Tabellenloses Kontaktformular mit Spamschutz

Inhalt

Abbildungen I

Listings I

1 Problemstellung 1

2 Tabellenloses Kontaktformular erstellen 1

2.1 Darstellung 1

2.2 Formularfelder 3

2.3 Struktur 3

2.4 Formatierung 3

2.5 Validierung 3

3 Inhalt des tabellenlosen Kontaktformulars versenden 4

3.1 Spamschutz 4

3.2 Zusammenstellung der Mail-Parameter 4

3.3 Erfolg oder Misserfolg anzeigen. 4

4 E-Mail-Eingang beim Empfänger 5

5 Listing des gesamten Quellcodes 6

Literatur II

Abbildungen

Abbildung 1: Mit Platzhaltern und Fehlermeldungen gefülltes Kontaktformular nach dem Absenden 2

Abbildung 2: Mit Inhalten gefülltes Kontaktformular vor dem Absenden 2

Abbildung 3: Erfolgsmeldung beim E-Mail-Versand 4

Abbildung 4: Fehlermeldung beim E-Mail-Versand 4

Abbildung 5: E-Mail-Eingang beim Empfänger 5

Listings

Listing 1: Gesamter Quellcode des tabellenlosen Kontaktformulars mit Spamschutz 12

1 Problemstellung

In diesem Beitrag wird beschrieben,

- wie ein *tabellenloses* Kontaktformular mit Hilfe der Gestaltungs- und Formatierungssprache *CSS* erstellt werden kann,
- wie alle erfassten Formularwerte nach dem Absenden des Formulars erhalten bleiben[1],
- wie das Formular mit einfachen Mitteln gegen Schadsoftware (engl. *spambots, spider, crawler*, etc.) geschützt werden kann.

Zunächst wird das betreffende Kontaktformular präsentiert. Danach werden einfache Schutzmaßnahmen beschrieben, wie es vor Eingabefehlern und/oder Missbrauch gesichert werden kann. Dabei kommen u. a. das sog. "Honigtopf" (engl. *honeypot*)-Verfahren und das Messen der Eingabegeschwindigkeit zum Einsatz.

Zum Verständnis werden Grundkenntnisse der Programmiersprachen CSS, HTML und PHP vorausgesetzt.[2]

2 Tabellenloses Kontaktformular erstellen

2.1 Darstellung

Als Beispiel dient ein tabellenloses Kontaktformular mit 6 sichtbaren und 2 unsichtbaren Formularfeldern. Wird ein nur mit Platzhaltern belegtes Kontaktformular abgesandt,

- bleiben die Platzhalter erhalten,
- aber Fehlermeldungen werden neben allen Formularfeldern ausgegeben (s. Abbildung 1).

Das Verhalten des Kontaktformulars ändert sich nicht, wenn es durch konkrete Eingaben eines Benutzers ausgefüllt und abgesandt wird:

- alle erfassten Formularwerte bleiben nach dem Absenden erhalten,
- neben falsch ausgefüllten Formularfeldern werden entsprechende Fehlermeldungen angezeigt.

Der Vorteil ist, dass nur die fehlerhaften Eingabewerte vom Benutzer korrigiert werden müssen, aber nicht alle übrigen Eingaben.

[1] Ein solches Formular (engl. *sticky form*) ist einfach ein Standard-HTML-Formular, das aber die erfassten Formularwerte bewahrt und erneut anzeigt, nachdem es abgesandt wurde. Vgl. dazu [1] und [3].

[2] **CSS** steht für ***C**ascading **S**tyle **S**heets* und ist eine Gestaltungs- und Formatierungssprache. **HTML** steht für ***H**yper**T**ext **M**arkup **L**anguage* und ist eine Auszeichnungssprache für Webseiten. **PHP** steht für ***P**ersonal **H**ome **P**age* und ist eine serverseitige Skriptsprache für Webanwendungen.

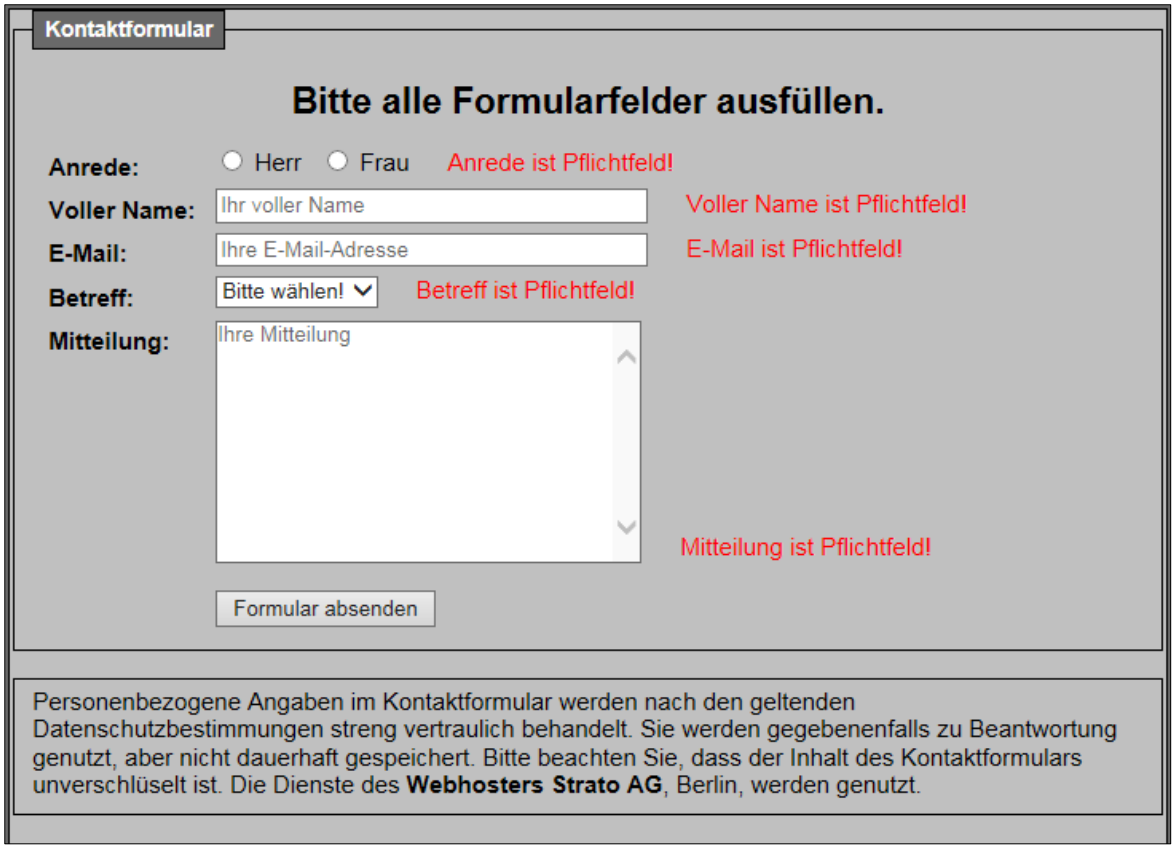


Abbildung 1: Mit Platzhaltern und Fehlermeldungen gefülltes Kontaktformular nach dem Absenden

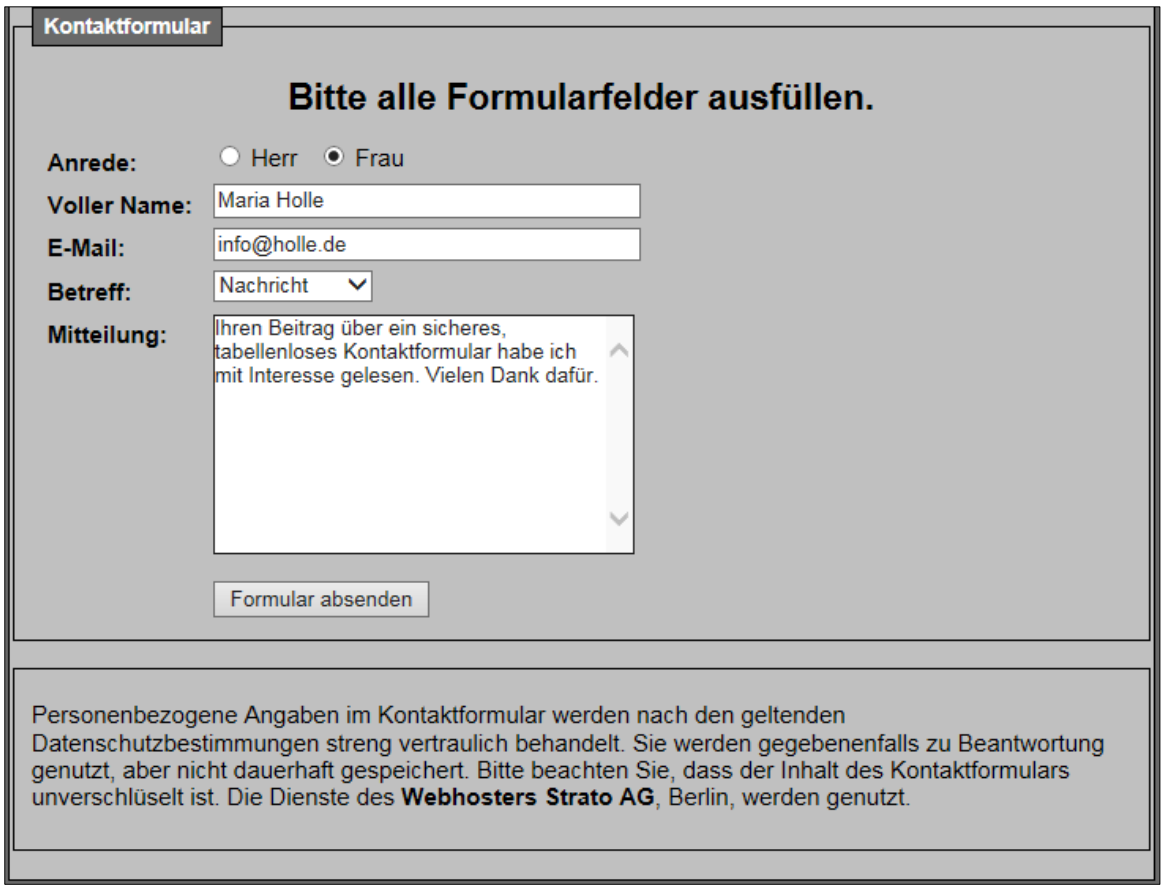


Abbildung 2: Mit Inhalten gefülltes Kontaktformular vor dem Absenden

2.2 Formularfelder

Das Kontaktformular umfasst 2 unsichtbare und 6 sichtbare Formularfelder. Tabelle 1 enthält die Einzelheiten dieser 8 Formularfelder (s. Tabelle 1):

Bezeichnung	HTML Name-Attribut	Formularfeld	HTML-Feldtyp	Sichtbarkeit
keine	url	einzeiliges Textfeld	text	nein
keine	frm_load_tm	einzeiliges Textfeld	hidden	nein
Anrede	salutation	Knöpfe	radio	ja
Name	fullname	einzeiliges Textfeld	text	ja
E-Mail	email	einzeiliges Textfeld	email	ja
Betreff	subject	Auswahlliste	select	ja
Nachricht	message	mehrzeiliges Textfeld	textarea	ja
absenden	submit	Schaltfläche	button	ja

Tabelle 1: Eigenschaft der 9 Felder des tabellenlosen Kontaktformulars

Die beiden unsichtbaren Formularfelder (s. Tabelle 1) werden zum Schutz des Kontaktformulars vor Spam benutzt:

- Das Formularfeld mit dem Namen **url** dient zum Anlegen einer sog. Honigtopf-Falle. Die Unsichtbarkeit wird durch die Zuweisung der CSS-Klasse „antispam“ (d.h. display:none) erreicht.
- Das Formularfeld mit dem Namen **frm-load_tm** besitzt ein ***hidden***-Attribut. Es verhindert die Anzeige des Formularfelds. Der zugehörige Feldwert wird durch das ***value***-Attribut festgelegt. Es enthält hier den Ladezeitpunkt ($_SERVER['REQUEST_TIME']) des Kontaktformulars.
- Für das Formularfeld mit dem Namen **subject** wird statt eines einzeiligen Textfelds eine *Auswahlliste* definiert. Dadurch erhöht sich allerdings die Komplexität des Formulars. Vgl. dazu [1].

2.3 Struktur

Die Strukturierung des tabellenlosen Kontaktformulars erfolgt mit der Auszeichnungssprache **html** (s. Listing 1, Zeile 1 ff).

2.4 Formatierung

Die Formatierung des Kontaktformulars erfolgt mit der Gestaltungs- und Formatierungssprache **CSS.** (s. Zeilen 6 - 57). Die Anweisungen in den Zeilen 46 (label{*float:left;*}) und 53 (*br{clear:left;}*) bewirken den tabellenlosen Aufbau des Kontaktformulars. Mehr dazu in [2].

Das tabellenlose Kontaktformular wird im sog. body zwischen den HTML-Tags <form> … </form> (s. Listing 1, Zeilen 232 - 298) definiert. Das <form>-Element selbst besitzt das Attribut *action*. Der zugehörige Formularcode enthält die super globale Variable $ _SERVER ["PHP_SELF"], die den Dateinamen des aktuell ausgeführten PHP-Skripts zurückgibt. Die missbräuchliche Ausnutzung dieser Variablen durch Hacker kann mit der PHP-Funktion *htmlspecialchars()* vermieden werden. Sie wandelt Sonderzeichen in HTML-Code um (s. Listing 1, Zeile 233).

2.5 Validierung

Die Validierung der sichtbaren Felder des tabellenlosen Kontaktformulars erfolgt nach dessen Absendung mit dem zugehörigen PHP-Code (s. Zeilen 159 – 231). Von dort aus werden verschiedene PHP-Funktionen ausgerufen (s. Zeilen 61 - 146), die entsprechende Tests durchführen.

3 Inhalt des tabellenlosen Kontaktformulars versenden

3.1 Spamschutz

Nachdem die Validierung der sichtbaren Formularfelder erfolgreich durchgeführt wurde (s. Listing 1, Zeile 200), erfolgen vor dem Versenden ihres Inhalts per E-Mail bezüglich Spamschutz noch zwei weitere Bereinigungen mit den PHP-Funktionen **sanitize** (s. Zeilen 116 – 126) bzw**. heal**[3] (s. Zeilen 128 – 141). Aufgerufen werden diese beiden Funktionen durch den PHP-Code in den Zeilen 302 - 306 bzw. 319 - 321.

3.2 Zusammenstellung der Mail-Parameter

Zum Versenden der Formulardaten wird die PHP eingebaute Funktion ***mail*** verwendet. Dieser Funktion werden folgende 4 Parameter übergeben (s. Zeile 356):

$mailTo	Empfänger der E-Mail	(s. Zeile 330)
$mailSubject	Betreff der E-Mail	(s. Zeilen 323)
$mailBody	Die zu sendende Nachricht	(s. Zeilen 334 – 344)
$mailHeader	Zusätzliche Versandangaben	(s. Zeilen 346 – 353)

3.3 Erfolg oder Misserfolg anzeigen.

Die in PHP eingebaute **mail**-Funktion gibt *true* zurück, wenn die E-Mail erfolgreich für den Versand akzeptiert wurde, sonst *false*. Dementsprechend wird eine Erfolgsmeldung (s. Abbildung 3) oder eine Fehlermeldung (s. Abbildung 4) ausgegeben (s. Listing 1 , Zeile 358 ff).

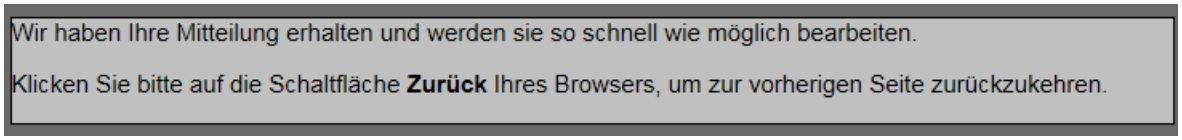


Abbildung 3: Erfolgsmeldung beim E-Mail-Versand

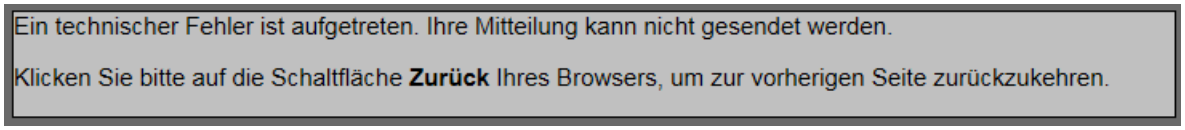
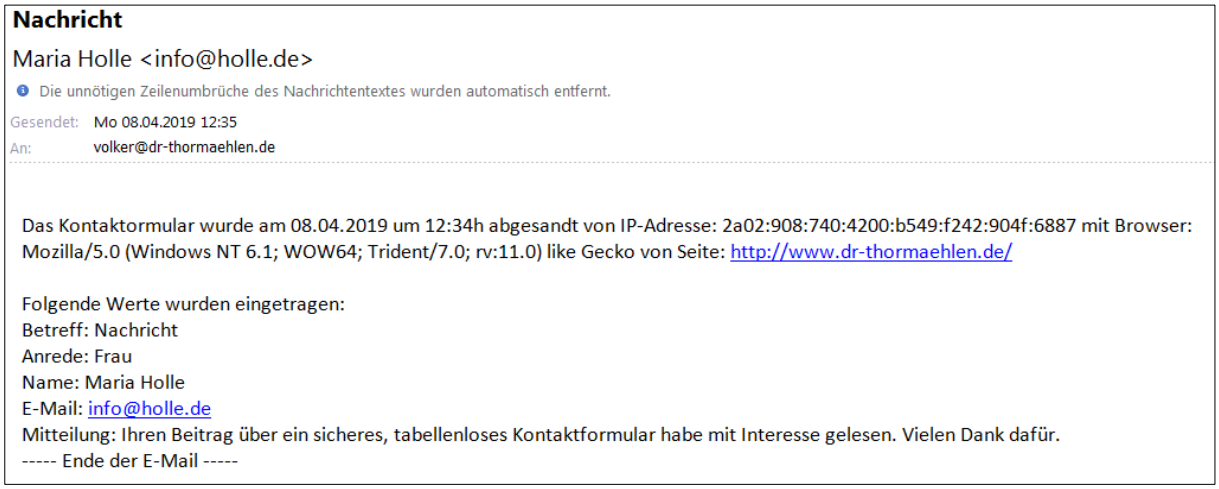


Abbildung 4: Fehlermeldung beim E-Mail-Versand

[3] Quelle: [4]. Wie bereits erwähnt, können Spam-Angriffe auf E-Mail-Header verhindert werden, indem die vordefinierten Steuerzeichen für neue Zeile (/n) und/oder Wagenrücklauf (/r) aus den Formulardaten entfernt werden.

4 E-Mail-Eingang beim Empfänger

Wird der Inhalt des Kontaktformular (s. Abbildung 2) erfolgreich versandt, erhält der Empfänger der entsprechenden E-Mail folgende Nachricht (s. Abbildung 5):



Nachricht

Maria Holle <info@holle.de>

Die unnötigen Zeilenumbrüche des Nachrichtentextes wurden automatisch entfernt.

Gesendet: Mo 08.04.2019 12:35
An: volker@dr-thormaehlen.de

Das Kontaktormular wurde am 08.04.2019 um 12:34h abgesandt von IP-Adresse: 2a02:908:740:4200:b549:f242:904f:6887 mit Browser: Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko von Seite: http://www.dr-thormaehlen.de/

Folgende Werte wurden eingetragen:
Betreff: Nachricht
Anrede: Frau
Name: Maria Holle
E-Mail: info@holle.de
Mitteilung: Ihren Beitrag über ein sicheres, tabellenloses Kontaktformular habe mit Interesse gelesen. Vielen Dank dafür.
----- Ende der E-Mail -----

Abbildung 5: E-Mail-Eingang beim Empfänger

5 Listing des gesamten Quellcodes

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Kontaktformular ohne Tabelle mit Spamschutz</title>
   <style type="text/css">
      html {
         height:100.5%;
         font-size:62.5%;
         line-height:1.5em;
      }
      body {
         font-family:arial,verdana,sans-serif;
         font-size:1.3em;
         background-color:dimgrey;
      }
      form {
         width:725px;
         margin:0 auto;
         border:1px solid black;
         background-color:silver;
      }
      fieldset {
         width:auto;
         border:1px solid black;
      }
      legend {
         padding:0.2em 0.5em;
         border:1px solid black;
         color:white;
         background-color:dimgrey;
         font-size:90%;
         text-align:right;
      }
      textarea {
         font-family:arial,verdana,sans-serif;
         padding:0px;
         margin:0px;
         text-align:left;
         white-space:normal;
         word-wrap:normal;
         width:265px;
         min-height:150px;
      }
      label {
         float:left;
         width:100px;
         padding-left:5px;
         margin:5px;
         text-align:left;
         font-weight:bold;
      }
      br{clear:left;}
      .hvr:hover{background-color:yellow;}
      .error{color:red;padding-left:20px;}
```

```
      .antispam{display:none;}
   </style>
</head>
<body>
   <?php
   function test_input($data) {
      $data = trim($data);             // Überflüssige Zeichen am Anfang und Ende entfernen.
      $data = stripslashes($data);     // Maskierungszeichen entfernen.
      $data = htmlspecialchars($data); // Sonderzeichen in HTML-Code umwandeln.
      return $data;
   }

   function removeMultipleSpaces($Nm) {   // doppelte Leerzeichen entfernen.
      return preg_replace("/\s{2,}/", ' ', trim($Nm));
   }

   function validName($pattern,$Nm) {  // Name auf Gültigkeit prüfen.
      $Nm = removeMultipleSpaces($Nm); // doppelte Leerzeichen entfernen.
      if (strLen($Nm) < 2) {
         return "Name ist zu kurz, min. 2 Zeichen.";
      } else {
         if (!preg_match($pattern, $Nm)) { // Name mit Muster vergleichen.
            return "Numerische Zeichen im Namen gefunden!";
         }
      }
      return "";
   }

   function validSurname($nachName) {
      // Initiale des Nachnamens prüfen.
      // Wenn nur 1 Wort im Nachnamen, dann ...
      if (str_word_count($nachName, 0, 'äüöÄÜÖß') == 1) {
         $firstChar = htmlentities($nachName[0]);
         // Wenn erster Buchstabe in Kleinschreibung, dann ...
         if($firstChar == strtolower($firstChar)) {
            return "Initiale in Kleinschreibung!";
         }
      }
      return "";
   }

   function isValidEmail($email) {
      // E-Mail-Format auf Gültigkeit überprüfen.
      return filter_var($email, FILTER_VALIDATE_EMAIL) && preg_match('/@.+\./', $email);
   }

   function validate($str) {
      // Sonderzeichen in HTML-Code umwandeln.
      return trim(htmlspecialchars($str));
   }

   function parse_fullname($Nm) {
      // Zahl der Leerzeichen in der übergebenen Zeichenkette prüfen.
      $name_parts = explode(' ', $Nm);
      if (sizeof($name_parts) < 2) {
         return "Nur 1 Namensteil gefunden.";
      }
      return "";
   }

```

```
   function sanitize($data){
      // Entfernt HTML- und PHP-Tags und wandelt Sonderzeichen in HTML-Code um.
      $data = trim(strip_tags(htmlspecialchars($data)));
      // Entfernt Maskierungszeichen z. B.  Backslashes vor Anführungszeichen.
      $data = stripslashes($data);
      // E-Mail Einfügungen (d. h. Angriffsmuster) entfernen.
      $data = preg_replace("/(href|content-transfer-encoding:|content-type:|mime-
version:|multipart-mixed:|" .
         "Subject:|to:|cc:|bcc:|from:|reply-to:)/ims", "", $data);
      return $data;
   }

   function heal($data) {
      // Definierte Steuerzeichen im Formulartext durch Leerzeichen ersetzen.
      $injections = array(
         '/(\n+)/i',
         '/(\r+)/i',
         '/(\t+)/i',
         '/(%0A+)/i',
         '/(%0D+)/i',
         '/(%08+)/i',
         '/(%09+)/i'
      );
      $data= preg_replace($injections,' ',$data);
      return trim($data);
   }

   function SpamBotFilling($loadTm,$requestTm,$fillTm) {
      // Messung der Zeit zum Ausfüllen des Formlars
      return (($requestTm - $loadTm) < $fillTm) ? true : false;
   }

   // Variablen definieren und Anfangswerte setzen.
   $salutation    = $fullname    = $email    = $subject    = $message = "";
   $salutationErr = $fullnameErr = $emailErr = $subjectErr = $messageErr = "";
   // Annahme: Die Formulardaten sind fehlerfrei.
   $flawless = true;
   // Assoziativen Array (d. h. Array mit benannten Schlüsseln) für den Betreff definieren.
   $subjects = array('1'=>'Anfrage','2'=>'Kommentar','3'=>'Nachricht','4'=>'Sonstiges');

   // Fehlermeldungen komplett abschalten.
   error_reporting(0);

   // Formulardaten prüfen.
   if('POST' == $_SERVER['REQUEST_METHOD']) {
      if (!isset($_SERVER['HTTP_REFERER']) || (parse_url($_SERVER['HTTP_REFERER'],
PHP_URL_HOST) != $_SERVER['SERVER_NAME'])) {
         die('Direkter Aufruf eines PHP-Skripts ist nicht erlaubt');
      }
      // Zeitmessung.
      if (isset($_POST['frm_load_tm']) && !empty($_POST['frm_load_tm'])) {
         $loadTm = intVal(sanitize($_POST["frm_load_tm"]));
         $currTm = intVal($_SERVER['REQUEST_TIME']);
         if (SpamBotFilling($loadTm,$currTm,2)) {
            die('Achtung: Das Kontaktformur wurde vermutlich von einem sog. Spambot belegt.');
         }
      }

      // Nur Fomulardaten ohne Spam versenden.
      if (isset($_POST['url']) && !empty($_POST['url'])) {
```

```
        die('Achtung: Das Kontaktformur wurde vermutlich von einem sog. Spambot belegt.');
    }

    if(empty($_POST["salutation"])) {
        $salutationErr ="Anrede ist Pflichtfeld!";
        $flawless = false;
    } else {
        $salutation= test_input($_POST["salutation"]);
    }

    $selectedVal = $_POST['subject'];
    if ($selectedVal == "0") {
        $subjectErr = "Betreff ist Pflichtfeld!";
        $flawless = false;
    }

    if (empty($_POST["fullname"])) {
        $fullnameErr = "Voller Name ist Pflichtfeld!";
    } else {
        $fullname = test_input($_POST["fullname"]);
        // Muster akzeptiert Umlaute, aber keine Ziffern.
        $muster = "/^[\D]+$/";
        $fullnameErr = validName($muster,$fullname);
        if (strlen($fullnameErr) > 0) {
            $flawless = false;
        } else {
            $fullnameErr = parse_fullname($fullname);
            if (strlen($fullnameErr) > 0) {
                $flawless = false;
            }
        }
    }

    if (empty($_POST["email"])) {
        $emailErr = "E-Mail ist Pflichtfeld!";
        $flawless = false;
    } else {
        $email = test_input($_POST["email"]);
        if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
            $emailErr = "Ihre E-Mail ist ungültig!";
            $flawless = false;
        }
    }

    if(empty(trim($_POST["message"]))){
        $messageErr = "Mitteilung ist Pflichtfeld!";
        $flawless = false;
    } else {
        $message = test_input($_POST["message"]);
        if (strlen($message) < 10) {
            $messageErr = "Ihre Mitteilung ist zu kurz, min. 10 Zeichen!";
            $flawless = false;
        }
    }
}
?>
<form name="myForm" method="post" accept-charset="UTF-8"
    action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
    <fieldset>
        <legend><b>Kontaktformular</b></legend>
```

```
         <center>
            <h2>Bitte alle Formularfelder ausfüllen.</h2>
         </center>

         <label for="url" class="antispam" >Nichts eingeben!:</label>
         <input class="antispam" type="text" name="url" size="30" >

         <input type="hidden" name="frm_load_tm" autocomplete="off"
            value="<?php echo $_SERVER['REQUEST_TIME'];?>" />

         <label for="salutation">Anrede:</label>
         <input type="radio" class="hvr" name="salutation" value="m"
         <?php
            printf("%s", ($salutation=='m' ? " checked" : ""));
            ?> > Herr  
         <input type="radio" class="hvr" name="salutation" value="w"
         <?php
            printf("%s", ($salutation=='w' ? " checked" : ""));
            ?> > Frau
         <span class="error"><?php echo $salutationErr;?></span><br>

         <label for="fullname">Voller Name:</label>
         <input type="text" class="hvr" name="fullname" size="40" placeholder="Ihr voller
Name"
            value="<?php printf("%s", $fullname);?>">
         <span class="error"><?php echo $fullnameErr;?></span><br>

         <label for="email">E-Mail:</label>
         <input type="email" class="hvr" name="email"   size="40" placeholder="Ihre E-Mail-
Adresse"
            value="<?php printf("%s", $email);?>">
         <span class="error"><?php echo $emailErr;?></span><br>

         <label for="subject">Betreff:</label>
         <select class="hvr" name="subject">
            <option value="0">Bitte wählen!</option>
            <?php
               foreach($subjects as $id=>$subject){
                  $sel = ($id == $selectedVal) ? 'selected' : '';
                  echo "<option $sel value=\"$id\">$subject</option>";

               }
            ?>
         </select>
         <span class="error"><?php echo $subjectErr;?></span><br>

         <label for="message">Mitteilung:</label>
         <textarea class="hvr" name="message" placeholder="Ihre Mitteilung"><?php echo
validate($_POST['message']);?></textarea>
         <span class="error"><?php echo $messageErr;?></span><br><br>

         <label> </label>
         <input name="submit" class="hvr" type="submit" value="Formular absenden">

      </fieldset><br>
      <fieldset>
         <p>Personenbezogene Angaben im Kontaktformular werden nach den geltenden
Datenschutzbestimmungen streng vertraulich behandelt.
         Sie werden gegebenenfalls zu Beantwortung genutzt, aber nicht dauerhaft gespeichert.
         Bitte beachten Sie, dass der Inhalt des Kontaktformulars unverschlüselt ist.
```

```
         Die Dienste des <b>Webhosters Strato AG</b>, Berlin, werden genutzt.</p>
      </fieldset><br>
   </form><br>
   <?php
   if(isset($_POST['submit']) && $flawless) {
      // Bereinigungen vornehmen.
      $salutation    = sanitize($_POST['salutation']);
      $subject       = sanitize($_POST['subject']);
      $fullName      = sanitize($_POST['fullname']);
      $mailFrom      = sanitize($_POST['email']);
      $message       = sanitize($_POST['message']);
      // Die Funktion nl2br konvertiert alle Zeilenumbrüche (/n) in HTML-Zeilenumbrüche.
      $message       = nl2br($message);
      $message       = wordwrap($message,70);   // max. 70 Zeichen je Zeile.

      // Anrede bestimmen.
      $salutation = heal($salutation);
      if ($salutation == "m") {
         $salutation = "Herr";
      } else {
         $salutation = "Frau";
      }
      // Definierte Steuerzeichen durch Leerzeichen ersetzen.
      $fullName   = heal($fullname);
      $mailFrom   = heal($email);
      $message    = heal($message);
      // Betreff bestimmen.
      $mailSubject = $subjects{$selectedVal};

      $from_ip       = $_SERVER['REMOTE_ADDR'];    // IP-Adresse
      // Gibt die URL einer Webseite zurück, von der aus die aktuelle Webseite aufgerufen
wurde.
      $referer       = isset($_SERVER['HTTP_REFERER']) ? $_SERVER['HTTP_REFERER'] : '';
      $from_browser  = $_SERVER['HTTP_USER_AGENT']; // Browser
      $mailTo        = "volker@dr-thormaehlen.de";  // Empfänger der E-Mail
      $rn            = "\r\n";                      // r = carrige return, n = line feed

      // Mitteilung zusammenstellen.
      $mailBody = $rn . "Das Kontaktormular wurde am ". date("d.m.Y")." um ".date("H:i")."h
abgesandt\n";
      $mailBody.= "von IP-Adresse: ".$from_ip." mit Browser: ".$from_browser." von Seite:
".$referer.$rn.$rn;
      $mailBody.= "Folgende Werte wurden eingetragen: ".$rn.$rn;
      $mailBody.= "Betreff: ".$mailSubject.$rn;
      $mailBody.= "Anrede: ".$salutation.$rn;
      $mailBody.= "Name: ".$fullName.$rn;
      $mailBody.= "E-Mail: ".$mailFrom.$rn;
      $mailBody.= "Mitteilung: ".$message.$rn;
      $mailBody.= "----- Ende der E-Mail -----".$rn;

      $mailHeader  = "MIME-Version: 1.0".$rn;  // MIME: Multipurpose Internet Mail Extension
      // reiner Text, also ohne Formatierung.
      $mailHeader .= "Content-Type: text/plain;charset=utf-8".$rn;
      // Zusätzliche Header anlegen.
      $mailHeader .= "From: ".$fullName."<$mailFrom>".$rn;
      $mailHeader .= "Reply-To: ".$mailFrom.$rn;
      $mailHeader .= "Return-Path: ".$mailFrom.$rn;
      $mailHeader .= "X-Mailer: PHP/".phpversion().$rn;

      // fehlerfreies Kontakftformular per E-Mail verschicken
```

```
      $isMailed = mail($mailTo,$mailSubject,$mailBody,$mailHeader);
      ?>
      <!—Erfolgs- oder Fehlermeldung ausgeben -->
      <html>
      <head>
         <title>Vielen Dank für Ihre E-Mail!</title>
      </head>
      <body>
         <form>
            <?php
               if($isMailed) {
                  echo "Wir haben Ihre Mitteilung erhalten und werden sie so schnell wie
möglich bearbeiten.";
               } else {
                  echo "Ein technischer Fehler ist aufgetreten. Ihre Mitteilung kann nicht
gesendet werden.";
               }
            ?>
            <p>Klicken Sie bitte auf die Schaltfläche <b>Zurück</b> Ihres Browsers, um zur
vorherigen Seite zurückzukehren.</p>
         </form>
      </body>
      </html>
<?php }  ?>
</body>
</html>
```

Listing 1: Gesamter Quellcode des tabellenlosen Kontaktformulars mit Spamschutz

Literatur

[1] o. V., „PHP/Komplexe Formulare - Sticky Forms: Den Zustand eines Formulars erhalten,“ 19 01 2009. [Online]. Available: http://mikiwiki.org/wiki/PHP/Komplexe_Formulare#Sticky_Forms:_Den_Zustand_eines_Formulars_erhalten. [Zugriff am 8 4 2019].

[2] G. Livan, „Creating a CSS TableLess Form (using Labels or DIVs),“ 28 02 2015. [Online]. Available: https://www.bitrepository.com/how-to-create-a-tableless-form.html. [Zugriff am 8 4 2919].

[3] L. Ullman, „Creating Dynamic Web Sites with PHP and MySQL,“ 14 7 2006. [Online]. Available: http://www.peachpit.com/articles/article.aspx?p=483799&seqNum=7. [Zugriff am 8 4 2019].

[4] o. V., „Sending emails in PHP & email injection attacks,“ 15 10 2006. [Online]. Available: https://phpsense.com/2006/php-email-injection-attacks/. [Zugriff am 8 4 2019].