

Trendoo

Integrazione SMS e CONTATTI via Webservice

Indice generale

Integrazione SMS via Webservice	3
1. INTRODUZIONE.....	3
2. FORMATO DEI PARAMETRI.....	3
3. INVIO DI SMS.....	4
4. STATO DEI MESSAGGI.....	5
5. STORICO DEI MESSAGGI.....	6
6. CREDITO SMS DISPONIBILE.....	6
7. MESSAGGI RICEVUTI TRAMITE IL SERVIZIO DI RICEZIONE	7
8. GESTIONE SUBACCOUNTS.....	8
9. GESTIONE CREDITI SUBACCOUNTS.....	9
10. RICERCA E GESTIONE GRUPPI.....	11
11. RICERCA CONTATTI.....	11
12. GESTIONE CONTATTI.....	13
Integrazione Contatti via Webservice	16
1. INTRODUZIONE.....	16
2. FORMATO DEI PARAMETRI.....	16
3. GESTIONE GRUPPI.....	16
Aggiunta di un gruppo:.....	17
Modifica di un gruppo:.....	17
Eliminazione di un gruppo:.....	17
4. GESTIONE CONTATTI.....	17
Elenco dell'indirizzo email di tutti i contatti di un gruppo:.....	17
Elenco dei numeri telefonici di tutti i contatti di un gruppo:.....	17

Aggiunta di un contatto a un gruppo:.....	17
Aggiunta di un contatto (solo indirizzo email) a un gruppo:.....	18
Aggiunta di un contatto (solo TELEFONO) a un gruppo:.....	18
Aggiunta di più contatti a un gruppo:.....	18
Modifica di un contatto:.....	18
Eliminazione di un contatto da un gruppo:.....	18
Eliminazione di un contatto (identificato dal solo indirizzo email) da un gruppo:.....	18
Eliminazione di un contatto (identificato dal solo telefono) da un gruppo:.....	18
Eliminazione di un contatto:.....	18
Eliminazione di un contatto (identificato dal solo indirizzo email):.....	18
Eliminazione di un contatto (identificato dal solo telefono):.....	18
Eliminazione di più contatti:.....	19
Eliminazione di più contatti (identificati dai soli indirizzi email):.....	19
Eliminazione di più contatti (identificati dai soli telefoni):.....	19
5. GESTIONE CAMPAGNE EMAIL.....	19
Elenco delle campagne attive.....	20
Creazione di un invio email associato ad una campagna.....	20
Codici di Errore	21

Integrazione SMS via Webservice

1. INTRODUZIONE

I server Trendoo mettono a disposizione degli sviluppatori un'interfaccia via Webservice per l'invio e la ricezione dei messaggi SMS attraverso il gateway trendoo, e permettono inoltre di recuperare tutti i dati riguardanti lo stato dei messaggi e lo storico SMS.

Nel caso fosse necessaria una connessione criptata, è possibile effettuare le stesse richieste utilizzando la connessione https.

2. FORMATO DEI PARAMETRI

- [il WSDL del Webservice può essere raggiunto all'indirizzo:
<https://api.trendoo.net/trendoows?wsdl>
- [il Webservice è compatibile SOAP 1.1, per massima compatibilità con tutte le versioni;
- [tutte le richieste Webservice devono avere due parametri obbligatori username e password, che rappresentano le credenziali personali di accesso al portale trendoo;
- [un numero di telefono in formato internazionale è una qualunque sequenza di cifre preceduta da un carattere '+' (esempio, +393471234567);
- [le tipologie di SMS sono sempre codificate con le seguenti stringhe di 2 caratteri:
 - [**GS**: SMS Gold Standard (SMS di alta qualità con notifica di ricezione ma senza la possibilità di personalizzare il mittente);
 - [**GP**: SMS Gold Plus (SMS di alta qualità con notifica di ricezione e mittente personalizzato);
 - [**SI**: SMS Silver (**SMS di bassa qualità inviati tramite operatori esteri, senza garanzia di consegna e mittente fisso**);
- [tutte le richieste che non vanno a buon fine lanciano un Webservice "Fault" (ad es. in ambiente .NET è System.ServiceModel.FaultException) contenente il codice dell'errore ed una descrizione testuale nel formato:
<numeric-errorcode>:<errormessage>
- [i codici nazione utilizzati sono i codici a 2 caratteri standard ISO 3166
http://www.iso.org/iso/english_country_names_and_code_elements
- [il charset utilizzato è **UTF-8**.

3. INVIO DI SMS

INVIO DI SMS SINGOLI:

`sendSMS(String username, String password, String message type,
String sender, String recipient, String message,
String order id): String`

INVIO DI SMS MULTIPLI:

`sendMultipleSMS(String username, String password, String message type, String
sender, Array Of String recipients,
String message, String order id): String`

INVIO POSTICIPATO DI SMS SINGOLI:

`scheduleSMS(String username, String password, String message type,
String sender, String recipient, String message,
String order id, Date send when): String`

INVIO POSTICIPATO DI SMS MULTIPLI:

`scheduleMultipleSMS(String username, String password,
String message type, String sender,
Array of String recipients, String message,
String order id, Date send when): String`

L'invio di SMS richiede i seguenti parametri obbligatori:

1. `message_type`: tipo di messaggio; (GS, GP, SI);
 2. `recipient/recipients`: un numero o una lista di numeri telefonici; in formato internazionale o in formato nazionale (in questo caso la nazione utilizzata per l'invio sarà la nazione dell'account Trendoo utilizzato);
 3. `message`: il testo del messaggio da inviare, massimo 160 caratteri per un unico SMS, massimo 1000 caratteri per un SMS concatenato. Se il testo supera i 160 caratteri verranno addebitati "n" SMS composti da 153 caratteri ciascuno. Attenzione: i caratteri "€", "[", "]", "{", "}", "|", "\", "^", "~", vengono calcolati come 2 caratteri;
 4. `sender`: mittente, nel caso di SMS con mittente personalizzato (GP), massimo 16 caratteri numerici per un numero di telefono in formato internazionale oppure 11 caratteri alfanumerici per una stringa di testo (Es: "+123456789012345" oppure "Azienda_Spa"). Il campo deve essere lasciato vuoto o nullo nel caso di invio di tipi di messaggi che non hanno mittente personalizzato (GP,SI), altrimenti verrà lanciata un'eccezione!
- [`scheduled_delivery_time`: data e ora per l'invio posticipato;

Tutte le operazioni restituiscono:

- [order_id: una stringa alfanumerica da associare all'invio, indispensabile per ottenere successivamente le informazioni riguardanti lo stato dei messaggi. La lunghezza massima è di 32 caratteri, tutti i caratteri eccedenti non saranno considerati. Se non specificato, ne viene restituito uno generato dal server, unico per ogni invio.

4. STATO DEI MESSAGGI

getSMSStatus(String username, String password, String orderId)
: Array Of WSMMessageStatus

```
WSMessageStatus : class {  
    String recipient  
    String status  
    Date deliveryDate  
}
```

Successivamente all'invio dei messaggi è possibile richiederne lo stato. Specificando come unico parametro l'id dell'SMS (parametro orderId) si richiede al server l'esito dell'invio e viene restituita una lista di oggetti WSMMessageStatus, ciascuno dei quali rappresenta lo stato d'invio per ogni singolo destinatario del messaggio.

Il campo delivery_date è valorizzato esclusivamente per gli SMS effettivamente consegnati senza errori (stato DLVRD).

Il campo status può assumere i seguenti valori:

SCHEDULED	// posticipato, non ancora inviato
SENT	// inviato, non attende delivery
DLVRD	// l'SMS è stato correttamente ricevuto
ERROR	// errore nella consegna dell'SMS (Es. numero non esistente)
TIMEOUT	// l'operatore non ha fornito informazioni sull'SMS entro 48 ore
TOOM4NUM	// troppi SMS per lo stesso destinatario nelle ultime 24 ore
TOOM4USER	// troppi SMS inviati dall'utente nelle ultime 24 ore
UNKNPFX	// prefisso SMS non valido o sconosciuto
UNKNRCPT	// numero di telefono del destinatario non valido o sconosciuto
WAIT4DLVR	// messaggio inviato, in attesa di delivery
WAITING	// in attesa, non ancora inviato
UNKNOWN	// stato sconosciuto

E' inoltre possibile richiedere di poter ricevere lo stato dei messaggi attraverso un HTTP/POST su un proprio indirizzo, i parametri passati sono:

- [order_id: identificativo univoco alfanumerico associato all'SMS creato al momento dell'invio;
- [recipient: numero di telefono del destinatario dell'SMS;



- [status: stringa contenente lo status del messaggio i cui valori possibili sono quelli del campo status riportato in precedenza;
- [delivery_date: data/ora di ricezione dell'SMS sul telefono del destinatario, solo per gli SMS effettivamente consegnati senza errori (stato DLVRD).

Per abilitare questa opzione, contattaci all'indirizzo supporto@trendoo.com.

5. STORICO DEI MESSAGGI

getHistory(String username, String password, Date from, Date to)
: Array Of WSSentMessage

```
WSSentMessage : class {  
    String messageId  
    Date date  
    String messageType  
    String sender  
    int recipients  
    Date sentWhen  
}
```

E' possibile richiedere al server Trendoo la lista degli SMS inviati nell'intervallo temporale fra due date specifiche; viene restituita una lista di oggetti WSSentMessage, ciascuno dei quali contenente l'id del messaggio, la data di creazione, il tipo di SMS, il mittente (se SMS con mittente personalizzato, altrimenti il campo è vuoto), il numero di destinatari, e la data/ora di invio posticipato (se l'SMS è stato creato come SMS posticipato, altrimenti il campo è vuoto).

6. CREDITO SMS DISPONIBILE

PER APPLICAZIONI SINGOLA NAZIONE / SINGOLO TIPO DI MESSAGGIO:

getAvailableSMS(String username, String password, String messageType, String nation) : Long

Questa procedura richiede il numero di SMS disponibili per un particolare messageType in una certa nazione. Questa è la procedura che si consiglia di utilizzare nel caso di invii SMS destinati principalmente in una singola nazione.

PER APPLICAZIONI COMPLESSE MULTI-NAZIONE / MULTI-CREDITI:

getCredits(String username, String password) : Array of WSSMSCredit



```
WSSMSCredit : class {  
    String messageType;  
    String nation;  
    long credit;  
}
```

Richiede la disponibilità di credito del proprio account trendoo. Viene restituita una lista di oggetti WSSMSCredit, ciascuno dei quali contenente il tipo di credito, la nazione di appartenenza (vuoto se credito internazionale) e la disponibilità SMS. La procedura restituisce non solo i crediti per i tipi SMS Gold (GS), SMS Gold Plus (GP) e SMS Silver (SI), ma anche i crediti disponibili relativi agli invii di tipo "estero" (codice EE).

7. MESSAGGI RICEVUTI TRAMITE IL SERVIZIO DI RICEZIONE

L'utente proprietario di uno o più servizi di ricezione SMS, dedicati o condivisi, che decida di richiedere i messaggi ai server trendoo, può utilizzare indifferentemente uno dei seguenti metodi:

getNewReceivedMessages(String username, String password)
: Array of WSSRMessage

non necessita di alcun parametro, richiede al server tutti i messaggi "nuovi", ovvero, tutti quelli ricevuti dall'ultima richiesta (attenzione! questo metodo richiede attivazione da parte di trendoo, se pertanto si desidera richiedere i messaggi tramite questo metodo, contattaci all'indirizzo supporto@trendoo.it). Per l'integrazione all'interno di un applicativo non web suggeriamo l'utilizzo di questo metodo.

getReceivedMessages(String username, String password, Date from, Date to)
: Array of WSSRMessage

richiede al server tutti gli SMS ricevuti in un particolare intervallo di date, utilizzando i parametri date_from e date_to.

**getReceivedMessagesById(String username, String password,
Long messageId) : Array of WSSRMessage**

richiede tutti gli SMS ricevuti che abbiano un identificativo univoco superiore a quello passato come parametro; tutti gli identificativi sono numeri interi maggiori di zero, e quindi, passando zero come parametro, si otterranno in risposta tutti gli SMS ricevuti dall'utente; il nome del parametro dell'identificativo è messageId.

Per ognuna delle richieste, la risposta assume sempre il medesimo formato:



```
WSSRMessage : class {  
    long messageId  
    String receiver  
    String sender  
    String messageText  
    Date smsDate  
    String keyword  
}
```

Dove il campo receiver è il numero di telefono della SIM sulla quale è stato ricevuto l'SMS, mentre il campo sender è il numero di telefono del mittente dell'SMS; il campo keyword contiene la prima parola del testo dell'SMS (per i servizi di ricezione condivisi).

E' inoltre possibile richiedere di poter ricevere i propri messaggi del servizio di ricezione tramite un HTTP/POST su un proprio indirizzo, i parametri passati sono:

```
[ id: identificativo univoco dell'SMS;  
[ text: testo dell'SMS;  
[ sender: mittente dell'SMS;  
[ recipient: destinatario dell'SMS, ovvero, il numero del proprio servizio di ricezione;  
[ delivery_date: data di ricezione dell'SMS.
```

8.GESTIONE SUBACCOUNTS

Un utente superaccount può utilizzare i seguenti metodi per verificare e gestire i propri subaccounts:

```
getSubaccounts(String username, String password)  
    : Array Of WsSubaccount
```

Restituisce una lista contenente tutti i subaccount dell'utente nel formato

```
WSSubaccount : class {  
    int credit_mode  
    String company_name  
    String fiscal_code  
    String vat_number  
    String name  
    String surname  
    String email  
    String address  
    String city  
    String province  
    String zip
```




```
String mobile  
String login  
String password  
boolean active  
}
```

Dove il campo `credit_mode` identifica il tipo di credito (0=credito del superaccount, 1=credito proprio, 2= pacchetti creati dal superaccount), mentre il campo `active` indica se il subaccount è attivo.

createSubaccount(String `username`, String `password`, WsSubaccount `subaccount`) :
WsSubaccount

crea e restituisce un nuovo subaccount; login e password sono generati da trendoo, e devono essere letti dall'oggetto WsSubaccount restituito

lockSubaccount(String `username`, String `password`, WsSubaccount `subaccount`) :
WsSubaccount

e

unlockSubaccount(String `username`, String `password`, WsSubaccount `subaccount`) :
WsSubaccount

permettono di attivare/disattivare un subaccount; in entrambi i casi viene restituito un WsSubaccount.

9. GESTIONE CREDITI SUBACCOUNTS

Un utente superaccount può dare/togliere crediti ai propri subaccounts.
In base alla tipologia di credito del subaccount, sono disponibili metodi differenti.

Ai subaccount con `credit_mode 0` non è possibile dare/togliere crediti.

Ai subaccount con `credit_mode 1` è possibile dare/togliere crediti tramite il metodo

moveCredits(String `username`, String `password`, WsCreditMovement `creditmovement`) :
void

Passando come parametro un movimento crediti, definito così:

```
WSCreditMovement : class {  
    String subaccount_login  
    boolean super_to_sub  
    int amount
```



```
String sms_type
String[] sms_types
double price
double[] pricePerMessage
boolean is_donation
long id_purchase
Date recording_date
int availableAmount
}
```

Dove il campo `super_to_sub` identifica la direzione del movimento (`true`=dal superaccount al sub, `false` viceversa), il campo `is_donation` deve essere `false`, i campi `sms_types` e `price_per_message` devono essere nulli, i campi `sms_type` e `amount` indicano il tipo di messaggio e la quantità spostati.

Per visualizzare i crediti del subaccount con `credit_mode 1` si utilizza il metodo

```
getSubaccountCredits(String username, String password, WsSubaccount subaccount) :
Array of WsCreditMovement
```

Ai subaccount con `credit_mode 2` è possibile dare crediti con il metodo

```
createPurchase(String username, String password, WsCreditMovement creditmovement) :
void
```

popolando i campi `sms_types` e `pricePerMessage` con tipi di messaggio e relativo prezzo, impostando `is_donation` a `true` e indicando in `amount` la quantità di crediti e in `price` il prezzo indicativo del pacchetto. Il campo `sms_type` va lasciato vuoto.

Ai subaccount con `credit_mode 2` è possibile togliere crediti con il metodo

```
deletePurchase(String username, String password, WsCreditMovement creditmovement) :
void
```

Per visualizzare gli acquisti del subaccount con `credit_mode 2` si utilizza il metodo

```
getPurchases(String username, String password, WsSubaccount subaccount) : Array of
WsCreditMovement
```

10. RICERCA E GESTIONE GRUPPI

ELENCO DI GRUPPI PRESENTI:

listGroups(String username, String password) : Array of Group

```
Group : class {  
    long id_group  
    String name  
    int group_type  
    long contactsCount  
}
```

AGGIUNTA DI UN GRUPPO:

addGroup(String username, String password, String name) : Group

MODIFICA DI UN GRUPPO:

**renameGroup(String username, String password,
Group group) : Group**

ELIMINAZIONE DI UN GRUPPO:

**delGroup(String username, String password,
Group group) : boolean**

I dati contenuti nel gruppo sono:

- [id_group: identificativo del gruppo;
- [name: nome del gruppo, minimo 2 caratteri, massimo 128 caratteri;
- [group_type: tipologia del gruppo(0: gruppo utente, 1: gruppo blacklist, 2: gruppo selfsubscribe);
- [contactsCount: numero di contatti contenuti nel gruppo.

Tutte le operazioni di list e get restituiscono il gruppo o i gruppi richiesti; le operazioni di add e rename restituiscono il gruppo creato o modificato; l'eliminazione restituisce true se la cancellazione è andata a buon fine, false altrimenti.

11. RICERCA CONTATTI

RICERCA DI CONTATTI IN UN GRUPPO:

**searchContacts(String username, String password, Group group,
String email, String phone, String name, String surname)**



: Array of Contact

```
Contact : class {  
    long id_contact  
    String email  
    String name  
    String surname  
    String phone_number  
    String address  
    String city  
    String province  
    String zip  
    Date birthdate  
    String sex  
}
```

ELENCO DI TUTTI I CONTATTI IN UN GRUPPO:

getGroupContacts(String username, String password, Group group)
: Array of Contact

I dati contenuti nel contatto sono:

- [id_contact: identificativo del contatto;
- [email: email del contatto (unico dato indispensabile), massimo 256 caratteri;
- [name: nome del contatto, massimo 64 caratteri;
- [surname: cognome del contatto, massimo 64 caratteri;
- [conferma, 3: non riceve email, 4: indirizzo non valido);
- [phone_number: telefono del contatto, massimo 22 caratteri;
- [address: indirizzo del contatto, massimo 128 caratteri;
- [city: città del contatto, massimo 32 caratteri;
- [province: provincia del contatto, massimo 32 caratteri;
- [zip: CAP del contatto, massimo 8 caratteri;
- [birthdate: data di nascita del contatto;
- [sex: sesso del contatto (M, F);

Il campo group non è obbligatorio; indica il gruppo all'interno del quale sono cercati i contatti.

Le operazioni restituiscono la lista di contatti che corrispondono a tutti i parametri di ricerca specificati.

12. GESTIONE CONTATTI

ELENCO DELL'INDIRIZZO EMAIL DI TUTTI I CONTATTI DI UN GRUPPO:

getGroupEmails(String **username**, String **password**,
Group **group**) : Array of String

ELENCO DEI NUMERI TELEFONICI DI TUTTI I CONTATTI DI UN GRUPPO:

getGroupPhones(String **username**, String **password**,
Group **group**) : Array of String

AGGIUNTA DI UN CONTATTO A UN GRUPPO:

groupAddContact(String **username**, String **password**,
Group **group**, Contact **contact**) : boolean

AGGIUNTA DI UN CONTATTO (SOLO INDIRIZZO EMAIL) A UN GRUPPO:

groupAddEmail(String **username**, String **password**,
Group **group**, String **email**) : boolean

AGGIUNTA DI UN CONTATTO (SOLO TELEFONO) A UN GRUPPO:

groupAddPhone(String **username**, String **password**,
Group **group**, String **phone**) : boolean

AGGIUNTA DI PIÙ CONTATTI A UN GRUPPO:

groupAddContacts(String **username**, String **password**,
Group **group**, Array of Contact **contacts**) : int

MODIFICA DI UN CONTATTO:

updateContact(String **username**, String **password**,
Contact **contact**) : Contact

ELIMINAZIONE DI UN CONTATTO DA UN GRUPPO:

groupDelContact(String **username**, String **password**,
Group **group**, Contact **contact**) : boolean

ELIMINAZIONE DI UN CONTATTO (IDENTIFICATO DAL SOLO INDIRIZZO EMAIL) DA UN GRUPPO:

groupDelEmail(String **username**, String **password**,

Group **group**, String **email**) : boolean

ELIMINAZIONE DI UN CONTATTO (IDENTIFICATO DAL SOLO TELEFONO) DA UN GRUPPO:

groupDelPhone(String **username**, String **password**,
Group **group**, String **phone**) : boolean

ELIMINAZIONE DI UN CONTATTO DA TUTTI I GRUPPI:

deleteContact(String **username**, String **password**,
Contact **contact**) : boolean

ELIMINAZIONE DI UN CONTATTO (IDENTIFICATO DAL SOLO INDIRIZZO EMAIL) DA TUTTI I GRUPPI:

deleteEmail(String **username**, String **password**,
String **email**) : boolean

ELIMINAZIONE DI UN CONTATTO (IDENTIFICATO DAL SOLO TELEFONO) DA TUTTI I GRUPPI:

deletePhone(String **username**, String **password**,
String **phone**) : boolean

ELIMINAZIONE DI PIÙ CONTATTI DA TUTTI I GRUPPI:

deleteContacts(String **username**, String **password**,
Array of Contact **contacts**) : int

ELIMINAZIONE DI PIÙ CONTATTI (IDENTIFICATI DAI SOLI INDIRIZZI EMAIL) DA TUTTI I GRUPPI:

deleteEmails(String **username**, String **password**,
Array of String **emails**) : int

ELIMINAZIONE DI PIÙ CONTATTI (IDENTIFICATI DAI SOLI TELEFONI) DA TUTTI I GRUPPI:

deletePhones(String **username**, String **password**,
Array of String **phones**) : int

Tutti i parametri sono obbligatori.

Le operazioni di get restituiscono la lista di email o di numeri telefonici dei contatti presenti nel gruppo.

L'operazione di aggiornamento restituisce il contatto modificato.

Le operazioni di aggiunta/eliminazione singolo contatto restituiscono true se l'operazione è andata a buon fine, false altrimenti.



Le operazioni di aggiunta/eliminazione di più contatti restituiscono il numero di contatti aggiunti/eliminati.

Se si prova ad associare un contatto in un gruppo nel quale è già presente un contatto con la stessa email/telefono, il nuovo contatto non viene associato; non è quindi possibile avere indirizzi email o numeri telefonici duplicati all'interno di uno stesso gruppo.

Integrazione Contatti via Webservice

1. INTRODUZIONE

I server Trendoo mettono a disposizione degli sviluppatori un'interfaccia via Webservice per l'invio e la ricezione dei messaggi SMS attraverso il gateway trendoo, e permettono inoltre di recuperare tutti i dati riguardanti lo stato dei messaggi e lo storico SMS.

Nel caso fosse necessaria una connessione criptata, è possibile effettuare le stesse richieste utilizzando la connessione https.

2. FORMATO DEI PARAMETRI

- [il WSDL del Webservice può essere raggiunto all'indirizzo:
<https://api.trendoo.it/trendoows?wsdl>
- [il Webservice è compatibile SOAP 1.1, per massima compatibilità con le tutte le versioni;
- [tutte le richieste Webservice devono avere due parametri obbligatori username e password, che rappresentano le credenziali personali di accesso al portale trendoo;
- [un numero di telefono in formato internazionale è una qualunque sequenza di cifre preceduta da un carattere '+' (esempio, +393471234567);
- [tutte le richieste che non vanno a buon fine lanciano un Webservice "Fault" (ad es. in ambiente .NET è System.ServiceModel.FaultException) contenente il codice dell'errore ed una descrizione testuale nel formato:
<numeric-errorcode>:<errormessage>
- [i codici nazione utilizzati sono i codici a 2 caratteri standard ISO 3166
http://www.iso.org/iso/english_country_names_and_code_elements
- [il charset utilizzato è **UTF-8**.

3. GESTIONE GRUPPI

Vengono qui elencate le API per la gestione dei gruppi. Tutte le operazioni di list e get restituiscono il gruppo o i gruppi richiesti; le operazioni di add e rename restituiscono il gruppo creato o modificato; l'eliminazione restituisce true se la cancellazione è andata a buon fine, false altrimenti.

In seguito si farà riferimento ad oggetti di tipo "**Group**" così definiti:

```
Group : class {  
    long id_group  
    String name  
    long contactsCount  
}
```




I dati contenuti nel gruppo sono:

- [id_group: identificativo del gruppo;
- [name: nome del gruppo, minimo 2 caratteri, massimo 128 caratteri;
- [contactsCount: numero di contatti contenuti nel gruppo.

Elenco di gruppi presenti:

listGroups(String username, String password)

out: Array di oggetti Group

Aggiunta di un gruppo:

addGroup(String username, String password, String name)

out: oggetto Group

Modifica di un gruppo:

renameGroup(String username, String password, Group group)

out: oggetto Group

Eliminazione di un gruppo:

delGroup(String username, String password, Group group)

out: boolean

4. GESTIONE CONTATTI

Vengono qui elencate le API per la gestione dei gruppi. Tutti i parametri sono obbligatori. Le operazioni di get restituiscono la lista di email o di numeri telefonici dei contatti presenti nel gruppo.

L'operazione di aggiunta o aggiornamento di un singolo contatto restituisce il contatto modificato o aggiunto.

L'operazione di eliminazione singolo contatto restituisce true se l'operazione è andata a buon fine, false altrimenti.

Le operazioni di aggiunta/eliminazione di più contatti restituiscono il numero di contatti aggiunti/eliminati.

Elenco dell'indirizzo email di tutti i contatti di un gruppo:

getGroupEmails(String username, String password, Group group)

out: Array of String

Elenco dei numeri telefonici di tutti i contatti di un gruppo:

getGroupPhones(String username, String password, Group group)

out: Array of String

Aggiunta di un contatto a un gruppo:

groupAddContact(String username, String password,
Group group, Contact contact, boolean sendOptIn)

out: Contact

Aggiunta di un contatto (solo indirizzo email) a un gruppo:

groupAddEmail(String **username**, String **password**,
Group **group**, String **email**, boolean **sendOptIn**)

out: Contact

Aggiunta di un contatto (solo TELEFONO) a un gruppo:

groupAddPhone(String **username**, String **password**,
Group **group**, String **phone**)

out: Contact

Aggiunta di più contatti a un gruppo:

groupAddContacts(String **username**, String **password**,
Group **group**, Array of Contact **contacts**, boolean **sendOptIn**)

out: int

Modifica di un contatto:

updateContact(String **username**, String **password**,
Contact **contact**)

out: Contact

Eliminazione di un contatto da un gruppo:

groupDelContact(String **username**, String **password**,
Group **group**, Contact **contact**)

out: boolean

Eliminazione di un contatto (identificato dal solo indirizzo email) da un gruppo:

groupDelEmail(String **username**, String **password**,
Group **group**, String **email**)

out: boolean

Eliminazione di un contatto (identificato dal solo telefono) da un gruppo:

groupDelPhone(String **username**, String **password**,
Group **group**, String **phone**)

out: boolean

Eliminazione di un contatto:

deleteContact(String **username**, String **password**, Contact **contact**)

out: boolean

Eliminazione di un contatto (identificato dal solo indirizzo email):

deleteEmail(String **username**, String **password**, String **email**)

out: boolean

Eliminazione di un contatto (identificato dal solo telefono):

deletePhone(String **username**, String **password**, String **phone**)

out: boolean

Eliminazione di più contatti:

deleteContacts(String **username**, String **password**,
Array of Contact **contacts**)

out: int

Eliminazione di più contatti (identificati dai soli indirizzi email):

deleteEmails(String **username**, String **password**, Array of String **emails**)

out: int

Eliminazione di più contatti (identificati dai soli telefoni):

deletePhones(String **username**, String **password**, Array of String **phones**)

out: int

5. GESTIONE CAMPAGNE EMAIL

Vengono qui elencate le API per la gestione delle campagne email. In seguito si farà riferimento agli oggetti Campaign e CampaignIssue così definiti:

Campaign : class {

```
    long id_campaign
    String label
    String title
    String id_language
    String emailSender
    String sender
    String emailReplyTo
    String company
    String address
    String url
    String copyright
    String phone
    String fax
    boolean sendToFriend
    String optInSubject
    int status
```

}

- I dati contenuti nell'oggetto Campaign sono:
- **id_campaign**: identificativo numerico della campagna
- **label**: Identificativo testuale della campagna
- **title**: Nome della campagna
- **id_language**: lingua in formato ISO 639-2
- **sender**: mittente della campagna
- **emailSender**: indirizzo email del mittente della campagna
- **emailReplyTo**: indirizzo email di risposta
- **company**: Nome azienda

- address: Indirizzo azienda
- url: indirizzo del sito web
- copyright: copyright
- phone: Numero di telefono
- fax: Numero di fax
- sendToFriend: se true il footer della campagna conterrà il link “inoltra ad un amico”
- optInSubject: Subject utilizzato per le mail di opt-in
- status: stato della campagna (ignorato in input)

```
CampaignIssue : class {  
    long id_issue  
    long id_campaign  
    String webAnalytics  
    String messageText  
    String messageHTML  
    String subject  
    String url_editor  
}
```

I dati contenuti nell'oggetto CampaignIssue sono:

- id_issue: identificativo numerico dell'invio
- id_campaign: identificativo numerico della campagna a cui è associato l'invio
- webAnalytics: parametri web-analytics da acodare ai link presenti nella mail (facoltativo)
- messageText: Corpo della mail in formato testuale codificato base64 (opzionale)
- messageHTML: Corpo della mail in formato HTML codificato base64 (opzionale)
- subject: oggetto della mail (Opzionale)
- url_editor: link alla pagina per la composizione del messaggio e per la gestione dell'invio.

Elenco delle campagne attive

getActiveCampaign(String username, String password)

out: Array of Campaign

Creazione di un invio email associato ad una campagna

**getGroupEmails(String username, String password,
Array of Group group, CampaignIssue issue)**

out: CampaignIssue

Codici di Errore

//COMMON ERRORS

AUTHENTICATION_ERROR = 1,"Invalid username or password"
INSUFFICIENT_CREDIT = 2,"Insufficient credit"
TOO_MANY_SIMULTANEOUS_REQUESTS = 3,"Too many simultaneous requests"
MISSING_PARAMETER = 4,"Parameter '\$1\$' must be specified"
INVALID_PARAMETER_VALUE = 5,"Value '\$2\$' is invalid for parameter '\$1\$'"
ONE_INVALID_PARAMETER = 5,"One of the parameters specified is invalid, please see manual for details"
DATE_OUT_OF_RANGE = 6,"Specified date of parameter \$1\$ is out of range"
SYSTEM_UNAVAILABLE = 7,"System unavailable. Please, try again later. Thank you."
INVALID_SERVICE = 11,"Invalid service"
INVALID_AGREEMENT_VERSION = 12,"Invalid legal agreement version"
INVALID_NATION_SERVICE = 13,"Invalid service for user nation"
INVALID_DATA = 14,"Invalid data"
TPOA_SPECIFIED_NOT_YET_VALIDATED = 15, "Sender specified not yet validated by Agcom"

// SMS ERRORS

TPOA_SPECIFIED_FOR_NON_CUSTOM_TPOA_MT = 8,"The message type you specified has no custom sender, but you specified it"
MESSAGE_TOO_LONG = 9,"SMS text too long!"

// MAIL ERRORS

INVALID_GROUP = 50,"Invalid group for user"
INVALID_GROUP_NAME = 51,"Invalid group name \$1\$"
DUPLICATE_GROUP_NAME = 52,"A group with name \$1\$ already exists. No duplicates allowed!"
READ_ONLY_GROUP = 53,"Impossible to write on this group. Permission denied"
INVALID_CONTACT= 54,"Invalid contact(s)"
INVALID_CONTACT_ATTRIBUTES = 55,"Invalid contact attributes"
NO_CAMPAIGNS_FOUND = 56,"No campaigns found"
INVALID_CAMPAIGN = 57,"Invalid campaign for user"
INVALID_SENDER_EMAIL_ADDRESS = 58,"Invalid sender email address"
INVALID_CAMPAIGN_ISSUE = 59,"Invalid campaign issue"
ERROR_CONTACT_LOCK = 60,"Impossible to acquire lock on contacts"

// SUBACCOUNT ERRORS

USER_NOT_SUPERACCOUNT = 101,"User is not superaccount"
WRONG_SUBACCOUNT_TYPE = 102,"Invalid subaccount type"
NO_MORE_SUBACCOUNTS = 103,"You have reached your subaccounts limit, please contact us to have more subaccounts!"
NO_CREDIT = 104,"No credit to delete"

//TPOA AGCOM ERRORS

TPOA_VALIDATION_NOT_ALLOWED = 201,"TPOA Validation not allowed"
INVALID_USER_TPOA = 202,"Invalid TPOA for user"
INVALID_AGCOM_TPOA = 203,"Invalid AGCOM TPOA \$1\$ "
DUPLICATE_AGCOM_TPOA = 204,"Duplicate AGCOM TPOA \$1\$ "