Mini-PHP-Coding-Guidelines

Autor	Nachricht	
oxpus	Betreff des Beitrags: Mini-PHP-Coding-Guidelines	Do 11.MÃ¤r, 2010 18:04

Kurzer Ausflug zu PHP-Kommentaren

Code:
```
//
```

kommentiert die Zeile aus, vor der diese Zeichen stehen.

PHP ignoriert damit zur Laufzeit des Scripts den kompletten Rest der betreffenden Zeile.

Dieses kann auch nach einem Befehl eingetragen werden. Zum Beispiel so:

Code:
```
echo 'Hello World'; // Einfache Ausgabe
```

Damit führt PHP den echo-Befehl noch aus, alles was aber hinter // in dieser Zeile kommt, wird wieder ignoriert.

Code:
```
/*
```

Beginn eines Kommentarbereiches

Code:
```
*/
```

Ende eines Kommentarbereiches

Alles, was zwischen /* und */ steht, wird von PHP als Kommentar angesehen und zur Laufzeit komplett ignoriert.

Die erste Methode verwendet man, um eine oder wenige Zeilen an Kommentar im Script zu hinterlegen.

/* und */ wird dagegen für grössere Kommentarabschnitte verwendet.

So findest man in phpBB 3-Dateien meist am Anfang einen Kommentarbereich, der wie folgt aussieht:

Code:
```
/*
*
* @bezeichner     Kommentartext
*
*/
```

All dieses wird von PHP komplett ignoriert.

Auch wäre sowas denkbar:

Code:
```
/*
Kommentartext
*/
```

Kurzer Abstecher zu den einzelnen Code-Zeilen und "Begrenzern" von Befehlen

<u>Methode 1:</u>

Für PHP wird ein Befehl mit einem ";" (Semikolon - ohne die hier dargestellten Hochkommas) abgeschlossen.

So stellt für PHP

Code:

```
echo ('Hello World');
```

einen Befehl dar, der auszuführen ist.

Übrigends kann man das auch so schreiben:

Code:

```
echo 'Hello World';
```

Funktioniert, ist aber nicht "sauber" programmiert.

Und genau das wäre dann falsch:

Code:

```
echo 'Hello World' echo 'Ich bins';
```

PHP kann damit nichts anfangen und zeigt eine Fehlermeldung an, da nach 'Hello World' für PHP der echo-Befehl zu Ende ist und ein Semikolon erwartet wird, aber ein weiteres echo erscheint.

Das Semikolon darf daher am Ende eines Befehles, der nicht mit geschweiften Klammern erstellt wird (siehe weiter unten) nie fehlen.

Ausser: Es wäre der letzte Befehl im Script, dann dürfte das auch sein.

So ist diese Datei auch korrekt, trotz fehlendem Semikolons am Ende des Befehls:

Code:

```
<?php

phpinfo()

?>
```

Eine solche Datei würde übrigends die PHP-Infos anzeigen, die man auch im Admin-Bereich mit der entsprechenden Funktion anzeigen lassen kann.

Dennoch sollte auch hier phpinfo() besser auch mit einem Semikolon abgeschlossen werden.

Also so:

Code:

```
phpinfo();
```

Man kann durch das Semikolon aber auch mehrere Befehle in einer Zeile hintereinander schreiben.

So zum Beispiel:

Code:

```
echo ('Hello World'); $a = $a + 1; $b = 100;
```

Funktioniert prima, ist aber unleserlicher.

Daher sollte nach jedem Ende eines Befehls mindestens eine neue Zeile anfangen.

Also nach unserem Beipiel wäre das "sauberer" Code:

Code:

```
echo ('Hello World');
$a = $a + 1;
$b = 100;
```

Besser wäre noch

Code:

```
echo ('Hello World');

$a = $a + 1;
$b = 100;
```

Die zusätzliche Leerzeile hilft mit, den Code besser lesen zu können.

Es gibt aber auch noch die Möglichkeit, einen Befehl mit geschweiften Klammern zu definieren.
Also { für den Anfang und } für das Ende des Bereiches.
Das wird z. B. bei Schleifen und Bedingungen oft eingesetzt.
Hier ein paar "saubere" Beispiele:

Code:
```
// Bedingung
if ($a == 1)
{
// Code bei "WAHR"
}
else
{
// Code bei "FALSCH"
}
```

Code:
```
// Schleife mit for
for ($i = 1; $i < sizeof($array); $i++)
{
// Auszuführender Code
}
// Hier wird innerhalb der "Schleifendefinition" jeder "Befehl" mit einem Semikolon vom nächsten getrennt!!
```

Code:
```
// Schleife mit while
while ($row = $db->sql_fetchrow($result))
{
// Auszuführender Code
}
```

Code:
```
// Schleife mit do...while
do
{
// Auszuführender Code
}
while ($row = $db->sql_fetchrow($result));
```

Achtung: Bei letzterer Schleife wird zwar ein Code-Bereich durch die Klammern eingeschlossen und damit innerhalb der Schleife abgearbeitet, da aber nach while keine weiteren Anweisungen mehr folgen, die zur Schleife gehören, muss hier ein Semikolon als "Befehlende" gesetzt werden!