Using SMTP relay from your application

Sending transactional emails from your app with MailUp's SMTP relay

Below you can find some code samples showing how to send a transactional email using MailUp's SMTP relay service (SMTP+) from a client application. You can also send transactional messages using the MailUp API for transactional emails.

Disclaimer: no warranties on sample code

The code samples below are "as is", without warranty of any kind. MailUp shall not be held liable for any direct, indirect or consequential damages or costs of any type arising out of any action taken by you or other related to the example code.

That said, if you run into any issues, please contact us.

Also, let us know if you happen to create code that can be used with other programming languages, so that we can share it on this page (good karma!).

Using a reply-to address different from the sender address

The use inside the header of messages sent with SMTP+, of a REPLY-TO address different from the FROM email address must be enabled by MailUp. If you need this, please contact our support team by writing an email to support@mailup.com.

PHP code sample

This is a PHP snippet that requires PHPMailer and the credentials of a MailUp SMTP+ user.

PHP - Sending transactional emails using SMTP+

```
<?php
//##########################################################
//Include the PHPmailer class.
//
//PHPmailer allows you to create and  transport an email an email message.
//The PHPmailer class is available at this address:
//https://github.com/PHPMailer/PHPMailer/
//##########################################################
require_once "class.phpmailer.php";
$EmailMessage = new PHPmailer();
$EmailMessage->SetLanguage('en','language/');        //Define the language

//**********************************************
//SMTP configuration
//**********************************************
$EmailMessage->IsSMTP();        //Specify usage of SMTP Server
$EmailMessage->Host = "fast.smtpok.com";        //SMTP+ Server address
$EmailMessage->Port="25";        //SET the SMTP Server port
$EmailMessage->Username = "smtpplus_username";        //SMTP+ authentication: username
$EmailMessage->Password = "smtpplus_password";        //SMTP+ authentication: password
$EmailMessage->SMTPAuth = true;        //Authentication required

//**********************************************
//Email data
//**********************************************
$EmailMessage->IsHTML(true);        //Set the email format
$EmailMessage->FromName= "myname";        //From: display name
$EmailMessage->From="sender_address@domain.com";        //From: email address
$EmailMessage->AddAddress("recipient_address@domail.com");        //Add one or more recipients
$EmailMessage->Subject="My first email sent via SMTP+";        //Set the email subject
$EmailMessage->Body="<b>Hello,<b><br/>This is my first email.";        //Set the email body
$EmailMessage->AltBody="Hello, This is my first email..";        //Set the email text part

//**********************************************
//Send the email
//**********************************************
if(!$EmailMessage->Send())
{
echo "<h1>Error sending the email.</h1>";        //Email was not sent
}
else
{
echo "Email has been sent";        //Mail sent
}

?>
```

SMTP+: advanced features and sample code

Host domains

for your transactional sending application you must set:

- fast.smtpok.com (as suggested on MailUp admin console when creating a new SMTP+ user)

```
$EmailMessage->Host = "fast.smtpok.com";        //SMTP+ Server address
```

Advanced settings

Our SMTP relay service supports a **X-SMTPAPI** custom header that allows you to access several features by specifying additional parameters through a JSON object.

The following table includes the parameters that can be added through X-SMTPAPI to override the settings that are defined at the MailUp list or at the SMTP+ user level. None of these fields is mandatory.

X-SMTPAPI value is processed by MailUp but it is not actually added to the headers of the delivered message.

Field name	Description
`CampaignName`	String that is displayed as aggregate message name on admin console
`CampaignCode`	String that is used for message aggregation,overrides list settings. When not specified, MailUp applies the aggregation type of used SMTP+ account
`Header`	Boolean, overrides the list settings about adding default header or not.
`Footer`	Boolean, overrides the list settings about adding default footer or not.
`ClickTracking`	Boolean, overrides the list settings about click tracking.
`ViewTracking`	Boolean, overrides the list settings about open tracking.
`Priority`	Integer, from 1 (highest) to 5 (lowest), overrides SMTP+ account settings
`Schedule`	Used to schedule send. An example of expected format is "2015-02-26T15:55+02:00". Send is immediate if field is equal to "null" or specifies a date-time that occurs in the past
`DynamicFields`	Array of key-value pairs to be used as merge tags ("N" = name or key, "V" = value)

`SkipDynamicFields`	When this field is set to "true", placeholders (i.e. text between square brackets) are not managed as merge tags.
	
	Example:
	
	`SkipDynamicFields= true -> [name] is not replaced by the recipient's name.`
	`SkipDynamicFields= false, null or not specified -> [name] is replaced by the recipient's name.`
	
	Please note that MailUp specific tags like [email] and [idoptin] cannot be skipped, hence the "SkipDynamic Fields= true" setting does not apply to them.

Example

```
{
 "CampaignName":"myCampaign",
 "CampaignCode":"Code001",
 "Header":false,
 "Footer":true,
 "ClickTracking":true,
 "ViewTracking":true,
 "Priority":1,
 "Schedule":null,
 "DynamicFields":[{"N":"itemid","V":"ABC-1234"},{"N":"description","V":"Womens Regular and Wide-
Calf Knee-High Studded Riding Boot"},{"N":"price","V":"$149.99"}],
 "SkipDynamicFields":null
}
```

Use merge tags

Among the various important features, SMTP+ supports merge tags (also called *dynamic fields* in MailUp).

You can use two kinds of merge tags:

- merging values defined in recipient fields
- merging values defined in the header of the message

In both cases, the syntax to be used in the message code is:

- name of the tag in lowercase
- between square brackets

For example, if you created a custom recipient field in your MailUp account called FirstName, the merge tag to use in the subject or body of the email message would be:

```
[firstname]
```

Merge tags based on recipient fields

You can merge tags for any of those recipient fields. Substitution occurs at the time the message issent, when those fields contain a value. Otherwise empty values will be displayed in place of the placeholder.

```
// Merge tags are replaced at sending time by empty values when no value is found for the recipient.
// No action occurs when the merge tag does not correspond to a personal data field.
$EmailMessage->Subject="Hello [firstname]: your order has been shipped!";
$EmailMessage->Body="Hello [firstname],<br /><br />we thought you'd like to know that order
[latestorderid] was <strong>shipped on [latestshippedorderdate]</strong>.<br /><br />Thanks again for
visiting our online store!";
```

Merge tags based on value pairs defined in the message header

With SMTP+ you can reuse a message template (i.e. the same subject and body) and customize it with key-value pairs that are specified through custom headers added to the message. Also tables and HTML content can be used as a merge tag's value.

```
// Placeholder are replaced at sending time by empty values when no value was previously added for
the recipient.
// No action occurs when the specified placeholder does not correspond to a personal data field for
that account.
$EmailMessage->Subject="Recommended books for [customername]";        //Set the email subject
$EmailMessage->Body="<b>Hello [customername], we recommend you [title1] by  [author1]. Buy now at
[price1].<b><br/> Please find enclosed your invoice <b><br/>. Kind regards";        //Set the email
body
$EmailMessage->AddCustomHeader("X-SMTPAPI: {\"DynamicFields\":[{\"N\":\"customername\",\"V\":\"
John\"},{\"N\":\"title1\",\"V\":\"Become an email guru\"},{\"N\":\"price1\",\"V\":\"$10.99\"},{\"N\":
\"author1\",\"V\":\"Martha G.\"}]}");
```

Custom reports based on campaign code

In order to create meaningful statistics for transactional mailings, MailUp aggregates sending data based on a variety of parameters. For example, you can instruct the system to aggregate by message subject, or SMTP+ user, etc.

SMTP+ introduces the ability to aggregate sending data based on a custom ***CampaignCode*** parameter. This allows you to create a customized aggregation when none of the available options suits the desired behavior.

For example, let's say that you wanted to aggregate sending statistics based on a specific keyword that is inside the message subject: there would be no way to meet your needs with the predefined aggregation options that are available in the SMTP+ settings within the MailUp admin console.

The *CampaignCode* parameter comes to the rescue, allowing you to specify a code within the message header and force aggregation based on that code.

An additional parameter called ***CampaignName*** can be used to specify a name to be displayed on inside the MailUp admin console for those.

```
// All the emails with CampaignCode = "Code001" will be treated as different sending of the same
message
// Message name ("subject" field) on admin console is set by the CampaignName of the first call with
CampaignCode = "Code001". CampaignName of further calls with the same Campaign code is ignored.
$EmailMessage->AddCustomHeader("X-SMTPAPI: {\"CampaignName\":\"Custom Aggregation 001\",\"
CampaignCode\":\"Code001\"}");
```

Message scheduling

Another important feature added with SMTP+ is support deferred sending. Specifically, you can now schedule a message so that it is sent at a specified date and time (with time zone).

Note: the message is sent immediately when the specified date and time has already occurred.

```
// Schedule sending on Feb 26th 2015 at 15:55 UTC+2
$EmailMessage->AddCustomHeader("X-SMTPAPI: {\"Schedule\":\"2015-02-26T15:55+02:00\"}");
```

Combining instructions in the message header

Use commas as the separator for the X-SMTPAPI parameters that you wish to set in the message header.

```
// Schedule sending + Set Campaign Name & code
$EmailMessage->AddCustomHeader("X-SMTPAPI: {\"CampaignName\":\"Custom Aggregation 001\",\"
CampaignCode\":\"Code001\",\"Schedule\":\"2015-02-26T15:55+02:00\"}");
```