



## Dokumentacja smsAPI wersja 2.9



## SPIS TREŚCI

Historia zmian	3
1. Wprowadzenie	4
1.1. Rozpoczęcie współpracy	4
1.2. Filtr IP dla interfejsu API	4

### I. WYSYŁKA SMS

2. Wysyłanie pojedynczych SMS'ów	4
2.1. Wysyłanie ecoSMS	5
2.2. Wysyłanie fastSMS	6
2.3. Wysyłanie SMSa o zaplanowanej godzinie/dacie	7
2.3.1 Zmiana czasu na Timestamp	7
2.4. Usuwanie zaplanowanej wiadomości	7
3. Wysyłanie masowe SMS'ów	8
3.1. Wysyłanie masowe spersonalizowanych wiadomości z wykorzystaniem parametrów oraz z wykorzystaniem parametru IDX	8
4. Wysyłanie wiadomości z wykorzystaniem szablonów	9
5. Wysyłanie WAP PUSH	10
5.1. Konwerter WAP Push	10
6. Wysyłanie vCard	10
6.1. Konwerter vCard	11
7. mail2SMS – Wysyłanie SMS za pomocą e-mail	12
8. Sprawdzenie ilości punktów na koncie	12

### II. SPRAWDZANIE RAPORTÓW DORĘCZENIA WIADOMOŚCI SMS

9. Odbieranie raportu doręczenia wiadomości - CALLBACK	12
9.1. Sprawdzenie statusu wiadomości – w przypadku nie odebrania przez wywołanie zwrotne CALLBACK	13

### III. WYSYŁKA MMS

10. Wysyłanie MMS	14
10.1. Odbieranie raporty wiadomości MMS - CALLBACK	14

### IV. ODBIÓR WIADOMOŚCI SMS ORAZ MMS

11. Odbiór SMS/MMS	15
11.1. Odbiór SMS	15
11.2. Odbiór MMS	15

### V. KORZYSTANIE Z SERWISU ZA POMOCĄ SOAP

12. Korzystanie z serwisu smsAPI.pl przy wykorzystaniu <b>WebServices</b> (SOAP)	16
12.1. Lista funkcji	16

12.2. Lista parametrów	17
12.3. Wysyłanie pojedynczej wiadomości	18
12.4. Wysyłanie wiadomości masowych	19
12.5. Usuwanie wiadomości zaplanowanych	19
12.6. Sprawdzanie ilości punktów	19
12.7. Sprawdzanie raportów doręczenia SMS po ID wiadomości	20
12.8. Sprawdzanie raportów doręczenia SMS po dacie wysłania wiadomości	20
12.9. Dodawanie numeru do książki telefonicznej	21
12.10. Usuwanie numeru z książki telefonicznej	21
12.11. Pobieranie numerów z książki telefonicznej użytkownika	21
12.12. Dodawanie grupy do książki telefonicznej	22
12.13. Usuwanie grupy z książki telefonicznej	22
12.14. Pobieranie listy grup z książki telefonicznej użytkownika	22
12.15. Pobieranie dostępnych pól nadawcy	23

## VI. PODSUMOWANIE

13. Uwagi końcowe	23
Dodatek nr 1 - Lista raportów doręczeń wiadomości	24
Dodatek nr 2 - Kody błędów	24
Dodatek nr 3 - Kodowanie	25
Dodatek nr 4 - Przykładowe skrypty	25

## Historia zmian

Wersja	Data	Zmiany
Wersja 2.9	08.07.2010	1. Rozszerzenie funkcji sprawdzania ilości punktów w serwisie smsAPI.pl o pokazywanie ilości dostępnych SMS.
Wersja 2.8	05.05.2010	1. Zmiana w zwracaniu wyniku podczas wysyłania wiadomości masowych korzystając z metody SOAP
Wersja 2.7	18.03.2010	1. Opis parametru priority=1
Wersja 2.6	24.02.2010	1. Filtr IP dla interfejsu API
Wersja 2.5	22.01.2010	1. Wysyłanie wiadomości MMS
Wersja 2.4	11.11.2009	1. Odbieranie wiadomości MMS
Wersja 2.3	28.08.2009	1. Opis korzystania z serwisu smsAPI.pl za pomocą WebServices
Wersja 2.2	10.07.2009	1. Parametr nunicode 2. Parametr single=1 3. Sprawdzanie statusu wiadomości – kilka statusów za pomocą jednego zapytania
Wersja 2.1		1. Wysyłanie ecoSMS poprzez MailtoSMS 2. Zmiana odbioru raportów doręczenia poprzez CALLBACK
Wersja 2.0		1. Wysyłanie ecoSMS
Wersja 1.9		1. Wysyłanie wiadomości WAP PUSH

		2. Wysyłka z wykorzystaniem parametru IDX
Wersja 1.8		1. Wysyłanie z wykorzystaniem szablonów 2. Wysyłka masowa ze spersonalizowaną treścią
Wersja 1.7		1. Wysyłka o określonej dacie/godzinie

## 1. Wprowadzenie

Platforma smsAPI została skierowana do użytkowników chcących rozbudować swoje aplikacje o system wysyłania SMSów. Aplikacja ta w prosty sposób umożliwia integrację dowolnego serwisu z bramką SMS. Głównymi atutami naszego serwisu oprócz prostej implementacji jest możliwość nadawania wiadomości z własnego nr telefonu (dowolny polski operator) lub z własnej nazwy (maksymalnie 11 znaków). Każdy wysłany SMS za pośrednictwem systemu posiada unikalny numer identyfikacyjny pozwalając na sprawdzenie raportu doręczenia wiadomości.

### 1.1 Rozpoczęcie współpracy

W celu rozpoczęcia współpracy należy utworzyć konto w serwisie smsAPI.pl

**Adres URL:** <http://www.smsapi.pl/register.html>

Utworzone konto jest gotowe do użytku, jednak zalecamy weryfikację numeru nadawcy lub ustawienia własnej nazwy. Konta bez weryfikacji numeru domyślnie posiadają jako nadawcę wiadomości nazwę serwisu „smsAPI.pl”. Weryfikacja własnego numeru obciąża jednorazowo konto jak za wysyłkę jednego proSMS'a, natomiast ustawienie nazwy nadawcy jest usługą całkowicie darmową.

### 1.2 Filtr IP dla interfejsu API

W celu dodatkowego zabezpieczenie interfejsu API w zakładce „Ustawienia” → „Moje konto” w polu **Filtr IP dla interfejsu API** ustawić można adresy IP z których możliwa będzie wysyłka wiadomości (w przypadku próby dokonania wysyłki z innego IP system zwróci błąd: ERROR:105). Adresy należy oddzielić średnikami.

## 2. Wysyłanie pojedynczych SMS'ów

Adresy URL do połączenia z aplikacją – zwane dalej „Adresem połączenia”:

- <http://api.smsapi.pl/send.do> - dla połączeń standardowych
- <https://ssl.smsapi.pl/send.do> - dla połączeń szyfrowanych SSL 128bit v3.0
- <http://smsapi.pl/webservices> – dla połączenia za pomocą WebServices (SOAP)

Wysyłanie SMSów odbywa się przez wysłanie metodą **GET** lub **POST** danych do adresu połączenia:

Parametr	Opis
<i>username</i> *	Nazwa użytkownika w serwisie smsAPI
<i>password</i> *	Hasło do Twojego konta w serwisie smsAPI zaszyfrowane w MD5
<i>to</i> *	Numer odbiorcy wiadomości w formacie 48xxxxxxxx lub xxxxxxxx. Np. 48505602702 lub 505602702.
<i>message</i> *	Treść wiadomości. Standardowo do 160 znaków lub 70 znaków w przypadku wystąpienia chociaż jednego znaku specjalnego (polskie znaki uważane są za specjalne). Maksymalna długość wiadomości wynosi 457 znaków (lub 201 ze znakami specjalnymi) i jest wysłana jako 3 połączone SMSy, obciążając konto zgodnie z aktualnym cennikiem. Więcej szczegółów odnośnie znaków specjalnych znajduje się w pkt 13.
<i>from</i>	Numer lub nazwa nadawcy wiadomości. Pozostawienie pola pustego powoduje wysłanie wiadomości od „smsAPI.pl”. Przyjmowane są tylko numery i nazwy zweryfikowane. (&from=aktywna_nazwa). Pole nadawcy należy dodać po

	zalogowaniu na stronie smsAPI.pl, w zakładce Ustawienia → Pola nadawcy.
<i>encoding</i>	Parametr określa kodowanie polskich znaków w SMSie. Domyślne kodowanie jest windows-1250. Jeżeli występuje konieczność zmiany kodowania, należy użyć parametru encoding z danymi: <ul style="list-style-type: none"> <li>- dla iso-8859-2 (latin2) – należy podać wartość „iso-8859-2”,</li> <li>- dla utf-8 – należy podać wartość „utf-8”.</li> </ul>
<i>flash</i>	Wysyłanie wiadomości trybem „flash”, odbywa się poprzez podanie parametru flash o wartości „1”. SMSy flash są automatycznie wyświetlane na ekranie głównym telefonu komórkowego i nie są przechowywane w skrzynce odbiorczej (jeśli nie zostaną zapisane). (&flash=1)
<i>test</i>	Wiadomość nie jest wysyłana, wyświetlana jest jedynie odpowiedź (w celach testowych). (&test=1)
<i>details</i>	W odpowiedzi zawarte jest więcej szczegółów. (Treść wiadomości, długość wiadomości, ilość części z jakich składa się wiadomość). (&details=1)
<i>date</i>	Data w formacie timestamp. Określa kiedy wiadomość ma być wysłana. (&date=1237383500). W przypadku wstawienia daty przeszłej wiadomość zostanie wysłana od razu.
<i>datacoding</i>	Parametr pozwalający na wysyłanie wiadomości WAP PUSH. (&datacoding=bin)
<i>idx</i>	Opcjonalny parametr użytkownika wysyłany z wiadomością a następnie zwracany przy wywołaniu zwrotnym CALLBACK. (&idx=123)
<i>single</i>	Ustawienie 1 zabezpiecza przed wysłaniem wiadomości składających się z kilku części. ( <b>ERROR:12</b> )
<i>eco</i>	Ustawienie parametru &eco=1 spowoduje wysłanie wiadomości przy wykorzystaniu ecoSMS (brak możliwości wyboru pola nadawcy, wiadomość wysyłana z losowego numeru dziesięciocyfrowego) szczegóły dotyczące ecoSMS znajdują się na naszej stronie: <a href="http://www.smsapi.pl/">http://www.smsapi.pl/</a>
<i>nounicode</i>	Ustawienie 1 zabezpiecza przed wysłaniem wiadomości ze znakami specjalnymi (w tym polskimi) ( <b>ERROR:11</b> ).
<i>priority</i>	Ustawienie 1 spowoduje wysłanie wiadomości przy wykorzystaniu osobnego kanału zapewniającego szybkie doręczenie wiadomości. Z parametru korzystać można wyłącznie podczas wysyłania wiadomości proSMS, Ilość punktów za wysyłkę pomnożona będzie przez 1.5 <b>Uwaga!</b> Dla tego parametru zabronione jest prowadzenie wysyłek masowych i marketingowych.

\* - pole obowiązkowe

Zapytanie: **[http://api.smsapi.pl/send.do?](http://api.smsapi.pl/send.do?username=uzytkownik&password=haslo&from=nazwa&to=48500500500&message=test%20wiadomosci)  
**username=uzytkownik&password=haslo&from=nazwa  
&to=48500500500&message=test wiadomosci****

Odpowiedź: **OK:<ID>:<POINTS>**

lub w razie błędu

**ERROR:<ERR>**

**<ID>** numer ID wysłanego SMSa. Numer należy zachować w celu sprawdzenia statusu wiadomości.

**<POINTS>** ilość pobranych punktów (np. SMS składający się z 3 części zwróci wartość 3).

**<ERR>** kod błędu (zobacz tabela kodów błędów w punkcie 15)

Przykład: **OK:17101000090360359:0.135**

lub

**ERROR:102**

## 2.1 Wysłanie wiadomości ecoSMS

Wysyłanie wiadomości ecoSMS odbywa się z losowego numeru (brak możliwości wyboru pola nadawcy). W celu wysłania wiadomości ecoSMS należy użyć parametr `&eco=1` (również przy wysyłaniu wiadomości masowych).

Zapytanie: **http://api.smsapi.pl/send.do?  
username=uzytkownik&password=haslo&to=48500500500&eco=1&message=test wiadomosci eco**

Odpowiedź: **OK:<ID>:<POINTS>**

lub w razie błędu

**ERROR:<ERR>**

**<ID>** numer ID wysłanego SMSa. Numer należy zachować w celu sprawdzenia statusu wiadomości.

**<POINTS>** ilość pobranych punktów (zobacz tabelę poniżej).

**<ERR>** kod błędu (zobacz tabela kodów błędu)

Przykład: **OK:17101000090360359:0.06**

lub

**ERROR:102**

## 2.2 Wysłanie wiadomości fastSMS

Ustawienie parametru `priority=1` spowoduje wysłanie wiadomości przy wykorzystaniu osobnego kanału zapewniającego szybkie doręczenie wiadomości. Z parametru korzystać można wyłącznie podczas wysyłania wiadomości proSMS, Ilość punktów za wysyłkę pomnożona będzie przez 1.5

Zapytanie: **http://api.smsapi.pl/send.do?  
username=uzytkownik&password=haslo&to=48500500500&priority=1&message=test wiadomosci fastSMS**

Odpowiedź: **OK:<ID>:<POINTS>**

lub w razie błędu

**ERROR:<ERR>**

**<ID>** numer ID wysłanego SMSa. Numer należy zachować w celu sprawdzenia statusu wiadomości.

**<POINTS>** ilość pobranych punktów (zobacz tabelę poniżej).

**<ERR>** kod błędu (zobacz tabela kodów błędu)

Przykład: **OK:17101000090360359:0.2025**

lub

**ERROR:102**

**UWAGA!** Dla tego parametru zabronione jest prowadzenie wysyłek masowych i marketingowych. W przypadku wysyłki do więcej niż jednego numeru w jednym odwołaniu wiadomości zostaną wysłane jako proSMS

## 2.3 Wysyłanie SMSa o zaplanowanej godzinie/dacie

W celu wysłania wiadomości o określonej godzinie/dacie musi zostać użyty parametr **&date=** wskazujący datę którą należy skonwertować do postaci **timestamp**. Do konwersji wykorzystać można narzędzie znajdujące się na stronie <http://www.smsapi.pl/generators.html>. Zakładka "Narzędzia" → "Generatory" → "Timestamp".

Zapytanie: **http://api.smsapi.pl/send.do?username=uzytkownik&password=haslo&to=48500500500&date=1577878200&message=wiadomosc zaplanowana**

Odpowiedź: **OK:<ID>:<POINTS>**

lub w razie błędu

**ERROR:<ERR>**

### 2.3.1 Zmiana czasu na Timestamp

Poniżej przedstawione jest źródło przykładowego skryptu zamieniającego czas na timestamp.

```
<?php
// Data przekazywana jest do skryptu metodą GET jako 'data' w formacie DD.MM.RRRR hh:mm:ss
if ($_GET['data']) {
    $timestamp = strtotime($_GET['data']);
    if ($timestamp > 0){
        if ($timestamp > time()){
            echo"Timestamp z daty (".date('d.m.Y H:i:s',$timestamp).") to $timestamp";
        }
        else echo "<strong>Data przeszła!</strong>";
    }
    else echo "<strong>Niepoprawny format daty!</strong>";
}
?>
```

## 2.4. Usuwanie zaplanowanej wiadomości

Parametr	Opis
<i>username</i> *	Nazwa użytkownika w serwisie smsAPI
<i>password</i> *	Hasło do Twojego konta w serwisie smsAPI zaszyfrowane w MD5
<i>sch_del</i> *	ID zaplanowanej wiadomości (zwrócony podczas wysyłki).

\* - pole obowiązkowe

Zapytanie: **http://api.smsapi.pl/send.do?username=uzytkownik&password=haslo&sch\_del=09040616088106874**

Odpowiedź: **OK**

lub gdy ID wiadomości nie istnieje

**ERROR:301**

Przykład ID: 09040616088106874

### 3. Wysłanie masowe SMS'ów

Wysyłanie SMSów do grupy odbiorców odbywa się tak jak do pojedynczego odbiorcy (opis w punkcie 2) lecz z podaniem wielu numerów telefonów w parametrze „to”.

Dodatkowo cała wysyłka powinna odbywać się metodą **POST** do adresu połączenia. W przypadku wysyłki metodą **GET** przy większej ilości numerów, część parametrów może zostać ucięta i w efekcie niedostarczona w całości do serwisu smsAPI.

Jeżeli cena za wysłanie wszystkich SMSów jest większa od ilości punktów dostępnych w serwisie, zostanie zwrócony komunikat o błędzie (**103**) i SMSy nie zostaną wysłane. Jeżeli zostały podane błędne numery (nierozpoznane przez serwis smsAPI ze względu na błędny prefiks), wszystkie SMSy zostaną wysłane z pominięciem tych numerów, wówczas ilość otrzymanych raportów będzie różnić się od ilości wysłanych SMSów. Sytuacja ta nie dotyczy numerów z poprawnym prefiksem (np. 500000000), gdzie SMS zostanie wysłany, a następnie odrzucony gdyż numer nie istnieje (punkty za wysłanie SMSa zostaną pobrane). Numer, które się powtarzają zostaną wysłane tylko 1 raz.

Zapytanie: **http://api.smsapi.pl/send.do?username=uzytkownik&password=haslo&from=nazwa&to=48500500500,48501501501,48502502502&message=test wiadomosci**

Odpowiedź: **OK:<ID>:<POINTS>:<PHONE>;...;...;...**

**<ID>** numer ID wysłanego SMSa. Numer należy zachować w celu sprawdzenia statusu wiadomości.

**<POINTS>** ilość pobranych punktów (np. sms wysłany w 3 wiadomościach zwróci wartość 3).

**<PHONE>** numer telefonu odbiorcy.

Przykład:

**OK:17101000090360359:0.13:500500500;OK:17101000090360359:0.13:501500501;  
OK:17101000090360359:0.13:502502502;**

Warto zauważyć, że przy wysyłce SMSów do grupy odbiorców dodatkowo zwracany jest numer telefonu oraz każdy SMS jest zakańczany średnikiem (również ostatni).

Zalecana maksymalna ilość jednorazowej wysyłki (jedno wywołanie) wynosi **1.000** wiadomości metodą **POST** oraz do **200** wiadomości metodą **GET**.

#### 3.1. Wysłanie masowe spersonalizowanych wiadomości z wykorzystaniem parametrów oraz z wykorzystaniem parametru IDX

Istnieje możliwość wysłania do 100 spersonalizowanych wiadomości przy pomocy jednego wywołania wykorzystując parametry. W razie potrzeby wysłania większej ilości wiadomości można wywołać równoległe kilka odwołań.

Parametry powinny być zdefiniowane w wywołaniu jako **param1,param2,param3, param4**, które zastępowały będą [%1%], [%2%], [%3%] oraz [%4%] w treści wiadomości. Wartości parametrów muszą być oddzielone znakiem pipe „|” według wzoru:

**param1=Ania|Michał|Andrzej&param2=Nowak|Kowalski|Nowakowski**

Liczba parametrów musi być identyczna z liczbą numerów do których wysłane mają być wiadomości w innym przypadku zwrócony będzie błąd: ERROR:18 i wiadomość nie zostanie wysłana. **WAŻNE!** Długość wiadomości może być różna w zależności od długości parametru!

Jeżeli jeden z numerów będzie błędny zostanie on pominięty i wiadomości zostaną wysłane z pominięciem tego numeru.

## Parametry

Parametry wykorzystane mogą być po zdefiniowaniu ich miejsca w wiadomości:

<b>[%1%]</b>	Tekst parametru 1 (param1)
<b>[%2%]</b>	Tekst parametru 2 (param2)
<b>[%3%]</b>	Tekst parametru 3 (param3)
<b>[%4%]</b>	Tekst parametru 4 (param4)

### Przykład:

```
http://api.smsapi.pl/send.do?
username=login&password=haslo_md5&from=nadawca&to=48600111222,
48500111222&mess_age=Test wiadomosci, parametr1: [%1%] parametr2: [%2%]
&param1=Jan|Ania&param2=30|40
```

Wiadomości będą miały następującą treść:

Wiadomość 1 : Test wiadomosci, parametr1: **Jan** parametr2: **30**

Wiadomość 2 : Test wiadomosci, parametr1: **Ania** parametr2: **40**

Istnieje możliwość wysyłania masowych wiadomości z parametrem użytkownika **IDX** różnym dla każdej wiadomości. Parametr ten zwracany jest w wywołaniu zwrotnym CALLBACK. W celu przypisania różnych parametrów różnym wiadomością należy oddzielić je znakiem "pipe" co przedstawia poniższy przykład:

**&idx=idx1|idx2|idx3|idx4**

Liczba parametrów **IDX** musi być identyczna z liczbą wysyłanych wiadomości.

## 4. Wysyłanie wiadomości z wykorzystaniem szablonów

Przy wykorzystaniu szablonów w bardzo prosty sposób zmienić można treść SMSów powiadamiających (w sklepach, serwisach internetowych, klinikach itp.) bez ingerencji w kod skryptu odpowiedzialnego za wysyłanie wiadomości SMS.

W celu skorzystania z szablonów należy:

- Po zalogowaniu na stronie <http://www.smsapi.pl> dodać szablon w zakładce „Szablony”
- W miejsce, w którym ma występować parametr należy wpisać [%N%] gdzie N to liczba od 1 do 4
- W celu wykorzystania szablonu w interfejsie API należy wybrać jeden z szablonów poprzez wpisanie parametru &templates=nazwa\_szablonu
- Oprócz podstawowych parametrów w trakcie używania szablonów dostępne są parametry:

Parametr	Opis
<i>template</i>	Nazwa szablonu
<i>paramN</i>	Parametr wstawiany w miejsce [%N%] w szablonie gdzie N to liczba od 1 do 4
<i>single</i>	Jeżeli wiadomość będzie składała się z więcej niż 160 znaków nie zostanie wysłana i zwrócony zostanie ERROR:12 (&single=1)

### Przykład:

Nazwa szablonu: **Powiadom**

Treść szablonu: Witaj [%1%], Twoje zamówienie zostało zrealizowane. Numer listu przewozowego to [%2%] sledzenie przesyłek na stronie.

```
http://api.smsapi.pl/send.do?
username=login&password=haslo_md5&from=nadawca&to=48600111222
&template=powiadom&param1=Marcin&param2=BG12344423
```

Treść wysłanej wiadomości:

Witaj **Marcin**, Twoje zamówienie zostało zrealizowane. Numer listu przewozowego to **BG12344423** sledzenie przesyłek na stronie.

## 5. Wysyłanie WAP PUSH

W celu wysłania wiadomości WAP PUSH link oraz treść wiadomości muszą być przekonwertowane do postaci binarnej. Do tego celu służy konwerter do którego link znajduje się na stronie <http://www.smsapi.pl/generators.html>. Zakładka "Narzędzia" → "Generatory" → "WAP Push".

Zwrócony wynik konwersji należy wkleić jako treść wiadomości. W wywołaniu dodatkowo należy użyć parametrów **&datacoding=bin** oraz **&udh=0605040b8423f0**. Poniżej przedstawiony został przykład wysłania wiadomości WAP Push.

### Przykład:

<http://api.smsapi.pl/send.do?>

```
username=uzytkownik&password=haslo&from=nazwa&to=48501502503&datacoding=bin&udh=0605040b8423f0&message=0605040b8423f0860601ae02056a0045c60c03687474703a2f2f777772e736d736170692e706c000701034e61737a61207374726f6e613a000101
```

### 5.1 Konwerter WAP Push

Poniżej przedstawione jest źródło przykładowego skryptu konwertującego wiadomość WAP Push do postaci binarnej.

```
<?php
//Adres URL oraz treść wiadomości podawane są do skryptu metodą GET jak 'url_wap' oraz 'mess'

$udh = "0605040b8423f0";
$pre = "860601ae02056a0045c60c03";
$mid = "00070103";
$end = "000101";

if ($_GET['url_wap'] || $_GET['mess']){
    $url_wap = str_replace("http://", "", $_GET['url_wap']);
    $url_wap = AsciiToHex($_GET['url_wap']);
    $mess = AsciiToHex($_GET['mess']);
    $message = $pre.$url_wap.$mid.$mess.$end;
    echo"Treść wiadomości:<br />
    <textarea cols=\"35\" rows=\"5\" name=\"gate\">$message</textarea>";
}

function AsciiToHex ($ascii) {
    $hexadecimal = "";
    for ($i = 0; $i < strlen($ascii); $i++) {
        $byte = strtoupper(dechex(ord($ascii{$i})));
        $byte = str_repeat('0', 2 - strlen($byte)).$byte;
        $hexadecimal.=$byte;
    }
    return $hexadecimal;
}
?>
```

## 6. Wysyłanie vCard

W celu wysłania wiadomości vCard dane kontaktu muszą być przekonwertowane do postaci binarnej. Do tego celu służy konwerter do którego link znajduje się na stronie <http://www.smsapi.pl/generators.html>. Zakładka "Narzędzia" → "Generatory" → "vCard".

Zwrócony wynik konwersji należy wkleić jako treść wiadomości. W wywołaniu dodatkowo należy użyć parametrów **&datacoding=bin** oraz **&udh=06050423F4000**. Poniżej przedstawiony został przykład wysłania wiadomości vCard.

Konwersja danych wizytówki do postaci binarnej:

**Przykład:**

<http://api.smsapi.pl/send.do?>

[username=uzytownik&password=haslo&from=nazwa&to=48501502503&datacoding=bin&udh=06050423F4000&message=](http://api.smsapi.pl/send.do?username=uzytownik&password=haslo&from=nazwa&to=48501502503&datacoding=bin&udh=06050423F4000&message=)

## 6.1 Konwerter vCard

Poniżej przedstawione jest źródło przykładowego skryptu konwertującego wizytówkę vCard do postaci binarnej.

```
<?php
//Dane wizytówki podawane są do skryptu metodą GET jako 'name'-imię, 'surname'-nazwisko, 'cell'-telefon
komórkowy,'www'-adres strony internetowej, 'mail'-adres e-mail
if ($_GET['name'] || $_GET['surname'] || $_GET['cell'] || $_GET['www'] || $_GET['mail']){
    $name = $_GET['name'];
    $surname = $_GET['surname'];
    $cell = $_GET['cell'];
    $www = $_GET['www'];
    $mail = $_GET['mail'];
    $message = "BEGIN:VCARD\r\nVERSION:2.1\r\n";
    if ($name && $surname) {
        $message .= "FN:$name $surname\r\nN:$surname;$name;;;,\r\n";
    }
    else {
        if ($name) {
            $message .= "FN:$name\r\nN:$name;;;,\r\n";
        }
        else {
            if ($surname) {
                $message .= "FN:$surname\r\nN:$surname;;;,\r\n";
            }
        }
    }
    if ($cell) {
        $message .= "TEL;PREF;CELL:$cell\r\n";
    }
    if ($mail) {
        $message .= "EMAIL;INTERNET:$mail\r\n";
    }
    if ($www) {
        $message .= "URL:$www\r\n";
    }
    $message .= "END:VCARD";
    $message = AsciiToHex($message);
    echo"Treść wiadomości:<br />
    <textarea cols="35" rows="5" name="gate">$message</textarea>";
}

function AsciiToHex ($ascii) {
    $hexadecimal = "";
    for ($i = 0; $i < strlen($ascii); $i++) {
        $byte = strtoupper(dechex(ord($ascii{$i})));
        $byte = str_repeat('0', 2 - strlen($byte)).$byte;
        $hexadecimal.=$byte;
    }
    return $hexadecimal;
}
?>
```

## 7. mail2SMS – Wysyłanie SMS za pomocą e-mail

Aby wysłać smsa za pomocą maila należy wysłać maila według schematu:

ADRES: **send.do@smsapi.pl**

TEMAT: **login@haslo\_32znaki\_w\_md5**

TREŚĆ: **from=nadawca&to=numer&message=tresc wiadomosci**

**UWAGA!** Hasło należy podać w formie zaszyfrowanej algorytmem MD5!

Przykład:

ADRES: **send.do@smsapi.pl**

TEMAT: **login@8456fkty567gb3bg37b357b3457b3457**

TREŚĆ: **from=nadawca&to=numer&message=tresc wiadomosci**

Dodanie parametru **raport=1** spowoduje odsyłanie maila z raportem (potwierdzenie wysłania wiadomości lub błąd – jest to przydatne w trakcie testowania usługi):

ADRES: **send.do@smsapi.pl**

TEMAT: **login@haslo\_32znaki\_w\_md5**

TREŚĆ: **from=nadawca&to=numer&raport=1&message=tresc wiadomosci**

Wiadomości mogą być wysłane w kodowaniu plain / quotedprintable / base64.

**Ważne!** Numer telefonu nie może zawierać znaku plus "+" na początku. **Parametr message musi znajdować się na końcu treści wiadomości.**

Nazwa nadawcy (zmienna &from=) musi być aktywna.

W celu wysłanie ecoSMS przez mail2SMS należy dodać parametr eco=1.

## 8. Sprawdzenie ilości punktów na koncie

Parametr	Opis
<i>username</i> *	Nazwa użytkownika w serwisie smsAPI
<i>password</i> *	Hasło do Twojego konta w serwisie smsAPI zaszyfrowane w MD5
<i>points</i> *	Należy podać wartość „1”
<i>details</i>	Dodatkowo wyświetlana jest ilość SMSów pro oraz eco

\* - pole obowiązkowe

Zapytanie: **http://api.smsapi.pl/send.do?username=uzytkownik&password=haslo&points=1&details=1**

Odpowiedź: **Points:<PKT>;<proSMS>;<ecoSMS>**  
**<PKT>** ilość punktów dostępnych dla tego użytkownika w serwisie smsAPI  
**<proSMS>** ilość SMSów typu proSMS w serwisie smsAPI  
**<ecoSMS>** ilość SMSów typu ecoSMS w serwisie smsAPI

Przykład: **Points:100;606;1428**

## 9. Odbieranie raportu doręczenia wiadomości - CALLBACK

**UWAGA!** Użytkownicy którzy korzystali z CALLBACK przed 25.06.2009, chcący korzystać z nowej wersji CALLBACK proszeni są o kontakt z smsAPI.pl w celu przełączenia konta na nowy sposób odbioru raportów.

W celu sprawdzenia statusu wiadomości należy na stronie <http://www.smsapi.pl> w zakładce „Historia” ustawić w „Adres callback do odbioru raportów doręczenia” adres do skryptu do którego przekazywany będzie raport doręczenia wiadomości. Aby móc wprowadzić adres callback, skrypt musi być umieszczony w podanej lokalizacji.

Np.: [http://www.moja\\_strona.pl/status\\_update.php](http://www.moja_strona.pl/status_update.php)

Ważne, aby podany adres był poprawnym adresem, tzn. aby wskazany skrypt istniał na serwerze.

Do skryptu wysyłane będą status wiadomości (od 1 do 5 w zależności od tego dla ilu wiadomości zmienił się status). Parametry będą podane metodą GET i będą oddzielone przecinkami np. :

```
$_GET['MsgId']=09062414383994024,09062414383994025
```

```
$_GET['status']=403,404
```

Poniższa tabela przedstawia wysyłane parametry:

Parametr	Opis
<b>MsgId*</b>	ID wysłanej wiadomości
<b>status*</b>	Kod zwrotu wg tabeli kodów zwrotów z 'Dodatek nr 1'
<b>idx*</b>	Opcjonalny parametr użytkownika wysłany z SMS'em
<b>donedate*</b>	Czas dostarczenia wiadomości w formacie timestamp
<b>username*</b>	Nazwa użytkownika wysyłającego wiadomość

\*Należy zachować wielkość znaków!

**Skrypt powinien zwracać OK**, w innym przypadku nastąpią ponowne odwołania z interwałem 300 sekund. Jako zabezpieczenie można dopisać sprawdzanie adresów IP naszych serwerów 195.82.172.37 , 94.23.205.124 , 188.40.45.130, 78.46.57.204

### 9.1. Sprawdzenie statusu wiadomości – w przypadku nie odebrania przez wywołanie zwrotne CALLBACK

Parametr	Opis
<i>username</i> *	Nazwa użytkownika w serwisie smsAPI
<i>password</i> *	Hasło do Twojego konta w serwisie smsAPI zaszyfrowane w MD5
<i>status</i> *	Numer wiadomości (zwrócony podczas wysyłki). Istnieje możliwość podania kilku numerów oddzielonych przecinkami.
<i>MsgId</i>	ID wysłanej wiadomości

\* - pole obowiązkowe

Zapytanie: **<http://api.smsapi.pl/send.do?username=uzytownik&password=haslo&status=17101000090360359,17101000090360358>**

Odpowiedź: **OK:<NID>:<MsgID>,OK:<NID>:<MsgID>**  
**<NID>** kod statusu wiadomości (patrz tabela kodów zwrotu).

Przykład: **OK:404:17101000090360359,OK:404:17101000090360358**

**UWAGA!** Do sprawdzania raportu doręczenia należy stosować wywołanie zwrotne CALLBACK. Powyżej przedstawiona metoda służy do sprawdzenia raportu doręczenia w momencie gdy nie został on odebrany z powodu chwilowej niedostępności lub błędu skryptu CALLBACK. Przy zbyt częstych odwołaniach system będzie zwracał ERROR:500

## 10. Wysyłanie wiadomości MMS

Wysyłanie wiadomości MMS odbywa się za pomocą odwołania do adresu <http://api.smsapi.pl/mmssend.do> (dla połączeń ssl: <https://ssl.smsapi.pl/mmssend.do>) i podanie odpowiednich parametrów.

Poniższa tabela przedstawia parametry niezbędne do wysłania wiadomości MMS.

Parametr	Opis
username	Nazwa użytkownika w serwisie smsAPI
password	Hasło do Twojego konta w serwisie smsAPI zaszyfrowane w MD5
to	Numer odbiorcy wiadomości w formacie 48xxxxxxxx lub xxxxxxxx. Np. 48505602702 lub 505602702.
subject	Temat wiadomości MMS
date	Data w formacie timestamp kiedy wiadomość ma być wysłana
smil	Wiadomość MMS w formacie SMIL (przykłady w punkcie 3)

Zapytanie: **[http://api.smsapi.pl/mmssend.do?username=uzytownik  
&password=haslo&to=48500500500&subject=Testowy MMS&smil=\[smil\]](http://api.smsapi.pl/mmssend.do?username=uzytownik&password=haslo&to=48500500500&subject=Testowy MMS&smil=[smil])**

Odpowiedź: **OK:<ID>:<POINTS>**

lub w razie błędu

**ERROR:<ERR>**

**<ID>** numer ID wysłanego MMSa. Numer należy zachować w celu sprawdzenia statusu wiadomości.

**<POINTS>** ilość pobranych punktów

**<ERR>** kod błędu (zobacz tabela kodów błędów w punkcie 4)

Przykład: **OK:17101000090360359:0.4**

lub

**ERROR:102**

### PRZYKŁAD WYSYŁKI

#### 10.1. Odbieranie raportu doręczenia wiadomości - CALLBACK

W celu sprawdzenia statusu wiadomości należy na stronie <http://www.smsapi.pl> w zakładce „Historia MMS” ustawić w „Adres callback do odbioru raportów doręczenia MMS” adres do skryptu do którego przekazywany będzie raport doręczenia wiadomości.

Np.: [http://www.moja\\_strona.pl/status\\_update.php](http://www.moja_strona.pl/status_update.php)

Ważne, aby podany adres był poprawnym adresem, tzn. aby wskazany skrypt istniał na serwerze.

Do skryptu wysyłane będą statusy wiadomości. Parametry będą podane metodą GET np. :

**MsgId=09062414383994024**

**status=404**

Poniższa tabela przedstawia wysyłane parametry:

Parametr	Opis
<b>Msgld*</b>	ID wysłanej wiadomości
<b>status*</b>	Kod doręczenia wiadomości
<b>donedate*</b>	Czas dostarczenia wiadomości
<b>username*</b>	Nazwa użytkownika z którego została wysłana wiadomość

\*Należy zachować wielkość znaków!

**Skrypt powinien zwracać OK**, w innym przypadku nastąpią ponowne odwołania z interwałem 300 sekund. Jako zabezpieczenie można dopisać sprawdzanie adresów IP naszych serwerów 195.82.172.37 , 94.23.205.124 , 188.40.45.130 , 78.46.57.204 .

## 11. Odbiór SMS/MMS

Serwis **smsAPI** oferuje również odbiór wiadomości SMS oraz MMS. Odbierać można odpowiedzi na wysłane wiadomości ecoSMS (do 24h od momentu wysłania wiadomości) lub wiadomości SMS i/lub MMS wysłane na wykupiony numer dedykowany.

### 11.1. Odbiór SMS

Odbiór wiadomości odbywać się będzie za pomocą skryptu, do którego adres należy ustawić w zakładce Odbiór SMS/MMS → Odbiór SMS. Działanie skryptu opisane jest poniżej:

Po odebraniu wiadomości odwoływać będziemy się do skryptu stworzonego przez Państwa, który powinien obsługiwać 4 parametry z tablicy POST:

Parametr	Opis
sms_to	numer telefonu odbiorcy
sms_from	numer telefonu nadawcy
sms_text	treść wiadomości
sms_date	czas w postaci timestamp pobrany z SMS'a

Jako zabezpieczenie prosimy dopisać sprawdzanie adresów IP naszych serwerów 195.82.172.37, 94.23.205.124, 188.40.45.130, 94.23.95.250 , 78.46.57.204. Dane wysyłane są w kodowaniu UTF8.

Skrypt powinien zwracać OK, w innym przypadku nastąpią ponowne odwołania z interwałem 300 sekund.

Odwołanie może działać z szyfrowaniem SSL (adres <https://..>).

### 11.2. Odbiór MMS

Odbiór wiadomości odbywać się będzie za pomocą skryptu, do którego adres należy ustawić w zakładce Odbiór SMS/MMS → Odbiór MMS. Działanie skryptu opisane jest poniżej:

Po odebraniu wiadomości odwoływać będziemy się do skryptu stworzonego przez Państwa, który powinien obsługiwać parametry z tablicy POST oraz FILES podane poniżej:

**Tablica POST:**

Parametr	Opis
mms_to	numer telefonu odbiorcy
mms_from	numer telefonu nadawcy
mms_subject	temat wiadomości
mms_date	czas w postaci timestamp pobrany z MMS'a

**Tablica FILES:**

Parametr	Opis
name	oryginalna nazwa wysyłanego pliku
type	typ MIME wysyłanego pliku (JPEG, GIF, ...)
tmp_name	tymczasowa nazwa pliku, który został wysłany na serwer
error	numer błędu (0 oznacza prawidłowe wysłanie)
size	rozmiar wysyłanego pliku (w bajtach)

Jako zabezpieczenie prosimy dopisać sprawdzanie adresów IP naszych serwerów 195.82.172.37, 94.23.205.124, 188.40.45.130, 94.23.95.250, 78.46.57.204. Dane wysyłane są w kodowaniu UTF8.

Skrypt powinien zwracać OK, w innym przypadku nastąpią ponowne odwołania z interwałem 300 sekund. PRZYKŁAD

Odwołanie może działać z szyfrowaniem SSL (adres https://..).

## 12. Korzystanie z serwisu smsAPI.pl przy wykorzystaniu WebServices (SOAP)

Serwis smsAPI udostępnia możliwość połączenia za pomocą WebServices.

WebService SOAP znajduje się pod adresem:

<http://www.smsapi.pl/webservices>

<https://www.smsapi.pl/webservices> (dla połączeń SSL)

WSDL można znaleźć pod adresem

<http://www.smsapi.pl/webservices?wsdl>

<https://www.smsapi.pl/webservices?wsdl> (dla połączeń SSL)

### 12.1. Lista funkcji

Nazwa funkcji	Opis działania
soap_SendSms	Wysyła pojedynczą wiadomość.
soap_SendMulti	Masowa wysyłka wiadomości.
soap_DelSchSmsByld	Usuwanie wiadomości zaplanowanych
soap_CheckPoints	Sprawdzanie ilości punktów na koncie.
soap_GetSmsBylds	Sprawdzanie raportów doręczenia wiadomości po wskazaniu ich ID
soap_GetSmsByDate	Sprawdzanie raportów doręczenia wiadomości wysłanych w określonym przedziale czasowym
soap_AddNumber	Dodaje numer do książki telefonicznej

soap_DelNumber	Usuwa numer z książki telefonicznej
soap_GetNumbers	Pobiera numer telefonów z książki telefonicznej użytkownika
soap_AddGroup	Dodaje grupę do książki telefonicznej
soap_DelGroup	Usuwa grupę z książki telefonicznej
soap_GetGroups	Pobiera grupy z książki telefonicznej użytkownika
soap_GetSenders	Pobiera dostępne pola nadawcy

## 12.2. Lista parametrów

Parametr (typ)	Pole (typ)	Opis
<b>Client</b> (xsd:complexType)	'username' (xsd:string)	nazwa użytkownika
	'password' (xsd:string)	Hasło w md5
<b>sms_param</b> (xsd:complexType)	'to' (xsd:string)	numer telefonu
	'from' (xsd:string)	pole nadawcy
	'message' (xsd:string)	treść wiadomości
	'eco' (xsd:int)	ustawienie 1 powoduje wysłanie ecoSMS
	'date' (xsd:int)	timestamp daty kiedy wiadomość ma być wysłana
	'arParams' (tns:ArrayOfString)	tablica parametrów max=4
	'idx' (xsd:string)	parametr idx zwracany w callbacku
	'single' (xsd:int)	ustawienie 1 zabezpiecza przed wysłaniem podwójnej wiadomości
	'no_unicode' (xsd:int)	ustawienie 1 zabezpiecza przed wysłaniem wiadomości ze znakami specjalnymi
	'datacoding' (xsd:string)	parametr pozwalający na wysyłanie wiadomości WAP PUSH. (datacoding => bin)
'id_partner' (xsd:int)	parametr wskazujący partnera	
'test' (xsd:int)	ustawienie 1 spowoduje wyświetlenie odpowiedzi takiej jak przy wysłaniu wiadomości, jednak wiadomość nie zostanie wysłana (nie obciąży konta)	
<b>sms_multi_param</b> (xsd:complexType)	'arTo' (tns:ArrayOfString)	tablica numerów telefonów
	'from' (xsd:string)	pole nadawcy
	'message' (xsd:string)	treść wiadomości
	'eco' (xsd:int)	ustawienie 1 powoduje wysłanie ecoSMS
	'date' (xsd:int)	timestamp daty kiedy wiadomość ma być wysłana
	'arParams' (tns:ArrayOfParam)	tablica w której znajdują się tablice parametrów arParam
	'arIdx' (tns:ArrayOfString)	tablica parametrów idx zwracanych w callbacku

	'single' (xsd:int)	ustawienie 1 zabezpiecza przed wysłaniem wiadomości składającej się z kilku części (ERROR:12)
	'no_unicode' (xsd:int)	ustawienie 1 zabezpiecza przed wysłaniem wiadomości ze znakami specjalnymi (ERROR:11)
	'datacoding' (xsd:string)	parametr pozwalający na wysyłanie wiadomości WAP PUSH. (datacoding => bin)
	'id_partner' (xsd:int)	parametr wskazujący partnera
	'test' (xsd:int)	ustawienie 1 spowoduje wyświetlenie odpowiedzi takiej jak przy wysłaniu wiadomości, jednak wiadomość nie zostanie wysłana (nie obciążą konta)
<b>group</b> (xsd:complexType)	'id' (xsd:int)	id grupy w książce telefonicznej
	'name' (xsd:string)	nazwa grupy
	'info' (xsd:string)	opis grupy
	'num_count'	Ilość numerów w grupie
<b>id_group</b> (xsd:int)		id grupy w książce telefonicznej
<b>number</b> (xsd:complexType)	'number' (xsd:int)	numer telefonu
	'name' (xsd:string)	nazwa w książce telefonicznej
	'id_group' (xsd:int)	numer grupy użytkowników w książce telefonicznej
<b>date_from</b> (xsd:int)		timestamp daty od której mają być pobrane raporty
<b>date_to</b> (xsd:int)		timestamp daty do której mają być pobrane raporty
<b>arlds</b> (tns:ArrayOfString)		tabela parametrów id wiadomości

### 12.3. Wysyłanie pojedynczej wiadomości

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_SendSms</b>	Client (tns:Client)	username, password
	sms_param (tns:sms_param)	to, message

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu (Dodatek nr 2)
<b>response</b> (tns:ArrayOfString)	Array ( [0] => 09082612378015806 - <b>ID wiadomości</b> )
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania
<b>count</b> (xsd:int)	ilość wysłanych SMSów
<b>points</b> (xsd:float)	ilość pobranych punktów za wysyłkę

## 12.4. Wysyłanie wiadomości masowych

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_SendMulti</b>	Client (tns:Client)	username, password
	sms_param (tns:sms_multi_param)	arTo, message

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu (Dodatek nr 2)
<b>response</b> (tns:ArrayOfString)	Array ( [0] => 09082612435449856:48500222333 – ID pierwszej wiadomości [1] => 09082612438226939:48600111999 – ID drugiej wiadomości [2] => 09082612434686822:48700222111 – ID trzeciej wiadomości )
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania
<b>count</b> (xsd:int)	ilość wysłanych SMSów
<b>points</b> (xsd:float)	ilość pobranych punktów za wysyłkę

## 12.5. Usuwanie wiadomości zaplanowanych

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_DelSchSmsById</b>	Client (tns:Client)	username, password
	id (xsd:string)	ID wiadomości do usunięcia

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu (Dodatek nr 2)
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania

## 12.6. Sprawdzanie ilości punktów

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_CheckPoints</b>	Client (tns:Client)	username, password

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania
<b>points</b> (xsd:float)	ilość punktów na koncie

**12.7. Sprawdzanie raportów doręczenia SMS po ID wiadomości**

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_GetSmsByIds</b>	Client (tns:Client)	username, password
	arlds (tns:ArrayOfString)	przynajmniej jeden parametr

powrót do spisu funkcji

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania
<b>arSms</b> (tns:ArrayOfSms)	<pre> Array (   [0] =&gt; Array   (     [id] =&gt; 0908112341234123 – <b>ID wiadomości</b> (xsd:string)     [to] =&gt; 48500600700 – <b>numer telefonu odbiorcy</b> (xsd:string)     [from] =&gt; smsAPI.pl – <b>pole nadawcy</b> (xsd:string)     [message] =&gt; wiadomosc z serwisu – <b>treść wiadomości</b> (xsd:string)     [flash] =&gt; 0 – <b>czy wiadomość flash</b> (xsd:int)     [eco] =&gt; 0 – <b>czy wiadomość eco</b> (xsd:int)     [date] =&gt; 1249897478 – <b>data odebrania SMS'a przez serwis</b>(xsd:int)     [status] =&gt; 404 – <b>raport doręczenia SMS'a</b> (xsd:int)     [submitdate] =&gt; 1249897479 – <b>data wysłania SMS'a</b> (xsd:int)     [doneatdate] =&gt; 1249897485 – <b>data doręczenia SMS'a</b> (xsd:int)   ) ) </pre>

**12.8. Sprawdzanie raportów doręczenia SMS po dacie wysłania wiadomości**

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_GetSmsByDate</b>	Client (tns:Client)	username, password
	date_from (xsd:int)	timestamp: „od daty”
	date_to (xsd:int)	timestamp: „do daty”

powrót do spisu funkcji

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania
<b>arSms</b> (tns:ArrayOfSms)	<pre> Array (   [0] =&gt; Array   (     [id] =&gt; 0908112341234123 – <b>ID wiadomości</b> (xsd:string)     [to] =&gt; 48500600700 – <b>numer telefonu odbiorcy</b> (xsd:string)     [from] =&gt; smsAPI.pl – <b>pole nadawcy</b> (xsd:string)     [message] =&gt; wiadomosc z serwisu – <b>treść wiadomości</b> (xsd:string)     [flash] =&gt; 0 – <b>czy wiadomość flash</b> (xsd:int)     [eco] =&gt; 0 – <b>czy wiadomość eco</b> (xsd:int)     [date] =&gt; 1249897478 – <b>data odebrania SMS'a przez serwis</b>(xsd:int)     [status] =&gt; 404 – <b>raport doręczenia SMS'a</b> (xsd:int)     [submitdate] =&gt; 1249897479 – <b>data wysłania SMS'a</b> (xsd:int)   ) ) </pre>

```
[doneDate] => 1249897485 – data doręczenia SMS'a (xsd:int)
    )
)
```

### 12.9. Dodawanie numeru do książki telefonicznej

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_AddNumber</b>	Client (tns:Client)	username, password
	number (tns:number)	number, name, id_group

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania

### 12.10. Usuwanie numeru z książki telefonicznej

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_DelNumber</b>	Client (tns:Client)	username, password
	number (tns:number)	number, name, id_group

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania

### 12.11. Pobieranie numerów z książki telefonicznej użytkownika

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_GetNumbers</b>	Client (tns:Client)	username, password
	id_group (xsd:int)	Id grupy

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania
<b>arNumbers</b> (tns:ArrayOfNumber)	<pre>Array (     [0] =&gt; Array     (         [number] =&gt; 48500600700 – numer telefonu (xsd:string)         [name] =&gt; Mariusz – nazwa w książce (xsd:string)         [id_group] =&gt; 290 – Id grupy (xsd:int)     ) )</pre>

**12.12. Dodawanie grupy do książki telefonicznej**

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_AddGroup</b>	Client (tns:Client)	username, password
	group (tns:group)	name

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania
<b>id_group</b> (xsd:int)	Id grupy

**12.13. Usuwanie grupy z książki telefonicznej**

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_DelGroup</b>	Client (tns:Client)	username, password
	group (tns:group)	name

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania
<b>num_count</b> (xsd:int)	Ilość numerów w grupie

**12.14. Pobieranie listy grup z książki telefonicznej użytkownika**

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_GetGroups</b>	Client (tns:Client)	username, password

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania
<b>arGroups</b> (tns:ArrayOfGroup)	<pre> Array (   [0] =&gt; Array   (     [id] =&gt; 162 – <b>Id grupy</b> (xsd:string)     [name] =&gt; smsapi – <b>nazwa grupy</b> (xsd:string)     [info] =&gt; pierwsza grupa - <b>opis grupy</b> (xsd:string)     [num_count] =&gt; 2 – <b>ilość numerów w grupie</b> (xsd:int)   ) ) </pre>

### 12.15. Pobieranie dostępnych pól nadawcy

Nazwa metody	Parametry (typ)	Wymagane pola
<b>soap_GetSenders</b>	Client (tns:Client)	username, password

[powrót do spisu funkcji](#)

Odpowiedź (typ)	Opis
<b>result</b> (xsd:int)	0 = OK lub kod błędu
<b>description</b> (xsd:string)	opis błędu lub wykonanego działania
<b>arSenders</b> (tns:ArrayOfString)	Array ( [0] => Nadawca_1 [1] => Nadawca_2 [2] => Nadawca_3 [3] => Nadawca_4 )

### 13. Uwagi końcowe

Numer telefonu powinny być podawane w formacie 11 cyfrowym (np. 48502602702). W przypadku gdy zostanie podany numer bez prefiksu 48, zostanie on automatycznie dopisany.

Za znaki specjalne uważa się wszystkie znaki nie spełniające wyrażenia regularnego: @£\$¥èéùìòçøå\_{}|[-]|ÆæßÉ!"#%&'()\*+,-./0-9:;<=>?A-ZÄÖÑÛŞıa-zäöñüà <enter>

#### Przykładowa implementacja w skryptach PHP:

```
// Funkcja zwraca wynik 1 lub 0 w zależności czy w zmiennej $_message występują
// znaki specjalne.

function is_spec_char($_message) {
    mb_regex_encoding('UTF-8');
    if(mb_eregi("[^\@£$¥èéùìòçøå_{}|[-]|ÆæßÉ!\"#%&'()*+,-./0-9:;<=>?A-ZÄÖÑÛŞıa-zäöñüà \r\n]",$_message)) return 1;
    else return 0;
}
```

**UWAGA!** Znaki: ^ { } [ ] ~ \ | <enter> liczone są **podwójnie** (w przypadku wysyłania wiadomości bez znaków specjalnych) ze względu na wymogi specyfikacji GSM.

**Adresy IP serwisu smsAPI:** 195.82.172.37, 94.23.205.124, 188.40.45.130, 94.23.95.250, 78.46.57.204

#### Tabela ilości wysłanych SMS w zależności od długość wiadomości:

Bez znaków specjalnych		Ze znakami specjalnymi (w tym PL)	
Ilość znaków	Ilość punktów	Ilość znaków	Ilość punktów
160	1 SMS	70	1 SMS
305	2 SMSy	134	2 SMSy
457	3 SMSy	201	3 SMSy

## Dodatek nr 1 – Lista raportów doręczeń wiadomości

Tabela kodów zwrotu:

Numer NID	STATUS	Opis
401	<i>NOT_FOUND</i>	Błędny numer ID lub raport wygasł
402	<i>EXPIRED</i>	Wiadomość wygasła
403	<i>SENT</i>	Wiadomość została wysłana
404	<i>DELIVERED</i>	Wiadomość została odebrana przez odbiorcę
405	<i>UNDELIVERED</i>	Wiadomość niedostarczona (nieważny numer, błąd roamingu)
406	<i>FAILED</i>	Błąd podczas dostarczania wiadomości - prosimy zgłosić
407	<i>REJECTED</i>	Wiadomość odrzucona (nieдоступny roaming)
408	<i>UNKNOWN</i>	Brak raportu (SMS doręczony lub brak możliwości doręczenia)
409	<i>QUEUED</i>	Wiadomość oczekuje do wysłania.
410	<i>ACCEPTED</i>	Wiadomość dostarczona do centrum operatora

## Dodatek nr 2 - Kody błędów

Tabela kodów błędów:

ERROR	Opis
11	Zbyt długa lub brak wiadomości
12	Wiadomość zawiera ponad 160 znaków (gdy użyty parametr &single=1)
13	Nieprawidłowy numer odbiorcy
14	Nieprawidłowe pole nadawcy
17	Nie można wysłać FLASH ze znakami specjalnymi
18	Nieprawidłowa liczba parametrów
19	Za dużo wiadomości w jednym odwołaniu (maksymalnie 100)
20	Nieprawidłowa liczba parametrów IDX
21	Wiadomość MMS ma za duży rozmiar (maksymalnie 100kB)
22	Błędny format SMIL
101	Niepoprawne lub brak danych autoryzacji
102	Nieprawidłowy login lub hasło
103	Brak punktów dla tego użytkownika
104	Brak szablonu
105	Błędny adres IP (włączony filtr IP dla interfejsu API)
200	Nieudana próba wysłania wiadomości
300	Nieprawidłowa wartość pola <i>points</i> (przy użyciu pola <i>points</i> jest wymagana wartość 1)
301	ID wiadomości nie istnieje
400	Nieprawidłowy ID statusu wiadomości
999	Wewnętrzny błąd systemu (prosimy zgłosić)

## Dodatek nr 3 - Kodowanie

Domyślnie kodowanie znaków ustawione jest na windows-1250. Jednak do wysyłania wiadomości można użyć jednego z poniżej przedstawionych rodzajów kodowania. W tym celu wykorzystać należy parametr **&encoding**.

```
'iso-8859-1'  
'iso-8859-2'  
'iso-8859-3'  
'iso-8859-4'  
'iso-8859-5'  
'iso-8859-7'  
'windows-1250'  
'windows-1251'  
'utf-8'
```

Przykład:

```
http://api.smsapi.pl/send.do?username=uzytownik&password=haslo&to=4850000000  
&encoding=utf-8&message=wiadomosc
```

## Dodatek nr 4 – Przykładowe skrypty

### Wysłanie SMSa funkcją fopen

```
<?php  
  
$username = "uzytownik";  
$password = md5("haslo");  
$to = "48502602702";  
$from = "48502607702";  
$message= urlencode("moja wiadomosc");  
  
if ($username && $password && $to && $message) {  
    $data = "?username=$username&password=$password&to=$to&message=$message&from=$from";  
    $plik = fopen('http://api.smsapi.pl/send.do'. $data, 'r');  
    $wynik = fread($plik, 1024);  
    fclose($plik);  
  
    echo $wynik;  
}  
  
?>
```

### Odebranie raportu CALLBACK

```
<?php  
  
// UWAGA! Poniższy przykład jest tylko poglądowy. Nie sprawdza on danych  
// wejściowych co jest zalecane dla zmiennych GET: MsgId i status, oraz nie  
// sprawdza czy zmienne nie są podane przez osoby trzecie. Zaleca się sprawdzenie  
// adresu IP komputera wywołującego skrypt.  
//  
// Adresy IP serwisu smsAPI: 195.82.172.37, 94.23.205.124, 188.40.45.130, 94.23.95.250, 78.46.57.204  
//  
// W przypadku wysłania z wiadomością parametru idx zostanie on zwrócony $_GET['idx']  
  
if($_GET['MsgId'] && $_GET['status'] ) {  
    mysql_select_db('nazwa_bazy',mysql_connect('localhost','login','haslo'));  
    $arIds = explode(',', $_GET['MsgId']);  
    $arStatus = explode(',', $_GET['status']);  
    $arIdx = explode(',', $_GET['idx']);
```

```
if($arIds) foreach($arIds as $k => $v){
    mysql_query("UPDATE sms SET sms_status = ".$arStatus[$k].", sms_index = ".$arIdx[$k]." WHERE
    sms_id=".$v." LIMIT 1");
}
mysql_close();

echo "OK";
}

?>
```

## Przykładowa wysyłka PHP połączona z formularzem HTML

```
<?php
if (strlen($_POST['sms_from'])>0 &&
    strlen($_POST['sms_to'])>=9 &&
    strlen($_POST['sms_message'])>0)
{
    $username = "login";
    $password = md5("haslo");
    $from = $_POST['sms_from'];
    $to = $_POST['sms_to'];
    $message= urlencode($_POST['sms_message']);
    echo file_get_contents("http://api.smsapi.pl/send.do?".
    "username=$username&password=$password&to=$to&".
    "message=$message&from=$from",FALSE,NULL,0,100);
}
?>
<html>
<head>
<meta http-equiv="Content-type" content="text/html; charset=windows-1250">
</head>
<body>
<form name="sms" method="POST" action="">
<table>
<tr>
<td>Od:</td>
<td><input type="text" name="sms_from" value=""></td>
</tr>
<tr>
<td>Do:</td>
<td><input type="text" name="sms_to" value=""></td>
</tr>
<tr>
<td>Wiadomość:</td>
<td><textarea name="sms_message"></textarea></td>
</tr>
<tr>
<td colspan="2"><input type="submit" value="wyślij"></td>
</tr>
</table>
</form>
</body>
</html>
```

## Przykład wysyłki za pomocą SOAP

```
<?php
require_once('lib/nusoap.php'); //bibliotekę SOAP należy pobrać z internetu

$client = new nusoap_client('http://www.smsapi.pl/webservices?wsdl', true);

$klient = Array (
    'username' => 'login',
    'password' => '6bd407d999a3ba1a792b4250e1f92304' //hasło w md5
```

```

    );

$sms_param = Array (
    'to' => 48500111222,
    'message' => 'Hello world!',
    'eco' => 1,
    'test'=> 0,
    'from' => 'smsAPI.pl',
    );

$result = $client->call (
    'soap_SendSms',
    Array (
        'client' => $client,
        'sms_param' => $sms_param
    )
);

if ($client->fault) {
    echo '<h2>Fault</h2><pre>';
    print_r($result);
    echo '</pre>';
} else {
    $err = $client->getError();
    if ($err) {
        echo '<h2>Error</h2><pre>' . $err . '</pre>';
    } else {
        echo '<h2>Result</h2><pre>';
        print_r($result);
        echo '</pre>';
    }
}
}
?>

```

## Przykład skryptu wysyłającego wiadomość MMS

```

<?php

$username = 'login';
$password = md5('haslo');
$to = "48500000000";
$subject = urlencode("smsAPI.pl");
$smil= urlencode('<smil><head><layout><root-layout width="160" height="134"/><region id="Image"
width="160" height="114" left="0" top="0"/><region id="Text" width="160" height="20" left="0"
top="120"/><region id="Audio" width="106" height="1" left="0" top="120"/></layout></head><body><par
dur="0s"><text src="http://www.smsapi.pl/mms.txt"
region="Text" /><audio src="http://www.smsapi.pl/mms.wav" region="Audio" /></par></body></smil>');

if ($username && $password) {
    $data = "?username=$username&password=$password&to=$to&subject=$subject&smil=$smil";
    $plik = fopen('http://api.smsapi.pl/mmssend.do'.$data,'r');
    $wynik = fread($plik,1024);
    fclose($plik);
    echo $wynik;
}

?>

```

## Przykładowa wiadomość MMS w formacie SMIL

```
<smil>
  <head>
    <layout>
      <root-layout width="160" height="135"/>
      <region id="Image" width="160" height="114" left="0" top="0"/> //Rozmiar obrazka
      <region id="Text" width="160" height="20" left="0" top="120"/> //Rozmiar pola tekst.
      <region id="Audio" width="106" height="1" left="0" top="120"/> //Rozmiar pola audio.
    </layout>
  </head>
  <body>
    <par dur="0s">
       //Adres do obrazka
      <text src="http://www.smsapi.pl/mms.txt" region="Text" /> //Adres do pliku tekstowego
      <audio src="http://www.smsapi.pl/mms.wav" region="Audio" /> //Adres do pliku audio
    </par>
  </body>
</smil>
```

## Przykładowy skrypt odbioru MMS

```
<?php
$IP = $_SERVER['REMOTE_ADDR'];

if ($IP=='195.82.172.37' || $IP=='94.23.205.124' || $IP=='188.40.45.130' || $IP=='94.23.95.250') {
    $recived=0;
    foreach($_FILES as $plik) {

        if(is_uploaded_file($plik['tmp_name'])) {

            if(move_uploaded_file($plik['tmp_name'],'mms/'.$plik['name'])) {
                $recived=1;
            }
        }
    }
    if ($recived == 1) {
        echo "OK";
    }
}
?>
```

## Przykładowa wiadomość MMS wysłana do skryptu

```
POST: Array
(
    [mms_subject] => Testowy MMS
    [mms_from] => 48600000000
    [mms_to] => 48500000000
    [mms_date_rcv] => 1256031673
    [mms_date] => 1256031643
)

FILES: Array
(
    [0] => Array
        (
            [name] => 0.JPG
            [type] => image/jpeg
            [tmp_name] => /tmp/phpGXutNv
            [error] => 0
            [size] => 25097
        )
)
```

```
)  
  
[1] => Array  
(  
  [name] => 1.txt  
  [type] => text/plain  
  [tmp_name] => /tmp/phpaoM6dU  
  [error] => 0  
  [size] => 14  
)  
  
[2] => Array  
(  
  [name] => 3.amr  
  [type] => audio/AMR  
  [tmp_name] => /tmp/php6l0KEi  
  [error] => 0  
  [size] => 27078  
)  
)
```