



Schnittstellen- beschreibung



Inhaltsverzeichnis

1 Versions-Historie.....	4
1 Allgemein	5
1.1 Einleitung – Was kann man mit SMSworkx eigentlich machen?.....	5
2 SMS Typen.....	6
3 SOAP.....	7
3.1 SOAP Funktionen.....	7
3.1.1 SendSimpleSMS.....	7
3.1.1.1 SOAP.....	8
3.1.1.2 HTTP GET.....	9
3.1.1.3 HTTP POST.....	9
3.1.2 SendSMS und SendSMSText.....	10
3.1.2.1 SOAP.....	11
3.1.2.2 HTTP GET.....	12
3.1.2.3 HTTP POST.....	12
3.1.3 SendUDH.....	13
3.1.3.1 SOAP.....	14
3.1.3.2 HTTP GET.....	14
3.1.3.3 HTTP POST.....	15
3.1.4 SMSBin-Typen	15
3.1.5 SendBinData.....	16
3.1.5.1 SOAP	16
3.1.5.2 HTTP GET.....	17
3.1.5.3 HTTP POST.....	17
3.1.6 SendBinURL.....	18
3.1.6.1 SOAP	18
3.1.6.2 HTTP GET.....	19
3.1.6.3 HTTP POST.....	19
3.1.7 Senden von Logos und Klingeltönen	20
3.1.7.1 Senden über die SOAP-Schnittstelle.....	20
3.1.7.2 SendHex.....	20
3.1.7.3 Parameter.....	20
3.1.8 VB Sample mit SMSCOMAPI.....	21
3.1.8.1 Perl Sample.....	22
3.1.8.2 PHP Sample.....	23
3.2 Versand von mehreren SMS pro Auftrag.....	24
3.2.1 Alle Empfänger selber Text.....	24
3.2.2 Mehrere Empfänger unterschiedlicher Text.....	24
4 SMTP (e-Mail 2 SMS).....	25
4.1 Allgemein.....	25
4.2 Aufbau einer Email.....	25
4.3 Beispiel.....	25
5 RückSMS / Delivery Reports.....	26
5.1 Zustellung per HTTP-Request.....	26
5.2 Zustellung per Email.....	26
5.2.1 Alternativer Empfänger der RückSMS.....	26
5.3 Per Abfrage.....	26
5.3.1 Abfrage auf einer Webseite.....	26
5.3.2 Received SMS.....	27
5.3.2.1 SOAP.....	27
5.3.2.2 HTTP GET.....	28
5.3.2.3 HTTP POST.....	28

6 Statistik.....	29
6.1 Account-Abfrage.....	29
6.1.1 QueryBalance	29
6.1.1.1 SOAP.....	29
6.1.1.2 HTTP GET.....	29
6.1.1.3 HTTP POST.....	30
7 Send MMS.....	31
7.1.1 Aufbau.....	31
7.1.1.1 SOAP.....	32
7.1.1.2 HTTP GET.....	32
7.1.1.3 HTTP POST.....	33
8 Glossar.....	34
9 Sonstiges.....	36

1 Versions-Historie

Datum	Autor	Bemerkung
06.06.2001	Jürgen Weisental	Erstellung der Schnittstellenbeschreibung
07.04.2004	Jürgen Weisental	Send MMS
21.09.2004	Jürgen Weisental	SMS Typ 19 – Lange SMS mit Absendererkennung über Direktanbindung
22.09.2004	Martin Ammer	Rücksms / delivery Reports
29.03.2005	Boris Stark	Redaktionelle Überarbeitung der Schnittstellen-Beschreibung

1 Allgemein

1.1 Einleitung – Was kann man mit SMSworkx eigentlich machen?

Die SMSworkx Produktlinie bietet die Möglichkeit wichtige Informationen, wie zum Beispiel Werbung, Termine, neue Produkte, Geburtstagsgrüße und vieles mehr per SMS auf die Mobiltelefone von Mitarbeitern, Gästen, Geschäftspartnern und Kunden zu senden. Hierbei werden alle deutschen Mobilfunkunternehmen unterstützt. Des Weiteren besteht die Möglichkeit an SMS auch weitest gehend weltweit zu versenden.

Um Ihnen die Anbindung an unsere SMSworkx Produktreihe zu erleichtern, finden Sie im Folgenden eine ausführliche Schnittstellenbeschreibung. Es wurde versucht die einzelnen Möglichkeiten so einfach und präzise wie möglich zu erläutern.

1.2 Möglichkeiten eine SMS über eine Schnittstelle zu versenden

SOAP	Protokoll, mit dem man Daten zwischen Systemen austauschen kann und Remote Procedure Calls durchgeführt werden können. SOAP verwendet zur Darstellung der Daten XML, zur Übertragung der Nachrichten Internet-Protokolle wie z.B. TCP und HTTP
HTTP GET	Die Einfachste und wohl am meist genutzten Methode, SMS per Schnittstelle zu Übergeben.
HTTP POST	HTTP Schnittstelle
SMTP	Übertragung als E-Mail (E-Mail to SMS) Achtung: diese Schnittstelle ist nicht für den Versand von Massen-SMS geeignet. Es kann immer nur eine SMS an mehrere Empfänger verschickt werden
UCP	Diese Schnittstelle ermöglicht den Empfang und den Versand von Daten im UCP-Format. Grundlage dieser Schnittstelle ist das von der Firma GMC entwickelte Universal Computer Protocol (UCP), welches ursprünglich für die Kommunikation zwischen einem Short Message Service Center (SMSC) eines Mobilfunk-Netzes und einem UCP-Client entworfen wurde. Achtung: zurzeit noch nicht verfügbar
SMPP	Ein auf TCP/IP basierendes Industriestandard-Protokoll zur Anbindung von SMSC zu SMSC (Short Message Service Centern) bzw. zu SMS-Service-Plattformen wie z.B. smsworkx. Achtung: in der Testphase
FTP	Schnittstelle um SMS über FTP-Zugang (File Transfer Protocol) zu übertragen. Achtung: zurzeit noch nicht verfügbar

2 SMS Typen

Nach folgend erhalten Sie eine Beschreibung der möglichen SMS-Typen:

Typ	Beschreibung
1	SMS mit Absenderkennung über ausländische Routen
2	Flash SMS
3	kleines Logo (72 x 14 Pixel – Nokia)
4	Picture SMS (maximal 120 Zeichen – Nokia)
5	Klingelton im RTTL Format (Nokia)
6	SMS ohne Absenderkennung (Standard SMS / Rück-SMS)
7	UDH
8	Bookmark (Test)
9	VCARD 1
10	VCARD 2 (Test)
12	EMS Sound (Test)
13	EMS Animation (Test)
14	EMS Bitmap (Test)
15	großes Logo (72 x 28 Pixel – Nokia)
16	SMS ohne Absenderkennung mit Auslieferungsstatus (Delivery Notification – Versand-Benachrichtigung)
17	lange SMS (wird mit 7 Bit codiert; pro 152 Zeichen wird eine SMS benötigt)
18	SMS mit Absenderkennung über Direktanbindung (High Quality Routen)
19	lange SMS mit Absendererkennung über Direktanbindung (wird mit 7 Bit codiert: pro 152 Zeichen wird eine SMS benötigt)

3 SOAP

Die Daten werden über eine HTTP SOAP Schnittstelle gesendet.

Die WSDL Datei (XML Formatbeschreibung zum Senden von SMS an das SMSworkx Gateway) finden Sie unter:

<http://soap.smsworkx.de/send.asmx?WSDL>

Um das Versenden von SMS zu unterbinden, verwenden Sie bitte als User und Passwort „Test“.

Achtung: Es gibt von jeder Funktion zwei Versionen. Die Erste unterstützt SOAPRPC (z.B. PoketSoap). Version 2 ist für andere Anwendungen zu verwenden.

3.1 SOAP Funktionen

3.1.1 SendSimpleSMS

Bei mehreren Empfängern oder unterschiedlichen Texten siehe: Punkt 3.2.2

Die einfachste Möglichkeit eine SMS zu übertragen.

Bitte öffnen Sie <http://soap.smsworkx.de/send.asmx> und füllen Sie die Felder zum Testen aus.

Folgende Parameter können verwendet werden:

User	String	Benutzername
Password	String	Passwort
Recipient	String	Empfänger
SMSText	String	Text

Es können alle möglichen Zeichen verwendet werden, da die SOAP-Schnittstelle als Kodierung der Zeichen UTF-8 verwendet.

Nach Versand der SMS erhalten Sie diese Rückgabewerte:

OK = Alles erfolgreich!

ERROR + Beschreibung des Fehlers

3.1.1.1 SOAP

Hier ein SOAP Beispiel. Es enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden:

```
POST /send.asmx HTTP/1.1
Host: soap.smsworkx.de
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://cetix.de/SendSMS/SendSimpleSMS"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="http://cetix.de/SendSMS"
xmlns:types="http://cetix.de/SendSMS/encodedTypes"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body
soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <tns:SendSimpleSMS>
      <User xsi:type="xsd:string">string</User>
      <Password xsi:type="xsd:string">string</Password>
      <Recipient xsi:type="xsd:string">string</Recipient>
      <SMSText xsi:type="xsd:string">string</SMSText>
    </tns:SendSimpleSMS>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="http://cetix.de/SendSMS"
xmlns:types="http://cetix.de/SendSMS/encodedTypes"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body
soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <tns:SendSimpleSMSResponse>
      <SendSimpleSMSResult xsi:type="xsd:string">string</SendSimpleSMSResult>
    </tns:SendSimpleSMSResponse>
  </soap:Body>
</soap:Envelope>
```

3.1.1.2 HTTP GET

Hier ein HTTP GET Beispiel. Es enthält Aufruf und Antwort. Die gezeigten Platzhalter müssen mit tatsächlichen Werten ersetzt werden.

```
GET /send.asmx/SendSimpleSMS?User=string&Password=string&Recipient=string&SMSText=string HTTP/1.1
Host: soap.smsworkx.de
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

3.1.1.3 HTTP POST

Hier ein HTTP POST Beispiel. Es enthält Aufruf und Antwort. Die gezeigten Platzhalter müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx/SendSimpleSMS HTTP/1.1
Host: soap.smsworkx.de
Content-Type: application/x-www-form-urlencoded
Content-Length: length

User=string&Password=string&Recipient=string&SMSText=string
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

3.1.2 SendSMS und SendSMSText

Bei mehreren Empfängern oder unterschiedlichen Texten siehe: Punkt 3.2.2

Sendet SMS mit dem gleichen Text an mehrere Empfänger.

Bitte <http://soap.smsworkx.de/send.asmx> öffnen und die Felder zum Testen füllen.

User	String	Benutzername
Password	String	Passwort
Caption	String	Beschreibung des SMS Auftrages (dient der Identifizierung im User Menü)
Sender	String	Absender (maximal 11 Zeichen oder eine Nummer)
Recipient	String Array	Empfänger der SMS und parametrisierte Felder Beispiel: +491711234567;Hans;Peter;
SMSText	String	Text der SMS mit 2 parametrisierbaren Feldern #FELD1# und #FELD2# Beispiel: Hallo Herr #FELD1# alles Gute zum #FELD2# Geburtstag.
SmsTyp	INT	Siehe Tabelle
SendDate	DateTime	Sendezeit des Auftrages

Nach Versand der SMS erhalten Sie diese Rückgabewerte:

OK = Alles erfolgreich!

ERROR + Beschreibung des Fehlers

3.1.2.1 SOAP

Hier ein SOAP Beispiel. Es enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden

```
POST /send.asmx HTTP/1.1
Host: soap.smsworkx.de
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://cetix.de/SendSMS/SendText"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="http://cetix.de/SendSMS"
xmlns:types="http://cetix.de/SendSMS/encodedTypes"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body
soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <tns:SendText>
      <User xsi:type="xsd:string">string</User>
      <Password xsi:type="xsd:string">string</Password>
      <Caption xsi:type="xsd:string">string</Caption>
      <Sender xsi:type="xsd:string">string</Sender>
      <Recipient xsi:type="xsd:string">string</Recipient>
      <SMSText xsi:type="xsd:string">string</SMSText>
      <SmsTyp xsi:type="xsd:string">string</SmsTyp>
      <SendDate xsi:type="xsd:string">string</SendDate>
    </tns:SendText>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="http://cetix.de/SendSMS"
xmlns:types="http://cetix.de/SendSMS/encodedTypes"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body
soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <tns:SendTextResponse>
      <SendTextResult xsi:type="xsd:string">string</SendTextResult>
    </tns:SendTextResponse>
  </soap:Body>
</soap:Envelope>
```

3.1.2.2 HTTP GET

Hier ein http GET Beispiel. Es enthält Aufruf und Antwort. Die gezeigten Platzhalter müssen mit tatsächlichen Werten ersetzt werden.

```
GET /send.asmx/SendText?User=string&Password=string&Caption=string&Sender=string&Recipient=string&SMSText=string&SmsTyp=string&SendDate=string HTTP/1.1
Host: soap.smsworkx.de
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

3.1.2.3 HTTP POST

Hier ein HTTP POST Beispiel. Es enthält Aufruf und Antwort. Die gezeigten Platzhalter müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx/SendText HTTP/1.1
Host: soap.smsworkx.de
Content-Type: application/x-www-form-urlencoded
Content-Length: length

User=string&Password=string&Caption=string&Sender=string&Recipient=string&SMSText=string&SmsTyp=string&SendDate=string
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```


3.1.3.1 SOAP

Hier ein SOAP Beispiel. Es enthält Aufruf und Antwort. Die gezeigten **Platzhalter** müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx HTTP/1.1
Host: soap.smsworkx.de
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://cetix.de/SendSMS/SendUDH"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schem-
as.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SendUDH xmlns="http://cetix.de/SendSMS">
      <User>string</User>
      <Password>string</Password>
      <Recipient>string</Recipient>
      <Sender>string</Sender>
      <Data>string</Data>
      <sendDate>string</sendDate>
    </SendUDH>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schem-
as.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SendUDHResponse xmlns="http://cetix.de/SendSMS">
      <SendUDHResult>string</SendUDHResult>
    </SendUDHResponse>
  </soap:Body>
</soap:Envelope>
```

3.1.3.2 HTTP GET

Hier ein HTTP GET Beispiel. Es enthält Aufruf und Antwort. Die gezeigten **Platzhalter** müssen mit tatsächlichen Werten ersetzt werden.

```
GET /send.asmx/SendUDH?User=string&Password=string&Recipi-
ent=string&Sender=string&Data=string&sendDate=string HTTP/1.1
Host: soap.smsworkx.de
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

3.1.3.3 HTTP POST

Hier ein HTTP POST Beispiel. Es enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx/SendUDH HTTP/1.1
Host: soap.smsworkx.de
Content-Type: application/x-www-form-urlencoded
Content-Length: length

User=string&Password=string&Recipient=string&Sender=string&Data=string&send
Date=string
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

3.1.4 SMSBin-Typen

Nach folgend erhalten Sie eine Auflistung der möglichen SMSBin-Typen (eine Beschreibung der Abkürzungen finden Sie im Glossar):

Typ	Beschreibung
10	BMP
11	JPG
12	GIF
13	NOL
14	PNG
31	RTTL

3.1.5 SendBinData

Senden von Binären Daten

User Benutzerkennung
Password Passwort
Recipient Empfänger
strMsg Nachricht
MsgType siehe Api Doku 4.1
binType siehe Api Doku 3.1.4
Data ByteArray der Datei
sendDate Sendezeit der Nachricht

Ergebnis:

OK MsgID oder Error Nachricht

3.1.5.1 SOAP

Hier ein SOAP Beispiel. Es enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx HTTP/1.1
Host: soap.smsworkx.de
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://cetix.de/SendSMS/SendBinData"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schem-
as.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SendBinData xmlns="http://cetix.de/SendSMS">
      <User>string</User>
      <Password>string</Password>
      <Recipient>string</Recipient>
      <strMsg>string</strMsg>
      <MsgType>int</MsgType>
      <binTyp>int</binTyp>
      <Data>base64Binary</Data>
      <sendDate>dateTime</sendDate>
    </SendBinData>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schem-
as.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SendBinDataResponse xmlns="http://cetix.de/SendSMS">
      <SendBinDataResult>string</SendBinDataResult>
    </SendBinDataResponse>
  </soap:Body>
</soap:Envelope>
```

3.1.5.2 HTTP GET

Hier ein HTTP GET Beispiel. Es enthält Aufruf und Antwort. Die gezeigten Platzhalter müssen mit tatsächlichen Werten ersetzt werden

```
GET /send.asmx/SendBinData?User=string&Password=string&Recipient=string&strMsg=string&MsgType=string&binTyp=string&Data=string&Data=string&sendDate=string HTTP/1.1
Host: soap.smsworkx.de
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

3.1.5.3 HTTP POST

Hier ein HTTP POST Beispiel. Es enthält Aufruf und Antwort. Die gezeigten Platzhalter müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx/SendBinData HTTP/1.1
Host: soap.smsworkx.de
Content-Type: application/x-www-form-urlencoded
Content-Length: length

User=string&Password=string&Recipient=string&strMsg=string&MsgType=string&binTyp=string&Data=string&Data=string&sendDate=string
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

3.1.6 SendBinURL

User	Benutzerkennung
Password	Passwort
Recipient	Empfänger
MsgType	siehe Api Doku 4.1
binType	siehe Api Doku 3.1.4
URL	URL der Binaren Datei muss erreichbar sein.
sendDate	Sendezeit der Nachricht

Ergebnis:

OK MsgID oder Error Nachricht

3.1.6.1 SOAP

Hier ist ein SOAP Beispiel, dieses enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden

```
POST /send.asmx HTTP/1.1
Host: soap.smsworkx.de
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://cetix.de/SendSMS/SendBinURL"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schem-
as.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SendBinURL xmlns="http://cetix.de/SendSMS">
      <User>string</User>
      <Password>string</Password>
      <Recipient>string</Recipient>
      <MsgType>int</MsgType>
      <binTyp>int</binTyp>
      <URL>string</URL>
      <sendDate>dateTime</sendDate>
    </SendBinURL>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schem-
as.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SendBinURLResponse xmlns="http://cetix.de/SendSMS">
      <SendBinURLResult>string</SendBinURLResult>
    </SendBinURLResponse>
  </soap:Body>
</soap:Envelope>
```

3.1.6.2 HTTP GET

Hier ist ein HTTP GET Beispiel, dieses enthält Aufruf und Antwort. Die gezeigten Platzhalter müssen mit tatsächlichen Werten ersetzt werden

```
GET /send.asmx/SendBinURL?User=string&Password=string&Recipient=string&MsgType=string&binTyp=string&URL=string&sendDate=string HTTP/1.1
Host: soap.smsworkx.de
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

3.1.6.3 HTTP POST

Hier ist ein HTTP POST Beispiel, dieses enthält Aufruf und Antwort. Die gezeigten Platzhalter müssen mit tatsächlichen Werten ersetzt werden

```
POST /send.asmx/SendBinURL HTTP/1.1
Host: soap.smsworkx.de
Content-Type: application/x-www-form-urlencoded
Content-Length: length

User=string&Password=string&Recipient=string&MsgType=string&binTyp=string&URL=string&sendDate=string
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

3.1.7 Senden von Logos und Klingeltönen

3.1.7.1 Senden über die SOAP-Schnittstelle

Unter <http://soap.smsworkx.de/send.asmx> kann der Webservice angesprochen werden. Um eine Logo oder Klingelton zu senden wird die Funktion SendHex verwendet.

3.1.7.2 SendHex

3.1.7.3 Parameter

User	Benutzername
Password	Passwort
Recipient	Empfänger des Logos/Klingelton
HexData*	Daten des zu senden Objektes in Hex codiert
MsgType	Endung der Datei Mögliche Optionen
	Logos: GIF, BMP, NOL, JPG (72x 14 Pix)
	Klingeltöne: MID und RTTL

- Die Datei muss von Binär nach Hex codiert werden dabei ist darauf zu achten das jedes Byte in 2 Hex Zeichen umgewandelt wird.

Beispiel: Binär 0 wird zu Hex 00

3.1.8 VB Sample mit SMSCOMAPI

Bei mehreren Empfängern oder unterschiedlichen Texten siehe: Punkt 2.3

Vor dem Versenden über COM-Schnittstelle erst:

(<http://www.smsworkx.de/docs/smscomapi.exe>) installieren!

```
Dim oSend As smsworkx.SendSMS
    Set oSend = New smsworkx.SendSMS

    'SMS Typ ist in der API Beschreibung aufgeführt.

    MsgBox oSend.SendSMS("serial", "password", "", "Hallo Welt aus VB",
"01714911446", Now, 6)

' Syntax ist Benutzername, Passwort, Caption, Text, Empfänger, Sendezeit und
SmsTyp, (Optional Absender)
```

3.1.8.1 Perl Sample

Bei mehreren Empfängern oder unterschiedlichen Texten siehe: Punkt 3.2

Benötigt [SOAP::Lite](#)
und [Crypt::SSLeay](#), wenn HTTPS verwendet werden soll.

```
use SOAP::Lite;
use strict;
my $s = SOAP::Lite

    -> uri('http://cetix.de/SendSMS')
#    HTTPS Verbindung
    -> proxy('http://soap.smsworkx.de/send.asmx')

    -> on_action(sub{sprintf '%s/%s', @_ })
#    -> on_debug(sub{print@_});

#Senden über die Einfach Methode

my $User = SOAP::Data->name('User' => 'test')->type('string')->
>uri('http://cetix.de/SendSMS');
my $Pw = SOAP::Data->name('Password' => 'test')->type('string')->
>uri('http://cetix.de/SendSMS');
my $Empf = SOAP::Data->name('Recipient' => '49171123457')->type('string')->
>uri('http://cetix.de/SendSMS');
my $Text = SOAP::Data->name('SMSText' => 'Hallo Welt')->type('string')->
>uri('http://cetix.de/SendSMS');

my $result = $s->SendSimpleSMS2($User,$Pw, $Empf, $Text)->result;

print "\nResult: " . $result . "\n";
```

3.1.8.2 PHP Sample

Bei mehreren Empfängern oder unterschiedlichen Texten siehe: Punkt 3.2

```
<?php
function PostToHost($host, $path, $data_to_send) {
    $fp = fsockopen($host, 80);
    fputs($fp, "POST $path HTTP/1.1\r\n");
    fputs($fp, "Host: $host\r\n");
    fputs($fp, "Content-type: application/x-www-form-urlencoded\r\n");
    fputs($fp, "Content-length: ". strlen($data_to_send) ."\r\n");
    fputs($fp, "Connection: close\r\n\r\n");
    fputs($fp, $data_to_send);
    while(!feof($fp)) {
        $res .= fgets($fp, 128);
    }
    fclose($fp);
    return $res;
}

function SendSMS($user,$pw,$jobid,$Msg,$Rcp,$MsgTyp,$Sender)
{
    //Vorbereiten der Daten
    $data = "User=$user&Password=$pw&Caption=" . rawurlencode($jobid) .
"&Sender=" . rawurlencode($Sender) . "&SMSText=" . rawurlencode($Msg) .
"&Recipient=" . rawurlencode($Rcp) . "&SmsTyp=$MsgTyp&SendDate=";

    //Senden an den Server
    $x = PostToHost(
        "soap.smsworkx.de",
        "/send.asmx/SendText",
        $data
    );

    //Auswerten des Ergebnisses

    $iStart = strpos($x,"SendSMS\>") + 9;
    $iEnde = strpos($x,">",$iStart);

    return substr($x,$iStart,$iEnde-$iStart);
}

?>

<html>
<head><title>PHP Form Post SMS Sample</title></head>
<body>

        SMS Result: <?php echo SendSMS("test","test","Caption","Hallo
PHP","01711234567",6,""); ?>

</body>
</html>
```

3.2 Versand von mehreren SMS pro Auftrag

Diese Funktionen gelten für alle oben genannten SMS Übertragungsarten.

3.2.1 Alle Empfänger selber Text

Mehrere Empfänger erhalten die gleiche SMS.

- im Empfängerfeld (Recipient) werden die einzelnen Empfänger durch ein \n (Hex 10) getrennt

Beispiel: Normale SMS

```
0171124567;FELD1;FELD2/n
```

SMS an mehrere Empfänger

```
Nummer  Feld1  Feld2
0171124567;Michael;Müller\n
0171124568;Klaus;Mustermann\n
0171124569;Helmut;Schmit
```

- SMS Text: Hier könnte Ihr Nachricht stehen ...

3.2.2 Mehrere Empfänger unterschiedlicher Text

Im Text können mehrere Felder vergeben werden (siehe Beispiel).

- Die einzelnen Empfänger werden durch ein \n (Hex10) getrennt.

Beispiel:

```
Nummer  Feld1          Feld2
0171124567;Michael Müller;45trx45\n
0171124568;Klaus;Mustermann\n
0171124569;Helmut;Schmit
```

- SMS Text

Beispiel:

Hallo #FELD1# am Do haben wir eine Party. Dein Code ist #FELD2#.

4 SMTP (e-Mail 2 SMS)

4.1 Allgemein

Sie können als SMTP Server mail.smsworkx.de angeben (Port 25).

Empfänger der SMS ist Handynummer@sms.smsworkx.de

Ausgewertet werden nur die Felder „An“ und „CC“.

4.2 Aufbau einer Email

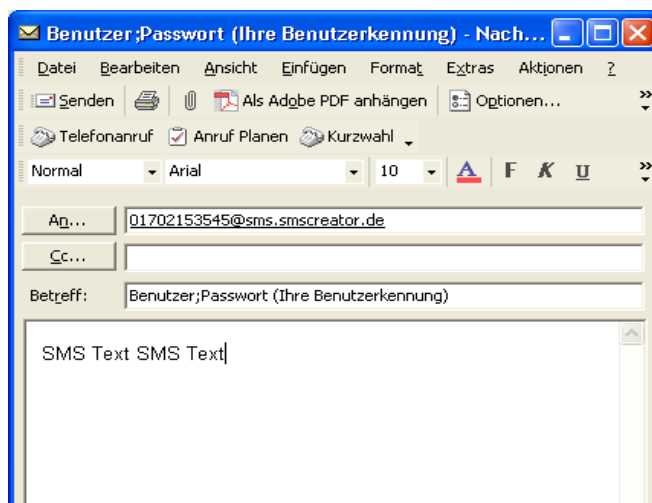
Empfänger: Handynummer@Sms.smsworkx.de
Subject: Benutzer;Passwort;SendeZeit;SenderId
Body: Text der Nachricht (die ersten 160 Zeichen werden verwendet)
Empfänger: Empfänger der SMS
Subject:
Benutzer Account Name
Passwort Account Passwort
SendeZeit Optional die Zeit wann die SMS übertragen werden soll.
Format: dd.mm.yyyy HH:MM:SS
SenderId Absender ID als Text max 11 Zeichen.
Wird diese Option gewählt können keine Rück SMS oder Nummern Überprüfung aktiviert werden

4.3 Beispiel

An: 49171123456@sms.smsworkx.de

Subject: Test;Test;01.01.2004;Party!

Body: Hallo Jens happy new year!



5 RückSMS / Delivery Reports

RückSMS bzw. Delivery Reports können auf mehrere Arten zugestellt und abgefragt werden:

- Zustellung per http-Request
- Zustellung per Email
- Per Abfrage auf Webseite

5.1 Zustellung per HTTP-Request

Bei dieser Zustellung, benötigen Sie eine öffentlich zugängliche Internetseite, auf die wir diese Nachrichten zustellen können.

Hier ein Beispiellink:

<http://gateway.test.de/netxp.php?id=#ID#&text=#MSG#&from=#MOBIL#&smstyp=#STATUS#>

5.2 Zustellung per Email

Hier benötigen wir eine Email-Adresse von Ihnen, auf die wir die RückSMS zustellen können. Delivery Reports werden nicht per Email zugestellt.

Ausgabepunkte sind hier:

Sender :
Datum :
Nachricht:

5.2.1 Alternativer Empfänger der RückSMS

Sie können in jedem Job einen alternativen Empfänger für die Rücksms angeben. In der Caption backmail:empfaenger@email.de angeben. Jede auf dieses Job Empfangene SMS wird nun an die angegebene Adresse gesendet.

5.3 Per Abfrage

5.3.1 Abfrage auf einer Webseite

Mit dem Link: <http://soap.smsworkx.de/smscrecived/checkdata.aspx>

Können Sie eine Internetseite aufrufen und mit Hilfe Ihrer Seriennummer und des Passwortes, alle Rücksms bzw. delivery Reports abrufen.

Abbrufkriterien sind hier:

Datum
Format (csv oder HTML Tabelle)
Jobld
Nur Statusberichte (Alle, Nur Erfolgreiche oder Nur Fehler)

5.3.2 Received SMS

Liefert alle Nachrichten ab einem Zeitpunkt

User Benutzerkennung
Password Passwort
strLastDate von Datum

Ergebnis: CSV String

Aufbau:

Sender;Datum;Nachricht;NewLineZeichen

5.3.2.1 SOAP

Hier ist ein SOAP Beispiel, dieses enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx HTTP/1.1
Host: soap.smsworkx.de
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://cetix.de/SendSMS/RecivedSMS"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schem-
as.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RecivedSMS xmlns="http://cetix.de/SendSMS">
      <User>string</User>
      <Password>string</Password>
      <strLastDate>string</strLastDate>
    </RecivedSMS>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schem-
as.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RecivedSMSResponse xmlns="http://cetix.de/SendSMS">
      <RecivedSMSResult>string</RecivedSMSResult>
    </RecivedSMSResponse>
  </soap:Body>
</soap:Envelope>
```

5.3.2.2 HTTP GET

Hier ist ein HTTP GET Beispiel, dieses enthält Aufruf und Antwort. Die gezeigten Platzhalter müssen mit tatsächlichen Werten ersetzt werden.

```
GET /send.asmx/RecivedSMS?User=string&Password=string&strLastDate=string
HTTP/1.1
Host: soap.smsworkx.de
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

5.3.2.3 HTTP POST

Hier ist ein HTTP POST Beispiel, dieses enthält Aufruf und Antwort. Die gezeigten Platzhalter müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx/RecivedSMS HTTP/1.1
Host: soap.smsworkx.de
Content-Type: application/x-www-form-urlencoded
Content-Length: length

User=string&Password=string&strLastDate=string
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

6 Statistik

6.1 Account-Abfrage

6.1.1 QueryBalance

Liefert den aktuellen Account. Es werden nur Haupt-Accounts zurückgegeben.

6.1.1.1 SOAP

Hier ist ein SOAP Beispiel, dieses enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx HTTP/1.1
Host: soap.smsworkx.de
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://cetix.de/SendSMS/QueryBalance"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schem-
as.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <QueryBalance xmlns="http://cetix.de/SendSMS">
      <User>string</User>
      <Password>string</Password>
    </QueryBalance>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schem-
as.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <QueryBalanceResponse xmlns="http://cetix.de/SendSMS">
      <QueryBalanceResult>string</QueryBalanceResult>
    </QueryBalanceResponse>
  </soap:Body>
</soap:Envelope>
```

6.1.1.2 HTTP GET

Hier ist ein HTTP GET Beispiel, dieses enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden.

```
GET /send.asmx/QueryBalance?User=string&Password=string HTTP/1.1
Host: soap.smsworkx.de
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

6.1.1.3 HTTP POST

Hier ist ein HTTP POST Beispiel, dieses enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx/QueryBalance HTTP/1.1
Host: soap.smsworkx.de
Content-Type: application/x-www-form-urlencoded
Content-Length: length

User=string&Password=string
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

7 Send MMS

7.1.1 Aufbau

Bitte <http://soap.smsworkx.de/send.asmx> öffnen und die Felder zum Testen füllen.

User	String	Benutzername
Password	String	Passwort
Data	String	Aufbau als XML Struktur

```
<MMS>
<SUBJECT>Titel der MMS</SUBJECT>
<IMAGE1>Link auf das Bild</IMAGE1>
<IMAGEDATA1>Base64 Kodiertes
Bild</IMAGEDATA1>
<TEXT1>Text der MMS</TEXT1>
<IMAGE2> Anderes Bild </IMAGE2>
<SOUND1>Link auf den Sound </SOUND1>
alt. <SOUNDDATA1> Base64 Kodiertes Sound im
</SOUNDDATA1>
<TEXT2>Text für 2 Bild</TEXT2>
<SMIL> Um eine SMS Speziell zu formatieren
kann eine SMIL Datei angegeben werden.
Infos unter http://www.w3.org/TR/REC-smil
</SMIL>
</MMS>
```

Es können n Bilder und Text Elemente vorkommen.
Werden Links angegeben, müssen diese zum
Übertragungszeitpunkt erreichbar sein.

Bilder in den Formaten: JPG,GIF,PNG
Größe 160 x 120 Pixel. Sollte ein Bild größer sein, wird
automatisch eine Verkleinerung auf diese Werte
durchgeführt.
Sounds im Format **amr**

(Um wav/mp3-dateien in amr-Files zu konvertieren
empfehlen wir von Nokia (<http://www.forum.nokia.com>) den
Nokia Multimedia Converter 2.0 Beta 2. Downloadbar, nach
kostenloser Registrierung unter:

<http://www.forum.nokia.com/main/1,6566,030,00.html?fsrPar am=2-3-/main/1,6566,030,00.html&fileID=2998>)

Beispiel:

```
<MMS>
<SUBJECT>Michael</SUBJECT>
<IMAGE1>http://www.smsworkx.de/mms/image1.jpg
</IMAGE1>
<TEXT1>Alle Gute zum Geburtstag</TEXT1>
</MMS>
```

SendDate	String	Sendezeit: Wenn keine Sendezeit angegeben wird, wird die MMS sofort versendet.
----------	--------	---

7.1.1.1 SOAP

Hier ein SOAP Beispiel. Es enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx HTTP/1.1
Host: soap.smsworkx.de
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://cetix.de/SendSMS/SendMMS"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SendMMS xmlns="http://cetix.de/SendSMS">
      <User>string</User>
      <Password>string</Password>
      <Recipient>string</Recipient>
      <Sender>string</Sender>
      <Data>string</Data>
      <SendDate>string</SendDate>
    </SendMMS>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SendMMSResponse xmlns="http://cetix.de/SendSMS">
      <SendMMSResult>string</SendMMSResult>
    </SendMMSResponse>
  </soap:Body>
</soap:Envelope>
```

7.1.1.2 HTTP GET

Hier ein HTTP GET Beispiel. Es enthält Aufruf und Antwort. Die gezeigten [Platzhalter](#) müssen mit tatsächlichen Werten ersetzt werden.

```
GET /send.asmx/SendMMS?User=string&Password=string&Recipient=string&Sender=string&Data=string&SendDate=string HTTP/1.1
Host: soap.smsworkx.de
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

7.1.1.3 HTTP POST

Hier ein HTTP POST Beispiel. Es enthält Aufruf und Antwort. Die gezeigten **Platzhalter** müssen mit tatsächlichen Werten ersetzt werden.

```
POST /send.asmx/SendMMS HTTP/1.1
Host: soap.smsworkx.de
Content-Type: application/x-www-form-urlencoded
Content-Length: length

User=string&Password=string&Recipient=string&Sender=string&Data=string&SendDate=string
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://cetix.de/SendSMS">string</string>
```

8 Glossar

AMR	Abkürzung für „Audio Modem Riser“
BMP	Bitmap Format - zur verlustfreien Speicherung von Rastergrafiken
EMS	Enhanced Message Service – Dienst für die Kommunikation zwischen Mobiltelefonen (es können damit auch Logos, Bilder und Klingeltöne versendet werden). EMS ist nicht auf 160 Zeichen beschränkt.
Flash SMS	“blinkende SMS” – das Handydisplay blinkt beim Erhalt dieses SMS-Typen. Die SMS kann nur gelesen werden. Das Speichern der SMS ist nicht möglich.
FTP	File Transfer Protocol – Netzwerkprotokoll zur Übertragung von Daten
GIF	Graphics Interchange Format – digitales Bildformat mit guter verlustfreier Komprimierung für Bilder mit geringer Farbtiefe
High Quality Routen	SMS werden direkt an die jeweiligen SMSCs gesendet
HTTP	Hypertext Transfer Protocol – Protokoll zur Übertragung von Daten
JPG	Joint Photographic Experts Group – verlustbehaftetes Kompressionsverfahren für digitale Bilder
MID	MIDI - Musical Instrument Digital Interface
NOL	Operator Logo Dateien (Hersteller Logos)
PNG	Portable Network Graphics – Dateiformat zur Speicherung von Bilddaten
RTTL	Ring Tone Transfer Language
SOAP	Protokoll, mit dem man Daten zwischen Systemen austauschen kann und Remote Procedure Calls durchgeführt werden können. SOAP verwendet zur Darstellung der Daten XML, zur Übertragung der Nachrichten Internet-Protokolle wie z.B. TCP und HTTP
SMPP	Short Message Peer to Peer Protocol
SMS ohne Absenderkennung	SMS werden per Modem versendet. Im Absender steht eine festgelegte Handynummer. Diese Nummer wird für alle weiteren SMS als Absendernummer verwendet.
SMSC	Short Message Service Center (der Mobilfunk-Anbieter)
SMTP	Simple Mail Transfer Protocol – Protokoll zur Versendung von E-Mails
TCP/IP	Transmission Control Protocol / Internet Protocol
UCP	Universal Computer Protocol
UDH	User Data Handy

VCARD 1 / 2	elektronische Visitenkarten für Mobilfunkgeräte (wird nur von bestimmten Herstellern unterstützt)
WSDL	Web Services Discription Language – plattform-, programmiersprachen- und protkollunabhängiger XML-Standard zur Beschreibung von Netzwerkdiensten zum Austausch von Nachrichten
XML	Extensible Markup Language - Standard zur Erstellung strukturierter, maschinen- und menschen lesbarer Dateien

9 Sonstiges

Sollten Sie dennoch Probleme, Fragen und/oder Anregungen haben, können Sie uns diese gerne per e-Mail an: Support@SMSworkx.de senden.

Wir bedanken uns für den Erwerb unseres Produktes.

Ihr smsworkx-Team

