

SMTP Connector

The **SMTP Connector** allows you to perform email operations using protocol which is known as SMTP (Simple Mail Transfer Protocol). You can choose the required operation from the dropdown using templates from your BPMN process.

Prerequisites

To start working with the **SMTP Connector**, an email account must be created to a particular mailbox which acts as host for SMTP connection. The following parameters are necessary for establishing smtp connection.

- **Hostname:** A hostname is a unique name or label assigned to any device connected to a computer network, in this case its location where email server is hosted. It enables us to establish an smtp connection by email server address through which email operations has to be implemented.
- **Username: & Password:** These are the user credentials which enables the server to quickly identify the unique user for providing privileges.
- **Port Number:** SMTP mailbox port number.

Create an SMTP connector task

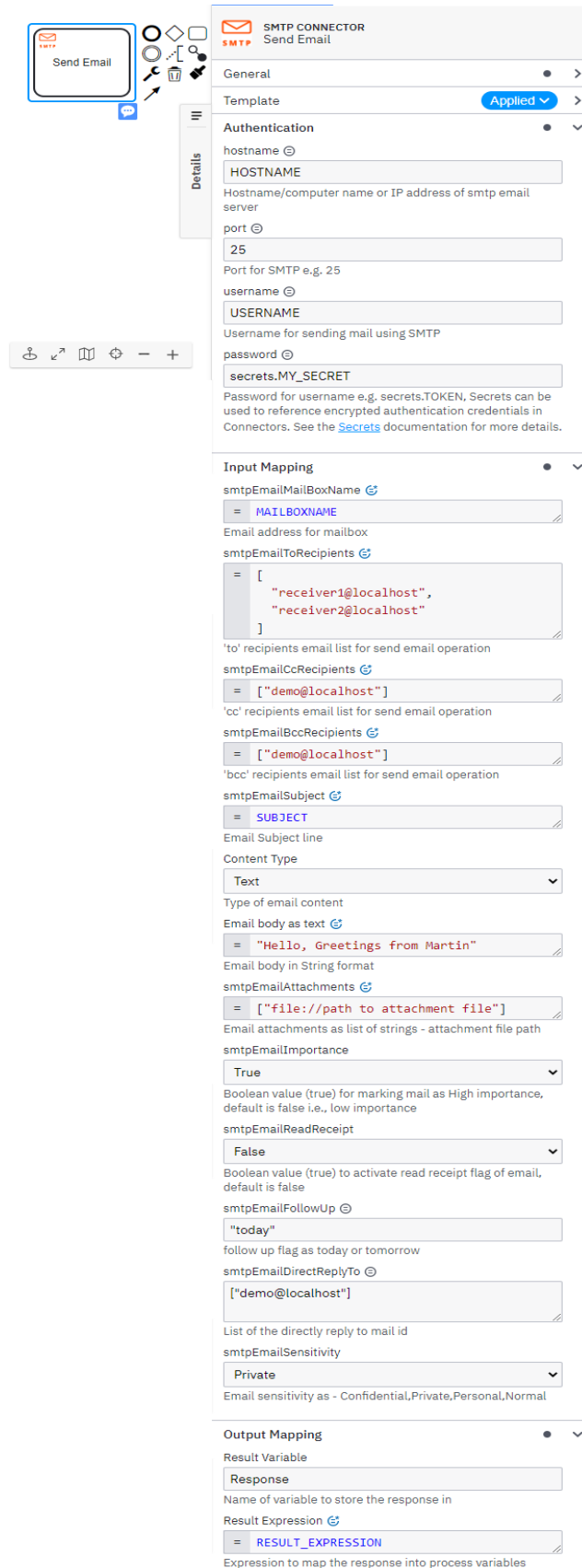
To use an SMTP Connector in your process, either change the type of existing task by clicking on it and using the wrench-shaped Change type context menu icon or create a new Connector task by using the Append Connector context menu. Follow our [guide on using Connectors](#) to learn more.

Sending Emails through SMTP Connector

To make the **SMTP Connector** executable, fill out the mandatory fields highlighted in red in the properties panel.

SMTP Connector server connection object takes – host, username, password and port number. e.g., localhost, username, password (as secrets Token e.g., secrets.MY_SECRET), 5170.

Firstly, we must initiate the task using BPMN diagram in either desktop modeler or web modeler. And in this example, we are using desktop modeler.



SMTP CONNECTOR
Send Email

General

Template: **Applied**

Authentication

hostname: HOSTNAME
Hostname/computer name or IP address of smtp email server

port: 25
Port for SMTP e.g. 25

username: USERNAME
Username for sending mail using SMTP

password: secrets.MY_SECRET
Password for username e.g. secrets.TOKEN, Secrets can be used to reference encrypted authentication credentials in Connectors. See the [Secrets](#) documentation for more details.

Input Mapping

smtpEmailMailBoxName: MAILBOXNAME
Email address for mailbox

smtpEmailToRecipients: ["receiver1@localhost", "receiver2@localhost"]
'to' recipients email list for send email operation

smtpEmailCcRecipients: ["demo@localhost"]
'cc' recipients email list for send email operation

smtpEmailBccRecipients: ["demo@localhost"]
'bcc' recipients email list for send email operation

smtpEmailSubject: SUBJECT
Email Subject line

Content Type: Text
Type of email content

Email body as text: "Hello, Greetings from Martin"
Email body in String format

smtpEmailAttachments: ["file://path to attachment file"]
Email attachments as list of strings - attachment file path

smtpEmailImportance: True
Boolean value (true) for marking mail as High importance, default is false i.e., low importance

smtpEmailReadReceipt: False
Boolean value (true) to activate read receipt flag of email, default is false

smtpEmailFollowUp: "today"
follow up flag as today or tomorrow

smtpEmailDirectReplyTo: ["demo@localhost"]
List of the directly reply to mail id

smtpEmailSensitivity: Private
Email sensitivity as - Confidential, Private, Personal, Normal

Output Mapping

Result Variable: Response
Name of variable to store the response in

Result Expression: RESULT_EXPRESSION
Expression to map the response into process variables

Following are the further stages involved in the process:

Adding authentication details, input parameters involved in the process of email transfer from sender to receiver, output parameters and error parameters.

1. Authentication Details:

- a) Provide details related to hostname, port number, username and password.
- b) As password is sensitive credential, use secrets token to store it internally and access it through secrets.MY_SECRET.

2. Input Mapping:

Input Mapping involves following input parameters.

a) MailBox:

In smtpEmailMailBoxName field, we must enter mailbox name through which we are establishing smtp connection. (e.g., abc@gmail.com). It's the source from where we want to send an email.

b) EmailToRecipients:

The smtpEmailToRecipients field requires list of all recipient email Ids. The email content sent to each recipient that is listed in this field. And It can't be empty. It requires at least one recipient email id in the list.

For example, [user1@hostname, user2@hostname, user3@hostname]

c) EmailCcRecipients:

smtpEmailCcRecipients (Carbon Copy - CC) mean list of recipients who also able to get email which is transferring from sender to receiver and all (sender, receiver, CcRecipient) will know that message has been sent to other Cc recipients also.

For example, [user1@hostname, user2@hostname, user3@hostname]

d) EmailBccRecipients:

smtpEmailBccRecipients (Blind Carbon Copy - BCC) mean list of recipients who also able to get email which is transferring from sender to receiver. But, In the Case of smtpEmailBccRecipients (Blind Carbon Copy) one in bcc list unable to know that who the other bcc recipients also got the same email message. And Rest is all same about CC & BCC.

For example, [user1@hostname, user2@hostname, user3@hostname]

e) Subject:

The smtpEmailSubject field corresponds to the subject(title) of the mail. The subject of an email depicts the purpose or overview of the email body.

For Example, “Request for an Appointment”, “Confirmation of Application Submission.”

f) Content Type:

This field requires mail content type. It can be either text or html. Text is chosen if the mail body is in the form of manually typed textual content. And html is chosen if mail body contain web pages.

g) Body:

This field requires email message body. It is the actual message that is being transferred from sender to receiver(s) over an established smtp connector through email server. The detailed representation of message is placed in this field.

If content type is text - body should be string.
e.g., "Hi, \n Greetings from Camunda Team!!"

If content type is html - provide email body html file.
e.g., "D:/emailContent.html"

h) Email Attachments:

The smtpEmailAttachments field, requires list of file attachments to be listed as part of the email for enabling reference related to the content provided in the corresponding email. It requires zero or more attachments, because adding file attachments is not mandatory because it's our choice based on the content that has been transferring from sender to receiver(s). So, the list can be empty.

i) Importance:

The Importance marker enables the users to prioritize and categorize their emails based on whether emails are highly important or less important. So, smtpEmailImportance helps in marking importance label of the email. It can be either high or low. It provides a drop-down menu, and we must select whether it is high or low based on our requirement.

j) Read Receipts:

The Read Receipts feature enables the confirmation to the sender based on whether an email is opened by the recipient(s) or not. It is very helpful in tracking the delivery of the corresponding email and used to know whether it's successfully delivered and engaged with the recipient or not. The smtpEmailReadReceipt provides a drop-down menu and one can select true or false.

Selecting “true” enables the read receipts tracking of the email delivery for the user and selecting “false” just ignores this service.

k) Follow Up:

The smtpFollowUp feature provides a follow-up flag. It provides a drop-down menu and we have to select whether it is today or tomorrow. Basically, It is a reminder message the one can send after the business virtual meetings, phone calls which can include thanking the others

and providing them the contact and professional information.

It can also be sent to receiver of one's previous email if there is no response from them for the previous mail. We must provide when it has to occurred and after the specified time follow up will get enabled.

l) Direct Reply:

The smtpEmailDirectReplyTo field requires list of all email Ids to which the current email would have to be directly replied.

m) Sensitivity:

The smtpEmailSensitivity field provides a drop-down menu in which we must select one item where each one is "confidential", "personal", "private", "normal". The Sensitivity of an email corresponds to different categories such as Confidential, Personal, Private and Normal.

Through this we can enable the receiver(s) of an email treat the email message as corresponding sensitivity label. The receivers who receive the emails with private sensitivity label are unable to forward or redirect those emails as per their inbox rules.

3. Output Mapping:

- a) We maintain Result variable which stores the acknowledge message which tells the status of sending email through smtp connector as SUCCESS or FAIL.
- b) We also maintain Result Expression in which we can declare new process variables that are used in the process engine from the Response Data of Result Variable.

4. Error Mapping:

- a) We maintain Error Expression in Error Mapping so that if there is any error occurred during sending the email through smtp connector, the error expression performed.

(e.g., bpmnError("404", "Error in Sending Mail!!")).

- b) We can make use of FEEL expressions to let Error expression act based on the kind of error it gets as below:

```
If error.code == "404" then
    bpmnError("404", "Got a 404 Error!!")
else if error.code == "500" then
    bpmnError("500", "Got a 500 Error!!")
else
    null
```

The SMTP Connector makes use of Camunda Connector SDK (for performing various tasks like authentication, requests, responses, service tasks) along which takes all the input parameters and apply corresponding service tasks based on the inputs. After that, it stores the result in result variables and errors in error variables which get mapped to Output mapping and error mapping fields and specified action is performed. The workflow of this entire process is carried out by engine. In this way, SMTP Connector is implemented to send Emails.

Appendix & FAQ

How can I authenticate my SMTP Connector?

The **SMTP Connector** needs the credentials for connection. Hostname (host) – of the server where database is hosted, Port (port) – on which smtp server is running, Username (username) – User with proper privilege for operation and Password (password) – User password, which need to be saved as a Token in Secret vault and input can be provided as: secrets.TOKEN_NAME

What is smtpEmailReadReceipt input?

It is an email feature through which one can avail the confirmation about whether the email message has been seen by recipient and whether the recipient opened the email or not. The accepted values are True/False.

What is smtpEmailSensitivity input?

The Sensitivity of an email corresponds to different categories such as Confidential, Personal, Private and Normal. Through this we can enable the receiver(s) of an email to treat the email message as corresponding sensitivity label. The receivers who receive the emails with private sensitivity label are unable to forward or redirect those emails as per their inbox rules.

What are smtpEmailCcRecipients & smtpEmailBccRecipients?

smtpEmailCcRecipients (Carbon Copy) mean list of recipients who also able to get email which is transferring from sender to receiver and all (sender, receiver, CcRecipient) will know that message has been sent to other Cc recipients also.

But, In the Case of smtpEmailBccRecipients (Blind Carbon Copy) one in bcc list unable to know that who the other bcc recipients also got the same email message. And Rest is all same about CC & BCC.