OOP Objektorientierte Programmierung in PHP - Part 3

Hallo liebe Community!

Dies ist mein erstes Tutorial also seit nicht zu streng mit der Kritik, über Verbesserungsvorschläge würde ich mich dennoch freuen!

Ich setze voraus, dass man weiß wie Funktionen geschrieben werden und dass man mit Variablen umgehen kann. Außerdem sollte man folgende Parts meines Tutorials gelesen haben:

- [wiki]OOP Objektorientierte Programmierung in PHP - Part 1[/wiki]
- [wiki]OOP Objektorientierte Programmierung in PHP - Part 2[/wiki]

Ich werde weiterhin mit unseren Tollen Raumschiffen spielen xD

So, jetzt stellt euch vor das euer Browsergame das beste auf der Welt ist und ihr es ewig weiter entwickelt. So nach 50 Jahren seid ihr dann sehr alt und auch ein wenig vergesslich. Nun vergesst ihr zum Beispiel dass man Raumschiffe nicht umbauen kann! Ein Jäger wird niemals ein Zerstörer werden! Dafür und für andere wesentlich sinvollere Anwendungsbereiche wurden die Konstanten erfunden (wer nicht weiß was eine Konstante ist, PN dann kommt nen Tut, sonst php manual). In einer Klasse kann man leider nicht einfach eine Konstante definieren. Man muss einer Klassenkonstanten einen konstanten Wert geben, und das schon zur implementierung des Programms. Das heißt man darf keine Funktion, keinen Variable, nicht mal eine mathematische Rechenoperation benutzen, um den Wert einer Konstanten zu definieren. Das sieht dann so aus:

Quellcode

```
1.  <?php
2.  class Jaeger
3.  {
4.  private $leben;
5.  private $farbe
6.  const RAUMSCHIFFTYP = "Jäger";
7.  public function __construct($farbe = "schwarz")
8.  {
9.  $this->farbe = $farbe;
10. $this->leben = 100;
11. }
12. public function fliegen()
13. {
14. echo "I beliiiiiiiieeeeve I can flllaaaayyyyy";
15. }
16. public function getLeben()
17. {
18. return $this->leben;
19. }
20. public function setLeben($leben)
21. {
22. $this->leben = $leben;
23. }
24. public function getFarbe()
25. {
26. return "Das Raumschiff ist in ".$this->farbe." angestrichen";
27. }
28. public function getRaumschifftyp()
29. {
30. return self::RAUMSCHIFFTYP;
31. }
```

```
38. }
39. 
40. $Schiff = new Jaeger();
41. echo $Schiff->getLeben();
42. echo $Schiff->getFarbe();
43. echo $Schiff->getRaumschifftyp();
44. echo $Schiff::RAUMSCHIFFTYP;
45. echo Jaeger::RAUMSCHIFFTYP;
46. ?>
```

Alles anzeigen

Am Anfang der Klasse wird die Konstante (constant) initialisiert. Diese kann sich im gesamten Script NICHT mehr ändern. Um auf die Konstante zuzugreifen wird eine andere Variante wie für "normale" Eigenschaften benutzt (Z. 36,44&45). Im 1. Part erwähnte ich schon das "->" der Pfeil nicht die einzige Möglichkeit ist auf properties zuzugreifen. Der doppelte Doppelpunkt "::" ist eine weitere Art des Zugriffs. Mit dem "::" kann man jedoch nur auf statische (static, wird wahrscheinlich in Part 5 dran kommen) Eigenschaften von KLASSEN (nicht Objekten) und auf Klassenkonstante (die ja auch statisch sind) zugreifen. Um auf eine Klassenkostante zuzugreifen verwendet man in der Klasse selber statt "$this" "self". Um von außerhalb der Klasse auf die Konstante zuzugreifen kann man entweder das Objekt oder den Klassennamen, jeweils gefolgt von dem "::" verwenden. Der Übersichtlichkeit wegen nimmt man eher das Objekt ("$Schiff").

Im nächsten Part gehe ich auf Vererbung ein. Interfaces werden NICHT behandelt (einige meinen das hätte mit Vererbung zu tun...) [wiki]OOP Objektorientierte Programmierung in PHP - Part 4[/wiki]

n0x-f0x