



PHP-Counter IP

EASY-LINE

Entwurf und Umsetzung eines
PHP basierten Zählers mit MySQL Anbindung

Die jeweils aktuelle Version finden Sie auf der Homepage
www.honigs-home.de

Stand: 06.08.2004

© 2001–2004
by Holger Niggemann

Inhaltsverzeichnis

Inhaltsverzeichnis.....	2
Eigenschaften unseres Counters?.....	3
Datei oder Datenbank?	3
Entwurf der Datenbank	3
Datenbank erstellen.....	4
Der Counter	5
Datenbankverbindung prüfen.....	5
Konfiguration festlegen.....	5
Alte IP's löschen.....	5
Gesamtzählerstand ermitteln	6
Zählerstand erhöhen	6
Formatierte Ausgabe.....	6
Ausgabe falls die DB nicht läuft.....	7
Dateien erzeugen	7
Quellcode Datei: counter.php	8
Quellcode Datei: test.php.....	10
Quellcode Datei: dbTabelle.sql.....	11



Oft findet man auf Internetseiten einen Counter (Zähler), der die Anzahl der Besucher wiedergibt. Einige Seiten verwenden dabei einen externen Counter von einem Drittanbieter. Wir wollen uns hier mit dem Entwurf und der Umsetzung eines eigenen Counters beschäftigen. Hierdurch erreichen wir

- eine optimale Integration und Anpassbarkeit an unser Seitenlayout
- keine lästigen Werbeeinblendungen
- „echte“ Besucherzahlen durch IP-Sperren
- gute Verfügbarkeit
- hohe Geschwindigkeit

Eigenschaften unseres Counters?

Wir beginnen mit der Überlegung welche Eigenschaften unser Counter besitzen soll:

Von Vorteil wäre es wenn wir die Mindestanzahl der Zahlenstellen bestimmt werden könnten. Solange diese unterschritten wird, sollen die fehlenden Stellen mit Nullen aufgefüllt werden (z.B. 12 => 00012).

Ein Besucher sollte innerhalb eines bestimmten Zeitraumes nur einmal gezählt werden. Das künstliche Erhöhen durch einfaches Betätigen der Reloadtaste soll somit verhindert werden. Dieser Zeitraum muss anpassbar sein.

Eine Anzeigeunterdrückung dürfte ebenfalls von Vorteil sein (z.B. wenn der User nicht über die Startseite auf die Homepage gelangt).

Das Layout des Counters sollte ohne Änderung am Quelltext anpassen sein.

Datei oder Datenbank?

Prinzipiell gibt es zwei unterschiedliche Ansätze. Die Daten werden in eine Textdatei oder in einer Datenbank geschrieben.

Das verwenden einer Textdatei ist etwas einfacher zu realisieren. Problematisch ist diese Variante allerdings sobald mehrere Besucher gleichzeitig auf die Seite zugreifen. Um eine Datei zu ändern, muss diese zuerst geöffnet werden, dann müssen die entsprechenden Daten gelesen und ausgewertet werden, erst jetzt erfolgt das Schreiben und die Datei wird wieder geschlossen. Während des kompletten Vorganges benötigt man exklusive Zugriffsrechte und die Datei ist während dieses Zeitraumes für andere Verwendungen gesperrt.

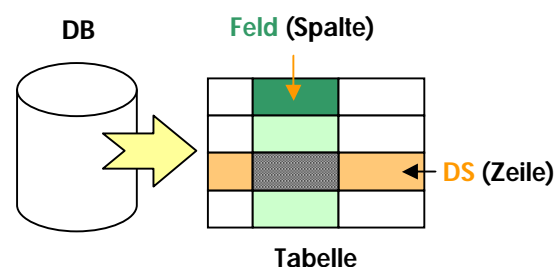
Alternativ lassen sich die Daten in einer Datenbank ablegen. Die DB ist in der Lage mehrere Zugriffe "gleichzeitig" auszuführen. Weit verbreitet ist die MySQL Datenbank.

Wer hohe Benutzerzahlen (mehrere hundert pro Tag) erwartet, sollte lieber auf die Variante mit der DB setzen. Wir entscheiden uns hier ebenfalls für die Datenbank Variante.

Entwurf der Datenbank

Bevor wir mit der Programmierung des Counters beginnen, müssen wir uns um die Planung und Erstellung der DB kümmern.

Hier ein kurzer Überblick zum Thema DB.



Eine DB besteht in der Regel aus mehreren Tabellen, indem die einzelnen Datenfelder untergebracht sind. Jedes Datenfeld repräsentiert dabei den Kopf einer Spalte. Alle Datenfelder bilden einen Datensatz und entsprechen einer Zeile der Tabelle.



Unser Counter soll in einer einzigen Tabelle untergebracht werden. In dieser müssen wir eine Möglichkeit schaffen die **IP-Adresse** und das **Datum** zu speichern. Die IP-Adresse bringen wir in einem Datenfeld vom Typ **varchar** (Zeichen) mit einer Länge von **20** unter. Beim Datum gibt es zwei Möglichkeiten. Da wäre die Verwendung des Datentyps **timestamp**, der einen Zeitabdruck im Format Jahr, Monat, Tag, Uhrzeit hinterlässt. Alternativ dazu lässt sich auch der Datentyp **bigint unsigned** (ohne Vorzeichen => Verdopplung des möglichen Wertebereiches) verwenden. In diesem werden einfach die Sekunden die seit einem bestimmten Bezugsdatum verstrichen sind, gespeichert. Wir haben uns in diesem Fall für die **bigint**-Variante entschieden.

Damit jeder Datensatz einzigartig ist und es den gleichen Datensatz nicht mehrfach gibt (mehrere Rechner hinter einem Router greifen gleichzeitig auf die Seite zu), führen wir zusätzlich ein Feld **'id'** ein, welches sich bei jedem neuen Datensatz automatisch um 1 erhöht (AUTO_INCREMENT). Als Datentyp wählen wir hier ebenfalls **bigint, unsigned**, da wir ja auf sehr sehr viele Besucher hoffen ;-)

id	ip	edat
1	120	
...		
100	127.0.0.1	1090956847
101	192.120.26.2	1090956853
...		

Den aktuellen Zählerstand speichern wir in der Zeile **id=1** im Feld **ip** ab (hier 120). Auf diese Weise sparen wir eine Spalte, die nur einmal genutzt würde.

Datenbank erstellen

Bei vielen Providern kommt MySQL zum Einsatz. Meist wird es so sein, dass Ihr Provider Ihnen eine Datenbank vorgibt oder

dass Sie diese automatisch (z.B. über Confixx...) erzeugen können. Sollten Sie jedoch mit der MySQL-Konsole arbeiten, so müssen Sie sich mit Benutzernamen und Ihrem Passwort an der Konsole anmelden. Verwenden Sie folgendes Statement zum Erzeugen der Datenbank:

```
CREATE DATABASE IF NOT EXISTS `dbname`
```

`dbname` steht hierbei für den Namen den Ihre Datenbank tragen soll.

Um uns das weitere vorgehen zu vereinfachen, benutzen Sie folgendes Skript um eine Tabelle "Counter" und die Datenfelder anzulegen.

```
# Rechner: localhost
# Datenbank: testdb
# Tabelle: 'counter'

CREATE TABLE `counter` (
  `id` bigint(20) unsigned
  NOT NULL auto_increment,
  `ip` varchar(20) NOT NULL default '',
  `edat` bigint(20) unsigned
  NOT NULL default '0',
  PRIMARY KEY (`id`)
) TYPE=MyISAM;

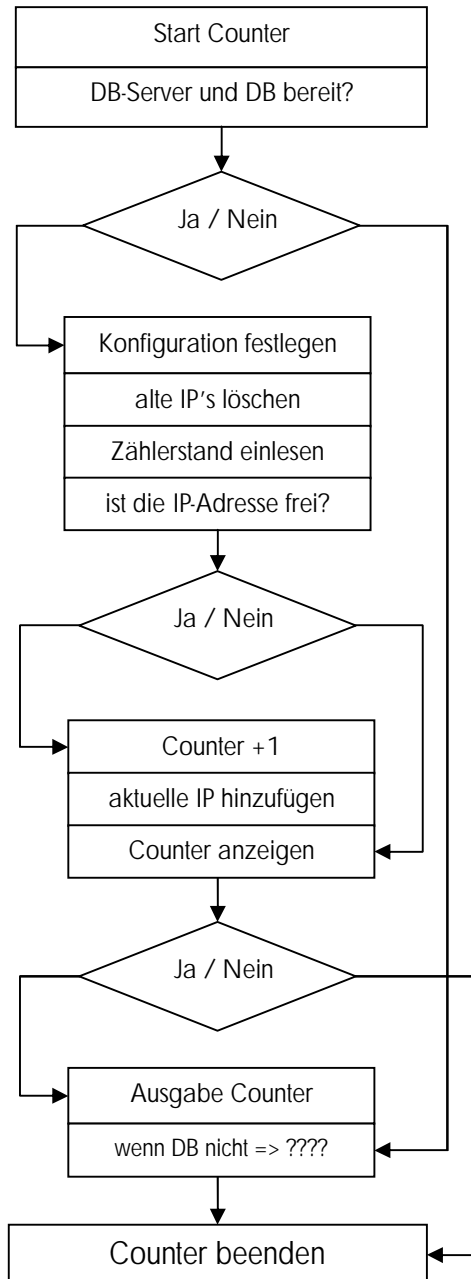
INSERT INTO counter (id, ip)
VALUES (1,0);
```

Komfortabler lassen sich MySQL-Datenbanken über so genannte FrontEnd's bedienen. Ein Vertreter dieser Gattung ist z.B. das MySQL Control Center (Windows und Linux), welches auf der Homepage <http://www.mysql.com/> kostenlos zum Download angeboten wird. Hier können Sie das obige Skript einfach in den Bereich SQL-Abfrage über die Zwischenablage einfügen und ausführen.

Eine weitere nützliche auf PHP basierende Variante, ist phpMyAdmin. Diese wird häufig von Providern eingesetzt, da das ganze „Skript“ im Browser läuft. Für denjenigen, der einen Server und PHP zu Hause laufen hat, steht das "Programm" kostenlos zum Download unter <http://phpmyadmin.sourceforge.net/> bereit.

Der Counter

Nach der ganzen Vorbereitung kommen wir nun endlich zum eigentlichen, dem Counter. Das folgende Ablaufdiagramm gibt die prinzipiellen Überlegungen zum Counter wieder.



Datenbankverbindung prüfen

Direkt nach dem Start muss geprüft werden, ob die Verbindung zum Datenbank-Server und zur Datenbank funktioniert. Sollte dieses nicht der Fall sein, so kann der Counter nicht arbeiten und wird sofort mit Ausgabe von ? beendet.

Konfiguration festlegen

Am Anfang unseres Skriptes wollen wir mit Hilfe von Variablen unseren Counter konfigurieren. Dieses hat später den Vorteil, dass bei Änderungen nicht der ganze Quelltext durchsucht werden muss. Bei den Namen der Variablen sollte man immer größte Sorgfalt walten lassen. Eine später nicht mehr zuordbare Variablenstruktur erschwert die Fehlersuche oder eine spätere Erweiterung. Variablen die hier mit \$comp beginnen dienen dem Vergleich (compare).

```
// Konfiguration
$digits = 5;
$compTime = time()-10;
$compIP = $GLOBALS["tempIP"];
```

Mit \$digits legen wir die minimale Anzahl der Stellen fest. In \$compTime speichern wir die aktuelle Zeit und ziehen von dieser die Anzahl der Sekunden ab, für die eine IP-Adresse nicht gezählt werden soll. In \$compIP speichern wir genau diese IP-Adresse zwischen (\$GLOBALS, da die Variable außerhalb der Funktion deklariert wurde).

Alte IP's löschen

Nun ist es an der Zeit die alten IP-Adressen, also diejenigen die älter sind als die Zeit, die in \$compTime festgelegt wurde, aus der DB zu löschen. Genau das bewirkt folgende MySQL Formulierung:

```
// alte IP-Adressen freigeben
$query = "DELETE FROM counter
         WHERE edat<='<math>\$compTime</math>' AND
         id!='1'";
mysql_query($query);
```

Gesamtzählerstand ermitteln

Jetzt ist ein guter Zeitpunkt den aktuellen Gesamtzählerstand aus der DB zu lesen. Dieses wollten wir ja immer im Datenfeld id=0 unter ip speichern. Folgende Anweisung liebt den Zählerstand aus und speichert ihn in der Variable \$counter.

```
// aktuellen Counterstand auslesen
$query = "SELECT ip FROM counter
         WHERE id='1'";
$result = mysql_query($query);
$row = mysql_fetch_row($result);
$counter = $row[0];
```

Zählerstand erhöhen

Bevor wir den Gesamtzählerstand um 1 erhöhen, müssen wir prüfen, ob die IP-Adresse des aktuellen Besuchers (\$compIP) noch gesperrt ist. Hierzu suchen wir einfach in der gesamten Tabelle nach dieser IP. Die Anzahl der Treffer muss > 1 sein.

```
// schauen ob aktuelle IP gesperrt
$query = "SELECT ip FROM counter
         WHERE ip='$compIP'";
if ( dbCount($query) < 1) {
    // IP ist frei, IP eintragen
    $query = "INSERT INTO counter
            (ip,edat) ".VALUES
            ('$compIP',".time().")";
    mysql_query($query);
    // Counter +1
    $counter = $counter+1;
    $query = "UPDATE counter SET
            ip=$counter WHERE id='1'";
    mysql_query($query);
}
```

Wenn dieses stimmt, können wir gleich die aktuelle IP-Adresse in der Datenbank eintragen. Somit ist diese IP dann für xxx Sekunden (festgelegt in der Konfigurationsvariable \$compTime) für die nächsten Besuche gesperrt. Nun bleibt nur noch den Zählerstand um 1 zu erhöhen und diesen neuen Wert in die DB einzutragen. Sollte die Trefferanzahl >= 1 sein so ist diese Adresse noch gesperrt und es passiert nicht.

dbCount sowie dbOpen sind eigene Funktionen die uns den Umgang mit der Datenbank vereinfacht und zusätzlich den Quellcode des Counter übersichtlicher gestaltet. Den Quellcode finden Sie am Ende dieses Tutorials im Quelltext von counter.php.

Formatierte Ausgabe

Wenn eine Ausgabe des Counters erfolgen soll

```
if ($view == "true") {
    // Nullen auffuellen (Mindestanzahl)
```

ist es nun an der Zeit zu prüfen ob die Mindeslänge (Stellenanzahl) erreicht ist.

```
$x = strlen($counter);
$d = $digits - $x;
```

Sollte dieses nicht der Fall sein (\$d>1) wird mit

```
if ($d >= 1) {
    for ($i=0; $i < $d; $i++) {
        echo '<span class="counter">
            0</span>';
    }
}
```

der Counter mit den benötigten Stellen an führenden Nullen ergänzt. Nun kann der Zählerstand formatiert mit

```
// Ausgabe des Zaehlerstands
for ($i=0; $i < $x; $i++) {
    echo '<span class="counter">'.
        substr($counter, $i,
            1).'</span>';
}
else {
    echo "\n";
}
```

ausgegeben werden. Jede Zahl des Counters wird dabei mit der CSS-Klasse (Cascade Style Sheets) "counter" formatiert. Die Eigenschaften dieser Klasse (und somit das Aussehen des Counters), lassen sich in der PHP-Datei die den Counter aufruft, oder aber in einer externen Styledatei festlegen. Falls das Layout es erfordert, kann anstelle von 'span' natürlich auch ein 'div'-Block verwendet werden. Wer lieber einen grafischen Counter wünscht, braucht die

Ausgabe nur dementsprechend zu ändern. Am besten eignen sich dafür Grafiken mit dem Namen 0.gif 1.gif ...

In einer For-Schleife muss dann lediglich der Counter in Teilstrings zerlegt werden und die entsprechende Grafik ausgegeben werden.

Ausgabe falls die DB nicht läuft

Falls die Datenbankverbindung nicht funktioniert, kann man die Anzeige komplett unterdrücken, oder aber man gibt als Alternative ? anstelle der Zahlen aus. Dieses ist in einigen Fällen sinnvoll, da es Seitenlayouts gibt, die sich nicht mehr korrekt verhalten würden, falls keine Ausgabe des Counters erfolgen würde.

```
} else if ($view=="true") {  
  // Fehler DB nicht bereit ?  
  for ($i=0; $i < $digits; $i++) {  
    echo '<span class="counter">  
      ?</span>';  
  }  
}
```

Zum Schluss müssen wir die Datenbankverbindung nur noch schließen.

```
@mysql_close();  
}
```

Dateien erzeugen

Auf den nächsten Seiten finden Sie den kompletten Quelltext der Datei counter.php, test.php und dbTabelle.sql

Um den Counter zu testen machen Sie den Quelltext für die Datei counter.php, kopieren diesen über die Zwischenablage in einen Editor Ihrer Wahl und speichern die Datei unter dem Namen counter.php

Auf gleiche Weise verfahren Sie mit dem Quelltext zur Datei test.php

Speichern Sie beide Dateien in ein Verzeichnis. Nachdem Ihre Datenbank

angelegt ist und die Tabelle Counter erzeugt wurden können Sie die Datei test.php aufrufen.



Viel Spaß mit dem Counter...

Quellcode Datei: counter.php

```

<?php

$tempIP = $_HTTP_SERVER_VARS["REMOTE_ADDR"];

//-----//
// DB-Counter //
//-----//

function dbCounter($view="true") {
    if (dbOpen()) {
        // Konfiguration
        $digits = 5;
        $compTime = time()-10;
        $compIP = $GLOBALS["tempIP"];
        // alte IP-Adressen freigeben
        $query = "DELETE FROM counter WHERE edat<=' $compTime' AND id!='1'";
        mysql_query($query);
        // aktuellen Counterstand auslesen
        $query = "SELECT ip FROM counter WHERE id='1'";
        $result = mysql_query($query);
        $row = mysql_fetch_row($result);
        $counter = $row[0];
        // schauen ob die aktuelle IP noch gesperrt ist
        $query = "SELECT ip FROM counter WHERE ip=' $compIP'";
        if ( dbCount($query) < 1 ) {
            // IP ist frei oder noch nicht vorhanden -> IP eintragen
            $query = "INSERT INTO counter (ip,edat) ".
                "VALUES (' $compIP',".time().")";
            mysql_query($query);
            // Counter +1
            $counter = $counter+1;
            $query = "UPDATE counter SET ip=$counter WHERE id='1'";
            mysql_query($query);
        }
        if ($view == "true") {
            // Nullen auffuellen (Mindestanzahl)
            $x = strlen($counter);
            $d = $digits - $x;
            if ($d >= 1) {
                for ($i=0; $i < $d; $i++) {
                    echo '<span class="counter">0</span>';
                }
            }
            // Ausgabe des Zaehlerstands
            for ($i=0; $i < $x; $i++) {
                echo '<span class="counter">'.substr($counter, $i, 1).'</span>';
            }
        } else {
            echo "\n";
        }
    } else if ($view=="true") {
        // Fehler DB nicht bereit ? ausgeben
        for ($i=0; $i < $digits; $i++) {
            echo '<span class="counter">?</span>';
        }
    }
    @mysql_close();
}

```

```
//-----//
// DB-Funktionen //
// dbOpen() = true falls Verbindung erfolgreich //
// dbOpen() = false falls Fehler aufgetreten ist //
// // //
// dbCount(query) = liefert die Anzahl der Treffer einer Abfrage //
//-----//

function dbOpen() {
    // Datenbankserver Verbindung ANPASSEN AN IHRE DATEN
    $dbCon = @mysql_connect("localhost", "root", "root");
    if ($dbCon) {
        // Datenbankname ANPASSEN AN IHRE DATEN
        $dbSel = mysql_select_db("testDB", $dbCon);
        if ($dbSel) {
            return true;
        } else {
            return false;
        }
    } else {
        return false;
    }
}

function dbCount ($query) {
    $result = mysql_query($query);
    if ($result)
        return mysql_num_rows($result);
    else
        return -1;
}
?>
```

Quellcode Datei: test.php

```

<?php
echo '<?xml version="1.0" encoding="UTF-8"?>'. "\n";
echo '<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ';
echo '"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">'. "\n";
echo '<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de">'. "\n";
?>
<head>
<title>Testseite PHP-Counter-IP</title>
<style type="text/css">
<!--
/* Hier werden die Formate fuer den Counter definiert ... */
span.counter {
margin: 0px 2px 0px 0px;
padding: 1px 3px 1px 3px;
color: #3726B3;
font-size: 1.2em;
background-color: #FBF8D0;
border-right: 1px solid #FFC000;
border-bottom: 1px solid #FFC000;
border-top: 1px solid #FCDF88;
border-left: 1px solid #FCDF88;
}
-->
</style>

<?php
include ("counter.php");
?>
</head>

<body>
<h1>Testseite PHP-Counter IP</h1>
<p>
Hallo Sie sind bereit der <?= dbCounter(true) ?> Besucher.
</p>
<p>
F&uuml;r Testzwecke wurde der IP-Sperre auf 10 Sekunden gesetzt...
<br />
Viel Spa&szlig; mit dem Counter
</p>
</body>
</html>

```

Quellcode Datei: dbTabelle.sql

```
# Rechner: localhost
# Datenbank: testdb
# Tabelle: 'counter'

CREATE TABLE `counter` (
  `id` bigint(20) unsigned NOT NULL auto_increment,
  `ip` varchar(20) NOT NULL default '',
  `edat` bigint(20) unsigned NOT NULL default '0',
  PRIMARY KEY (`id`)
) TYPE=MyISAM;

INSERT INTO counter (id, ip) VALUES (1,0);
```