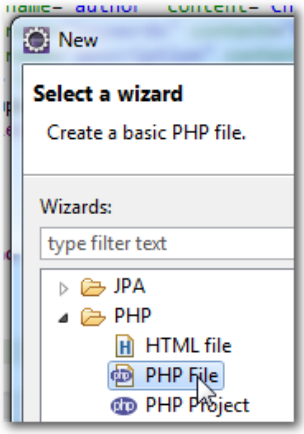
Wir werden diese Verzeichnisse zu einem späteren Zeitpunkt benötigen.



Verzeichnis benennen.

Geben Sie den Verzeichnisnamen → css an und klicken Sie auf die Schaltfläche → Finish.

Wiederholen Sie diesen Vorgang für das Verzeichnis → images



Neue PHP-Datei erzeugen.

Klicken Sie dazu das WebContent-Verzeichnis an und wählen Sie im Kontext-Menü (rechte Maustaste) die Optionen → New → Other → PHP → PHP File.

Klicken Sie dann auf die Schaltfläche → Next

Geben Die den Dateinamen ein
→ index.php

und klicken Sie abschließend auf die Schaltfläche
→ finish

index.php

```
1  <!DOCTYPE HTML>
2
3  <html>
4  <head>
5       <title>**Hallo Welt!**</title>
6       <meta name="author" content="Christine Janischek">
7       <meta name="keywords" content="Hallo, Welt">
8       <meta name="description" content="Meine erste PHP-Seite">
9       <style type="text/css">
10         @import "css/styles.css";
11      </style>
12 </head>
13 <body>
14
15     <?php
16         echo "Hallo sch&ouml;ne Welt!"
17     ?>
18
19 </body>
20 </html>
```

Index.php

Einfache Ausgaben in PHP erzeugen.

Ergänzen Sie nun den angezeigten Quellcode und Speichern Sie die Veränderung (STRG + S):

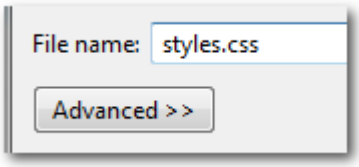
```
<?php
      echo "Hallo sch&ouml;ne Welt!"
?>
```

Eingabehilfe:

```
<!DOCTYPE HTML>

<html>
<head>
      <title>**Hallo Welt!**</title>
       <meta name="author" content="Ihr Name">
       <meta name="keywords"
            content="Hallo, Welt">
       <meta name="description"
            content="Meine erste PHP-Seite">
       <style type="text/css">
            @import "css/styles.css";
       </style>
</head>
<body>

      <!-- hier fehlt Quellcode --->

</body>
</html>
```

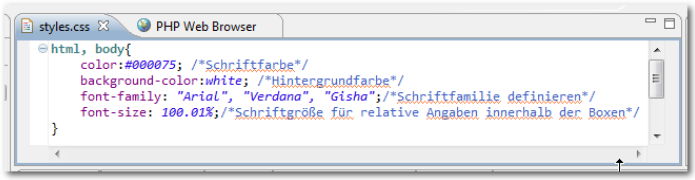
Stildatei erzeugen.

Klicken Sie dazu auf das Verzeichnis → css und wählen Sie im Kontext-Menü (rechte Maustaste) die Optionen → New → Other → Web → CSS File → Schaltfläche Next →



Geben Sie als Dateinamen → styles.css ein und klicken Sie abschließend auf die Schaltfläche

→ finish



Einfache Formatierung im CSS festlegen.

Öffnen Sie die Datei mit einem Doppelklick und übertragen Sie die folgenden Stilvorgaben:

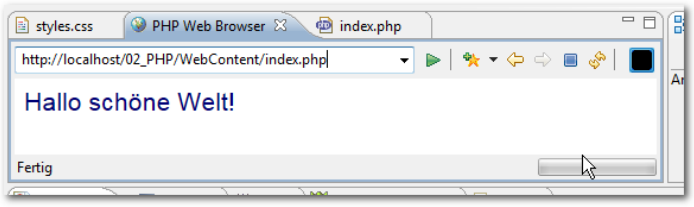
Eingabehilfe:

```
html, body{
/*Schriftfarbe*/
	color:#000075;

/*Hintergrundfarbe*/
	background-color:white;

/*Schriftfamilie definieren*/
	font-family: "Arial", "Verdana", "Gisha";

/*Schriftgröße für relative Angaben innerhalb der Boxen*/
	font-size: 100.01%;
}
```

im Web Browser

Testen Sie die Anwendung.

Öffnen Sie dazu den Internetbrowser und geben Sie den Pfad zur gerade erstellen Datei an. Geben Sie dazu den in der Grafik angezeigten Pfad ein.



View: index.php

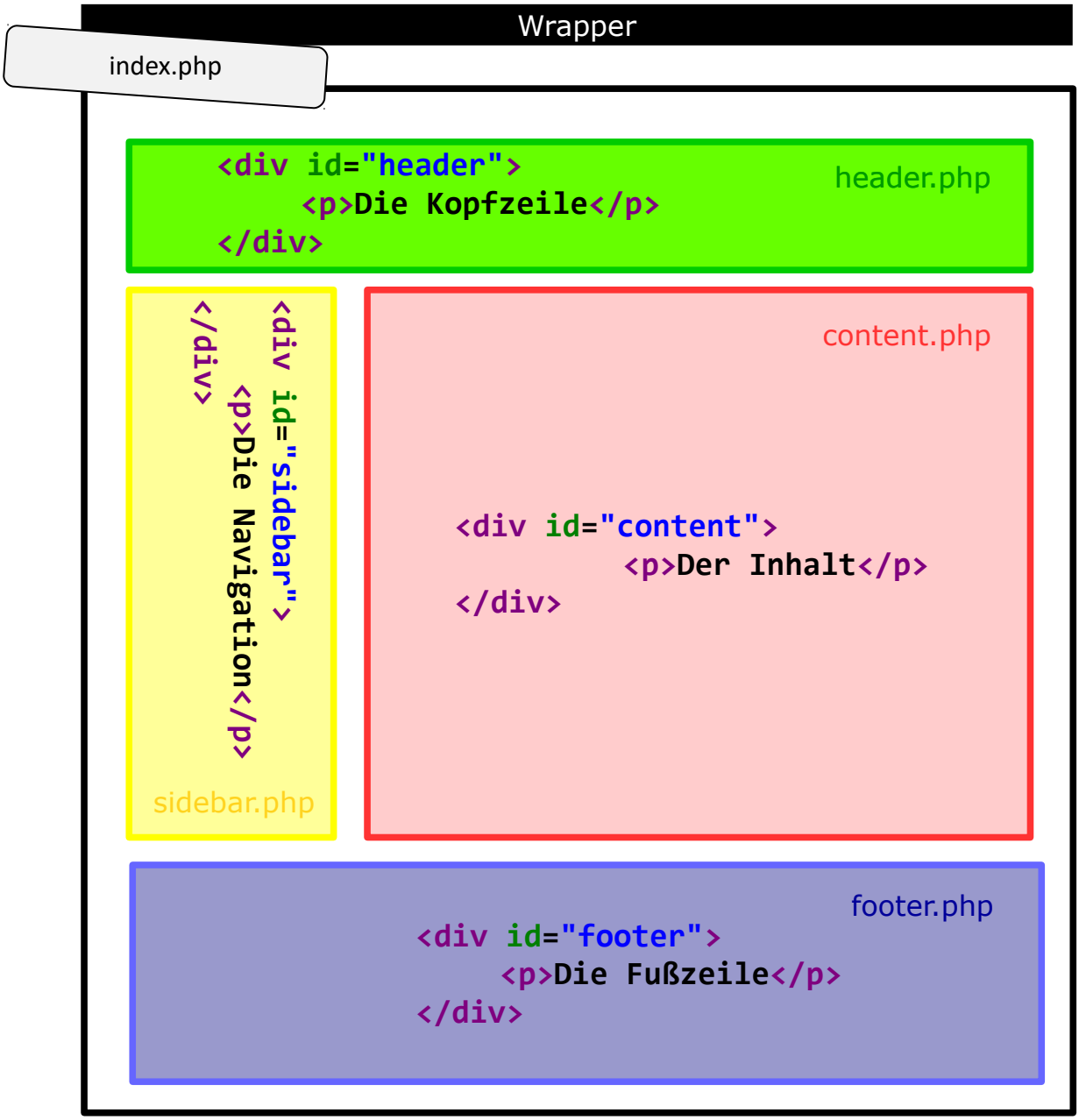
http://localhost/01_PHP/WebContent/index.php

Herzlichen Glückwunsch Sie haben ihre erste PHP-Seite erstellt.

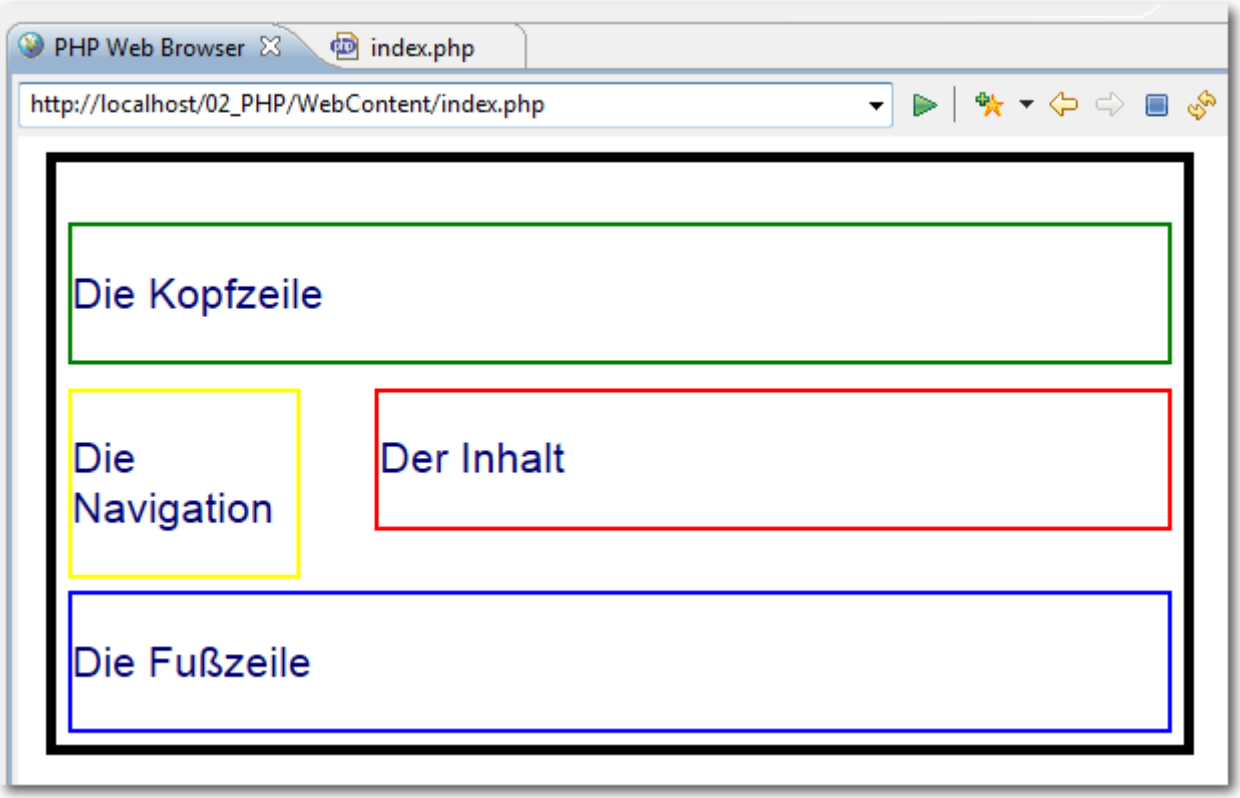
3 Grundgerüst eines dynamischen Web-Layouts in PHP

3.1 Informationsblatt: Grundgerüst eines dynamisches Web-Layout

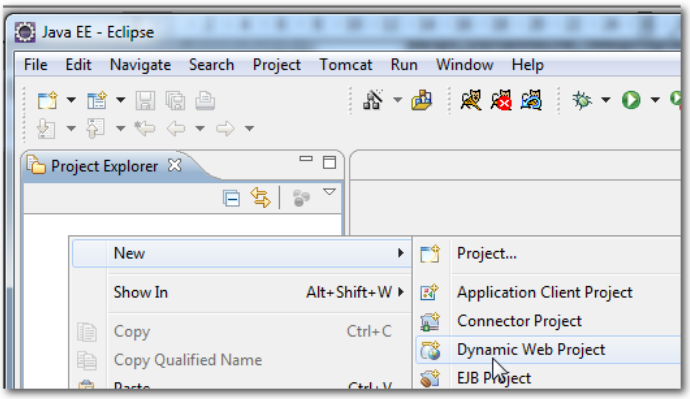
Thema: 	Grundgerüst eines dynamischen Web-Layouts in PHP Autor: Christine Janischek

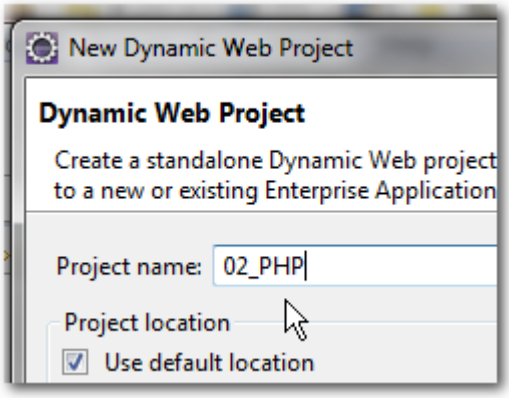
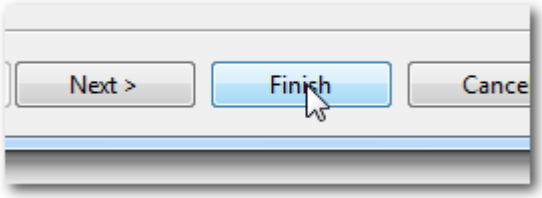
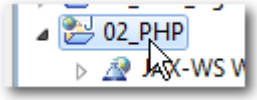


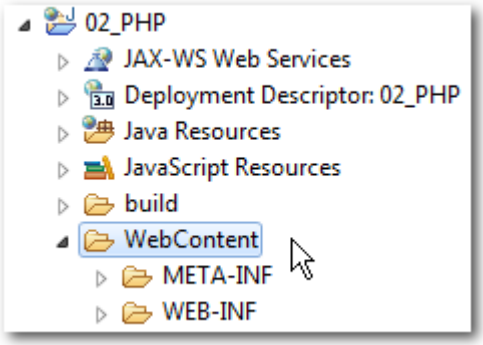
3.2 Leittext: Ein dynamisches Web-Layout in PHP



View: Ergebnis der Übung (index.php)

	*Neues Projekt erstellen.* Für die Dynamische Webseite benötigen wir ein neues Projektverzeichnis vom Typ → Dynamic Web Project Klicken Sie dazu im linken Fenster von Eclipse für das Kontext-Menü (rechte Maustaste) und wählen Sie die Option → New → Dynamic Web Project aus. Hinweis: Sie finden die Option auch → New → Other → Web → Dynamic Web Project.
	Projektname festlegen. Geben Sie den Namen für Ihr Projekt an und

<table>
<tr>
<td> 
</td>
<td>belassen Sie alle anderen Einstellungen. Schließen Sie den Vorgang mit einem Klick auf die Schaltfläche → Finish ab.
 
</td>
</tr>
<tr>
<td>Aktuelles Projekt:
 

 
</td>
<td>Öffnen Sie das Projektverzeichnis.

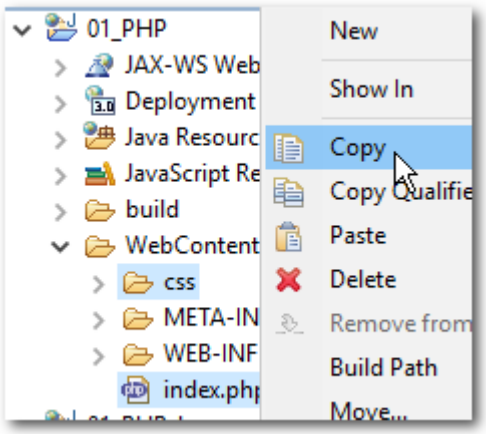
Klicken Sie dazu auf den kleinen blauen Pfeil links neben dem Projektname und wählen Sie mit einem Klick das WebContent-Verzeichnis aus.

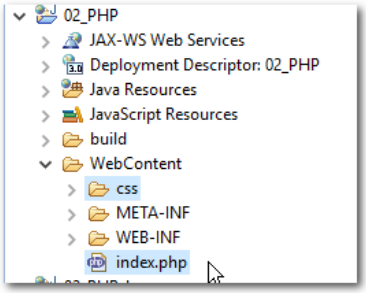
→ WebContent

Hinweis:
In diesem Verzeichnis werden wir die Inhalte des Projektes platzieren.</td>
</tr>
<tr>
<td>Letztes Projekt:</td>
<td>Prinzip der Wiederverwendung.

Öffnen und kopieren Sie die Ergebnisse des letzten Projektes in das WebContent-Verzeichnis des aktuellen Projektes.

Aktuelles Projekt:</td>
</tr>
</table>





Legen Sie ein Verzeichnis an für Bilder und Grafiken. Nennen Sie das Verzeichnis → images. Das Verzeichnis bleibt vorerst leer. Zu einem späteren Zeitpunkt das Verzeichnis nutzen.

Fortsetzung:

```
#header{
/*Rahmen 2 Pixel breit durchgehend grün*/
        border:2px solid green;
}

#content{
/*Rahmen 2 Pixel breit durchgehend grün*/
       border:2px solid red;

/*Ausrichtung des Elements rechts*/
       float: right;

/*Breite der Box: relativ 70%*/
       width: 70%;

/*Höhe der Box: automatische Ausrichtung*/
       height:auto;
}

#sidebar{
/*Rahmen 2 Pixel breit durchgehend gelb*/
       border:2px solid yellow;

/*Ausrichtung des Elements links*/
       float: left;

/*Breite der Box: relativ 20%*/
       width: 20%;

/*Höhe der Box: automatische Ausrichtung*/
       height:auto;
}

#footer{
```

Ergänzen Sie die Stildatei um das Box-Modell.

Wir legen für alle Seitenbestandteile eine Div-Box an und legen eine eigene Rahmenfarbe fest.

Öffnen Sie die Stildatei → styles.css und ergänzen Sie den angezeigten Quellcode für das Box-Modell und speichern Sie die Änderungen (STRG + S).

Hinweis:
Die wrapper-Box wird später den Container bilden der die ganze Seite im Internetbrowser zusammenhält und ausrichtet. Weitere Details folgen zu einem späteren Zeitpunkt.

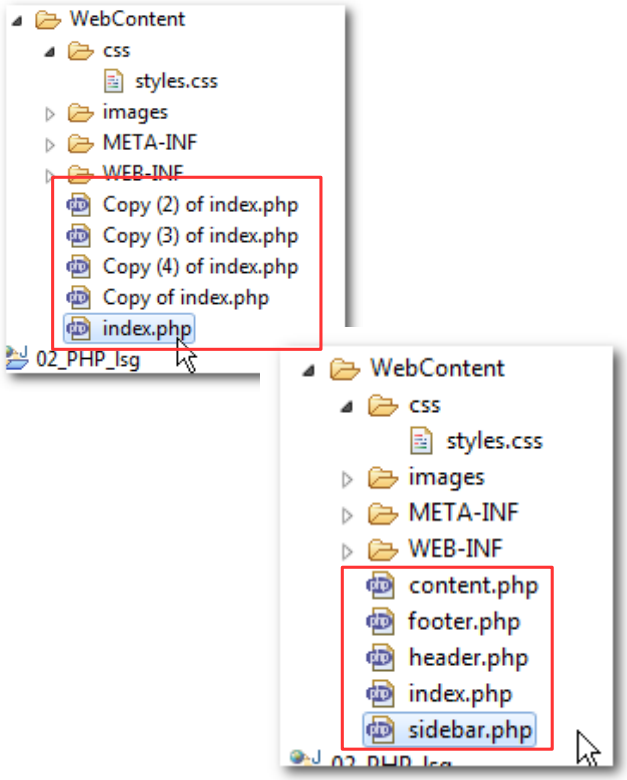
Anfang:

```
div{
/*Aussenabstand: relativ 1% von allen Seiten*/
       margin: 1%;
}

#wrapper{
/*Block Element*/
       display:block;

/*Rahmen 5 Pixel breit durchgehend schwarz*/
       border:5px solid black;

/*Breite der Box: relativ 90%*/
       width: 90%;
```

```
/*Rahmen 2 Pixel breit durchgehend blau*/
	border:2px solid blue;

/*Beendet das Umfließen von Elementen*/
	clear:both;
}
```

}

Ergänzen Sie auch den nebenstehenden Quellcode.



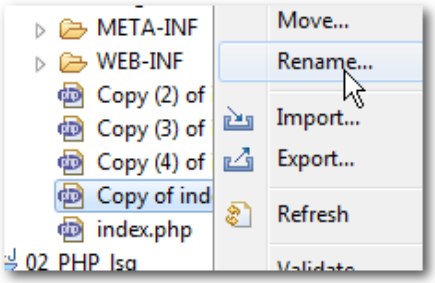
Prinzip der Zerlegung.

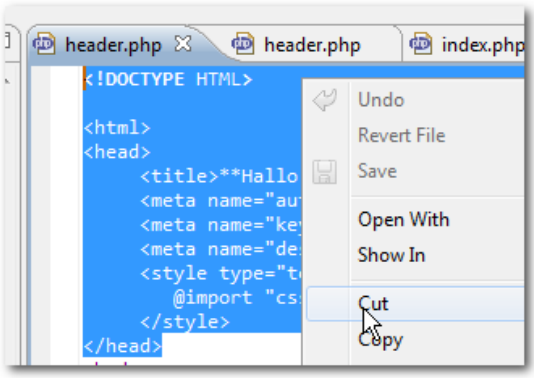
Um das Box-Modell zu nutzen benötigen wir weitere PHP-Dateien für

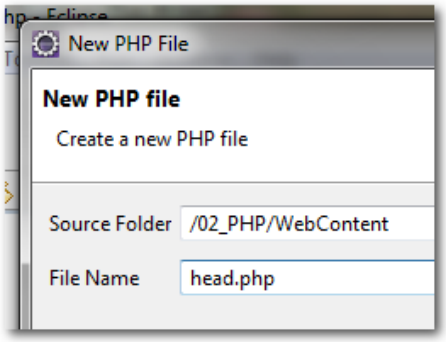
- die Kopfzeile (header),
- den Inhalt (content),
- die Navigation (sidebar) und
- die Fußzeile (footer).

Jeder Seitenbestandteil wird also in eine extra PHP-Datei ausgelagert.

Kopieren Sie dazu die Datei → index.php (→ STRG + C → STRG +V) vier mal und benennen Sie die Dateien wie in der nebenstehenden Grafik angezeigt um.







Linkes Fenster → Kontext-Menü → New → Other → PHP → PHP-File → Schaltfläche Next klicken → File Name eingeben → Schaltfläche Finish klicken

Kopf zerlegen.

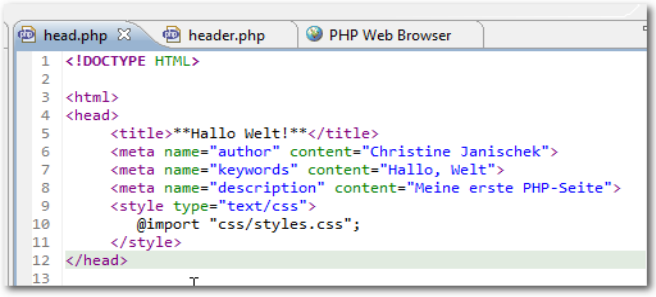
Wir zerlegen und differenzieren dazu

→ head

→ header

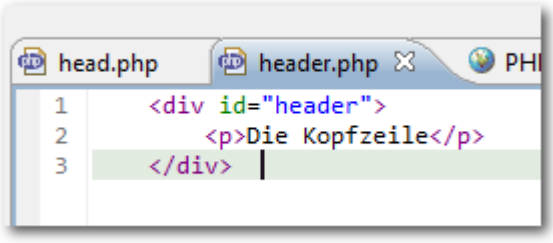
Öffnen Sie dazu zuerst die Datei
→ header.php im PHP Editor.

Schneiden Sie den oberen Teil (DOCTYPE und head) aus → STRG +X und fügen Sie den Quellcode in eine neue PHP-Datei ein STRG +V.



head.php

Nennen Sie die Datei → head.php und speichern Sie den Inhalt ab.



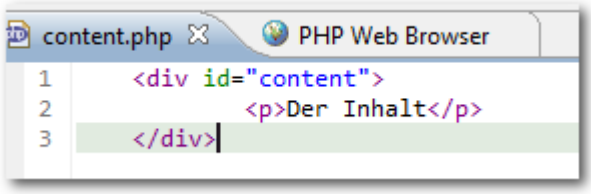
Inhalt der Kopfzeile (header) definieren.

Wechseln Sie dazu erneut in die Datei → header.php.

Verändern Sie den Quellcode wie folgt:

```
<div id="header">
      <p>Die Kopfzeile</p>
</div>
```

Speichern Sie die Veränderungen
→ STRG +S



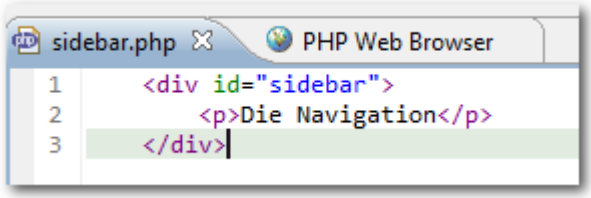
Den Inhalt (content) definieren.

Öffnen die Sie Datei → content.php.

Ändern Sie die Datei → content.php, wie folgt:

```
<div id="content">
    <p>Der Inhalt</p>
</div>
```

Speichern Sie die Veränderungen
→ STRG +S



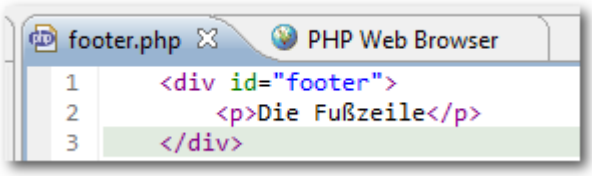
Inhalte der Navigation (sidebar) definieren.

Ändern Sie auf die gleiche Weise den Quellcode für die → sidebar.php und footer.php.

Ändern Sie die Datei → sidebar.php, wie folgt:

```
<div id="sidebar">
    <p>Die Navigation</p>
</div>
```

Speichern Sie die Veränderungen
→ STRG + S

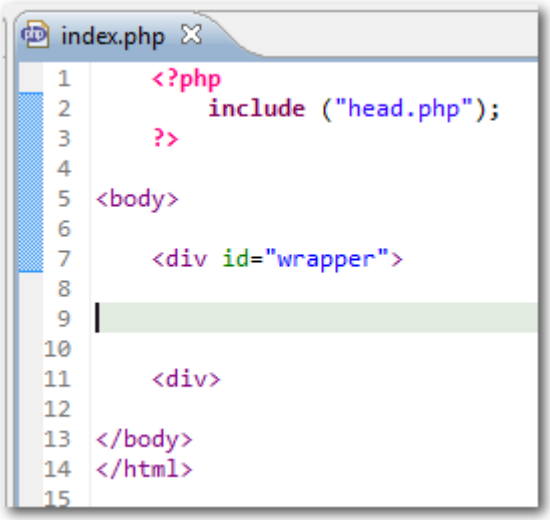


Inhalte der Fußzeile (footer) definieren.

Ändern Sie die Datei → footer.php, wie folgt:

```
<div id="footer">
    <p>Die Fu&szlig;zeile</p>
</div>
```

Speichern Sie die Veränderungen
→ STRG +S



index.php

Der PHP-include-Befehl:

```
<?php
      include ("dateiname.php");
?>
```

Wenn der include-Befehl innerhalb einer Funktion aufgerufen wird, verhält sich der gesamte Code aus der aufgerufenen Datei, wie wenn er in der Funktion stände. Folglich hat er den selben Variablen-Gültigkeitsbereich wie diese Funktion.

Zusammenführung aller Seitenbestandteile

In der Datei → index.php werden alle Seiten:

```
→ head.php
→ header.php
→ sidebar.php
→ content.php
→ footer.php
```

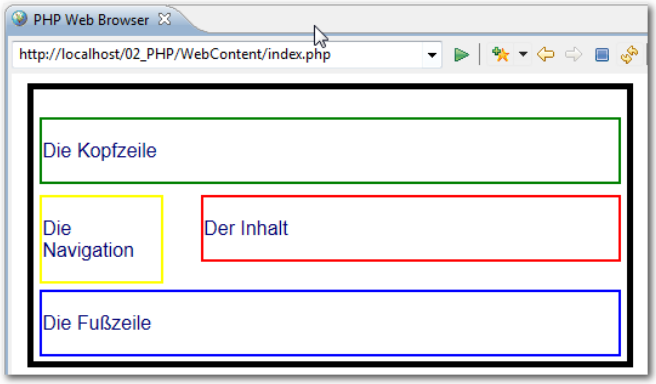
integriert (zusammengeführt). Die wrapper-Box umfasst davon alle sichtbaren Seiten innerhalb des body-Tags.

Übernehmen Sie dazu den nebenstehenden Quellcode in die Datei → index.php und ergänzen Sie nachträglich die fehlenden Bestandteile:

```
<?php
      include ("head.php");
?>

<body>

      <div id="wrapper">
            ...
      </div>

</body>
</html>
```

Nutzen Sie die Angaben im Informationsblatt.

Im Web Browser



View: index.php

Testen Sie die Anwendung.

Öffnen Sie dazu den Internetbrowser und geben Sie den Pfad zur gerade erstellen Datei an. Geben Sie dazu den in der Grafik angezeigten Pfad ein.

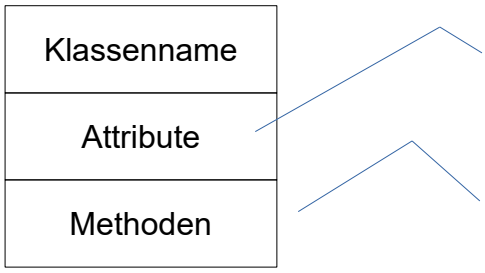
http://localhost/02_PHP/WebContent/index.php

Herzlichen Glückwunsch Sie haben ihr erstes dynamisches Layout erstellt.

4 Bmirechner: Formulare auswerten

4.1 Informationsblatt: Objekten, Klassen, Attribute und Methoden

Thema:	Einführung in PHP Autor: Christine Janischek Informationsblatt: Objekten, Klassen, Attribute, Methoden, etc.



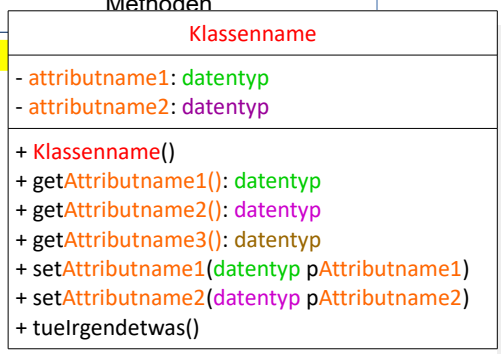
Klasse in UML

Eigenschaften
Attributname: Datentyp;

Datentypen: String, double, integer, boolean

Verhaltensweisen
Konstruktor & Methoden:

Hier: Getter- und Setter-Methoden



Detailansicht einer Klasse in UML

```php
class Klassennamen {

// Deklaration der Eigenschaften (Attribute)
    private $attributname1;
    private $attributname2;


// Standard (Default) Konstruktor
    public function __construct(){
     }

// Getter-Methoden: Ermittelt Eigenschaftswert eines Objektes
    public function getAttributname1 () {
        return $this->attributname1;
    }
    public function getAttributname2() {
        return $this->attributname2;
    }

// Setter-Methoden: Übermittelt Eigenschaftswert an das Attribut des Objektes
    public function setAttributname1( $pAttributname1) {
        $this->attributname1 = $pAttributname1;
    }
    public function setAttributname2( $pAttributname2) {
        $this->attributname2 = $pAttributname2;
    }

// Sonstige Methoden: Methoden die mehr können als nur er- und übermitteln
    public function tueIrgendetwas(){
            // Sonstige Methoden
    }

}
```

Erläuterungen	

Attributname,
Attributwert
 	
(syn. Variable)
	Attributnamen sind Eigenschaften die eine Sache bzw. ein Ding, näher beschreiben. Wird einer Eigenschaften ein konkreter Wert zugeordnet, handelt es sich um einen Attributwert. Sie werden in der Klasse selbst mit einem Datentyp und einem Namen ausgestattet. Dieser Vorgang heißt **Deklaration**. Ein oft verwendetes Synonym heißt Variable. Wird einer Eigenschaft ein konkreter Wert zugewiesen ergibt sich der Attriubutwert.
	
	Attributname:
	→ `bezeichnung`
	
	Attributwert:
	→ `"iPhone 6plus"`
	
	Beispiel:
	`Deklaration und Initialisierung (Wertezuweisung):`
	`private $bezeichnung = "iPhone 6plus";`
	
	`Reine Deklaration:`
	`private $bezeichnung;`
	
	`Reine Initialisierung:`
	`$this->$bezeichnung = "iPhone 6plus";`

Datentyp	Der Datentyp legt die Art des Wertes fest der verarbeitet werden soll und reserviert den Speicherplatz für den Wert. In PHP ist es i.d.R. nicht nötig den Datentyp explizit zu deklarieren. Der Typ eines Attributs bzw. einer Variablen wird automatisch durch den zugewiesenen Wert bestimmt. Es gibt jedoch Situationen, in denen das Erzwingen von Datentypen erforderlich ist. Wie in Java gibt es die gängigen primitiven Datentypen int, long, float, double, boolean und char. Auch der komplexe Datentyp string kann für Zeichenketten verwendet werden.
Zugriffsmodifikatoren	Werden für die Nutzung des Rechtesystems in objektorientierten Sprachen genutzt. Beispiele: **+ public:** steht für öffentlich, von außen sichtbar und zugänglich. **- private:** steht für privat, von außen nicht zugänglich (für Objekte anderer Klassen). **# protected:** steht für geschützt und erlaubt den Zugriff innerhalb der Klasse und von abgeleiteten Klassen.
Klasse	Die Klasse ist ein Muster, eine Vorlage die eine ganze Menge an Objekten mit ihren Eigenschaften und Verhaltensweisen beschreibt (definiert = deklariert). Beispiel:

<table>
<tr><td></td><td>Die Klasse Produkt beschreibt eine ganze Menge an möglichen Produkten die erzeugt werden können und denen dann Werte, also spezifische Eigenschaften zugewiesen werden können. Die Zuweisung expliziter Eigenschaftswerte erfolgt über die SET-Methode.

<pre>produkt1.setBezeichnung("iPhone 6plus");</pre>
Ist der Eigenschaftswert für die Bezeichnung an das Objekt der Klasse Produkt zugewiesen (übermittelt) kann er nachträglich vom System ermittelt werden. Die Ermittlung expliziter Eigenschaftswerte erfolgt über die GET-Methode.

<pre>produkt1.getBezeichnung();</pre></td></tr>
<tr><td>Objekt</td><td>Ein Objekt ist die Verwendung der Klasse für die konkrete Ausstattung einer Sache, eines „Dings“. Die Eigenschaften bekommen konkreten Werte und die Verhaltensweisen können auf die Dinge angewendet werden. Der Methodenaufruf (Verhaltensweise) erfolgt immer am konkreten Objekt (z.B. am produkt1)

Beispiel:
Ein Objekt der Klasse Produkt soll erzeugt werden, um auf die Verhaltensweisen des Objektes zugreifen zu können. Im Beispiel soll der Verkaufspreis für ein konkretes Produkt berechnet werden.

Erzeugung:
<pre>private Produkt produkt1 = new Produkt();</pre>
Verkaufspreis berechnen:
<pre>produkt1.vkpBerechnen();</pre></td></tr>
<tr><td>Methode</td><td>Methoden sind Verhaltensweisen die beschreiben wie etwas, eine Aufgabe, erledigt werden soll. Diese Beschreibung nennt sich auch Implementierung.

Beispiel:
Für die Klasse Produkt werden die SET- und GET-Methode für den Eigenschaftswert Bezeichnung und die Verhaltensweise für die Berechnung des Verkaufspreises implementiert.

SET-Methode:
<pre>	public function setBezeichnung($pBezeichnung) {
		$this->bezeichnung = $pBezeichnung;
	}</pre>
GET-Methode:
<pre>	public function getBezeichnung() {
		return $this->bezeichnung;
	}</pre>
Für die Methode vkpBerechnen():
<pre>public function vkpBerechnen() {
	$this->vkp = $this->hk + $this->gewinnzuschlag;
}</pre></td></tr>
</table>

4.2 Arbeitsblatt: Objektorientierung

Thema:	Einführung in PHP Autor: Christine Janischek Arbeitsblatt: Fundamentale Aspekte und Konzepte objektorientierter Softwareentwicklung

Sprache: Alphabet, Grammatik und Vokabular

Softwareentwicklung in der Praxis: Kennzeichnen Sie alle **wahren** Aussagen.

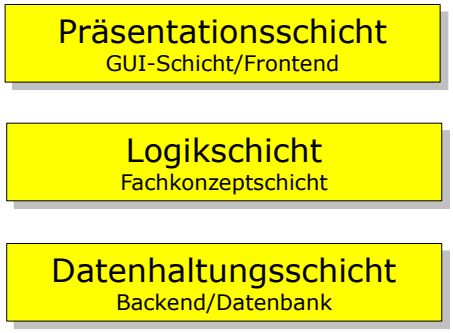
	Attributnamen werden in der Regel kleingeschrieben.
	Jedes Attribut repräsentiert eine Eigenschaft von Objekten einer Klasse.
	Der Zustand eines Objektes ist durch seine Werte bestimmt.
	Die Set-Methode (Setter) übermittelt einen Attributwert an das Klassenattribut des aktuellen Objektes einer Klasse.
	Klassennamen werden grundsätzlich großgeschrieben und stehen im Plural (Mehrzahl).
	Methoden mit Rückgabewert sind vom Typ „void“.
	Von einer Klasse kann man genau ein Objekte erzeugen.
	Der Dateiname einer Klasse muss mit dem Klassennamen übereinstimmen.
	int, String, double und boolean sind primitive Datentypen.
	Methoden werden in der Regel mit dem Zugriffsmodifikator „private“ deklariert.
	Der Konstruktor einer Klasse entspricht nicht dem Klassenname.
	Methoden sind Verhaltensweisen (Adjektive, Verben) und werden in objektorientierten Sprachen am Anfang großgeschrieben.
	Der Konstruktor einer Klasse verwendet den Zugriffsmodifikator „private“.
	Methoden beinhalten den Quellcode für Verhaltensweisen von Objekten einer Klasse.

4.3 Arbeitsblatt: Modell-View-Controller-Prinzip

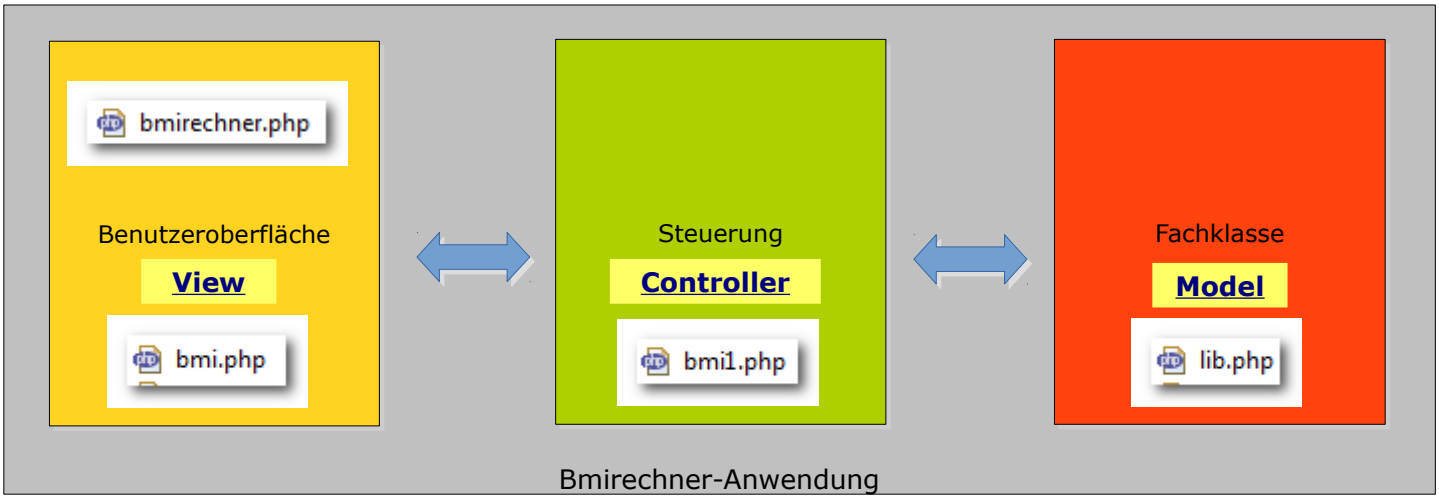
Thema:	Einführung in PHP Autor: Christine Janischek Arbeitsblatt: Model-View-Controller-Prinzip

Ein Grundprinzip in der Softwareentwicklung ist Heutzutage die Entkoppelung der Benutzeroberfläche und dem Fachkonzept. Der Vorteil dieses Vorgehens liegt in der guten Wartbar- und Erweiterbarkeit. Die Hardwareabhängigkeiten können in einer Schicht isoliert werden. Änderungen der Benutzungsoberfläche sind unabhängig vom Rest des Systems.

Es hat sich durchgesetzt, die Informationssysteme in einer Drei-Schichten-Architektur-Muster (three-tier architecture) zu modellieren.



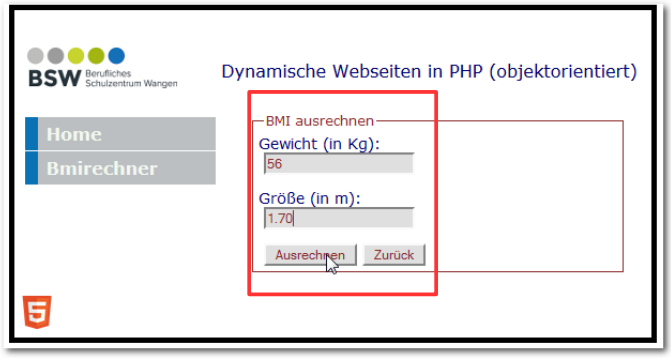
Das MVC-Muster (Model-View-Controller) baut auf dieser Idee auf. Damit können die Daten eines Datenmodells (Model) auf verschiedene Art und Weise auf der Benutzungsoberfläche dargestellt (Views) werden, Steuerungsobjekte (Controllers) übernehmen den kontrollierten Zugriff und dienen als Schutzschicht. Die Verarbeitung erfolgt in den Objekten der jeweiligen Fachklasse.



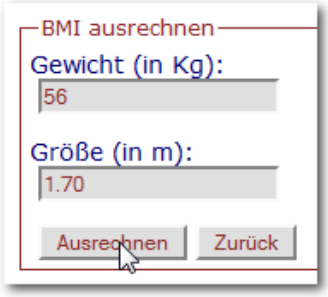
Arbeitsauftrag:

1. Erzeugen Sie ein neues Projekt um den → Bmirechner umzusetzen. Nutzen Sie dazu den folgenden Leittext.
2. Dokumentieren Sie welche einzelnen Schritte für die Umsetzung der View, des Controller und des Model notwendig sind.

4.4 Leittext: Bmirechner



Ergebnis der Übung: → bmirechner.php (View)



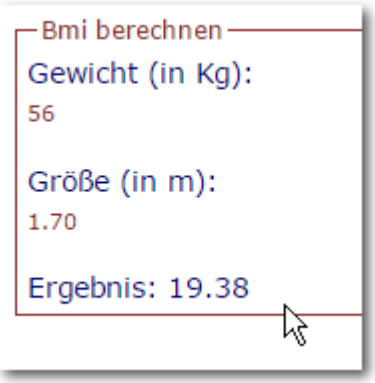
Eingabe-Formular: → bmi.php (View)

Bmirechner

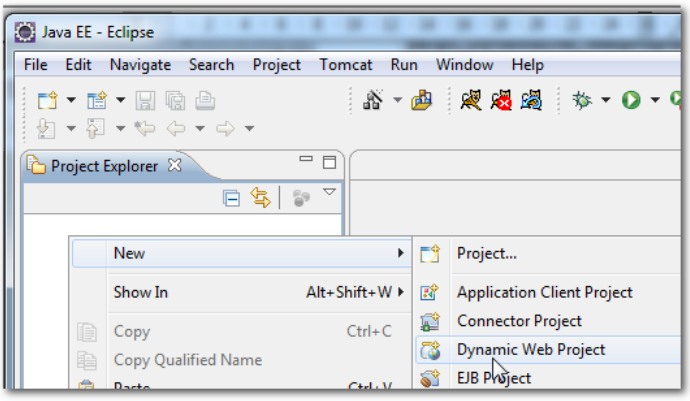
```
-$gewicht
-$groesse
-$ergebnis
+ __construct()

+ set_gewicht($pGewicht)
+ set_groesse($pGroesse)
+ set_ergebnis($pErgebnis)
+ get_gewicht()
+ get_groesse()
+ get_ergebnis()
+ berechne_bmi()
```

Reduzierte UML-Klasse: Bmirechner

Bibliothek: → lib.php (Model)



Steuerungs- und Ausgabe-Datei: → bmi1.php (Controller)



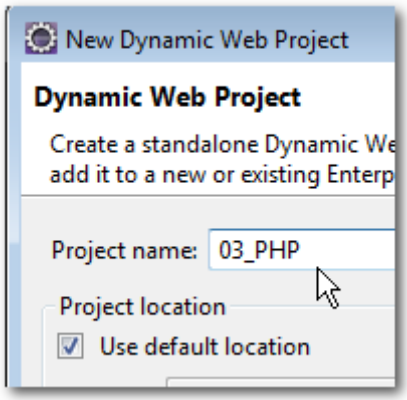
Neues Projekt erstellen.

Für die Dynamische Webseite benötigen wir ein neues Projektverzeichnis vom Typ

→ Dynamic Web Project

Klicken Sie dazu im linken Fenster von Eclipse für das Kontext-Menü (rechte Maustaste) und wählen Sie die Option → New → Dynamic Web Project aus.

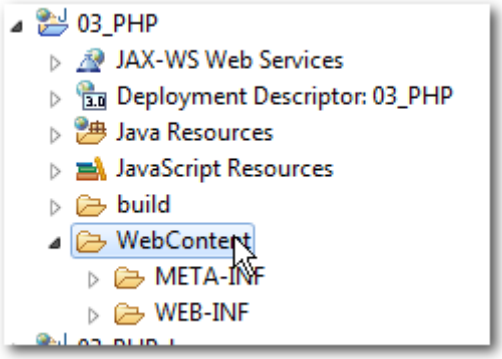
Hinweis:
Sie finden die Option auch → New → Other → Web → Dynamic Web Project.



Projektname festlegen.

Geben Sie den Namen für Ihr Projekt an und belassen Sie alle anderen Einstellungen. Schließen Sie den Vorgang mit einem Klick auf die Schaltfläche → Finish ab.

Aktuelles Projekt (vorher):



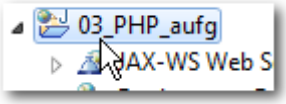
Öffnen Sie das Projektverzeichnis.

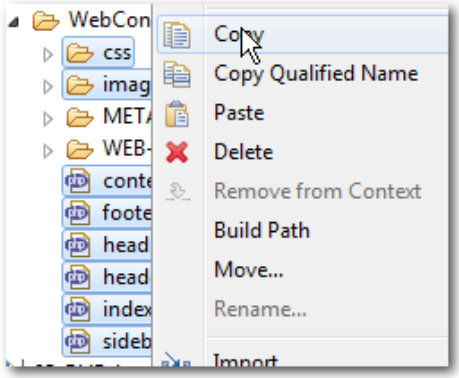
Klicken Sie dazu auf den kleinen blauen Pfeil links neben dem Projektname und wählen Sie mit einem Klick das WebContent-Verzeichnis aus.

→ WebContent

Hinweis:
In diesem Verzeichnis werden wir die Inhalte des Projektes platzieren.

Vorgegebenes Projekt:

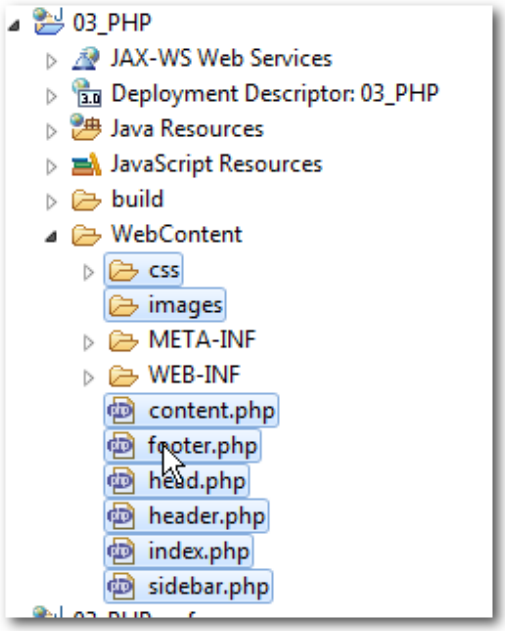




Prinzip der Wiederverwendung.

Wir verwenden ein vorgegebenes Seitenlayout.

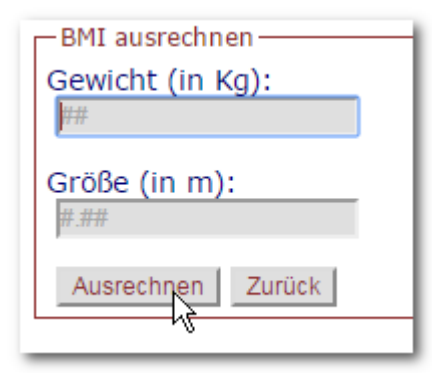
Öffnen und kopieren Sie dazu die Inhalte aus dem Projekt *03_PHP_aufg* in das WebContent-Verzeichnis des aktuellen Projektes.

	Aktuelles Projekt (nachher):  Für die Implementierung der Anwendung folgen Sie weiter dem Leittext.

View

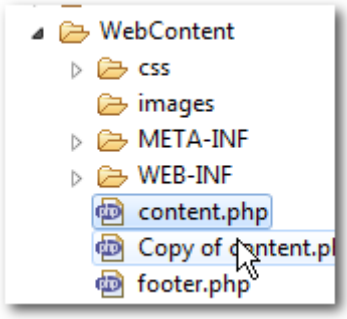
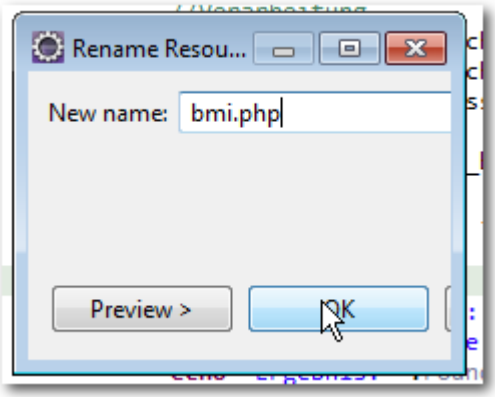
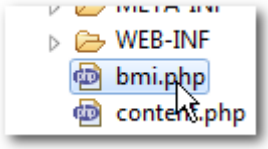
Formular:

→ Bmirechner
→ Bmirechner (opt)
→ Darlehensrechner
→ Notenrechner
→ Rabattrechner
→ Taschenrechner



Eingabe-Formular erzeugen: bmi.php

→ zurück zum Arbeitsblatt

	*Formulardatei erzeugen.* Kopieren Sie dazu die Datei content.php mit STRG + C und fügen Sie die Datei STRG +V ein. Benennen Sie die Datei um. Klicken Sie die Kopie an und wählen Sie dazu im Kontext-Menü die Option → Rename.
	*Dateiname festlegen.* Geben Sie den Dateiname ein und klicken Sie auf die Schaltfläche OK. Öffnen Sie die Datei mit einem Doppelklick auf den Dateinamen: 

```
<form name="bmirechnerformular"
      method="post"     action="bmi1.php">

...
</form>
```

Quellcode für das Formular einbetten.

Fügen Sie dazu die Form-Box in die Content-Box ein. Definieren Sie die Eigenschaften für den Namen, Methode und Aktion, wie angegeben.

Name
Bezeichner für das Formular.

Method
In der Regel wird die Methode → post verwendet.

- Post: unterdrückt die Angabe der Formparameter in der URL, die Länge der URL ist nicht begrenzt und der Nutzer kann auf die Auswertung nicht verlinken.
- Get: zeigt die Formparameter in der URL an, die Länge der URL ist auf 3000 Zeichen begrenzt und der Nutzer kann auf die Auswertung verlinken.

Action
Enthält den Dateinamen der ereignissteuernden PHP-Datei → Controller. Die Datei wird aufgerufen, wenn der Nutzer die Schaltfläche vom Typ → submit im Formular anklickt.

```
<fieldset>
...
</fieldset>
```

Fieldset definieren.

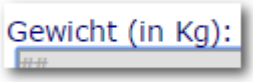
Fügen Sie in die Form-Box die Fieldset-Box ein. Mit HTML5 ist es sinnvoll Formulare in sinngemäße Blöcke zusammenzufassen. Sie dienen der Gruppierung zusammengehörender Formularkomponenten.

View

```
<legend>BMI ausrechnen</legend>
```

Legende definieren.

Fügen Sie in die Fieldset-Box die Legend-Box ein. Definieren Sie für die Legende den Titel.



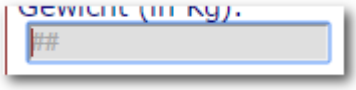
View

```
<label for="tfGewicht">
      Gewicht (in Kg):
</label> <br />
```

Label-Komponente definieren.

Fügen Sie unterhalb der Legende und innerhalb der Fieldset-Box die Label-Komponente ein. Die Label-Komponente dient als Bezeichner für ein Eingabefeld, z.B einem Textfeld. Für die Eigenschaft → for wird der Komponentenname des Eingabefeldes angegeben.

```
<input type="text" name="tfGewicht"
       id="tfGewicht" placeholder="##"
       required="required"
       autofocus="autofocus" /><br /><br />
```



View

Eigenschaften eines Textfeldes:

type
die Eigenschaft legt fest, auf welche Daten (Art der Erfassung und Datentyp) erfasst werden sollen.

Name
Bezeichner für die Komponente.

id
Wird gesetzt um das Feld, falls notwendig im Stylesheet individuell formatieren zu können.

placeholder
Enthält einen Eingabehinweis, um den Benutzer bei der korrekten Eingabe der Daten zu unterstützen.

required
Ist ein Flag (boolscher Wert) das gesetzt wird, wenn die Dateneingabe durch den Benutzer zwingend erforderlich ist. Neu in HTML5.

autofocus
Ist ein Flag (boolscher Wert) sorgt dafür, dass die Komponente beim laden der Seite fokussiert wird → der Curser blinkt. Neu in HTML5.

Textfeld-Komponente definieren.

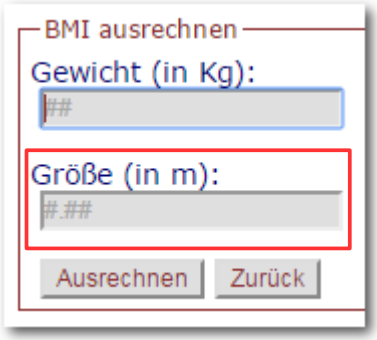
Fügen Sie unterhalb der Label-Komponente und innerhalb der Fieldset-Box die Textfeld-Komponente ein.

Ein Textfeld ist eine Input-Komponente (Eingabefeld) vom Typ → text.

Weitere Beispiele für Eingabe-Komponenten:
https://wiki.selfhtml.org/wiki/HTML/Formulare

Es gibt weitere Eingabefelder:

Programming Rules			
Bezeichnung	Typ (type)	Präfix	Beispiel
Textfeld	text	tf	tfGewicht
Passwortfeld	password	tf	tfPasswort
Radiobutton	radio	rb	rbGeschlecht
Checkbox	checkbox	cb	cbSalami
Button	submit	bt	btAusrechnen
Button	button	bt	btZurueck



Komponenten für die Eingabe der Größe einbetten.

Fügen Sie wie zuvor für das Gewicht die notwendigen Komponenten für die Größe ein.

name
→ tfGroesse

id
→ tfGroesse

placeholder
→ #.##

required
→ required

```
<input type="submit"
       value="Ausrechnen" name='bmiAusrechnen'/>
```

```
<?php echo"<input type='button'
       value='Zur&uuml;ck'
              onClick='history.back()' />" ?>
```

Schaltflächen einbetten.

Fügen Sie unterhalb der Textfeld-Komponente und innerhalb der Fieldset-Box die Button-Komponenten für die Schaltfläche → Ausrechnen und → Zurück ein.

Mit dem Funktionsaufruf
`<?php echo "..." ?>`
wird die Ausgabe (→ `echo`) der Schaltfläche → zurück erzeugt. Auf diese Weise können wir via PHP das Ereignis über die Eigenschaft → `onClick` einbetten.

Methodenaufruf → `history.back()`:
Ist eine Javascript-Funktion. Die Methode ladet die vorhergehende URL aus der Historie des aktuellen Browserfensters.

Im Web Browser

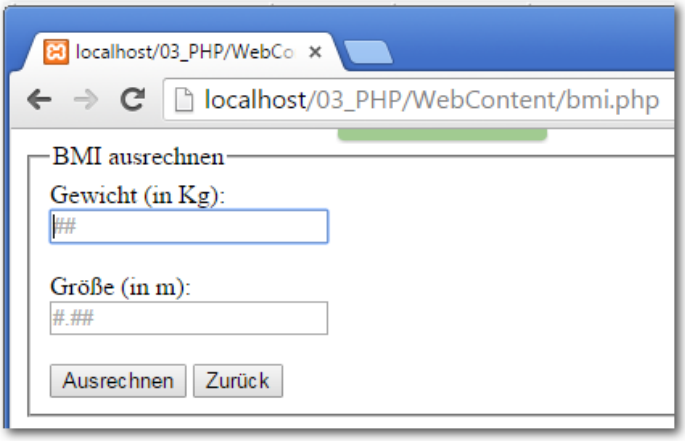
Testen Sie das Formular.

Öffnen Sie dazu den Internetbrowser und geben Sie den Pfad zur gerade erstellen Datei an.

Geben Sie dazu den Pfad ein.

http://localhost/03_PHPnoop/bmi.php

Herzlichen Glückwunsch Sie haben Ihr Formular erstellt.



View: bmi.php

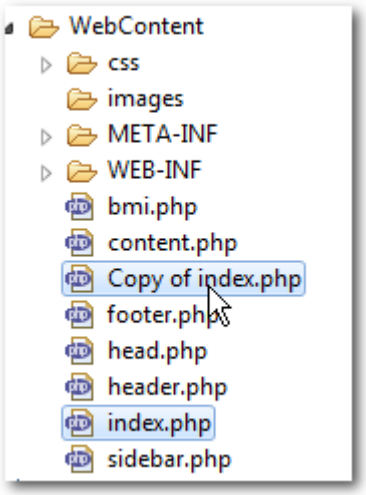
Betten Sie nun das Formular in Ihr dynamisches Layout ein. Folgen Sie dazu dem Leittext.

View

Box-Modell:
→ Bmirechner
→ Bmirechner (opt)
→ Darlehensrechner
→ Notenrechner
→ Rabattrechner
→ Taschenrechner

Eingabe-Formular einbetten: bmirechner.php

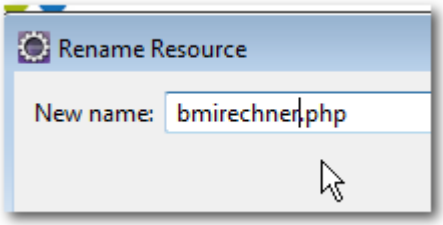
→ zurück zum Arbeitsblatt

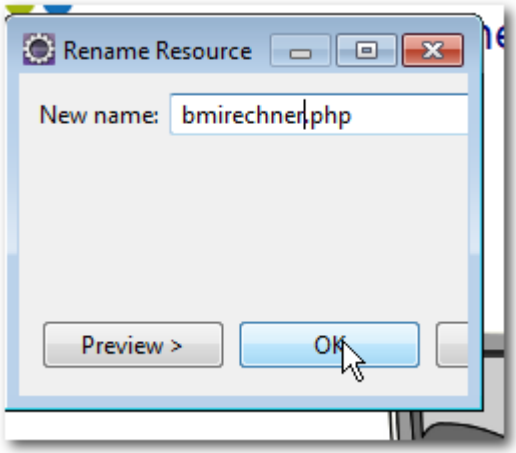


Formulardatei in das bestehende Box-Model der Seite einbetten.

Kopieren Sie dazu die Datei → index.php mit STRG + C und fügen Sie die Datei STRG +V ein.

Benennen Sie die Datei um. Klicken Sie den Dateinamen im linken Frame an und wählen Sie dazu im Kontext-Menü (rechte Maustaste) die Option → Rename.

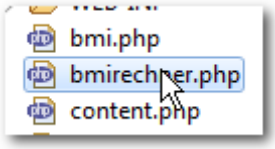


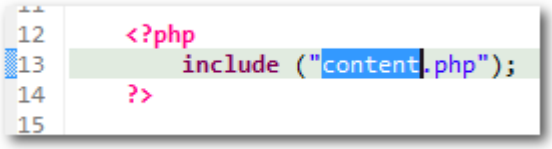


Dateiname festlegen.

Geben Sie den Dateiname ein und klicken Sie auf die Schaltfläche OK.

Öffnen Sie die Datei mit einem Doppelklick auf den Dateinamen:





```
12      <?php
13          include ("content.php");
14      ?>
15
```

```
<?php
    include ("bmi.php");
?>
```

Formular referenzieren.

Ersetzen Sie in der Datei → bmirechner.php die Referenz durch die Referenz auf die Formular-datei → bmi.php

View: sidebar.php

```
<ul>
      <li><a href="index.php">Home</a></li>
      <li><a href="bmirechner.php"
      target="_parent">Bmirechner</a></li>
</ul>
```

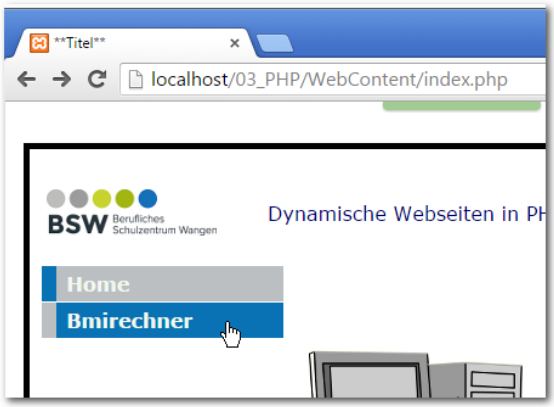
Navigation erweitern.

Öffnen Sie die Datei → sidebar.php. Erweitern Sie die ungeordnete Liste innerhalb der Side-bar-Box um die Referenz (das Listenelement) auf die bmirechner.php-Datei.

Im Web Browser

Testen Sie die Anwendung.

Öffnen Sie dazu den Internetbrowser und ge-ben Sie den Pfad zur gerade erstellen Datei an. Geben Sie dazu den in der Grafik angezeigten Pfad ein.

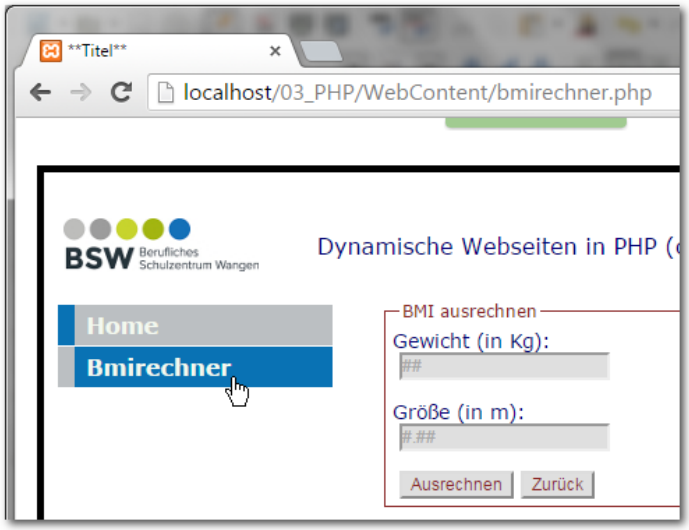


View: index.php

http://localhost/03_PHP/WebContent/index.php

Klicken Sie in der Navigationsleiste auf die Option Bmirechner.

Herzlichen Glückwunsch Sie haben den Rechner erfolgreich in die Seite eingebettet.

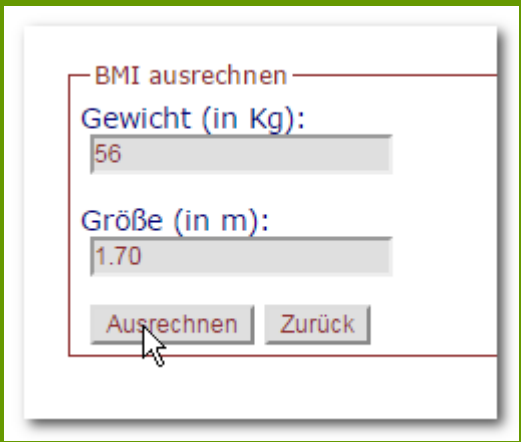


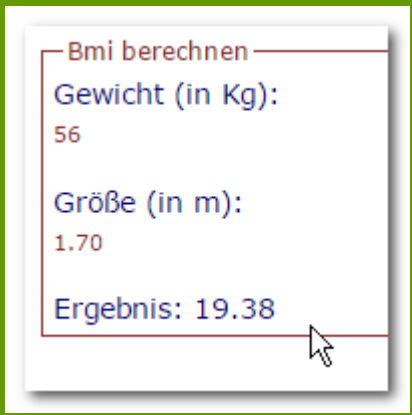
Der Rechner rechnet noch nicht! Folgen Sie dazu weiter dem Leittext.

Controller

Ereignissteuerung:

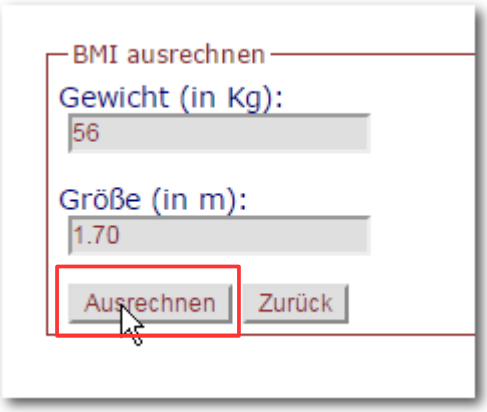
→ Bmirechner
→ Bmirechner (opt)
→ Darlehensrechner
→ Notenrechner
→ Rabattrechner
→ Taschenrechner





Ereignissteuerung: bmi1.php

→ zurück zum Arbeitsblatt



Funktionalitäten einbetten.

In den nächsten Schritten kümmern wir uns um die Funktionalität → Ausrechnen. Das Ereignis, also die Auswertung der Eingaben durch den Benutzer sollen dann erfolgen, wenn die Schaltfläche → Ausrechnen angeklickt wird.

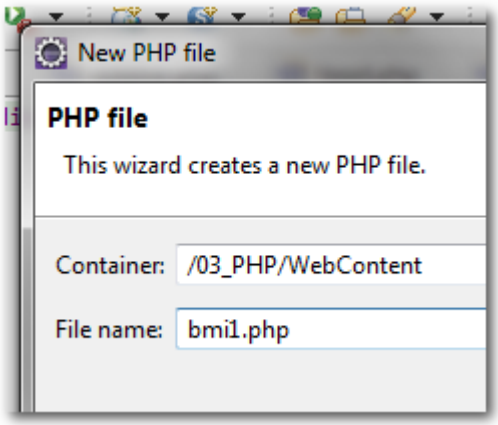
Diese Auswertung kann eine Seitenbeschreibungssprache HTML nicht leisten. Wir nutzen deshalb die Programmiersprache PHP dafür.

Die Controllerdatei fehlt noch!

Im Moment rufen Wir mit dem Klick auf die Schaltfläche → Ausrechnen eine Datei → bmi1.php auf (siehe Form-Action in der Datei → bmi.php):

```
<form name="bmirechnerformular" method="post" action="bmi1.php">
    <fieldset>
        <legend>BMI ausrechnen</legend>
```

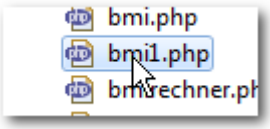
Diese Datei existiert aber noch nicht, deshalb erscheint auch die nebenstehende Fehlermeldung.



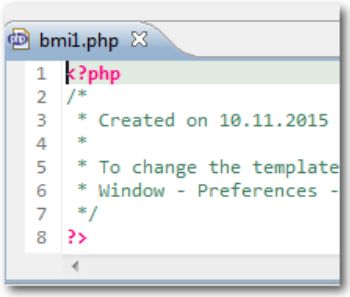
Controllerdatei einbetten.

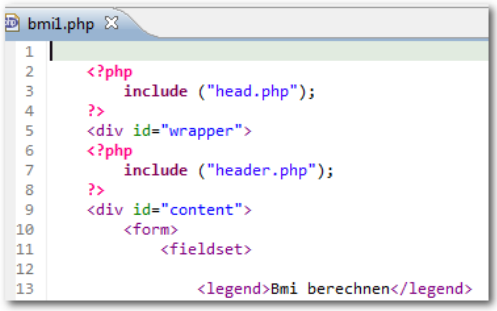
Wir werde diese Datei nun erstellen und das EVA-Prinzip (**E**ingaben lesen, **v**erarbeiten und **a**usgeben) mit Hilfe von PHP anwenden.

Erzeugen Sie dazu die Datei → bmi1.php.



Öffnen Sie diese Datei mit einem Klick auf den Dateinamen.





Eingabehilfe:

```
<?php
	include ("head.php");
?>
<body>
<div id="wrapper">
	<?php
		include ("header.php");
	?>
	<div id="content">
		<form>
			<fieldset>
```

Grundgerüst unseres Layouts einbetten.

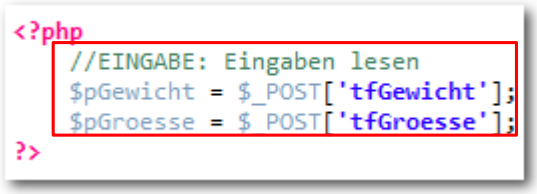
Ersetzen Sie dazu den angegebenen Quellcode durch den nebenstehend angezeigten Quellcode.

Integrieren Sie dazu die Dateien:
→ head.php
→ header.php

und die Boxen:
→ wrapper
→ content
→ form
→ fieldset
→ legend

Hinweis:
Sie müssen die Content-, Form- und Fieldset-Box im Anschluss an die folgenden Implementierungen korrekt schließen!

```
            <legend>Bmi berechnen</legend>
```



PHP-Tag

```
<?php
        …
?>
```

EINGABE: Eingaben lesen

```
$pGewicht = $_POST['tfGewicht'];
$pGroesse = $_POST['tfGroesse'];
```

Eingabe: Eingaben Lesen.

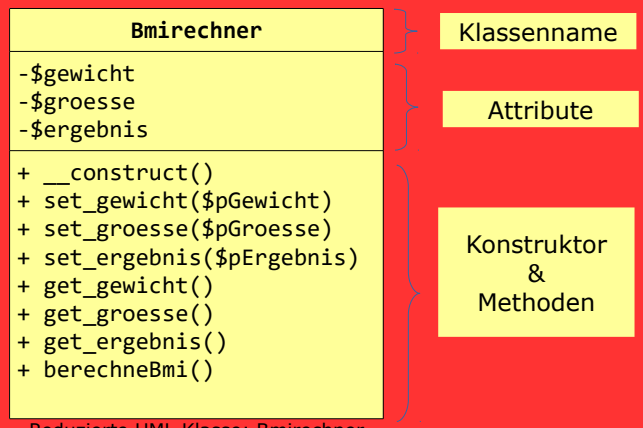
Wir lesen nun im Ersten Schritt die Eingabewerte aus den Texteingabefeldern → tfGewicht, und → tfGroesse und übernehmen diese Werte in die PHP-Parameterattribute (Variablen) → pGewicht und → pGroesse.

Öffnen Sie dazu unterhalb der Legenden-Box eine neues PHP-Tag und übernehmen Sie den nebenstehenden Quellcode.

Die eigentliche Verarbeitung der eingegebenen Daten werden wir objektorientiert realisieren. Damit ergibt sich die Notwendigkeit eine Klasse, deren Eigenschaften und Verhaltensweisen zu definieren. Folgen Sie dazu weiter dem Leittext.

Model

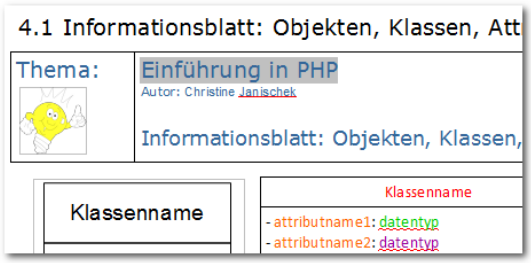
Fachklassen:
→ Bmirechner
→ Bmirechner (opt)
→ Darlehensrechner
→ Notenrechner
→ Rabattrechner
→ Taschenrechner



Reduzierte UML-Klasse: Bmirechner

UML-Klasse → Bmirechner: lib.php

→ zurück zum Arbeitsblatt

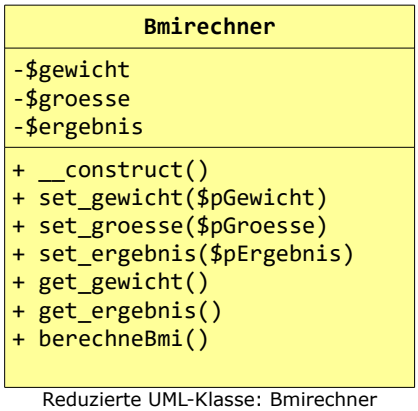


Verarbeitung

Nutzen Sie das Informationsblatt. Klären Sie die Begriffe und informieren Sie sich über das Grundgerüst einer Klasse in PHP.

Wir implementieren die Fachklasse → Bmirechner, indem wir sie mit dem benötigten Quellcode ausstatten.

Entsprechend den Vorgaben (Anforderungen) der nebenstehend angezeigten UML-Klasse, werden wir das in den kommenden Schritten tun.



Reduzierte UML-Klasse: Bmirechner

Hinweis:
PHP ist eine Sprache mit dynamischer und impliziter Typiserung. Um einen Datentyp expizit zu setzten können Sie die Funktion → settype(var_name,

Fachklasse implementieren.

Egal in welcher Programmiersprache wir heutzutage programmieren, das Prinzip lautet nahezu immer → Objektorientierung. Im Großen und Ganzen geht es dabei darum Ordnung zu halten, d. h. den Quellcode auf eine bestimmte Art und Weise zu strukturieren.

Das Prinzip stellt u. a. sicher, dass wir die Anwendung später problemlos erweitern können und Quellcodebestandteile wiederverwenden können.

Wir halten uns von Anfang an, an genau dieses Prinzip und werden es in den nächsten Schrit-

var_type) nutzen.

ten erstmals anwenden.

```
//####VERARBEITUNG:#####
//################################################
//Eine Klasse: Vorlage für eine ganze Menge von Objekten
    class Bmirechner {
        //Eigenschaften: Attribute
        private $gewicht;
        private $groesse;
        private $ergebnis;

    }
```

Fachklasse: Klasse und Attribute deklarieren.

Erweitern Sie dazu den gerade eingefügten Quellcode unterhalb des bereits implementierten Quellcodes, um den nebenstehenden Quellcode.

Im Ersten Schritt werden wir die Klasse als „inner class" implementieren. Später wird es sinnvoll sein, die Klasse in eine extra Datei auszulagern.

Deklaration einer Klasse

```
class Bmirechner {
        ...
}
```

Deklaration der Attribute (Eigenschaften)

```
//Eigenschaften: Attribute
private $gewicht;
private $groesse;
private $ergebnis;
```

```
/*Standard Konstruktor (ohne Parameter und leer):
 * Zum erzeugen von Objekten einer Klasse*/
public function __construct(){}
```

Standard-Konstruktor einer PHP-Klasse

```
public function __construct(){}
```

Fachklasse: Konstruktor deklarieren.

Mit Hilfe des Konstruktors können wir später beliebig viele Objekte dieser Klasse erzeugen.

Erweitern Sie dazu den gerade eingefügten Quellcode unterhalb der deklarierten Attribute, wie nebenstehend angezeigt.

Getter:

```
/*Getter: Get-Methoden ermitteln den Wert einer
Eigenschaft eines Objektes der Klasse*/
public function get_gewicht() {
    return $this->gewicht;
}
```

Setter:

Fachklasse: Get- und Set-Methoden deklarieren.

Die Get- und Set-Methoden dienen später zur Über- und Ermittlung von Attributwerten eines Objektes.

Erweitern Sie dazu den gerade eingefügten Quellcode unterhalb des Konstruktors, wie nebenstehend angezeigt.

Getter:

```
public function get_gewicht() {
        return $this->gewicht;
```

```php
/*Setter: Get-Methoden übermitteln den
 * Eigenschaftswert an ein Objektes der Klasse*/
public function set_gewicht($pGewicht) {
    $this->gewicht = $pGewicht;
}
```

```
}
```

Setter:

```php
public function set_gewicht($pGewicht) {
        $this->gewicht = $pGewicht;
}
```

Implementieren Sie auch den Quellcode für die Getter und Setter der übrigen Attribute:
→ groesse
→ ergebnis

```php
//Verhaltensweisen: Sonstige Methoden
public function berechne_bmi() {
    $mBmi = $this->get_gewicht()/($this->get_groesse()
                * $this->get_groesse());
    $this->set_ergebnis($mBmi);
```

```
}
```

Fachklasse: Methode für die Berechnung.

Erweitern Sie dazu den gerade eingefügten Quellcode unterhalb Getter und Setter, wie nebenstehend angezeigt.

Verhaltensweisen: Sonstige Methoden

```php
public function berechne_bmi() {

        $mBmi = $this->get_gewicht()/
                ($this->get_groesse() *
                        $this->get_groesse());

        $this->set_ergebnis($mBmi);
}
```

Die Fachklasse ist nun komplett! Kontrollieren Sie selbst ob die Klasse mittels der geschwungenen Klammer → } geschlossen wird.

```php
/*Es wird ein neues Objekt der
 * Klasse bmirechner erzeugt*/
$dieDaten = new Bmirechner();
```

Ein Objekt der Klasse Bmirechner erzeugen

```php
$dieDaten = new Bmirechner();
```

Objekt einer Klasse erzeugen.

Um die Verarbeitung der Daten einleiten zu können müssen wir ein Objekt der Klasse erzeugen.

Erweitern Sie dazu den gerade eingefügten Quellcode unterhalb der eingefügten Klasse, wie nebenstehend angezeigt.

```php
/*Methodenaufruf: Die Eingabewerte werden an
 * die Attribute des Objektes übermittelt*/
$dieDaten -> set_gewicht($pGewicht);
$dieDaten -> set_groesse($pGroesse);
```

Übermittlung der Eingabewerte

Eingabewerte übermitteln.

In Objektorientierten Sprachen erfolgen Methodenaufrufe am Objekt der jeweiligen Klasse. Im vorliegenden Fall werden die Eingabewerte an das Objekt übermittelt, damit im Anschluss daran die Berechnung erfolgen kann.

```
$dieDaten -> set_gewicht($pGewicht);
$dieDaten -> set_groesse($pGroesse);
```

Erweitern Sie dazu den gerade eingefügten Quellcode unterhalb des erzeugten Objektes.

```
/*Methodenaufruf: Die Berechnung wird für
 * das Objekt durchgeführt*/
$dieDaten -> berechne_bmi();
```

Medodenaufruf: Berechnung des BMI

```
$dieDaten -> berechne_bmi();
```

Berechnende Methode am Objekt aufrufen.

Erweitern Sie dazu den gerade eingefügten Quellcode unterhalb der aufgerufenen Setter.

```
//Ergebis ermitteln und zuweisen
$ausgabe =  $dieDaten -> get_ergebnis();
```

Ermittlung und Zuweisung eines Wertes

```
$ausgabe =   $dieDaten -> get_ergebnis();
```

Ergebnis ermitteln und zuweisen.

Mit dem Methodenaufruf des Getters wird das Ergebnis ermittelt und einem Attribut (Variablen) → $ausgabe zugewiesen.

Erweitern Sie dazu den gerade eingefügten Quellcode unterhalb der Berechnung.

```
//####AUSGABE#######
echo "Gewicht (in Kg): <h5>"
        .$dieDaten -> get_gewicht()."</h5>";
echo "Gr&ouml;&szlig;e (in m): <h5>"
        .$dieDaten -> get_groesse()."</h5>";
echo "Ergebnis: ".round($ausgabe,2);
```

Eingabehilfe:

```
echo "Gewicht (in Kg): <h5>"
       .$dieDaten -> get_gewicht()."</h5>";
echo "Gr&ouml;&szlig;e (in m): <h5>"
       .$dieDaten -> get_groesse()."</h5>";
echo "Ergebnis: ".round($ausgabe,2);
```

Ausgabe erzeugen.

Wir erzeugen nun die gewünschte Ausgabe auf der Benutzeroberfläche.

Erweitern Sie dazu den gerade eingefügten Quellcode unterhalb des ermittelten Ergebnisses.

```
.round($ausgabe,2)
```

mit dem Methodenaufruf wird die Ausgabe auf 2 Kommastellen gerundet bevor die Ausgabe auf der Benutzeroberfläche erfolgt.

Kontrollieren Sie ob das PHP-Tag abschließend mittels → `?>` geschlossen wird.

```
            </fieldset><!--Fieldset-Box schließen-->
        </form><!--Form-Box schließen-->
    </div><!--Content-Box schließen-->

    <?php
        include ("sidebar.php");
    ?>

    <?php
        include ("footer.php");
    ?>
    </div><!--Wrapper-Box schließen-->
</body>
</html>
```

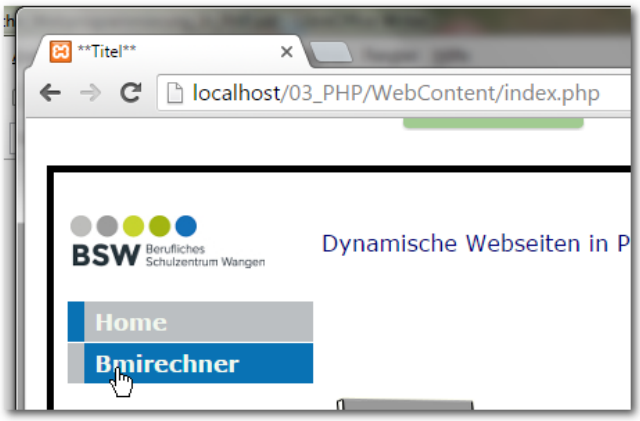
Boxen schließen und Dateien einbetten.

Schließen Sie alle noch offenen Boxen in der richtigen Reihenfolge und inkludieren Sie die Navigation und die Fußzeile.

Eingabehilfe:

```
        </fieldset><!--Fieldset-Box schließen-->
            </form><!--Form-Box schließen-->
        </div><!--Content-Box schließen-->

        <?php
            include ("sidebar.php");
        ?>

        <?php
            include ("footer.php");
        ?>
        </div><!--Wrapper-Box schließen-->
</body>
</html>
```

Im Web Browser



View: index.php

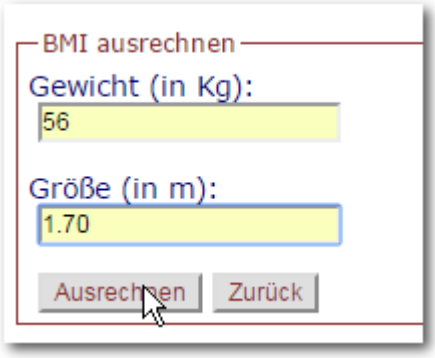
Zwischenergebnis testen.

Öffnen Sie dazu den Internetbrowser und geben Sie den Pfad zur gerade erstellen Datei an. Geben Sie dazu den in der Grafik angezeigten Pfad ein.

http://localhost/03_PHP/WebContent/index.php

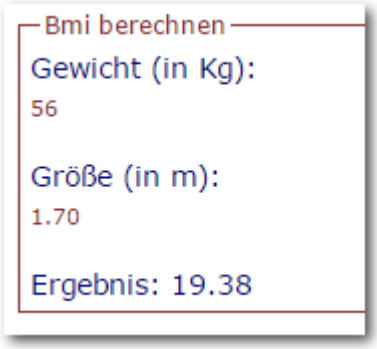
Klicken Sie in der Navigationsleiste auf die Option Bmirechner.

Der Rechner sollte nun rechnen!



Testdaten für den Anwendungsfall.

Geben Sie die nebenstehenden Daten ein und berechnen Sie das Ergebnis.



Testergebnis kontrollieren.

Kontrollieren Sie die Ausgabe.

Herzlichen Glückwunsch der Rechner rechnet schon.

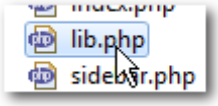
Die Struktur des Quellcodes ist allerdings noch suboptimal. Folgen Sie für die Optimierung dem Leittext.

Klasse auslagern.

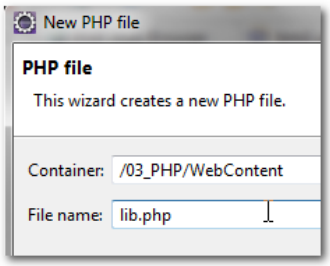
Die Fachklasse Bestandteil des Modells und gehört nicht zum Controller. → siehe MVC-konzept.

Außerdem wollen wir künftig unsere Seite um weitere Rechner (Bmirechner, Taschenrechner, Notenrechner,...) erweitern möchten und diese auf unterschiedlichen Unterseiten nutzen können. Deshalb lagern wir den Quellcode der Fachklasse in eine extra Datei aus.

Erzeugen Sie dazu eine neue Datei
→ lib.php



Diese Datei wird unsere eigene PHP-Bibliothek werden! Öffnen Sie diese Datei mit einem Klick auf den Dateinamen.



Controller: Klasse ausschneiden
→ bmi1.php

```
        //####VERARBEITUNG#######
        //################################################
        //Eine Klasse: Vorlage für eine ganze Menge von Objekten
        class Bmirechner {

            //Eigenschaften: Attribute
            private $gewicht;
            private $groesse;
            private $ergebnis;

            /*Standard Konstruktor (ohne Parameter und leer):
             * Zum erzeugen von Objekten einer Klasse*/
            public function __construct(){}
```

...
Bibliothek: Klasse einfügen
→ lib.php

```
<?php
//####VERARBEITUNG#######
//################################################
//Eine Klasse: Vorlage für eine ganze Menge von Objekten
class Bmirechner {

    //Eigenschaften: Attribute
    private $gewicht;
    private $groesse;
    private $ergebnis;
```

...

Fachklasse ausschneiden und in Bibliothek einfügen.

Markieren Sie in der Datei → bmi1.php die den Quellcode für die ganze Klasse und schneiden Sie ihn aus:→ STRG + X

Wechseln Sie in die Datei → lib.php und fügen Sie den Quellcode innerhalb des PHP-Tags ein: STRG + V

Speichern Sie die Änderungen in beiden Dateien (→ bmi1.php und → lib.php).

Bibliothek im head referenzieren.

```
<?php
    include("lib.php");
?>
```

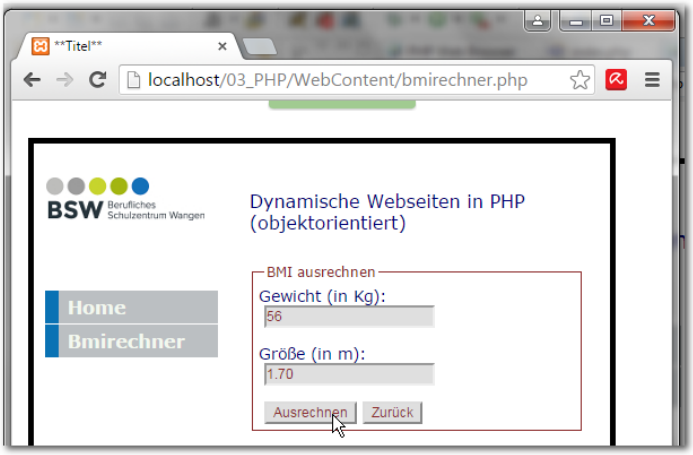
Eingabehilfe:

```
<?php
        include("lib.php");
?>
```

Damit wir das Bmirechner-Objektund dessen Eigenschaften und Verhaltensweisen in der Datei → bmi1.php trotz Auslagerung nutzen können, müssen wir ähnlich, wie beim Stylesheet, sicherstellen, dass wir auf die Datei → lib.php verweisen.

Ergänzen und speichern Sie dazu im

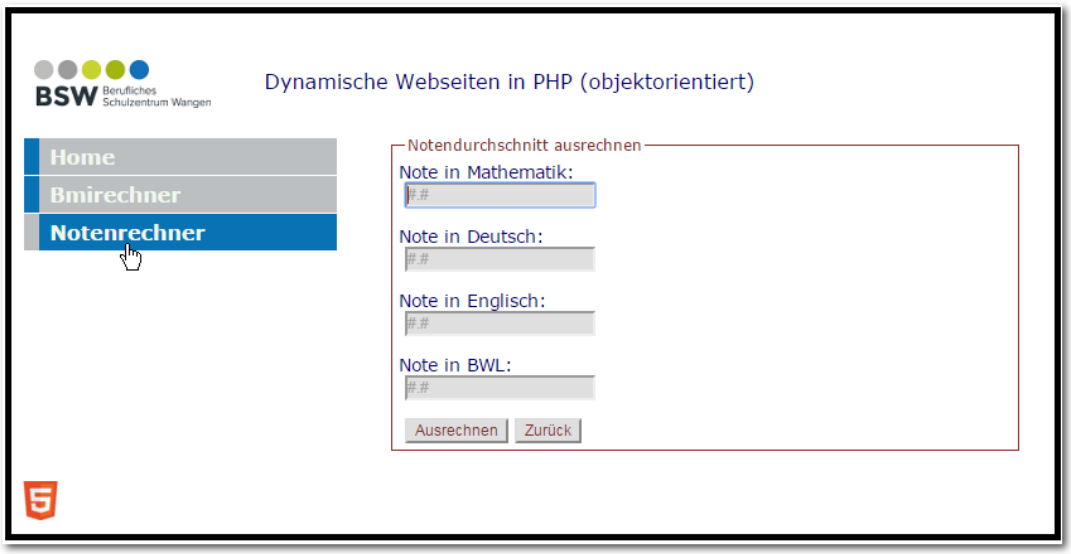
```
<head>
...
</head>
```

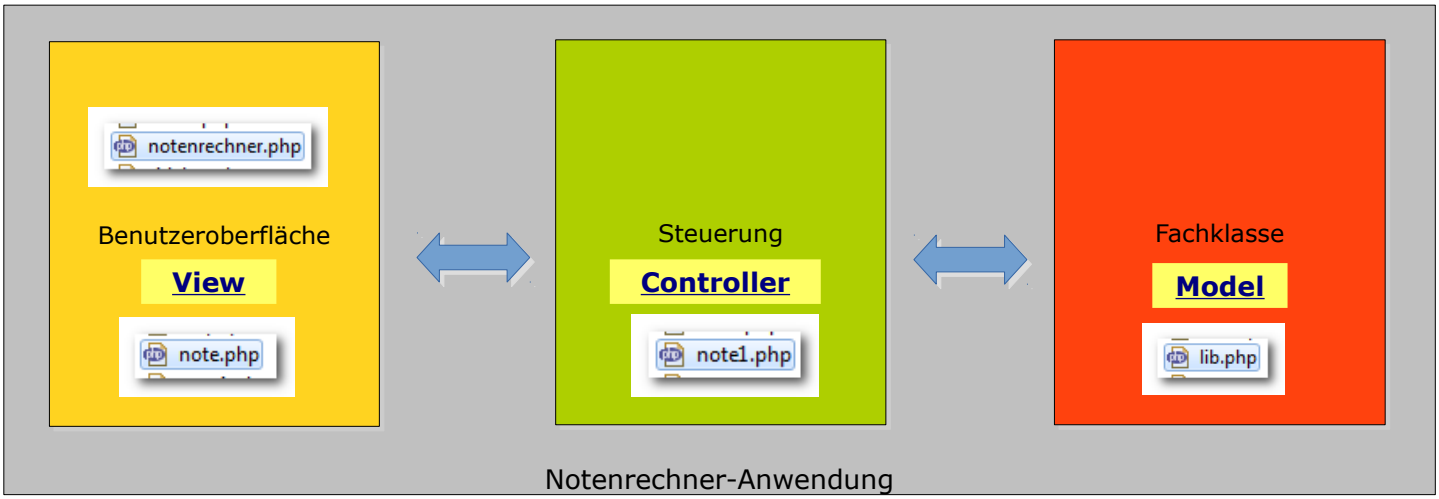
	den notwendigen PHP-include-Befehl, wie nebenstehend angezeigt. Hinweis: Bedenken Sie, künftig können Sie alle Klassen in die Bibliothek (lib.php) auslagern.
	Testen Sie die Anwendung erneut. An der Funktionalität sollte sich nichts geändert haben.  Mit der vorgenommen Auslagerung können wir nun die berechnende Methode → berechne_bmi(), immer wieder und an unterschiedlichen Stellen, nutzen. **Herzlichen Glückwunsch Sie haben ihre eigene PHP-Bibliothek geschaffen/erweitert.**

5 Notenrechner: Übung Formulare auswerten

5.1 Arbeitsblatt: Notenrechner

Thema:	Übung: Formulare auswerten Autor: Christine Janischek Arbeitsblatt: Notenrechner

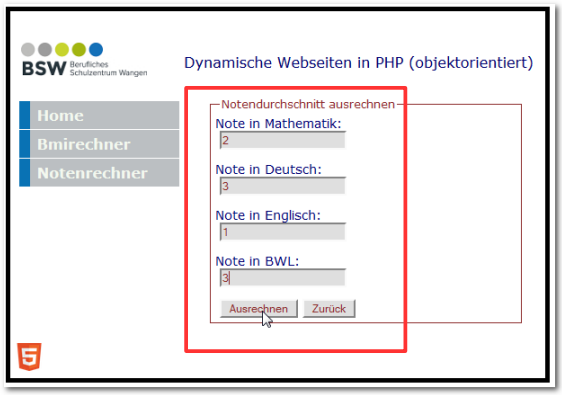




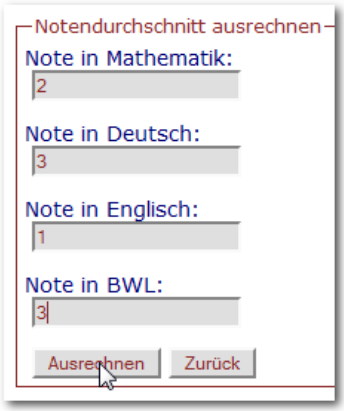
Arbeitsauftrag:

1. Erzeugen Sie ein neues Projekt um den → Notenrechner umzusetzen. Nutzen Sie dazu den folgenden Leittext.
2. Dokumentieren Sie welche einzelnen Schritte für die Umsetzung der View, des Controllers und des Models notwendig sind.

5.2 Leittext: Notenrechner



Ergebnis der Übung: → notenrechner.php (View)

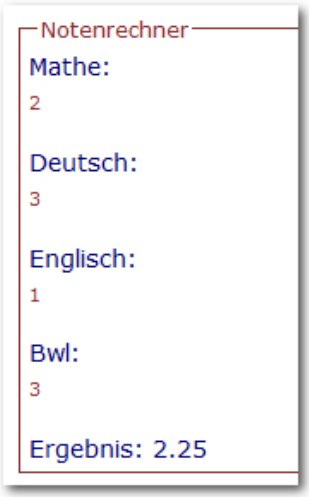


Eingabe-Formular: → note.php (View)

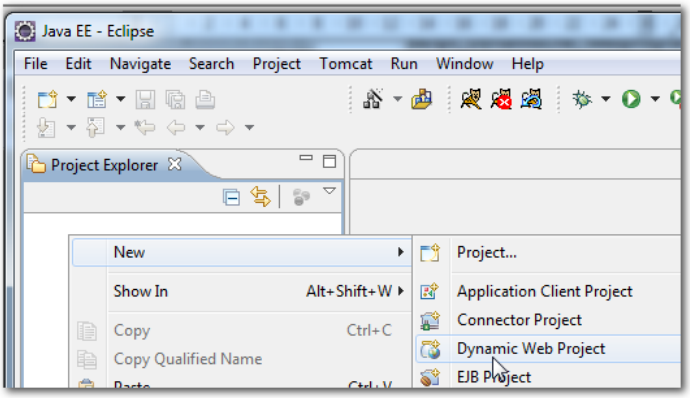
Notenrechner
-$mathe -$deutsch -$englisch -$bwl -$ergebnis
+ __construct() + set_mathe($pMathe) + set_deutsch($pDeutsch) + set_englisch($pEnglisch) + set_bwl($pBwl) + set_ergebnis($pErgebnis) + get_mathe() + get_deutsch() + get_englisch() + get_bwl() + get_ergebnis() + berechneDurchschnitt()

Reduzierte UML-Klasse: Notenrechner

Bibliothek: → lib.php (Model)



Ausgabe-Datei: → note1.php (Controller)



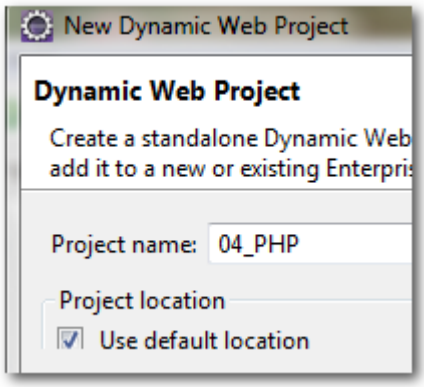
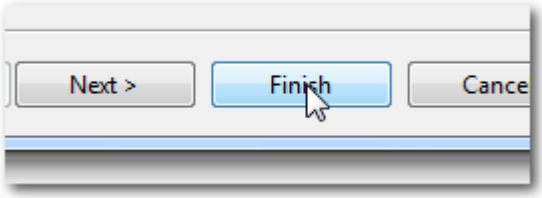
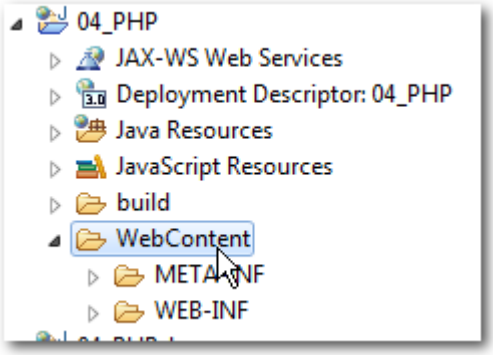
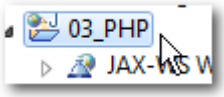
Neues Projekt erstellen.

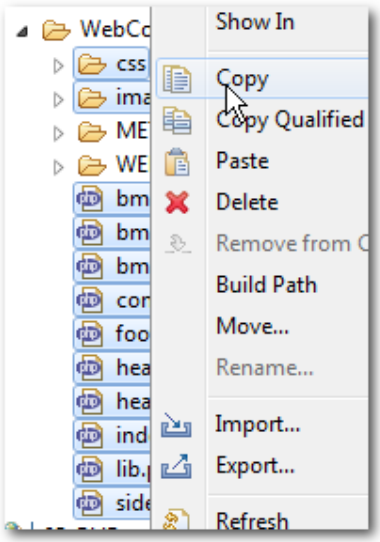
Für die Dynamische Webseite benötigen wir ein neues Projektverzeichnis vom Typ

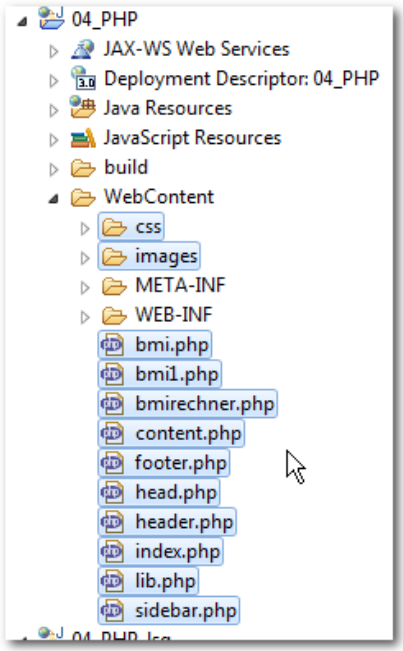
→ Dynamic Web Project

Klicken Sie dazu im linken Fenster von Eclipse für das Kontext-Menü (rechte Maustaste) und wählen Sie die Option → New → Dynamic Web Project aus.

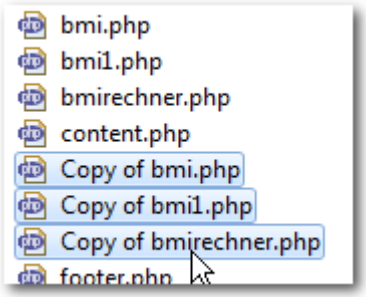
Hinweis:
Sie finden die Option auch → New → Other →

	Web → Dynamic Web Project.
	*Projektname festlegen.* Geben Sie den Namen für Ihr Projekt an und belassen Sie alle anderen Einstellungen. Schließen Sie den Vorgang mit einem Klick auf die Schaltfläche → Finish ab. 
Aktuelles Projekt (vorher): 	*Öffnen Sie das Projektverzeichnis.* Klicken Sie dazu auf den kleinen blauen Pfeil links neben dem Projektname und wählen Sie mit einem Klick das WebContent-Verzeichnis aus. → WebContent Hinweis: In diesem Verzeichnis werden wir die Inhalte des Projektes platzieren.
Letztes Projekt: 	*Prinzip der Wiederverwendung.* Öffnen und kopieren Sie die Inhalte aus dem letzten Projekt in das WebContent-Verzeichnis des aktuellen Projektes. Aktuelles Projekt (nachher):





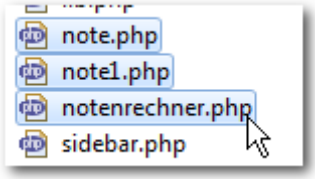
Für die Implementierung der Anwendung folgen Sie weiter dem Leittext.



Letztes Projekt kopieren und modifizieren.

Wir werden die Bmirechner-Dateien wiederverwenden.

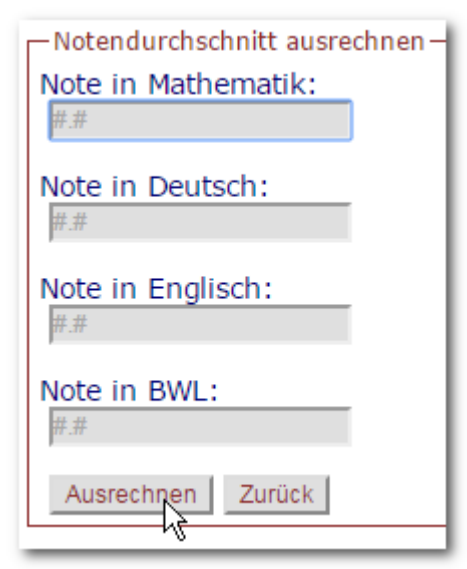
Kopieren Sie diese Dateien und benennen Sie die Dateien anschließend um:



Hinweis:
Um die Kopie umzubenennen können Sie den Dateinamen anklicken und die Taste → F2 klicken.

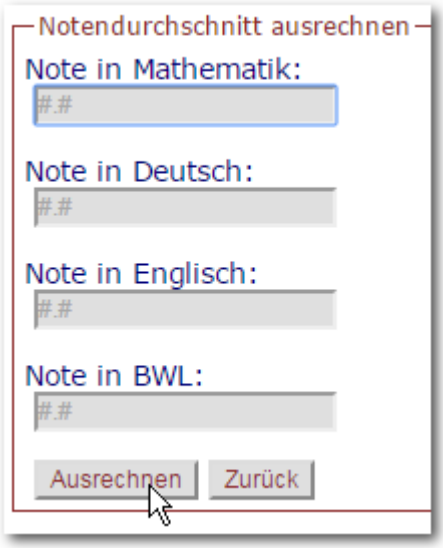
View

Formular:
→ Bmirechner
→ Bmirechner (opt)
→ Darlehensrechner
→ Notenrechner
→ Rabattrechner
→ Taschenrechner



Eingabe-Formular erzeugen: note.php

→ zurück zum Arbeitsblatt

	*Formulardatei anpassen (modifizieren).* Öffnen Sie dazu die Datei → note.php.
Name anpassen:  Action anpassen:	*Name und Action modifizieren.* Benennen Sie die Eigenschaften → name und → action, wie nebenstehend angezeigt, um.

```
action="note1.php">
```

Legenden-Box

```
<fieldset>
    <legend>Durchschnitt ausrechnen</legend>
```

Legende anpassen.

Benennen Sie den Legenden-Titel, wie nebenstehend angezeigt, um.

Label- und Textfelder anpassen

```
<label for="tfMathe">Note in Mathematik:</label> <br />
<input
    type="text" name="tfMathe" id="tfMathe"
    placeholder="#.#" required="required"
    autofocus="autofocus" /> <br /><br />

<label for="tfDeutsch">Note in Deutsch:</label>  <br />
<input
    type="text" name="tfDeutsch" id="tfDeutsch"
    placeholder="#.#" required="required"
     /> <br /><br />
```

Button (submit) anpassen

```
<input type="submit" value="Ausrechnen" name='DurchschnittAusrechnen'/>
```

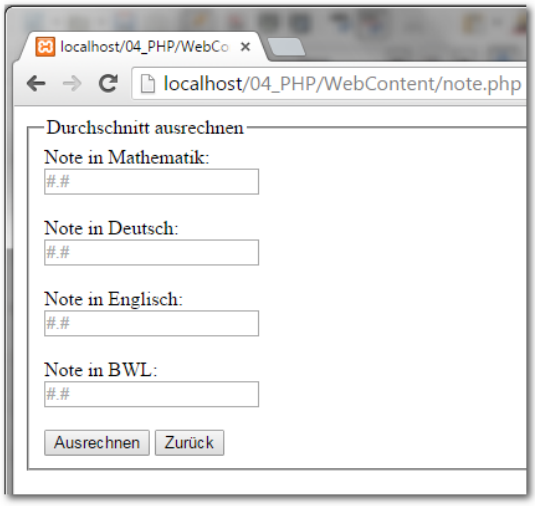
Komponenten anpassen und erweitern.

Benennen Sie die Eigenschaften → for, → name, → id und → placeholder für die Label- und Textfeld-Komponenten an, wie nebenstehend angezeigt, um.

Erweitern Sie die fehlenden Label- und Textfeld-Komponenten für Englisch und BWL.

Benennen Sie die Eingenschaft → name für die Button-Komponente vom Typ → submit wie nebenstehend angezeigt, um.

Im Web Browser



View: note.php

Testen Sie das Formular.

Öffnen Sie dazu den Internetbrowser und geben Sie den Pfad zur gerade erstellen Datei an.

Geben Sie dazu den Pfad ein.

http://localhost/04_PHP/WebContent/note.php

Herzlichen Glückwunsch Sie haben Ihr Formular erstellt.

Betten Sie nun das Formular in Ihr dynamisches Layout ein. Folgen Sie dazu dem Leittext.

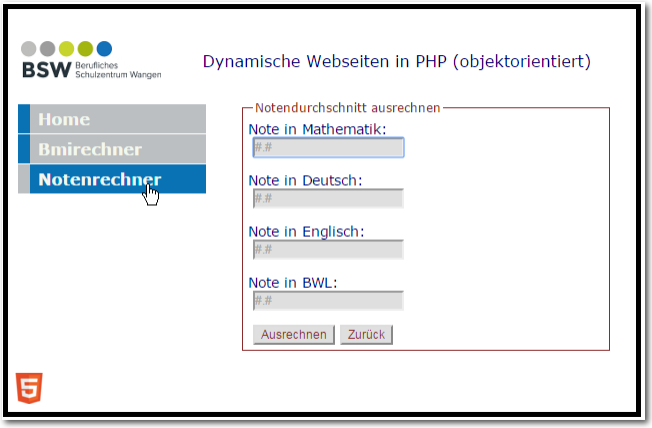
View

Box-Modell:

→ Bmirechner
→ Bmirechner (opt)
→ Darlehensrechner
→ Notenrechner
→ Rabattrechner
→ Taschenrechner

Eingabe-Formular einbetten: notenrechner.php

→ zurück zum Arbeitsblatt



Formulardatei in das bestehende Box-Modell der Seite einbetten.

Öffnen Sie dazu die Datei → notenrechner.php.

Vorher:

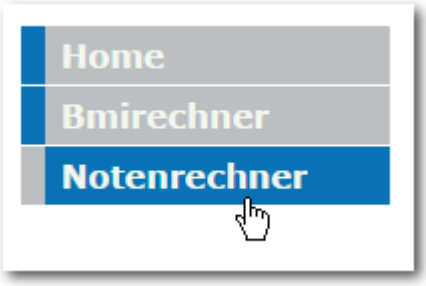
```
<?php
    include ("bmi.php");
?>
```

Nachher:

```
<?php
    include ("note.php");
?>
```

Formularreferenz anpassen.

Ändern Sie den Dateinamen im PHP-INCLUDE-Befehl, um das Formular einzubetten.



Navigation erweitern.

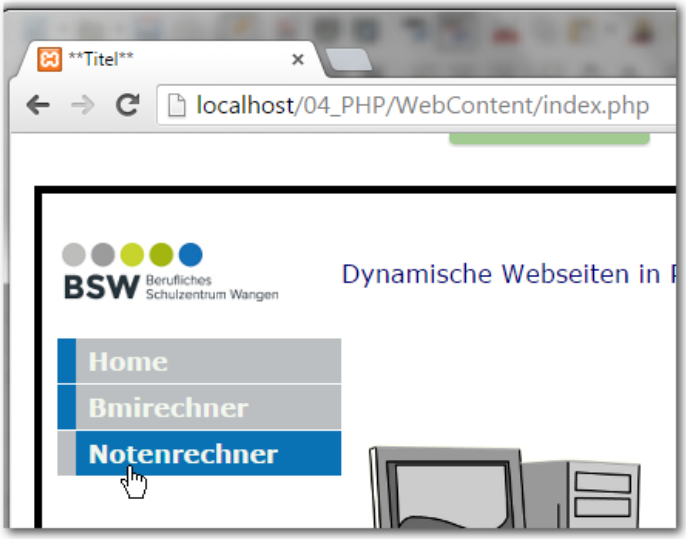
Erweitern Sie die Navigation um den Verweis auf den → Notenrechner.

Öffnen Sie dazu die Datei → sidebar.php.

Fügen Sie den Quellcode an entsprechender Stelle ein:

```
<li><a href="notenrechner.php"
      target="_parent">Notenrechner</a></li>
```

Im Web Browser



View: index.php

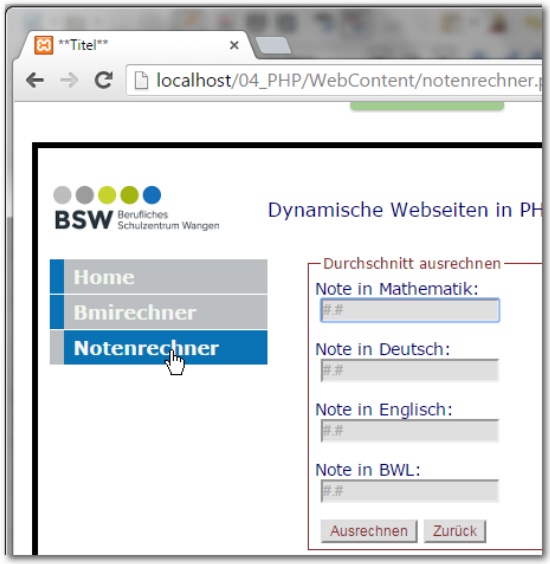
Testen Sie die Anwendung.

Öffnen Sie dazu den Internetbrowser und geben Sie den Pfad zur gerade erstellen Datei an. Geben Sie dazu den in der Grafik angezeigten Pfad ein.

http://localhost/04_PHP/WebContent/notenrechner.php

Klicken Sie in der Navigationsleiste auf die Option Bmirechner.

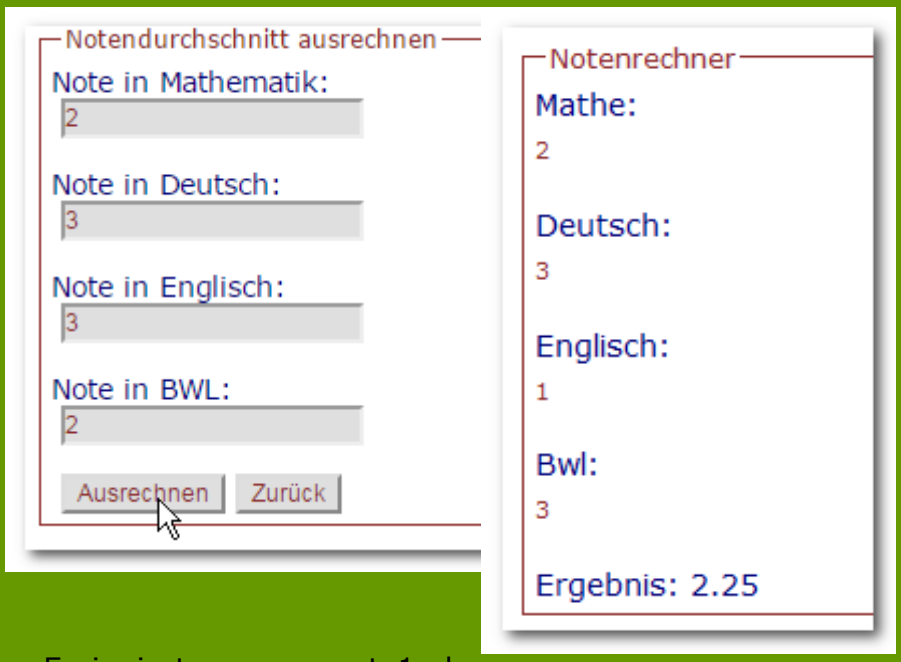
Herzlichen Glückwunsch Sie haben den Rechner erfolgreich in die Seit eingebettet.



Der Rechner rechnet noch nicht! Folgen Sie dazu weiter dem Leittext.

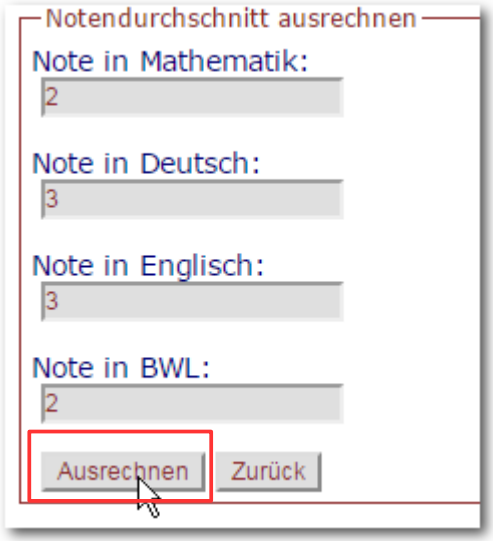
Controller

Ereignissteuerung:
→ Bmirechner
→ Bmirechner (opt)
→ Darlehensrechner
→ Notenrechner
→ Rabattrechner
→ Taschenrechner



Ereignissteuerung: note1.php

→ zurück zum Arbeitsblatt



Steuerungsdatei anpassen (modifizieren).

Öffnen Sie dazu die Datei → note1.php.

Die Ereignissteuerung muss erweitert und verändert werden.

Wir wenden dazu das EVA-Prinzip an:
→ Eingabe
→ Verarbeitung
→ Ausgabe

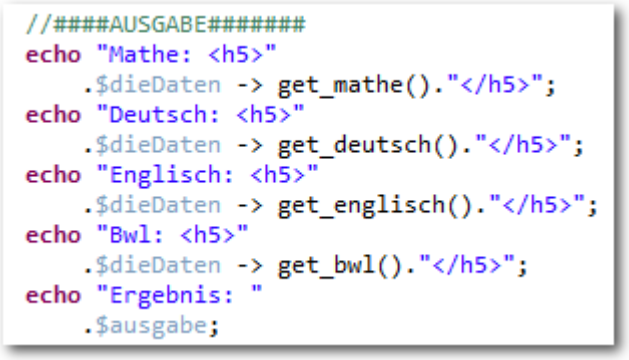
Legenden-Box
`<fieldset>`
`<legend>Durchschnitt berechnen</legend>`
	Legende anpassen.
	
	Benennen Sie den Legenden-Titel, wie nebenstehend angezeigt, um.

`//####EINGABE:#####`
`//Eingaben lesen`
`$pMathe = $_POST['tfMathe'];`
`$pDeutsch = $_POST['tfDeutsch'];`
	Eingabe: Formulardaten lesen.
	
	Passen Sie die Übernahme der Formulardaten in die lokalen Attribute an.
	
	Ergänzen Sie die Anweisungen für die Übernahme der noch fehlenden Formulardaten.

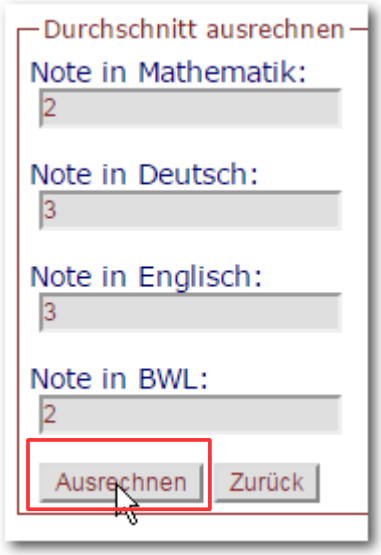
`/*Es wird ein neues Objekt der`
`* Klasse bmirechner erzeugt*/`
`$dieDaten = new Notenrechner();`
	Verarbeitung: Objekt der Klasse erzeugen.
	
	Wir werden für das Model im Anschluss eine Klasse → Notenrechner erzeugen.
	
	Für die Verarbeitung der Formulardaten benötigen wir dieses Objekt.
	
	Verändern Sie die Anweisung zur Erzeugung eines Objektes der Klasse → Notenrechner.

`/*Methodenaufruf: Die Eingabewerte werden`
`* an die Attribute des Objektes übermittelt*/`
`$dieDaten -> set_mathe($pMathe);`
`$dieDaten -> set_deutsch($pDeutsch);`
	Verarbeitung: Set-Methodenaufrufe anpassen
	
	Alle Eingabewerte müssen an das Objekt der Fachklasse übermittelt werden. Dazu dienen die Setter der Fachklasse.
	
	Verändern und erweitern Sie die Methodenaufrufe, um die Eingabewerte zu übermitteln.

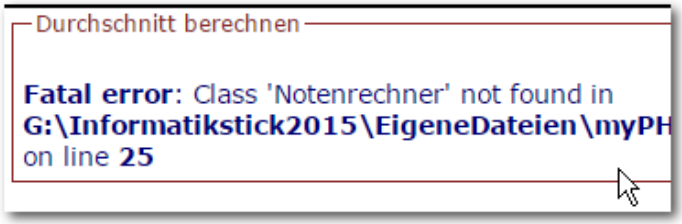
`/*Methodenaufruf: Die Berechnung wird`
`* für das Objekt durchgeführt*/`
`$dieDaten -> berechne_durchschnitt();`
	Verarbeitung: Berechnende Methoden aufrufen
	
	Passen Sie den Methodenaufruf an, wie nebenstehend angezeigt.


`//####AUSGABE#######`
`echo "Mathe: <h5>"`
`.$dieDaten -> get_mathe()."</h5>";`
`echo "Deutsch: <h5>"`
`.$dieDaten -> get_deutsch()."</h5>";`
`echo "Englisch: <h5>"`
`.$dieDaten -> get_englisch()."</h5>";`
`echo "Bwl: <h5>"`
`.$dieDaten -> get_bwl()."</h5>";`
`echo "Ergebnis: "`
`.$ausgabe;`
	Ausgabe: Ausgaben anpassen
	
	Passen Sie die Anweisungen für die Ausgaben auf der Benutzeroberfläche an und erweitern Sie die fehlenden Anweisungen, wie nebenstehend angezeigt.

Im Web Browser	*Zwischenergebnis testen.* Öffnen Sie dazu den Internetbrowser und geben Sie den Pfad zur gerade erstellen Datei an. Geben Sie dazu den in der Grafik angezeigten Pfad ein. http://localhost/04_PHP/WebContent/notenrechner.php Klicken Sie in der Navigationsleiste auf die Option → Notenrechner. Rechnet der Rechner schon?



View: notenrechner.php



Fehler beheben.

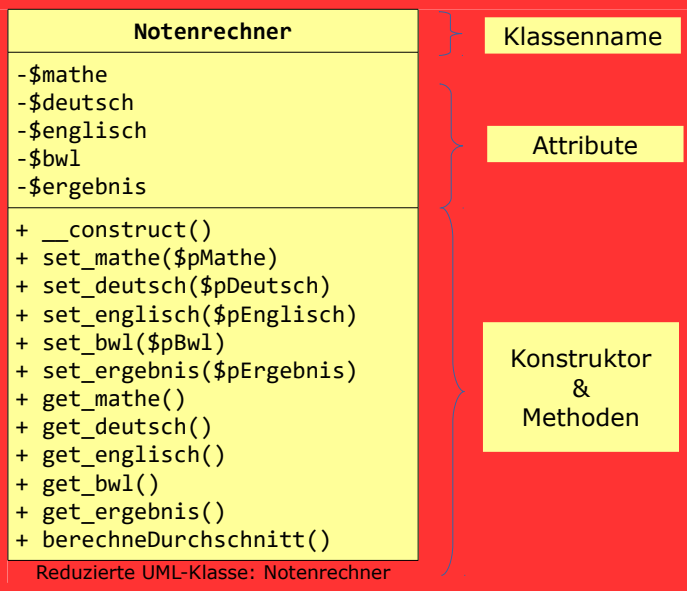
Wenn Sie die Eingaben tätigen und mit einem Klick auf die Schaltfläche → Ausrechnen ds Ereignis auslösen, meldet System:

„Class 'Notenrechner' not found...

Wir werden diesen Fehler beheben indem wir die noch fehlende Klasse → Notenrechner (unser Model erweitern und in der Bibliothek (→ lib.php) einbetten.

Model

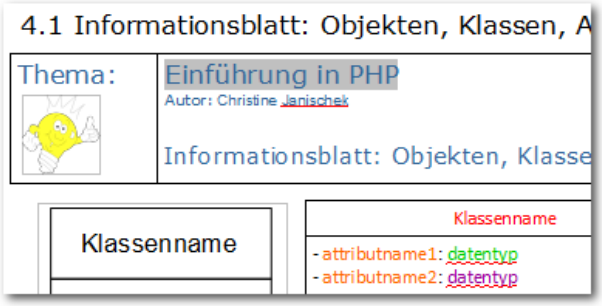
Fachklassen:
→ Bmirechner
→ Bmirechner (opt)
→ Darlehensrechner
→ Notenrechner
→ Rabattrechner
→ Taschenrechner



Reduzierte UML-Klasse: Notenrechner

UML-Klasse → Notenrechner: lib.php

→ zurück zum Arbeitsblatt

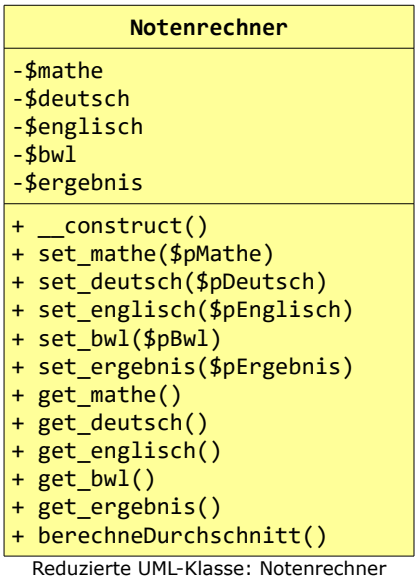


Verarbeitung

Nutzen Sie das Informationsblatt. Klären Sie die Begriffe und informieren Sie sich über das Grundgerüst einer Klasse in PHP.

Wir implementieren die Fachklasse → Notenrechner, indem wir sie mit dem benötigten Quellcode ausstatten.

Entsprechend den Vorgaben (Anforderungen) der nebenstehend angezeigten UML-Klasse, werden wir das in den kommenden Schritten tun.



Reduzierte UML-Klasse: Notenrechner

Berechnung:

```
durchschnitt = (mathe+deutsch+englisch+bwl)/4;
```

Fachklasse implementieren.

Öffnen Sie dazu die Bibliotheksdatei → lib.php.

Integrieren Sie unterhalb der bereits enthaltenen Klasse(n) , die neue Fachklasse:
→ Notenrechner

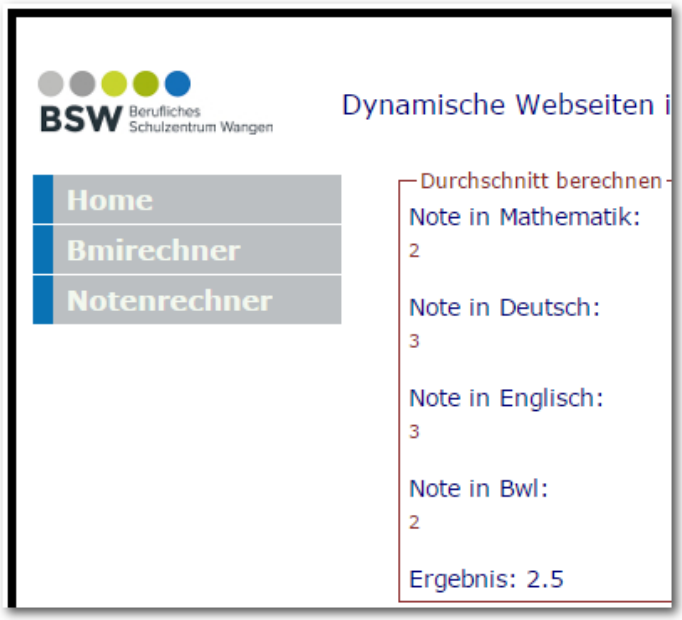
Gehen Sie vor wie zuvor für die Klasse → Bmirechner beschrieben:
→ Klasse und Attribute deklarieren. (lesen)
→ Konstruktor deklarieren.(lesen)
→ Get- und Set-Methoden deklarieren.(lesen)
→ Methode für die Berechnung. (lesen)

Hinweis:
Berücksichtigen Sie bitte unbedingt die Vorgaben aus der nebenstehenden UML-Klassen, um unnötige Syntaxfehler zu vermeiden.

Hinweis: Falls Fehler angezeigt werden berücksichtigen Sie die folgenden Tipps:

→ Beheben Sie die Fehler von oben nach unten
→ Testen Sie nach jeder Korrektur
→ Nutzen Sie die Zeilen- und Dateiangabe
→ Prüfen Sie die Attribut- und Methodennamen
→ Prüfen Sie die Klammersetzung
→ Prüfen Sie ob Zeichen (z.B. „;“) fehlen

Testen Sie die Anwendung erneut.



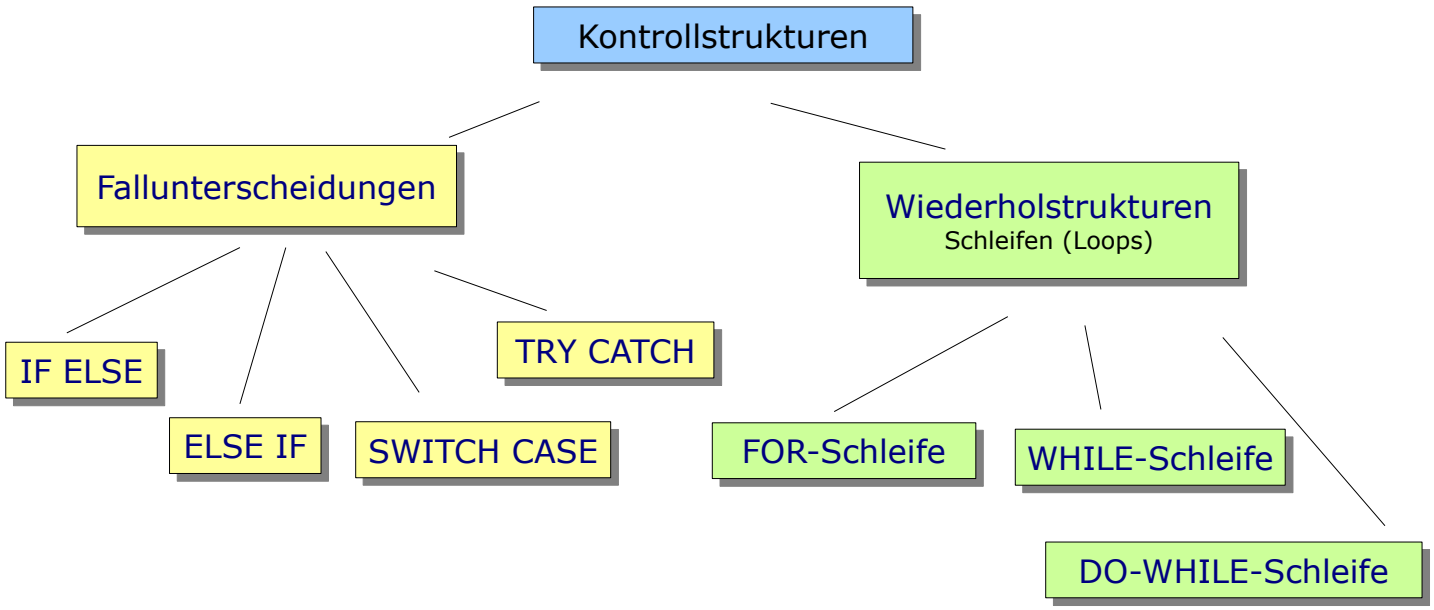
Mit dieser Vorgehensweise können Sie künftig beliebig viele Klassen integrieren und an beliebiger Stelle im System nutzen.

Herzlichen Glückwunsch Sie haben die Anwendung objektorientiert implementiert.

6 Taschenrechner: Fallunterscheidungen

6.1 Informationsblatt: Kontrollstrukturen → Fallunterscheidungen

Thema:	Kontrollstrukturen: Fallunterscheidungen Autor: Christine Janischek Informationsblatt: Überblick zu Fallunterscheidungen



Hinweis: Fallunterscheidungen sind Bestandteil von Verhaltensweisen (Methoden)		
Fallunterscheidung	**Grundgerüst**	**Besonderheit**

IF ELSE	`if(Bedingung){`
	`  //Anweisung(en)`
	`}else{`
	`  //weitere Anweisung(en)`
	`}`
		Die Bedingung muss zum Wahrheitswert „true" oder „false" evaluieren können.
		
		Handelt 'nicht-geschachtelt' nur genau zwei Fälle ab.
		
		Große Werte zuerst prüfen, um logische Fehler zu vermeiden!

ELSE IF	`if(Bedingung1){` `  //Anweisung(en)`	Kann beliebig viele Fälle behandeln.

	`}else if(Bedingung2){`
	`  //weitere Anweisung(en)`
	`}else if(Bedingung3){`
	`  //weitere Anweisung(en)`
	`}else{`
	`  //weitere Anweisung(en)`
	`}`
		Große Werte zuerst prüfen, um logische Fehler zu vermeiden!
		
		Die ELSE IF prüft nur solange bis eine Bedingung zu → true evaluiert, weitere wahren Bedingungen werden ignoriert.

SWITCH CASE	`switch(Ausdruck){`
	
	`  case ausdruck_1:`
	`  //Anweisung(en)`
	`  break;`
	
	`  case ausdruck_2:`
	`  //Anweisung(en)`
	`  break;`
	
	`  case ausdruck_3:`
	`  //Anweisung(en)`
	`  break;`
	
	`  default:`
	`  //Anweisung(en)`
	`}`
		Kann Fälle in Abhängigkeit des Ausdrucks abhandeln. Der Ausdruck enthält einen Wert, die Wertigkeit kann variieren und enthält zumeist eines primitiven Datentyps (int, double, char,...).
		
		Mit dem → break wird nach erfolgreicher Prüfung die Kontrollstruktur verlassen.
		
		Wenn eine Methode mit Rückgabewert implementiert wird erzeugt der Aufruf → return denselben Effekt, wie mit dem → break. Der Einsatz des break-Befehls ist damit hinfällig.

TRY CATCH	`try {`
	`  //Anweisung(en)`
	`} catch (Exception $e) {`
	`  //Anweisung(en)`
	`}`
		Ausnahmebehandlung (Exception-Handling)

6.2 Arbeitsblatt: Übung einfache Methoden und Fallunterscheidungen

Thema:	Kontrollstrukturen: Fallunterscheidungen Autor: Christine Janischek Arbeitsblatt: Übung einfache Methoden und Fallunterscheidungen

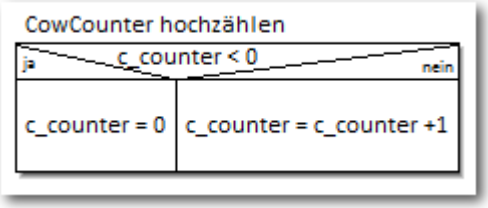
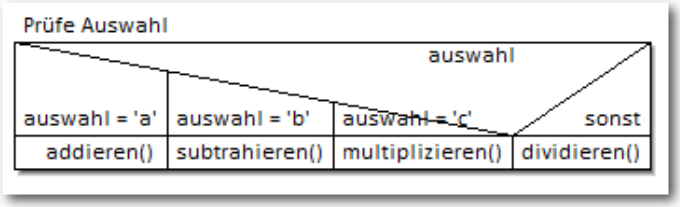
Information:

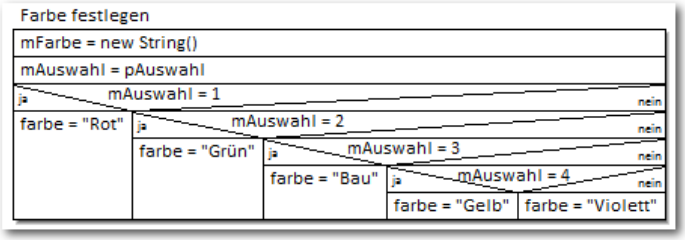
Ein Programmierer benötigt in der Praxis vielfach die Möglichkeit alternative Lösungswege zu entwickeln. Einige Strukturen sog. Algorithmen und die Kontrollstrukturen an sich, sind dazu das passende Handwerkszeug. Nutzen und erweitern Sie den Methodentester aus dem Kapitel Methoden, Kontrollstrukturen und Algorithmen im E-Learning um die Lösungen zu Testen.

Aufgabe:

Üben Sie mit Hilfe der folgenden Aufgaben Ihr algorithmisches Denken. Setzen Sie dazu die folgenden Struktogramme in Quellcode (Programmcode) um. Nutzen Sie dazu nach Bedarf die Kontrollstrukturen aus der Übersicht.

Programmieren Sie händisch mittels Papier und Bleistift!

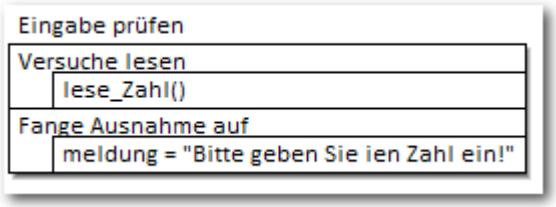
	*Kühe zählen.* Wir möchten eine Anwendung entwickeln die Kühe zählen kann. Eine Methode soll in der Lage sein den Zähler um eine Einheit zu erhöhen Schreiben Sie die Anweisungen in eine Methode `→ cowCounter hochzählen` der Klasse Kuh.
	*Auswahl prüfen.* Die Rechenoperation soll für einen Taschenrechner individuell wählbar sein. Schreiben Sie die Anweisungen in eine Methode `→ berechnen` der Klasse Taschenrechner. Implementieren Sie auch die verwendeten Methoden für die Addition, Subtraktion, Multiplikation und Division zweier Zahlen.



Farbe festlegen.

Die Farbe soll für Grafiken individuell bestimmbar sein.

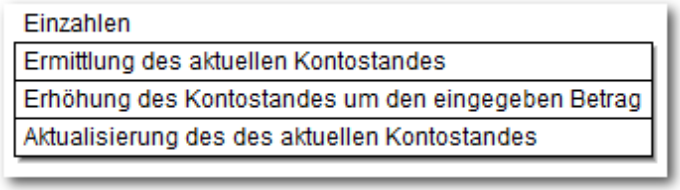
Schreiben Sie die Anweisungen in eine Methode → `Farbe festlegen` der Klasse Grafik.



Eingabe prüfen.

Die Eingabe soll geprüft werden.

Schreiben Sie die Anweisungen in eine Methode → `check Eingabe` der Klasse Checker.

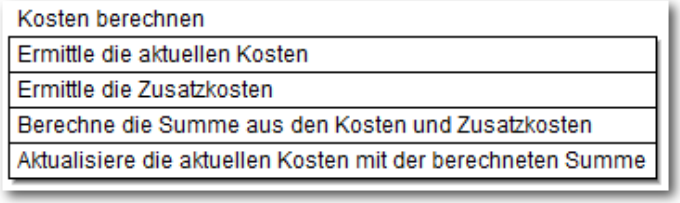


Einzahlen.

Die Einzahlung auf einem Konto soll ermöglicht werden.

Schreiben Sie die Anweisungen in eine Methode → `einzahlen` der Klasse Konto.

Erstellen Sie 2 Varianten:
1.V. → mit Rückgabewert
2.V → ohne Rückgabewert



Kosten berechnen.

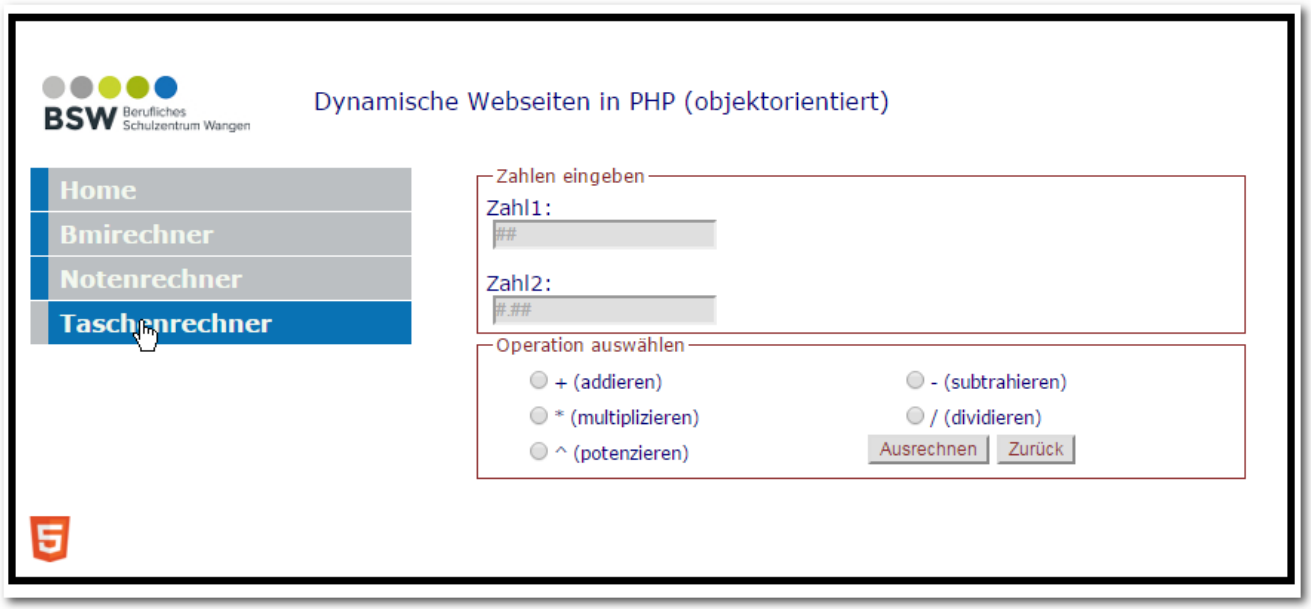
Die Kosten für ein Haus ist mit Zusatzkosten verbunden.

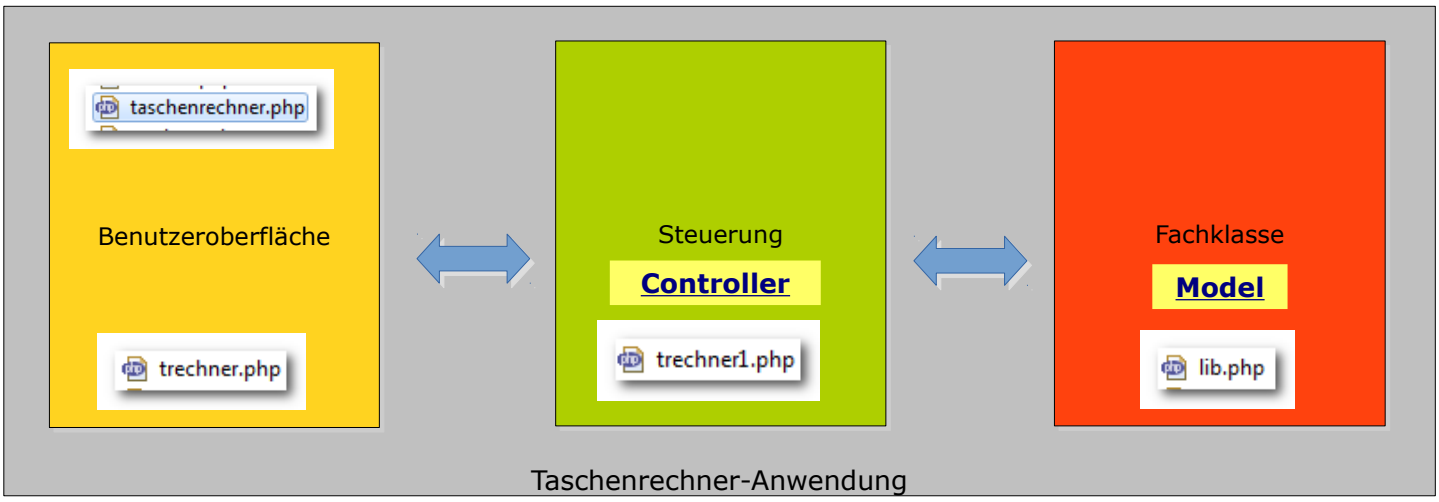
Schreiben Sie die Anweisungen in eine Methode → `berechne Kosten` der Klasse Haus.

Erstellen Sie 2 Varianten:
1.V. → mit Parameter und mit Rückgabewert
2.V → mit Parameter und ohne Rückgabewert

6.3 Arbeitsblatt: Taschenrechner

Thema:	Kontrollstrukturen: Fallunterscheidungen Autor: Christine Janischek Arbeitsblatt: Taschenrechner



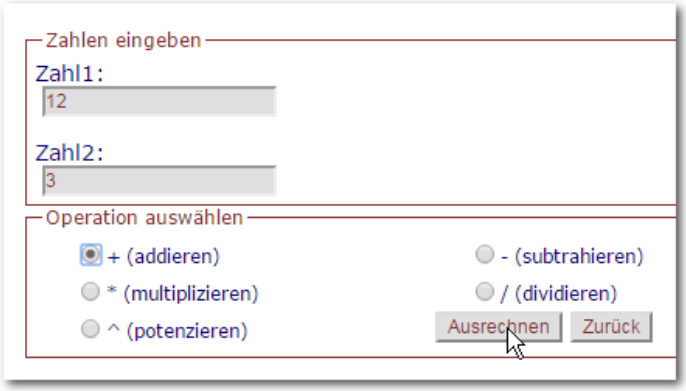


Arbeitsauftrag:

1. Erzeugen Sie ein neues Projekt um den → Taschenrechner umzusetzen. Nutzen Sie dazu den folgenden Leittext.
2. Dokumentieren Sie welche einzelnen Schritte für die Umsetzung der View, des Controllers und des Models notwendig sind.

6.4 Leittext: Taschenrechner

Ergebnis der Übung: → taschenrechner.php (View)



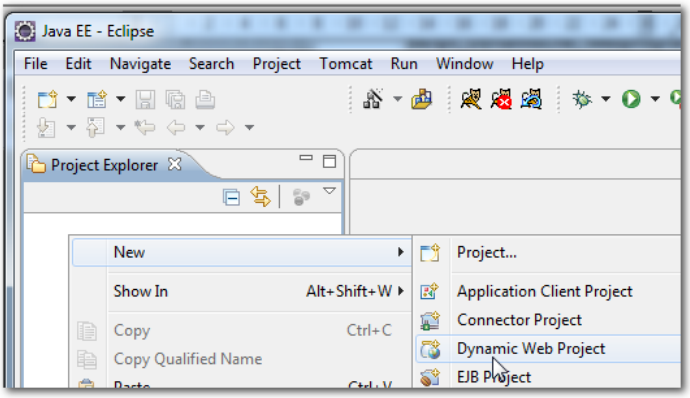
Eingabe-Formular: → trechner.php (View)

Taschenrechner
-$zahl1 -$zahl2 -$operator -$ergebnis
+ __construct() + set_zahl1($pZahl1) + set_zahl2($pZahl2) + set_operator($pOperator) + set_ergebnis($pErgebnis) + get_zahl1() + get_zahl2() + get_operator() + get_ergebnis() + addieren() + subtrahieren() + multiplizieren() + dividieren() + berechne()

Reduzierte UML-Klasse: Taschenrechner

Bibliothek: → lib.php (Model)

Ausgabe-Datei: → trechner1.php (Controller)



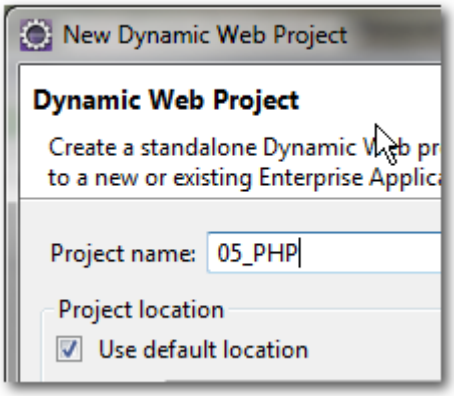
Neues Projekt erstellen.

Für die Dynamische Webseite benötigen wir ein neues Projektverzeichnis vom Typ

→ Dynamic Web Project

Klicken Sie dazu im linken Fenster von Eclipse für das Kontext-Menü (rechte Maustaste) und wählen Sie die Option → New → Dynamic Web Project aus.

Hinweis:
Sie finden die Option auch → New → Other → Web → Dynamic Web Project.



Projektname festlegen.

Geben Sie den Namen für Ihr Projekt an und belassen Sie alle anderen Einstellungen.

Schließen Sie den Vorgang mit einem Klick auf die Schaltfläche → Finish ab.

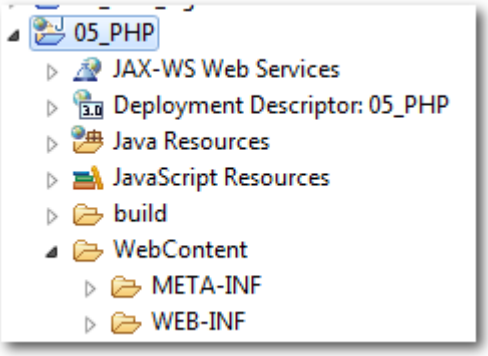
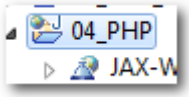
Aktuelles Projekt (vorher):

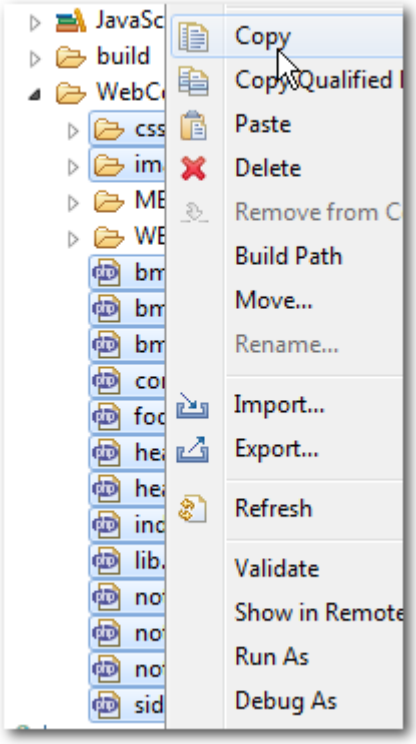
Öffnen Sie das Projektverzeichnis.

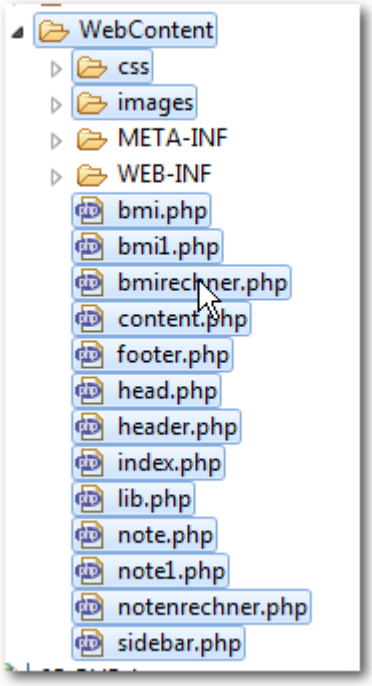
Klicken Sie dazu auf den kleinen blauen Pfeil links neben dem Projektname und wählen Sie mit einem Klick das WebContent-Verzeichnis aus.

→ WebContent

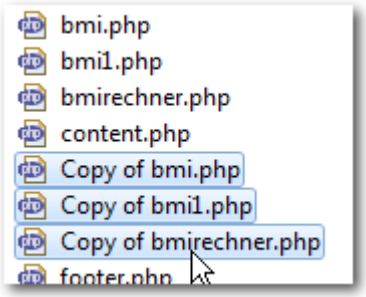
Hinweis:

	In diesem Verzeichnis werden wir die Inhalte des Projektes platzieren.
Letzten Projekt: 	*Prinzip der Wiederverwendung.* Öffnen und kopieren Sie die Inhalte aus dem letzten Projekt in das WebContent-Verzeichnis des aktuellen Projektes. Aktuelles Projekt (nachher):





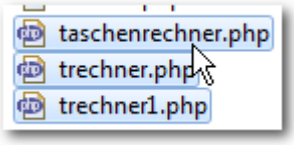
Für die Implementierung der Anwendung folgen Sie weiter dem Leittext.



Letztes Projekt kopieren und modifizieren.

Wir werden die Bmirechner-Dateien wiederverwenden, da dieses Projekt mit zwei Textfeldern die meisten Ähnlichkeiten aufweist.

Kopieren Sie diese Dateien und benennen Sie die Dateien anschließend um:

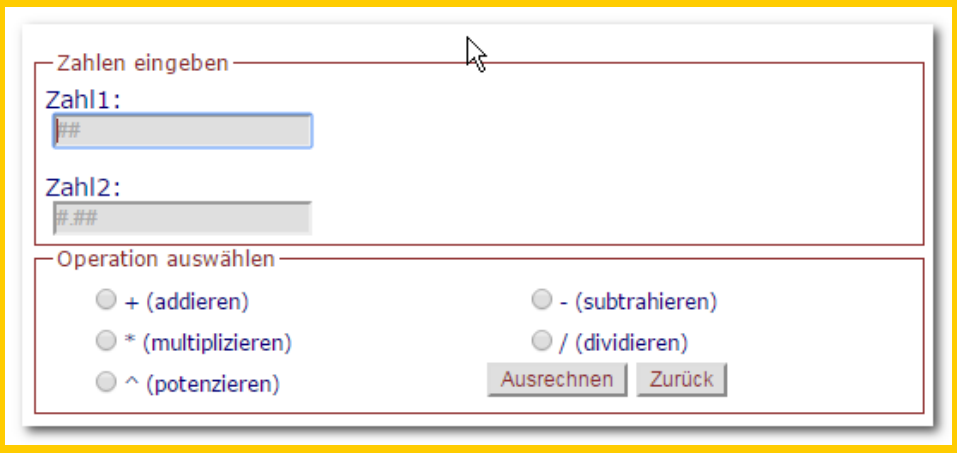


Hinweis:
Um die Kopie umzubenennen können Sie den Dateinamen anklicken und die Taste → F2 klicken.

View

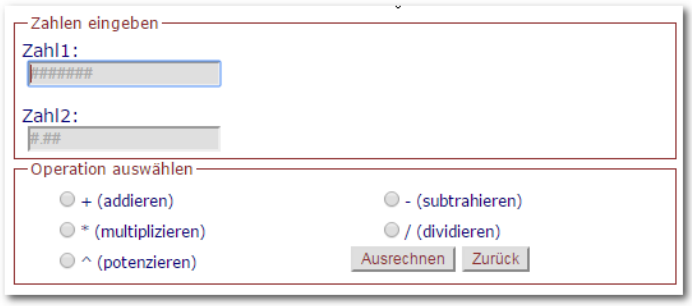
Formular:

- → Bmirechner
- → Bmirechner (opt)
- → Darlehensrechner
- → Notenrechner
- → Rabattrechner
- → Taschenrechner



Eingabe-Formular erzeugen: trechner.php

→ zurück zum Arbeitsblatt



Formulardatei anpassen (modifizieren).

Öffnen Sie dazu die Datei → trechner.php.

Name anpassen:

```
t >
 <form name="taschenrechnerformular"
```

Action anpassen:

```
action="trechner1.php"
```

Name und Action modifizieren.

Benennen Sie die Eigenschaften → name und → action, wie nebenstehend angezeigt, um.

Legenden-Box

```
<fieldset>
    <legend>Zahlen eingeben</legend>
```

Legende anpassen.

Label- und Textfelder anpassen

```
<label for="tfGewicht">Gewicht (in Kg):</label> <br />
<input
    type="number" name="tfGewicht" id="tfGewicht"
    placeholder="##" required="required"
    autofocus="autofocus" />  <br /><br />

<label for="tfGroesse">Gr&ouml;&szlig;e (in m):</label>  <br />
<input
    type="number" name="tfGroesse" id="tfGroesse"
    placeholder="#.##" required="required"
    autofocus="autofocus" /> <br /><br />
```

Komponenten anpassen und erweitern.

Passen Sie die bestehenden Lable und Textfelder an das Taschenrechnerformular an.

Benennen Sie die Eingenschaft → name für die Button-Komponente vom Typ → submit wie nebenstehend angezeigt, um.

Button (submit) anpassen

```
<input type="submit" value="Ausrechnen" name='Ausrechnen'/>
```

```
        autofocus="autofocus" /> <br /><br />
</fieldset>
<fieldset>
    <legend>Operation ausw&auml;hlen</legend>

    <input type="submit" value="Ausrechnen" name='Ausrechnen'/>
      <?php echo"<input type='button' value='Zur&uuml;ck' onClick='history.back()' />" ?>
</fieldset>
```

Weitere Fieldset-Box definieren.

Kopieren Sie dazu die bestehende Fieldset-Box und fügen Sie diese unterhalb ein.

Entfernen Sie aus der oberen Fieldset-Box die Schaltflächen.

Entfernen Sie innerhalb der unteren Fieldset-Box die die Label- und Textfeld-Komponenten.

Benennen Sie die Legenden-Box um, wie nebenstehend angezeigt.

```
<legend>Operation auswählen</legend>

<fieldset id="left">
    <input type="radio" name="rbOperator" value="1" />
        + (addieren)<br />
</fieldset>
<fieldset id="right">
    <input type="radio" name="rbOperator" value="2" />
        - (subtrahieren)<br />
</fieldset>
<fieldset id="left">
    <input type="radio" name="rbOperator" value="3" />
        * (multiplizieren)<br />
</fieldset>
<fieldset id="right">
    <input type="radio" name="rbOperator" value="4" />
        / (dividieren)<br />
</fieldset>
<fieldset id="left">
    <input type="radio" name="rbOperator" value="5" />
        ^ (potenzieren)<br />
</fieldset>
```

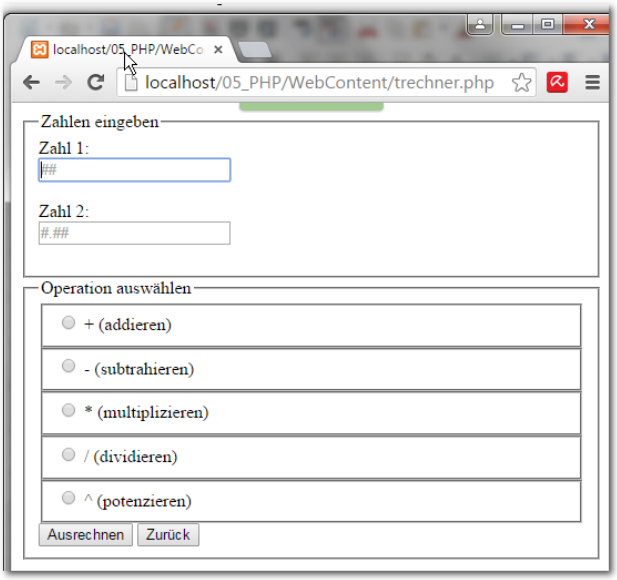
Input-Komponenten vom Typ 'radio' einfügen.

Fügen Sie dazu unterhalb der Legenden-Box die Fieldsets mit den ids:
→ left
→ right
ein. Fügen Sie jeweils die Input-Komponenten vom Typ 'radio' ein

Achten Sie darauf die Eigenschaften:
→ type
→ name
→ value
entsprechend den Vorgaben zu definieren.

Im Web Browser

Testen Sie das Formular.



View: trechner.php

Öffnen Sie dazu den Internetbrowser und geben Sie den Pfad zur gerade erstellen Datei an.

Geben Sie dazu den Pfad ein.

http://localhost/05_PHP/WebContent/trechner.php

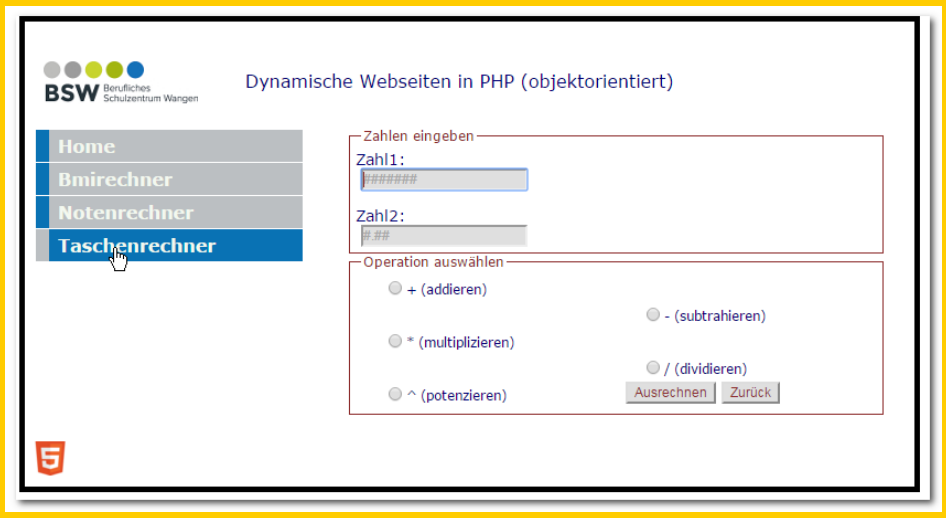
Herzlichen Glückwunsch Sie haben Ihr Formular erstellt.

Betten Sie nun das Formular in Ihr dynamisches Layout ein. Folgen Sie dazu dem Leittext.

View

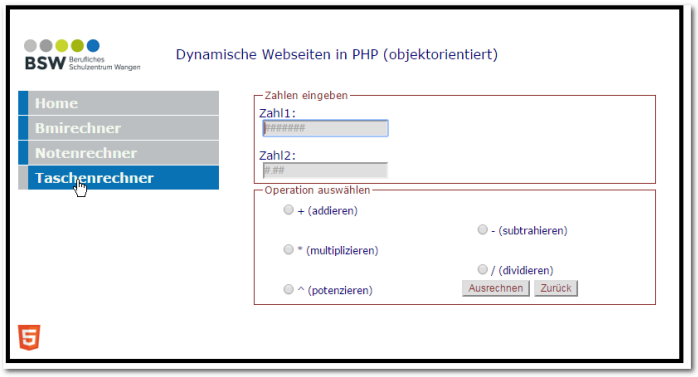
Box-Modell:

→ Bmirechner
→ Bmirechner (opt)
→ Darlehensrechner
→ Notenrechner
→ Rabattrechner
→ Taschenrechner



Eingabe-Formular einbetten: taschenrechner.php

→ zurück zum Arbeitsblatt



Formulardatei in das bestehende Box-Modell der Seite einbetten.

Öffnen Sie dazu die Datei
→ taschenrechner.php.

Vorher:

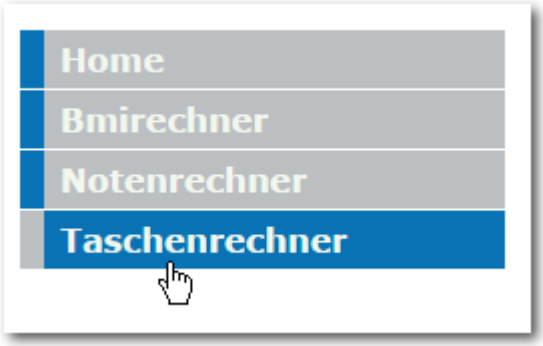
```
<?php
    include ("bmi.php");
?>
```

Nachher:

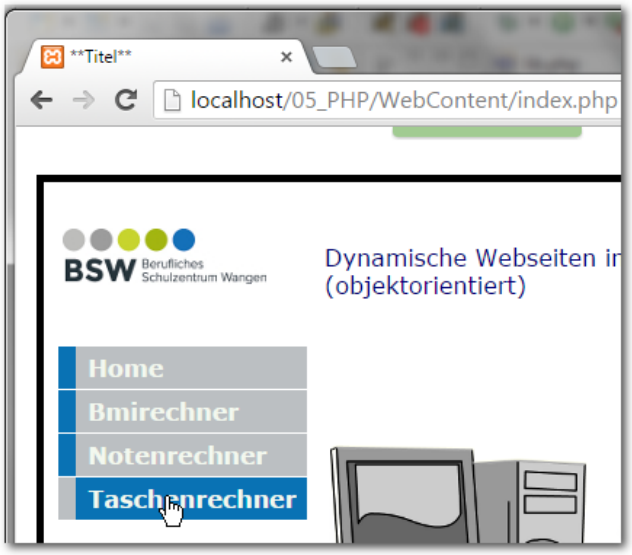
```
<?php
    include ("trechner.php");
?>
```

Formularreferenz anpassen.

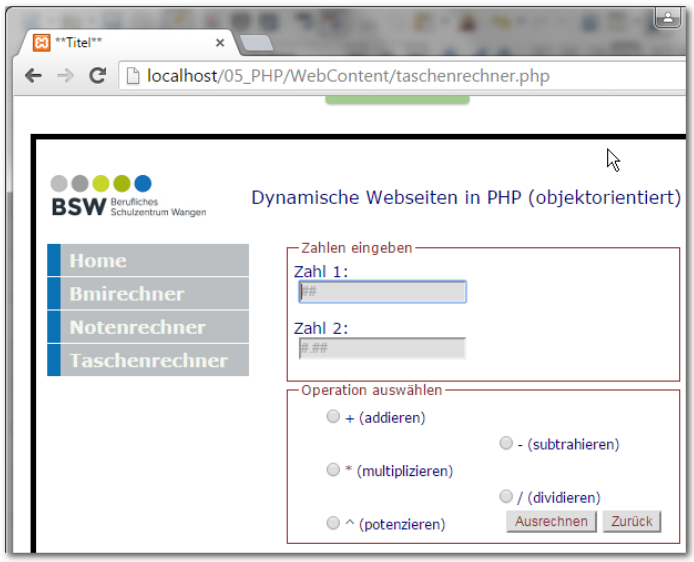
Ändern Sie den Dateinamen im PHP-INCLUDE-Befehl, um das Formular einzubetten.


Home
Bmirechner
Notenrechner
Taschenrechner

Im Web Browser	*Testen Sie die Anwendung.* Öffnen Sie dazu den Internetbrowser und geben Sie den Pfad zur gerade erstellen Datei an. Geben Sie dazu den in der Grafik angezeigten Pfad ein. http://localhost/05_PHP/WebContent/taschenrechner.php Klicken Sie in der Navigationsleiste auf die Option Taschenrechner. **Herzlichen Glückwunsch Sie haben den Rechner erfolgreich in die Seite eingebettet.**



View: index.php



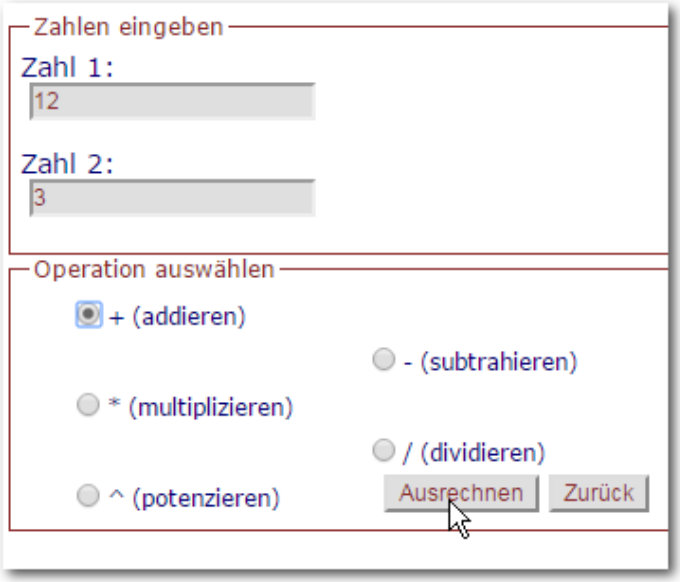
Der Rechner rechnet noch nicht! Folgen Sie dazu weiter dem Leittext.

Controller

Ereignissteuerung:
- → Bmirechner
- → Bmirechner (opt)
- → Darlehensrechner
- → Notenrechner
- → Rabattrechner
- → Taschenrechner

Ereignissteuerung: trechner1.php

→ zurück zum Arbeitsblatt



Steuerungsdatei anpassen (modifizieren).

Öffnen Sie dazu die Datei → trechner1.php.

Die Ereignissteuerung muss erweitert und verändert werden.

Wir wenden dazu das EVA-Prinzip an:
→ Eingabe
→ Verarbeitung
→ Ausgabe

Legenden-Box

```
<fieldset>

    <legend>Ergebnis berechnen</legend>
```

Legende anpassen.

Benennen Sie den Legenden-Titel, wie nebenstehend angezeigt, um.

```
//####EINGABE:#####
//Eingaben lesen
          $pZahl1 = $_POST['tfZahl1'];
          $pZahl2 = $_POST['tfZahl2'];
```

Eingabe: Formulardaten lesen.

Passen Sie die Übernahme der Formulardaten in die lokalen Attribute an.

```
//####VERARBEITUNG#######
    /*Es wird ein neues Objekt
     * der Klasse erzeugt*/
    $dieDaten = new Taschenrechner();
```

Verarbeitung: Objekt der Klasse erzeugen.

Wir werden für das Model im Anschluss eine Klasse → Taschenrechner erzeugen.

Für die Verarbeitung der Formulardaten benötigen wir dieses Objekt.

```php
//Prüfung: Wurde ein Radio-Button ausgewählt?
  if(isset($_POST['rbOperator'])){
    $pOperator = $_POST['rbOperator'];
  }else{
    $pOperator = 0;
  }

  $dieDaten -> set_operator($pOperator);
```

Prüfende Methode vom Typ → bool
→ `isset(...)`

Eigenschaftswert des Operators übermitteln
`$dieDaten -> set_operator($pOperator);`

Verarbeitung: Input-Radio prüfen und Eigenschaftswert übermitteln.

Bevor wir die Daten des Operators übernehme, sollten wir prüfen, ob überhaupt eine Eingabe erfolgt ist.

Informieren Sie sich anhand der Informationsblätter über Kontrollstrukturen. (lesen)

`isset(...)` ist seit PHP 4 eine Methode der PHP-Bilbliothek. Die Methode prüft eine Variable, ob Sie nicht → NULL ist liefert dann den Wahrheitswert → true" oder → false.

Im Anschluss an die Prüfung kann der Eigenschaftswert des Operators an das Objekt der Fachklasse übermittelt werden.

Lösen Sie im Anschluss an dieses Projekt, zum besseren Verständnis, das Aufgabenblatt:
→ Übung einfache Methoden und Fallunterscheidungen

```php
/*Methodenaufruf: Die Eingabewerte
 * werden an die Attribute des
 * Objektes übermittelt*/
$dieDaten -> set_zahl1($pZahl1);
$dieDaten -> set_zahl2($pZahl2);
```

Verarbeitung: Set-Methodenaufrufe anpassen

Alle Eingabewerte müssen an das Objekt der Fachklasse übermittelt werden. Dazu dienen die Setter der Fachklasse.

Verändern und erweitern Sie die Methodenaufrufe, um die Eingabewerte zu übermitteln.

```php
/*Methodenaufruf: Die Berechnung
 * wird für das Objekt durchgeführt*/
$dieDaten -> berechne();
```

Verarbeitung: Berechnende Methoden aufrufen

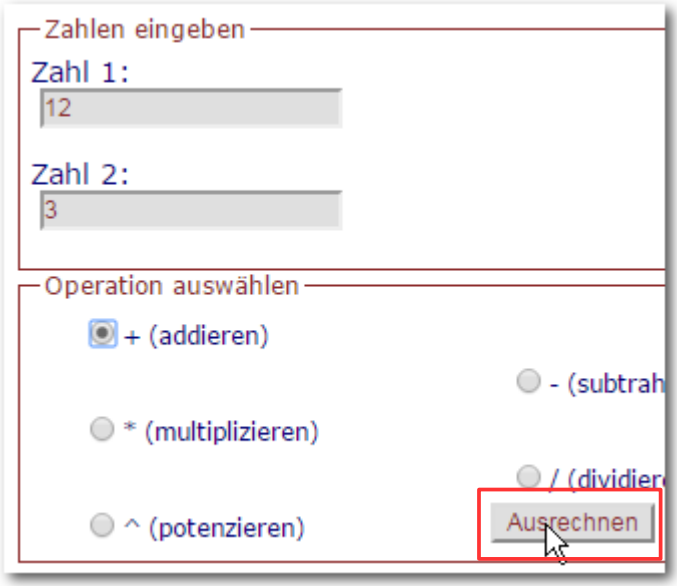
Passen Sie den Methodenaufruf an, wie nebenstehend angezeigt.

```
//####AUSGABE#######
echo "Zahl1: <h5>"
    .$dieDaten -> get_zahl1()."</h5>";
echo "Zahl2: <h5>"
    .$dieDaten -> get_zahl2()."</h5>";
echo "Operator: <h5>"
    .$dieDaten -> get_operator()."</h5>";
echo "Ergebnis: "
    .$ausgabe;
```

Ausgabe: Ausgaben anpassen

Passen Sie die Anweisungen für die Ausgaben auf der Benutzeroberfläche an und erweitern Sie die fehlenden Anweisungen, wie nebenstehend angezeigt.

Im Web Browser



View: taschenrechner.php

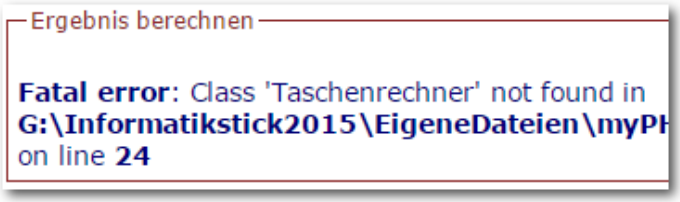
Zwischenergebnis testen.

Öffnen Sie dazu den Internetbrowser und geben Sie den Pfad zur gerade erstellen Datei an. Geben Sie dazu den in der Grafik angezeigten Pfad ein.

http://localhost/05_PHP/WebContent/taschenrechner.php

Klicken Sie in der Navigationsleiste auf die Option → Taschenrechner.

Rechnet der Rechner schon?



Fehler beheben.

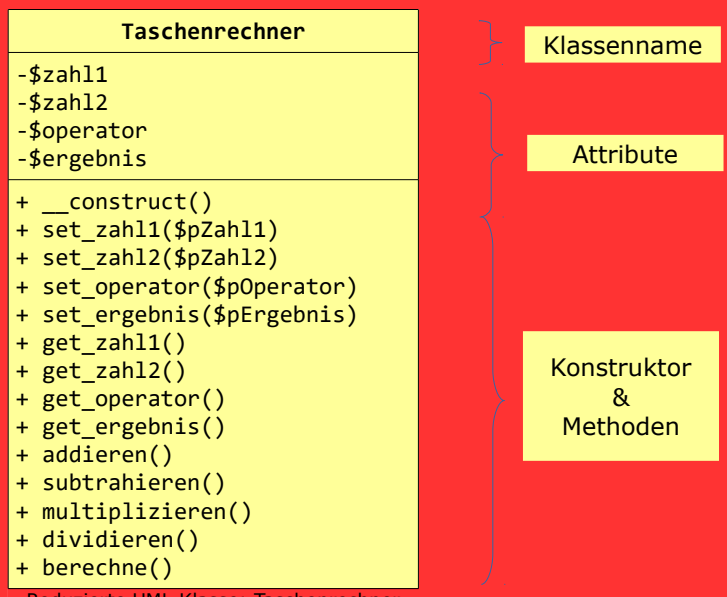
Wenn Sie die Eingaben tätigen und mit einem Klick auf die Schaltfläche → Ausrechnen ds Ereignis auslösen, meldet System:

„Class 'Taschenrechner' not found...“

Wir werden diesen Fehler beheben indem wir die noch fehlende Klasse → Taschenrechner (unser Model erweitern und in der Bibliothek (→ lib.php) einbetten.

Model

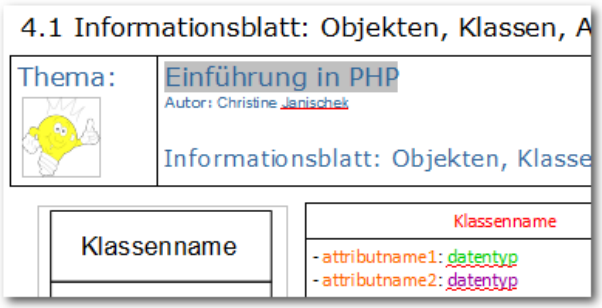
Fachklassen:
- → Bmirechner
- → Bmirechner (opt)
- → Darlehensrechner
- → Notenrechner
- → Rabattrechner
- → Taschenrechner



Reduzierte UML-Klasse: Taschenrechner

UML-Klasse → Taschenrechner: lib.php

→ zurück zum Arbeitsblatt



Verarbeitung

Nutzen Sie das Informationsblatt. Klären Sie die Begriffe und informieren Sie sich über das Grundgerüst einer Klasse in PHP.

Wir implementieren die Fachklasse → Noten-rechner, indem wir sie mit dem benötigten Quellcode ausstatten.

Entsprechend den Vorgaben (Anforderungen) der nebenstehend angezeigten UML-Klasse, werden wir das in den kommenden Schritten tun.

Taschenrechner
-$zahl1 -$zahl2 -$operator -$ergebnis
+ __construct() + set_zahl1($pZahl1) + set_zahl2($pZahl2) + set_operator($pOperator) + set_ergebnis($pErgebnis) + get_zahl1() + get_zahl2() + get_operator() + get_ergebnis() + addieren() + subtrahieren() + multiplizieren() + dividieren() + berechne()

Reduzierte UML-Klasse: Taschenrechner

Berechnungen:

```
addieren:
      ergebnis = zahl1 + zahl2
```

Implementierung in der Klasse Taschenrechner:

```php
public function addieren(){
    /*Rechenoperator in
     * lokales Attribut übernehmen*/
    $mOperator = "+";

    //Berechnung definieren
    $pErgebnis =  $this->get_zahl1()
            + $this->get_zahl2();

    /*Operator und Ergebnis in das
     * Objekt übermitteln*/
    $this->set_operator($mOperator);
    $this->set_ergebnis($pErgebnis);
}
```

Fachklasse implementieren.

Öffnen Sie dazu die Bibliotheksdatei → lib.php.

Integrieren Sie unterhalb der bereits enthaltenen Klasse(n) , die neue Fachklasse:
→ Taschenrechner

Gehen Sie vor wie zuvor für die Klasse → Notenrechner beschrieben:
→ Klasse und Attribute deklarieren. (lesen)
→ Konstruktor deklarieren.(lesen)
→ Get- und Set-Methoden deklarieren.(lesen)
→ Methode für die Berechnung(en). (lesen)

Hinweis:
Berücksichtigen Sie bitte unbedingt die Vorgaben aus der nebenstehenden UML-Klassen, um unnötige Syntaxfehler zu vermeiden.

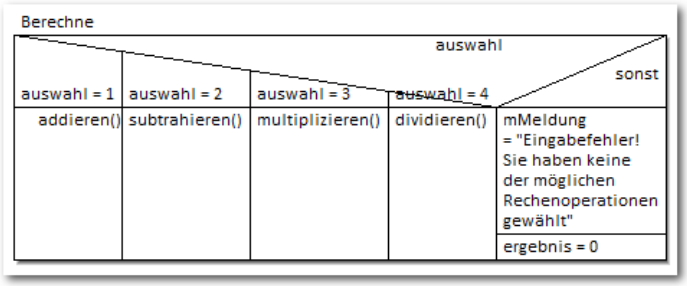
Implementieren Sie auch die Methoden:

```
subtrahieren:
      ergebnis = zahl1 - zahl2
```

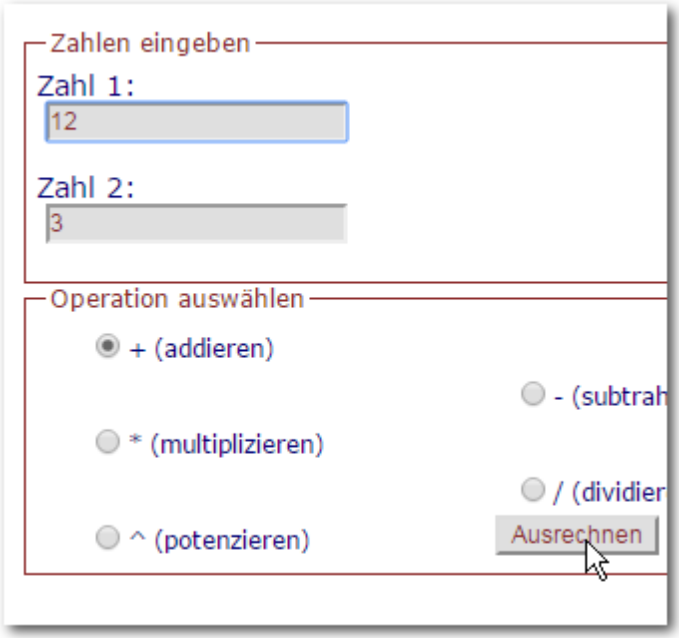
```
multiplizieren:
      ergebnis = zahl1 * zahl2
```

```
dividieren:
      ergebnis = zahl1 / zahl2
```

Für die Methode → berechne() soll gelten:



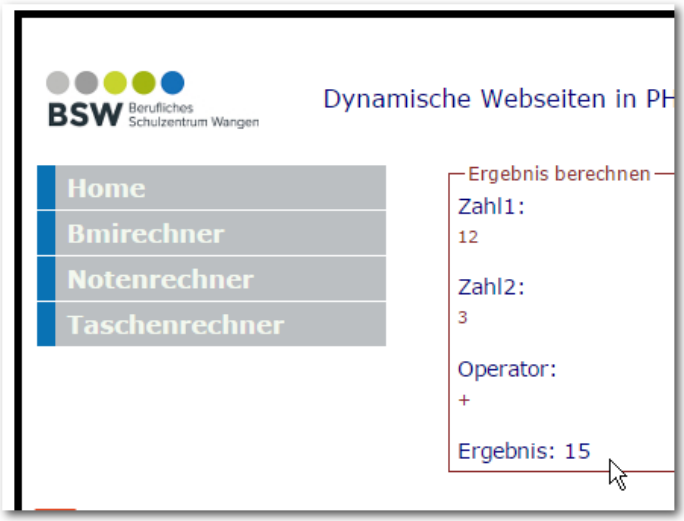
Wählen Sie eine geeignete Kontrollstruktur und implementieren Sie die Methode → berechne().



Hinweis: Falls Fehler angezeigt werden berücksichtigen Sie die folgenden Tipps:

→ Beheben Sie die Fehler von oben nach unten
→ Testen Sie nach jeder Korrektur
→ Nutzen Sie die Zeilen- und Dateiangabe
→ Prüfen Sie die Attribut- und Methodennamen
→ Prüfen Sie die Klammersetzung
→ Prüfen Sie ob Zeichen (z.B. „;“) fehlen

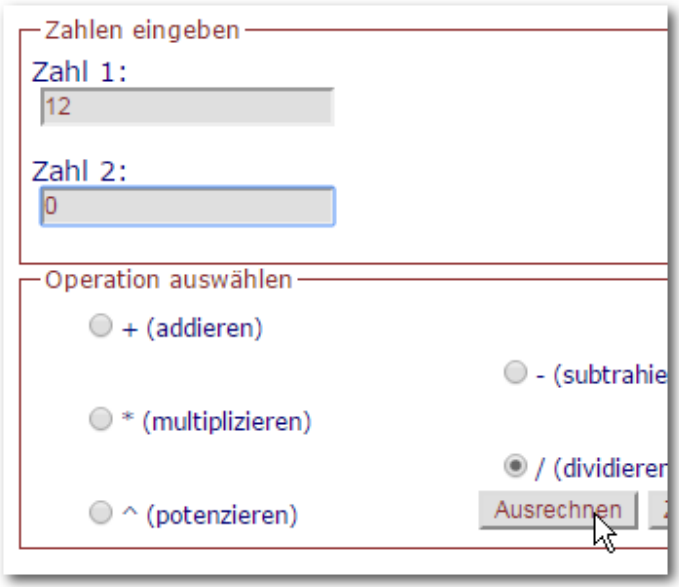
Testen Sie die Anwendung erneut.



Mit dieser Vorgehensweise können Sie künftig beliebig viele Klassen integrieren und an beliebiger Stelle im System nutzen.

Herzlichen Glückwunsch Sie haben die Anwendung objektorientiert implementiert.

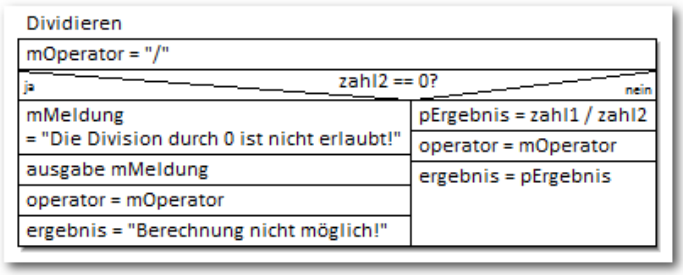
Lösen Sie im Anschluss an dieses Projekt, zum besseren Verständnis, das Aufgabenblatt:
→ Übung einfache Methoden und Fallunterscheidungen



Zusatzaufgabe

Testen Sie die Division durch 0, wie nebenstehenden angezeigt.

Implementieren Sie die Behandlung dieses Sonderfalls, indem Sie die Methode → dividieren() erweitern:



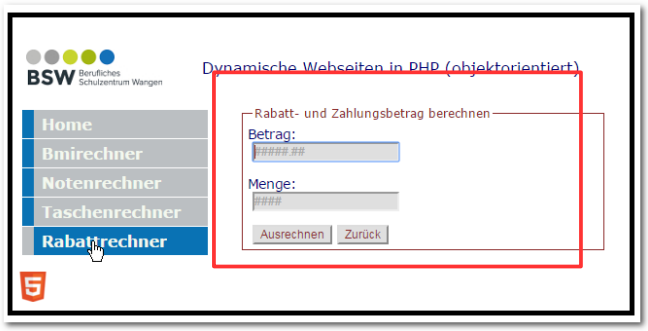
Testen Sie die Anwendung erneut!

7 Rabattrechner: Übung Fallunterscheidungen

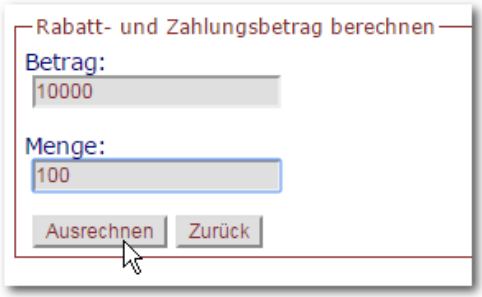
7.1 Informationsblatt: Rabattrechner

Thema:	Übung Fallunterscheidungen Autor: Christine Janischek Informationsblatt: Rabattrechner

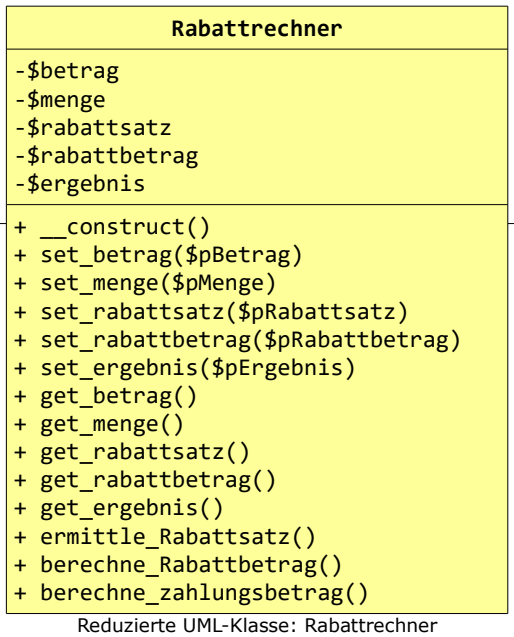
→ zurück zum Arbeitsblatt



Ergebnis der Übung: → rabattrechner.php (View)

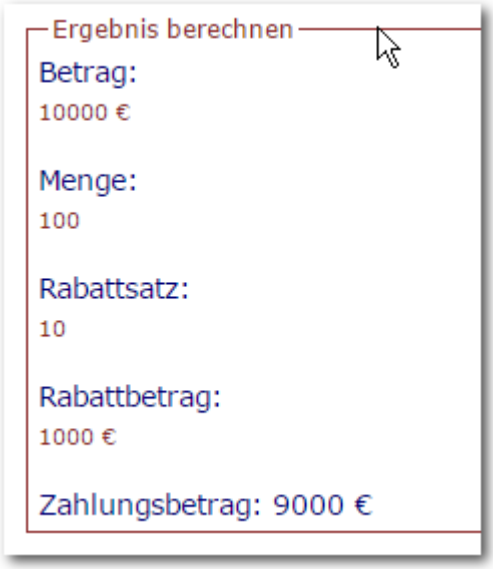


Eingabe-Formular: → rabatt.php (View)



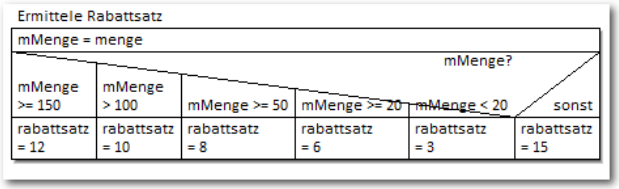
Reduzierte UML-Klasse: Rabattrechner

Bibliothek: → lib.php (Model)



Ausgabe-Datei: → rabatt1.php (Controller)

Ermittlung des Rabattsatzes:



Berechnungen

Des Rabattbetrags:

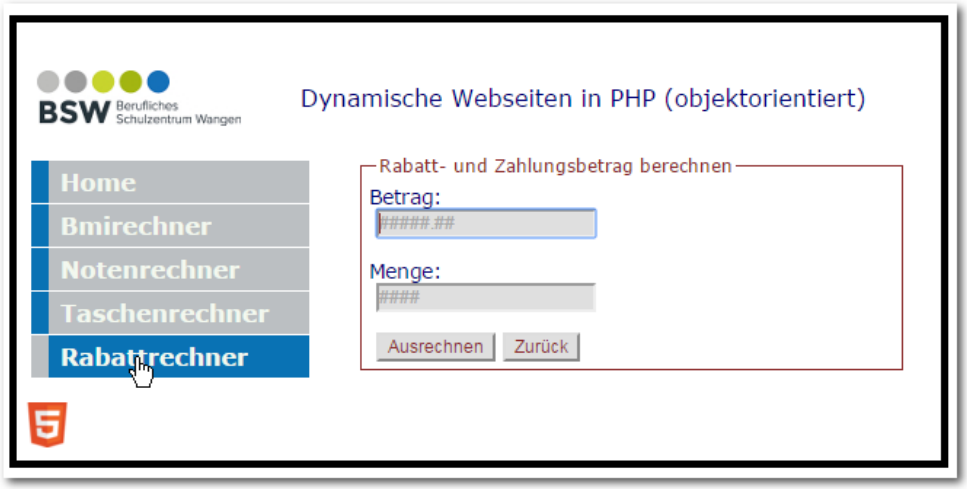
```
rabattbetrag = betrag/100 * rabattsatz;
```

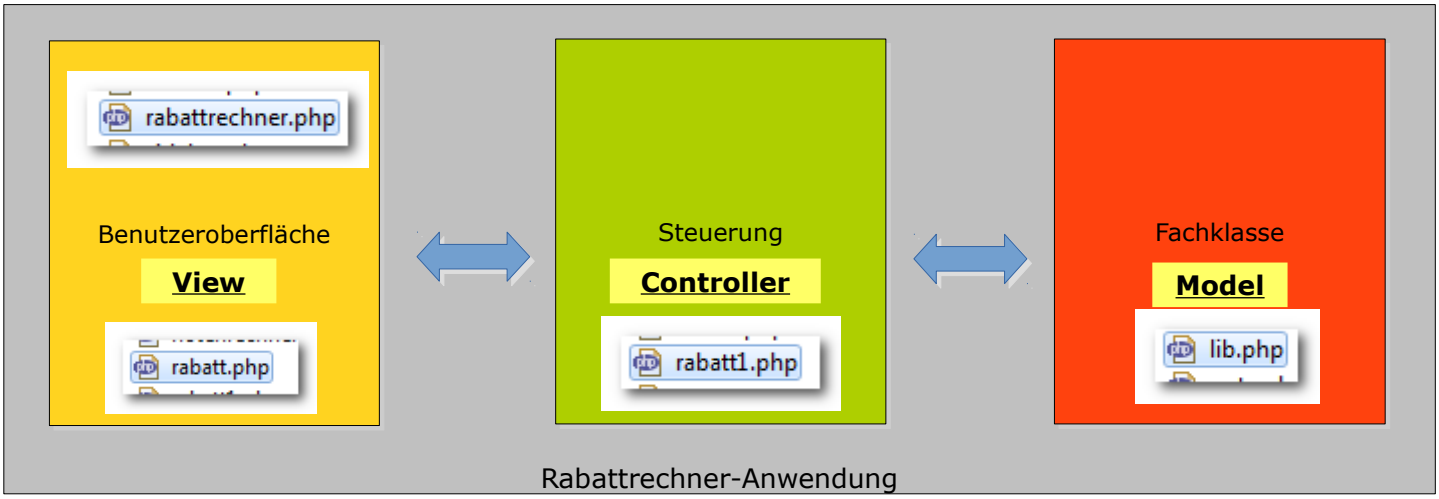
Des Zahlungsbetrags:

```
zahlungsbetrag = $betrag - rabattbetrag;
```

7.2 Arbeitsblatt: Rabattrechner

Thema:	Übung Fallunterscheidungen Autor: Christine Janischek Arbeitsblatt: Rabattrechner





Arbeitsauftrag:

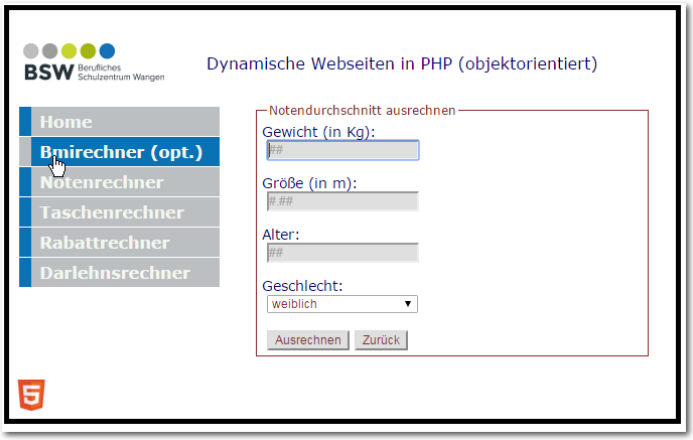
1. Erzeugen Sie ein neues Projekt um den → Rabattrechner umzusetzen.
2. Implementieren Sie den Rabattrechner und berücksichtigen Sie dazu die Vorgaben auf dem Informationsblatt.
3. Dokumentieren Sie welche einzelnen Schritte für die Umsetzung der View, des Controllers und des Models notwendig sind.

8 Bmirechner: Übung Fallunterscheidung

8.1 Informationsblatt: Bmirechner Erweiterung

Thema:	Übung Fallunterscheidungen Autor: Christine Janischek Informationsblatt: Bmirechner Erweiterung

→ zurück zum Arbeitsblatt



Ergebnis der Übung: → bmirechner.php (View)

Eingabe-Formular: → bmi.php (View)

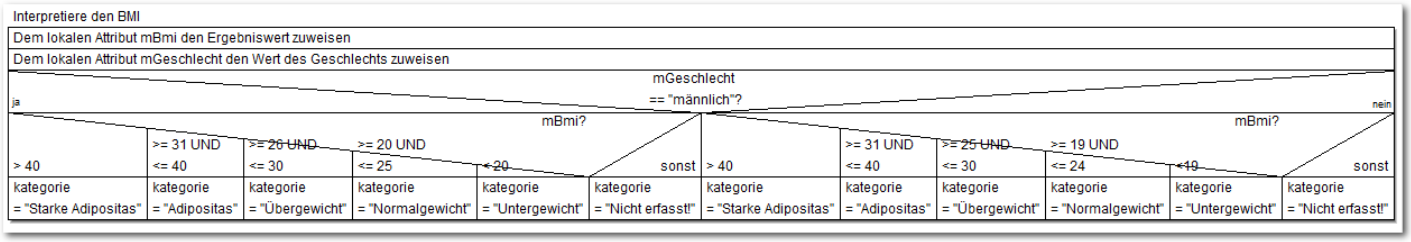
Bibliothek:
→ lib.php (Model)

Bmirechner
-$gewicht -$groesse -$alter -$geschlecht -$kategorie -$ergebnis
+ __construct() + set_gewicht($pGewicht) + set_groesse($pGroesse) + set_alter($pAlter) + set_geschlecht($pGeschlecht) + set_kategorie($pKategorie) + set_ergebnis($pErgebnis) + get_groesse() + get_gewicht() + get_alter() + get_geschlecht() + get_kategorie() + get_ergebnis() + berechne_bmi() + interpretiere_bmi()

Reduzierte UML-Klasse: Bmirechner

Ausgabe-Datei: → bmi1.php (Controller)

Für die Interpretation des BMI (die Ermittlung der Kategorie):



Tabellarische Darstellung:
Quelle: http://www.bmi-rechner.net/bmi-tabelle.htm

	BMI männlich	BMI weiblich
Untergewicht	unter 20	unter 19
Normalgewicht	20-25	19-24
Übergewicht	26-30	25-30
Adipositas	31-40	31-40
starke Adipositas	größer 40	größer 40

Formular-Komponente ☐Drop-Down-Menü für die Auswahl des Geschlechts

```
<select name='ddGeschlecht' id="ddGeschlecht"
      required="required" autofocus="autofocus" size="1">
      <option>weiblich</option>
      <option>m&auml;nnlich</option>
</select>
```

Für die Besimmung des optimalen BMI

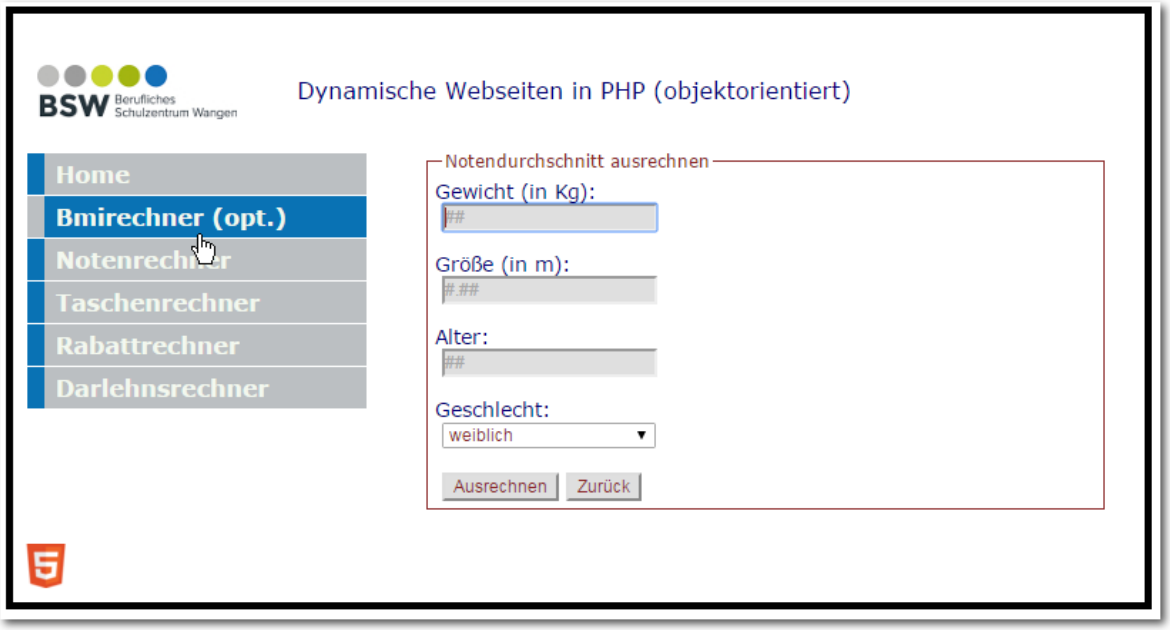
Tabellarische Darstellung:

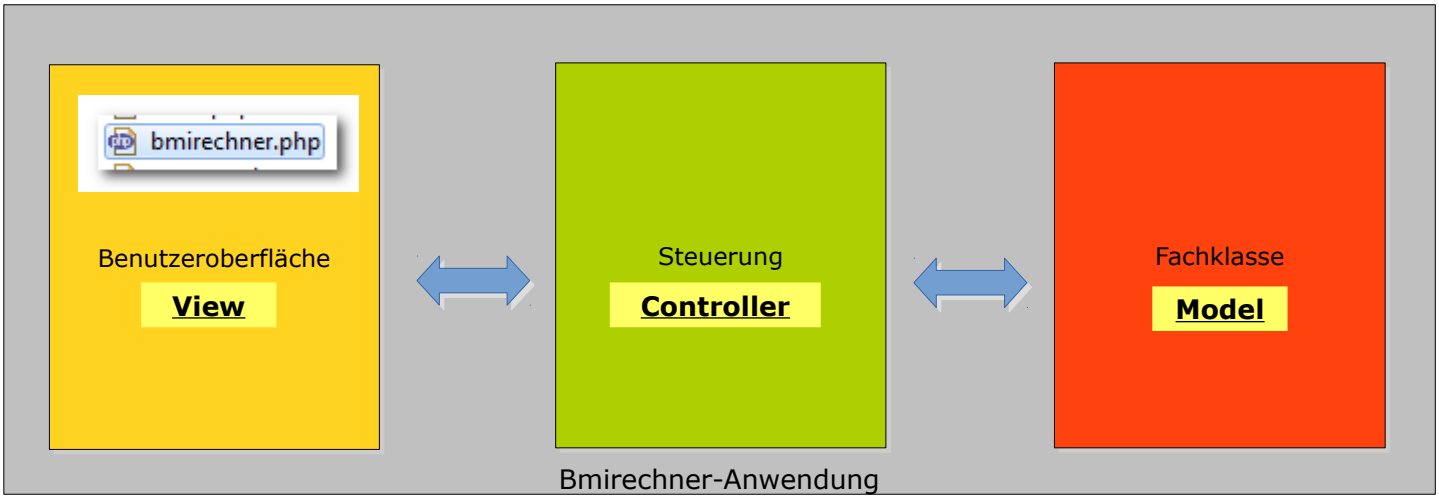
Alter	optimaler BMI
19-24	19-24
25-34	20-25
35-44	21-26
45-54	22-27
55-64	23-28
älter als 65	24-29

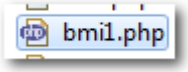
Quelle: http://www.bmi-rechner.net/bmi-tabelle.htm

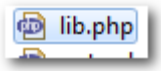
8.2 Arbeitsblatt: Bmirechner Erweiterung

Thema:	Übung Fallunterscheidungen Autor: Christine Janischek Arbeitsblatt: Bmirechner Erweiterung









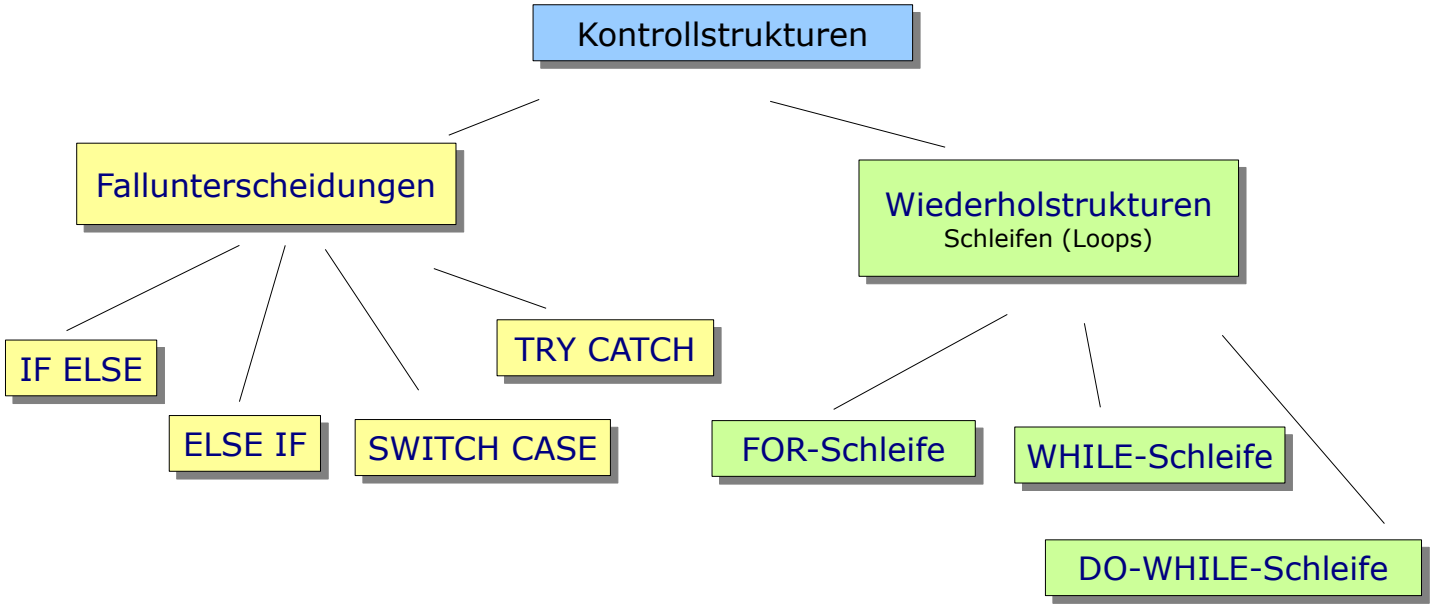
Arbeitsauftrag:

1. Erzeugen Sie ein neues Projekt um den → Bmirechner zu erweitern.
2. Implementieren Sie die Erweiterung und berücksichtigen Sie dazu die Vorgaben auf dem Informationsblatt.
3. Dokumentieren Sie welche einzelnen Schritte für die Umsetzung der View, des Controllers und des Models notwendig sind.

9 Darlehensrechner: Wiederholstrukturen

9.1 Informationsblatt: Kontrollstrukturen → Wiederholstrukturen

Thema:	Kontrollstrukturen: Wiederholstrukturen Autor: Christine Janischek Informationsblatt: Überblick zu Wiederholstrukturen



Hinweis: Wiederholstrukturen sind Bestandteil von Verhaltensweisen (Methoden)		
Wiederholstrukturen	**Grundgerüst**	**Besonderheit**

FOR-SCHLEIFE	`for (ausdruck1; ausdruck2; ausdruck3){`
	`  //Anweisung(en)`
	`}`
	
	Beispiel:
	`for($i = 10;$i > 0;$i--) {`
	`  echo $i;`
	`}`
		Ist die komplexeste Schleife.
		
		Ausdruck1 wird vor der Schleife ausgeführt und enthält in der Regel die Deklaration und Initatialisierung der Schleifenzählervariablen.
		
		Am Anfang eines jeden Schleifendurchlaufs wird die Anweisung in Ausdruck2 ausgeführt. Wenn diese → True ist wird die Schleife fortgesetzt

		und die darunterliegenden Anweisungen werden ausgeführt. Anderenfalls (→ False) wird der Vorgang abgebrochen. Am Ende eines jeden Schleifendurchlaufs wird die Anweisung in Ausdruck3 ausgeführt.

WHILE-SCHLEIFE	`while (bedingung){`
	`  //Anweisung(en)`
	`  //Abbruchbedingung`
	`}`
	
	Beispiel:
	`$i = 10;`
	`while($i > 0) {`
	`  echo $i;`
	`  $i--;`
	
	`}`
		Ist die einfachste Schleife.
		
		Die Anweisungen innerhalb der Schleife werden wiederholt ausgeführt, solange die Bedingung zutrifft, also zu → True evaluiert.
		
		Eine Abbruchbedingung z.B. ein Zähler stellt in der Regel sicher, dass die Schleife nicht endlos ausgeführt wird, die Bedingung zu gegebener Zeit zu → False evaluiert und der Vorgang abgebrochen wird.

DO-WHILE-SCHLEIFE	`do{`
	`  //Anweisung(en)`
	`  //Abbruchbedingung`
	`}while(bedingung);`
	
	Beispiel:
	`$i = 10;`
	`do {`
	`  echo $i;`
	`  $i--;`
	
	`} while ($i > 0);`
		Ist der While-Schleife sehr ähnlich.
		
		Der Unterschied liegt daran, dass die Bedingung erst am Ende des Schleifendurchlaufs geprüft wird.

9.2 Informationsblatt: Darlehensrechner

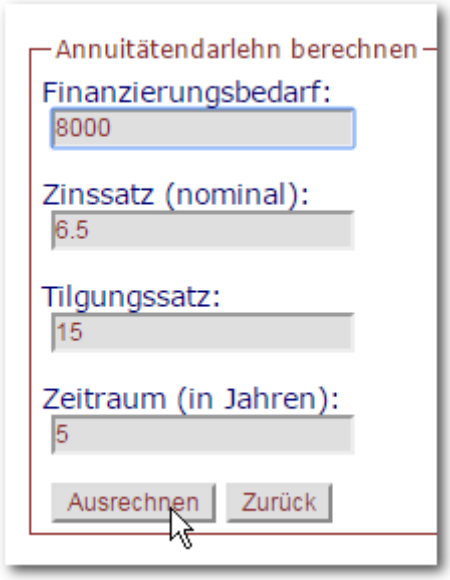
View

Box-Modell:
→ Bmirechner
→ Bmirechner (opt)
→ Darlehensrechner
→ Notenrechner
→ Rabattrechner
→ Taschenrechner

Eingabe-Formular erzeugen: darlehensrechner.php

→ zurück zum Arbeitsblatt

Ergebnis der Übung: → darlehensrechner.php (View)

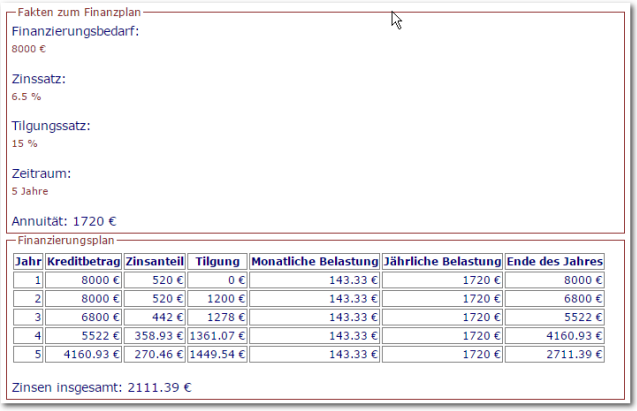


Eingabe-Formular: → darlehen.php (View)

Darlehensrechner
-$finanzierungsbedarf -$zinssatz -$tilgungssatz -$zeitraum -$jahr = 1 -$kreditbetrag -$zinsanteil_jahr -$tilgung_jahr -$belastung_monat -$belastung_jahr -$kontostand_ende_jahr -$summme_zinsanteile -$ergebnis
+ __construct() Getter + Setter aller Attribute + berechne_Annuitaet() + ermittle_Kreditbetrag() + berechne_Zinsanteil() + ermittle_Belastung_jahr() + ermittle_Belastung_mon() + berechne_Tilgung_jahr() + berechne_Kontostand_ende_jahr() + berechne_Zinsanteile_kummuliert()

Reduzierte UML-Klasse: Darlehensrechner

Bibliothek: → lib.php (Model)



Jahr	Kreditbetrag	Zinsanteil	Tilgung	Monatliche Belastung	Jährliche Belastung	Ende des Jahres
1	8000 €	520 €	0 €	143.33 €	1720 €	8000 €
2	8000 €	520 €	1200 €	143.33 €	1720 €	6800 €
3	6800 €	442 €	1278 €	143.33 €	1720 €	5522 €
4	5522 €	358.93 €	1361.07 €	143.33 €	1720 €	4160.93 €
5	4160.93 €	270.46 €	1449.54 €	143.33 €	1720 €	2711.39 €

Zinsen insgesamt: 2111.39 €

Ausgabe-Datei: → darlehen1.php (Controller)

```
/*Tabelle*/

th,td{
	border: 1px solid grey;/*Rahmen keiner*/
	font-size: 75%;/*relative Schriftgröße*/
}

td{
	text-align: right;
}
```

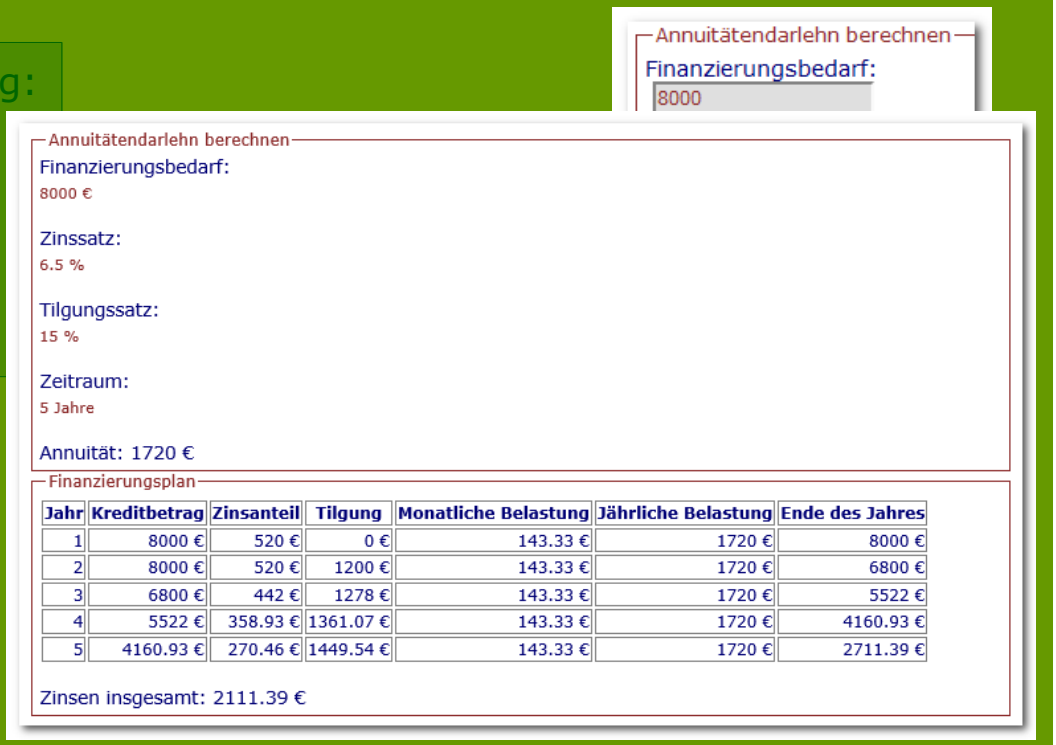
Stylesheet-Angaben für
die Formatierung der Tabelle → style.css

Controller

Ereignissteuerung: darlehen1.php

Ereignissteuerung:

- → Bmirechner
- → Bmirechner (opt)
- → Darlehensrechner
- → Notenrechner
- → Rabattrechner
- → Taschenrechner



Jahr	Kreditbetrag	Zinsanteil	Tilgung	Monatliche Belastung	Jährliche Belastung	Ende des Jahres
1	8000 €	520 €	0 €	143.33 €	1720 €	8000 €
2	8000 €	520 €	1200 €	143.33 €	1720 €	6800 €
3	6800 €	442 €	1278 €	143.33 €	1720 €	5522 €
4	5522 €	358.93 €	1361.07 €	143.33 €	1720 €	4160.93 €
5	4160.93 €	270.46 €	1449.54 €	143.33 €	1720 €	2711.39 €

→ zurück zum Arbeitsblatt

```
Anwendungsfall:

Der Benutzer gibt die Eingabewerte ein. Wenn der Benutzer auf die Schaltfläche → Ausrechnen klickt
werden die folgenden Schritte ausgeführt (darlehen1.php):

/*SCHRITT-FÜR-SCHRITT zur gewünschten Ausgabe*/
/*######EINGABE###*/
   1. Lesen aller Eingabewerte. Dazu werden, wie sonst auch üblich, alle Werte aus den Textfeldern
      an lokale Attribute übermittelt.
/*######VERARBEITUNG###*/
   2. Erzeugung eines Objektes der Klasse Darlehensrechner
   3. Übermitteln der Eigenschaftswerte an das Objekt der Fachklasse.
   4. Berechnung der Annutität
   5. Dem lokalen Attribut → $ausgabe wird die berechnete Annuität (→ ergebnis) zugewiesen.
/*######VERARBEITUNG UND AUSGABE###*/
   6. Ausgabe des Finanzierungsbedarfs
   7. Ausgabe des Zinssatzes
   8. Ausgabe des Tilgungssatzes
   9. Ausgabe des Zeitraums
   10. Ausgabe der Annuität
   11. Ausgabe
         ◦ des Schließ-Tag des aktuellen Fieldsets,
         ◦ öffnen des neuen Fieldsets,
         ◦ die Legenden-Box mit Inhalt
         ◦ das Eröffnungstag der Tabelle und
```

→ Teil1: darlehen1.php

→ Teil2: darlehen1.php

- die Kopfzeile der Tabelle

12. Das aktuelle Jahr und den aktuellen Zeitraum jeweils in ein lokales Attribut übernehmen.
13. Solange der aktuelle Zeitraum größer 0 ist soll
    - der Wert des aktuellen Jahres an das Objekt der Klasse übermittelt
    - der Kreditbetrag ermittelt
    - der Zinsanteil berechnet
    - die jährliche Belastung ermittelt
    - die monatliche Belastung ermittelten
    - die Tilgung pro Jahr berechnet
    - der Kontostand am Ende des Jahres berechnet
    - die Zinsanteile kumuliert werden.
    - Des weiteren soll
        - der Kreditbetrag
        - der Zinsanteil
        - die jährliche Tilgung
        - die monatliche Belastung
        - die jährliche Belastung
        - der Kontostand am Ende des Jahres und die
        - Summe der Zinsanteile
    - anschließend lokalen Attributen zugewiesen werden
    - und alle Werte in einer Tabellenzeile ausgegeben werden. Jeder der folgenden Werte soll dazu in eine Spalte geschrieben werden
        - das aktuelle Jahr
        - der Kreditbetrag gerundet auf zwei Nachkommastellen
        - der Zinsanteil gerundet auf zwei Nachkommastellen
        - jährliche Tilgung gerundet auf zwei Nachkommastellen
        - die monatliche Belastung gerundet auf zwei Nachkommastellen
        - die jährliche Belastung gerundet auf zwei Nachkommastellen
        - und den Kontostand am Ende des Jahres gerundet auf zwei Nachkommastellen
    - abschließend soll das aktuelle Jahr inkrementiert („hochgezählt“),
    - der Zeitraum dekrementiert („runtergezählt“) und der Wiederholvorgang geschlossen werden.
14. Erzeugen Sie abschließend die Ausgabe mit dem Schließ-Tag für die Tabelle und der Ausgabe für die Summe der Zinsanteile.

Ergänzen Sie die Lücken im Quellcode, übernehmen und testen Sie den Quellcode für die Controller-Datei → darlehen1.php im Anschluss!

Teil 1: darlehen1.php

```
13      <?php
14      //Eingaben lesen
15                  -1-                                                    ;
16                  -2-
17              $pTilgungssatz = $_POST['tfTilgungssatz'];
18              $pZeitraum = $_POST['tfZeitraum'];
19
20
21      //Verarbeitung
22          //Objekt erzeugen
23          $dieDaten = new Darlehnsrechner();
24
25                  -3-
26                  -4-
27          $dieDaten -> set_zinssatz($pZinssatz);
28          $dieDaten -> set_tilgungssatz($pTilgungssatz);
29          $dieDaten -> set_zeitraum($pZeitraum);
30
31          $dieDaten -> berechne_Annuitaet();
32          $ausgabe =  $dieDaten -> get_ergebnis();
33
34                  -5-
35      echo "Finanzierungsbedarf: <h5>".$dieDaten -> get_finanzierungsbedarf()." €</h5>";
36      echo "Zinssatz: <h5>".$dieDaten -> get_zinssatz()." %</h5>";
37      echo "Tilgungssatz: <h5>".$dieDaten -> get_tilgungssatz()." %</h5>";
38      echo "Zeitraum: <h5>".$dieDaten -> get_zeitraum()." Jahre</h5>";
39      echo "Annuit&auml;t: ".$ausgabe." €";
```

→ zurück zum Anwendungsfall

Studieren Sie den Anwendungsfall und ergänzen Sie die fehlenden Quellcodezeilen	
-1-	
-2-	
-3-	
-4-	
-5-	
-6-	

Teil 2: darlehen1.php

```
40
41                  //Ausgabe: Tabellenkopf für den Finanzplan ausgeben
42                  echo "</fieldset>
43                  <fieldset>
44                      <legend>Finanzierungsplan</legend>
45                      <table>
46                          <tr>
47                                                  -7-
48                                                  -8-
49                                                  -9-
50                              <th>Tilgung</th>
51                              <th>Monatliche Belastung</th>
52                              <th>J&auml;hrliche Belastung</th>
53                              <th>Ende des Jahres</th>
54                          </tr>";
55
56
57                          //Aktuelles Jahr und Aktuellen Zeitraum in lokales Attribut übernehmen
58                          $mJahr = $dieDaten -> get_jahr();
59                          $mZeitraum = $dieDaten -> get_zeitraum();
60
61                          //Ausgabe mit Schleife: Finanzplan, zeilenweise
62                          while(      -10-      ){
63
64                              //Zeilenweise (jedes Jahr) Berechnung der Teilergebnisse
65                              $dieDaten -> set_jahr($mJahr);
66                              $dieDaten -> ermittle_Kreditbetrag();
67                                                  -11-
68                                                  -12-
69                              $dieDaten -> ermittle_Belastung_mon();
70                              $dieDaten -> berechne_Tilgung_jahr();
71                              $dieDaten -> berechne_Kontostand_ende_jahr();
72                              $dieDaten -> berechne_Zinsanteile_kummuliert();
```

→ zurück zum Anwendungsfall

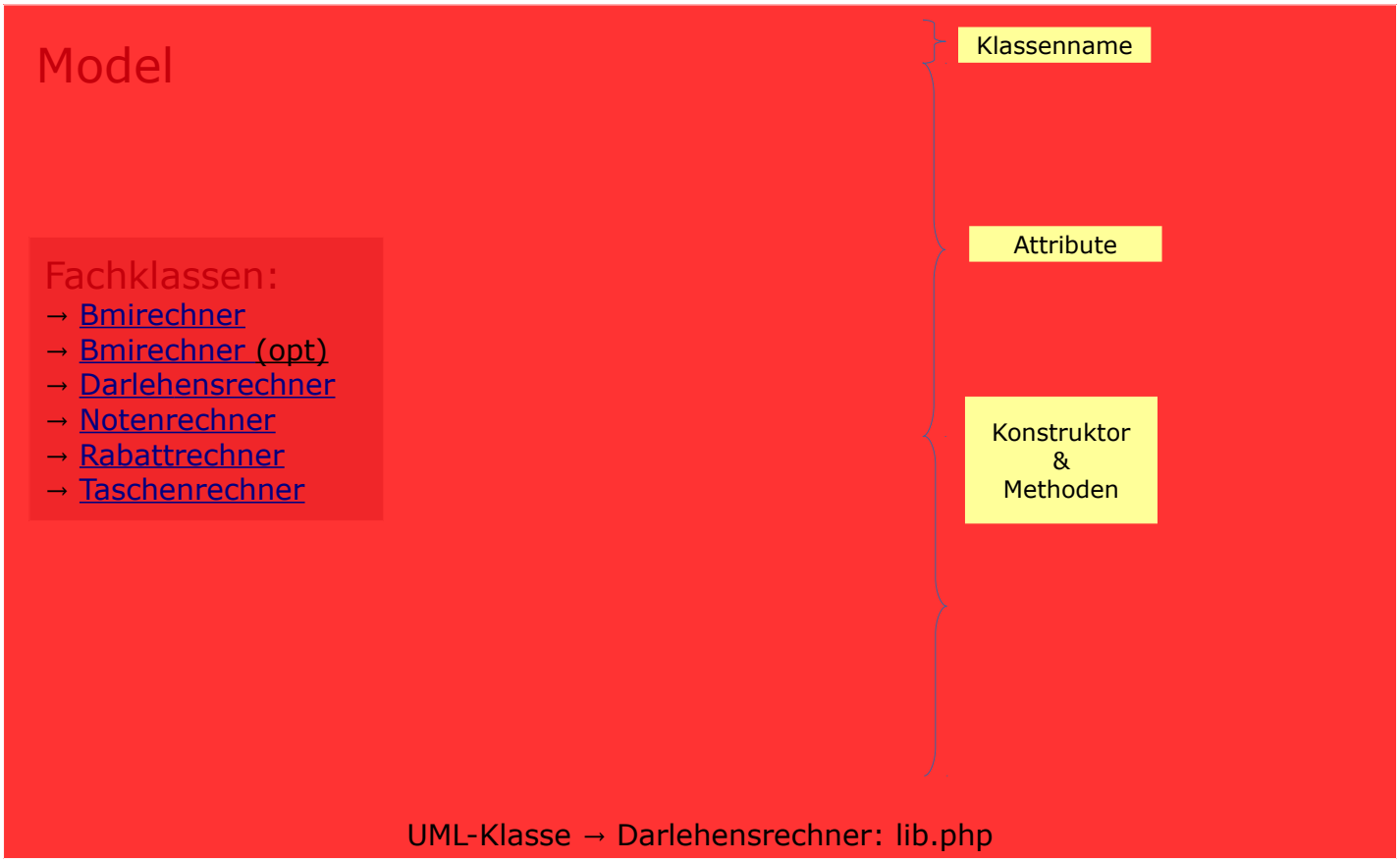
Studieren Sie den Anwendungsfall und ergänzen Sie die fehlenden Quellcodezeilen	
-7-	
-8-	
-9-	
-10-	
-11-	
-12-	

Teil 3: darlehen1.php

```
73
74                              //Zeilenweise (jedes Jahr) Ermittlung der Teilergebnisse
75                                                  -13-
76                                                  -14-
77                              $mTilgung_jahr = $dieDaten -> get_tilgung_jahr();
78                              $mBelastung_mon = $dieDaten -> get_belastung_monat();
79                              $mBelastung_jahr = $dieDaten -> get_belastung_jahr();
80                              $mKontostand_ende_jahr = $dieDaten -> get_kontostand_ende_jahr();
81                              $mSumme_zinsanteile = $dieDaten -> get_summe_zinsanteile();
82
83                              //Zeilenweise (jedes Jahr) die gerundeten Teilergebnisse ausgeben
84                              echo "<tr>
85                                      <td>".$mJahr."</td>
86                                      <td>".round($mKreditbetrag,2)." €</td>
87                                                  -15-
88                                                  -16-
89                                      <td>".round($mBelastung_mon,2)." €</td>
90                                      <td>".round($mBelastung_jahr,2)." €</td>
91                                      <td>".round($mKontostand_ende_jahr,2)." €</td>
92                                  </tr>";
93
94                                  //Jahr inkrementieren
95                                                  -17-
96
97                                  //Abbruchbedingung für die Schleife, Zeitraum dekrementieren
98                                                  -18-
99                          }
100                     echo "</table> <br />Zinsen insgesamt: ".round($mSumme_zinsanteile,2)." €
101
102             </fieldset>
103         </form>
104     </div>";
105     ?>
```

→ zurück zum Anwendungsfall

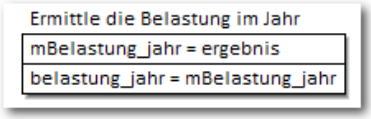
Studieren Sie den Anwendungsfall und ergänzen Sie die fehlenden Quellcodezeilen	
-13-	
-14-	
-15-	
-16-	
-17-	
-18-	

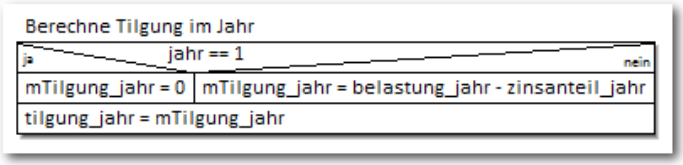


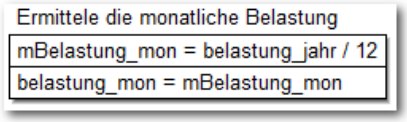
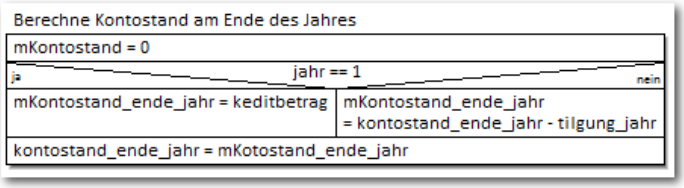
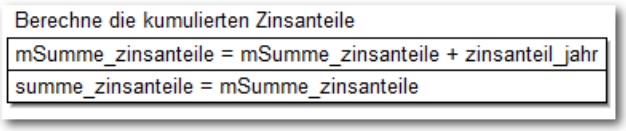
UML-Klasse → Darlehensrechner: lib.php

→ zurück zum Arbeitsblatt

Berechne Annuität: Berechne Annuität mAnnuitaet = finanzierungsbedarf * (zinssatz + tilgungssatz)/100 ergabnis = mAnnuitaet	*Der Annuität.* Im vorliegenden Fall handelt es sich um ein Annuitäten-darlehen. Mittels der folgenden Berechnung kann die Annuität er-mittelt werden. Formel: ergebnis = finanzierungsbedarf * (zinssatz + tilgungssatz)/100
Ermittle Kreditbetrag: Ermittle Kreditbetrag ja / jahr == 1 / nein mKreditbetrag = finanzierungsbedarf | mKreditbetrag = kontostand_ende_jahr kreditbetrag = mKreditbetrag | kreditbetrag = mKreditbetrag	*Der Kreditbetrag.* Für das Erste Jahr soll für den Kreditbetrag der eingege-bene Finanzierungsbedarf übernommen werden. Für alle anderen Jahre soll für den Kreditbetrag der Kontostand am Ende des Jahres übernommen werden.
Berechne den Zinsanteil: Berechne den Zinsanteil mZinsanteil = kreditbetrag * zinssatz /100 zinsanteil_jahr = mZinsanteil	*Der Zinsanteil.* `zinsanteil = kreditbetrag * zinssatz /100`

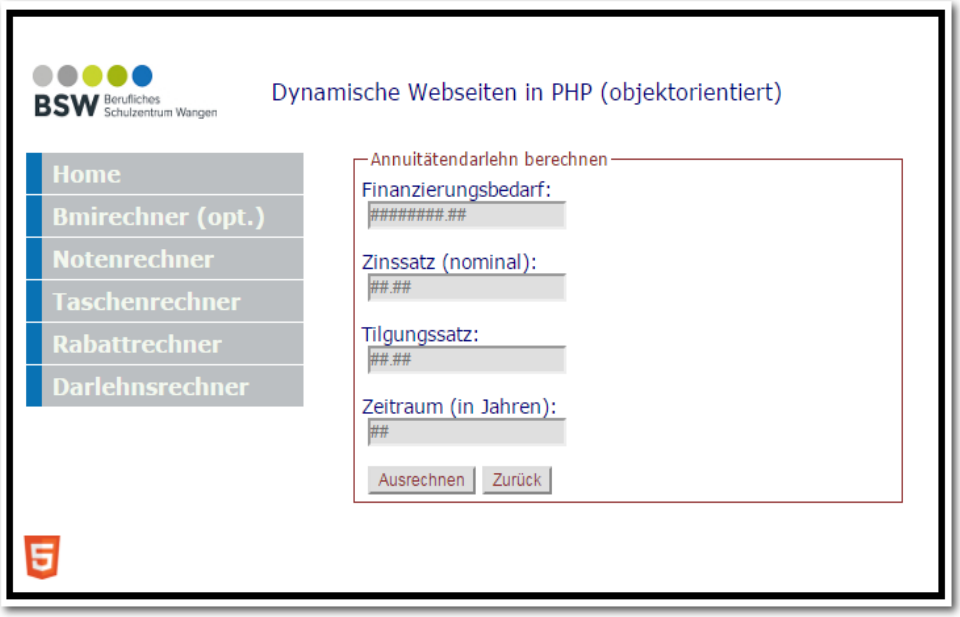
Ermittle Belastung im Jahr: 	*Die Belastung im Jahr.* Entspricht beim Annuitätendarlehen der Annuität.

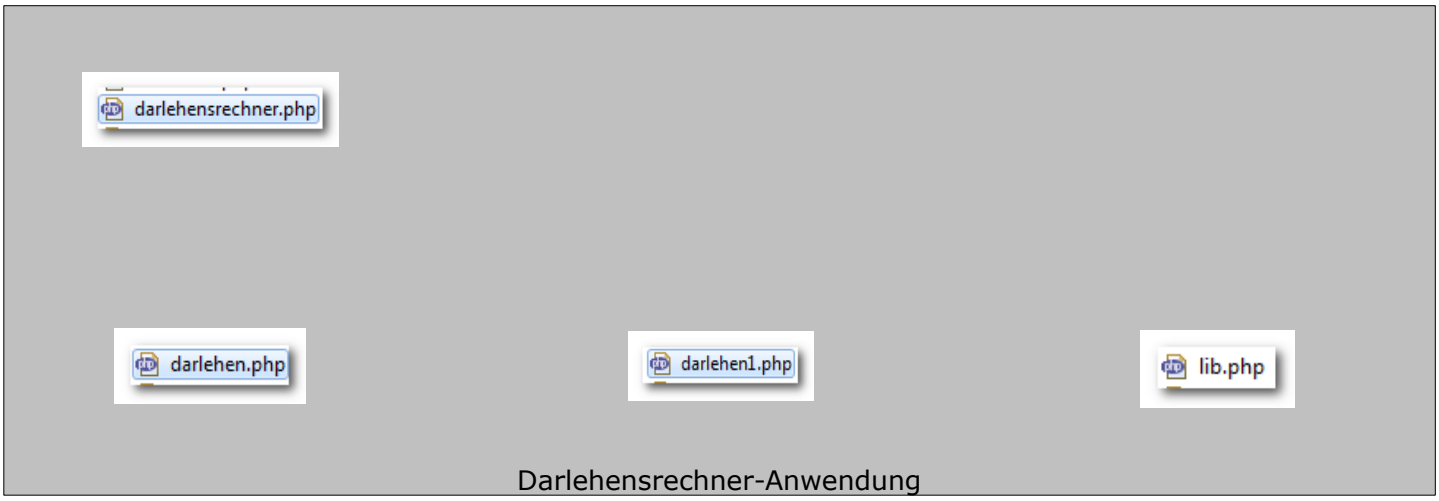
Ermittle Tilgung im Jahr
 
	Die Tilgung im Jahr.
	
	Für das Erste Jahr soll keine Tilgung anfallen:
	`tilgung im jahr = 0;`
	
	Anderenfalls soll die Tilgung berechnet werden:
	`tilgung im jahr`
	`= belastung im jahr – zinsantal im jahr`

Ermittle Belastung im Monat: 	Die Belastung im Monat. Entspricht 1/12 der jährlichen Belastung. `belastung im monat= belastung im jahr/12;`
Ermittle Kontostand am Ende des Jahres: 	*Der Kontostand am Ende des Jahres.* Im Ersten Jahr soll der Kontostand am Ende des Jahres dem Kreditbetrag entsprechen. In allen folgenden Jahren ergibt sich der Kontostand am Ende des Jahres aus dem Kontostand am Ende des letzten Jahres abzüglich der Tilgung im Jahr.
Berechne Zinsanteile kumuliert: 	*Die kumulierten Zinsanteile.* Die wichtigste Frage für einen Kreditnehmer ist immer: „Was ist der Preis für den Kredit?“ Der Preis ergibt sich aus den Zinszahlungen, welche über die gesamte Laufzeit hinweg gezahlt werden müssen. Summiert man diese Beträge auf ergibt sich die Summe aller Zinsanteile.

9.3 Arbeitsblatt: Darlehensrechner

Thema:	Kontrollstrukturen: Wiederholstrukturen Autor: Christine Janischek Arbeitsblatt: Darlehensrechner





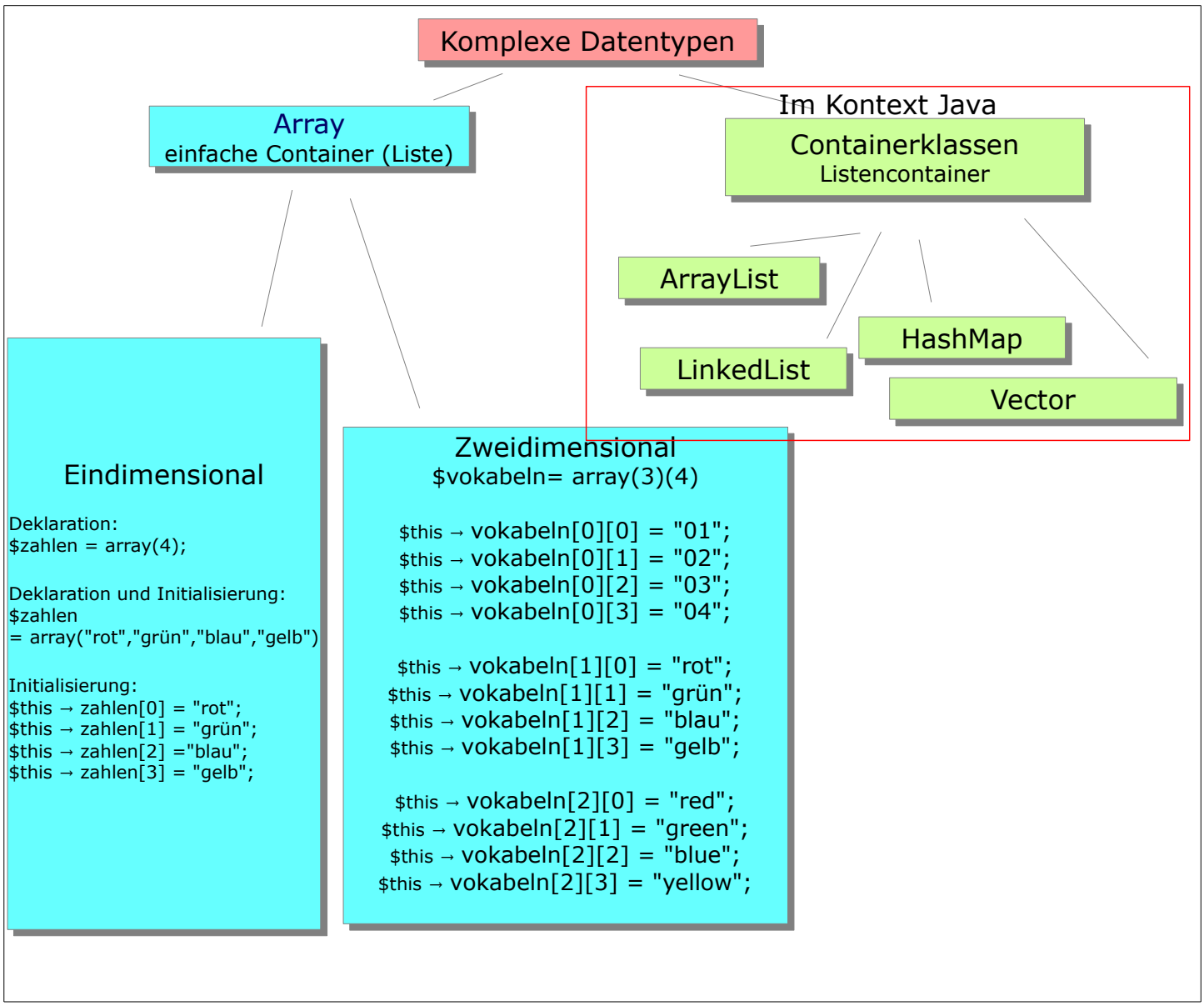
Darlehensrechner-Anwendung

Arbeitsauftrag:

1. Erzeugen Sie ein neues Projekt um den → Darlehensrechner umzusetzen.
2. Implementieren Sie den → Darlehensrechner und berücksichtigen Sie dazu die Vorgaben und Hilfestellungen auf dem Informationsblatt.
3. Dokumentieren Sie welche einzelnen Schritte für die Umsetzung der View, des Controllers und des Models notwendig sind.

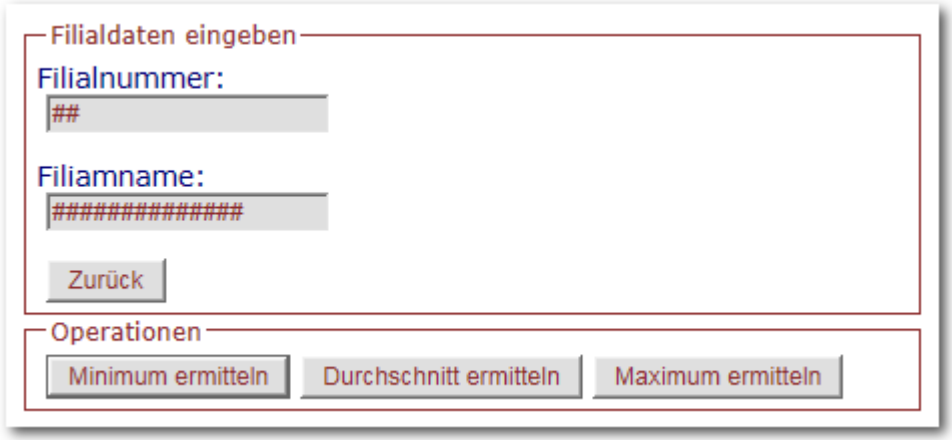
10 Umsatzrechner: Übung Wiederholstrukturen

10.1 Informationsblatt: Datencontainer



10.2 Arbeitsblatt: Umsatzrechner

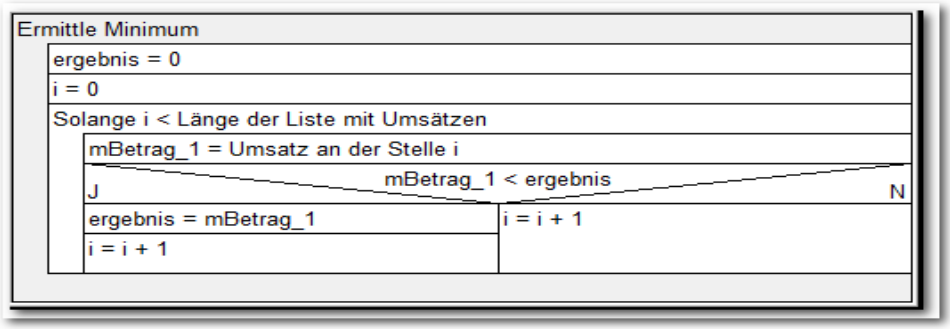
Thema: 	Kontrollstrukturen: Wiederholstrukturen Autor: Christine Janischek Arbeitsblatt: Umsatzrechner

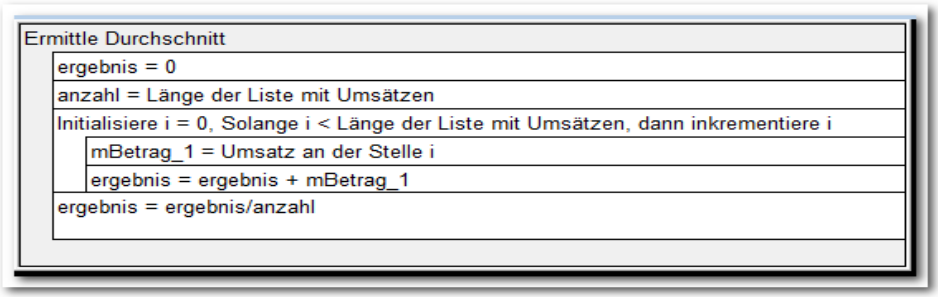


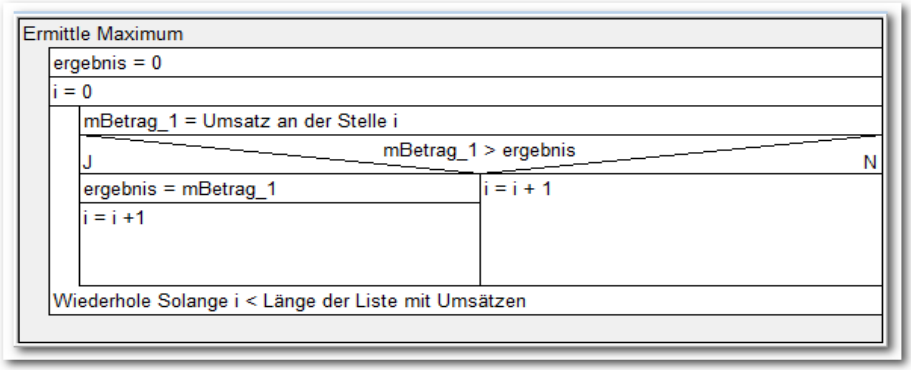
Filiale
- Filialname:int - Filialname:String - umsatz:double[] = new double[12] - ergebnis:double
+ Filiale() + getFilialnummer():String + setFilialnummer(String pFilialnummer) + getFilialname():String + setFilialname(String pFilialname) + getErgebnis():double + setErgebnis(double pErgebnis) + ermittleMinimum():void + ermittleDurchschnitt():void + ermittleMaximum():void

UML-Klasse: Filiale

Ihr Unternehmen hat mittlerweile einige Filialen. Für jede Filiale wird der Umsatz der letzten zwölf Monate in einer einfachen Liste erfasst.





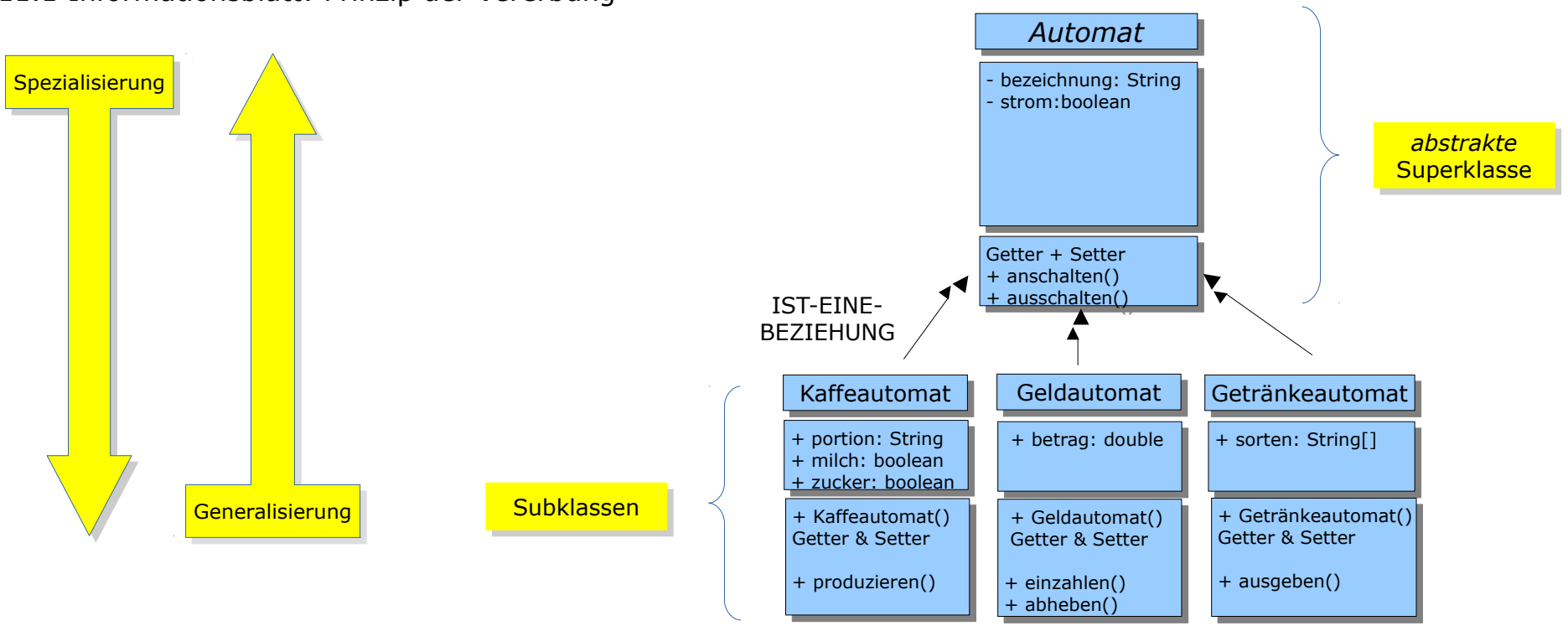


Arbeitsauftrag:

1. Initialisieren Sie die Liste Umsatz der Filiale in der Jahnstrasse 15 mit den folgenden Werten:
   1000,1500,1100, 1200, 1150, 950,500,8800, 16000, 433, 8000, 5000
2. Welche Arten von Listentypen (Container) kennen Sie?
3. Nennen Sie fünf Kontrollstrukturen.
4. Erstellen Sie anhand der vorgegebenen Struktogramme den Quellcode für die Verhaltensweisen: ermittle Minimum, ermittle Maximum und ermittle Durchschnitt den Programmcode. Testen Sie den Quellcode!
5. Es hat sich ein Denkfehler eingeschlichen! Identifizieren Sie den Fehler und finden Sie einen Lösungsweg.
6. Welche Art Kontrollstrukturen eignen sich für die Behandlung des geschilderten Problems?

11 Rechner: Optimierung durch Vererbung

11.1 Informationsblatt: Prinzip der Vererbung



→zurück zum Arbeitsblatt

11.2 Arbeitsblatt: Gemeinsamkeiten und Unterschiede

Thema: 	Vererbung Autor: Christine Janischek Arbeitsblatt: Gemeinsamkeiten und Unterschiede

Superklassen	*Vererbende Klasse.*

`abstract class Automat{`
 	
`//Kein Konstruktor da Abstrakt`
`...`
 	
`}`
	Deklariert werden alle Eigenschaften und Verhaltensweisen die für alle Objekte der Subklassen gelten.
	
	Abstrakte:
	Abstrakte Klassen dienen rein der Verwaltung der Gemeinsamkeiten (Eigenschaften und Verhaltensweisen). Von abstrakten Klassen können wir kein Objekt erzeugen. Der Automat selbst existiert also nicht! Es gibt also ausschließlich spezielle Automaten.

Subklassen	*Erbende Klasse.*

`class Kaffeeautomat extends Automat{`
 	
`...`
 	
`}`
	Deklariert werden alle Eigenschaften und Verhaltensweisen die für das Objekt selbst besonders sind. Eine erbende Klasse kann die Verhaltensweisen und Eigenschaften der Subklasse nutzen.

Arbeitsauftrag:

1. Erzeugen Sie ein neues Projekt um den → Rechner umzusetzen.
2. Erzeugen Sie ein UML-Klassendiagramm. Finden Sie heraus welche Gemeinsamkeiten und Unterschiede unsere Rechner aufweisen. Berücksichtigen Sie dazu die Vorgaben und Hilfestellungen auf dem Informationsblatt.
3. Implementieren Sie anschließend die Klasse → Rechner und wenden Sie das Prinzip der Vererbung an.
4. Dokumentieren Sie welche einzelnen Schritte für die Umsetzung der View, des Controllers und des Models notwendig sind.

12 PHP und Datenbanken

12.1 Einführung in Sessions

12.1.1 Informationsblatt

Thema:	Sessions Autor: Christine Janischek Informationsblatt: Sessions

PHP Sessions

Mit PHP Sessions haben wir die Möglichkeit bestimmte Daten während einer Folge von Aufrufen auf einer Webseite festzuhalten und ggf. auf anderen Seiten weiterzuverarbeiten. Im Kontext von Webseiten die beabsichtigen Daten aus Relationalen Datenbanken zu nutzen (Daten einfügen, Daten auswählen, Daten verändern und Daten löschen), können wir auf den Einsatz von Sessions nicht verzichten.

Dem Besucher wird im Hintergrund eine Session-ID zugeordnet. Somit kann PHP den Besucher identifizieren. Diese Session ID wird entweder als Cookie, als verdeckte Formularfelder oder als Attributwert an die URL gehängt.

Zusatzinformationen können dazu in sogenannte Sessionvariable übernommen und ggf. weiterverwendet werden. Für die Handhabung von Sessions nutzen wir Heutzutage bestehende Bibliotheken sog. Frameworks.

→ zurück zum Arbeitsblatt

`<?php`
`  session_start();`
`?>`
	Mit **`session_start();`** sagen wir dem PHP Script, dass diese Seite mit Session arbeitet. Der Quellcode muss immer ganz oben stehen. Der Methodenaufruf erfüllt zum einen den Zweck eine Session (Sitzung) zu starten und zum Anderen kann eine bestehende Sitzung fortgeführt werden.

`<?php`
`  $_SESSION['name'] = "wert";`
`?>`
	Wir registrieren eine Session-Variable. Dazu weisen wir den Wert in einer Sessionvariable zu.

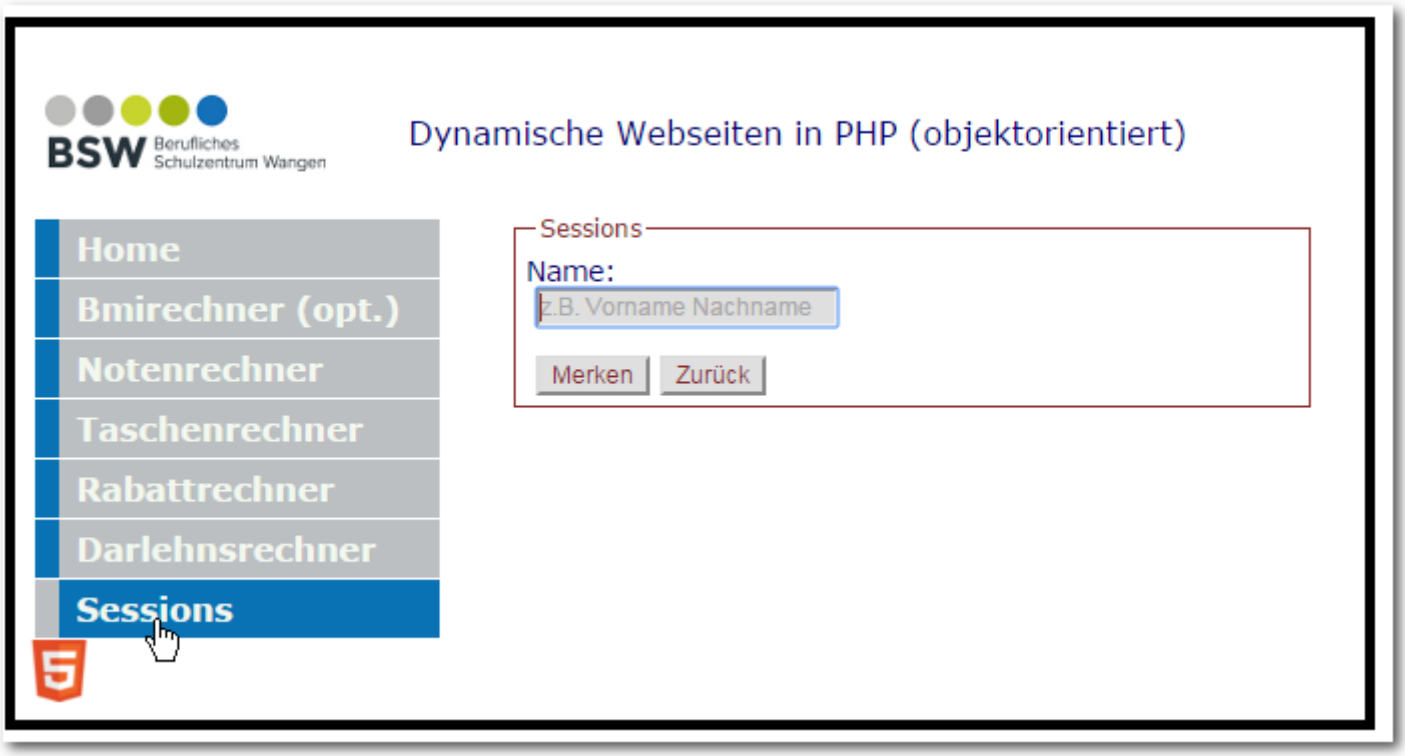
`<?php`
`  $pName = $_SESSION['name'];`
`?>`
	Für den Fall, dass wir nach einem Seitenwechsel die Session fortsetzen, wird der Wert aus der Sessionvariable erneut in eine lokale Variable übermittelt.

`<?php`
`  session_destroy();`
`?>`
	Löschen der Sitzung. Globale Variablen und das Session-Cookie werden nicht gelöscht.
	
	Alle Session-Variablen werden mit dem Methodenaufruf gelöscht:

	$_SESSION = array();

12.1.2 Arbeitsblatt

Thema: 	Sessions Autor: Christine Janischek Arbeitsblatt: Sessions



Arbeitsauftrag:

1. Wir folgen dem informatischen Prinzip der Wiederverwendbarkeit und kopieren dazu den Inhalt aus dem vorgegebenen Projektverzeichnis in neues Projektverzeichnis.
2. Folgen Sie die folgenden Arbeitsanweisungen und nutzen Sie ergänzend die Hinweise auf dem Informationsblatt.
3. Testen Sie das Projekt.
4. Helfen Sie Kollegen und dokumentieren Sie Ihre Vorgehensweise und Ergebnisse.

Arbeitsmaterialien und Übungsprojekte finden Sie im E-Learning:

→ E-Learning OOP

session.php

Ergänzen Sie den fehlenden Quellcode an entsprechender Stelle.

Verweisen Sie in der Form-Action auf die auszuführende Datei:

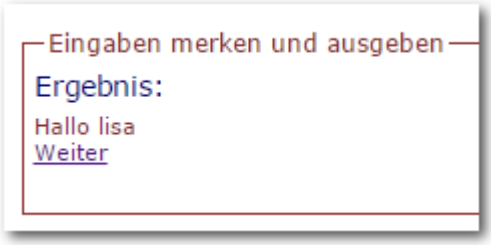
```
→ session1.php
```

Ergänzen Sie die Bezeichnung:

```
<label for="tfName">Name:</label> <br />
```

Ergänzen Sie das Texteingabefeld für den Namen:

```
<input type="text" name="tfName" id="tfName"
       placeholder="z.B. Vorname Nachname"
       required="required"
       autofocus="autofocus" /> <br /><br />
```



session1.php

Ergänzen Sie den fehlenden Quellcode an entsprechender Stelle.

Sie sollten die Sitzung starten:

```
→ session_start();
```

Eingabewert lesen:

```
$pName = $_POST ['tfName'];
```

Session registieren:

```
$_SESSION['username'] = $pName;
```

Ergebnis ausgeben:

```
echo "Ergebnis: <h5>Hallo "
             .$_SESSION['username']." <br />";
```

Verweis auf die nächste Seite:

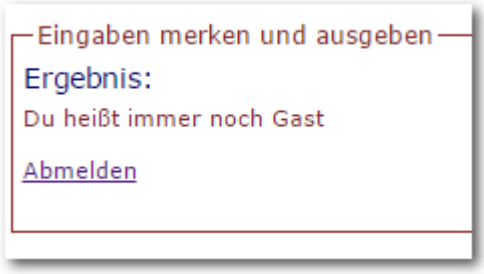
```
echo "<a href=session2.php>Weiter</a>";
```

Verweis zurück:

```
echo "<a href=session1.php
onClick='history.back()'>Zur&uuml;ck</a></h5>";
```

Ergänzen Sie den fehlenden Quellcode an entsprechender Stelle.

Sie sollen die Session verwenden (fortsetzen):



session2.php

```
session_start(); //Ganz wichtig
```

Übernehmen Sie den Wert aus der Session in ein lokale Variable:

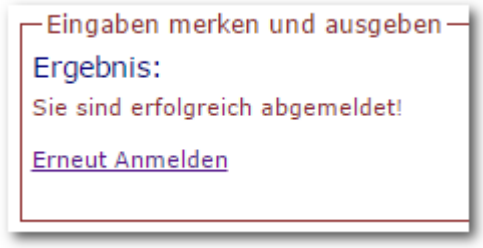
```
$pName = $_SESSION['username'];
```

Ergebnis ausgeben:

```
echo "Ergebnis: <h5>Du heißt immer noch ".
      $_SESSION['username']." <br /><br />";
```

Verweisen Sie auf eine andere Seite:

```
echo "<a href=session3.php>Abmelden</a></h5>";
```



session3.php

Ergänzen Sie den fehlenden Quellcode an entsprechender Stelle.

Sie sollen die Session verwenden (fortsetzen):

```
session_start(); //Ganz wichtig
```

Übernehmen Sie den Wert aus der Session in ein lokale Variable:

```
$pName = $_SESSION['username'];
```

Löschen aller Session-Variablen:

```
$_SESSION = array();
```

Löschen Sie die Daten der Sitzung:

```
session_destroy();
```

Text ausgeben:

```
echo "Ergebnis: <h5>Sie sind erfolgreich
      abgemeldet!<br /><br />";
```

Verweis auf die Startseite der Session-Lektion:

```
echo "<a href=sessionseite.php>
      Erneut Anmelden</a></h5>";
```

12.2 Einführung in die Verschlüsselung

12.2.1 Informationsblatt

Thema:	Verschlüsselung Autor: Christine Janischek Informationsblatt: Ver- und Entschlüsseln (z.B von Passwörtern)

PHP Hash Funktion
Wer Zugangsdaten oder besonders vertrauliche Daten (Bankdaten) speichern möchte (z.B. in einer Datenbank), sollte diese nach Möglichkeit verschlüsseln.

Zusätzliche Informationen zu:
bcrypt
password_hash()
password_verify()

→ zurück zum Arbeitsblatt

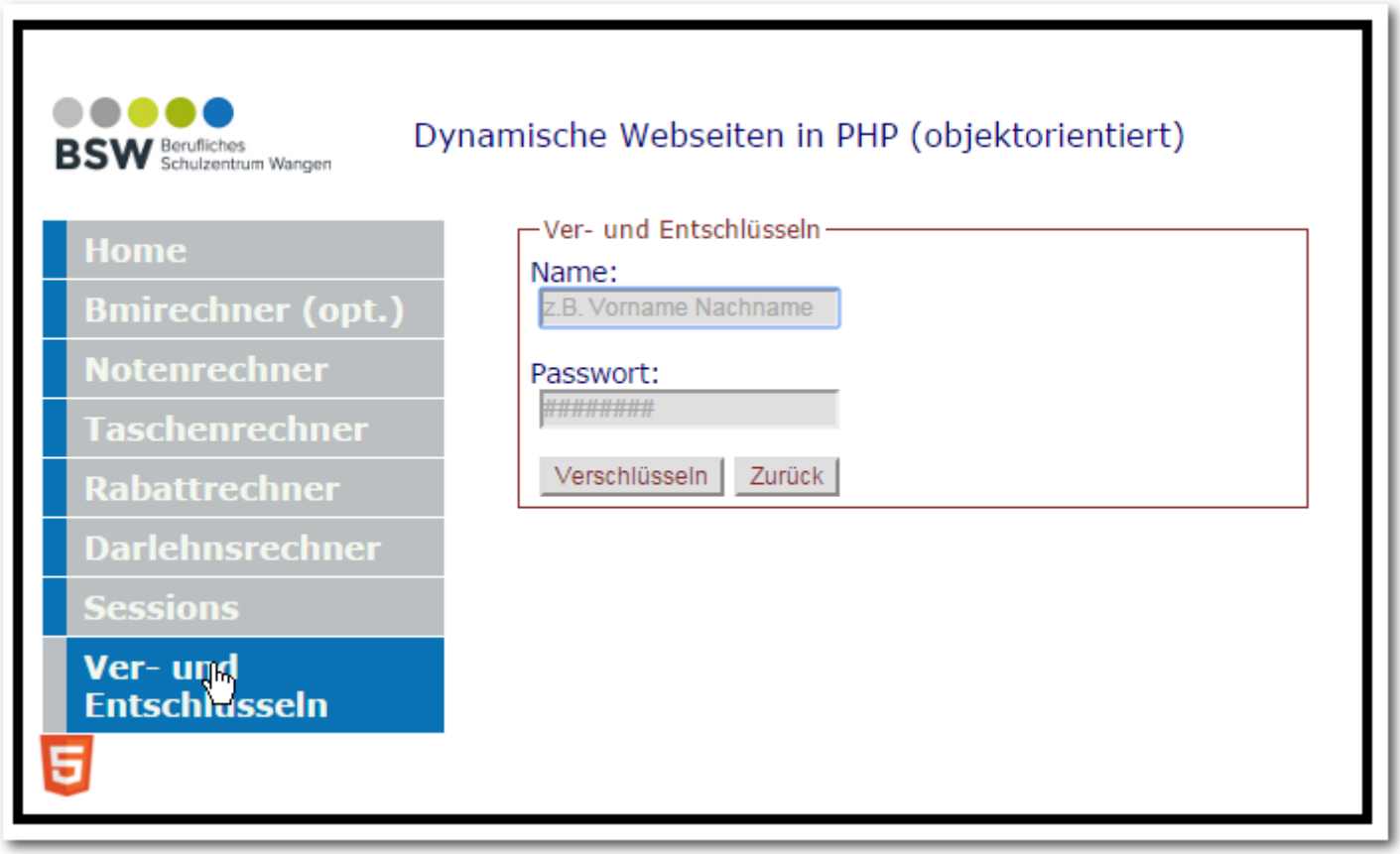
`password_hash(`
`  $pPasswort.$pTrick,`
`  PASSWORD_DEFAULT,`
`  array('cost'=>12))`
	Verschlüsseln:
	
	1.Parameter
	Ein String der das Passwort enthält. Hier wird, aus Gründen der Sicherheit, dem Passwort ($pPasswort) ein weiteres geheimes Wort ($pTrick) angehängt.
	
	2.Parameter
	Angabe des verwendeten Verschlüsselungsalgorithmus.
	
	3.Parameter
	Die nicht zwingende Angabe (optional) für den Aufwand (Zeit, Variablen) des Algorithmus. Wird ggf durch eine Ganze Zahl festgelegt. Falls keine Angabe erfolgt ist der Standardwert 10. Die Funktion liefert als Rückgabe den Hashwert.
	
	→ Dieser Wert wird in der DB gespeichert.

`password_verify('passwort',$hash)`	Passwort prüfen: 1.Parameter Enthält das eingegebene Passwort mit dem angehängten geheimen Wort. 2.Parameter Enthält den in der Datenbank gespeicherten ggf. zugehörigen Hashwert. Die Funktion liefert als Rückgabewert 0 (falsch → Passwort konnte nicht verifiziert werden!) oder 1 ,

	also wahr (→ Passwort verifiziert), falls der Hashwert zum Eingabewert & Geheimwert passt.

12.2.2 Arbeitsblatt

Thema: 	Verschlüsselung Autor: Christine Janischek Arbeitsblatt: Ver- und Entschlüsseln (z.B von Passwörtern)

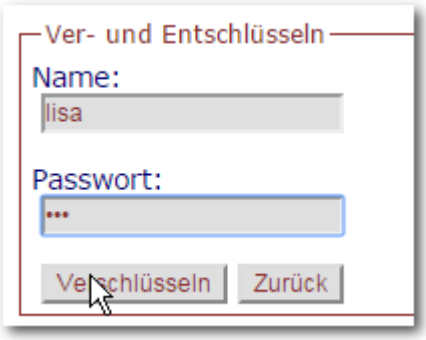


Arbeitsauftrag:

1. Wir folgen dem informatischen Prinzip der Wiederverwendbarkeit und kopieren dazu den Inhalt aus dem vorgegebenen Projektverzeichnis in neues Projektverzeichnis.
2. Folgen Sie die folgenden Arbeitsanweisungen und nutzen Sie das Informationsblatt.
3. Testen Sie das Projekt.
4. Helfen Sie Kollegen und dokumentieren Sie Ihre Vorgehensweise und Ergebnisse.

Arbeitsmaterialien und Übungsprojekte finden Sie im E-Learning:

→ E-Learning OOP



hash.php

Ergänzen Sie den fehlenden Quellcode an entsprechender Stelle.

Verweisen Sie in der Form-Action auf die auszuführende Datei:

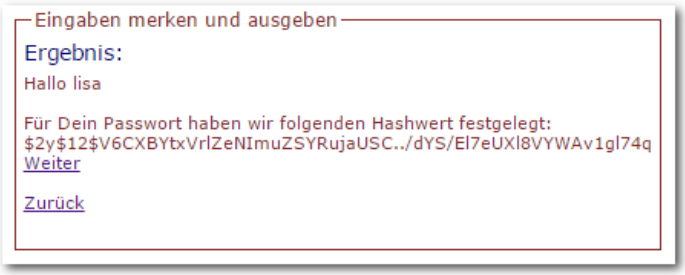
```
→ hash1.php
```

Ergänzen Sie die fehlende Bezeichnung:

```
<label for="tfPasswort">Passwort:</label><br />
```

Ergänzen Sie das fehlende Texteingabefeld für das Passwort:

```
<input    type="password"    name="tfPasswort"
       id="tfPasswort"   placeholder="########"
       required="required"
       autofocus="autofocus" /> <br /><br />
```



hash1.php

Ergänzen Sie den fehlenden Quellcode an entsprechender Stelle.

Sie sollten die Sitzung starten:

```
→ session_start();
```

Eingabewert lesen:
(simuliert) Formulardaten bei der Registrierung

```
$pName = $_POST ['tfName'];
$pPasswort =  $_POST ['tfPasswort'];
```

Geheimer Wert ("pTrick") festlegen:

```
$pTrick = 'Angsthase!';
```

Erzeugen Sie mit Hilfe der PHP-Hash-Funktion einen Hashwert: Funktioniert nur ab PHP Version 5.5:

```
$pHash = password_hash($pPasswort.$pTrick,
       PASSWORD_DEFAULT,array('cost'=>12));
```

Registrieren Sie die Session:

```
$_SESSION['username'] = $pName;
$_SESSION['password'] = $pPasswort;
$_SESSION['geheim'] = $pTrick;
```

Ergebnis ausgeben:

```
echo "Ergebnis: <h5>Hallo "
              .$_SESSION['username']." <br />";
```

Veranlassen Sie die Ausgabe des erzeugten Hash-Wertes für das Passwort:

```
echo  "Für  Dein  Passwort  haben  wir  folgenden
      Hashwert festgelegt:<br />"
```

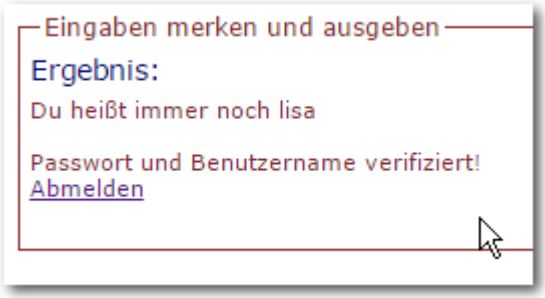
```
        .$pHash."<br />";
```

Verweis auf die nächste Seite:

```
echo "<a href=hash2.php>Weiter</a>
      <br /><br />";
```

Verweis zurück:

```
echo "<a href=hash1.php
onClick='history.back()'>Zur&uuml;ck</a></h5>";
```



hash2.php

Ergänzen Sie den fehlenden Quellcode an entsprechender Stelle.

Sie sollen die Session verwenden (fortsetzen):

```
session_start(); //Ganz wichtig
```

Übernehmen Sie die Werte aus der Session in lokale Variable:

```
$pName = $_SESSION['username'];
$pPasswort = $_SESSION['password'];
$pTrick = $_SESSION['geheim'];
```

Erzeugen Sie einen Datencontainer der die Selektion der Daten (Benutzername, Passwort) aus der Datenbank vorübergehend simulieren soll:

```
$db_daten = array(
      'uname' => 'lisa',
      'pw' => '$2y$12$FWf2TMp2kunBNVUTSB
      D6ey34Al36FrSR2Yyd6xDCWwV/MKJi6Cdu',
                                    );
```

Erzeugen Sie den Quellcode für die Verifizierung und übernehmen Sie das Ergebnis in eine lokale Variable:

```
$valid = password_verify($pPasswort.$pTrick,
      $db_daten['pw']);
```

Prüfen Sie ob die Verifizierung erfolgreich war und der richtige Benutzername eingegeben wurde:

```
if($valid == 1 && $pName == $db_daten['uname'])
{
      pMeldung = "Passwort und Benutzername
            verifiziert!";

      //Verweis auf Abmeldeseite
      $pMeldung = $pMeldung ."<br /><a
            href=hash3.php>Abmelden</a></h5>";

}else{

      $pMeldung = "Passwort oder Benutzername
```

```
                konnte nicht verifiziert werden!";

        //Verweis auf Anmeldeseite
        $pMeldung = $pMeldung ."<br />
        <a href=hashseite.php>Erneut
        Anmelden</a></h5>";
        session_destroy();
}
```

Ergebnis ausgeben:

```
echo  "Ergebnis:  <h5>Du  heißt  immer  noch  ".
      $_SESSION['username']." <br /><br />";
```

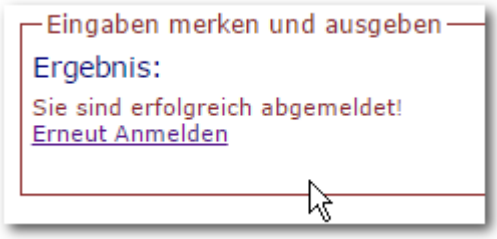
Meldung ausgeben:

```
echo $pMeldung ."<br />";
```

Verweisen Sie auf eine andere Seite:

```
echo "<a href=hash3.php>Abmelden</a></h5>";
```

→ zum Informationsblatt



hash3.php

Ergänzen Sie den fehlenden Quellcode an entsprechender Stelle.

Sie sollen die Session verwenden (fortsetzen):

```
session_start(); //Ganz wichtig
```

Übernehmen Sie den Wert aus der Session in ein lokale Variable:

```
$pName = $_SESSION['username'];
```

Löschen aller Session-Variablen:

```
$_SESSION = array();
```

Löschen der Sitzung:

```
session_destroy();
```

Text ausgeben:

```
echo "Ergebnis: <h5>Sie sind erfolgreich
      abgemeldet!<br /><br />";
```

Verweis auf die Startseite der Session-Lektion:

```
echo "<a href=hashseite.php>
      Erneut Anmelden</a></h5>";
```

12.3 Einführung Datenbankzugriff

12.3.1 Informationsblatt

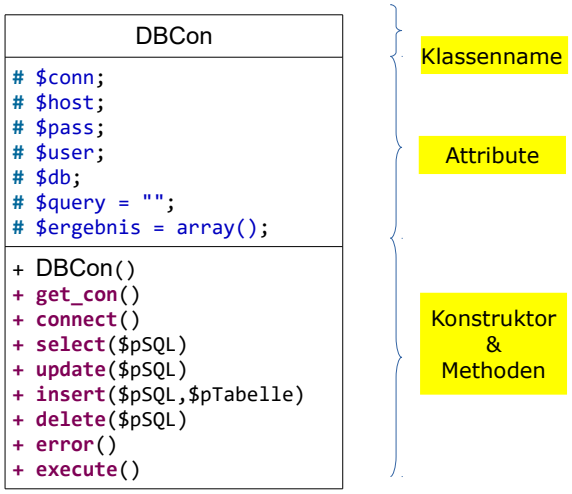
Thema:	Datenbankzugriff Autor: Christine Janischek Informationsblatt: Datenbankzugriff

PHP Datenbankanbindung für das Gäste
Ausser der Datenbank benötigen wir eine PHP-Klasse (DBCon). Diese Klasse legen wir in unserer PHP-Bibliothek (lib.php) ab. Sie stellt die benötigten Eigenschaften und Verhaltensweisen für unsere Datenverbindungsobjekte zur Verfügung.

Voraussetzung für die Implementierung des Gästebuchs ist der Import der dazugehörigen Datenbank:

→ freundedb_Struktur.sql
→ freundedb_Daten.sql

Außerdem müssen der Webserver (Apache) und das Datenbanksystem MySQL über das Xampp-Software-Paket oder ggf. über das Startmenü der Digitalen Tasche (Informatikstick) gestartet werden.



Algorithmik, Code Reading

Was passiert im Quellcode. Wir analysieren den Quellcode zeilenweise...

`dblogin1.php:`

Der Benutzer gibt den Benutzername und sein Passwort ein. Wenn der Benutzer auf die Schaltfläche → Anmelden klickt werden die folgenden Schritte ausgeführt:

1. Die Sitzung wird gestartet

2. Aus dem eingegeben Passwort (pPasswort) und dem geheimen Wort (pTrick) wird, wie zuvor auch, ein Hashwert erzeugt.
3. Alle Werte (pName, pPasswort und pTrick) werden an entsprechende Sessionvariable übergeben.
4. Zum testen lassen wir uns den erzeugten Hashwert auf der Benutzeroberfläche ausgeben.

dblogin2.php:

Mit einem Klick auf die Schaltfläche → Weiter werden dann die folgenden Schritte ausgeführt.

1. Die Sitzung wird fortgeführt
2. Die in den Sitzungsvariablen gespeicherten Werte werden an lokale Variablen übergeben.
3. Die Datebankverbindung wir hergestellt. Dazu wird ein Datenverbindungsobjekt erzeugt und die Verbindung aufgebaut.
4. Das Datenbankverbindungsobjekt wird einer Sessionvariablen übergeben.
5. Die SQL-Auswahl-Abfrage wird erzeugt und an eine lokale Variable übergeben.
6. Die Abfrage wird mittels des Datenbankverbindungsobjektes auf der Datenbank ausgeführt, der Rückgabewert wird in die lokale Variable $ergebnis übernommen. Das Ergebnis beinhaltet nun die Zugangsdaten aller Benutzer (ist eine Liste).
7. Schleife/Wiederholung: Schrittweise (zeilenweise) werden die Zugangsdaten dem Ergebnis entnommen
   - Werte der Zeile (Datensatz) werden in lokale Variablen/Attribute übergeben
   - Das Passwort wird verifiziert, der Rückgabewert der Verifizierung wird in eine lokale Variable übernommen (0 oder 1)
   - Fallunterscheidung:

     1.Fall:
     Wenn die Verifizierung erfolgreich und der Benutzername stimmig sind (Wert = 1)
     - Wird die Meldung "Passwort verifiziert!" erzeugt
     - Die Werte an die Sessionvariablen übergeben
     - Eine positive Ausgabe erzeugt und der Wiederholvorgang wird abgebrochen
   - Fallunterscheidung:

     2.Fall:
     Ansonsten, also wenn keine Übereinstimmung zustande kommt (Wert = 0)

- Wird die Meldung "Passwort konnte nicht verifiziert werden!" erzeugt.

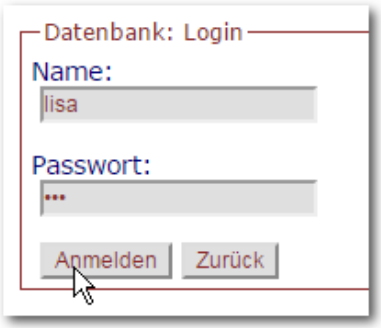
8. Ein Text mit der erzeugten Meldung wird auf der Benutzeroberfläche ausgeben.

dblogin3.php:

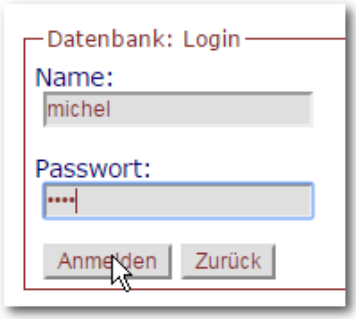
Mit einem Klick auf die Schaltfläche → Abmelden werden dann die folgenden Schritte ausgeführt.

1. Die Sitzung wird gestartet bzw. fortgeführt.
2. Die in den Sitzungsvariablen gespeicherten Werte werden an lokale Variablen übergeben.
3. Daten der Sitzung werden gelöscht.
4. Die Sitzung wird gelöscht.
5. Ausgabe wird erzeugt.

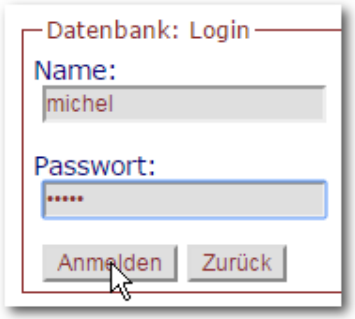
Testfälle		
		
Testfall 1: Eingabe Korrekt!	Testfall 2: Eingabe Korrekt!	Testfall 3: Eingabe Falsch!
Benutzername: lisa Passwort: 123	Benutzername: michel Passwort: 1234	Benutzername: michel Passwort: 12345



Lisa loggt sich richtig ein



Michel loggt sich richtig ein



Michel loggt sich falsch ein

Fragen zum Verständnis des Quellcodes:

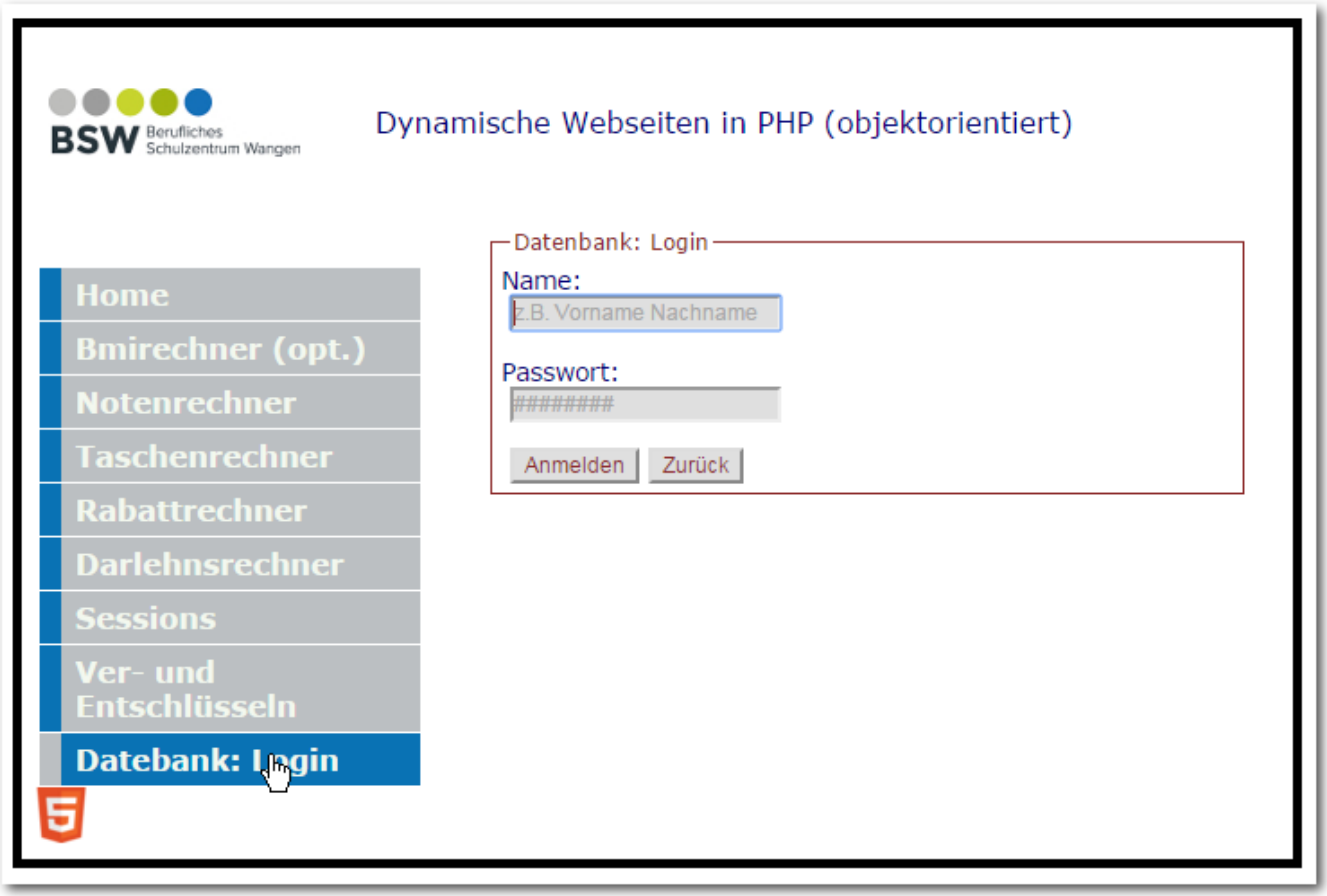
1. Wie lauten die Quellcode-Anweisungen in PHP Anweisung, um eine Gästebuch-Sitzung (Session) zu starten bzw. fortzuführen?
2. Wie lautet die Quellcode-Anweisung in PHP Anweisung, um Werte an entsprechende Sitzungsvariablen, zu übergeben?
3. Wie lautet die Gegenoperation im Quellcode in PHP, um einen in einer Sitzungsvariablen gespeicherten Wert an eine lokale Variablen, zu übergeben?
4. Wie lauten die Quellcode-Anweisungen in PHP, um ein neues Datenbankverbindungsobjekt und den Aufbau der Datenbankverbindung zu erzeugen?
5. Wie lautet die Quellcode-Anweisungen in PHP, um das eingegebene Passwort, zu verifizieren?
6. Formulieren Sie in SQL die Auswahlabfrage zur Ermittlung der Zugangsdaten aller Benutzer des Gästebuchs!
7. Wie lautet die Meldung für den Fall, dass bei der Ermittlung der Zugangsdaten aller Benutzer des Gästebuchs keine Daten ermittelt werden konnten?
8. Wie lauten die Quellcode-Anweisungen in PHP, um die Daten der Sitzung abschließend zu löschen?

Zusatzaufgabe:
Studieren Sie die Datenbankarchitektur (freundedb). Wie müssen die SQL-Abfragen lauten, um:

- Posts (Beiträge) anzuzeigen?
- Post (Beitrag) löschen?
- Post (Beitrag) ändern?
- Post (Beitrag) einfügen?

12.3.2 Arbeitsblatt

Thema:	Datenbankzugriff Autor: Christine Janischek Arbeitsblatt: Datenbankzugriff



Arbeitsauftrag:

1. Wir folgen dem informatischen Prinzip der Wiederverwendbarkeit und kopieren dazu den Inhalt aus dem vorgegebenen Projektverzeichnis in neues Projektverzeichnis.
2. Importieren Sie über Ihr Datenbank-Management-System (PHPmyadmin, MySQLWorkbench) die Struktur und Daten der Datenbank (freundedb).
3. Lesen Sie den Anwendungsfall aufmerksam durch.
4. Testen Sie das Projekt. (Testfälle)
5. Beantworten Sie im Anschluss die Fragen.

Arbeitsmaterialien und Übungsprojekte finden Sie im E-Learning:

→ E-Learning OOP

12.4 Einführung Datenbankoperationen

12.4.1 Informationsblatt

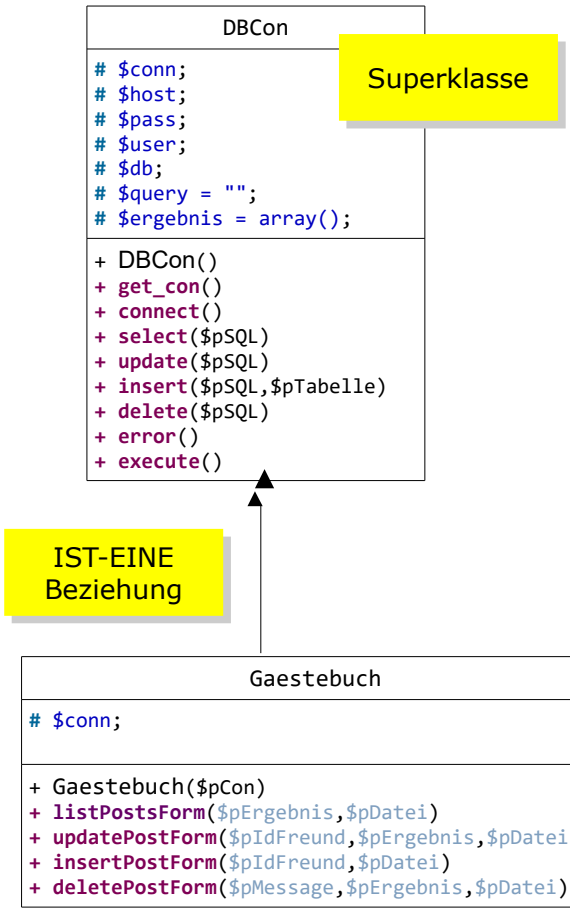
Thema:	Datenbankoperationen Autor: Christine Janischek Informationsblatt: Datenbankoperationen für das Gästebuch

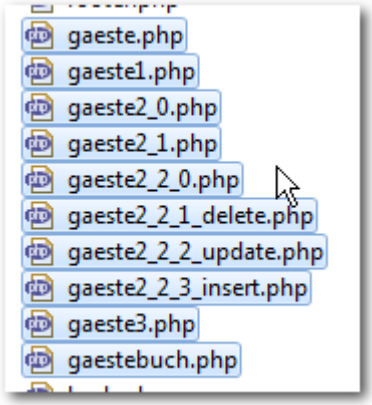
PHP Gästebuch
Voraussetzung für die Implementierung des Gästebuchs ist der Import der dazugehörigen Datenbank:

→ freundedb_Struktur.sql
→ freundedb_Daten.sql

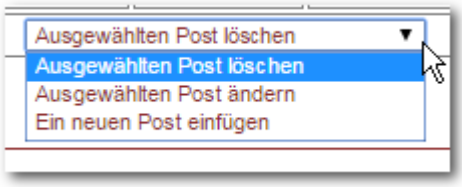
Außerdem müssen der Webserver (Apache) und das Datenbanksystem MySQL über Xampp-Software-Paket oder alternativ über das Startmenü der Digitalen Tasche (Informatikstick) gestartet werden.

→ zurück zum Arbeitsblatt





Dateien



Datenbankoperationen

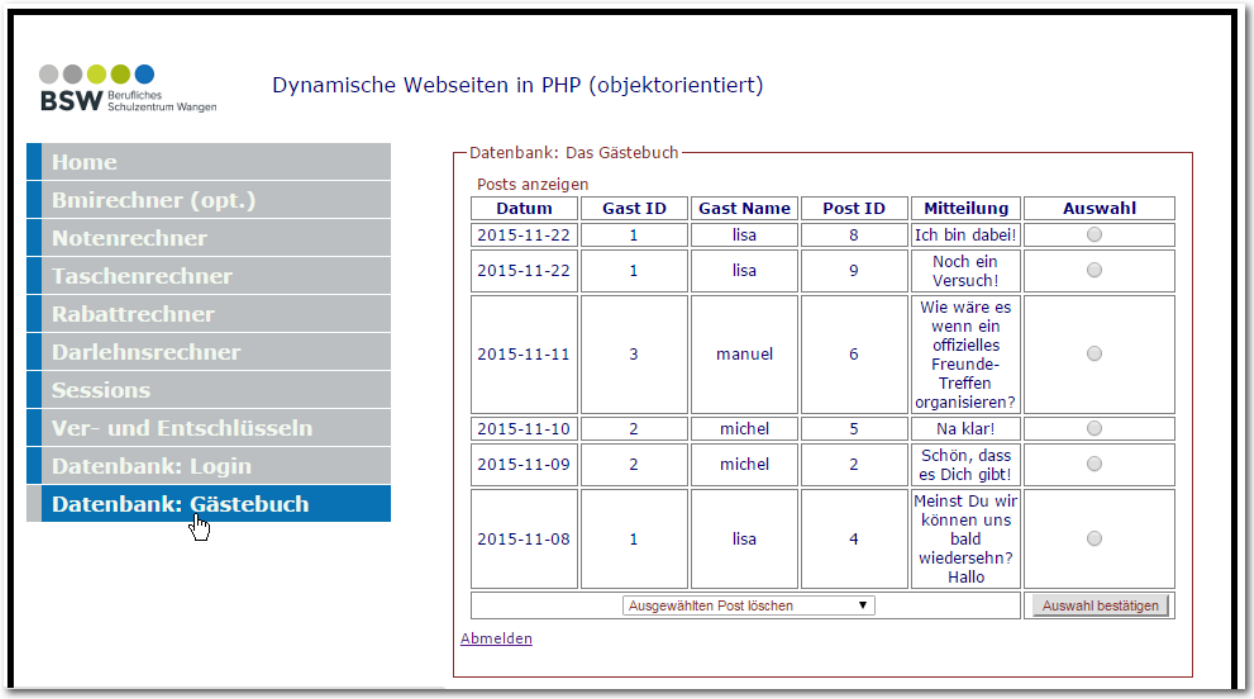
Vertiefung der wichtigsten Prinzipien objektorientierter Programmierung

	→ Abstraktion Verwendung von Klassenkonzepten zur Sicherstellung der Wiederverwendbarkeit und Erweiterbarkeit → Kapselung Verwendung von Zugriffsmodifikatoren (z.B. private, public, protected,...) um den direkten Zugriff auf Werte regulieren zu können und Sicherheit zu gewährleisten. → Vererbung Zur Sicherstellung der Wiederverwendbarkeit und Erweiterbarkeit von Eigenschaften und Verhaltensweisen von Objekten. → Polymorphie Vielgestaltigkeit von Methoden (deren Implementierung). Um Flexibilität und Variabilität bezüglich der Implementierung von Verhaltensweisen zu gewährleisten

Anwendungsfälle Gästebuch		
		
Testfall 1: Eintrag löschen!	Testfall 2: Eintrag ändern!	Testfall 3: Eintrag einfügen!
Benutzername: lisa Passwort: 123 Datenbankoperation: Ausgewählter Post löschen ausloggen!	Benutzername: michel Passwort: 1234 Datenbankoperation: Ausgewählter Post ändern ausloggen!	Benutzername: lisa Passwort: 123 Datenbankoperation: Einen neuen Post einfügen ausloggen!

12.4.2 Arbeitsblatt

Thema:	Datenbankoperationen Autor: Christine Janischek Arbeitsblatt: Gästebuch



Anzeige aller Posts

Arbeitsauftrag:
Sie sollen die bestehende Anwendung optimieren!

1. Wir folgen dem informatischen Prinzip der Wiederverwendbarkeit und kopieren dazu den Inhalt aus dem vorgegebenen Projektverzeichnis in neues Projektverzeichnis.
2. Studieren Sie die Architektur und Implementierung der Anwendung aufmerksam. Testen Sie die Anwendung! Beachten und Nutzen Sie die Hinweise im Informationsblatt.
3. Lokalisieren Sie die SQL-Abfragen und Wiederholstrukturen im Quellcode.
4. Versuchen Sie durch die Anwendung der informatischen Prinzipien „Kapselung" und „Wiederverwendung" den Quellcode zu optimieren.

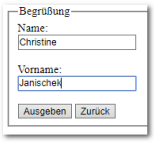
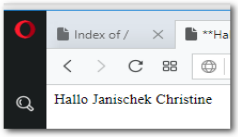
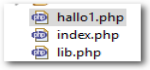
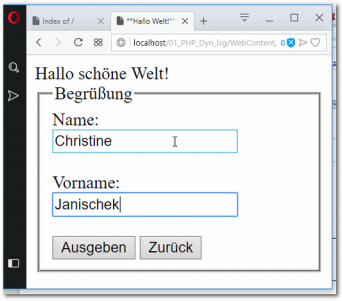
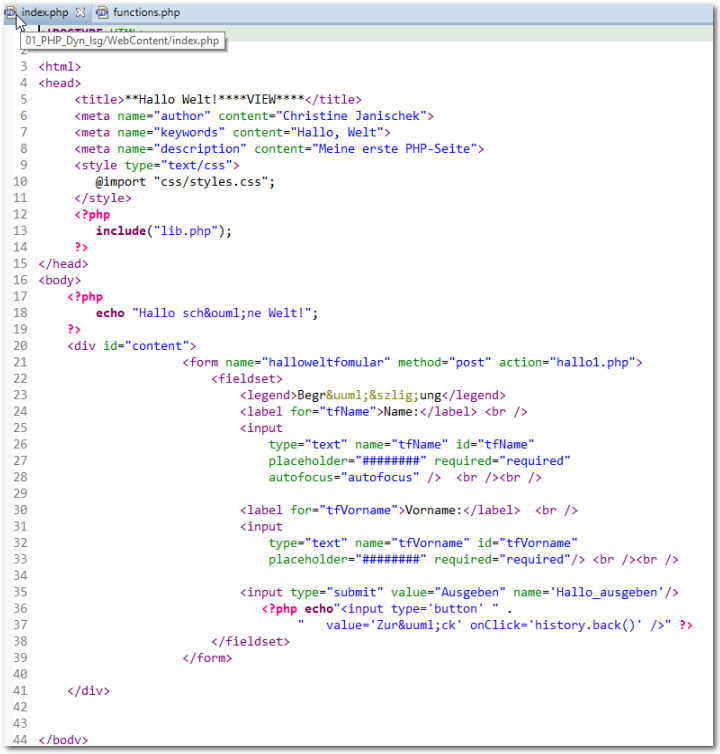
Arbeitsmaterialien und Übungsprojekte finden Sie im E-Learning:

→ E-Learning OOP

13 Sonstiges Arbeitsmaterial

13.1.1 Arbeitsblatt: Hallo Welt **** Eine dynamisch-objektorientierte Lösung

Thema: 	Hallo Welt Autor: Christine Janischek Seite 1: Eine dynamisch-objektorientierte Lösung ****VIEW****

EINGABE:	AUSGABE:
	
	Projektdateien
 Formular: index.php	Quellcode (Source): 

Thema:	Hallo Welt Autor: Christine Janischek Seite 2: Eine dynamische Lösung ****CONTROLLER****

hallo1.php | functions.php

01_PHP_Dyn_lsg/WebContent/hallo1.php

```
1  <!DOCTYPE HTML>
2
3  <html>
4  <head>
5       <title>**Hallo Welt!****CONTROLLER****</title>
6       <meta name="author" content="Christine Janischek">
7       <meta name="keywords" content="Hallo, Welt">
8       <meta name="description" content="Meine erste PHP-Seite">
9       <style type="text/css">
10         @import "css/styles.css";
11      </style>
12      <?php
13         include("lib.php");
14      ?>
15 </head>
16 <body>
17          <?php
18          //####(E) für EINGABE#####
19
20          //Eingaben übernehmen
21          $pName = $_POST['tfName'];
22          $pVorname = $_POST['tfVorname'];
23
24          //####(V) für VERARBEITUNG#####
25          $diePerson = new Person();
26
27          $diePerson -> leseEingaben($pName,$pVorname);
28
29          //####(A) für AUSGABE######
30          $diePerson -> sagHallo();
31          ?>
32
33 </body>
34 </html>
35
```

hallo1.php

Thema:	Hallo Welt Autor: Christine Janischek Seite 2: Eine dynamische Lösung ****MODELL****

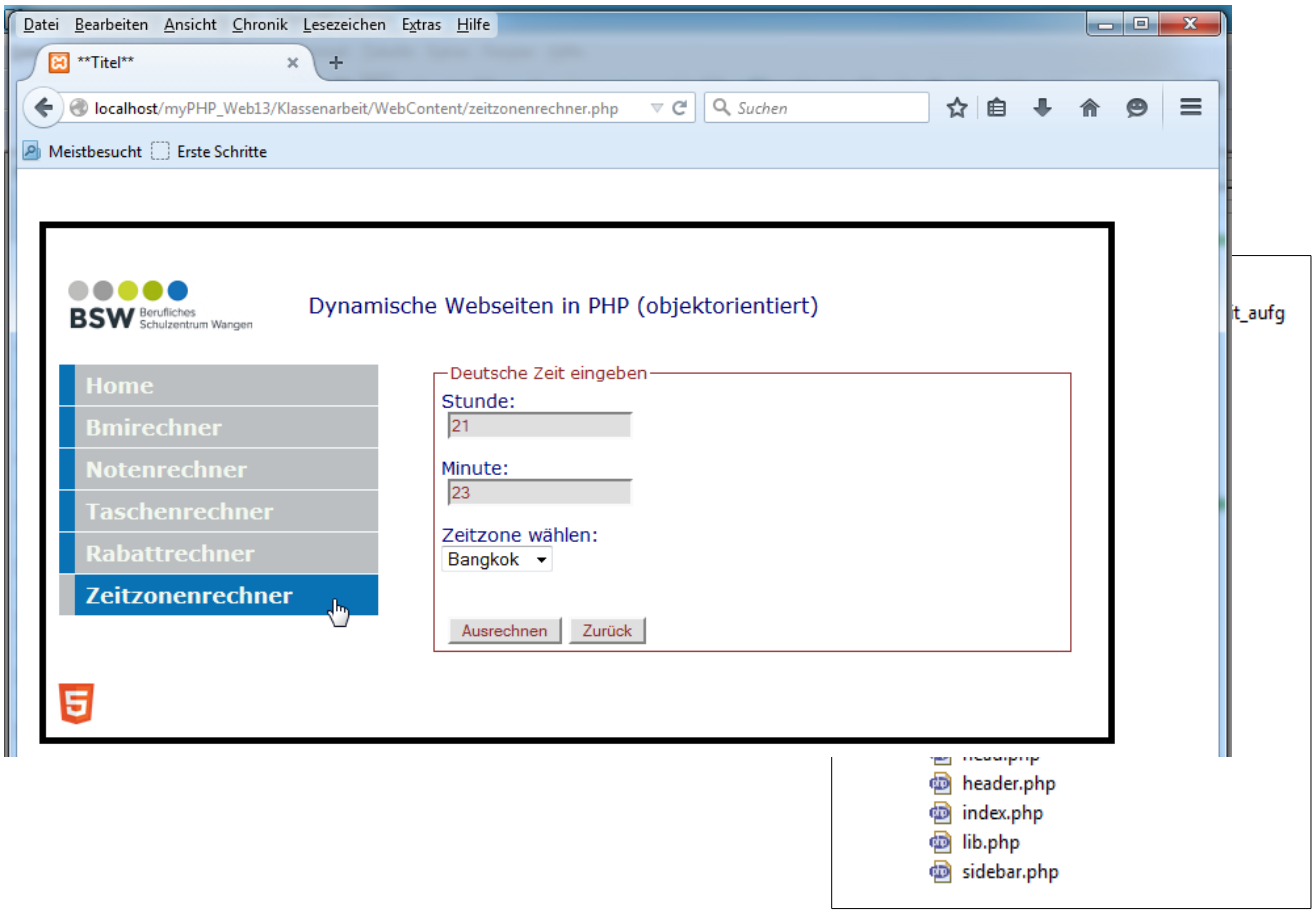
lib.php | functions.php

```php
<?php
//****MODEL****
class Person{
    //Eigenschaften: Attribute
    private $name;
    private $vorname;

    //Konstruktor: Ohne Parameter, Leer, zum Erzeugen von Objekten dieser Klasse
    public function __construct(){}

    //Getter: Get-Methoden ermitteln den Wert einer Eigenschaft eines Objektes der Klasse
    public function get_name() {
        return $this->name;
    }

    public function get_vorname() {
        return $this->vorname;
    }

    //Setter: Get-Methoden übermitteln den Eigenschaftswert an ein Objektes der Klasse
    public function set_name($pName) {
        $this->name = $pName;
    }

    public function set_vorname($pVorname) {
        $this->vorname = $pVorname;
    }

    public function leseEingaben($pName, $pVorname) {
        $this-> set_name($pName);
        $this-> set_vorname($pVorname);
    }

    public function sagHallo(){
        echo "Hallo ".$this -> get_vorname()." ".$this -> get_name();
    }


}
?>
```

lib.php

13.1.2 Arbeitsblatt: Fallunterscheidungen am Beispiel des Zeitzonenrechners

Thema:	Der Zeitzonenrechner Autor: Christine Janischek Kontrollstrukturen: Fallunterscheidungen



Arbeitsauftrag:

1. Erzeugen Sie ein Projektverzeichnis → Zeitzonenrechner
2. Erweitern Sie die Lib um die benötigte (Fach-)klasse.
3. Implementieren Sie den Quellcode für die Methode *ermittle_faktor()* und *berechne()*.

Zusatzinformation:

Zone	**Faktor** **(Zeitverschiebung)**
New York	-6
London	-1
Tokio	+5
Sydney	+10

Für	**ergebnis**
AmVortag: (stunde + faktor) >= 24	stunde + faktor – 24
AmFolgetag: (stunde + faktor) < 0	stunde + faktor + 24
Am gleichen Tag: (stunde + faktor) < 24	stunde + faktor

14 Ich-Kann-Listen

14.1 Grundgerüst einer Klasse, Begriffe, Ereignissteuerung

Informatik: Grundgerüst einer Klasse, Begriffe, Ereignissteuerung

Ich kann…

Für Schüler

… zum Thema „Grundgerüst einer Klasse, Begriffe, Ereignissteuerung“	stimmt völlig	stimmt eher	stimmt wenig	stimmt gar nicht	Wiederholungs- und Übungsmöglichkeiten
Inhalt					
… das Client-Server-Prinzip an einem konkreten Beispiel erläutern.					
… mindestens drei Vorteile der Objektorientierten Programmierung nennen und an konkreten Beispielen erläutern.					
… ein einfaches dynamisches Layout erzeugen.					
… eine Klasse in einem UML-Klassendiagramm darstellen und kann die Bedeutung der Notationselemente erklären.					
… den Quellcode (die Anweisung) für die Deklaration einer Klasse implementieren.					
… den Quellcode (die Anweisungen) um Variablen/Attribute zu deklarieren, implementieren.					
… den Quellcode (die Anweisungen) um den Standardkonstruktor einer Klasse zu deklarieren.					
… den Quellcode (die Anweisungen) für den Getter (Get-Methode) eines Attributs/Variablen implementieren.					

… den Quellcode (die Anweisungen) für den Setter (Set-Methode) eines Attributs/Variablen implementieren.					
… den Quellcode (die Anweisungen) für Methoden die einfache Berechnungen enthalten implementieren.					
… den Quellcode (die Anweisungen) für die Erzeugung eines Objektes einer Klasse implementieren.					
… den Quellcode (die Anweisungen) um Werte an ein bestehendes Objekt zu übermitteln, implementieren.					
… den Quellcode (die Anweisungen) um Werte eines bestehenden Objektes zu ermitteln, implementieren.					
… den Quellcode (die Anweisungen) für den Methodenaufruf eines bestehenden Objektes implementieren.					
… den Quellcode (die Anweisungen) für die Erzeugung einer Ausgabe auf der Benutzeroberfläche implementieren.					
… das EVA-Prinzip anhand einer Controller-Datei erklären.					
… die Architektur einer Anwendung (MVC-Konzept) an einem konkreten Beispiel erläutern.					
… auf die Einbettung meiner Bibliothek (→ lib.php) in einer Anwendung implementieren.					
… Testfälle für eine bestehende Fachklasse formulieren.					

14.2 Methoden, Algorithmen, Kontrollstrukturen

Informatik: Methoden, Algorithmen, Kontrollstrukturen

Ich kann…

Für Schüler

… zum Thema „Inhalt von Methoden“	stimmt völlig	stimmt eher	stimmt wenig	stimmt gar nicht	Wiederholungs- und Übungsmöglichkeiten
Inhalt					
… den Quellcode (die Anweisung) für die Deklaration einer Methode mit Parameter und ohne Rückgabewert vornehmen.					
… den Quellcode (die Anweisung) für die Deklaration einer Methode ohne Parameter aber mit Rückgabewert vornehmen.					
… den Quellcode (die Anweisung) für die Deklaration einer Methode ohne Parameter aber mit Rückgabewert vornehmen.					
… den Quellcode (die Anweisungen) für Methoden anhand der Vorgaben und unter Verwendung geeigneter Kontrollstrukturen implementieren.					
… die Besonderheiten der Fallunterscheidung SWITCH CASE nennen und bei der Entwicklung berücksichtigen.					
… die Grundprinzipien der objektorientierten Programmierung nennen und anhand ausgewählter Beispiele erläutern.					
… den Quellcode (die Anweisungen) für Fallunterscheidungen die mehrere Bedingungen benötigen (Intervallabfragen) logisch verknüp-					

fen und implementieren.					
… die logischen Vergleichsoperatoren in PHP nennen und deren Unterschiede erläutern.					
… Wiederholvorgänge mittels einer Schleifenstruktur abhandeln und implementieren.					

14.3 Vererbung

Informatik: Vererbung

Ich kann…

Für Schüler

… zum Thema „Vererbung“	stimmt völlig	stimmt eher	stimmt wenig	stimmt gar nicht	Wiederholungs- und Übungsmöglichkeiten
Inhalt					
… die Gemeinsamkeiten und Unterschiede bestehender Anwendungen erkennen.					
… anhand eines Anwendungsfalles Vererbungsstrukturen erkennen und modellieren.					
… die Eigenschaften und Verhaltensweisen für Super- und Subklassen in einem UML-Klassendiagramm definieren.					
… den Quellcode für eine erbende Klasse implementieren					
… erklären wozu eine abstrakte Klasse sinnvoll sein kann.					
… eine abstrakte Klasse in UML modellieren.					
… Quellcode für eine abstrakte Klasse deklarieren.					
… vererbte Verhaltensweisen in der Implementierung der Anwendung nutzen.					
… Optimierungsmöglichkeiten an bestehenden Anwendungen erkennen und erläutern.					

14.4 PHP und Relationale Datenbanken

Informatik: PHP und Relationale Datenbanken

Ich kann…

Für Schüler

… zum Thema „PHP und Relationale Datenbanken“	stimmt völlig	stimmt eher	stimmt wenig	stimmt gar nicht	Wiederholungs- und Übungsmöglichkeiten
Inhalt					
… den Quellcode (die Anweisung) zum starten bzw. fortführen einer Session implementieren.					
… den Quellcode (die Anweisungen) um Werte an entsprechende Sessionvariable zu übergeben, implementieren.					
… den Quellcode (die Anweisungen) um einen in einer Sitzungsvariablen gespeicherten Wert an eine lokale Variablen zu übergeben, implementieren.					
… den Quellcode (die Anweisungen) zum Erzeugen eines neuen Datenbankverbindungsobjektes und den Aufbau der Datenbankverbindung, implementieren.					
… den Quellcode (die Anweisungen) um das eingegebene Passwort, zu verifizieren, implementieren.					
… die SQL (die Anweisungen) Auswahlabfrage zur Ermittlung der Zugangsdaten aller Benutzer, formulieren.					
… den Quellcode (die Anweisungen) um die Daten der Sitzung abschließend (zu zerstören, implementieren.					

14.5 PHP und Datenbankoperationen

Informatik: PHP und Datenbankoperationen

Ich kann…

Für Schüler

… zum Thema „PHP und Datenbankoperation“	stimmt völlig	stimmt eher	stimmt wenig	stimmt gar nicht	Wiederholungs- und Übungsmöglichkeiten
Inhalt					
… typische Datenbankoperationen einer datenbankgestützten Anwendung nennen.					
… die SQL-Befehle für diese typischen Datenbankoperationen nennen und an Beispielen erläutern.					
… unter welchen Gesichtspunkten der Einsatz einer Datenbank notwendig ist.					
… den Quellcode einer Datenbankgestützten Anwendung verstehen.					
… Wiederholungen im Quellcode identifizieren.					
… Lösungsansätze für eine Optimierung erarbeiten und erläutern.					
… kleine Optimierungen im Quellcode selbstständig implementieren.					