Send Email from Service Suite

You can send an email using the Service Suite email client, using both OAuth and legacy email clients (such as gmail). with To, From, cc, subject, body, bcc parameters and attachments.

OAuth cannot send emails on behalf of a user; they must be sent from a service account that acts on behalf of the application itself. This is similar to the process documented for Azure setup in [Configuring Email Clients for Outlook 365 Accounts using Oauth2.0.]

Password required

The following method allows you to send email, but does not require a password.

```
var mailService = createOAuthMailClient("dummy@example.com", "QWERTYpoi123!",
    "QWPg646HDmAVa8dnABtTfdKgd6eZPujvtXXYYKM=", "7899fa0a-a899-481c-bdc2-132eec26f708",
    "d9d0bb61-6874-4c0e-8cae-59ad5bce1d4e");

mailService.loginWithOAuth();

var msg = mailService.createMailMessage("Email subject","Email body");

//add recipients
msg.addTo(["user1@appvance.com"]);
msg.addCC(["user2@appvance.ai"]);
msg.addBCC(["user3@appvance.com"]);

//add attachment
msg.addAttachment("prueba.jpeg", readBytes("{ds}/test.jpeg"));

//verify email was sent
var sent = mailService.sendEmail(msg);
```

```
assertTrue(sent);

//****validations****
wait(10000);

//refresh folder or switch to another folder
mailService.setFromFolder("INBOX");

//get last unread message
var lastUnread = mailService.getLastUnreadMessages(1);
assertTrue(lastUnread.length === 1);

//get all emails
var allEmails = mailService.getAllMessages();
assertTrue(allEmails.length === mailService.getMessageCount());



mailService.logout();
```

Password not required

The following method allows you to send email, but does not require a password. This method uses `GraphServiceClient`.

```
public string SendEmail(string fromAddress, string toAddress, string CcAddress, string
subject, string message, string tenanatID , string clientID , string clientSecret)
    {
            try
            {

            var credentials = new ClientSecretCredential(
                                tenanatID, clientID, clientSecret,
                            new TokenCredentialOptions { AuthorityHost =
AzureAuthorityHosts.AzurePublicCloud });
            GraphServiceClient graphServiceClient = new GraphServiceClient(credentials);



            string[] toMail = toAddress.Split(',');
```

```
List<Recipient> toRecipients = new List<Recipient>();
int i = 0;
for (i = 0; i < toMail.Count(); i++)
{
    Recipient toRecipient = new Recipient();
    EmailAddress toEmailAddress = new EmailAddress();

    toEmailAddress.Address = toMail[i];
    toRecipient.EmailAddress = toEmailAddress;
    toRecipients.Add(toRecipient);
}

List<Recipient> ccRecipients = new List<Recipient>();
if (!string.IsNullOrEmpty(CcAddress))
{
    string[] ccMail = CcAddress.Split(',');
    int j = 0;
    for (j = 0; j < ccMail.Count(); j++)
    {
        Recipient ccRecipient = new Recipient();
        EmailAddress ccEmailAddress = new EmailAddress();

        ccEmailAddress.Address = ccMail[j];
        ccRecipient.EmailAddress = ccEmailAddress;
        ccRecipients.Add(ccRecipient);
    }
}
var mailMessage = new Message
{
    Subject = subject,

    Body = new ItemBody
    {
        ContentType = BodyType.Html,
        Content = message
    },
    ToRecipients = toRecipients,
    CcRecipients = ccRecipients

};
```

```
            // Send mail as the given user.
            graphServiceClient
                .Users[fromAddress]
                 .SendMail(mailMessage, true)
                 .Request()
                 .PostAsync().Wait();

            return "Email successfully sent.";

        }
        catch (Exception ex)
        {

            return "Send Email Failed.\\r\\n" + ex.Message;
        }
}
```

Last updated: January 16, 2024

If you have any feedback on this topic, please contact us at documentation@appvance.ai

---

Revision #8
Created 2026-07-31 15:39:36 UTC by Admin
Updated 2026-11-17 22:06:07 UTC by AI_Mananger