



MAILUP INTEGRATION PROCEDURES

MailUp Version: **v7.1**

Document Version: **v018**

Revision: **N. Gorni**

Authors: **A. Miscia, N. Gorni, M. Nizzotti, S. Casotti**

Editors: **E. A. Murphy**

Last Update: **February 16th, 2010**



SUMMARY

- 1 DEFINITIONS..... 4
- 2 SUBSCRIPTION FORM AND STANDARD APIs 6
 - 2.1 GET / POST ACTION APIs.....6
 - 2.2 XML API FOR WINDOWS SERVER.....8
 - 2.3 XML API FOR NON WINDOWS12
- 3 BATCH FTP INTEGRATION 15
- 4 POST OPERATIONS..... 17
- 5 WEB SERVICES 18
 - MAILUP API STATUS.....18
 - WEB SERVICE AUTHENTICATION19
 - WEB SERVICE METHODS19
 - API ACTIVATION or REGISTRATION19
 - WS_MAILUPIMPORT21
 - WS_MailUpImport.GetNLists method21
 - WSMailUpImport.CreateGroup method22
 - WSMailUpImport.NewImportProcess method23
 - WSMailUpImport.GetProcessDetails method.....24
 - WSMailUpImport.StartProcess method.....24
 - WSMailUpImport.StartImportProcesses method25
 - Appendix A – XML FEED28
 - Check queue status28
 - Read MailUp Statistics in an external IFRAME.....29
 - WS_MAILUPSEND (SOAP)30
 - Web service: WS_MailupSend31
 - WebService MailupSend31
 - Login31
 - Logout.....31
 - GetLists32
 - GetGroups.....32
 - Method GetTemplates33
 - Method GetNewsletters.....33
 - Method CreateNewsletter34
 - Method SendNewsletter35
 - Method SendNewsletterFast36
 - Method GetNewsletterDeliveryStatus37
 - Method StartDelivery.....37
 - Method StopDelivery38
 - Method SendSingleNewsletter.....38
 - Method GetNewsletterQueues39
 - Method RemoveNewsletterQueue.....40
 - Method ScheduleNewsletterQueue41
 - Method GetSMS.....41
 - Method CreateSMS.....42
 - Method SendSMS.....42
 - Method SendSMSFast.....43
 - Method SendSingleSMS44
 - Method GetSMSDeliveryStatus.....44

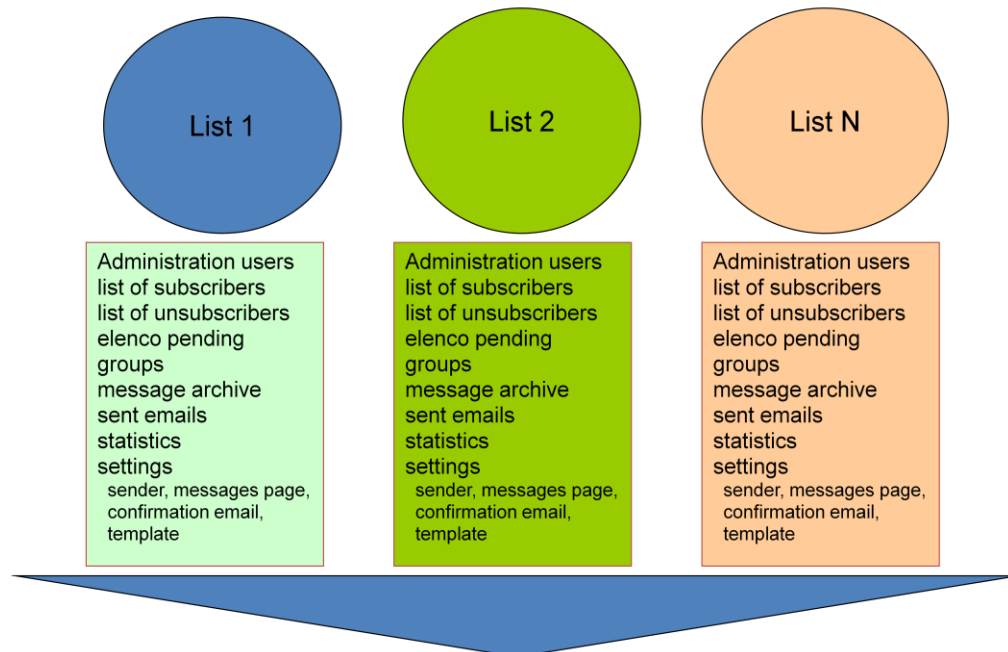
- Method StartDeliverySMS*.....45
- Method StopDeliverySMS*45
- Method GetSMSQueues*.....46
- Method RemoveSMSQueue*47
- Method ScheduleNewsletterQueue*47
- Method GetCredits*.....47
- WEBSERVICE MAILUPMANAGE (SOAP)48
 - Method Login*.....48
 - Logout*.....48
 - Method GetLists*.....49
 - Method GetGroups*49
 - Method CreateList*50
 - Method UpdateList*.....52
 - Method DeleteList*.....53
 - Method CreateGroup*.....54
 - Method UpdateGroup*.....54
 - Method DeleteGroup*55
- WEBSERVICE MAILUPREPORT (SOAP)56
 - Method Login*.....56
 - Method Logout*56
 - Method CreateXmlStatistiche*.....56
 - Method ReportByUser*57
 - Method ReportByMessage*58
- MAILUP SEND SAMPLES58
 - Codice C#*.....58
 - logout: C#*.....59
 - Get lists: C#*60
 - Create a message in C#*.....60
- APPENDIX.....61
- OTHERS OUTPUT SAMPLES.....63
 - Method GetNewsletterDelivery*63
- 6 HELP DESK REFERENCES..... 65

*** ALL THE APIs AND INTEGRATIONS ARE FREE OF CHARGE ***

1 DEFINITIONS

In MailUp data is structured in independent LISTS, which are isolated one another, and the key is the email address. LISTS are useful because they allow to manage different clients/portals with the same console, or different newsletters also in different languages. Inside each LIST there are all the settings (dialog messages with users when they register or cancel), contact lists, messages, statistic reports.

1.a Mailup Console (es. host newsletter.companyXY.com)



1 Queue for sent emails (with bandwidth fixed by contract)

LIST = An independent grouping of personal data, newsletter archives, senders, statistics, automatic email messages, pending lists, subscribers lists, group lists, unsubscribers lists.

NEWSLETTER = A single email message to send.

OPTIN = It refers to all subscribed users. They appear as "enabled" in their user's form. Their status can be changed manually.

OPTOUT = It refers to all unsubscribed users. They appear as "disabled" in their user's form.

PENDING = Refers to all pending users. They registered on the portal and received a confirmation request email (double opt-in), but they haven't clicked yet on the subscription link.

GROUPS = It's a set of aspects/qualities that can be activated or deactivated for each user. Any user can belong to different groups at the same time (as happens with "Categories" in MS

Outlook). A group can be created or deleted at any time. A group can also be deleted by cancelling it's user list. During the import phase, it is possible to import lists of users in one or more groups. If the user already was registered in other groups, he will be updated also in the other assigned groups (unless the "REPLACE GROUP" feature is selected).

FILTERS = They can be applied to the personal data fields and allow to dynamically subdivide users, following preset conditions. Groups and filters can be complementary: using them together can help profiling users in any possible way. It is possible to save and move a list of users generated from a combination of filters in a new group by using the "ADVANCED LIST" feature.

PERSONAL DATA FIELDS = Every user registered to a newsletter can have up to 40 personal data fields, which can be max 100 characters long. The choice of the number and of what fieldsto label is up to the client during the first setup. Every user can always be part of endless LISTS and of endless GROUPS.

EMAIL = it is the key field in the MailUp database. An email can belong to more than one LIST and can be cancelled from each single group (except multiple double opt-out procedures).

***Example of a user personal file.** In every LIST a user can have a different personal file, with different data. The Group becomes a flag of the single user file.*

Edit subscriber

<u>NAME VIEWED</u>	May
<u>EMAIL</u>	corbel@hotmail.com
<u>LIST</u>	News
<u>USER STATUS</u>	Enabled
<u>USER CODE</u>	{B1E92AE4-B9B4-4B96-B00B-84F01F50AA75}
<u>GROUPS</u>	<input type="checkbox"/> First Importation <input type="checkbox"/> Group A <input checked="" type="checkbox"/> Test <input type="checkbox"/> Test Group

PERSONAL INFORMATION:

First name

Second name

Company

Address

EDIT

2 SUBSCRIPTION FORM AND STANDARD APIs

2.1 GET / POST ACTION APIs

It's the form that allows users to register themselves to the newsletters, through the double opt-in method. This form can be generated automatically and can be inserted in any site (Linux, Windows..). Included is also a check-up to verify that the format of the email written in the form is correct. In the MailUp 5.5 console and superior there are a number of forms already available and ready to use on the client's site. To activate or deactivate them go to the Settings / Edit Lists.

- a. In the Settings menu of MailUp see the voice "SUBSCRIPTION FORMS" to find examples of preset and editable forms. E.G., you can use the first form to register a new user in a list or in a group only by the email address (the Javascript check is included).
- b. The form has to be structured as shown (the field "name" can be obtained by the TABLE OF CODES in the Settings menu of MailUp console). It's important to note that the type, for all parameters, is completely arbitrary: any kind of type can be set.

- First part of form:

```
<form method="post" action="http://example.mailupnet.it/frontend/subscribe.aspx">
```

- Email field (mandatory):

```
<input name="email" type="text" size="60" maxlength="100">
```

- Fields of the distribution lists (mandatory):

```
<input type="hidden" name="list" value="1">
```

Value is the List Code. See images below.

When the user should choose between registering to different lists it is possible to list them using the same name for the different field. E.g.:

```
<input name="list" type="checkbox" value="2">Second list <br>  
<input name="list" type="checkbox" value="3">Third list <br>
```

In case that the user has registered to more than one list only one confirmation email will be sent. As a rule, the confirmation email and the message pages used will be the ones of the first list specified. In the previous example, it is on the List 1.

- Groups (optional):

```
<input name="group" type="checkbox" value="1"> First group <br>  
<input name="group" type="checkbox" value="2"> Second group <br>
```

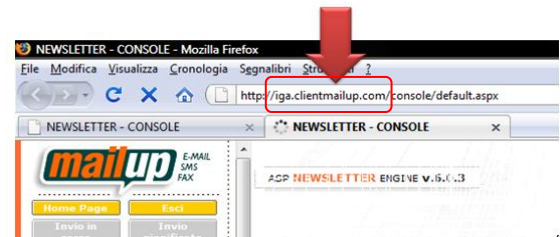
Value is the Group. See images below.

- Personal data (optional):

<input name="campo1" type="text" size="60" maxlength="100"> First name
 <input name="campo1" type="text" size="60" maxlength="100"> Last name

In the form it's the Client's responsibility to set the fields, tents, colors, styles, checkboxes, controls, freely based on one's needs. The fields which are not necessary can be removed by the registration form. If there are extra fields with names which are not recognized, they will be ignored.

(* Note: <http://example.mailupnet.it/frontend/> must be replaced with the third level host as defined in the MailUp setup form. (or customer.mailupnet.info)



Distribution Lists	
LIST CODE	LIST NAME
1	NEWS
2	TEXT

Group list		
LIST CODE	GROUP CODE	GROUP NAME
1	1	TEST
1	2	TEST GROUP
1	3	FIRST IMPORTATION
1	4	GROUP A

List of Personal Data Fields	
FIELD CODE	FIELD NAME
CAMPO1	FIRST NAME
CAMPO2	SECOND NAME
CAMPO3	COMPANY
CAMPO4	ADDRESS

Example of TABLE OF CODES. To open it click on **Settings** in the MailUp console and click "TABLE OF CODES".

2.2 XML API FOR WINDOWS SERVER

In the event that the client's portal already has its own subscription form linked to other databases (e.g. to register in a database, to access reserved areas, to authenticate a user) it's possible to integrate to the page's code the following code, so that new subscribers will be registered at the same time also in the MailUp's database, with or without confirmation request.

ASP 3.0 Version

```
<%@LANGUAGE="VBSCRIPT"%>
<%
on error resume next
```

```
Dim strEmail, intList, intGroup
Dim xml, url, response_val
```

```
strEmail=request("email") ' e-mail of user
intList=request("list") ' the list
intGroup=request("group") ' the group where the user will be registered (optional)
bInConfirm = TRUE 'confirmation request
csvFldNames = "field1; field 2; field 3; field " 'The names of the fields are in the Table of Codes
csvFldValues = "First Name;Last Name;Company;M" 'The values of the fields have to be assigned in the same
order of the csvFldNames
retCode = 1 'if retCode = 1 the value of the request return i s an error code, if retCode = 0 the error text is given
back
```

```
The page that will be called on our servers
url=" http://customer.mailup.info/frontend/xmlSubscribe.aspx"
```

```
' XMLHTTP version 3.0:
Set xml = Server.CreateObject("MSXML2.ServerXMLHTTP")
xml.Open "POST",url,false
```

```
'To send the request
xml.setRequestHeader "Content-Type", "application/x-www-form-urlencoded"
xml.Send "email=" & strEmail & "&list=" & intList & "&group=" & intGroup & "&confirm=" & bInConfirm &
"&csvFldNames=" & csvFldNames& "&csvFldValues=" & csvFldValues & "&retCode=" & retCode
```

```
'If the parameter retCode=1 is set, the return values will be:
```

```
'0 : Operation completed
```

```
'1 : Generic error
```

```
'2 : Invalid Address
```

```
'3 : Already subscribed user
```

```
if err.number=0 then
```

```
    'Value returned from the call
```

```
    response_val=xml.responseText
```

```
    Set xml = Nothing
```

```
else
```

```
    'Call had negative result (E.g. Network error )
```

```
    response_val=1
```

```
    Set xml=nothing
```

```
end if
```

'Once the return value is obtained it is possible to decide what to do/visualize

```
Select Case response_val
```

```
Case 0
```

```
' Display a message stating that User subscribed successfully
```

```
' ...
```

```
Case 1
```

```
' Display a message stating that Service is not accessible at the moment
```

```
' ...
```

```
Case 2
```

```
' Display a message stating that the email address is invalid
```

```
' ...
```

```
Case 3
```

```
' Display a message stating that User is already subscribed to the list
```

```
Case Else
```

```
' Display a message stating that Service is temporary unavailable
```

```
End Select
```

'The parameter retCode=0 is set the error text is returned

'Recovery of return value

```
response_val=xml.responseText
```

```
Set xml = Nothing
```

```
%>
```

.NET VERSION (C#)

The following code has been developed using the C#

```
<script runat="server" language="C#">
```

```
void Subscribe()
```

```
{
```

```
string retCode = "1"; // if retCode = 1 return value is an error code, retCode=0 if return value is a message
```

```
string ret_val = SubscribeUser(retCode);
```

```
// Once the return value is obtained it is possible to decide what to do/visualize
```

```
// *****
```

```
// If the parameter retCode=1 is set, the return values will be:
```

```
//0 : Operation completed
```

```
//1 : Generic error
```

```
//2 : Invalid Address
```

```
//3 : Already subscribed user
```

```
// *****
```

```
switch(ret_val)
```

```
{
```

```
case "0":
```

```
        // Display a message stating that User subscribed successfully
        break;
case "1":
    // Display a message stating that Service is not accessible at the moment
break;
case "2":
    // Display a message stating that the email address is invalid
    break;
case "3":
    // Display a message stating that User is already subscribed to the list
    break;
default:
    // Display a message stating that Service is temporary unavailable
    break;
}

// *****
// If the parameter retCode=0 is set, the text of the error is returned
// *****
//The value ret_val can be associated for example to a Label object
}
// This feature calls the user registration page and gives back the return value
string SubscribeUser(string retCode)
{
    string strEmail = Request.Params["email"]; // user e-mail
    string intList = Request.Params["list"]; // list
    string intGroup = Request.Params["group"]; // group (optional)
    bool blnConfirm = true; //Confirmation request
    string csvFldNames = "field1;field2;field3;field4"; //The names of the fields are in the Table of Codes
    string csvFldValues = " First Name;Last Name;Company;M" 'The values of the fields have to be assigned in
    the same order of the csvFldNames
    string result = ""; // return value

    string url = "http://newsletter.nomedominio.tld/frontend/xmlSubscribe.aspx";
    url += "?list=" + intList + "&group=" + intGroup + "&email=" + strEmail + "&confirm=" + blnConfirm.ToString()
    + "&csvFldNames=" + csvFldNames + "&csvFldValues=" + csvFldValues + "&retCode=" + retCode;

    System.Net.HttpWebRequest wreq = (System.Net.HttpWebRequest)System.Net.WebRequest.Create(url);
    wreq.Method = "GET";
    wreq.Timeout = 10000;

    System.Net.HttpWebResponse wr = (System.Net.HttpWebResponse)wreq.GetResponse();

    // If the answer of the page is correct it is possible to deduce the return value
    if (wr.StatusCode == System.Net.HttpStatusCode.OK)
    {
        System.IO.Stream s = wr.GetResponseStream();
        System.Text.Encoding enc = System.Text.Encoding.GetEncoding("utf-8");
        System.IO.StreamReader readStream = new System.IO.StreamReader( s, enc );

    // Return value
        result = readStream.ReadToEnd();
    }
}
```

```
return result;  
}  
</script>
```

The procedures that can be called externally through http are the following:

- **User subscription**

URL: <http://newsletter.nomedominio.tld/frontend/xmlSubscribe.aspx>

(MAN = Mandatory)

(1) Email and/or sms: at least one is mandatory.

PARAMETERS AT ENTRANCE:

NAME	MAN	DESCRIPTION
Email	NO (1)	Email address of user
List	YES	Distribution list
sms	NO (1)	Mobile phone number including International prefix
Group	NO	Groups in which the user can be registered
Confirm	NO	Parameter to enable –or not – the email confirmation request Accepted Values: 0/1/false/true – Default Values: true
csvFldNames	NO	Codes of the personal data fields (separated by ;)
csvFldValues	NO	Values that correspond to the codes of the personal data fields (separated by ;)
retCode	NO	Parameter to enable – or not – the return of the return code of the call Accepted Values: 0/1 – Default Values: 0

RETURNED VALUES:

CODE	DESCRIPTION
0	Operation completed with success
1	Generic error
2	Invalid email address
3	User already subscribed

- **User unsubscription**

URL: <http://newsletter.nomedominio.tld/frontend/xmlUnSubscribe.aspx>

PARAMETERS AT ENTRANCE:

NAME	MAN	DESCRIPTION
ListGuid	YES	Alphanumeric code of the distribution list
List	YES	List code of distribution
Email	NO (1)	Address of user to remove

sms	NO (1)	Mobile phone number including International prefix
-----	--------	--

RETURNED VALUES:

CODE	DESCRIPTION
1	Generic error
2	User unsubscribes succesfully
3	User not existent / already unsubscribed

NOTE: In case of positive out come the user will be moved to the opt-out and will receive a confirmation mail where needed.

- Edit User

URL: <http://newsletter.nomedominio.tld/frontend/xmlUpdSubscriber.aspx>

PARAMETERS AT ENTRANCE:

NAME	MAN	DESCRIPTION
ListGuid	YES	Alphanumeric code of the distribution list
List	YES	List code of distribution
sms	NO (1)	Mobile phone number including International prefix
Email	NO (1)	Address of user to edit
Replace	NO	Parameter to enable -or not - the substitution of groups for the user in question Accepted Values: 0/1/false/true - Default Values: false
Group	NO	Groups in which the user can be registered
csvFldNames	NO	Codes of the personal data fields (separated by ;)
csvFldValues	NO	Values that correspond to the codes of the personal data fields (separated by ;)

RETURNED VALUES:

CODICE	DESCRIZIONE
0	Change completed succesfully
1	Generic error

2.3 XML API FOR NON WINDOWS

In the event that the client's portal already has its own subscription form linked to other databases (e.g. to register in a database, to access reserved areas, to authenticate a user) it's possible to integrate to the page's code the following code, so that new subscribers will be registered at the same time also in the MailUp's database, with or without confirmation request.

Necessary calls to register a contact

To register a new receiver it is necessary to send a call (both POST and GET) to the following address:

<http://newsletter.domainname.tld/frontend/xmlSubscribe.aspx>

You must insert parameters as follows (in any order):

- List: code list * (Mandatory)
- Group: code of the group where the user is subscribed * (Optional)
- E-mail: addressee's email address (Mandatory)
- Confirm: Boolean value which indicates whether the user is subscribed directly (false) or in the confirmation email is to be sent (true) (optional: default: false)
- csvFldNames: codes lists of personal data fields that are to be updated, separated by semicolon (;) * (Optional)
- csvFldValues: values corresponding to personal data fields that are to be updated, separated by ; and in the same order as fields specified in parameter "csvFldNames" (Optional).

* To find the codes to use go to the "TABLE OF CODES" link in the Settings menu on the MailUp console.

Here is a practical example

I want to register to the distribution "General List" in the group "Group 1" without asking for confirmation and to the list "List2" sending a confirmation e-mail with my name:

name: John Smith

e-mail: John@smith.com

company: NWEB S.r.l.

Gender: M

To subscribe to the 2 lists above I have to send calls to 2 different posts (to simplify I will use the GET method accessing the page directly with the adequate query string).

From table of codes I see that the code for "general list" is 1, while the one for list "list 2" is 8, furthermore " Group 1" of general list has code 6 and the corresponding personal data fields are: Field 1, Field 2, Field 3, Field 5 (First Name, Last Name, Company, Gender).

As a result, the posts will be:

1. <http://customer.mailup.info/frontend/xmlSubscribe.aspx?list=1&group=6&email=john@smith.com&confirm=false&csvFldNames=field1;field2;field3;field5&csvFldValues=John;Smith;NWEB S.r.l.;M>
2. <http://customer.mailup.info/frontend/xmlSubscribe.aspx?list=8&email=john@smith.com&confirm=true&csvFldNames=field1;field2;field3;field5&csvFldValues=Alberto;Miscia;NWEB>

S.r.l.;M

In the second case a confirmation mail will be sent (this feature can be easily and freely set by using the "CONFIRMATION EMAIL REQUEST" link and choosing the apposite distribution list in the Settings menu on the MailUp console.

The user will be effectively subscribed only when he/she will click on the registration link on the email. If that doesn't happen this contact will be included in the "Pending" list.

The page called (xmlsubscribe.aspx) returns the following values :

- 0 : Operation completed
- 1 : Generic error
- 2 : Invalid email address
- 3 : Already subscribed user

These values can be used in the registration procedures to check newsletter subscription results.

Common Errors

After registering a subscriber, it appears in the PENDING LIST instead of SUBSCRIBERS LIST. This happens when some fields are not correctly spelled. Be sure to include CAMPO1 CAMPO2 CAMPO3 (uppercase is not important) and not NAME, SURNAME...

Otherwise check the "confirm" parameter: if set 0, confirmation email will not be sent and the user will be inserted directly in the subscribers list. Use 0 or 1, not "true" or "false".

3 BATCH FTP INTEGRATION

MailUp can both read or write a CSV file each night. File may contain the list of subscribers and/or unsubscribers.

For the batch procedures it is enough for the client to save, in his/her own FTP space a csv or xml file of the list of the subscribers and/or unsubscribers. Every night MailUp will collect the file and will load it in the system, managing also the duplication and of the non import of contacts already present in the Unsubscribers of MailUp (unless different setting specifics have been made). On request, MailUp can also save in the same FTP space other data (list of unsubscribers, wrong emails, statistic logs...).

FTP integrations are free of charge.

In each row the number of separators (;) has to be the same.

Sample1

```
nome@libero.it;pippo;rossi;gino spa  
Giuseppe@hotmail.com;giuseppe;bianchi;acme srl  
Mario@tin.it;mario;verdi;zucchis spa  
marcella@gmail.com;marcella;blui;ginger spa
```

Each field is 100 charaters long

Please not that at the end of each row there's not the ; symbol

Sample 2

```
nome@libero.it;pippo;rossi;gino spa;0  
Giuseppe@hotmail.com;giuseppe;bianchi;acme srl;1  
Mario@tin.it;mario;verdi;zucchis spa;1  
marcella@gmail.com;marcella;blui;ginger spa;1
```

In this case user is subscribed where "1" and unsubscribed where "0". Status may be forced if requested.

Sample3

```
nome@libero.it;pippo;rossi;gino spa;2;105;1  
nome@libero.it;pippo;rossi;gino spa;2;120;1  
Giuseppe@hotmail.com;giuseppe;bianchi;acme srl;2;106;1
```

Where

nome@libero.it;pippo;rossi;gino spa;2;105;1 → subscribed to list 2, group 105
nome@libero.it;pippo;rossi;gino spa;4;120;1 → subscribed to list 4, group 120
Giuseppe@hotmail.com;giuseppe;bianchi;acme srl;2;106;1 → subscribed to list 2, group 106
Mario@tin.it;mario;verdi;zucchis spa;6;;0 → unsubscribed from list 1

To know which are the list number and the group number , see the Tables of Codes.

Distribution Lists	
LIST CODE	LIST NAME
1	NEWS
2	TEXT

Group list		
LIST CODE	GROUP CODE	GROUP NAME
1	1	TEST
1	2	TEST GROUP
1	3	FIRST IMPORTATION
1	4	GROUP A

List of Personal Data Fields	
FIELD CODE	FIELD NAME
CAMPO1	FIRST NAME
CAMPO2	SECOND NAME
CAMPO3	COMPANY
CAMPO4	ADDRESS

Example of TABLE OF CODES. To open it click on **Settings** in the MailUp console and click "TABLE OF CODES".

4 POST OPERATIONS

After each subscribe or unsubscribe action, MailUp can call an external procedure or provide and automatic action.

Sample 1

After each subscription, MailUp calls

`http://customerdomain.tld/private/sub.php?email=[email]&userid=[id]&hash=[hashcode]`

Sample2

After each unsubscription, MailUp calls

`http://customerdomain.tld/private/unsub.php?email=[email]&userid=[id]&hash=[hashcode]`

5 WEB SERVICES

OVERVIEW

Scope of this draft is to explain the integration between MailUp and Third Party System through a Webservice, defined as "MailUp Import Webservice" (WSMailUpImport).

The integrations will allow:

- Third Party System recipients > MailUp List/Group
- MailUp lists details > Third Party System

In this way, in the Third Party System you can manage multiple list subscription, or users segmentation. MailUp will receive that data, and also manage the double-opt-in subscription, for one or multiple recipients.

This web service has an extra cost. Ask to your Sales rep. For ProductCart customers there's no cost.

GLOSSARY

- **MailUp customer**
A business subscribing to the MailUp service.
- **Console URL / NL_URL**
MailUp defines a unique URL for each MailUp customer. For this reason, Third Party System will need to know the exact url to which appropriate requests will be sent (e.g. <http://mailing.firstcompany.com/Services/WSMailup.asmx>)
- **WS_MailUpImport**
A Web Service designed to import recipients information from a third party application.
- **MailUp API Account**
A MailUp API Account is created automatically when a MailUp Account is generated (new customer) or manually via the MailUp console (existing customer). A MailUp API Account is made of a User Name and Password. See Manage menu / web service.

MAILUP API STATUS

Once a MailUp API Account has been created, the API must be turned ON to receive requests. The API is turned OFF by default, for added security.

The API can be turned on (activated) either manually via the MailUp Console or programmatically when the first, successfully authenticated request is performed by the third-party application.

The API can be turned OFF at any time via the MailUp Console.

WEB SERVICE AUTHENTICATION

WS_MailUpImport will authenticate requests via the MailUp API Account username, password, and IP address of the sender (third-party application).

Authentication will fail:

- If the MailUp API Status is OFF
- If the user name or password are not correct
- If the IP address of the third-party application has changed since the first, successfully authenticated request. The IP address can be changed manually via the MailUp Console.

WEB SERVICE METHODS

Methods exposed by the WS will allow to:

- Get a record of all existing distribution lists along with group names
- Create a new Group of contacts within a list
- Send appropriate parameters and a list of subscribers in order to create a new import process
- Get the status of the process
- Start the process

Each method will return an XML structure defined as follows:

XML string that contains a "MailUp Message". The MailUp Message always contains a return code and the requested information (if available).

The return code can contain

- a negative value = an error code
- a zero value = execution succeeded
- a positive value = an internal ID of the MailUp system

```
<mailupMessage>
  <mailupBody>
    <ReturnCode>0</ReturnCode>
    ...
  </mailupBody>
</mailupMessage>
```

API ACTIVATION or REGISTRATION

MailUp will provide an ASP.NET Web Form for the third party application to active the API (if not already active) and register itself with it (i.e. provide its IP address). The form will accept with the following parameters (both GET and POST methods):

- **usr**: MailUp API Account user name

- **pwd**: MailUp API Account password
- **nl_url**: Console URL
- **ws_name**: Web service name (only for compatibility purposes, it will be always the same)

Once the credentials have been authenticated successfully, and after confirming that the user used in the authentication process has the right permissions to use the API, the third party application is registered to use the API

Please Note: The User and Password must be the User and Password of the MailUp API Account, and not the User and Password used to log into the MailUp Console).

The form will return a MailUp Message with the result of the request:

Request the service activation:

http://mailing.mycompany/frontend/WSActivation.aspx?usr=<usr>&pwd=<pwd>&nl_url=<nl_url>&nl_name=<nl_name>&ws_name=<ws_name>

A sample of the response:

```
<mailupMessage>
  <mailupBody>
    <ReturnCode>0</ReturnCode>
    <WS_Activation>
      <WS_Name> WS_MailUpImport </WS_Name>
      <DateOfRequest>2008-01-16</DateOfRequest>
      <User>admin</User>
    </WS_Activation>
  </mailupBody>
</mailupMessage >
```

The ReturnCode can assume the following value:

- 0 request execution succeeded
- 2 ws name has not been specified ⁽¹⁾
- 4 user name has not been specified ⁽¹⁾
- 8 password has not been specified ⁽¹⁾
- 16 nl url has not been specified ⁽¹⁾
- 1000 unrecognized error
- 1001 the account is not valid
- 1002 the password is not valid
- 1003 suspended account
- 1004 inactive account
- 1005 expired account
- 1006 the web service is not enabled
- 1007 the web service is not active
- 1008 the web service is already active
- 1009 web service activation error
- 1010 IP registration error

⁽¹⁾ the ReturnCode can be a combination of these codes.

WS_MailUpImport

Authentication

The WebService will require username and password for authentication; the credential must be passed within the SOAP Header of the SOAP Message:

```
<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:Header>
    <Authentication xmlns="http://tempuri.org/">
      <User>username</User>
      <Password>password</Password>
    </Authentication>
  </soap:Header>
  <soap:Body>
    ...
  </soap:Body>
</soap:Envelope>
```

Note: at the moment we did not take into consideration any encryption algorithm just to focalize on the process

Error Codes

If the authentication fails, the method return one of the following ReturnCode:

- 1000 unrecognized error
- 1001 the account is not valid
- 1002 the password is not valid
- 1003 suspended account
- 1004 inactive account
- 1005 expired account
- 1006 the web service is not enabled
- 1007 the web service is not active
- 1011 IP is not registered
- 1012 IP is registered but has the "deny access" flag

All the methods that require the *idList* and *listGuid* as input paramters will verify the correspondence of the specified values and can return one of the following return codes:

- 100 unrecognized error
- 101 verification failed
- 102 list Guid format is not valid

WS_MailUpImport.GetNILists method

GetNILists(),

Gets a mailupMessage that contains "Lists" and "Groups" (e.g. 3 lists: "Technical Support Subscribers", "Company News Subscribers", "Product News Subscribers": each list can then contain separate groups).

```
<mailupMessage>
  <mailupBody>
    <ReturnCode>0</ReturnCode>
    <Lists>
      <List idList="1" listGUID="F9168C5E-CEB2-4FAA-B6BF-329BF39FA1E4">
        <Groups>
          <Group idGroup="1" groupName="First Group" />
          <Group idGroup="2" groupName="Second Group" />
        </Groups>
      </List>
    </Lists>
  </mailupBody>
</mailupMessage>
```

Error Codes

The specific error codes for this method are:

- 0 request execution succeeded
- 200 unrecognized error

WSMailUpImport.CreateGroup method

CreateGroup(int idList, int listGUID, string newGroupName)

Creates a new group for the specified list.

Parameters

- idList*: the identifier of the list
- listGuid*: the list GUID
- newGroupName*: the name of the group to be created

If a group with the given name already exists in the list, a new group will not be created, and an error code is returned within the mailup Message. Otherwise the idGroup given to the new group is returned.

```
<mailupMessage>
  <mailupBody>
    <ReturnCode>12</ReturnCode>
  </mailupBody>
</mailupMessage>
```

Error Codes

The specific error codes for this method are:

- 300 unrecognized error
- 301 the list has not been specified
- 302 the group name has not been specified
- 303 the group already exists

WSMailUpImport.NewImportProcess method

NewProcessImport(int idList, int listGUID, idGroup, string xmlDoc, string idGroups, int importType, int mobileInputType, bool asPending, bool ConfirmEmail, bool asOptOut, bool forceOptIn, bool replaceGroups, int idConfirmNL)

Creates a new import process for the contacts in the xmlFeed.

Parameters

idList: the identifier of the list

listGuid: the list GUID

idGroup: the identifier of the group

xmlDoc: a valid XML string that contains the contacts to be imported (**View Appendix A**

for details)

idGroups: group id

importType: specify the import type (set = 3)

mobileInputType: specify the mobile number input type (**View Appendix A for details**)

1=mobile number includes the international code

2=international code and mobile number are specified as two different fields)

asPending: include pending subscribers, Send Subscription Confirmation Request to pending accounts to whom you have already sent it, but who have not yet responded. (set = false)

ConfirmEmail: send confirmation email (set = false)

asOptOut: import subscribers as optout (set = false)

forceOptIn: force subscribers status as optin (set = false)

replaceGroups: replace groups (set = false)

idConfirmNL: confirmation newsletter id (set = 0, the confirmation will be automatically created)

If the import process is created, the ID of the new process (idProcess) is returned within the mailupBody, otherwise an error code will be returned.

```
<mailupMessage>
  <mailupBody>
    <ReturnCode>17</ReturnCode>
  </mailupBody>
</mailupMessage>
```

Error Codes

The specific error codes for this method are:

-400 unrecognized error

-401 xmlDoc is empty

-402 convert xml to csv failed

-403 create new import process failed

-410 cannot create confirmation email

WSMailUpImport.GetProcessDetails method

GetProcessDetails(int idList, int listGUID, int idProcess)

Gets the status and the details of the given import process.

Parameters

idList: the identifier of the list

listGuid: the list GUID

idProcess: the identifier of the import process

ReturnCode is 0 if the request succeeded, otherwise a negative error code.

The mailupBody contains the following information:

- start date
- end date
- the number of contacts to import
- the number of contacts imported (grouped by status: New, Existing, Optout)
- the process status (running, completed, waiting, error)

Note: "Mobile" leafs are meant for SMS delivery and are not part of Third Party System integration.

```
<mailupMessage>
  <mailupBody>
    <ReturnCode>0</ReturnCode>
    <ImportProcess idProcess="17">
      <StartDate>2008-01-16</StartDate>
      <EndDate></EndDate>
      <TotalContacts>10000</TotalContacts>
      <NewMobile>4500</ NewMobile >
      <NewEmail>1000</NewEmail>
      <OptOutMobile>15</OptOut>
      <OptOutEmail>15</OptOut>
      <ExistingMobile>100</ExistingMobile>
      <ExistingEmail>150</ExistingEmail>
      <StatusCode>2</StatusCode>
    </ ImportProcess >
  </mailupBody>
</mailupMessage>
```

Error Codes

The specific error codes for this method are:

-500 unrecognized error

-501 idProcess not found

WSMailUpImport.StartProcess method

StartProcess(int idList, int listGUID, int idProcess)

Starts the specified import process,

Parameters

- idList*: the identifier of the list
- listGuid*: the list GUID
- idProcess*: the identifier of the import process

If the process cannot be activated an error code will be returned within the mailupBody

```
<mailupMessage>
  <mailupBody>
    <ReturnCode>0</ReturnCode>
  </mailupBody>
</mailupMessage>
```

Error Codes

The specific error codes for this method are:

- 600 unrecognized error
- 601 an import process is already running for the list
- 602 an import process is already running for a different list
- 603 error checking process status
- 604 error starting the process job

WSMailUpImport.StartImportProcesses method

```
public string StartImportProcesses(string listsIDs, string listsGUIDs, string xmlDoc, string
groupsIDs,
    int importType, int mobileInputType, bool asPending, bool ConfirmEmail, bool asOptOut,
bool forceOptIn)
```

Creates a automatically start a new import process for the contacts in the xmlFeed. It is not necessary specify the newsletter ID to send as confirmation email; the process will automatically create a confirmation newsletter based on the default confirmation email.

Parameters

- listsIDs*: comma separated of lists IDs
- listsGUIDs*: comma separated of lists GUIDs
- xmlDoc*: a valid XML string that contains the contacts to be imported (**View Appendix A for details**)
- groupsIDs*: comma separated of groups IDs; for each list can be specified one or more groups (groups IDs separated by "," char)
- importType*: specify the import type (set = 3)
- mobileInputType*: specify the mobile number input type (**View Appendix A for details**)
1=mobile number includes the international code
2=international code and mobile number are specified as two different fields)
- asPending*: include pending subscribers, Send Subscription Confirmation Request to pending accounts to whom you have already sent it, but who have not yet responded. (set = false)
- ConfirmEmail*: send confirmation email (set = false)

asOptOut: import subscribers as optout (set = false)
forceOptIn: force subscribers status as optin (set = false)
replaceGroups: replace groups (set = false)

How to use the parameters listsIDs, listsGUIDs and groupsIDs

a) specify lists

listsIDs = "84;85;86" import contacts into lists 84, 85, 86

b) specify lists GUIDs

listsGUIDs = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx;xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx;xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx"

c) specify groups for each list

the contacts will be imported into list 84 and groups 12, 13

the contacts will be imported into list 85 and group 23

the contacts will be imported into list 86 and no groups

groupsIDs = "12,13;23;0"

Return message:

The return message contains a generic error code and a specific error code for each import process that has been requested.

```
<mailupMessage>
  <mailupBody>
    <ReturnCode>0</ReturnCode>
    <processes>
      <process>
        <processID>0</processID>
        <listID>84</listID>
        <ReturnCode>-410</ReturnCode>
      </process>
      <process>
        <processID>18</processID>
        <listID>85</listID>
        <ReturnCode>0</ReturnCode>
      </process>
      <process>
        <processID>19</processID>
        <listID>86</listID>
        <ReturnCode>0</ReturnCode>
      </process>
    </processes>
  </mailupBody>
</mailupMessage>
```

Error Codes

The specific error codes for this method are:

- 400 unrecognized error
- 401 xmlDoc is empty
- 402 convert xml to csv failed
- 403 create new import process failed

- 410 can not create confirmation email
- 450 listsIDs and listsGUIDs must contain the same number of elements
- 600 unrecognized error
- 601 an import process is already running for the list
- 602 an import process is already running for a different list
- 603 error checking process status
- 604 error starting the process job

Appendix A – XML FEED

The XML structure for the subscribers import are the following

```
<subscribers>
  <subscriber email="user@nweb.it" Prefix="+39" Number="3470055123" Name="">
    <campo1>first name</campo1>
    <campo2>last name</campo2>
    <campo3>Nweb SRL</campo3>
  </subscriber>
</subscribers>
```

The mobile number can be specified in two mode:

- Using a single field

```
<subscriber email="user@nweb.it" Prefix="" Number="+393470055123" Name="">
```

- Or using two fields

```
<subscriber email="user@nweb.it" Prefix="+39" Number="3470055123" Name="">
```

Check queue status

<http://host.cliente.it/frontend/queueStatus.aspx?idList=3&listGuid=bd1235d5-9213-4116-8231-12321g00123d>

```
<?xml version="1.0" encoding="utf-8" ?>
<statoCode>
  <coda>
    <idLista>1</idLista>
    <Lista>Newsletter</Lista>
    <idNewsletter>9</idNewsletter>
    <Newsletter>new emailing today</Newsletter>
    <tempoStimato>00:00:01</tempoStimato>
    <Destinatari>1</Destinatari>
    <percInvio>0</percInvio>
  </coda>
  <totaleCode>0</totaleCode>
  <totaleTempoStimato>00:00:00</totaleTempoStimato>
```

- IdLista: list code

- Lista: list name
- idNewsletter: message id
- tempo Stimato: estimated time to complete sending
- Destinatari: recipients
- percInvio: % of completion
- totalecode: number of queues
- totaletempostimato: sum of the estimated time to completion for all the queues

ListGUID is available in Settings menu, edit lists.

Read MailUp Statistics in an external IFRAME

We can provide through a FTP connection or web service connection, an XML file with the links to the main MailUp statistics reports. Links are public so they can be called externally within another application like a CMS, CRM, ERP or Ecommerce System.

```
<?xml version="1.0" encoding="utf-8"?>
<statisticheMessaggi>
  <idLista>1</idLista>
  <listGuid>df2342241-9e234-2363-fsdfs-dfswfe0dfbsd2sd</listGuid>
  <newsletters>
    <newsletter idNL="17">
      <subject>nuova prova campagna!</subject>
      <note>1232343</note>
      <dataCreazione>14/12/2007 16.26.20</dataCreazione>
      <riassuntivo_grafico>
http://host.cliente.it/frontend/show\_message\_reports.aspx?idList=1&listGuid=df2342241-9e234-2363-fsdfs-dfswfe0dfbsd2sd&idNL=17
      </riassuntivo_grafico>
      <aperture_grafico>
http://host.cliente.it/frontend/time\_views\_report.aspx?idList=1&listGuid=df2342241-9e234-2363-fsdfs-dfswfe0dfbsd2sd&idNL=17
      </aperture_grafico>
      <aperture_dettaglio>
http://host.cliente.it/frontend/user\_views\_report.aspx?idList=1&listGuid=df2342241-9e234-2363-fsdfs-dfswfe0dfbsd2sd&idNL=17
      </aperture_dettaglio>
      <aperture_filtri>
http://host.cliente.it/frontend/user\_views\_report\_filter.aspx?idList=1&listGuid=df2342241-9e234-2363-fsdfs-dfswfe0dfbsd2sd&idNL=17
      </aperture_filtri>
      <click_grafico>
http://host.cliente.it/frontend/click\_views\_report.aspx?idList=1&listGuid=df2342241-9e234-2363-fsdfs-dfswfe0dfbsd2sd&idNL=17
      </click_grafico>
      <click_dettaglio>
http://host.cliente.it/frontend/link\_tracking.aspx?idList=1&listGuid=df2342241-9e234-2363-fsdfs-dfswfe0dfbsd2sd&idNL=17
      </click_dettaglio>
      <click_filtri>
http://host.cliente.it/frontend/link\_tracking\_filter.aspx?idList=1&listGuid=df2342241-9e234-2363-fsdfs-dfswfe0dfbsd2sd&idNL=17
      </click_filtri>
      <rapporto_invii>
http://host.cliente.it/frontend/send\_report2.aspx?idList=1&listGuid=df2342241-9e234-2363-fsdfs-dfswfe0dfbsd2sd&idNL=17
      </rapporto_invii>
    </newsletter>
  </newsletters>
</statisticheMessaggi>
```

WS_MailupSend (SOAP)

Send with MailUp without using the MailUp console.

<http://services.mailupnet.it/MailupSend.asmx>

<http://services.mailupnet.it/MailupSend.asmx?WSDL>

Web service: WS_MailupSend

MailUp mette ora a disposizione un WebService per integrazione di MailUp in applicativi di terze parti, mettendo a disposizione una serie di metodi che permettono l'invio di Newsletter senza dover accedere alla console MailUp per effettuare le operazioni di creazione della newsletter, caricamento dei destinatari, creazione della coda, etc.

WebService MailupSend

Il servizio è disponibile all'indirizzo:

<http://services.mailupnet.it/MailupSend.asmx>

Il rispettivo WSDL è disponibile all'indirizzo:

<http://services.mailupnet.it/MailupSend.asmx?WSDL>

Il WebService è accessibile attraverso l'utilizzo del protocollo SOAP.

Login

Provides the accessKey needed for all the others methods

`string Login(string user, string pwd, string url)`

Output

```
LoginResult:  
  
<LoginResult>  
  <errorCode>0</errorCode>  
  <errorDescription></errorDescription>  
  <accessKey><![CDATA[accessKey]]></accessKey>  
</LoginResult>
```

Logout

Deletes the accessKey, otherwise it expires in 60 minutes.

`string Logout(string accessKey)`

Output:

```
LogoutResult
```

```
<LogoutResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
</LogoutResult>
```

GetLists

string GetLists(**string** accessKey)

accessKey: chiave di accesso ottenuta chiamando il Method Login

Output

```
GetListResult:
<GetListsResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <lists>
    <list>
      <listID>1</listID>
      <listName>List nr.1</listName>
    </list>
    <list>
      <listID>2</listID>
      <listName>List nr.2</listName>
    </list>
  </lists>
</GetListsResult>
```

GetGroups

string GetGroups(**string** accessKey, **int** listID)

Output

```
GetGroupsResult:
```

```
<GetGroupsResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <list>
    <listID>1</listID>
    <listName>Lista nr.1</listName>
    <groups>
      <group>
        <groupID>1</groupID>
        <groupName>Gruppo nr.1</groupName>
      </group>
      <group>
        <groupID>2</groupID>
        <groupName>Gruppo nr.2</groupName>
      </group>
    </groups>
  </list>
</GetGroupsResult>
```

Method GetTemplates

`string` GetTemplates(`string` accessKey, `int` listID)

Output:

```
GetTemplatesResult:
<GetTemplatesResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <list>
    <listID>1</listID>
    <listName>_DEMO_</listName>
    <templates>
      <template>
        <templateID>2976</templateID>
        <templateSubject>Blu banner 6 sezioni
vert.</templateSubject>
        <templateLink>Link to preview image</templateLink>
      </template>
    </templates>
  </list>
```

Method GetNewsletters

string GetNewsletters(**string** accessKey, **int** listID)

GetNewslettersResult:

```
<GetNewslettersResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <list>
    <listID>1</listID>
    <listName>Lista nr.1</listName>
    <newsletters>
      <newsletter>
        <newsletterID>1</newsletterID>
        <subject>Newsletter nr.1 Aprile 2009 | Le novità del
mese</subject>
        <creationdate>20090402</creationdate>
      </newsletter>
      <newsletter>
        <newsletterID>1</newsletterID>
        <subject>Newsletter nr.2 Maggio 2009 | Le novità del
mese</subject>
        <creationdate>20090505</creationdate>
      </newsletter>
    </newsletters>
  </list>
</GetNewslettersResult>
```

Method CreateNewsletter

string CreateNewsletter(**string** accessKey, **int** listID, **string** subject, **string** type, **string** content, **Option**[]) options)

type: (HTML, URL, FILE, TEXT)

content:

- type = HTML, content is the HTML code message
- type = TEXT, content is a PLAIN-TEXT message
- type = URL, content is the url where MailUp will find the html code of the message
- type = FILE, content is an byte array which will be used to build the html code

options: an array of key/value options

- note: just a comment field
- dyn_fld: use or not use the dynamic tags replacement (dyn_fld=true, dyn_fld=false)
- emb_img: set images as embedded or not (emb_img=true, emb_img=false)
- link_track: enable link tracking (all the protocols if not specified) (link_track=true, link_track=false).
- track_http: enable link tracking regarding the http protocol (track_http=true, track_http=false)
- track_https: enable link tracking regarding the HTTPS (track_https=true, track_https=false)
- track_mailto: enable link tracking regarding the MAILTO (track_mailto=true, track_mailto=false)
- track_ftp: enable link tracking regarding the FTP (track_ftp=true, track_ftp=false)
- track_news: enable link tracking regarding the NEWS (track_news=true, track_news=false)
- link_params: add this string to all tracked links. The string will not be inserted in the html message. It will be inserted by the track.aspx page before the final landing
- body_code: will be inserted in the <body> tag
- header: will be inserted in the <header> tag
- conf_msg: if the message will be used as an optin-request
- attach<nn>_data = base64 encoding del file
- attach<nn>_name = file name to be enclosed
- template = needed if the message needs a template. If template is specified, "content" and "body_code" will be ignored.

Output

```
CreateNewslettersResult:  
  
<CreateNewsletterResult>  
  <errorCode>0</errorCode>  
  <errorDescription></errorDescription>  
  <listID>1</listID>  
  <newsletterID>3</newsletterID>  
</CreateNewsletterResult>
```

Method SendNewsletter

string SendNewsletter(**string** accessKey, **int** listID, **int** newsletterID, **Option[]** options)

options:

- from_email: (mandatory)
- from_name: (mandatory)
- clear_stats: if set tot rue, all the previous message statistics will be erased
- send_to: (mandatory)
 - send_to=ALL → to all list subscribers
 - send_to=RECIPIENTS → to specified recipients
 - send_to=GROUPS → to some groups

- recipients:
 - if send_to = ALL → ignored recipient value
 - send_to=RECIPIENTS → recipients contains the recipients list with ; separator
 - send_to=GROUPS → recipients contains the list of the groups with ; separator
- exclude: a string with the list of groups to be excluded (separator is always ;)
- filters: a list with the filter IDs to be applied
- filters_as:
 - filters_as=AND
 - filters_as=OR
- deliverydt: scheduling date as yyyyMMddHHmmss. If not specified, the message is sent immediately.
- timezone: (ie 1, 2, 3)
- campo1; ...; campo<nn> = it may be used to update the anagraphic fields while sending to a recipients list.

Ie. Sending to

```
recipients=m.rossi@example.com;b.clinton@example.com;g.bush@example.com
campo1=Mario;Bill;George;
campo2=Rossi;Clinton;Bush
```

SendNewsletterResult:

```
<SendNewsletterResult>
  <errorCode>0</errorCode>
  <errorDescription>0</errorDescription>
  <listID>1</listID>
  <newsletterID>1</newsletterID>
  <deliveryStatus>Sending in progress</deliveryStatus>
</SendNewsletterResult>
```

Method SendNewsletterFast

```
string SendNewsletterFast(string accessKey, int listID, string subject, string type,
string content, Option[] options)
```

SendNewsletterFastResult:

```
<SendNewsletterFastResult>
  <errorCode>0</errorCode>
  <errorDescription>0</errorDescription>
  <listID>1</listID>
  <newsletterID>1</newsletterID>
```

```
<deliveryID>125</deliveryID>
<deliveryStatus>Sending in progress</deliveryStatus>
</SendNewsletterResult>
```

Method GetNewsletterDeliveryStatus

string GetNewsletterDeliveryStatus(**string** accessKey, **int** deliveryID)

GetNewsletterDeliveryStatusResult:

```
<GetNewsletterDeliveryStatusResult>
  <errorCode>0</errorCode>
  <deliveryStatus>
    <deliveryID>125</deliveryID>
    <listID>1</listID>
    <newsletterID>47</newsletterID>
    <status>SENDING</status>
    <details> </details>
    <toSend>100</toSend>
    <toBeSent>80</toBeSent>
    <progress>20</progress>
    <estimatedTime>0d 1h 33m 12s</estimatedTime>
  </deliveryStatus>
</GetNewsletterDeliveryStatusResult>
```

Method StartDelivery

string StartDelivery(**string** accessKey)

accessKey: chiave di accesso ottenuta chiamando il Method Login

GetGroupsResult:

```
<StartDeliveryResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <deliveryStatus>Sending in progress</deliveryStatus>
  <deliveryIDs>
    <delivery>
      <deliveryID>150</deliveryID>
```

```
<listID>1</listID>
<newsletterID>232</newsletterID>
</delivery>
<delivery>
  <deliveryID>151</deliveryID>
  <listID>2</listID>
  <newsletterID>275</newsletterID>
</delivery>
</deliveryIDs>
</StartDeliveryResult>
```

Method StopDelivery

string StartDelivery(**string** accessKey)

accessKey: chiave di accesso ottenuta chiamando il Method Login

GetGroupsResult:

```
<StopDeliveryResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
</StopDeliveryResult>
```

Method SendSingleNewsletter

string SendSingleNewsletter(**string** accessKey, **int** listID, **string** subject, **string** type, **string** content, **string** recipientEmail, **Option** [] options)

type: (HTML, URL, FILE, TEXT, ID)

content:

- type = HTML, content is the HTML code message
- type = TEXT, content is a PLAIN-TEXT message
- type = URL, content is the url where MailUp will find the html code of the message
- type = FILE, content is an byte array which will be used to build the html code
- type = ID, content is the message id to be sent

options: an array of key/value options

- note: just a comment field
- dyn_fld: use or not use the dynamic tags replacement (dyn_fld=true, dyn_fld=false)
- emb_img: set images as embedded or not (emb_img=true, emd_img=false)

- link_track: enable link tracking (all the protocols if not specified) (link_trcak=true, link_track=false).
 - track_http: enable link tracking regarding the http protocol (track_http=true, track_http=false)
 - track_https: enable link tracking regarding the HTTPS (track_https=true, track_https=false)
 - track_mailto: enable link tracking regarding the MAILTO (track_mailto=true, track_mailto=false)
 - track_ftp: enable link tracking regarding the FTP (track_ftp=true, track_ftp=false)
 - track_news: enable link tracking regarding the NEWS (track_news=true, track_news=false)
 - link_params: add this string to all tracked links. The string will not be inserted in the html message. It will be inserted by the track.aspx page before the final landing
 - body_code: will be inserted in the <body> tag
 - header: will be inserted in the <header> tag
 - conf_msg: if the message will be used as an optin-request
 - attach<nn>_data = base64 encoding del file
 - attach<nn>_name = file name to be enclosed
 - template = needed if the message needs a template. If template is specified, "content" and "body_code" will be ignored.
- from_email: sender email. If not specified will be used the default list value
 - from_name: sender name. If not specified will be used the default list value
 - track_delivery: enable sending logging (true, false)

SendSingleNewsletterResult:

```
<SendSingleNewsletterResult>
  <errorCode>0</errorCode>
  <errorDescription>Message sent to
support@mailup.it</errorDescription>
</SendSingleNewsletterResult>
```

Method GetNewsletterQueues

string GetNewsletterQueues(string accessKey, int listID)

GetNewsletterQueuesResult:

```
<GetNewsletterQueuesResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <ImmediateSendingQueues>
    <Queue>
      <listID>1</listID>
      <newsletterID>1144</newsletterID>
      <subject>Test </subject>
```

```
<deliveryID>1610</deliveryID>
<jobScheduleID>1235</jobScheduleID>
<toBeSent>6</toBeSent>
<progress>0</progress>
<estimatedTime>0d 00h 28m 27s</estimatedTime>
</Queue>
</ImmediateSendingQueues>
<DeferredSendingQueues>
  <Queue>
    <listID>157</listID>
    <newsletterID>1139</newsletterID>
    <subject>Newsletter // 2009-09-10 with attachments</subject>
    <deliveryID>1612</deliveryID>
    <jobScheduleID>1236</jobScheduleID>
    <toBeSent>4</toBeSent>
    <progress>0</progress>
    <estimatedTime>0d 00h 01m 15s</estimatedTime>
    <startDate></startDate>
    <startTime></startTime>
  </Queue>
</DeferredSendingQueues>
<WaitingToSendQueues>
  <Queue>
    <listID>168</listID>
    <newsletterID>1072</newsletterID>
    <subject>E-Mail link tracking</subject>
    <deliveryID>1611</deliveryID>
    <jobScheduleID>0</jobScheduleID>
    <toBeSent>4</toBeSent>
    <progress>0</progress>
    <estimatedTime>0d 00h 00m 21s</estimatedTime>
  </Queue>
</WaitingToSendQueues>
</GetNewsletterQueuesResult>
```

Method RemoveNewsletterQueue

string RemoveNewsletterQueue(**string** accessKey, **int** listID, **int** newsletterID, **int** deliveryID, **int** scheduleID, **bool** deleteQueue)

deleteQueue: if false, the queue will be removed and placed in the "NEWSLETTER WAITING FOR SENDING" queues. Otherwise is completely deleted.

RemoveNewsletterQueueResult:

```
<RemoveNewsletterQueueResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
</RemoveNewsletterQueueResult>
```

Method ScheduleNewsletterQueue

string ScheduleNewsletterQueue (**string** accessKey, **int** listID, **int** newsletterID, **int** scheduleID, **string** deliverydt, **int** timezone)

deliverydt: format is yyyyMMddHHmmss.

timezone: ie 1,2,3...

ScheduleNewsletterQueueResult:

```
<ScheduleNewsletterQueueResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <schedule>Delivery scheduled at 30/09/2009 9:30:00 GMT 1</schedule>
</ScheduleNewsletterQueueResult>
```

Method GetSMS

string GetSMS(**string** accessKey, **int** listID)

GetSMSResult:

```
<GetSMSResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <list>
    <listID>1</listID>
    <listName>List nr.1</listName>
    <messages>
      <SMS>
        <smsID>1</ smsID >
        <subject>SMS nr.1 April 2009 </subject>
        <creationdate>20090402</creationdate>
```

```
</SMS>
<newsletter>
  < smsID >1</ smsID >
  <subject>SMS nr.2 May 2009 </subject>
  <creationdate>20090505</creationdate>
</SMS>
</messages>
</list>
</GetSMSResult>
```

Method CreateSMS

string CreateSMS(**string** accessKey, **int** listID, **string** subject, **string** text, **Option** [] options)

options: an array with

- note: just a comment field
- dyn_fld: activate or not the dynamic tags replacement (dyn_fld=true, dyn_fld=false)
- from: sender name (10 characters) or number (12 characters)

CreateSMSResult:

```
<CreateSMSResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <listID>1</listID>
  <smsID>3</smsID>
</CreateSMSResult>
```

Method SendSMS

string SendSMS(**string** accessKey, **int** listID, **int** smsID, **Option** [] options)

options:

- from_email: (mandatory)
- from_name: (mandatory)
- clear_stats: if set to true, all the previous message statistics will be erased
- send_to: (mandatory)
 - send_to=ALL → to all list subscribers
 - send_to=RECIPIENTS → to specified recipients

- send_to=GROUPS → to some groups
- recipients:
 - if send_to = ALL → ignored recipient value
 - send_to=RECIPIENTS → recipients contains the recipients list with ; separator
 - send_to=GROUPS → recipients contains the list of the groups with ; separator
- exclude: a string with the list of groups to be excluded (separator is always ;)
- filters: a list with the filter IDs to be applied
- filters_as:
filters_as=AND
filters_as=OR
- deliverydt: scheduling date as yyyyMMddHHmmss. If not specified, the message is sent immediately.
- timezone: (ie 1, 2, 3)

SendSMSResult:

```
<SendSMSResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <listID>1</listID>
  <smsID>3</smsID>
  <deliveryID>31</deliveryID>
  <deliveryStatus>Sending in progress</deliveryStatus>
  <cost>1.5</cost>
</ SendSMSResult >
```

Method SendSMSFast

string SendSMSFast(**string** accessKey, **int** listID, **string** subject, **string** text, **Option** [] options)

SendSMSFastResult:

```
<SendSMSFastResult>
  <errorCode>0</errorCode>
  <errorDescription>0</errorDescription>
  <listID>1</listID>
  <smsID>1</newsletterID>
  <deliveryID>125</deliveryID>
  <deliveryStatus>Sending in progress</deliveryStatus>
< SendSMSFastResult >
```

Method SendSingleSMS

`string` SendSingleSMS(`string` accessKey, `int` listID, `string` subject, `string` type, `string` content, `string` numero, `Option`[] options)

type: tipo di origine del testo del messaggio (HTML, URL, FILE, TEXT, ID)

content: origine del testo del messaggio, dipende dal type:

- type = TEXT, content è testo del messaggio
- type = ID, content è l'identificativo della newsletter da inviare

options:

- note: just a comment field
- dyn_fld: dyn_fld=true, dyn_fld=false
- from:
- track_delivery: true, false

SendSingleSMSResult:

```
<SendSingleSMSResult>
  <errorCode>0</errorCode>
  <errorDescription>Message sent +393401111111</errorDescription>
</SendSingleSMSResult>
```

Method GetSMSDeliveryStatus

`string` GetSMSDeliveryStatus(`string` accessKey, `int` deliveryID)

GetSMSDeliveryStatusResult:

```
<GetSMSDeliveryStatusResult>
  <erroCode>0</erroCode>
  <deliveryStatus>
    <deliveryID>125</deliveryID>
    <listID>1</listID>
    <smsID>47</smsID>
    <status>SENDING</status>
    <details> </details>
    <toSend>100</toSend>
    <toBeSent>80</toBeSent>
    <progress>20</progress>
    <residualCost>135</residualCost>
```

```
</deliveryStatus>  
</GetSMSDeliveryStatusResult>
```

Method StartDeliverySMS

`string` StartDeliverySMS(`string` accessKey)

accessKey: chiave di accesso ottenuta chiamando il Method Login

```
GetGroupsResult:  
  
<StartDeliverySMSResult>  
  <errorCode>0</errorCode>  
  <errorDescription></errorDescription>  
  <deliveryStatus>Sending in progress</deliveryStatus>  
  <deliveryIDs>  
    <delivery>  
      <deliveryID>150</deliveryID>  
      <listID>1</listID>  
      <smsID>232</smsID>  
    </delivery>  
    <delivery>  
      <deliveryID>151</deliveryID>  
      <listID>2</listID>  
      <smsID>275</smsID>  
    </delivery>  
  </deliveryIDs>  
</StartDeliverySMSResult>
```

Method StopDeliverySMS

`string` StopDeliverySMS(`string` accessKey)

accessKey: chiave di accesso ottenuta chiamando il Method Login

```
GetGroupsResult:
```

```
<StopDeliverySMSResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
</StopDeliverySMSResult>
```

Method GetSMSQueues

string GetSMSQueues(**string** accessKey, **int** listID)

GetSMSQueuesResult:

```
<GetNewsletterQueuesResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <ImmediateSendingQueues>
    <Queue>
      <listID>1</listID>
      <smsID>1144</smsID>
      <subject>SMS // test nr.1 2009-10-28</subject>
      <deliveryID>1610</deliveryID>
      <jobScheduleID>1235</jobScheduleID>
      <toBeSent>6</toBeSent>
      <cost>9</cost>
      <progress>0</progress>
    </Queue>
  </ImmediateSendingQueues>
  <DeferredSendingQueues>
    <Queue>
      <listID>157</listID>
      <smsID>1139</smsID>
      <subject> SMS // test nr.2 2009-10-28</subject>
      <deliveryID>1612</deliveryID>
      <jobScheduleID>1236</jobScheduleID>
      <toBeSent>4</toBeSent>
      <cost>6</cost>
      <startDate></startDate>
      <startTime></startTime>
    </Queue>
  </DeferredSendingQueues>
  <WaitingToSendQueues>
    <Queue>
      <listID>168</listID>
```

```
<smsID>1072</smsID>
<subject> SMS // test nr.3 2009-10-28</subject>
<deliveryID>1611</deliveryID>
<jobScheduleID>0</jobScheduleID>
<toBeSent>4</toBeSent>
<cost>6</cost>
</Queue>
</WaitingToSendQueues>
</GetSMSQueuesResult>
```

Method RemoveSMSQueue

string RemoveSMSQueue(**string** accessKey, **int** deliveryID, **int** scheduleID, **bool** deleteQueue)

RemoveSMSQueueResult:

```
<RemoveSMSQueueResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
</RemoveSMSQueueResult>
```

Method ScheduleNewsletterQueue

string ScheduleSMSQueue(**string** accessKey, **int** listID, **int** smsID, **int** scheduleID, **string** deliverydt, **int** timezone)

ScheduleSMSQueueResult:

```
<ScheduleSMSQueueResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <schedule>Delivery scheduled at 30/09/2009 9:30:00 GMT 1</schedule>
</ScheduleSMSQueueResult>
```

Method GetCredits

`string` GetCredits(`string` accessKey)

accessKey: chiave di accesso ottenuta chiamando il Method Login

```
ScheduleSMSQueueResult:  
  
<GetCreditsResult>  
  <errorCode>0</errorCode>  
  <errorDescription></errorDescription>  
  <credits>12500</credits>  
</GetCreditsResult>
```

WebService MailupManage (SOAP)

<http://services.mailupnet.it/MailupManage.asmx>

<http://services.mailupnet.it/MailupManage.asmx?WSDL>

Method Login

`string` Login(`string` user, `string` pwd, `string` url)

```
LoginResult:  
  
<LoginResult>  
  <errorCode>0</errorCode>  
  <errorDescription></errorDescription>  
  <accessKey><![CDATA[accessKey]]></accessKey>  
</LoginResult>
```

Logout

`String` Logout(`string` accessKey)

```
LogoutResult
```

```
<LogoutResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
</LogoutResult>
```

Method GetLists

`string` GetLists(`string` accessKey)

`accessKey`: chiave di accesso ottenuta chiamando il Method Login

```
GetListResult

<GetListsResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <lists>
    <list>
      <listID>1</listID>
      <listName>Demo</listName>
    </list>
  </lists>
</GetListResult>
```

Method GetGroups

`string` GetGroups(`string` accessKey, `int` listID)

```
GetGroupResult

<GetGroupsResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <list>
    <listID>1</listID>
    <listName>Demo</listName>
  </list>
  <groups>
    <group>
      <groupID>1</listID>
```

```
<groupName>Demo</listName>
</group>
</groups>
</GetGroupsResult>
```

Method CreateList

string CreateList(**string** accessKey, **string** name, **string** defaultSettings, **bool** copyTemplate, **Option** [] options)

name: list name

defaultSettings: if "default" use the standard settings, otherwise you may set a different list ID to copy its settings.

copyTemplate: if true, all the templates will be copied

options: array

- owneremail: sender email address
- bouncedemail: return path address
- description:
- format: ("html", "text")
- charset: ("UTF-8", "ISO-8859-1", "windows-1250", "UTF-7", "windows-1251", "windows-1252", "windows-1253", "windows-1254", "windows-1255", "windows-1256", "windows-1257", "windows-1258", "ISO-8859-2", "ISO-8859-3", "ISO-8859-4", "ISO-8859-5", "ISO-8859-6", "ISO-8859-7", "ISO-8859-8", "ISO-8859-9", "ISO-8859-15", "cp866", "koi8-r", "koi8-u", "shift_jis", "ks_c_5601-1987", "EUC-KR", "BIG5", "HZ-GB-2312", "EUC-JP")
- public: if enabled, message archive will be public
- tracking: if enabled, opens will be tracked
- nl_sendername: sender name
- sms_sendername: sms sender
- optout_type: (1: Double with "CUSTOM OPT-OUT PAGE" displayed, 2: Single with MESSAGES PAGE displayed, 3: Double with standard opt-out confirmation page with the summary of the lists)
- sendmailoptout: if enabled, the EMAIL TO UNSUBSCRIBED is sent.
- notifyemail: if enabled, a notification will be sent to this email address
- frontendform: if enabled, front end public subscription forms will be available
- disclaimer: a text which is added at the end of some notification pages and forms.
- headerxabuse: X-Abuse header text
- kbmax: warn if the message size is over this value
- replyto: email address
- displayas: choose which fields will be used as recipients name (campo1,campo2 etc.)
- headerlistunsubscribe: set which fields are displayed in the MailUp console when clicking on Subscribes/List
- multipart_text: flag to enable the automatic creation of a TXT version of the message
- conversionlab_trackcode: conversionlab tracking code
- default_prefix: default International mobile phone prefix
- multi_optout_list: which list are to be displayed if optout_type = 3.

- subscribedemail: send the EMAL TO SUBSCRIBED after each subscription
- sendconfirmsms: send the SMS TO SUBSCRIBED message after each subscription
- senderfaxname:
- senderfax: number
- senderfirstname: for snail mail
- senderlastname: for snail mail
- sendercompanyname: for snail mail
- senderaddress: for snail mail
- senderpostalcode: zip code for snail mail
- sendercity: for snail mail
- senderprovince: state, for snail mail
- senderstate: country, for snail mail

CreateListResult

```
<InsertListResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <list>
    <listID>1</listID>
    <listName>Demo</listName>
    <hash></hash>
    <name></name>
    <description></description>
    <format></format>
    <charset></charset>
    <public></public>
    <tracking></tracking>
    <unsubscribe></unsubscribe>
    <owneremail></owneremail>
    <bounceemail></bounceemail>
    <header></header>
    <footer></footer>
    <footercustom></footercustom>
    <footermailup></footermailup>
    <nl_sendername></nl_sendername>
    <sms_sendername></sms_sendername>
    <optout_type></optout_type>
    <sendemailoptout></sendemailoptout>
    <notifyemail></notifyemail>
    <frontendform></frontendform>
    <disclaimer></disclaimer>
    <headerxabuse></headerxabuse>
    <kbmax></kbmax>
```

```
<headerreplyto></headerreplyto>
<displayas></displayas>
<headerlistunsubscribe></headerlistunsubscribe>
<smtp_confirm></smtp_confirm>
<multipart_text></multipart_text>
<conversionlab_trackcode></conversionlab_trackcode>
<default_prefix></default_prefix>
<multi_optout_list></multi_optout_list>
<subscribedemail></subscribedemail>
<sendconfirmsms></sendconfirmsms>
<senderfaxname></senderfaxname>
<senderfax></senderfax>
<senderfirstname></senderfirstname>
<senderlastname></senderlastname>
<sendercompanyname></sendercompanyname>
<senderaddress></senderaddress>
<senderpostalcode></senderpostalcode>
<sendercity></sendercity>
<senderprovince></senderprovince>
<senderstate></senderstate>
</list>
</InsertListResult>
```

Motodo UpdateList

`string` UpdateList(`string` accessKey, `int` listID, `bool` copyTemplate, `Option[]` options)

UpdateListResult

```
<UpdateListResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <list>
    <listID>1</listID>
    <listName>Demo</listName>
    <hash></hash>
    <name></name>
    <description></description>
    <format></format>
    <charset></charset>
    <public></public>
    <tracking></tracking>
```

```
<unsubscribe></unsubscribe>
<owneremail></owneremail>
<bounceemail></bounceemail>
<header></header>
<footer></footer>
<footercustom></footercustom>
<footermailup></footermailup>
<nl_sendername></nl_sendername>
<sms_sendername></sms_sendername>
<optout_type></optout_type>
<sendemailoptout></sendemailoptout>
<notifyemail></notifyemail>
<frontendform></frontendform>
<disclaimer></disclaimer>
<headerxabuse></headerxabuse>
<kbmax></kbmax>
<headerreplyto></headerreplyto>
<displayas></displayas>
<headerlistunsubscribe></headerlistunsubscribe>
<smtp_confirm></smtp_confirm>
<multipart_text></multipart_text>
<conversionlab_trackcode></conversionlab_trackcode>
<default_prefix></default_prefix>
<multi_optout_list></multi_optout_list>
<subscribedemail></subscribedemail>
<sendconfirmsms></sendconfirmsms>
<senderfaxname></senderfaxname>
<senderfax></senderfax>
<senderfirstname></senderfirstname>
<senderlastname></senderlastname>
<sendercompanyname></sendercompanyname>
<senderaddress></senderaddress>
<senderpostalcode></senderpostalcode>
<sendercity></sendercity>
<senderprovince></senderprovince>
<senderstate></senderstate>
</list>
</UpdateListResult>
```

Method DeleteList

string DeleteList(**string** accessKey, int listID)

```
DeleteListResult
```

```
<DeleteListResult>  
  <errorCode>0</errorCode>  
  <errorDescription></errorDescription>  
</DeleteListResult>
```

Method CreateGroup

`string` CreateGroup(`string` accessKey, `string` groupName, `string` groupNotes)

Output:

```
CreateGroupResult
```

```
<CreateGroupResult>  
  <errorCode>0</errorCode>  
  <errorDescription></errorDescription>  
  <list>  
    <listID>1</listID>  
    <listName>Demo</listName>  
    <groups>  
      <group>  
        <groupID>1</groupID>  
        <groupName>Demo</groupName>  
        <groupNotes>Note Gruppo</groupNotes>  
      </group>  
    </groups>  
  </list>  
</CreateGroupResult>
```

Method UpdateGroup

`string` UpdateGroup(`string` accessKey, `int` groupID, `Option[]` options)

```
CreateGroupsResult
```

```
<UpdateGroupResult>  
  <errorCode>0</errorCode>  
  <errorDescription></errorDescription>
```

```
<list>
  <listID>1</listID>
  <listName>Demo</listName>
  <groups>
    <group>
      <groupID>1</groupID>
      <groupName>Demo</groupName>
      <groupdNotes>Note Gruppo</groupNotes>
    </group>
  </groups>
</list>
</UpdateGroupResult>
```

Method DeleteGroup

`string DeleteList(string accessKey, int listID, int groupID, bool deleteUsers)`

deleteUsers: if enabled, it deletes not only the group but all the recipients belonging to this group

```
DeleteGroupResult
<DeleteGroupResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
</DeleteGroupResult>
```

WebService MailupReport (SOAP)

<http://services.mailupnet.it/MailupReport.asmx>

<http://services.mailupnet.it/MailupReport.asmx?WSDL>

Method Login

`string Login(string user, string pwd, string url)`

LoginResult:

```
<LoginResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
  <accessKey><![CDATA[accessKey]]></accessKey>
</LoginResult>
```

Method Logout

`string Logout(string accessKey)`

LogoutResult

```
<LogoutResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
</LogoutResult>
```

Metoto CreateXmlStatistiche

`string CreateXmlStatistiche(string accessKey)`

CreateXmlStatisticheResult

```
<CreateXmlStatisticheResult>
  <errorCode>0</errorCode>
  <errorDescription></errorDescription>
```

```
<list>
  <listID></listID>
  <listGUID></listGUID>
  <newsletters>
    <newsletter idNL="">
      <subject></subject>
      <note></note>
      <dataCreazione></dataCreazione>
      <riassuntivo_grafico></riassuntivo_grafico>
      <aperture_grafico></aperture_grafico>
      <aperture_dettaglio></aperture_dettaglio>
      <aperture_filtri></aperture_filtri>
      <click_grafico></click_grafico>
      <click_dettaglio></click_dettaglio>
      <click_filtri></click_filtri>
      <rapporto_invii></rapporto_invii>
    </newsletter>
  </newsletters>
</list>
</CreateXmlStatisticheResult>
```

Method ReportByUser

string ReportByUser(**string** accessKey, **string** email, **int** listID, **int** messageID)

Email: it will provide data regarding this email address

messageID: if 0, it will provide total data regarding all the messages in this list .

If the email address is unsubscribed, the highlighted data will not be available.

Esempio di ReportByUserResult

```
<ReportByUserResult>
  <errorCode></errorCode>
  <errorDescription></errorDescription>
  <Canali>
    <Email>Iscritto|Disiscritto|InAttesa|Non Registrato</Email>
    <SMS>Iscritto|Disiscritto|InAttesa|Non Registrato</SMS>
    <Posta>Iscritto|Disiscritto|InAttesa|Non Registrato</Posta>
    <Voce>Iscritto|Disiscritto|InAttesa|Non Registrato</Voce>
    <Fax>Iscritto|Disiscritto|InAttesa|Non Registrato</Fax>
  </Canali>
  <Bounce>Data e ora ultimo Bounce</Bounce>
```

```
<Clicks Url="" Totale="">
  <Click DataOra="" IP="" />
  ...
</Clicks>
<Opens Totale="">
  <Open DataOra="" IP="" />
  ...
</Opens>
<CampiAnagrafici>
  <campo id="1" Valore="" />
  ...
  <campo id="39" Valore="" />
</CampiAnagrafici>
</ReportByUserResult>
```

InAttesa means that the user is listed as PENDING.

Method ReportByMessage

`string` ReportByMessage(`string` accessKey, `int` listID, `int` messageID)

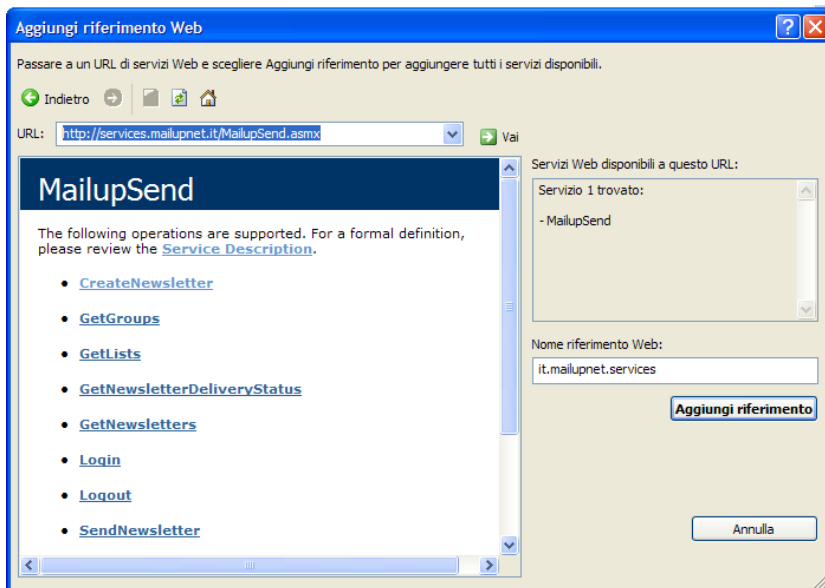
Esempio di ReportByMessageResult

```
<ReportByMessageResult>
  <errorCode></errorCode>
  <errorDescription></errorDescription>
  <Clicks Url="" Totale="">
    <Click Email="" Totale="" />
    ...
  </Clicks>
  <Opens Totale="">
    <Open Email="" Totale="" />
    ...
  </Opens>
</ReportByUserResult>
```

MailUp Send Samples

Codice C#

<http://service.mailupnet.it/MailupSend.asmx>



```

...
it.mailupnet.services.MailupSend myService =
new it.mailupnet.services.MailupSend();
string s = myService.Login(user, password, myURL);
...
// message output parsing
System.Xml.XmlDataDocument docXML = new System.Xml.XmlDataDocument();
docXML.LoadXml(s);

// get the error code
System.Xml.XmlNodeList error =
docXML.GetElementsByTagName("errorCode");
errorCode = Convert.ToInt32(error[0].InnerText);
...
// get the access key
if (errorCode == 0)
{
System.Xml.XmlNodeList key =
docXML.GetElementsByTagName("accessKey");
    accessKey = key[0].InnerText;
}
...

```

logout: C#

```
string s = myService.Logout(accessKey);
```

```
...
```

Get lists: C#

```
it.mailupnet.services.MailupSend myService =  
new it.mailupnet.services.MailupSend();  
string s = myService.Login(user, password, myURL);  
...  
string s = myService.GetLists(accessKey);  
...  
if (errorCode == 0)  
{  
// parsing...  
}
```

Create a message in C#

```
...  
// create the "note" option  
it.mailupnet.services.Option optNote = new  
it.mailupnet.services.Option();  
optNote.Key = "note";  
optNote.Value = "this are notes!";  
  
// enclose a file to the email message  
// 1. encoding base 64  
FileStream fs = new FileStream(@"C:\mydir\webservice.docx",  
FileStream.Open, FileAccess.Read);  
byte[] filebytes = new byte[fs.Length];  
fs.Read(filebytes, 0, Convert.ToInt32(fs.Length));  
string encodedData = Convert.ToBase64String(filebytes,  
Base64FormattingOptions.InsertLineBreaks);  
// 2. Create the option attach01_name" which will contain the file  
name  
it.mailupnet.services.Option optAttach01Name =  
new it.mailupnet.services.Option();  
optAttach01Name.Key = "attach01_name";  
optAttach01Name.Value = "test.docx";
```

```
// 3. Crete the option attach01_data"
it.mailupnet.services.Option optAttach01Data =
new it.mailupnet.services.Option();
optAttach01Data.Key = "attach01_data";
optAttach01Data.Value = encodedData;

// options array:
it.mailupnet.services.Option[] options = new
it.mailupnet.services.Option[3];
options[0] = optNote;
options[1] = optAttach01Name;
options[2] = optAttach01Data;

string html = " HTML code here!! ";
string s = myService.CreateNewsletter(accessKey,
1, "WebService // Newsletter da HTML", "HTML", html, options);
...
if (errorCode == 0)
{
// eseguo il parsing del messaggio di ritorno per ricavare il codice
// identificativo del messaggio creato
}
```

Appendix

Error Code	Error Description
Common Errors	
100	Login error: <exception message>
100	Login error: undefined error
100	Login error: cannot create accessKey
101	accessKey is not valid or is already expired
200	GetLists error: <exception message>
210	GetGroups error: <exception message>
Specific errors for newsletter methods	
220	GetNewsletters error: <exception message>
230	Error creating newsletter: <exception message>
231	Error creating newsletter: cannot create message
300	Error sending newsletter: <exception message>
301	Error creating queue
302	Error creating job

303	Cannot set job as "to send"
304	Cannot start sending: <exception message>
305	Error Scheduling job: <exception message>
306	Cannot schedule delivery for the specified date and time
307	Error creating scheduled job
320	Recipient E-Mail address is invalid
321	E-Mail address <recipient email> is in optout state for list nr. <list>
322	Error delivering message to <recipient email>: <exception message>
400	Error getting delivery status: <exception message>
401	None delivery job been found with the given deliveryID
450	Error reading sending queue: <exception message>
160	Error starting delivery
161	None E-Mail message to be send have been found
162	Cannot start delivery: <exception message>
170	Error sending stop request: <exception message>
<u>Specific errors for SMS methods</u>	
1220	GetSMS error: <exception message>
1230	Error creating sms: <exception message>
1231	Error creating newsletter: returned message id is zero
1300	Error sending newsletter: <exception message>
1301	Error creating queue
1302	Error creating job
1303	Cannot set job as "to send"
1304	Cannot start sending: <exception message>
1305	Error Scheduling job: <exception message>
1306	Cannot schedule delivery for the specified date and time
1307	Error creating scheduled job
1308	Sending denied: NO CREDITS
1309	Number or Prefix missing
1320	Number is invalid
1321	SMS number <recipient number> is in optout state for list nr. <list>
1322	Error delivering message to <recipient number>: <exception message>
1400	Error getting delivery status: <exception message>
1401	None delivery job been found with the given deliveryID
1450	Error reading sending queue: <exception message>
1160	Error starting delivery: <exception message>
1161	None messages to be send have been found

1162	Cannot start delivery: <exception message>
1170	Error sending stop request: <exception message>

Others output samples

Method GetNewsletterDelivery

```
<GetNewsletterDeliveryStatusResult>
  <errorCode>0</errorCode>
  <erroDescription></erroDescription>
  <deliveryStatus>
    <deliveryID>1578</deliveryID>
    <listID>157</listID>
    <newsletterID>1071</newsletterID>
    <status>COMPLETED</status>
    <details>Started at: 23/07/2009 12:06:18, Completed at:
23/07/2009 12:06:18</details>
    <toSend>1</toSend>
    <toBeSent>0</toBeSent>
    <progress>100</progress>
    <estimatedTime>0d 00h 00m 00s</estimatedTime>
  </deliveryStatus>
</GetNewsletterDeliveryStatusResult>
```

a) Scheduled sending output

```
<GetNewsletterDeliveryStatusResult>
  <errorCode>0</errorCode>
  <erroDescription></erroDescription>
  <deliveryStatus>
    <deliveryID>1576</deliveryID>
    <listID>157</listID>
    <newsletterID>1126</newsletterID>
    <status>SCHEDULED</status>
    <details>Delivery will start on 24/07/2009 9:30:00</details>
    <toSend>4</toSend>
    <toBeSent>0</toBeSent>
    <progress>0</progress>
    <estimatedTime>0d 00h 01m 58s</estimatedTime>
  </deliveryStatus>
</GetNewsletterDeliveryStatusResult>
```

b) Sending in progress

```
<GetNewsletterDeliveryStatusResult>
  <errorCode>0</errorCode>
  <erroDescription></erroDescription>
  <deliveryStatus>
    <deliveryID>1579</deliveryID>
    <listID>157</listID>
    <newsletterID>1127</newsletterID>
    <status>SENDING</status>
    <details>Started at: 23/07/2009 14:40:54</details>
    <toSend>59998</toSend>
    <toBeSent>59986</toBeSent>
    <progress>0</progress>
    <estimatedTime>16d 22h 03m 27s</estimatedTime>
  </deliveryStatus>
</GetNewsletterDeliveryStatusResult>
```

6 HELP DESK REFERENCES

- Email support (suggested for technical matters): support@mailup.it
- First level phone support: +1 888-9-MAILUP
- Second level phone support: +39 02 71040485

IMPORTANT: in case of errors always relate the exact text of the error and the link to the form that is being made.