OOP Objektorientierte Programmierung in PHP - Part 9

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]
- [wiki]OOP Objektorientierte Programmierung in PHP - Part 3[/wiki]
- [wiki]OOP Objektorientierte Programmierung in PHP - Part 4[/wiki]
- [wiki]OOP Objektorientierte Programmierung in PHP - Part 5[/wiki]
- [wiki]OOP Objektorientierte Programmierung in PHP - Part 6[/wiki]
- [wiki]OOP Objektorientierte Programmierung in PHP - Part 7[/wiki]
- [wiki]OOP Objektorientierte Programmierung in PHP - Part 8[/wiki]

In diesem Part werde ich erklären, wie man eine Klasse, von der nur ein Objekt erstellt werden kann.

Der Konstruktor muss private sein, damit nur die Klasse selber ein Objekt erstellen kann. Es muss eine statische Methode geben, die den Konstruktor aufruft. In dieser Methode muss noch geprüft werden, ob das Objekt schon erstellt wurde. Die Methode muss auchnoch das Objekt zurückgeben. Und es muss eine statische Eigenschaft geben, um das Objekt zu speichern.

Quellcode

```
1. class Foo
2. {
3. private static $obj = null;
4. private function __construct() //Der Konstruktor ist private
5. { }
6. private function __clone() //Diese Funktion wird aufgerufen wenn versucht wird, ein Objekt der Klasse zu klonen.
7. { } //Sie muss auch private sein, damit nur ein Objekt erstellt werden kann!
8. private static function getObj() //statische Methode um das Objekt zu bekommen
9. {
10. if(self::$obj === null) //Wenn es das objekt noch nicht gibt...
11. {
12. self::$obj = new self(); //erstelle es
13. }
14. return self::$obj; //gib das Objekt zurück
15. }
16. }
```

Alles anzeigen

Die Methode __clone wird (so ähnlich wie __construct oder __destruct) zu einem Bestimmten Event aufgerufen. Zu den sogennanten "magischen methoden" im nächsten Part mehr!

n0x-f0x