Dokumentation: XML/HTTP-Post Interface für MMS

Version 1.0

Inhaltsverzeichnis

Einleitung **1**

Versenden von MMS Nachrichten **2**

Versand einer einfachen MMS Nachricht 2
Versand einer einfachen MMS Nachricht mit variablem Absender 5
Versendung einer MMS Nachricht mit Termin 6

Auswertung der Rückantwort **8**

Bestätigungen 8
Fatale Fehler 9
Fehlercodes 10

Sichere Verbindung SSL **11**

Mögliche Fehler **13**

Beispiele **14**

Java 15
Perl 16
PHP 17

Anhang **19**

A. Übersicht über die Schlüsselelemente 19
B. DTD: btn-mms-send 23
C. DTD: btn-mms-response 23

Index **24**

Einleitung

Neben SMS Diensten bietet BEYOND THE NET nun auch den Versand von MMS Nachrichten an. Dies ist über ein entsprechendes XML/HTTP POST Interface möglich, das speziell auf den Versand von MMS Nachrichten zugeschnitten ist.

In der aktuellen Ausbaustufe ist nur der Versand einfacher Multimedia-Mitteilungen möglich, die eine Überschrift, ein Bild und einen Text enthalten. Der Zugang wird in Zukunft noch ausgebaut, so dass man eine beliebige Anzahl an Dateien mitsenden und die die Nachricht definierende SMIL Datei selbst erstellen kann.

Versenden von MMS Nachrichten

Um eine Nachricht über den Gateway zu verschicken muss zunächst eine gültige XML Datei erstellt werden. XML-Dateien sind einfach Textdateien, die entweder mit jedem beliebigen Editor erstellt werden können, oder die man auch ganz einfach mit Hilfe von Programmen erzeugen kann. Dabei ist man auch nicht auf eine bestimmte Programmiersprache festgelegt, sondern kann jede Programmiersprache verwenden, die mit Texten umgehen kann.

Für das Versenden der XML-Datei über das HTTP Protokoll ist es allerdings notwendig eine Programmiersprache zu verwenden, die Netzwerkverbindungen mindestens auf Socket-Basis unterstützt, besser aber gleich Funktionen oder Methoden für einen HTTP Aufruf mitbringt. Für Java und Perl haben beispielsweise eine solche HTTP Funktionalität. PHP hat leider keine Standardfunktionen um auf einem anderen Server einen HTTP Request durchzuführen. Hier ist es evtl. notwendig das HTTP Protokoll rudimentär selbst zu implementieren und dann eine Socketverbindung aufzubauen. Alternativ kann bei PHP auch ein Modul aus der PEAR Bibliothek benutzt werden, welches das HTTP-Protokoll implementiert.

Für die genannten Programmiersprachen existiert im Kapitel Beispiele jeweils ein Listing um eine einfache MMS Nachricht zu versenden. Damit soll die Verfahrensweise eines HTTP-Post Aufrufs, in der jeweiligen Programmiersprache, verdeutlicht werden.

Versand einer einfachen MMS Nachricht

Eine einfache MMS übermittelt ein Bild mit Überschrift und dazugehörendem Text auf das Mobiltelefon eines oder mehrerer Empfänger. Neben der Angabe des Inhalts der MMS und der Zielrufnummer(n) ist es notwendig die BenutzerID und das Passwort ihres Kontos bei BEYOND THE NET anzugeben. Den Preis für eine solche Nachricht finden Sie in ihrem Vertrag bzw. in unserer gesonderten Preistabelle, die auf unserer Internetseite (http://www.btn.de/) zum Download bereitsteht.

Die XML-Datei muss zunächst mit einem gültigen XML-Header beginnen. Dieser sieht folgendermaßen aus:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Beachten sie bitte, dass der Wert des Attributs encoding auf die von ihnen benutzte Zeichencodierung gesetzt werden muss, damit in der Nachricht evtl. vorhandene Umlaute oder andere Sonderzeichen korrekt dargestellt werden können. Ein typischer Wert für ein Java-Programm welches MMS Nachrichten versenden soll ist UTF-8. Andere Programme auf Unix Umgebungen verwenden meist den Zeichensatz ISO-8859-1. Weitere Informationen hierzu sind auf Seite 3Bei den vier Elementen zu finden.

Auf den XML-Header folgt die Angabe des sogenannten DOCTYPE. Diese verweist auf die sogenannte DTD, welche als eine Art Schablone, für den Aufbau einer XML-Datei ist. Anhand des DOCTYPE und der DTD kann ermittelt werden ob eine XML-Datei gültig ist oder nicht. Wenn sie ohne DOCTYPE Angabe eine ansonsten gültige XML-Datei erstellen, wird unser Server diese Datei trotzdem zurückweisen, da er ihre XML-Datei nicht gegen die DTD validieren kann. Für das Versenden von MMS Nachricht über BEYOND THE NET lautet der DOCTYPE stets folgendermaßen:

<!DOCTYPE btn-mms-send SYSTEM "http://www.btn.de/dtd/btn-mms-send.dtd">

In einer XML-Datei existiert immer ein sogenanntes Root-Element. Dieses umschließt alle anderen Elemente der XML-Datei. In unserer XML-Datei heißt dieses Root-Element btnmms-send und wird folgendermaßen notiert:
<btn-mms-send> (Hier folgen die restlichen Elemente der XML-Datei)

</btn-mms-send>

Zwischen diesen beiden Einträgen sind noch mindestens drei weitere Elemente zwingend erforderlich um eine MMS Nachricht versenden zu können. Zunächst ist da das sender Element, welches an erster Stelle innerhalb des Root-Elementes stehen muss. Im sender Element müssen die Daten ihres Kontos bei BEYOND THE NET als Attribute angegeben werden. Das ist zum einen ihre BenutzerID, die als das Attribut userid angegeben werden muss, und zum anderen ihr Passwort, welches als Attribut password angegeben werden muss. Für den Fall, dass ihre BenutzerID „ABC00000“, und ihr Passwort „xyz0123“ lautet sähe dieses Element folgendermaßen aus:

```
<sender userid="ABC00000" password="xyz0123"/>
```

Auf das Element sender muss das Element simple-mms folgen. Dieses Element hat selbst keine Attribute und enthält nur andere Elemente. Bei einer einfachen MMS Nachricht enthält das Element simple-mms nur die Elemente headline, picture, text und sound. Das Element headline gibt dabei die Überschrift der MMS Nachricht an. Das Element picture definiert das Bild, welches angezeigt werden soll. Das Element text wiederum enthält den Text der MMS Nachricht, der versendet werden soll und das Element sound fügt der Nachricht eine musikalische Komponente hinzu. Das sieht in XML dann folgendermaßen aus:

```
<simple-mms>
        <headline>(Hier steht die Überschrift)</headline>
        <picture type=“image/jpeg“>(Hier muss das Bild eingefügt werden)</picture>
        <text>(Hier steht der Text der MMS Nachricht)</text>
        <sound type=“audio/x-midi“>(Hier muss die Musik-Datei eingefügt werden)</sound>
</simple-mms>
```

Bei den vier Elementen headline, picture, text und sound können die entsprechenden Informationen direkt in die XML-Datei geschrieben werden. Dabei ist zu beachten, dass das Bild als Hexadezimaler String codiert wird. Alternativ dazu kann auch bei jedem dieser drei Elemente ein Dateiname über das filename Attribut auf unserem FTP-Server angegeben werden. Die Elemente selbst bleiben dabei leer, was dann so aussähe:

```
<simple-mms>
        <headline filename=“headline.txt“/>
        <picture filename=“picture.jpg“/>
        <text filename=“text.txt”/>
        <sound filename=”sound.mid”/>
</simple-mms>
```

Die Verwendung von filename Attributen und direkt in die XML-Datei geschriebenen Daten kann natürlich bei den verschiedenen Elementen nach Belieben kombiniert werden. Bei ein und demselben Element schließt sich die Angabe von filename Attribut und Inhalt im Element gegenseitig aus.
Beim picture und beim sound Element sollte der MIME-Typ der Datei angegeben werden. Insbesondere, wenn die Daten in die XML-Datei geschrieben werden, ist dies zwingend

erforderlich. Wird das filename Attribut verwendet, wird versucht den MIME-Typ anhand der Dateiendung zu erkennen. Schlägt dies aber fehl, kommt es zu einer Fehlermeldung.

Umlaute im Text oder in der Überschrift können auf verschiedenen Wegen codiert werden. Normalerweise können die Umlaute einfach im Zeichensatz ISO-8859-1 codieren, mit dem dann auch die Übertragung durchgeführt wird. Soll der Zeichensatz UTF-8 verwendet werden, müssen alle Zeichen außerhalb des US-ASCII Zeichensatzes mit Hilfe numerischer Entities umschrieben werden. Es ist allerdings zu beachten, dass es nicht immer ausreicht in dem XML-Header einfach den Zeichensatz ISO-8859-1 anzugeben. Auch die verwendete Programmiersprache muss die Zeichen in der richtigen Codierung ausgeben. Hier ist teilweise eine explizite Angabe erforderlich. Die Umschreibung der Umlaute mit numerischen Entities kann folgender Tabelle entnommen werden:

Zeichen	numerische Entity
ä	&#228;
ö	&#246;
ü	&#252;
Ä	&#196;
Ö	&#214;
Ü	&#220;
ß	&#223;

Auf das simple-mms Element muss mindestens ein destination Element folgen. In einem destination Element wird festgelegt an welchen Empfänger eine MMS Nachricht gesendet werden soll. Dabei können theoretisch beliebig viele destination Elemente angegeben werden. Die tatsächliche Anzahl ist nur durch ein eventuelles Timeout der Verbindung und der Auslastung des Servers begrenzt. Bei bis zu 5000 Empfängern pro MMS Nachricht in einer XML-Datei treten im Normalfall keine Probleme auf. Das destination Element muss die Mobilfunkrufnummer des Empfängers im Internationalen Format enthalten. Das Internationale Format hat folgenden Aufbau:
+<Länderkennung><Vorwahl ohne 0><Teilnehmernummer>

Ein destination Element mit der Mobilfunknummer, von einem deutschen Vodafone Mobilfunkteilnehmer, mit der Handy-Nummer 0172-1234567, würde dann folgendermaßen aussehen:

```
<destination>+491721234567</destination>
```


Damit ist die XML-Datei zum Versand einer einfachen MMS Nachricht vollständig. Es folgt nun die komplette XML-Datei, wie sie an das Gateway gesendet werden kann:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE btn-mms-send SYSTEM "http://www.btn.de/dtd/btnmms-send.dtd">

<btn-mms-send>
    <sender userid="XXX00000" password="xyz0123"/>
    <simple-mms>
        <headline>&#220;berschrift</headline> <picture
        type="image/jpeg">87FE8F9E97C8A...
(usw.)</picture>
        <text>Text</text>
        <sound type="audio/x-midi">E7C87F5A8…(usw.)</sound>
    </simple-mms>
    <destination>+491721234567</destination>
</btn-mms-send>
```

Diese Datei kann dann mit Hilfe eines HTTP Post Uploads an die folgende URL gesendet werden: http://xml2sms1.btn.de:8080/sendMMS/sendMMS.do

Versand einer einfachen MMS Nachricht mit variablem Absender

Die Beschreibung, wie eine MMS Nachricht mit variablem Absender verschickt werden kann, baut auf dem Versand einer einfachen MMS Nachricht auf. Alle dort behandelten Dinge werden als bekannt vorausgesetzt.

Um der MMS Nachricht einen individuellen Absender zuzuweisen, muss in das simple-mms Element das Element originator eingefügt werden. Bei diesem Element muss das Attribut type angegeben werden. Es kann die Werte text oder number enthalten. Im Element selbst ist dann der gewünschte Absender anzugeben. Das Element originator sieht dann z.B. so aus:

```
<originator type="text">www.btn.de</originator>
```

Möchten Sie als Absender eine Telefonnummer angeben geben sie unter dem Attribut type bitte number an. Beachten sie bitte, dass neben den Ziffern von 0 bis 9 auch das Plus-Zeichen „+“ in einem Absender vom Typ number erlaubt ist, damit Internationale Telefonnummern angegeben werden können. Das Element originator sieht dann z.B. so aus:

```
<originator type="number">+491779876543</originator>
```


Eine gültige XML-Datei, die eine MMS Nachricht mit einer Telefonnummer als Absender versendet, sieht wie folgt aus:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE btn-mms-send SYSTEM "http://www.btn.de/dtd/btnmms-send.dtd">

<btn-mms-send>
    <sender userid="XXX00000" password="xyz0123"/>
    <simple-mms>
        <text>TEXT</text>
        <originator
type="number">+491779876543</originator>
    </simple-mms>
    <destination>+491721234567</destination>
</btn-mms-send>
```

Versendung einer MMS Nachricht mit Termin

Die Beschreibung, wie eine MMS Nachricht mit Termin versendet werden kann, baut auf dem Versand einer einfachen MMS Nachricht auf. Alle dort behandelten Dinge werden als bekannt vorausgesetzt.

Um eine MMS Nachricht zu einem bestimmten Zeitpunkt zu senden, muss in das simple-mms Element das Element delivery eingefügt werden.

Das Element delivery benötigt zwingend zwei Attribute. Mit dem Attribut date kann man festlegen an welchem Tag genau die MMS Nachricht gesendet werden soll. Dabei kann sowohl ein deutsches (TT.MM.JJJJ), als auch ein amerikanisches (MM-TT-JJJJ) Datumsformat verwendet werden. Das Attribut time muss die Uhrzeit, zu der die MMS Nachricht versendet werden soll, enthalten. Hierbei muss das 24 Stunden Format (SS:MM) benutzt werden. Ein gültiges delivery Element sieht folgendermaßen aus:

```
<delivery date="21.09.2002" time="11:50"/>
```

Eine MMS Nachricht die dieses Element innerhalb des message Elementes beinhaltet wird versendet, wenn der Zeitpunkt 21. September 2002, 11:50 Uhr erreicht ist, oder bereits in der Vergangenheit liegt.

Eine komplette XML-Datei einer einfachen MMS Nachricht mit Terminversand sieht folgendermaßen aus:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE btn-mms-send SYSTEM "http://www.btn.de/dtd/btnmms-send.dtd">

<btn-mms-send>
    <sender userid="XXX00001" password="geheim"/>
    <simple-mms>
            <headline>&#220;berschrift</headline>
            <picture filename=”soundso.jpg”/>
            <text>Text</text>
                        <delivery date="21.09.2002" time="11:50"/>
    </simple-mms>
    <destination>+491712345678</destination>
</btn-mms-send>
```


Auswertung der Rückantwort

Wie die XML-Dateien, die sie erstellen müssen um MMS Nachrichten zu versenden, aussehen müssen, ist ihnen jetzt bekannt. In diesem Abschnitt soll es nun darum gehen, wie die XML-Datei, die ihnen als Antwort gesendet wird aussieht, und was sie daraus erkennen können.

Auch die Antwort-XML-Datei hat eine Schablone, also eine DTD. Diese finden sie ebenfalls im Anhang.

Die XML-Datei, die sie als Antwort erhalten beginnt selbstverständlich mit einem gültigen XML-Header:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Darauf folgt auch hier eine gültige! DOCTYPE Deklaration. Diese beinhaltet natürlich eine andere DTD, nämlich diejenige die ein Antwort-Dokument definiert.

```
<!DOCTYPE btn-mms-response SYSTEM "http://www.btn.de/dtd/btn-mmsresponse.dtd">
```

Darauf folgt auch hier wieder das Root Element, welches in einer Antwort Datei btn-mmsresponse heißt. Dieses Element enthält alle weiteren Elemente:

```
<btn-mms-response> (Hier folgen die restlichen Elemente der XML-Datei)

</btn-mms-response>
```

Welche Elemente im Root-Element enthalten sind, richtet sich danach um welche Art von Fehler es sich handelt. Wir unterscheiden grundsätzlich zwischen zwei Arten von Fehlern. Zum einen sind da die Bestätigungen, die ihnen den Eingang eines Auftrags zum Versand einer MMS Nachricht an einen einzelnen Empfänger bestätigen oder nicht. Zum anderen gibt es die Fatalen Fehler, die eine Bearbeitung der von ihnen übermittelten XML-Datei unmöglich machen.

Bestätigungen

Bestätigungen enthalten im Root-Element eins oder mehrere destination Elemente. Die Anzahl der destination Elemente in der Antwort-XML-Datei stimmt immer mit der Anzahl der destination Elemente in der von ihnen übermittelten XML-Datei überein.

Der Aufbau des destination Elementes in der Antwort-XML-Datei ist jedoch ein anderer als in der von ihnen erstellten XML-Datei. Typischerweise sehen ein solche destination Elemente folgendermaßen aus:

```
<destination result=“success“ errorcode=“0“>+491721234567</destination>

<destination result=“error“ errorcode=“1“ message=“Wrong Phone Number Format“>01779876543</destination>
```

Das erste Element meldet einen Erfolgreichen Eingang einer MMS Nachricht an die Telefonnummer +491721234567. Das zweite Element meldet eine falsch formatierte Telefonnummer 01779876543. Aus dem Attribut result können sie ablesen, ob die Übermittlung der Nachricht an uns für den im Element angegebenen Empfänger erfolgreich war oder nicht. War sie erfolgreich, enthält das Attribut result den Wert success. Ist speziell für

den im Element angegebenen Empfänger ein Fehler aufgetreten finden sie im Attribut result den Wert error. Aus den Werten in den Attributen errorcode und message können sie dann den Grund für den Fehler ermitteln.

Eine Beispielantwort könnte so aussehen:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE btn-mms-response SYSTEM
"http://www.btn.de/dtd/btn-mms-response.dtd">

<btn-mms-response>

                        <destination result="success" errorcode="0">
                +491729419388
        </destination>

                        <destination result="success" errorcode="0">
                +491729419388
        </destination>

</btn-mms-response>
```

Fatale Fehler

Fatale Fehler treten auf, wenn es ein Grundsätzliches Problem gibt, welches die Verarbeitung der von Ihnen übermittelten XML-Datei nicht erlaubt. Der simpelste Fall ist sicherlich der, dass ihre XML-Datei nicht wohlgeformt ist. Das kann an nicht geschlossenen Anführungszeichen, nicht geschlossenen Elementen oder sonstigen Schreibfehler liegen. Zum anderen kann es sein, dass ihre XML-Datei nicht gültig im Sinne der DTD war. Ihre XML Datei kann ungültig werden, wenn zwingend erforderliche Elemente und/oder Attribute fehlen, an falscher Stelle stehen, oder zu oft vorkommen. Ein fataler Fehler gibt ihnen immer genau darüber Auskunft was nicht stimmte und hilft ihnen somit bei der Fehlerbeseitigung.

Im Root-Element der XML-Datei steht bei einem fatalen nur ein Element mit dem Namen fatal. Dieses Element enthält die Attribute errorcode und message, die man auslesen kann um den Grund des Fehlers herauszufinden.

Ein typisches solches Element sieht folgendermaßen aus:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE btn-mms-response SYSTEM
"http://www.btn.de/dtd/btn-mms-response.dtd">

<btn-mms-response>

        <fatal errorcode="9" message="Error on line 9 of
document : The element type &quot;text&quot; must be
terminated by the matching end-tag &quot;&lt;/text&gt;&quot;." />

</btn-mms-response>
```


Fehlercodes

Welcher Fehler aufgetreten ist kann am besten anhand der Fehlercodes ausgewertet werden, die bei jedem Fehler in dem errorcode Attribut übergeben werden. Folgende Fehlercodes gibt es:

Fehlercode	Bedeutung
1	kein Prepaid Guthaben mehr vorhanden
2	BenutzerID und/oder Passwort falsch
3	interner Fehler
4	keine Berechtigung
5	Account gesperrt
6	IP-Check fehlgeschlagen
7	Ungültige Kombination von Versandoptionen
9	Fehler im Aufbau der XML Datei

Sichere Verbindung SSL

Im Normalfall wird der HTTP-Aufruf im Klartext übermittelt. Alternativ dazu kann der Aufruf auch als HTTPS-Aufruf stattfinden. Hier wird die sogenannte Secure Socket Layer (SSL) verwendet, die eine entsprechende Sicherheit durch Verschlüsselung garantiert. Wenn die verwendete Programmiersprache dies unterstützt und eine entsprechende Programmierung auf Kundenseite erfolgt, reicht es meist aus die Protokollangabe „http://“ durch „https://“ in der URL zu ersetzen und den Port von 8080 auf 443 zu ändern.

Hier ein Beispiel in PHP für den SSL Aufruf:

```
<?php

$request_xml = '<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE btn-sms-send SYSTEM "http://www.btn.de/dtd/btn-sms-send.dtd">
<btn-sms-send><sender userid="Benutzername*" password="Passwort*"/>
<message tarif="30"><text>Hier steht ihr text</text><originator type="text">Absender*</originator></message>
<destination>Empfänger*</destination></btn-sms-send>';

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL,'https://freiburg.btn.de:443/sendSMS/sendSMS.do');
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_ANY);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_TIMEOUT, 4);
curl_setopt($ch, CURLOPT_POSTFIELDS, $request_xml);
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Connection: close'));
$start = array_sum(explode(' ', microtime()));
$result = curl_exec($ch);
$stop = array_sum(explode(' ', microtime()));
$totalTime = $stop - $start;
if ( curl_errno($ch) ) {
   $result = 'ERROR -> ' . curl_errno($ch) . ': ' . curl_error($ch);
   } else {
   $returnCode = (int)curl_getinfo($ch, CURLINFO_HTTP_CODE);
   switch($returnCode){
   case 200:
         break;
         default:
         $result = 'HTTP ERROR -> ' . $returnCode;
         break;
         }
   }
curl_close($ch);
echo 'Total time for request: ' . $totalTime . "\n";
echo $result;
?>
```


Benutzername * = hier tragen Sie den Benutzernamen ein, mit dem Sie sich auch bei unseren SMS-Gateway anmelden (https://www.sms-gateway.net)

Passwort *= hier tragen Sie das Passwort ein, mit dem Sie sich auch bei unseren SMS-Gateway anmelden (https://www.sms-gateway.net)

Absender * = hier können Sie den Absender eintrage.

type= "number"

Wenn der Absender nur aus Ziffern und dem plus-Zeichen (“+”) besteht.

type= "text"

Der Absender kann aus beliebigen Zeichen bestehen.

Empfänger * = hier tragen Sie die Handynummer des Empfängers ein.

Mögliche Fehler

Problem: Sie bekommen den Fehler: Fataler Fehler Code 9: Error online X: The entity name must immediately follow the '&' in the entity reference.

Lösung: Sie haben im Inhalt eines Tags, meist geschieht dies im Text der Nachricht, ein kaufmännisches Und (&) verwendet. Dieses Zeichen ist ein Schlüsselsymbol in XMLDateien, welches für die sogenannten Entities verwendet wird. Um ein kaufmännisches Und (&) im Nachrichtentext zu verwenden, benutzen sie bitte stattdessen die Zeichenfolge „&“.

Beispiele

Die Programmierbeispiele sollen in erster Linie verdeutlichen, wie eine XML-Datei per HTTPPOST versendet werden kann. In den Beispielen werden XML-Dateien für den SMS Gateway versendet. Die XML-Datei auf MMS zu ändern sollte für einen Programmierer kein Problem sein.

Java

```
// ----------------------------------------------------
// sendsms.java Version 1.0
// ----------------------------------------------------
// BEYOND THE NET - Internet Service GmbH
// Giradetstr. 2-38
// 45131 Essen
// Germany
// ----------------------------------------------------
// This Application is a simple example how to send // sms messages via the http post interface at BTN.
// ----------------------------------------------------
// Created: 21.09.2002
// Last Modified: 21.09.2002
import java.net.*; import java.io.*; public class sendsms {
   // userid used to send the message
   private static String UserID = "your userid goes here";
   // password matching the userid
   private static String Password = "your password goes here";
   // destination mobile phone number
   private static String Destination = "your destination goes here";
   // text of the message that will be sent
   private static String Text = "This is a test sms message via the BTN sms gateway";
   public static void main(String [] args) { try {
   // make a new URL for the BTN webservice "send sms"
   URL myURL = new
URL("http://xml2sms1.btn.de:8080/sendSMS/sendSMS.do");

   // connect to this URL using a http connection
   HttpURLConnection myHttpCon =
(HttpURLConnection)myURL.openConnection();

   // specify to use the POST method and that we want to send content     myHttpCon.setRequestMethod("POST");
myHttpCon.setDoOutput(true);
   // get the output stream of the http connection     PrintWriter out = new
PrintWriter(myHttpCon.getOutputStream());
   // write the xml header and doctype into the stream     out.println("<?xml version=\"1.0\" encoding=\"UTF-
8\"?>");     out.println("<!DOCTYPE btn-sms-send

\"http://www.btn.de/dtd/btn-sms-send.dtd\">");
                SYSTEM \"http://www.btn.de/dtd/btn-sms-send.dtd\">");
```



```
        // starting with the root element     out.println("<btn-sms-send>");

        // specify the sender information     out.println("<sender userid=\""+UserID+"\"
password=\""+Password+"\"/>");

        // we try to send a simple text message
       out.println("<message><text>"+Text+"</text></message>");

        // specify the destination of the message
       out.println("<destination>"+Destination+"</destination>");

        // closing the root element and the output stream
        out.println("</btn-sms-send>");     out.close();

        // open the input stream to get the response
        InputStream in = myHttpCon.getInputStream();

        // read response from the input stream into a buffer an put
        // it out on stdout
        byte[] myBuffer = new byte[2048];     int i = in.read(myBuffer);     while(i > 0) {
                System.out.print(new String(myBuffer,0,i)); i = in.read(myBuffer);
        }

      } catch (Exception e) {
        // catching all exceptions and putting them out
        e.printStackTrace();
      }
  }

              }
```

Perl

```
#!/usr/bin/perl
# ------------------------------------------------
# sendsms.pl Version 1.0
# ------------------------------------------------
# BEYOND THE NET - Internet Service GmbH
# Giradetstr. 2-38
# 45131 Essen

# Germany
# ------------------------------------------------
# This script is a simple example how to send sms # messages via the http post interface at BTN.
# Perl modules HTTP::Request and LWP::UserAgent # have to be installed to use this script.
# ------------------------------------------------
# Created: 21.09.2002
# Last Modified: 21.09.2002
use HTTP::Request; use LWP::UserAgent; # Please modify this values to reach functionality
$UserID = 'your userid goes here';
$Password = 'your password goes here';
# You should customize this values to specify # where to send a sample message.
$Destination = 'your destination mobile phone number goes here'; $Text = 'This is a test via BTN SMS Gateway';
# creating new user agent
```

```
$useragent = LWP::UserAgent->new;
# creating new http post request
$request = HTTP::Request->new( POST =>
'http://xml2sms1.btn.de:8080/sendSMS/sendSMS.do');
# setting the content type of the request
$request->header(Content-Type => 'text/xml');
# filling the content with standard header, doctype and root element
$request->content($xmlfile = '<?xml version="1.0" encoding="ISO-8859-1"?><!DOCTYPE btn-sms-send SYSTEM
"http://www.btn.de/dtd/btn-sms-send.dtd"><btn-sms-send>');
# adding user information
$request->add_content("<sender userid=\"$UserID\" password=\"$Password\"/>");
# adding message text
$request-
>add_content("<message><text>$Text</text></message>");
# adding destination of the message
$request-
>add_content("<destination>$Destination</destination>");
# closing root element
$request->add_content("</btn-sms-send>");
# sending request
$response = $useragent->request($request);
# diplay response output print $response->content();
```

PHP

```
<html>
<head><title>send sms message via BTN gateway</title></head>
<body>
<h1>BEYOND THE NET - Internet Service GmbH</h1>
<p>This script is a small example demonstrating how to send
sms messages
via the HTTP Post interface of the BTN sms gateway</p>
<p>It shows the fundamentals in using HTTP Post in PHP. It sends
only simple messages. Adding other send options can be acomplished by altering the XML file sent to the
Server.</p>

<form action="sendsms.php" method="post" name="daten">
<p>userid:<br><input type="text" name="userid"></p>
<p>password:<br><input type="password" name="password"></p>
<p>destination:<br><input type="text"
name="destination"></p>
<p>text:<br><input type="text" name="text" value="Test via
BTN gateway"></p>
<p><input type="submit"></p>
<input type="hidden" name="action" value="doit">
</form>

<p>
<?php // function to send simple XML content via http

function postHttp($host, $port, $path, $Content) {
      $fp = fsockopen($host,$port,$errno,$errstr,30);
```

```
        fputs($fp, "POST $path HTTP/1.1\n"); fputs($fp, "Host: $host\n");
        fputs($fp, "Content-type: text/xml\n");
        fputs($fp, "Content-length: ".strlen($Content)."\n");
        fputs($fp, "Connection: close\n\n"); fputs($fp, $Content);

        while(!feof($fp)) {
                $RetVal .= fgets($fp,128);
        }

        return $RetVal;
}

if ($action == "doit") {

        $Content = '<?xml version="1.0" encoding="ISO-8859-1"?
>';
        $Content .= '<!DOCTYPE btn-sms-send SYSTEM
"http://www.btn.de/dtd/btn-sms-send.dtd">';
        $Content .= '<btn-sms-send>';
        $Content .= "<sender userid=\"$userid\" password=\"$password\"/>";
        $Content .= "<message><text>$text</text></message>";
        $Content .= "<destination>$destination</destination>";
        $Content .= '</btn-sms-send>';

        echo nl2br(htmlspecialchars(postHttp("xml2sms1.btn.de",8080,"/sen dSMS/sendSMS.do",$Content))); }

?>
</p>

</body>
</html>
```

Anhang

A. Übersicht über die Schlüsselelemente

<table>
<tr><th>Element</th><th>Beschreibung</th></tr>
<tr><td><btn-mms-send></td><td>Root-Element. Enthält alle weiteren Elemente der XML-Datei.

Dieses Element darf keine Attribute haben.</td></tr>
<tr><td><sender></td><td>Darf selbst keinen Inhalt haben, sondern hat nur Attribute.
<table>
<tr><th>Attribut</th><th>Beschreibung</th></tr>
<tr><td>userid=“...“</td><td>BenutzerID des Versenders der SMS Nachricht</td></tr>
<tr><td>password=”...”</td><td>Passwort des Versenders der SMS Nachricht.</td></tr>
<tr><td>customnumber=”...”</td><td>Optional. Selbst frei definierbare Kundennummer des Versenders der SMS Nachricht.</td></tr>
</table>
</td></tr>
<tr><td><simple-mms></td><td>Enthält weitere Elemente, die den Inhalt der MMS Nachricht festlegen. Mögliche enthaltenen Elementen sind:

• <headline>
• <picture>
• <text>
• <sound>
• <originator>
• <delivery></td></tr>
</table>

<table>
<tr><th>Element</th><th>Beschreibung</th></tr>
<tr><td><headline></td><td>Enthält die Überschrift der MMS Nachricht. Alternativ kann der Inhalt durch eine Datei auf dem FTP-Server bestimmt werden.
<table>
<tr><th>Attribut</th><th>Beschreibung</th></tr>
<tr><td>filename=”...”</td><td>Dateiname der zu versendenden Überschrift.</td></tr>
</table></td></tr>
<tr><td><picture></td><td>Enthält das in der MMS zu versendende Bild als Hexadezimaler String. Alternativ kann das Bild durch eine Datei auf dem FTP-Server bestimmt werden.
<table>
<tr><th>Attribut</th><th>Beschreibung</th></tr>
<tr><td>type=“...“</td><td>MIME-Typ der Bilddatei.</td></tr>
<tr><td>filename=”...”</td><td>Dateiname des zu versendenden Bildes.</td></tr>
</table></td></tr>
<tr><td><text></td><td>Enthält den Text der MMS Nachricht
<table>
<tr><th>Attribut</th><th>Beschreibung</th></tr>
<tr><td>filename=”...”</td><td>Dateiname des zu versendenden Textes.</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Element</th><th>Beschreibung</th></tr>
</table>

<table>
<tr><td><sound></td><td>Enthält eine Audio-Datei, die während der Nachricht abgespielt wird.<table><tr><th>Attribut</th><th>Beschreibung</th></tr><tr><td>type="..."</td><td>MIME-Typ der Musikdatei.</td></tr><tr><td>filename="..."</td><td>Dateiname der zu versendenden Datei.</td></tr></table></td></tr>
<tr><td><originator></td><td>Enthält den Variablen Absender der MMS Nachricht. Der Typ des Absenders wird durch Attribute bestimmt.<table><tr><th>Attribut</th><th>Beschreibung</th></tr><tr><td>type="..."</td><td>• number
Wenn der Absender nur aus Ziffern und dem plus-Zeichen ("+") besteht.
• text
Der Absender kann aus beliebigen Zeichen bestehen.</td></tr></table></td></tr>
</table>

<table>
<tr><td><delivery></td><td>Darf selbst keinen Inhalt haben, sondern hat nur Attribute.<table><tr><th>Attribut</th><th>Beschreibung</th></tr><tr><td>date=“...“</td><td>Datum zu dem die MMS Nachricht versendet werden soll.</td></tr><tr><td>time=“...“</td><td>Uhrzeit zu der die MMS Nachricht versendet werden soll</td></tr></table></td></tr>
</table>

Element	Beschreibung
<destination>	Enthält die Mobilfunkrufnummer eines Empfängers der MMS Nachricht.

B. DTD: btn-mms-send

```
<?xml version="1.0" encoding="UTF-8"?>

<!ELEMENT btn-sms-send (sender, simple-mms, destination+)>
<!ATTLIST btn-sms-send test CDATA #IMPLIED
>

<!ELEMENT sender EMPTY>
<!ATTLIST sender userid CDATA #REQUIRED password CDATA
      #REQUIRED customnumber CDATA #IMPLIED
>

<!ELEMENT simple-mms
(headline?,picture?,text?,originator?,delivery?)>

<!ELEMENT headline (#CDATA)>
<!ATTLIST headline filename CDATA #IMPLIED
>

<!ELEMENT picture (#CDATA)>
<!ATTLIST picture type CDATA #IMPLIED filename CDATA
      #IMPLIED
>

<!ELEMENT text (#CDATA)>
<!ATTLIST text filename CDATA #IMPLIED
>

<!ELEMENT originator (#CDATA)>
<!ATTLIST originator type (text | number) #REQUIRED
>

<!ELEMENT delivery EMPTY>
<!ATTLIST delivery date CDATA #REQUIRED time CDATA
      #REQUIRED
>

<!ELEMENT destination (#CDATA)>
```

C. DTD: btn-mms-response

```
<?xml version="1.0" encoding="UTF-8"?>

<!ELEMENT btn-mms-response (fatal | destination+)>
<!ELEMENT fatal (#PCDATA)>
<!ATTLIST fatal errorcode (1) #REQUIRED message CDATA
      #REQUIRED
>
<!ELEMENT destination (#PCDATA)>
<!ATTLIST destination result (success | error) #REQUIRED errorcode (0 | 1) "0"
      message CDATA "The Message was successfully sent."
>
```


Index

B

BenutzerID.....3

btn-mms-response.....9

btn-mms-send.....4

D

date.....7
delivery.....7
destination.....5
DOCTYPE.....3
DTD.....3

E

Encoding.....3

H

HTTP Protokoll.....3

I internationales Format.....5

O

originator.....6

P

password.....4

R

Root-Element.....4
Rückantwort.....9

S

sender.....4

simple-mms.....4

SSL 10

T

Termin.....7

text.....4

time.....7

U

Umlaute.....5

userid.....4

V

variabler Absender.....6

X

XML-Header.....3

