Mailit: OAuth 2.0 Setup Guide for Microsoft 365 and Outlook

Mailit Documentation

Version for Mailit 11

Contents

1 Introduction **4**

1.1 Key OAuth Concepts . . . 4

1.2 Why OAuth Authentication Is Required . . . 4

2 Quick Start — End-to-End Setup **5**

2.1 Personal Account . . . 5

2.1.1 OAuth for IMAP / POP / SMTP . . . 5

2.1.2 OAuth for Microsoft Graph . . . 7

2.2 Corporate Account — Own Mailbox . . . 8

2.2.1 OAuth for IMAP / POP / SMTP . . . 8

2.2.2 OAuth for Microsoft Graph . . . 9

2.3 Corporate Account — Shared Mailbox . . . 10

2.3.1 OAuth for Microsoft Graph (Delegated) . . . 10

2.3.2 OAuth for Microsoft Graph (Application) . . . 11

2.4 Server-Side — No User Login . . . 11

2.4.1 OAuth for IMAP / POP / SMTP (Application) . . . 11

2.4.2 OAuth for Microsoft Graph (Application) . . . 12

3 Settings Templates in Mailit Examples **13**

3.1 How Templates Work . . . 13

3.2 Available OAuth Presets . . . 13

4 Scenarios and Transports — Reference **15**

4.1 Scenario 1: Delegated + Authorization Code . . . 15

4.2 Scenario 2: Application + Client Credentials . . . 15

4.3 Which Transport to Choose . . . 16

4.4 Differences Between Transports and Scenarios . . . 16

5 Account Types — Reference **17**

6 Application Registration — Reference **17**

7 Permissions — Reference **18**

7.1 Where to Add Permissions . . . 18

7.2 Common Permissions . . . 18

7.3 Admin Consent . . . 18

8 Mailit Connection Fields — Reference **19**

9 Server Addresses and Ports — Reference **19**

9.1 Server Addresses . . . 19

9.2 Ports . . . 19

9.3 Local Web Server and Mailit Assets . . . 20

9.3.1 Deploying Mailit Assets . . . 20

9.3.2 Available URLs . . . 20

9.3.3 Plugin Functions for the Local Server . . . 20

10 Authorization Methods — Reference **21**

10.1 Browser Login (Scenario 1) . . . 21

10.2 No Login (Scenario 2) . . . 21

11 Shared Mailboxes — Reference **21**

11.1 Requirements . . . 21

11.2 Configuring Shared Mailbox Permissions . . . 21

11.3 Choosing a Transport for a Shared Mailbox . . . 21

12 Token Lifecycle and Automatic Refresh **21**

12.1 Scenario 1 — Refresh Token . . . 21

12.2 Scenario 2 — Client Credentials . . . 22

13 Working in a Multi-User Environment **22**

13.1 What Is Not Recommended . . . 22

13.2 Recommended Practices . . . 22

14 Service Limits and Scaling **22**

14.1 Microsoft Graph Limits (Outlook Mail) . . . 22

14.2 Exchange Online Limits . . . 23

14.3 Demo Applications . . . 23

14.4 Scaling Across Multiple Applications . . . 23

15 Troubleshooting **23**

15.1 Diagnostic Information . . . 23

15.2 Local Web Server Errors (-30 / -61) . . . 23

15.3 Where the Plug-in Log Is Located 24
15.4 Enabling the Plug-in Log 24
15.5 Common OAuth Errors 24

1 Introduction

Mailit supports high-security authentication standards. Some providers, such as Microsoft 365 and Google, require OAuth 2.0 (OAuth). This means Mailit does not handle your login credentials (username and password) at all. Instead, OAuth works by approving Mailit as an application and letting it access your mail.

This process involves authorization directly with your provider. You sign in on their website, and they issue a temporary token to Mailit. You can control and revoke this token at any time in your account settings. If the provider changes their security policy, they may ask you to reauthorize your account to refresh the token.

1.1 Key OAuth Concepts

Application Registration.

Before using OAuth, you must register an application in the provider's developer portal. You provide an Application Name and a **Redirect URI** (the address where the provider will send the user after authorization).

Client ID and Client Secret.

After registration, the provider issues credentials:

- **Client ID:** A publicly exposed string used to identify your application and build authorization URLs.
- **Client Secret:** A private string used to authenticate the application. *Note: In modern public-client scenarios (like native desktop apps), the secret is often not required or is treated as public.*

Tokens.

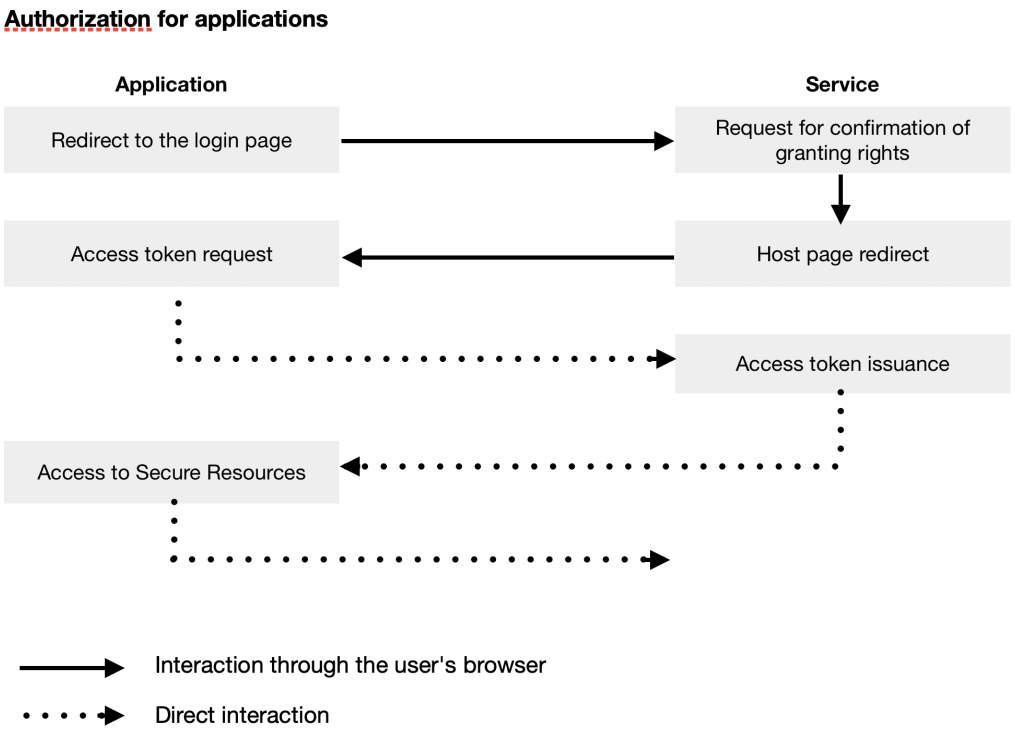
When you authorize the application, the provider issues tokens:

- **Access Token:** A random string used to access your data. It has a limited lifetime (usually 1 hour).
- **Refresh Token:** Issued alongside the Access Token. Mailit uses this to automatically request a new Access Token when the current one expires, so you don't have to sign in again.

1.2 Why OAuth Authentication Is Required

> ***Important.*** *Microsoft has disabled classic authentication methods (Basic Authentication) for IMAP and POP protocols. For SMTP AUTH, Microsoft follows a separate retirement schedule. The most reliable and future-proof method is* ***OAuth 2.0****. For Microsoft Graph and Gmail REST API, OAuth is mandatory.*

Therefore, everywhere in this guide where IMAP / POP / SMTP or REST API configuration is mentioned, it means **OAuth authentication setup**.



2 Quick Start — End-to-End Setup

This section contains a complete, self-contained setup for each typical case. Find the one that matches your situation and follow the steps in order.

> ***Before deploying to production**, read section 14 "Service Limits and Scaling". It explains how Microsoft counts quotas and why a shared application across many users can run into throttling.*

2.1 Personal Account

You have a personal Microsoft account: `outlook.com`, `hotmail.com`, or `live.com`. You are setting up OAuth yourself — no administrator is involved.

2.1.1 OAuth for IMAP / POP / SMTP

This guide explains how to connect Mailit to your personal Outlook.com, Hotmail, or Live mailbox using OAuth 2.0. You will receive email through IMAP or POP and send email through SMTP.

The connection uses delegated permissions with the Authorization Code flow. You sign in to Microsoft during initial setup and approve access to your mailbox.

Step 1: Prepare your account

For a personal Outlook.com account, you must enable the receiving protocol. Go to your Outlook.com settings under **Settings** → **Mail** → **Forwarding and IMAP**. Turn on access for IMAP or POP and save your changes.

> *After enabling IMAP/POP, if you have two-step verification enabled, you may need to confirm your recent sign-in activity. Check `https://account.live.com/activity` and approve the login if prompted.*

Step 2: Register the application

Sign in to the Microsoft Entra admin center (`https://entra.microsoft.com`), open **App registrations**, and select **New registration**.

- **Name:** Enter a name such as `Mailit OAuth`.
- **Supported account types:** Select **Accounts in any organizational directory and personal Microsoft accounts**.
- **Redirect URI:** Select the **Public client/native (mobile & desktop)** platform and enter `http://localhost:1234/oauth/`.

Click **Register** and copy the **Application (client) ID**.

> ***Client secret:*** *Because this is a native/public-client registration, you must not use a client secret. Leave the OAuth Secret field empty and keep `OAuth Add Secret To Request` turned off.*

Step 3: Add delegated permissions

Go to **API permissions** → **Add a permission** → **Microsoft Graph** → **Delegated permissions**. Add:

- `IMAP.AccessAsUser.All` — receive and manage email through IMAP.
- `POP.AccessAsUser.All` — receive email through POP (omit if you only use IMAP).
- `SMTP.Send` — send email through SMTP.
- `offline_access` — request a refresh token.

Step 4: Configure OAuth in Mailit

Open `Simple Receiving.fmp12`. Select **IMAP** (or **POP**) as the connection type and **OAuth** as the authentication method.

- On the **OAuth 2.0 Client Credentials** tab, select the `Microsoft Mail (Delegated)` preset.
- Enter your Application (client) ID in **OAuth Client ID**.
- Enter your full mailbox address in **User Name**.

> *In the **User Name** field, you must use your primary Microsoft account address (e.g., `user@outlook.com`), not an alias or a custom domain address.*

Step 5: Check the receiving and sending settings

Use the following connection settings:

Connection	Server	Port
IMAP receiving	`outlook.office365.com`	993 (SSL/TLS)
POP receiving	`outlook.office365.com`	995 (SSL/TLS)
SMTP sending	`smtp-mail.outlook.com`	587 (STARTTLS)

Step 6: Sign in and authorize access

Ensure the **Mailit Assets** folder is installed next to the `Mailit.fmplugin` file. If it is missing, open `Install Plug-in.fmp12` and click **Install Assets**. Click **Open OAuth2 in External browser**, sign in, and approve permissions.

Step 7: Test receiving and sending

Click **Test Connection**, then **Get Mail**. To test sending, configure `Simple Sending.fmp12` using OAuth and the same mailbox identity.

2.1.2 OAuth for Microsoft Graph

This guide explains how to connect Mailit to your personal Outlook.com mailbox using the Microsoft Graph REST API. The connection uses delegated permissions with the Authorization Code flow.

Step 1: Prepare your account

No special preparation is required for personal accounts when using Microsoft Graph. You do not need to enable IMAP or POP in Outlook settings.

Step 2: Register the application

Sign in to the Microsoft Entra admin center, open **App registrations**, and select **New registration**.

- **Name:** Enter a name such as `Mailit Graph`.
- **Supported account types:** Select **Accounts in any organizational directory and personal Microsoft accounts**.
- **Redirect URI:** Select the **Public client/native (mobile & desktop)** platform and enter `http://localhost:1234/oauth/`.

Click **Register** and copy the **Application (client) ID**.

> ***Client secret:*** *This is a public client. Leave the OAuth Secret field empty in Mailit and keep* `OAuth Add Secret To Request` *turned off.*

Step 3: Add delegated permissions

Go to **API permissions** → **Add a permission** → **Microsoft Graph** → **Delegated permissions**. Add:

- `Mail.ReadWrite` — read and manage email (does not allow sending).
- `Mail.Send` — send email.
- `User.Read` — read basic profile.
- `offline_access` — request a refresh token.

Step 4: Configure OAuth in Mailit

Open `Simple Receiving.fmp12`. Select **graphRest** as the connection type and **OAuth** as the authentication method.

- On the **OAuth 2.0 Client Credentials** tab, select the `Microsoft Graph (Delegated)` preset.
- Enter your Application (client) ID in **OAuth Client ID**.
- Enter your primary mailbox address in **User Name**.

Step 5: Sign in and authorize access

Ensure the **Mailit Assets** folder is installed. Click **Open OAuth2 in External browser**, sign in to your Microsoft account, and approve the requested permissions.

Step 6: Test receiving and sending

Click **Test Connection**. After a successful connection, open the **Receive Email** layout and click **Get Mail**. To test sending, use the sending example or your solution's send script with the same Graph connection.

2.2 Corporate Account — Own Mailbox

You have a corporate Microsoft 365 account. You want to work with **your own mailbox**. Depending on your organization, the application may be registered by you or by your administrator.

2.2.1 OAuth for IMAP / POP / SMTP

This guide explains how to connect Mailit to your corporate Microsoft 365 mailbox using OAuth 2.0 with IMAP, POP, and SMTP. The connection uses delegated permissions.

Step 1: Prepare your account

Your administrator may register the application for you. If so, request the **Application (client) ID** and confirm whether administrator approval is required. The administrator must also ensure SMTP AUTH is permitted for your mailbox in Exchange Online.

Step 2: Register the application

If you register it yourself:

- Sign in to the Microsoft Entra admin center and select **New registration**.
- **Name:** `Mailit OAuth`.
- **Supported account types: Accounts in this organizational directory only** (Single tenant).
- **Redirect URI: Public client/native (mobile & desktop)**, value `http://localhost:1234/oauth/`.

Click **Register** and copy the **Application (client) ID** and **Directory (tenant) ID**.

> ***Client secret:*** *This is a public client. Leave the OAuth Secret field empty in Mailit and keep* `OAuth Add Secret To Request` *turned off.*

Step 3: Add delegated permissions

Go to **API permissions** → **Add a permission** → **Microsoft Graph** → **Delegated permissions**. Add `IMAP.AccessAsUser.All`, `POP.AccessAsUser.All` (if needed), `SMTP.Send`, and `offline_access`. Click **Grant admin consent for [your organization]**.

Step 4: Configure OAuth in Mailit

Open `Simple Receiving.fmp12`, select **IMAP** (or **POP**), and choose **OAuth**.

- Select the `Microsoft Mail (Delegated)` preset.
- Enter the Application (client) ID in **OAuth Client ID**.
- Enter your corporate email address in **User Name**.

On the **OAuth 2.0 Additional Settings** tab, replace `common` in both the Authorization and Token URLs with your **Directory (tenant) ID**.

Step 5: Check the receiving and sending settings

Use `outlook.office365.com` on port 993 for IMAP and `smtp.office365.com` on port 587 for SMTP (STARTTLS).

Step 6: Sign in and authorize access

Ensure the **Mailit Assets** folder is installed. Click **Open OAuth2 in External browser**, sign in with your corporate account, and approve permissions.

Step 7: Test receiving and sending

Click **Test Connection**, then **Get Mail**. For sending, configure `Simple Sending.fmp12` with the same tenant ID, Client ID, and email address.

2.2.2 OAuth for Microsoft Graph

This guide explains how to connect Mailit to your corporate Microsoft 365 mailbox using the Microsoft Graph REST API. The connection uses delegated permissions.

Step 1: Prepare your account

Request the **Application (client) ID** and **Directory (tenant) ID** from your administrator if they manage the application registration.

Step 2: Register the application

If you register it yourself:

- Sign in to the Microsoft Entra admin center and select **New registration**.
- **Name:** `Mailit Graph`.
- **Supported account types: Accounts in this organizational directory only** (Single tenant).
- **Redirect URI: Public client/native (mobile & desktop)**, value `http://localhost:1234/oauth/`.

Copy the Application (client) ID and Directory (tenant) ID.

> ***Client secret:*** *Not required for this delegated public-client setup.*

Step 3: Add delegated permissions

Go to **API permissions** → **Microsoft Graph** → **Delegated permissions**. Add `Mail.ReadWrite`, `Mail.Send`, `User.Read`, and `offline_access`. Click **Grant admin consent for [your organization]** (requires administrator privileges).

Step 4: Configure OAuth in Mailit

Open `Simple Receiving.fmp12`, select **graphRest**, and choose **OAuth**.

- Select the `Microsoft Graph (Delegated)` preset.
- Enter the Application (client) ID.
- Enter your corporate email in **User Name**.

On the **OAuth 2.0 Additional Settings** tab, replace `common` with your **Directory (tenant) ID** in both URLs.

Step 5: Sign in and authorize access

Ensure the **Mailit Assets** folder is installed. Click **Open OAuth2 in External browser**, sign in with your corporate account, and approve permissions.

Step 6: Test receiving and sending

Click **Test Connection**, then **Get Mail**. For sending, use the same Graph connection settings in your send script.

2.3 Corporate Account — Shared Mailbox

You have a corporate Microsoft 365 account and need to work with a **shared mailbox**. This is done via Microsoft Graph. Your administrator must grant you access.

2.3.1 OAuth for Microsoft Graph (Delegated)

This guide explains how to connect Mailit to a shared mailbox using Microsoft Graph and delegated permissions. You will sign in with your own corporate account but access the shared mailbox.

Step 1: Prepare your account

Your administrator must grant your user account permission to access the shared mailbox. You need **Read and manage** (FullAccess) to receive email, and **Send as** to send email.

They can do this via PowerShell:

```
# Grant Read and Manage (Full Access)
Add-MailboxPermission -Identity "shared@company.com" -User "user@company.com
    " -AccessRights FullAccess -AutoMapping:$false

# Grant Send As
Add-RecipientPermission -Identity "shared@company.com" -Trustee "
    user@company.com" -AccessRights SendAs -Confirm:$false
```

Step 2: Register the application

Request the **Application (client) ID** from your administrator. This is a public client, so no Client Secret is required.

Step 3: Add delegated permissions

Go to **API permissions** → **Microsoft Graph** → **Delegated permissions**. Add the following permissions:

- `offline_access`
- `https://graph.microsoft.com/Mail.Read.Shared`
- `https://graph.microsoft.com/Mail.ReadWrite.Shared`
- `https://graph.microsoft.com/Mail.Send.Shared`

Click **Grant admin consent**.

Step 4: Configure OAuth in Mailit

Open `Simple Receiving.fmp12`, select **graphRest**, and choose **OAuth**.

- Select the `Microsoft Graph (Shared Mailbox)` preset.
- Enter the Application (client) ID.
- Enter the **shared mailbox email address** in **User Name** (e.g., `shared@company.com`).

Step 5: Sign in and authorize access

Ensure the **Mailit Assets** folder is installed. Click **Open OAuth2 in External browser**. Sign in with **your own** corporate account (not the shared mailbox). Approve the permissions.

Step 6: Test receiving and sending

Click **Test Connection**, then **Get Mail**. To test sending, use the same preset and shared mailbox address in your send script.

2.3.2 OAuth for Microsoft Graph (Application)

This guide explains how to access a shared mailbox using Microsoft Graph and Application permissions, without interactive user login. This setup requires administrator access.

Step 1: Prepare your account (Administrator)

You are performing this setup as a Microsoft 365 administrator. Application permissions grant access to all mailboxes in the tenant. To restrict access to a specific shared mailbox, you must configure Exchange Application RBAC (Resource Scoping) in Exchange Online.

Step 2: Register the application

- Sign in to the Microsoft Entra admin center as an administrator.
- Create a **New registration**.
- **Name:** `Mailit Service`.
- **Supported account types: Accounts in this organizational directory only** (Single tenant).
- **Redirect URI:** Leave blank.

Copy the **Application (client) ID** and **Directory (tenant) ID**. Go to **Certificates & secrets** and create a new **Client Secret**. Copy the secret value immediately.

Step 3: Add application permissions

Go to **API permissions** → **Microsoft Graph** → **Application permissions**. Add `Mail.Read`, `Mail.ReadWrite`, and `Mail.Send`. Click **Grant admin consent**.

Step 4: Configure OAuth in Mailit

Open `Simple Receiving.fmp12`, select **graphRest**, and choose **OAuth**.

- Select the `Microsoft Graph (Application)` preset.
- Enter the Application (client) ID and the Client Secret Value.
- On the **Additional Settings** tab, replace `{tenant}` with your Directory (tenant) ID in both URLs.
- Enter the **shared mailbox email address** in **User Name**.

Step 5: Test receiving and sending

Ensure the **Mailit Assets** folder is installed. No browser login is required; the application authenticates directly. Click **Test Connection** and **Get Mail**. To send, use the same configuration in your sending script.

2.4 Server-Side — No User Login

You are setting up background mail processing on a server, with no interactive user login. This requires Application permissions and admin consent. Only a Microsoft 365 administrator can perform this setup.

2.4.1 OAuth for IMAP / POP / SMTP (Application)

This guide explains how to configure background mail processing via IMAP/SMTP using Application permissions. No user is involved.

Step 1: Prepare your account (Administrator)

You are performing this setup as a Microsoft 365 administrator. Verify that SMTP AUTH is enabled for the target mailbox in Exchange Online.

Step 2: Register the application

Create a **New registration**. Use **Single tenant**, leave the Redirect URI blank, and name it `Mailit Service`. Copy the **Application (client) ID** and **Directory (tenant) ID**. Create a **Client Secret** and copy its value.

Step 3: Add application permissions

Go to **API permissions** → **Add a permission** → **APIs my organization uses** → **Office 365 Exchange Online** → **Application permissions**. Add `IMAP.AccessAsApp`, `POP.AccessAsApp` (if needed), and `SMTP.SendAsApp`. Click **Grant admin consent**.

Step 4: Register Service Principal in Exchange Online (Critical)

Entra ID permissions are not enough. You must register the service principal and grant it access to the target mailbox via PowerShell:

```
# Connect to Exchange Online
Connect-ExchangeOnline

# Register the Service Principal
# Note: ServiceId is the Object ID of the Enterprise Application in Entra ID
    .
New-ServicePrincipal -AppId "YOUR-CLIENT-ID" -ServiceId "YOUR-OBJECT-ID" -
    DisplayName "Mailit Service"

# Grant access to the target mailbox (User is the AppId, not an email)
Add-MailboxPermission -Identity "target@company.com" -User "YOUR-CLIENT-ID"
    -AccessRights FullAccess
```

Step 5: Configure OAuth in Mailit

Open `Simple Receiving.fmp12`, select **IMAP**, and choose **OAuth**.

- Select the `Microsoft Mail (Application)` preset.
- Enter the Client ID and Client Secret Value.
- On the **Additional Settings** tab, replace `{tenant}` with your Directory (tenant) ID.
- Enter the target mailbox email in **User Name**.

Step 6: Test receiving and sending

Ensure the **Mailit Assets** folder is installed. No browser login is required. Click **Test Connection**, then **Get Mail**. Use the same preset for sending.

2.4.2 OAuth for Microsoft Graph (Application)

This guide explains how to configure background mail processing via Microsoft Graph using Application permissions. No user is involved.

Step 1: Prepare your account (Administrator)

You are performing this setup as a Microsoft 365 administrator. Ensure that your organization's Conditional Access policies allow application-based authentication.

Step 2: Register the application

Create a **New registration**. Use **Single tenant**, leave the Redirect URI blank, and name it `Mailit Service`. Copy the **Application (client) ID** and **Directory (tenant) ID**. Create a **Client Secret** and copy its value.

Step 3: Add application permissions

Go to **API permissions** → **Microsoft Graph** → **Application permissions**. Add `Mail.Read`, `Mail.ReadWrite`, and `Mail.Send`. Click **Grant admin consent**.

Step 4: Configure OAuth in Mailit

Open `Simple Receiving.fmp12`, select **graphRest**, and choose **OAuth**.

- Select the `Microsoft Graph (Application)` preset.
- Enter the Client ID and Client Secret Value.
- On the **Additional Settings** tab, replace `{tenant}` with your Directory (tenant) ID.
- Enter the target mailbox email in **User Name**.

Step 5: Test receiving and sending

Ensure the **Mailit Assets** folder is installed. No browser login is required. Click **Test Connection**, then **Get Mail**. Use the same preset for sending.

3 Settings Templates in Mailit Examples

> ***Where the templates are located.*** *OAuth settings templates (presets)* ***are not part of the Mailit plug-in****. They are implemented as* ***scripts in the examples*** *supplied in the archive with the plug-in — primarily in* `Simple Receiving.fmp12`*.*

3.1 How Templates Work

The `Simple Receiving.fmp12` example contains scripts that automatically configure the connection:

- `Set authentication method` — determines the OAuth provider and calls the corresponding preset.
- `Set OAuth Presset` — calls the script by provider name (`Accounts::OAuth Provider`).
- `Set Server Port` — automatically sets the port based on the selected protocol and the `SSL` parameter.
- Individual OAuth preset scripts, one per template.

When you select a provider in the **OAuth Provider** list, the script fills in: **Server**, **SSL**, **OAuth Authorization URL**, **OAuth Token URL**, **OAuth Scope**, **OAuth Add Secret To Request**.

You do not need to enter these values manually if you are working with the example. All you need to do is select the correct **OAuth Provider**, specify the **OAuth Client ID**, specify the **OAuth Secret** (if required), and set the **User Name**.

3.2 Available OAuth Presets

There are seven templates — five for Microsoft and two for Google.

Microsoft Mail (Delegated) — classic protocols (POP / IMAP / SMTP), user login:

- Server: `outlook.office365.com`
- SSL: `On`
- Authorization URL: `https://login.microsoftonline.com/common/oauth2/v2.0/authorize`
- Token URL: `https://login.microsoftonline.com/common/oauth2/v2.0/token`
- **Scope:**

```
offline_access
https://outlook.office.com/POP.AccessAsUser.All
https://outlook.office.com/IMAP.AccessAsUser.All
```

  `https://outlook.office.com/SMTP.Send`
- OAuth Add Secret To Request: `Off`
- Client Secret: **not required**

Microsoft Mail (Application) — classic protocols (POP / IMAP / SMTP), no user login:

- Server: `outlook.office365.com`
- SSL: `On`
- Authorization URL: `https://login.microsoftonline.com/{tenant}/oauth2/v2.0/authorize`
- Token URL: `https://login.microsoftonline.com/{tenant}/oauth2/v2.0/token`
- Scope: `https://outlook.office365.com/.default`
- OAuth Add Secret To Request: `On`
- Client Secret: **required**

Microsoft Graph (Delegated) — REST API, user login:

- Server: `https://graph.microsoft.com/v1.0`
- SSL: `On`
- Authorization URL: `https://login.microsoftonline.com/common/oauth2/v2.0/authorize`
- Token URL: `https://login.microsoftonline.com/common/oauth2/v2.0/token`
- **Scope:**
  `offline_access`
  `https://graph.microsoft.com/Mail.ReadWrite`
  `https://graph.microsoft.com/Mail.Send`
  `https://graph.microsoft.com/User.Read`
- OAuth Add Secret To Request: `Off`
- Client Secret: **not required**

Microsoft Graph (Shared Mailbox) — REST API, shared mailbox, user login:

- Server: `https://graph.microsoft.com/v1.0`
- SSL: `On`
- Authorization URL: `https://login.microsoftonline.com/common/oauth2/v2.0/authorize`
- Token URL: `https://login.microsoftonline.com/common/oauth2/v2.0/token`
- **Scope:**
  `offline_access`
  `https://graph.microsoft.com/Mail.Read.Shared`
  `https://graph.microsoft.com/Mail.ReadWrite.Shared`
  `https://graph.microsoft.com/Mail.Send.Shared`
- OAuth Add Secret To Request: `Off`
- Client Secret: **not required**

Microsoft Graph (Application) — REST API, no user login:

- Server: `https://graph.microsoft.com/v1.0`
- SSL: `On`
- Authorization URL: `https://login.microsoftonline.com/{tenant}/oauth2/v2.0/authorize`
- Token URL: `https://login.microsoftonline.com/{tenant}/oauth2/v2.0/token`
- Scope: `https://graph.microsoft.com/.default`
- OAuth Add Secret To Request: `On`
- Client Secret: **required**

Google Mail (Delegated) — classic protocols (POP / IMAP / SMTP), personal accounts:

- Server: `imap.gmail.com`
- SSL: `On`
- Authorization URL: `https://accounts.google.com/o/oauth2/v2/auth`
- Token URL: `https://oauth2.googleapis.com/token`
- Scope: `https://mail.google.com/`

- OAuth Add Secret To Request: `On`
- Client Secret: **required**

Google API (Delegated) — Gmail REST API, personal accounts:

- Server: `https://gmail.googleapis.com`
- Port: `443`
- SSL: `On`
- Authorization URL: `https://accounts.google.com/o/oauth2/v2/auth`
- Token URL: `https://oauth2.googleapis.com/token`
- Scope: `https://mail.google.com/`
- OAuth Add Secret To Request: `On`
- Client Secret: **required**

> ***When Client Secret is required.*** *Microsoft Delegated — not required. Microsoft Application — required. Google — required.*

> ***About*** *`{tenant}`**.** Application templates use `{tenant}` as a placeholder. Replace it manually with your* ***Directory (tenant) ID****. Client Credentials flow does not work with `common`.*

> ***About*** *`common`**.** Delegated templates use `common`, which works for both personal and corporate accounts. If your administrator requires an explicit tenant, replace `common` with your tenant ID.*

4 Scenarios and Transports — Reference

4.1 Scenario 1: Delegated + Authorization Code

Permission type	Delegated permissions
OAuth flow	Authorization Code
Web authentication	Yes, user logs in via browser once
Refresh token	Yes, used for automatic refresh
Admin consent	Not required for personal, may be required for corporate

The application acts on behalf of the signed-in user. The user completes web authorization once, after which Mailit uses the refresh token to automatically refresh access.

4.2 Scenario 2: Application + Client Credentials

Permission type	Application permissions
OAuth flow	Client Credentials
Web authentication	No, there is no user login
Refresh token	No, the token is requested anew
Admin consent	Required

The application acts on its own behalf. The user does not log in via browser.

4.3 Which Transport to Choose

Both transports use the same IMAP functions of the plug-in and provide the same set of operations.

- **Choose IMAP / POP / SMTP if:** you prefer classic mail servers, your infrastructure is already set up for them, or you need to specify the server and port explicitly.
- **Choose Microsoft Graph if:** your administrator requires it, or you want a simpler setup for shared mailboxes.

4.4 Differences Between Transports and Scenarios

For each preset, this section shows the scenario, transport, where to add permissions, permission type, permissions, and scope.

`Microsoft Mail (Delegated)`

- **Scenario:** Delegated + Authorization Code
- **Transport:** POP / IMAP / SMTP
- **Where to add permissions:** API permissions → Microsoft Graph
- **Permission type:** Delegated
- **Permissions:** `IMAP.AccessAsUser.All`, `POP.AccessAsUser.All`, `SMTP.Send`
- **Scope:**

```
offline_access
https://outlook.office.com/IMAP.AccessAsUser.All
https://outlook.office.com/POP.AccessAsUser.All
https://outlook.office.com/SMTP.Send
```

`Microsoft Graph (Delegated)`

- **Scenario:** Delegated + Authorization Code
- **Transport:** Microsoft Graph
- **Where to add permissions:** API permissions → Microsoft Graph
- **Permission type:** Delegated
- **Permissions:** `Mail.ReadWrite`, `Mail.Send`, `User.Read`, `offline_access`
- **Scope:**

```
offline_access
https://graph.microsoft.com/Mail.ReadWrite
https://graph.microsoft.com/Mail.Send
https://graph.microsoft.com/User.Read
```

`Microsoft Graph (Shared Mailbox)`

- **Scenario:** Delegated + Authorization Code
- **Transport:** Microsoft Graph, shared mailbox
- **Where to add permissions:** API permissions → Microsoft Graph
- **Permission type:** Delegated
- **Permissions:** `Mail.Read.Shared`, `Mail.ReadWrite.Shared`, `Mail.Send.Shared`, `offline_access`
- **Scope:**

```
offline_access
https://graph.microsoft.com/Mail.Read.Shared
https://graph.microsoft.com/Mail.ReadWrite.Shared
https://graph.microsoft.com/Mail.Send.Shared
```

`Microsoft Mail (Application)`

- **Scenario:** Application + Client Credentials
- **Transport:** POP / IMAP / SMTP
- **Where to add permissions:** APIs my organization uses → Office 365 Exchange Online
- **Permission type:** Application
- **Permissions:** `IMAP.AccessAsApp`, `POP.AccessAsApp`, `SMTP.SendAsApp`
- **Scope:** `https://outlook.office365.com/.default`

`Microsoft Graph (Application)`

- **Scenario:** Application + Client Credentials
- **Transport:** Microsoft Graph
- **Where to add permissions:** API permissions → Microsoft Graph
- **Permission type:** Application
- **Permissions:** `Mail.Read`, `Mail.ReadWrite`, `Mail.Send`
- **Scope:** `https://graph.microsoft.com/.default`

5 Account Types — Reference

Account type	Description	Who registers the app
Personal	`outlook.com`, `hotmail.com`, `live.com`. No Microsoft 365 subscription.	You, under your personal Microsoft account.
Corporate	Microsoft Entra account with Microsoft 365 subscription.	Usually your Microsoft 365 administrator.
Shared mailbox	Corporate-only. Requires Exchange Online rights for users.	Administrator registers the app.
Application	No user. Requires Application permissions and admin consent.	Administrator only.

6 Application Registration — Reference

The `Simple Receiving.fmp12` example and this guide assume:

- **Personal accounts and delegated flows:** application registered with type **Public client/native (mobile & desktop)**, Redirect URI `http://localhost:1234/oauth/`. Client Secret is not required.
- **Application flows:** application registered as **Confidential client** (no Redirect URI needed). Client Secret is required.

7 Permissions — Reference

7.1 Where to Add Permissions

What you configure	Where to add	Permission type
Microsoft Graph, delegated	API permissions → Microsoft Graph	Delegated
Microsoft Graph, application	API permissions → Microsoft Graph	Application
IMAP / POP / SMTP, delegated	API permissions → Microsoft Graph	Delegated
IMAP / POP / SMTP, application	APIs my organization uses → Office 365 Exchange Online	Application

7.2 Common Permissions

- **Microsoft Graph — Delegated:** `offline_access`, `Mail.ReadWrite`, `Mail.Send`, `User.Read`.
- **Microsoft Graph — Delegated, shared mailbox:** `offline_access`, `https://graph.microsoft.com/Mail.Read.Shared`, `https://graph.microsoft.com/Mail.ReadWrite.Shared`, `https://graph.microsoft.com/Mail.Send.Shared`.
- **Microsoft Graph — Application:** `Mail.Read`, `Mail.ReadWrite`, `Mail.Send`.
- **IMAP / POP / SMTP — Delegated:** `IMAP.AccessAsUser.All`, `POP.AccessAsUser.All`, `SMTP.Send`.
- **IMAP / POP / SMTP — Application:** `IMAP.AccessAsApp`, `POP.AccessAsApp`, `SMTP.SendAsApp`.

7.3 Admin Consent

Account / scenario	Admin consent
Personal account, delegated	Not required
Corporate account, delegated	Recommended (once per organization)
Corporate account, application	**Required**

8 Mailit Connection Fields — Reference

Field	Description
OAuth Provider	Select the preset.
OAuth Client ID	Application (client) ID.
OAuth Secret	Client Secret Value. Required only for Application and Google templates.
User Name	Email of the mailbox. For shared mailbox — email of the shared mailbox.
Server	Filled in by the preset.
Server Port	Set automatically by the `Set Server Port` script.
SSL	Filled in by the preset.
OAuth Authorization URL	Filled in by the preset.
OAuth Token URL	Filled in by the preset.
OAuth Scope	Filled in by the preset.
OAuth Add Secret To Request	`Off` for Microsoft Delegated; `On` for Application and Google.
OAuth Refresh Token	Filled in automatically after authorization (Scenario 1 only).
OAuth Token Expires In	Filled in automatically after authorization (Scenario 1 only).

9 Server Addresses and Ports — Reference

9.1 Server Addresses

Transport	Server
Microsoft IMAP / POP / SMTP	`outlook.office365.com`
Microsoft Graph	`https://graph.microsoft.com/v1.0`
Google IMAP	`imap.gmail.com`
Google POP	`pop.gmail.com`
Google SMTP	`smtp.gmail.com`
Gmail API	`https://gmail.googleapis.com`

9.2 Ports

Protocol	SSL	Port
IMAP	`On`	993
IMAP	`Off`	143
POP	`On`	995
POP	`Off`	110
SMTP	`On`	465
SMTP	`Off`	587 (STARTTLS)

These are recommendations, not a rigid standard. *If the server requires a different port, specify it manually in the* ***Server Port*** *field.*

9.3 Local Web Server and Mailit Assets

The Mailit plug-in includes a built-in simple static web server. It is used for two main purposes:

- **OAuth Authorization:** Capturing the redirect from the provider after a user signs in (using `index.html`).
- **HTML Editors:** Saving and loading HTML message bodies and images when using the Summernote and Automizy editors in the FileMaker Web Viewer.

> ***Protocol limitation.*** *The local web server only supports the **HTTP** protocol. It does not support HTTPS. All local addresses must start with `http://localhost`.*

9.3.1 Deploying Mailit Assets

To start the server, the **Mailit Assets** folder must be deployed next to the plug-in file.

- It is unpacked **automatically** when you install the plug-in via `Install Plug-in.fmp12`.
- If the folder is missing, you can deploy it separately by opening `Install Plug-in.fmp12` and clicking the **Install Assets** button.

Typical plug-in installation paths:

- **macOS:** `/Users/[USER_NAME]/Library/ApplicationSupport/FileMaker/FileMakerPro/[FM_VERSION]/Extensions`
- **Windows:** `C:\Users\[USER_NAME]\AppData\Local\FileMaker\FileMakerPro\[FM_VERSION]\Extensions`

9.3.2 Available URLs

By default, the server runs at `http://localhost:1234`. The following endpoints are available:

- `http://localhost:1234/` — Serves static files from the `../Mailit Assets/` directory. Only GET requests are supported.
- `http://localhost:1234/oauth/index.html` — Used for OAuth authentication redirects.
- `http://localhost:1234/editor/upload` — Used by HTML editors to upload images. Only POST requests are supported. Saves files to the `../Mailit Assets/editor/upload/` directory.
- `http://localhost:1234/editor/load` — Loads field content into the editor. Only POST requests are supported. Requires a `fieldName` parameter (e.g., `fieldName=Messages::Body`).
- `http://localhost:1234/editor/save` — Saves editor content back to FileMaker. Only POST requests are supported. Requires `scriptFileName`, `scriptName`, and `scriptParameter` (which must be encoded using `encodeURIComponent()`).

> *The local web server only supports the HTTP protocol and does not execute server-side scripts like PHP or Python. It is designed solely for static files and editor communication.*

9.3.3 Plugin Functions for the Local Server

The FileMaker scripts use the following plug-in functions to manage the local server:

- `Emai_LocalWebServerStart` — Starts the server. Parameters: `host` (default: "localhost"), `port` (default: "1234"), `path` (optional path to Mailit Assets).
- `Emai_LocalWebServerStop` — Stops the server.
- `Emai_LocalWebServerStatus` — Returns the current status and settings of the server.

If the local server cannot start, check that the plug-in has read/write permissions to the local drive, and that the port (1234) is not being used by another application.

10 Authorization Methods — Reference

10.1 Browser Login (Scenario 1)

Two options:

- **Open OAuth2 Window in FileMaker** — WebView inside FileMaker. **Get Code** button at the bottom.
- **Open OAuth2 in External browser** — default browser. No **Get Code** button. After successful authentication, you will see the message: *"Token received and inserted into the password field, you can now close the current browser tab."*

10.2 No Login (Scenario 2)

No browser login required. Mailit uses client credentials to obtain an access token directly.

11 Shared Mailboxes — Reference

11.1 Requirements

- Corporate Microsoft 365 account — personal accounts are not supported.
- Exchange Online permissions on the shared mailbox: **Read and manage**, **Send as**, or **Send on behalf of**.
- Admin consent for the requested permissions.

11.2 Configuring Shared Mailbox Permissions

Method 1: Web interface (recommended). Microsoft 365 Admin Center `https://admin.cloud.microsoft` → Teams & groups → Shared mailboxes → MailboxDetails → Manage mailbox permissions.

```
# Grant Read and Manage (Full Access)
Add-MailboxPermission -Identity "shared@company.com" -User "user@company.com
    " -AccessRights FullAccess -AutoMapping:$false

# Grant Send As
Add-RecipientPermission -Identity "shared@company.com" -Trustee "
    user@company.com" -AccessRights SendAs -Confirm:$false
```

11.3 Choosing a Transport for a Shared Mailbox

- **Microsoft Graph** — simpler setup. Does not require assigning a primary SMTP address.
- **IMAP** / **POP** / **SMTP** — requires assigning a primary SMTP address.

Therefore, **Microsoft Graph is recommended** for shared mailboxes.

12 Token Lifecycle and Automatic Refresh

12.1 Scenario 1 — Refresh Token

Microsoft returns **Refresh Token** and **Token Expires In** alongside the access token. Both can be viewed on the **OAuth 2.0 Additional Settings** tab.

*In the plug-in examples, the **Refresh Token** field is displayed in clear text for debugging. In production, store it in a safe place.*

Automatic refresh. The `Connect to the mail server` script checks `Accounts::OAuth Token Expires In`. If the token has expired, it sends a POST to `Accounts::OAuth Token URL` with `client_id`, `client_secret` (if enabled), `refresh_token`, and `grant_type=refresh_token`. The script updates `Accounts::Password`, `Accounts::OAuth Refresh Token`, and `Accounts::OAuth Token Expires In`, then retries the connection.

If refresh fails, re-authorize via the browser.

12.2 Scenario 2 — Client Credentials

No refresh token. The application requests a new access token each time, using `grant_type=client_credentials`.

13 Working in a Multi-User Environment

13.1 What Is Not Recommended

Do not use a single shared mailbox account for multiple users if each goes through interactive OAuth authorization.

1. **Tokens overwrite each other.** Access token, refresh token, and expiration are stored per account record.
2. **Refresh token may become invalid.** Microsoft does not guarantee parallel use of the same refresh token.
3. **No audit trail.**
4. **Security risk.**
5. **Microsoft limits.**

13.2 Recommended Practices

Option A. Each user has their own mailbox account. Each employee authorizes under their own account.

Option B. Shared mailbox + personal authorization. Each user authorizes under their own account and is granted Shared permissions on the mailbox.

Option C. Shared mailbox + application. For server-side processing without interactive users, use Application permissions with admin consent.

14 Service Limits and Scaling

Microsoft applies quota limits to the combination of **"Application ID (App ID) + Mailbox"**. This means the quota is calculated per mailbox, per application. If one user exhausts the quota for their mailbox, it does not affect the quota for another user's mailbox, even if they use the same application.

14.1 Microsoft Graph Limits (Outlook Mail)

Limit	Value	Scope
Request count	10,000 per 10 minutes	One App ID per one mailbox
Concurrent requests	4	One App ID per one mailbox
Upload volume	150 MB per 5 minutes	One App ID per one mailbox

14.2 Exchange Online Limits

Limit	Value
Send rate	30 messages per minute
Recipients per day	10,000

14.3 Demo Applications

> ***Do not use a shared demo application in production.*** *A demo application with a pre-filled Client ID is intended for quick testing only. Quotas will be exhausted faster, Microsoft may flag the application, and you will have no control over who uses it.*

14.4 Scaling Across Multiple Applications

Multiple applications may serve different user groups.

> ***Important.*** *Microsoft has tightened its policy. Creating several applications to perform* ***the same task*** *(to circumvent throttling) is a* ***violation of the API Terms of Use****. Splitting applications across user groups or departments is acceptable. It must not be used to bypass limits.*

15 Troubleshooting

15.1 Diagnostic Information

Collect the following before contacting support:

- Mailit build and FileMaker version.
- Operating system.
- The step where the error occurred.
- Exact error text.
- Contents of the **Last Session Transcript**.
- The plug-in log `Mailit_FM[FM_VERSION].log`.

> *Remove authorization codes, tokens, secrets, and message contents before sending diagnostics.*

15.2 Local Web Server Errors (-30 / -61)

If the function `Emai_LocalWebServerStart` fails with error **-30** (incorrect option) or **-61** (file not found), it almost always means the local web server cannot start.

The most common causes are:

1. The **Mailit Assets** folder is missing next to the plug-in file.
2. FileMaker does not have read/write permissions to the **Mailit Assets** folder. This is especially common on Windows 11 if the folder was copied manually.
3. The port (default `1234`) is blocked by a firewall/antivirus or is already in use by another application.

How to fix:

1. Verify that the **Mailit Assets** folder exists in the FileMaker Extensions directory (see section 9.3 for paths).
2. If the folder exists, verify that the user running FileMaker has **read and write** permissions to this folder.
3. Open `Install Plug-in.fmp12`, go to the **Assets** tab, and click **Install Assets** to unpack the folder again automatically.
4. If the problem persists, change the port in the example's settings (**OAuth 2.0 Additional Settings** tab, **Local WebServer Port** field) to a different value, such as `8080` or `12345`.

> ***Diagnostics.*** *If the server still does not start, enable the plug-in log (see below), repeat the steps, and open the log file. Look for lines like "No additional plug-in files were found: [path]". This will show you the exact path where the plug-in is looking for the Assets folder.*

15.3 Where the Plug-in Log Is Located

The plug-in log `Mailit_FM[FM_VERSION].log` is located in the user's Documents folder.

- **macOS:** `/Users/[USER_NAME]/Documents/Mailit_FM[FM_VERSION].log`
- **Windows:** `C:\Users\[USER_NAME]\Documents\Mailit_FM[FM_VERSION].log`

15.4 Enabling the Plug-in Log

If the log file is disabled, enable it using the `Install Plug-in.fmp12` database:

1. Open `Install Plug-in.fmp12`.
2. Go to the **Log File** tab.
3. In the **Log file creation** field, select **Append**.
4. Click the **Apply settings in FM** button.
5. Restart FileMaker and repeat the steps that led to the error.

15.5 Common OAuth Errors

- `AADSTS50011` — Redirect URI mismatch. Check that it is exactly `http://localhost:1234/oauth/` and has a trailing slash.
- `invalid_scope` — Scopes in Mailit do not match the permissions in Entra/Google Cloud. Check against section 4.4.
- `invalid_grant` — Refresh token revoked or expired. Re-authorization is required.
- `invalid_client` — Client Secret issue. Check the secret value, `OAuth Add Secret To Request`, and secret expiration.
- `localhost` unreachable — Check that Mailit Assets are installed, the port is free, and FileMaker is running.
- `unauthorized_client` — Check that the app is registered as Single tenant and admin consent has been granted.