Objektorientierte Programmierung: Klassen, Objekte und Methoden

Die Klasse: Grundlage der objektorientierten Programmierung

- Klasse ist die Vorlage für ein Objekt
- Wird definiert mit Schlüsselwort `class`
- Die einzelnen Memeber sind Variablen, deshalb $
- Zugriffsrecht mit `public` und `private` definieren
  - Mit `public`: möglich von außerhalb der Klasse auf die Werte zuzugreifen und zu verändern
  - Mit `private`: nur möglich, aus der gleichen Klasse auf Variable zuzugreifen
- Möglich, Variable bereits in der Klasse zu initialisieren
  - Dh entsprechenden Standardwert zuweisen
  - Später möglich zu ändern

```php
<?php
class Car {
    private $seat;
    public $speed;
    public $fuel;
}
```

```php
<?php
class Car {
    private $seat = 5;
    public $speed;
    public $fuel;
}
```

Mit einer Klasse ein Objekt erzeugen

- Üblich Beschreibung der Klasse in eine eigene Datei auszulagern, deshalb erst einbinden
- Zugriff auf die `private` Member `sitzplaetze` direkt aus dem Hauptprogramm nicht möglich

› class_auto.php

```php
<?php
class Car {
    private $seat = 5;
    public $speed;
    public $fuel;
}
```

› Hauptprogramm

```php
<?php
include ("class_auto.php");
$myCar = new Car();

$myCar->speed = 150;
$myCar->fuel = 7;

print "Geschwindigkeit: " .$myCar->speed." km/h<br>\n";
print "Kraftstoffverbrauch: ".$myCar->fuel." 1100km<br>\n";
?>
```

Methoden für die Arbeit mit Objekten

- Um mit Objekten zu arbeiten, kommen häufig Methoden zum Einsatz
- Ähnlich aufgebaut wie Funktionen
- Unterschied: Methoden lassen sich lediglich auf Objekte der entsprechenden Klasse anwenden
- Methoden erlauben es auf private Eigenschaften des Objekts zuzugreifen
- Wird gleich wie eine Funktion eingeführt nur zusätzlich mit der Methode öffentlich oder privat

› class_auto.php

```php
<?php
class Car {
    private $seat = 5;
    public $speed;
    public $fuel;

    public function setSeat($count) {
        $this->seat = $count;
    }
}
```

› Hauptprogramm

```php
<?php
include ("class_auto.php");
$myCar = new Auto();

$myCar->speed = 150;
$myCar->fuel = 7;

print "Geschwindigkeit: " .$myCar->speed." km/h<br>\n";
print "Kraftstoffverbrauch: ".$myCar->fuel." 1100km<br>\n";
$myCar->setSeat(2);
```

Übung

1. Erzeuge eine Klasse für Produkte, die es ermöglicht, die festzulegen. Verwende für die Namen der einzelnen Member die Bezeichnungen Artikelnummer, Produktname, Preis, Beschreibung, Anzahl. Verhindere dabei den Zugriff von außerhalb der Funktion.
2. Erstelle Methoden, die es ermöglichen, Änderungen an den einzelnen Membern vorzunehmen. Erstelle ein Hauptprogramm, das das Objekt Bohrmaschine erzeugt und die Eigenschaften wie in Übung 1 vorgibt.
3. Erstelle eine Methode getSeats mit der du die Sitzplätze von Folie 4 ausgeben kannst. Wende die Methode im Hauptprogramm an

Dateien für die Speicherung von Daten

Daten aus einer Datei einlesen

```php
<?php
$handle = fopen('beispiel.txt', 'r');

$result = array();
if($handle) {
    $i = 0;
    while(!feof($handle)) {
        $content = fgets($handle);
        $result[$i] = $content;
        $i++;
        print $content."<br>\n";
    }
    fclose($handle);
} else {
    print "Datei konnte nicht geöffnet werden<br>\n";
}
```

- https://www.php.net/manual/de/function.fopen.php
- https://www.php.net/manual/de/function.fgets.php
- https://www.php.net/manual/de/function.feof.php
- https://www.php.net/manual/de/function.fclose.php

- Befehl `fopen()` öffnet eine Datei oder URL
- nach dem Dateinamen folgt der Modus => spezifiziert Zugriffstyp
  - `r` - nur zum Lesen geöffnet, platziert Dateizeiger am Dateianfang
  - `r+` - Lesen und Schreiben, platziert Dateizeiger am Dateianfang
  - `w` – nur zum Schreiben, platziert Dateizeiger am Dateiende.
  - `w+`- zum Schreiben und Lesen geöffnet, platziert Dateizeiger am Dateiende. Existiert Datei nicht, versucht, diese zu erzeugen
  - a – zum Erweitern der bestehenden Inhalte
- `fopen()` erzeugt als Rückgabewert einen Handle => bietet im Programm Zugriff auf Datei und muss in Variable gespeichert werden
  - Üblich `$handle` oder `$fh` (file handle) zu nennen
- Um jede Zeile aus File einzulesen => `while`-Schleife
  - `feof` - Prüft, ob ein Dateizeiger am Ende der Datei steht.
  - Solange er nicht am Ende ist hat er Wert `false`
- `fgets()` - Liest eine Zeile von der Position des Dateizeigers.
- `fclose()` - schließt den Zugriff auf die Datei wieder und sollte immer nach der letzten Verwendung des Handles eingefügt werden

Daten in einer Datei speichern

- Modus beim Öffnen für das Schreiben: `w`
- `fputs()` - Diese Funktion ist ein Alias für: `fwrite()`
  `fwrite()` schreibt den Inhalt der Zeichenkette `string` in die Datei, auf welche der Dateizeiger `handle` zeigt.
- Mit `\n` Zeilenumbruch in der Datei erzeugen

```php
<?php

$handle = fopen('beispiel.txt', 'w');
if($handle) {
    for($i = 0; $i < 10; $i++){
        fputs($handle, ($i+1) . "\n");
    }
    fclose($handle);
} else {
    print "Datei konnte nicht geöffnet werden.<br>\n";
}
```

- https://www.php.net/manual/de/function.fwrite.php

Beispiel

```php
<form method="post" action="sameSiteAgain.php">
    Name: <input type="text" name="name"><br>
    E-Mail: <input type="text" name="e-mail"><br>
    <input type="submit" value="Senden">
</form>
<?php
$handle = fopen("beispiel.txt","w");
if ($handle) {
    if (!empty($_POST['name']) && !empty($_POST['e-mail'])) {
        if ($_POST['name'] != "") {
            $name = $_POST['name'];
            fputs($handle, $name."\n");
        }
        if ($_POST['e-mail'] != "") {
            $email = $_POST['e-mail'];
            fputs($handle, $email."\n");
        }
    }
    fclose($handle);
} else {
    print "Die Datei konnte nicht geöffnet werden.<br>\n";
}
?>
```

- Superglobale Variablen
  Superglobals — Superglobals sind Built-in-Variablen, die immer in allen Gültigkeitsbereichen (s.g. Scopes) verfügbar sind
  $_GET
  $_POST
  $_REQUEST
  $_SERVER
  …

- https://www.php.net/manual/de/language.variables.superglobals.php
- https://www.php.net/manual/de/reserved.variables.request.php
- https://www.php.net/manual/de/reserved.variables.post.php
- https://www.php.net/manual/de/reserved.variables.get.php

Die Dateirechte beachten

- Auf Linx-Webservern gibt es Zugriffsrechte auf Dateien
- Sollen verhindern, dass Unbefugte die Datei verwenden, löschen, ändern
- Rechte werden in Zahl gespeichert
  - 0 – an der Datei bestehen keine Rechte
  - 1 – Datei darf ausgeführt werden
  - 2 & 4 – Lese- bzw. Schreibrechte
  - 6 – Lese und Schreibrechte
  - 3 – Nutzer darf Datei ausführen und lesen
- Wenn alle Nutzer Lese- und Schreibrechte erhalten sollen: 0666

Übung

- Schreibe ein Programm, das ein Formularfeld im Browser anzeigt. Dieses soll den Nutzer auffordern, eine beliebige Zahl einzugeben. Erstelle ein Textdokument mit verschiedenen Zahlen (jeweils in einer eigenen Zeile) und lese diese mit dem Programm ein. Multipliziere daraufhin die Zahlen mit dem Wert, den der Nutzer über den Browser eingegeben hat. Erstelle ein neues Textdokument und speichere die Ergebnisse darin ab.
  Achtung: Zahlen aus Formularfeldern sollten mit intval() erst in einen Integer formatiert werden

ENDE

QUELLE: PHP & MYSQL FÜR EINSTEIGER
ISBN: 978-3-96645-009-6