Objektorientierte Programmierung für Anfänger am Beispiel PHP

Johannes Mittendorfer
http://jmittendorfer.hostingsociety.com

19. August 2012

Abstract

Dieses Dokument soll die Vorteile der objektorientierten Programmierung erläutern und den Leser in die grundlegenden Begriffe und die Verwendung von Klassen, Objekten und Instanzen einführen. Es stellt keinesfalls eine komplette Beschreibung der OOP dar und erhebt auch nicht den Anspruch darauf. Vielmehr soll dieses Dokument nur zur Einführung in diese Methode des Programmierens dienen.

Inhaltsverzeichnis

1 Einleitung

Immer wieder hört man von den vielen Vorteilen der objektorientierten Programmierung. Viele können jedoch damit nicht viel anfangen. Auch mir war anfangs nicht klar, wozu ich diesen Programmierstil verwenden soll.

Doch wenn man die Struktur einmal erlernt hat, tun sich viele neue Möglichkeiten auf, so ist es auch möglich größere Projekte übersichtlich zu programmieren. Diese können so auch meist an andere zur Mitarbeit weitergegeben werden.

In diesem Dokument möchte ich die Grundprinzipien der objektorientierten Programmierung, kurz OOP, anhand der Sprache PHP aufzeigen und in einfachen Beispielen verständlich machen.

2 Wozu OOP?

Objektorientierung ist heute ein Grundprinzip der Programmierung ohne den niemand mehr auskommen sollt. Nur durch dessen Verwendung ist es möglich Programme, annähernd egal in welcher Sprache, zu strukturieren, sodass sie einfach verstanden werden können und auch bei längeren Texten übersichtlich bleiben.

2.1 Übersichtlichkeit

Wohl jeder, der eine Programmiersprache erlernt beginnt mit einem Programm in der prozeduralen Schreibweise. Das heißt: Das Programm wird von oben nach unten durchgearbeitet und springt höchstens mit einem *GOTO*-Befehl an eine andere Stelle.

Ein Beispiel für ein Programm in der prozeduralen Schreibweise, das die Zahlen von 1 bis 10 ausgibt:

```
<?php
  $counter = 0;
  start:
  $counter++;
  echo $counter;
  if($counter < 10){
    goto start;
  }
```

```
?>
```

Dieses Beispiel ist noch ziemlich einfach zu verstehen, aber wenn das Programm länger wird, ist man fast chancenlos es zu verstehen, gerade wenn man es nicht selbst geschrieben hat.

In der objektorientierten Schreibweise ist es meist ein Leichtes ein Programm zu verstehen, auch wenn man es nicht geschrieben hat:

```
<?php
  class Counter{
    public function count($from, $to){
      $output = "";
      $counter = $from;
      while($counter <= $to){
        $counter++;
        $output += $counter;
      }
      return $output;
    }
  }

  $c = new Counter();
  echo $c->count(1,10);
?>
```

Dies mag auf den ersten Blick komplizierter als das vorhergehende Beispiel aussehen, ist aber sehr viel einfacher zu verstehen, wenn sich einmal mit OOP auseinandergesetzt hat.

2.2 Einbindung in andere Dateien

In einem echten Programm wird es jedoch so aussehen, dass nur die beiden letzten Zeilen im Hauptprogramm stehen. Die Definition wir einfach in eine Datei ausgelagert:

```
<?php
  include("Counter.class.php");

  $c = new Counter();
  echo $c->count(1,10);
?>
```

Man muss sich bei dessen Benützung also gar nicht mit dem Inhalt auseinandersetzen, sondern kann einfach die Funktionen verwenden, die möglicherweise auch von einem anderen Programmierer erstellt worden sind.

3 Begriffe

3.1 Klasse

Die Klasse beinhaltet eine oder mehrere Methoden, die das eigentliche Programm ausmachen. Eingeleitet wir sie mit dem Codewort *class*.

```
<?php
  class MeineKlasse{
    // Methoden
  }
?>
```

3.2 Objekt

Das Objekt ist eine Instanz einer Klasse. Es können mehrere Instanzen einer Klasse erzeugt werden, die dann jeweils wieder im Ausgangszustand bereitsteht. Eine Instanz erzeugt man so:

```
<?php
  $my = new MeineKlasse();
?>
```

4 Beispiel Auto

An einfachsten ist alles noch immer mit einem Beispiel erklärt. Stellen Sie sich vor, Sie können mittels PHP Autos erzeugen.

4.1 Die Klasse

Natürlich benötigen wir als Erstes eine Klasse *Auto*:

```
<?php
  class Auto{
    private $farbe = "";

    function Auto($meinefarbe){
      $this->farbe = $meinefarbe;
    }
```

```
    public function zeigeFarbe(){
      return $this->farbe;
    }
  }
?>
```

Mittels `$farbe = "rot";` definiert man innerhalb der Klasse eine Eigenschaft. Eine Eigenschaft verhält sich wie eine Variable, die jedoch durch das Codewort *private* nur innerhalb der Klasse angesprochen werden kann.

Die Methode (Funktion) *Auto* ist ein sogenannter Konstruktor. Sie wird ausgrufen, wenn eine Instanz der Klasse erzeugt wird. Mittels eines Parameters im Konstruktor kann man eine Instanz mit einer anderen Eigenschaft erzeugen. In diesem Beispiel ein „Auto“mit der angegebenen Farbe.

Mit `$this->farbe = $meinefarbe;` weißt man die Farbe der Eigenschaft `$farbe` zu. `$this` bezieht sich immer auf die aktuelle Klasse.

Über die Funktion *zeigeFarbe* wird die Farbe wieder zurückgegeben. Sie ruft die Eigenschaft `$farbe` ab ung gibt sie mittels `return` zurück. Durch das Codewort *public* ist diese Funktion auch von außerhalb der Klasse erreichbar.

4.2 Die Instanz(en)

Um nun „Autos“zu erzeugen ist folgender Code notwendig:

```
<?php
  $rot = new Auto("rot");
  $blau = new Auto("blau");

  echo $rot->zeigeFarbe();
  echo $blau->zeigeFarbe();
?>
```

Es werden dabei zwei Instanzen der Klasse *Auto* erzeugt. Ein Auto ist Rot, das andere Blau. Nun kann über die Funktion *zeigeFarbe* die jeweiligen Farben ausgegeben werden.

5 Zusammenfassung

Die objektorientierte Programmierung stellt eine erhebliche Verbesserung des Programmierstils dar. Die Verwendung mag am Anfang etwas mühsam und

sinnlos erscheinen, aber beim ersten großen Projekt ist man über die klare Strukturierung des Programmes froh.

Darum ist zu empfehlen, dass bereits bei kleinen Programmen die Objektorientierung verwendet wird, um einen späteren Ausbau der Funktionen zu ermöglichen.

Ich hoffe ich habe Ihnen mit diesem Dokument die Möglichkeiten und Vorteile der OOP etwas näher gebracht.

6 Weiterführende Link

Erklärung von OOP bei Wikibooks:

http://de.wikibooks.org/wiki/Websiteentwicklung:_PHP:_OOP

Erklärung von Peter Kropff:

http://www.peterkropff.de/site/php/oop.htm