JackSMS API

Introduzione

Questo documento tratta le Web API di JackSMS per lo sviluppo di nuovi client da interfacciare al server.

Si tratta di Web API, quindi il protocollo utilizzato da client e server sarà quello HTTP.

L'host verso il quale effettuare le richieste è `q.jacksms.it`.

Ogni richiesta sarà quindi una richiesta HTTP, ed avrà una delle seguenti forme:

```
GET /login/action?outputFormat,clientVersion HTTP/1.1
Host: q.jacksms.it
[Eventuali headers]
```

Oppure

```
POST /login/action?outputFormat,clientVersion HTTP/1.1
Host: q.jacksms.it
[Eventuali headers]
```

`dati`

Login

Ogni chiamata alle API necessita di un parametro di identificazione, trasmesso come path dell'URL richiesto:

```
GET /login_string/commandAction HTTP/1.1
Host: q.jacksms.it
```

Login tramite username e password

L'autenticazione tramite username e password avviene mediante la creazione di una stringa formata dalla codifica in base64 di username e password, separati da un carattere di slash (/).

base64_encode(username) / base64_encode(password)

Esempio:

In caso di un utente con username `guest` e password `passwd`, la richiesta da forumlare sarà la seguente:

```
GET /Z3Vlc3Q=/cGFzc3dk/cmd HTTP/1.1
Host: q.jacksms.it
```

L'unica accortezza è quella di sostituire, dopo la codifica, ogni eventuale carattere + con un carattere -, ed ogni eventuale carattere / con un carattere _

Pseudocodice:

```
username = base64('username').replace('+','-').replace('/','_');
password = base64('password').replace('+','-').replace('/','_');
```

Login tramite session_id

Se il client ha memorizzato un session_id, ottenuto dal server tramite una precedente autenticazione mediante username e password, questi può essere usato al posto del metodo descritto precedentemente, come segue:

```
GET /session_id/cmd HTTP/1.1
Host: q.jacksms.it
```

I vantaggi di usare il session_id sono:

- Non si inviano i dati dell'utente in chiaro
- Il server è meno stressato.

C'è invece lo svantaggio di dover gestire un'eventuale sessione scaduta.Questo metodo di login dovrebbe essere implementato solo da client usati sul computer

Output Format

Il parametro `outputFormat` passato all'URL, opzionale con valore di default `csv`, determina il formato in cui verranno restituiti i dati. I valori possibili per questo parametro sono:

`csv`

`xml`

`jsn`

ClientVersion

Il parametro `clientVersion`, passato all'url, opzionale e con valore di default `mobile`, determina la versione del client che sta accedendo alle API. I valori possibili corrispondono alle versioni di JackSMS esistenti, e sono:

```
mobile

desktop

web

igoogle

wap

iphone

symbian
```

Comandi disponibili

getAbook

Restituisce la rubrica dell'utente in format minmale

Output

```
        <nome, numero>
```

getAbookFull

Restituisce la rubrica dell'utente comprensiva di tutti i dati

Output

```
<id_contatto, nome, numero, account_associato>
```

countQueued

Restituisce il numero di messaggi in coda nell'Instant Messenger

Output

```
<N>
```

getServices

Restituisce gli account salvati dall'utente, in formato minimale

Output

```
<service_id, nome_account, username_account, password_account,
dato3_account, dato4_account>
```

getServicesFull

Restituisce gli account salvati dall'utente, in formato completo

Output

```
<account_id, service_id, nome_account, username_account,
password_account, dato3_account, dato4_account>
```

Esempio:

```
http://q.jacksms.it/guest/guest/getAbookFull?csv

abook   15644   Test    3471234567      0

abook   53446   ciao    +39.347.1234567 1

abook   53447   ciao    +39.347.1234567 1

abook   53448   ciao    +39.347.1234567 1

abook   53449   ciao    +39.347.1234567 1

abook   53450   ciao    +39.347.1234567 1

abook   53451   ciao    +39.347.1234567 1
```

sendMessage

Il comando per inviare SMS, il più articolato.

Attraverso l'header J-R si inviano, separati da TAB (\t) l'id del servizio con cui inviare, il destinatario, i dati di accesso al servizio.

Attraverso l'hader J-M si invia il testo del messaggio.

Esempio:

```
GET /guest/guest/sendMessage HTTP/1.1

Host: q.jacksms.it
```

```
J-R: 1    +39.347.1234567    username    password    _    _

J-M: ciao, testo del messaggio
```

Output

```
<1, eventuale testo messaggio inviato, messaggi in coda>

<0, eventuale testo errore, messaggi in coda>

<session_id, Buffer immagine captcha codificata in base64>
```

Nel terzo caso si deve mostrare l'immagine e successivamente usare il comando /continueSend

continueSend

Il comando per completare l'invio di un SMS dopo l'inserimento del codice ottico.

Attraverso l'hader J-R si inviano, separati da TAB, il session_id e il valore del CAPTCHA.

Esempio:

```
GET /guest/guest/continueSend HTTP/1.1
```

```
Host: q.jacksms.it

J-R: 12345    cb4r2
```

Output

```
<1, eventuale testo messaggio inviato, messaggi in coda>

<0, eventuale testo errore, messaggi in coda>
```

getProviders

Il comando per ottenere la lista dei servizi supportati. il parametro opzionale exclude, inviato via post, esclude determinati servizi dall'essere restituiti.

exclude=id,id,id...

Esempio:

```
POST /guest/guest/continueSend HTTP/1.1

Host: q.jacksms.it


exclude=1,2,3
```

Output

<id_servizio, nome_servizio, lunghezza_massima_messaggio, nome_parametro_1, nome_parametro_2, nome_parametro_3, nome_parametro_4, descrizione_servizio>