

Spamschutz bei TYPO3

von *Bernd Warken* bei Fa. *Netcos AG*

Kapitel 0: Einleitung	3
<i>0.1 Vorwort</i>	3
<i>0.2 Lizenz</i>	3
Kapitel 1: Aktivierung und Konfiguration	4
<i>1.1 config.spamProtectEmailAddresses</i>	4
<i>1.2 config.spamProtectEmailAddresses_atSubst</i>	4
<i>1.3 config.spamProtectEmailAddresses_lastDotSubst</i>	4
Kapitel 2: Anwendung	6
<i>2.1 Inhalt</i>	6
<i>2.2 TypoScript-Template</i>	6
Kapitel 3: Fortgeschrittene Methoden	8
<i>3.1 Einfügen von Grafik</i>	8
<i>3.2 JavaScript für Buchstaben</i>	8
<i>3.3 Komplettes Beispiel</i>	9

Kapitel 0: Einleitung

0.1 Vorwort

TYPO3 hat einen Spamschutz. Damit werden auftretende Emailadressen im HTML-Quellcode unkenntlich verändert oder auch ein *JavaScript*-Programm zur Bearbeitung aufgerufen. Der Spamschutz wird durch zwei *TypoScript*-Befehle aktiviert und konfiguriert. Emailadressen können im Inhalt und in *TypoScript* verwendet werden. Dabei sind jedoch einige Dinge zu beachten.

0.2 Lizenz

Dieses Dokument ist gespeichert in der Datei **Spamschutz_de.sxw**. Die aktuelle Version ist **1.1** vom 27.02.2008. Autor ist **Bernd Warken** bei Firma [netcos AG](http://www.netcos.de).

Das Dokument steht unter der Lizenz **GPL (GNU General Public License) Version 3**. Der Originaltext dieser Lizenz ist erhältlich unter <http://www.gnu.org/licenses/gpl.html>, eine deutsche Übersetzung befindet sich unter <http://www.gnu.de/documents/gpl.de.html>.

Copyright © 2007, 2008 Bernd Warken.

Kapitel 1: Aktivierung und Konfiguration

Aktivierung und Konfiguration des Spamschutzes ist sehr leicht, sie erfolgen durch zwei *TypoScript*-Befehle.

1.1 *config.spamProtectEmailAddresses*

Die Aktivierung erfolgt durch den *TypoScript*-Befehl **config.spamProtectEmailAddresses**. Dieser Befehl wird zum TYPO3-Template hinzugefügt. Als Wert kann entweder eine Zahl von **-5** bis **5** oder das Kennwort **ascii** eingesetzt werden.

```
config.spamProtectEmailAddresses = Zahl
```

Falls eine *Zahl* verwendet wird, so darf nur **-5** bis **5** benutzt werden. Dadurch werden im HTML-Quelltext die ASCII-Code der Buchstaben der Emailadresse um diesen Bereich verschoben. Zum Beispiel hat der Buchstabe **z** den ASCII-Wert **#122**. Wurde als *Zahl* der Wert **1** eingegeben, so verschiebt sich dieser Buchstabe auf den ASCII-Wert **#123**, also auf den Buchstaben **{**, die geschweifte Klammer links.

Diese Verschiebung ist unkritisch für Zahlenwerte **-5** bis **1**. Wird jedoch eine *Zahl* von **2** bis **5** gewählt, so kann als Ergebnis der ASCII-Wert **#124** entstehen, der Buchstabe **|**. Dieser Vertikalstrich ist jedoch ein spezielles Zeichen für TYPO3 und sollte nicht benutzt werden. Daher ist es sinnvoll, sich auf die unkritischen Zahlenwerte zu beschränken.

Bei Verwendung einer *Zahl* wird zusätzlich zur Verschiebung der Buchstaben ein *JavaScript*-Programm bei der Auswahl der Emailadresse ausgeführt. Der Link der Emailadresse zeigt auf dieses Skript, so dass eine Auswertung durch Spambots fast unmöglich ist.

```
config.spamProtectEmailAddresses = ascii
```

Wird der Wert auf **ascii** gesetzt, so werden alle Buchstaben der Emailadresse im HTML-Quelltext einfach durch ihren jeweiligen ASCII-Code ersetzt. Die Ausführung des *JavaScript*-Programms bei der Auswahl der Emailadresse entfällt. Es ist Ihnen überlassen, ob Sie diesen Wert nehmen oder lieber eine Zahl verwenden.

1.2 *config.spamProtectEmailAddresses_atSubst*

Zusätzlich kann das Zeichen **@** innerhalb der Emailadresse noch durch einen anderen Text ersetzt werden. Eine gute Wahl für den alternativen Text ist **(at)**. Falls im Template der Befehl

```
config.spamProtectEmailAddresses_atSubst = (at)
```

gesetzt wird, so wird z.B. die Emailadresse **test@test.de** ersetzt durch **test(at)test.de**. Diese Ersetzung erfolgt automatisch. Man schreibt die Emailadresse mit dem **@**-Symbol auf, ohne sich direkt um die Veränderung zu kümmern. TYPO3 ergreift die Aufgabe, die Ersetzung durchzuführen.

1.3 *config.spamProtectEmailAddresses_lastDotSubst*

Zusätzlich kann das letzte Zeichen **.** (Punkt) innerhalb der Emailadresse noch durch einen anderen Text ersetzt werden. Eine Wahl für den alternativen Text ist z.B. **(dot)**. Falls im Template der Befehl

```
config.spamProtectEmailAddresses_lastDotSubst = (dot)
```

gesetzt wird, so wird z.B. die Emailadresse test@test.de ersetzt durch **test(at)test(dot)de**. Diese Ersetzung erfolgt automatisch. Man schreibt die Emailadresse mit dem @-Symbol und dem Punkt auf, ohne sich direkt um die Veränderung zu kümmern. TYPO3 ergreift die Aufgabe, die Ersetzung durchzuführen.

Der eigentliche Vorteil dieses Feldes ergibt sich erst bei der Handhabung mit JavaScript, wie in einem späteren Kapitel erklärt wird.

Dies vervollständigt die Konfiguration des Spamschutzes.

Kapitel 2: Anwendung

Zur Anwendung des Spamschutzes sind einige Dinge zu beachten. Die *TypoScript-Referenz* erklärt, dass der Spamschutz für Emailadressen in *Typolink* angewendet wird. Im normalen Inhalt geschieht dies durch das Einfügen von Links und im *TypoScript*-Template muss dies durch entsprechende Befehle aktiviert werden.

2.1 Inhalt

Der Textinhalt einer normalen Seite wird in TYPO3 mit dem *RTE-Editor* erstellt. Wenn man innerhalb des Textes einfach eine Emailadresse eingibt, so wird diese nicht als solche eingestuft. Man muss vielmehr die Emailadresse als Link hinzufügen, also den Text der Emailadresse markieren und den Button *Link einfügen* drücken.

Dadurch entsteht ein neues Fenster. Darin wählen wir den Bereich *Email*. Im Feld *Email Adresse* fügen wir die richtige Emailadresse ohne Änderung ein, unter Verwendung des @-Symbols. Wir klicken dann den Button *Link setzen*. Dadurch wird der Spamschutz für diese Emailadresse aktiviert.

2.2 TypoScript-Template

Wenn eine Emailadresse auf jeder Seite wiederholt wird, z.B. im Footer, muss dies im Template dargestellt werden. Wir gehen davon aus, dass eine *TemplaVoilà*-Datenstruktur mit **lib.footer** erzeugt wurde, beim klassischen TYPO3 wird statt dessen ein Marker verwendet.

Wir stellen **lib.footer** als **COA** aus, so dass wir den Code in mehrere Teile zerlegen können. Wir ordnen alle Elemente vor der Emailadresse einem Zahlenwert zu, die Emailadresse einem weiteren und alle Teile danach einem dritten Zahlenwert zu.

In dem Bereich mit der Emailadresse fügen wir den *Typolink*-Befehl **typolink.parameter** mit der richtigen Emailadresse als Wert hinzu.

Ein Textbeispiel ergibt sich durch:

```
lib.footer = COA
lib.footer {
  10 = HTML
  10.value (
    <p>Text vor Email
  )
  20 = TEXT
  20.value = email@testfirma.de
  20.typolink.parameter = email@testfirma.de
  30 = HTML
  30.value (
    Text nach Email</p>
  )
}
```

Ein Beispiel mit einer Tabelle würde dann folgendermaßen aussehen:

```
lib.footer = COA
lib.footer {
  10 = HTML
  10.value (
    <table>
    <tr>
    <td>
      <a href="http://www.testfirma.de">Testfirma AG</a>
    </td>
    <td>E-Mail:
  )
  20 = TEXT
  20.value = email@testfirma.de
  20.typolink.parameter = email@testfirma.de
  30 = HTML
  30.value (
    </td>
    </tr>
    </table>
  )
}
```

Bitte beachten Sie, dass die Werte für HTML runde Klammern verwenden.

Kapitel 3: Fortgeschrittene Methoden

Die Spam-Robots können erheblich ausgetrickst werden, wenn unsichtbare Grafiken eingefügt werden und einige Buchstaben durch JavaScript-Funktionen erzeugt werden. Dadurch wird die Emailadresse im HTML-Seitenquelltext stark verfälscht, ohne dass sich das Aussehen ändert.

3.1 Einfügen von Grafik

Wenn man innerhalb des Textes eine Grafik einfügt, wird das Copy & Paste unterdrückt. Wir verwenden als Grafik die Bilddatei **clear.gif**, die Teil von aktuellen TYPO3-Versionen ist. Sie befindet sich im Hauptverzeichnis des TYPO3-Codes. Dieses Bild ist nur 1 Pixel hoch und breit und hat eine transparente Farbe, so dass es unsichtbar ist und wenig Platz wegnimmt.

Innerhalb des Textes muss dann der folgende Code eingefügt werden:

```

```

Der **alt**-Wert muss gesetzt werden, damit hier nicht automatisch ein Wert erzeugt wird, der für Spam-Robots ausgewertet werden kann.

Am besten definiert man diesen Code als eine TypoScript-Konstante, da der Code viel Platz wegnimmt. Als Konstantennamen wähle ich hier **img_invis**.

```
img_invis = 
```

Diese Konstante lässt sich durch **{\$img_invis}** leicht und kurz in TypoScript-Text einbetten. Mögliche Einbettungswerte sind **config.spamProtectEmailAddresses_atSubst** und **config.spamProtectEmailAddresses_lastDotSubst**. In Sektion 3.3 zeigen wir ein komplettes Beispiel.

3.2 JavaScript für Buchstaben

Emailadressen können von Spam-Robots durch das @-Symbol und einen Punkt innerhalb eines Wortes identifiziert werden. TypoScript gestattet es, diese beiden Symbole durch etwas anderes zu ersetzen.

Wir entscheiden uns erst einmal das @-Symbol durch den Text (**at**) zu ersetzen. Als nächstes lassen wir alle Buchstaben einzeln durch JavaScript erzeugen, so dass keiner der Buchstaben im Quelltext direkt zu sehen ist; nur der Aufruf einer JavaScript-Funktion wird hier dargestellt. Dies dürfte es für Spam-Robots erheblich schwerer machen, die Emailadresse ausfindig zu machen.

Die Definition der JavaScript-Funktionen erfolgt im Header der HTML-Datei. Dies kann in TYPO3 programmiert werden durch Aufruf von **page.headerData** im Template-Setup. Diese Größe benötigt noch eine Zahl. Diese kann beliebig gewählt werden, sollte aber nicht bei anderen Werten von **page.headerData** vorkommen. Wir wählen dafür **250**.

Als Beispiel stellen wir eine einzige JavaScript-Funktion **letter** zur Erzeugung eines Buchstabens dar.

```
page.headerData.250 = TEXT
page.headerData.250.value (
<script type="text/javascript">
<!--
function letter() {
    document.write('a');
}
// -->
</script>
)
```

Diese JavaScript-Funktion lässt sich innerhalb eines TypoScript-Textes verwenden durch den Code

```
<script type="text/javascript"> letter() </script>
```

In Sektion **3.3** geben wir ein komplettes Beispiel mit mehreren **letter**-Funktionen.

3.3 Komplettes Beispiel

Als Beispiel geben wir die Zerlegung von **@** in **(at)**, die Darstellung von diesem Wert und dem Punkt innerhalb der Emailadresse durch JavaScript und die Umkreisung aller dieser Buchstaben mit einer unsichtbaren Grafik.

Zunächst erstellen wir die Definition der JavaScript-Funktionen zum Erzeugen einzelner Buchstaben. Für jeden zu erzeugenden Buchstaben geben wir eine eigene Funktion an. Die Namen dieser Funktionen sollten Sie aber nicht übernehmen, damit sich Spam-Robots nicht darauf einschließen können.

Diese Funktionen werden im Template-Setup definiert.

```

page.headerData.250 = TEXT
page.headerData.250.value (
<script type="text/javascript">
<!--
function letter_at1() {
    document.write('(');
}
function letter_at2() {
    document.write('a');
}
function letter_at3() {
    document.write('t');
}
function letter_at4() {
    document.write(')');
}
function letter_dot() {
    document.write('.');
}
// -->
</script>
)

```

Als nächstes definieren wir für langen Code Konstanten in Template-Constants:

```

img_invis = 
letter_at1 = <script type="text/javascript"> letter_at1() </script>
letter_at2 = <script type="text/javascript"> letter_at2() </script>
letter_at3 = <script type="text/javascript"> letter_at3() </script>
letter_at4 = <script type="text/javascript"> letter_at4() </script>
letter_dot = <script type="text/javascript"> letter_dot() </script>

```

Nun verwenden wir diese Konstanten, um die TypoScript-Antispam-Variablen im Template-Setup zu definieren:

```

config.spamProtectEmailAddresses = 1
config.spamProtectEmailAddresses_atSubst = {$img_invis} {$letter_at1} {$img_invis}
{$letter_at2} {$img_invis} {$letter_at3} {$img_invis} {$letter_at4} {$img_invis}
config.spamProtectEmailAddresses_lastDotSubst = {$img_invis} {$letter_dot} {$img_invis}

```

Entfernen Sie den Zeilenumbruch bei **_atSubst**.

Zum Testen erstellen Sie auf einer Seite eine Emailadresse. Schauen Sie sich den HTML-Quelltext dieser Seite an. Dann werden Sie erkennen, dass die Emailadresse sehr schwer erkannt werden kann.