

Qeedji

User manual 001B

briva_calendar-ews 1.11.10

Legal notices

1.11.10 (001B_en)

© 2020 Qeedji

Rights and Responsibilities

All rights reserved. No part of this manual may be reproduced in any form or by any means whatsoever without the written permission of the publisher. The products and services mentioned in this document may be trademarks and/or trademarks of their respective owners. The publisher and the author do not pretend to these brands.

Although every precaution has been taken in the preparation of this document, the publisher and the author assumes no responsibility for errors or omissions, or for damages resulting from the use of the information contained in this document or the use of programs and source code who can accompany it. Under no circumstances can the publisher and the author be held responsible for any loss of profits or any other commercial prejudice caused or that would have been caused directly or indirectly by this document.

Product information

The conception and specifications of the product may change without prior notice, and this applies to hardware, embedded software and this guide. Consumable items accessories may slightly differ than herein described as Qeedji is depending on the evolutions of its suppliers. This document contains confidential information; it can't be disclosed to any third parties without prior written authorization of Qeedji.

Safety instructions

Please read carefully the following instructions before switching the product on: - WARNING! Correct fitting and installation is of the utmost importance. Incorrect fitting and/or installation may result in personal injury or loss. Qeedji disclaims all liability, of whatever kind, if the product is assembled, fitted and/or installed in an incorrect manner. - Do not use the product near a water supply. - Do not pour anything on the product, like flammable liquids or material. - Do not expose the product to direct sun, near a heating source or a dust nor vibrations. - Do not obstruct holes, to be sure that air flows freely around the product. - Switch off the product during a storm. - Do not open the product in any circumstances.

Guarantee terms

Qeedji products are eligible for a warranty to cover genuine manufacturing defect for 3 years. Product failure occurring as the result of factors that do not constitute genuine manufacturing defect are not covered under the terms of the warranty and any repairs of this nature would be chargeable. For example: Inappropriate maintenance action, a non-authorized modification, a not specified environment utilization (see 'Safety instructions'), or if the product has been damaged after an impact, a fall, a bad manipulation or a storm consequence, an insufficient protection against heat, moisture or frost. This warranty is not transferrable. In addition, any repairs carried out by non-authorized personnel will invalidate the warranty.

WEEE Directive



This symbol means that your end of life equipment must not be disposed of with household waste but must be deposited at a collection point for waste electrical and electronic equipment or to your reseller. This will benefit the environment. In this context, a system for collecting and recycling has been implemented by the European Union

Table of content

Part I

Introduction	1.1
Microsoft Exchange and Office 365	1.2
Briva-calendar server installation	1.3
EWS calendar connector installation	1.4
EWS calendar connector configuration	1.5
EWS calendar connector's test URL	1.6

Part II

Contacts	2.1
----------------	-----

Part III

Appendix: Resource email and delegate email accounts creation with Powershell	3.1
Appendix: Azure AD application creation with Azure AD portal	3.2
Appendix: Azure AD Application creation with PowerShell module	3.3
Appendix: .xml calendar format	3.4
Appendix: .ics calendar format	3.5
Appendix: Alarm and cache persistency	3.6
Appendix: XML entities	3.7
Appendix: Turn on the basic authentication (obsolete)	3.8

1.1 Introduction

The Briva-calendar EWS solution is allowing to connect periodically to your Microsoft Exchange calendar or to your Office 365 calendar to gather the events of the day for one or several resources calendars and output them into:

- an *.ics* calendar, compliant with *Internet Calendar Scheduling* RFC,
- a *.xml* calendar, propriety format. For further information, refer to the chapter § [Appendix: .xml calendar format](#).

This document explains how to install and configure the EWS-calendar connector on the Briva-calendar server so that it can connect:

- to your Microsoft Exchange Server calendar and
- to your Office 365 calendar.

Briva-calendar server compatibility

The EWS-calendar connector must be installed on a Briva-calendar server 2.10.10.

1.2 Microsoft Exchange and Office 365

Calendar compatibility

- Microsoft Exchange Server 2007 to 2019
- Office 365

For Microsoft Exchange Server :

- the *Web services* must be enabled,
- the Microsoft Exchange Server must be on time, configured with a suitable time zone.

 *The Web services must be activated on Microsoft Exchange Server with the same suitable authentication mode used by the EWS connector.*

Authentication compatibility

The EWS-calendar connector supports these authentications:

- Basic authentication,
- Azure AD authentication (OAuth2), called also *modern* authentication.

Both authentications Basic and Azure AD are still supported in Office 365 :

- For any new Office 365 users, the default authentication activated is Azure AD ,
- For existing Office 365 users, the authentication activated may be either Azure AD OR Basic .

Resources email accounts and delegate email account

To work with Briva-calendar EWS , it is required to have one resource email account for each Microsoft Exchange resource (or per Office 365 resource). Each resource calendar must be controlled by a same *delegate* email account user.

If not already done, you must create and configure the resources email accounts and the *delegate* email account with `powershell` commands. To see some *Powershell* commands examples, refer to the chapter § [Appendix: Resource email and delegate email accounts creation with Powershell](#).

1.3 Briva-calendar server installation

If not already done, download the [Briva-calendar server 2.10.10](#) then execute *Innes Briva Server Setup V2.10.10.exe* on a MS-Windows Server computer to install it. The MS-Windows Server computer must be available on your local network.

The Briva-calendar server is by default installed in this directory:

- C:\Program Files (x86)\Innes Briva

The user data for the Briva-calendar server are stored by default in this directory:

- C:\Users\Public\Documents\Innes Briva

Briva-calendar server (C:\Program Files (x86)\Innes Briva\Server\bin\httpd.exe) starts automatically on port 80 when the MS-Windows server starts.

The start and stop shortcuts, available on MS-Windows App panel, allows to start or stop the Briva-calendar server .

List of calendar connectors installed

Briva-calendar server can support several types of calendar connector at a time.

When the Briva-calendar server is started, the Briva-calendar server Web configuration page is available with this URL:

- http://<myBrivaCalendarServer_login>:<myBrivaCalendarServer_password>:<myBrivaCalendarServer_port>@myBrivaCalendarServer_domain_or_IPV4_addr/.configuration/

For example

- <http://admin:admin@192.168.2.69/.configuration/>

☞ The default credential to connect as administrator is admin / admin .

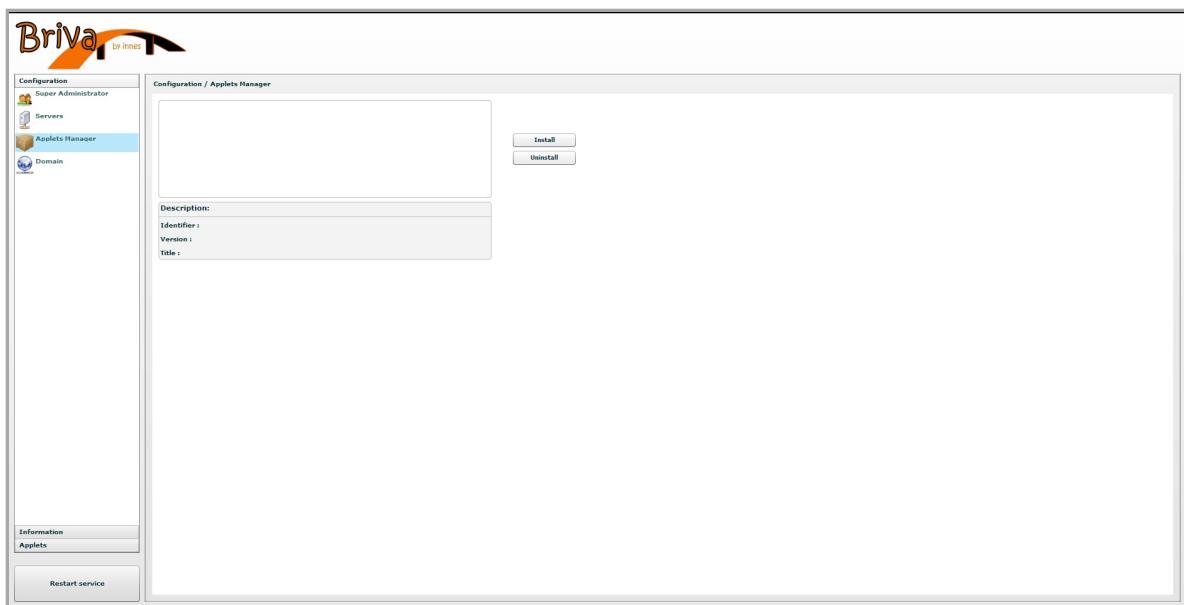
☞ It is advised to use DNS to access to your Briva-calendar server .

☞ To open the Briva-calendar server Web configure page, the browser must support the flash technology. If you have not, contact support@innes.pro who should help to find one.

Click on the Applets Manager tab. The Configuration / Applets Manager pane, displayed on the right, lists all the calendar connectors installed.

To know the version of a calendar connector, click on the appropriate calendar connector installed and check the Description pane just below displaying information about the connector:

- identifier ,
- version ,
- title .



Click on the Domain tab. Click on New to create a domain (for example: myDomain).

1.4 EWS calendar connector installation

Open the Briva-calendar server Web configure page, available with this URL:

- http://<myBrivaCalendarServer_login>:<myBrivaCalendarServer_password>:<myBrivaCalendarServer_port>@myBrivaCalendarServer_domain_or_IPV4_addr/.configuration/

For example

- <http://admin:admin@192.168.2.69/.configuration/>

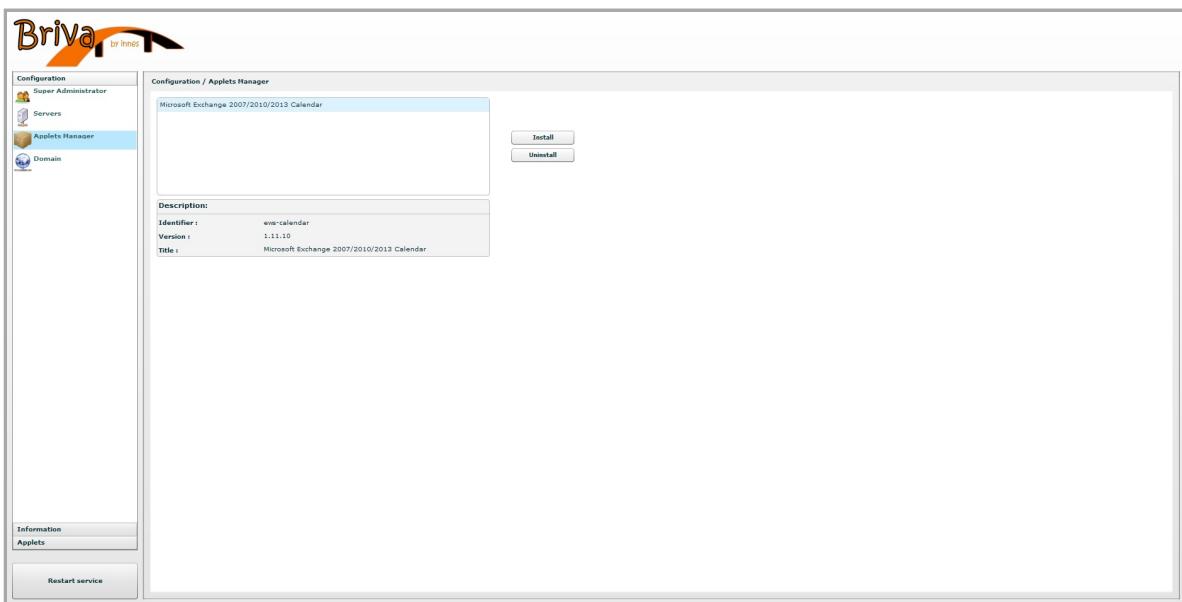
Click on the Applets Manager tab. The Configuration / Applets Manager pane, displayed on the right, lists all the calendar connectors installed.

If already installed, select the Microsoft Exchange 2007/2010/2013 Calendar connector and check that the version is 1.11.10 (or above). Otherwise:

- download the latest [EWS connector \(.saz\)](#),
- click on the Install button ,
- select the `ews-calendar-1.11.10.saz` and click on the Open button.

Wait a couple of time.

Click on the Microsoft Exchange 2007/2010/2013 Calendar connector installed and in the Description pane just below, check that the version is 1.11.10.



When successfully installed:

- these files are available in this directory `C:\Users\Public\Documents\Innes Briva\Server\.accounts\<mydomain>\.applets\ews-calendar\` :
 - `configuration.xml`,
 - `2ical.php`,
 - `2xml-daycalendar.php`.
- these files are available in this directory `C:\Users\Public\Documents\Innes Briva\Server\.shared\.applets\ews-calendar\` :
 - `2ical.php`,
 - `2xml-daycalendar.php`.

1.5 EWS calendar connector configuration

To configure the EWS calendar connector, edit the C:\Users\Public\Documents\Innes Briva\Server\accounts\<mydomain>\applets\ews-calendar\configuration.xml file to fill the required information to connect to your Microsoft Exchange Server or to your Office 365.

The configuration file template is auto-explained and contains different configuration examples.

- ⚠ Do modify the configuration.xml file by matching the XML syntax.
- ⚠ The file editor, allowing to make modifications in the configuration.xml file, must keep the characters encoded in UTF-8.

Modification of the configuration.xml to connect to your Microsoft Exchange Server or to your Office 365

The upper part of the configuration.xml (tag <scc:server>) allows to configure the connexion to your Microsoft Exchange Server or to your Office 365.

The connector is able to connect to several Microsoft Exchange Server machine or Office 365 accounts at a time with only one configuration.xml file. In this case, a unique serverId value must be created for each Microsoft Exchange Server or each Office 365 server.

In the configuration.xml, uncomment the block from to and fill with your data:

```
<scc:server>
...
</scc:server>
```

This is a serverId1 example with a Microsoft Exchange Server, available in your local network with the IPV4 address 192.168.1.100, supporting the Basic authentication:

```
<scc:server id="serverId1" type="ews">
  <scc:baseuri>http://192.168.1.100</scc:baseuri>
  <scc:label>myExchangeServer2016</scc:label>
  <scc:authentication type="http-authentication">
    <scc:scheme>basic</scc:scheme>
    <scc:credentials>
      <scc:username>myDelegate@exchange2016.contoso.pro</scc:username>
      <scc:password>myPassword</scc:password>
    </scc:credentials>
  </scc:authentication>
</scc:server>
```

This is a serverId2 example with Office 365 supporting the Basic authentication:

```
<scc:server id="serverId2" type="ews">
  <scc:baseuri>https://outlook.office365.com</scc:baseuri>
  <scc:label>MyOffice365</scc:label>
  <scc:authentication type="http-authentication">
    <scc:scheme>basic</scc:scheme>
    <scc:credentials>
      <scc:username>myDelegate@contoso.onmicrosoft.com</scc:username>
      <scc:password>myPassword</scc:password>
    </scc:credentials>
  </scc:authentication>
</scc:server>
```

This is a serverId3 example with Office 365 supporting the Azure AD authentication:

```
<scc:server id="serverId3" type="ews">
  <scc:baseuri>https://outlook.office365.com</scc:baseuri>
  <scc:label>MyOffice365</scc:label>
  <scc:authentication type="ms-azure-ad">
    <scc:scheme>ms-oauth-app</scc:scheme>
    <scc:credentials>
      <scc:username>myDelegate@contoso.onmicrosoft.com</scc:username>
      <scc:tenant-id>00000000-0000-0000-0000-000000000000</scc:tenant-id>
      <scc:client-id>00000000-0000-0000-0000-000000000000</scc:client-id>
      <scc:client-secret>mySecretxxxxxxxxyyyyzzzzz</scc:client-secret>
    </scc:credentials>
  </scc:authentication>
</scc:server>
```

Modification of the configuration.xml to create a calendarId supporting your resources calendars

A calendarId is the name of the the calendar you have to create to gather the event of one or several resources calendars. At least one calendarId must be created.

In the configuration.xml, several calendarIds can be created. All the calendarId created must have different values.

A calendar is created inside the tag with these tags:

- the value myCalendarId is a free text value,
- the value myServerId must match one of the serverId defined above.

```
<scc:calendars>
  <scc:calendar id="myCalendarId1" server="url(#myServerId1)">
  ...
  </scc:calendar>
  <scc:calendar id="myCalendarId2" server="url(#myServerId1)">
  ...
  </scc:calendar>
</scc:calendars>
```

This is an example of calendars tag with three calendarId:

- rooms12,
- roomsW1w2,
- roomsW3w4,

It is assumed in the example that:

- the resources account emails used inside tags are respectively available and properly configured in the Microsoft Exchange Server or Office 365 whose id (example url(#serverId1)) is specified just above,
- the resources calendar are shared properly with the same delegate account email with read/write access granted.

```

<scc:calendars>
  <scc:calendar id="rooms_1_2" server="url(#serverId1)">
    <param name="resource">room1@exchange2016.contoso.pro</param>
    <param name="resource">room2@exchange2016.contoso.pro</param>
    <param name="cachePersistence">300</param>
    <param name="startRelated">day-start</param>
    <param name="endRelated">day-end</param>
  </scc:calendar>
  <scc:calendar id="rooms_w1_w2" server="url(#serverId2)">
    <param name="resource" value="Room 1 (Custom label)">web_room1@contoso.onmicrosoft.com</param>
    <param name="resource">web_room2@contoso.onmicrosoft.com</param>
    <param name="cachePersistence">300</param>
    <param name="startRelated">day-start</param>
    <param name="endRelated">day-end</param>
  </scc:calendar>
  <scc:calendar id="rooms_w3_w4" server="url(#serverId3)">
    <param name="resource">web_room3@contoso.onmicrosoft.com</param>
    <param name="resource">web_room4@contoso.onmicrosoft.com</param>
    <param name="cachePersistence">300</param>
    <param name="startRelated">day-start</param>
    <param name="endRelated">day-end</param>
  </scc:calendar>
</scc:calendars>

```

FilterSensitivity parameter

The *filterSensitivity* parameter specifies which calendar items are filtered according to their sensitivity. The possible values are:

- *Personal*¹,
- *Confidential*¹,
- *Private*,
- *Normal*.

¹ Available on Microsoft Exchange Server only.

```

<scc:calendar id="rooms_1_2" server="url(#serverId)">
  <param name="filterSensitivity">Personal,Confidential,Private</param>
</scc:calendar>

```

 Microsoft Exchange Server does not manage properly the private *filterSensitivity* attribute when it is updated after meeting creation. To solve the issue, enter the meeting directly with the private option activated.

FilterImportance parameter

The *filterImportance* parameter specifies which calendar items are filtered according to their importance. The possible values are:

- *Low*¹,
- *Normal*,
- *High*¹.

```

<scc:calendar id="rooms_1_2" server="url(#serverId)">
  <param name="filterImportance">Normal,High</param>
</scc:calendar>

```

¹ Available on Microsoft Exchange Server only.

Room label rename

In the example, the resource name (ex: *web_room12345678*, name given by IT department) os overwritten by a friendly custom label (ex: *Room 115*).

```

<scc:calendar id="rooms_w1_w2" server="url(#serverId2)">
  <param name="resource" value="Room 115">web_room12345678@contoso.onmicrosoft.com</param>
  <param name="resource">web_room2@contoso.onmicrosoft.com</param>
  <param name="cachePersistence">300</param>
  <param name="startRelated">day-start</param>
  <param name="endRelated">day-end</param>
</scc:calendar>

```

Working range

To not support the events outside the working range, uncomment the block like explained.

```

<scc:generalsettings>
  <!-- Working day (Occurrence=1)-->
  <scc:workingday>
    <!-- define the start and end time slot of the working day -->
    <param name="dayStartTime">7:59</param>
    <param name="dayEndTime">19:01</param>
  </scc:workingday>
</scc:generalsettings>

```

1.6 EWS calendar connector's test URL

To test successfully the Briva-calendar EWS with your computer and get the event of the day:

- the resource email accounts must be created successfully in your calendar system,
- the delegate email account must be created successfully in your calendar system,
- the configuration.xml file of the ews-calendar connecotr must be properly configured:
 - calendar system datasource with appropriate:
 - calendar system URL
 - calendar system authentication type:
 - Basic :
 - delegate account email value,
 - password value,
 - Azure AD :
 - client secret value,
 - client ID value,
 - tenant ID value,
 - delegate account email value,
 - calendarId value, gathering the events of the day for or several resources calendars,
 - alarms .

Ensure that no network software product or network device prevent your computer to communicate with the Briva-calendar EWS .

.ics output format

This is the syntax for the test URL to get an .ics calendar file:

- `http://<BrivaCalendarServer_domain_or_ipv4_addr>/pluggnCast/.applets/.ews-calendar/2ical.php?calendarId=<calendarId>`

.xml output format

This is the syntax for the test URL to get an .xml calendar file:

- `http://<BrivaCalendarServer_domain_or_ipv4_addr>/pluggnCast/.applets/.ews-calendar/2xml-daycalendar.php?id=<calendarId>`

2.1 Contacts

For further information, please contact us by e-mail:

- **Technical support:** support@innes.pro,
- **Sales department:** sales@innes.pro.

Refer to the INNES Website for FAQ, application notes, and software downloads: <https://www.innes.pro/>

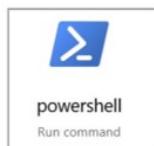
INNES SA
5A rue Pierre Joseph Colin
35700 RENNES

Tel: +33 (0)2 23 20 01 62
Fax: +33 (0)2 23 20 22 59

3.1 Appendix: Resource email and delegate email accounts creation with Powershell

Configuration using powerShell

On a MS-Windows computer, launch `powershell` with administrator rights.



!* *SSL is requested by the `powershell` client. If the SSL error is raised, unencrypted traffic is disabled in the client configuration. A temporary solution consists in disabling the SSL for the `powershell` session. In this case: type the following command lines.

```
cd WSMAN:\localhost\Client  
set-item .\allowunencrypted $true  
set-item .\trustedhosts IPAddressofyourpowershellclientcomputer
```

Execute Powershell commands for MS-Exchange online (o365)

On a MS-Windows computer, open `powershell` command with administrator rights and execute these commands:

```
Set-executionpolicy unrestricted  
$LiveCred = Get-Credential
```

Enter your MS-Exchange online (o365) login credentials then type:

```
$Session = New-PSSession -ConfigurationName Microsoft.Exchange -ConnectionUri  
https://ps.outlook.com/powershell/ -Credential $LiveCred -Authentication Basic -AllowRedirection  
Import-PSSession -Verbose $Session
```

Create a delegate account

For security reasons, the MS-Exchange administrator must avoid granting *read/write* access to all the users for all the resources.

!* One of the MS-Exchange recommendations is to remain compliant with the MS-Exchange workflow and be able to connect with a delegate user account, and not with a resource account. Consequently, the EWS calendar connector requires to use a delegate account having its own mailbox account. The login credentials (Id and/or password) of the delegate account are then used in the `configuration.xml` file of the EWS calendar connector. Read/write rights must be granted on the required resource calendar (and nothing else). It is important to give an appropriate specific name for this delegate account to be identified easily by all users. With MS-Exchange online (o365), no additional license is required for the delegate account.

The configuration of the MS-Exchange mainly consists in creating a delegate account which can read/write on the required room resources calendar. Once the delegate account is created and the delegation for all the resources is done, the rooms resources appear in the list of resources in the MS-Exchange web interface.

Create the delegate account used by EWS calendar connector:

- for MS-Exchange 2013-2016/2019 and MS-Exchange online (o365):

```
New-Mailbox -Alias Innes-delegate -Name "Innes-delegate" -FirstName "Innes" -LastName "Delegate"  
-DisplayName "Innes-Delegate" -MicrosoftOnlineServicesID "delegate@mydomain.onmicrosoft.com"  
-Password (ConvertTo-SecureString -String "1234abcd" -AsPlainText -Force) -ResetPasswordOnNextLogon $false
```

- For MS-Exchange 2007/2010:

```
New-Mailbox -Alias "Innes-delegate" -Name "Innes-delegate" -FirstName "Innes" -LastName "Delegate"  
-DisplayName "Innes-Delegate" -UserPrincipalName "Innes-Delegate@mycompany.com" -Password  
(ConvertTo-SecureString -String "1234abcd" -AsPlainText -Force) -ResetPasswordOnNextLogon $false
```

Resource creation: new mailbox for the resource

```
New-Mailbox -Name "Room ABC" -Room
```

Resource default configuration: auto-accept, organizer in subject, display subject, working hours

```
Set-CalendarProcessing -Identity "Room ABC" -AutomateProcessing AutoAccept  
-AddOrganizerToSubject $false -DeleteSubject $false  
-ScheduleOnlyDuringWorkHours $true
```

Resource calendar working hours

```
Set-MailboxCalendarConfiguration "Room ABC" -WorkingHoursStartTime 08:00:00  
-WorkingHoursEndTime 19:00:00 -Workdays Weekdays -WeekStartDay Monday  
-WorkingHoursTimeZone "Central Europe Standard Time"
```

Resource calendar access granting for the delegate account

- for MS-Exchange 2013-2016/2019 and MS-Exchange online (o365)

```
Add-MailboxPermission -Identity RoomABC@mydomain.onmicrosoft.com -User "Innes-Delegate"  
-AccessRights FullAccess -InheritanceType All -automapping $true
```

- for MS-Exchange 2007/2010

```
Add-MailboxPermission -Identity RoomABC@mydomain.com -User "Innes-Delegate"  
-AccessRights FullAccess -InheritanceType All -automapping $true
```

In this example, a delegate account `Innes-Delegate@mycompany.onmicrosoft.com` has been created using password `1234abcd`. The delegate account can handle the room `RoomABC@mycompany.onmicrosoft.com`.

!* Ensure under IIS that the MS-Exchange Web Services (used by ews-calendar) uses the same authentication mode: Basic, NTLM (Windows), or Digest.

Keep event's description in calendar

If the displaying of the event description is required, ensure that the event description is not deleted for the meetings in the resource mailboxes. To not remove description from the meetings for a given room:

- for MS-Exchange 2010/2013/2016/2019 and MS-Exchange online (o365)

```
Set-CalendarProcessing "Room 1" -DeleteComments $False
```

•

- for MS-Exchange 2007

```
Set-MailboxCalendarSettings "Room 1" -DeleteComments $False
```

Keep event's attachment in calendar

If the displaying of the event's attachment is required, ensure that the event's attachments are not removed automatically from the resource calendar:

- for MS-Exchange 2010/2013/2016/2019 and MS-Exchange online (o365)

```
Set-CalendarProcessing "Room 1" -DeleteAttachments $False
```

- for MS-Exchange 2007

```
Set-MailboxCalendarSettings "Room 1" -DeleteAttachments $False
```

 Only The .gif, .png and .jpg format are supported as event attachment.

Event autoaccept

When a meeting is created, it is stored both in the delegate calendar and in the resource calendar. The event booking must be in *AutoAccept* mode so that the event is automatically stored in the resource calendar without waiting confirmation. Check the *AutoProcessing* value by calling this Powershell command for resource:

```
Get-MailboxCalendarSettings "<Room_name>" | fl
```

Display resource event with only some privacy levels

To handle *private/confidential/personal* privacy levels, type this command for all your resources.

```
Set-CalendarProcessing "Room 2" -RemovePrivateProperty $False
```

3.2 Appendix: Azure AD application creation with Azure AD portal

You can create your Azure Active Directory (or AAD) application by following this Microsoft tutorial <https://docs.microsoft.com/en-us/graph/auth-register-app-v2>.

A procedure example is shown here after by connecting to the *Microsoft Azure* portal.

This procedure allows to generate you own ID and SECRET required in the App:

- Application (client) ID ,
- Directory (Tenant) ID ,
- Client secret .

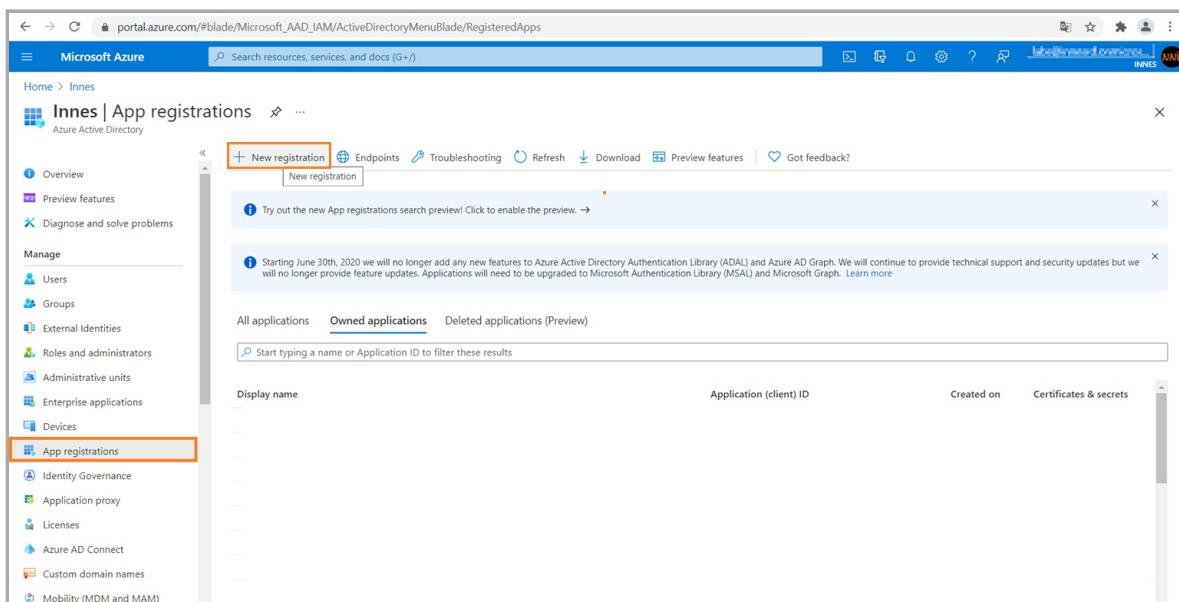
If you want to follow the PowerShell scripts procedure instead of following the procedure by connecting to the Azure AD portal, only PowerShell script for Azure Active Directory Application support 1.10.13 (and above) is supported. For further information, refer to the chapter § Appendix: Azure AD Application PowerShell module.

Connect on [Microsoft Azure portal: https://portal.azure.com/](https://portal.azure.com/) and sign in with your EWS administrator account login credentials.

Click on the left top menu and choose the *Azure Active directory* item.

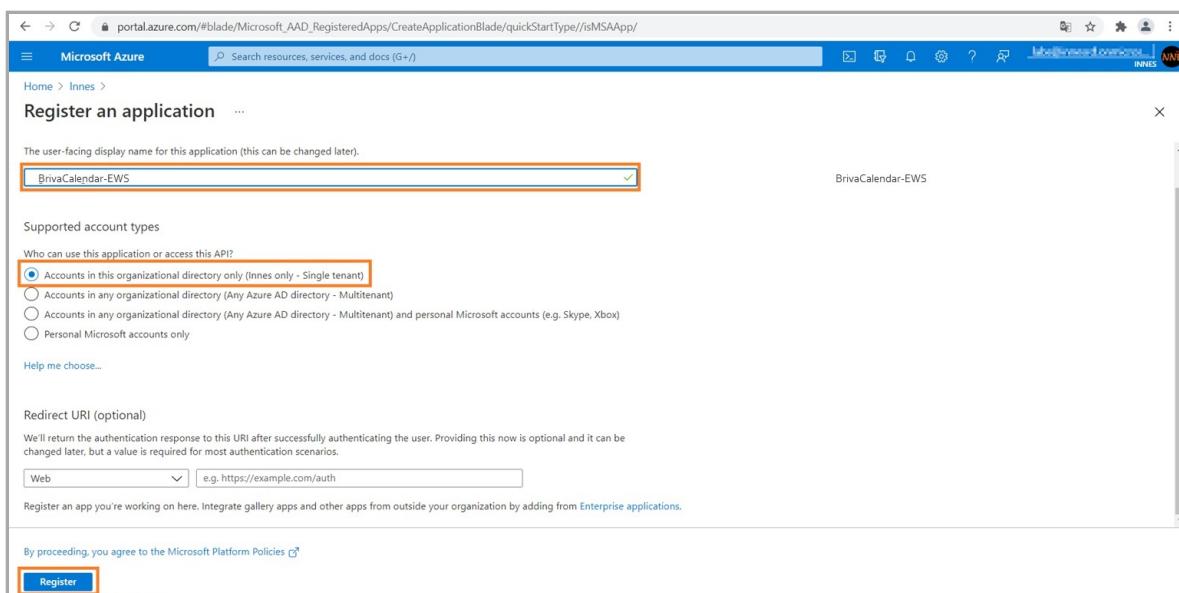
Application (client) ID and directory (Tenant) ID

On the App registrations menu, click on *New registration* https://portal.azure.com/#blade/Microsoft_AAD_IAM/ActiveDirectoryMenuBlade/RegisteredApps.

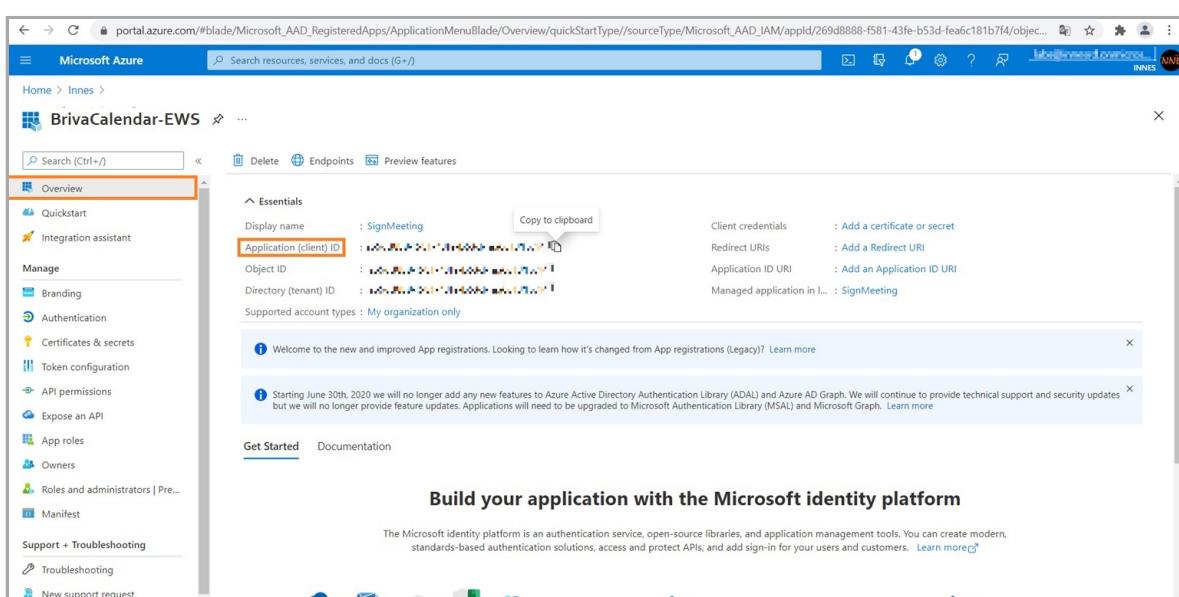


Enter an application name (e.g.: *BrivaCalendar-EWS*),

Select the appropriate Account in the organisation directory only (organisation only – Single tenant) radio button, and click on the Register button.



In the Overview menu, copy to clipboard the Application (client) ID value, the 1st value required in App configuration tab and store it preciously.

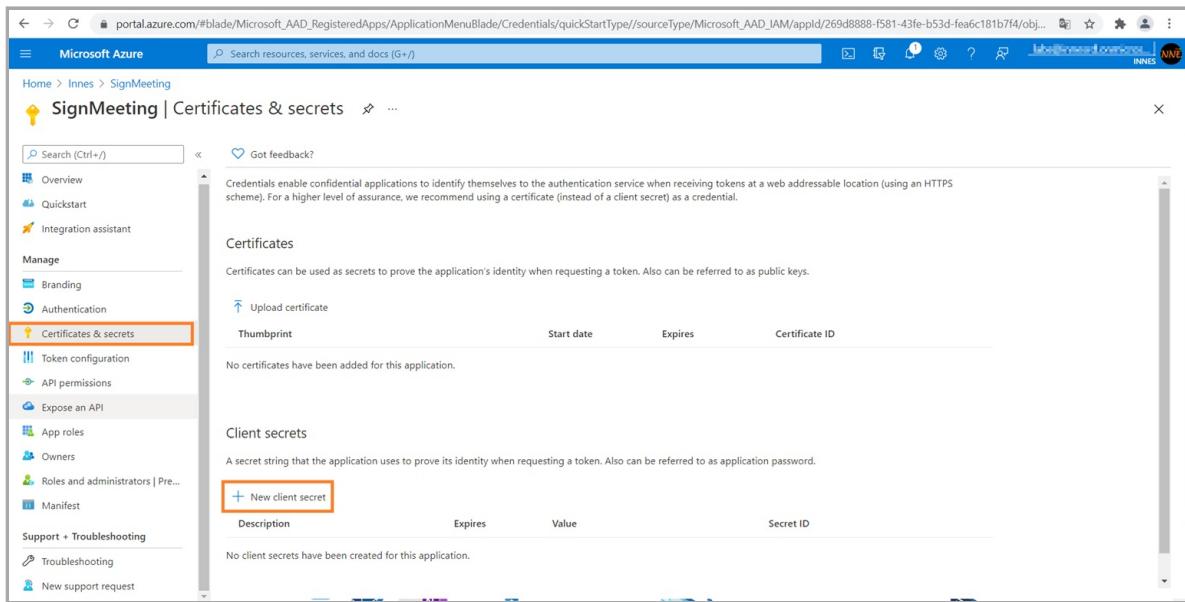


In the Overview menu, copy to clipboard the Directory (tenant) ID value, the 2nd value required in App configuration tab and store it preciously.

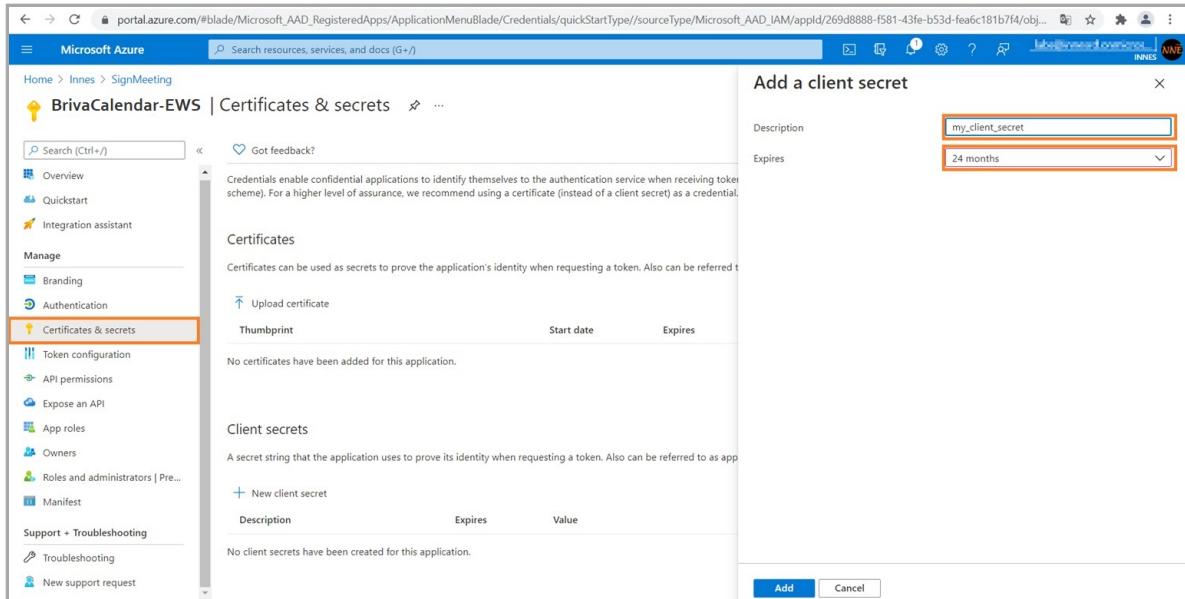
The screenshot shows the Microsoft Azure portal interface for managing app registrations. The left sidebar lists navigation options like Overview, Quickstart, Integration assistant, Manage (Branding, Authentication, Certificates & secrets, Token configuration, API permissions, Expose an API, App roles, Owners, Roles and administrators, Manifest), Support + Troubleshooting (New support request), and Troubleshooting. The main content area is titled 'Overview' and displays the 'Essentials' section for an app named 'SignMeeting'. It shows details such as Display name: SignMeeting, Application (client) ID: (redacted), Object ID: (redacted), Client credentials: Add a certificate or secret, Redirect URIs: Add a Redirect URI, Application ID URI: Add an Application ID URI, and Managed application in L... : SignMeeting. A 'Copy to clipboard' button is visible next to the Object ID. Below this, there are two informational cards: one about the new App registrations experience and another about the deprecation of ADAL and Graph starting June 30th, 2020. At the bottom, there are 'Get Started' and 'Documentation' links, and a decorative footer featuring various Microsoft icons.

Client secret

In the Certificates & secrets menu, click on the New client secret button.



Enter a name (e.g.: `my_secret_key`) and click on the Add button.



Copy into clip board the client secret value, the 3rd input for the App configuration tab and store it preciously.

⚠ Do it right now because the client secret value is not visible anymore as soon as you click on a new Web page.

Microsoft Azure | portal.azure.com

BrivaCalendar-EWS | Certificates & secrets

Certificates

Certificates can be used as secrets to prove the application's identity when requesting a token. Also can be referred to as public keys.

Upload certificate

Thumbprint	Start date	Expires	Certificate ID
No certificates have been added for this application.			

Client secrets

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

New client secret

Description	Expires	Value
my_client_secret	6/24/2023	6-Cd0Vxv6p1wwH4y8Q6Yr11_SY7.dU~T_t 8a3b5df4-fd5e-40b1-8127-3fce2607d75

Copy to clipboard

18

Grant permissions

In the API permissions menu, click on the Add a permission button.

The screenshot shows the Microsoft Azure portal interface. The left sidebar has a 'Manage' section with 'API permissions' highlighted. The main content area shows a table of configured permissions. One permission is listed: Microsoft Graph (1) - User.Read (Delegated) - Sign in and read user profile. The 'Admin consent required' column for this permission shows 'No'.

Select the Microsoft Graph button.

The screenshot shows the Microsoft Azure portal interface with a 'Request API permissions' dialog box open. The 'Microsoft APIs' tab is selected. Under 'Commonly used Microsoft APIs', the 'Microsoft Graph' section is highlighted with an orange box. Other sections shown include 'Azure DevOps', 'Azure Rights Management Services', 'Azure Service Management', 'Dynamics 365 Business Central', 'Dynamics CRM', 'Flow Service', 'Intune', 'Office 365 Management APIs', and 'OneNote'.

Select the Delegated permissions button.

The screenshot shows the Microsoft Azure portal interface. On the left, there's a sidebar with navigation links like Overview, Quickstart, Integration assistant | Preview, Manage, and API permissions (which is currently selected). The main content area is titled 'BrivaCalendar-EWS | API permissions'. It shows a table of 'Configured permissions' for Microsoft Graph, with one row for 'User.Read' (Type: Delegated, Description: Sign in and read basic profile information). To the right, a modal window titled 'Request API permissions' is open for Exchange. It has sections for 'Delegated permissions' (selected) and 'Application permissions' (disabled). Under 'Delegated permissions', it says 'Your application needs to access the API as the signed-in user.' At the bottom of the modal is a 'Grant admin consent for Innes' button.

In the EWS item, check the `EWS.AccessAsUser.All` item. Then click on the Add permissions button.

This screenshot shows the continuation of the API permissions configuration for BrivaCalendar-EWS. The left sidebar remains the same. The main area now shows the 'Select permissions' step of the 'Request API permissions' process for Exchange. A search bar at the top right says 'Start typing a reply url to filter these results'. Below it is a table with columns 'Permission' and 'Admin consent required'. Under the 'EWS (1)' section, there is a single item: 'EWS.AccessAsUser.All' with a description 'Access mailboxes as the signed-in user via Exchange Web Services'. A checkbox next to it is checked. At the bottom of the table are 'Add permissions' and 'Discard' buttons.

Click on the APIs my organization uses tab. Then in the filter input, enter the *Office 365 Exchange Online* filter value. Click on the *Office 365 Exchange Online* button.

The screenshot shows the Microsoft Azure portal interface. On the left, the navigation pane is visible with sections like Overview, Quickstart, Integration assistant, Manage, Branding & properties, Authentication, Certificates & secrets, Token configuration, API permissions, Expose an API, App roles, Owners, Roles and administrators, and Manifest. The 'API permissions' section is selected. In the center, there's a search bar and a 'Request API permissions' modal window. The modal has tabs for 'Microsoft APIs', 'APIs my organization uses' (which is selected), and 'My APIs'. Under 'APIs my organization uses', 'Office 365 Exchange Online' is listed with its application ID. Below this, a table shows configured permissions for Microsoft Graph and Office 365 Exchange Online, including 'EWS.AccessAsUser.All' and 'User.Read'.

Click on the Application permissions button. Then in the filter input, enter the *full_access_as_app* filter value. Check the *full_access_as_app* permissions checkbox.

This screenshot shows the same Microsoft Azure interface as the previous one, but the 'Application permissions' tab is selected in the 'Request API permissions' modal. The filter input field contains 'full_access_as_app'. A checkbox for this permission is checked, and the 'Permissions (1)' section shows it with the description 'Use Exchange Web Services with full access to all mailboxes' and a status of 'Yes'. Other permission options like 'EWS.AccessAsUser.All' and 'User.Read' are also listed.

Click then on the *Grant admin consent for <your_organisation>* button.

The screenshot shows the Microsoft Azure interface again, but now the 'Grant admin consent for innes' button in the 'Configured permissions' section is highlighted with a red box. This indicates the step where the user needs to grant administrative consent for the app's permissions.

Now the permissions are granted.

The screenshot shows the Microsoft Azure portal interface for managing app registrations. The left sidebar lists various management sections like Overview, Quickstart, Integration assistant, and API permissions, which is currently selected. The main content area displays the 'BrivaCalendar-EWS | API permissions' page. It includes a warning message about editing permissions and a note about admin consent requirements. A table lists three configured permissions:

API / Permissions name	Type	Description	Admin consent req...	Status
EWS.AccessAsUser.All	Delegated	Access mailboxes as the signed-in user via Exchange Web ...	No	Granted for INNIES (INNIES@ONMICROSOFT.COM)
User.Read	Delegated	Sign in and read user profile	No	Granted for INNIES (INNIES@ONMICROSOFT.COM)
full_access_as_app	Application	Use Exchange Web Services with full access to all mailboxes	Yes	Granted for INNIES (INNIES@ONMICROSOFT.COM)

Note: The `EWS.AccessAsUser.All` permission allows to authenticate with the delegate account: <https://docs.microsoft.com/en-us/exchange/client-developer/exchange-web-services/how-to-authenticate-an-ews-application-by-using-oauth#configure-for-delegated-authentication>.

3.3 Appendix: Azure AD Application creation with PowerShell module

⚠ To support BrivaCalendar-EWS 1.11.10, the PowerShell script for Azure Active Directory Application support (`Powershell_Innes_AAD`) must be 1.10.15 (or above).

Download the PowerShell script for Azure Active Directory Application support `Powershell_Innes_AAD-1.10.15.zip` from the [Innes Site Web](#) then follow the instructions below.

Compatibility

The `Powershell_Innes_AAD-1.10.15.zip` PowerShell script for Azure Active Directory application is compatible with PowerShell 5.X (deployed on Windows 10).

Introduction

This set of `Powershell` functions allows to:

- create an Azure Active Directory application, with the `New-AADApplication` function,
- remove an Azure Active Directory application, with the `Remove-AADApplication` function.

These functions are defined in the `PSAAD` PowerShell module stored in the `Modules\PSAAD\` directory.

The result of the `Powershell` functions is also stored in a JSON file.

Edit the file and store preciously the values which could be required for your application:

- the `clientId` value,
- the `tenantId` value,
- the `clientSecret` value.

Security

By default, the execution of local `Powershell` scripts are not allowed. You can change their execution rights by changing the `PowerShell` security policy. This modification has to be done once with the `Set-ExecutionPolicy` `Powershell` function. Your organisation may have to change it according to your security rules.

For example, to authorize the execution of all scripts, launch a `Powershell` console with administrator rights, and type:

```
PS > Set-ExecutionPolicy -ExecutionPolicy Unrestricted -scope CurrentUser
```

For further information, look at the cmdlet `Set-ExecutionPolicy` help page.

If you cannot allow the execution of unsigned local scripts, you can install the provided certificate in the list of authorized root certificates with the command:

```
PS > cd <your_path_to_the_scripts>\Powershell_Innes_AAD\Certificate  
PS > Import-PfxCertificate -FilePath InnesCodeSigningRootCA_1.pfx -CertStoreLocation .../  
cert:\CurrentUser\Root -Password $(ConvertTo-SecureString "1234" -AsPlainText -Force)
```

To import the .pfx certificate, you can also use the MS-Windows application `certmgr.msc`, select the Trusted Root Certification Authorities , right clic on ALL Tasks , select the Import item, select the file and enter the password 1234 . When ended, close the current Powershell console.

Prerequisite

Install the AzureAD module

Install the `AzureAD` module with the command below:

```
PS > Install-Module -name AzureAD -scope CurrentUser
```

Dependency

If this message is prompted, enter `y`.

```
The NuGet supplier is required to continue  
PowerShellGet requires the NuGet vendor, version 2.8.5.201 or later, to interact with the repositories.  
The NuGet provider must be available in "C:\Program Files\PackageManagement\ProviderAssemblies" or .../  
"C:\Users\<username>\AppData\Local\PackageManagement\ProviderAssemblies".  
You can also install the provider NuGet by executing the command "Install-PackageProvider -Name NuGet .../  
-MinimumVersion 2.8.5.201 -Force". Do you want that PowerShellGet installs and imports the NuGet provider now?  
[Y] Yes [N] No [S] Suspend [?] Help (default is "Y"):
```

If this message is prompted, enter `y`.

```
Unapproved repository  
You install the modules from an unapproved repository. If you approve this repository, .../  
change its InstallationPolicy value by running the Set-PSRepository command applet. .../  
Do you really want to install From PSGallery ?  
[Y] Yes [T] Yes for all [N] No [U] No for all [S] Suspend [?] Help (default is "N"):
```

Usage

To use one of the `Powershell` modules, you have to define the environment variable for `PSAAD`. You have 3 possibilities:

1. Either copy the directories under `Modules\` into a standard `Powershell` module installation directory, for example `C:\Program Files\WindowsPowerShell\Modules` . Then launch a `Powershell` console.
2. Or redefine the search variable for `Powershell` modules (the `$Env:PSModulePath` `Powershell` variable) each time you will use theses functions. In this case, launch a `Powershell` console, and type the line below, adapting it to your path. Each time you launch a new `Powershell` console, you need to enter it again.

Example:

```
PS > $Env:PSModulePath="$Env:PSModulePath;C:\Program Files (x86)\WindowsPowerShell\Modules"
```

3. Or redefine the search variable for `Powershell` modules in the Windows environment variables. For that, add the path `<your_path_to_the_scripts>\Powershell_Innes_AAD\Modules` to the environment variable `PSModulePath` . Then, launch afterwards a `Powershell` console.

To use the functions or get help, you must then import the module(s) with the `Import-Module` function. Example:

```
PS > Import-Module PSAAD
```

Depending on how your get the scripts, you may have this following warning:

```
Security Warning Run only scripts that you trust. While scripts from the Internet can be useful, .../  
this script can potentially harm your computer. Do you want to run \server\scripts\my.ps1? .../  
[D] Do not run [R] Run once [S] Suspend [?] Help (default is "D"):
```

To avoid this message, you can unblock the script files (to do only once):

```
PS > cd <your_path_to_the_scripts>\Powershell_Innes_AAD  
PS > dir -Recurse | Unblock-File
```

The `Get-Command` function allows you to list the functions defined in a module. Example:

```
PS > Get-Command -Module PSAAD
```

Answer example:

CommandType	Name	Version	Source
Function	New-AADApplication	1.10.14	PSAAD
Function	Remove-AADApplication	1.10.14	PSAAD

You can get help on each function of the module by using the standard cmdlet `Get-Help` with options:

- `-detailed`,
- `-full`,
- `-examples`.

Example:

```
PS > Get-Help -detailed New-AADApplication
```

NAME
New-AADApplication

SYNOPSIS
This function creates a Azure Active Directory application.

SYNTAX
`New-AADApplication [[-Credential] <PSCredential>] [[-tenantId] <String>] [-appName] <String> [-authorizations] <String[]> [[-LogFile] <String>] [<CommonParameters>]`

DESCRIPTION
This function creates a Azure Active Directory application.

PARAMETERS

`-Credential <PSCredential>`
Credential (admin profile) used to create the Azure Active Directory application. If absent, a dialog is displayed in the browser to enter the credentials.

`-tenantId <String>`
Azure Active Directory Tenant Id of the tenant in which the application has been created. This parameter is not mandatory. If absent, the tenantId is retrieved automatically after the credentials have been entered in the dialog.

`-appName <String>`
Name of the Azure Active Directory application.

`-authorizations <String[]>`
Authorization type:
- "signcom_m365" : to access to OneDrive and Web sites for SignCom application
- "signmeeting_ews": to access to MS-Exchange room mailbox resources for SignMeeting MS-Exchange application
- "signmeeting_m365": to access to M365 room mailbox resources for SignMeeting-M365 application
- "briva_calendar_ews": to access to MS-Exchange room mailbox resources for Briva Calendar EWS application
- "m365_room": to access to M365 room mailbox resource for SBL10e m365_room application
- "m365_user": to access to M365 user presence resource for SBL10e m365_user application
- "powerbi": to access to Power BI report

`-LogFile <String>`
Log file path

`<CommonParameters>`
This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (<https://go.microsoft.com/fwlink/?LinkID=113216>).

----- EXAMPLE 1 -----

```
PS C:\>$result = New-AADApplication -appname "my-App-Label" -authorizations "Authorization type"
```

A consent request will be sent in 30 seconds in your browser.
You must log into an administrator account of your organization and grant the necessary permissions.

```
PS C:\>$result
```

Name	Value
---	----
clientId	xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
objectId	xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
spId	xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
name	my-App-Label
tenantId	xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
clientSecret	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

REMARKS

To see the examples, type: "get-help New-AADApplication -examples".
For more information, type: "get-help New-AADApplication -detailed".
For technical information, type: "get-help New-AADApplication -full".

Example to create an Azure Active Directory application for Briva Calendar EWS

```
PS > $briva_calendar_ews = New-AADApplication -appname "BrivaCalendar-EWS" -authorizations "briva_calendar_ews"
```

⚠ Don't use space characters inside the appname else an error could be returned.

⚠ Don't use an already existing Appname else an error is returned.

⚠ Clicking on a Powershell window can suspend the command. In this case click again in the window to resume the command.*

A login popup is displayed . Enter once your EWS login credentials. This message is then displayed in a *Powershell* context.

You must log into an administrator account of your organisation and grant the necessary permissions.
A consent request will be sent within 30 seconds in your browser.

After thirty seconds, a login popup should be prompted (<https://login.microsoftonline.com/>) automatically in your default Web browser.

Enter again your EWS login credentials.

A new popup message with the *Permission requested, review for your organisation* title is prompted in your Web browser.

Click on the *Accept* button. Then a message is displayed in your Web browser showing that the consent is successful: *Success of the consent request*.

You can view the data of the created application by typing the following command :

```
PS > $brivacalendar_ews
Name          Value
----          -----
clientId      xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
objectId      xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
spId          xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
name          BrivaCalendar-EWS
tenantId     xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
clientSecret xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

The result of the *Powershell* function is also stored in a JSON file: `brivacalendar_ews.json`.

Edit the file and store preciously the values required for your application:

- the `clientId` value,
- the `tenantId` value,
- the `clientSecret` value.

Example to delete an Azure Active Directory application

```
PS > Remove-AADApplication -appname "BrivaCalendar-EWS"
```

A login popup is opened. Enter your EWS credentials. In case the values do not allow BrivaCalendar-EWS to work properly, check in Azure portal that the application has been created succesfully and the rights are properly granted. If not, wait for a while, the rights granting may take several hours.

3.4 Appendix: .xml calendar format

This is an example of .xml calendar output by the briva_calendar-ews (1.11.10 or above) with the URL syntax:

- http://<BrivaCalendarServer_domain_or_ipv4_addr>/plgnCast/.applets/.ews-calendar/2xml-daycalendar.php?id=<calendarId>

For example:

- http://192.168.1.100/plgnCast/.applets/.ews-calendar/2xml-daycalendar.php?id=room_1_2_3

```
<?xml version="1.0" encoding="utf-8"?>
<calendar>
  <event>
    <summary>myEvent1</summary>
    <description>myDesc1</description>
    <location>Room 1</location>
    <organizer>myDelegateAccount@contoso.com</organizer>
    <attendees>stefan.schmidt@contoso.com, john.smith@contoso.com</attendees>
    <date>26/12/2022</date>
    <timeslot>00:00 - 01:00</timeslot>
  </event>
  <event>
    <summary>myEvent2</summary>
    <description>myDesc2</description>
    <location>Room 2</location>
    <organizer>myDelegateAccount@contoso.com</organizer>
    <attendees>john.smith@contoso.com</attendees>
    <date>26/12/2022</date>
    <timeslot>01:00 - 02:00</timeslot>
  </event>
  <event>
    <summary>event3</summary>
    <description>myDesc3</description>
    <location>Room 3</location>
    <organizer>myDelegateAccount@contoso.com</organizer>
    <attendees>stefan.schmidt@contoso.com</attendees>
    <date>26/12/2022</date>
    <timeslot>02:00 - 03:00</timeslot>
  </event>
</calendar>
```

3.5 Appendix: .ics calendar format

The `.ics` format output by the `briva_calendar-ews` is compliant with the [Internet Calendaring and Scheduling RFC](#).

3.6 Appendix: Alarm and cache persistency

Alarm

Several alarms combinations can be created to:

- display or not the incoming events,
- display or not the events that are over,
- display events <n> minutes before they happen,
- display events until <n> minutes they are over.

This alarm combination allows to display all the events of the day:

```
<scc:calendars>
  <scc:calendar id="room1" server="url(#serverId)">
    <param name="startRelated">day-start</param>
    <param name="endRelated">day-end</param>
  </scc:calendar>
</scc:calendars>
```

This alarm combination allows to display only events happening now:

```
<scc:calendars>
  <scc:calendar id="room1" server="url(#serverId)">
    <param name="startRelated">event-start</param>
    <param name="endRelated">event-end</param>
  </scc:calendar>
</scc:calendars>
```

This alarm combination allows to display events 3600 seconds before they happen and until 3600 seconds after they are over:

```
<scc:calendars>
  <scc:calendar id="room1" server="url(#serverId)">
    <param name="startRelated">event-start</param>
    <param name="startOffset">-3600</param>
    <param name="endRelated">event-end</param>
    <param name="endOffset">3600</param>
  </scc:calendar>
</scc:calendars>
```

This alarm combination allows to display all the events of the day except those that are not yet started:

```
<scc:calendars>
  <scc:calendar id="room1" server="url(#serverId)">
    <param name="startRelated">day-start</param>
    <param name="endRelated">event-end</param>
  </scc:calendar>
</scc:calendars>
```

This alarm combination allows to display all the events of the day except those that are over:

```
<scc:calendars>
  <scc:calendar id="room1" server="url(#serverId)">
    <param name="startRelated">event-start</param>
    <param name="startOffset">-600</param>
    <param name="endRelated">day-end</param>
  </scc:calendar>
</scc:calendars>
```

Cache persistency parameter

The connector considers that during the cache duration (in seconds), there is no need to refresh the *calendarId* calendar. The default cache value is 300.

```
<param name="cachePersistence">300</param>
```

When the cache is not defined, the connector refreshes the calendar every minute.

3.7 Appendix: XML entities

Here are some XML entities:

XML entities	description
<	<
>	>
&	&
'	'
"	"

3.8 Appendix: Turn on the basic authentication (obsolete)

Since 2021, for security reasons, Microsoft has recently turned off by default the Basic authentication and turned-on the Azure AD authentication (OAuth2 or modern authentication) for any new Office 365 accounts.

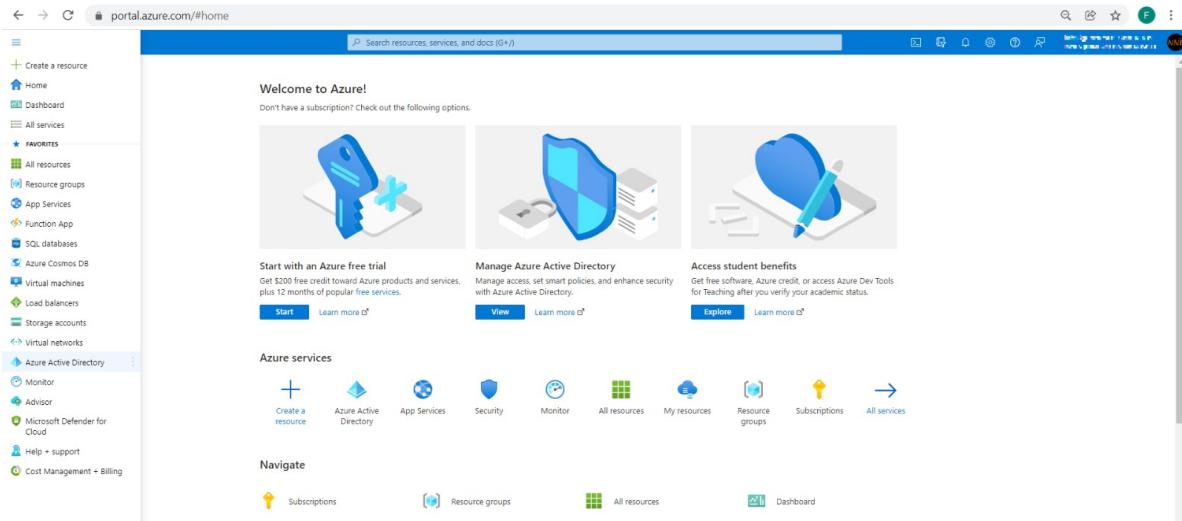
Anyway, if the customer accepts the security risks, Microsoft still accepts to keep turned-on, for any MS-Exchange 365 account, both:

- the basic authentication and
- the Azure AD (OAuth 2 or modern authentication).

Inactivate security default for your Office 365 server

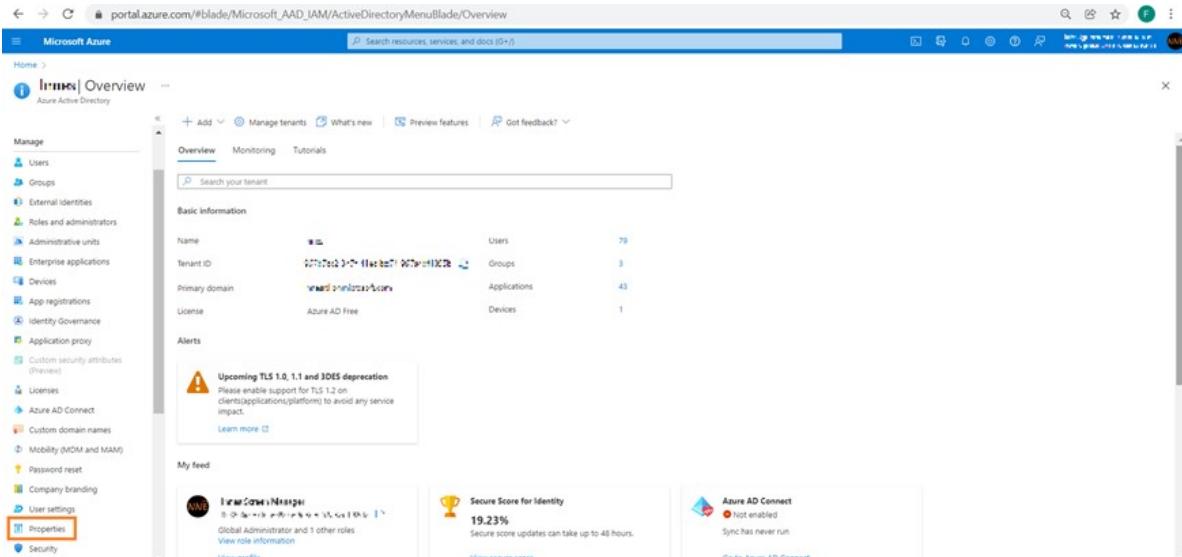
To activate the basic authentication to access to your Office 365 server, you need first to inactivate Security default:

- connect on Microsoft Azure portal: <https://portal.azure.com/> .
- sign in with your Office 365 account credentials (administrator profile)
- click on the left top menu and choose the *Azure Active directory* item.



The screenshot shows the Microsoft Azure portal homepage. On the left, there is a navigation sidebar with various service icons and links. The 'Azure Active Directory' link is highlighted with a blue selection bar. The main content area features a 'Welcome to Azure!' banner with three cards: 'Start with an Azure free trial', 'Manage Azure Active Directory', and 'Access student benefits'. Below this is a section titled 'Azure services' with icons for 'Create a resource', 'Azure Active Directory', 'App Services', 'Security', 'Monitor', 'All resources', 'My resources', 'Resource groups', 'Subscriptions', and 'All services'. At the bottom of the page, there are 'Navigate' links for 'Subscriptions', 'Resource groups', 'All resources', and 'Dashboard'.

- scroll to the bottom to click on the *Properties* item.



The screenshot shows the 'Overview' blade for Azure Active Directory. The left sidebar has a 'Properties' item highlighted with a red box. The main content area displays 'Basic Information' with the following data:

Name	Users	Groups
Tenant ID	00757e12-3127-4fbc-9207-26701128	3
Primary domain	www.simploncloud.com	43
License	Azure AD Free	1

There is also an 'Alerts' section with a warning about TLS 1.0, 1.1 and 3DES deprecation. The 'My feed' section includes items like 'Secure Score for Identity' (19.23%) and 'Azure AD Connect' (not enabled). The URL in the browser is https://portal.azure.com/#blade/Microsoft_AAD_IAM/ActiveDirectoryMenuBlade/Overview.

- click on the button *Manage Security defaults*.

- On the right panel, toggle to the right, meaning to the **No** value and click on the **Save** button to turn off Security defaults.

Wait for five minutes, the time for Microsoft to consider the modification.

To return to *security defaults* activated, toggle the button to the left, meaning to the **Yes** value (meaning `basic` authentication inactivated).

Activate the basic authentication for your Office 365 server with the Run Tests tool

This is a Microsoft article explaining a way to activate back the `basic` authentication for your Office 365 account. <https://techcommunity.microsoft.com/t5/exchange-team-blog/basic-authentication-and-exchange-online-september-2021-update/ba-p/2772210>

Procedure:

- on your Web browser, connect to your Office 365 portal with your account credentials (administrator profile).
- scroll in the Microsoft article page to the blue bottom and click on the *Diag: Enable Basic Auth in EXO* button, shortcut to following URL:
<https://admin.microsoft.com/AdminPortal/?searchSolutions=Diag%20Enable%20Basic%20Auth%20in%20EXO#/homepage>

The screenshot shows a Microsoft Tech Community article titled "Proactive Protection Expansion". The article discusses the plan to disable Basic Auth for some customers starting early 2022. It highlights that Modern Auth will not be affected. A "Limited Opt Out" section explains how users can request to disable specific protocols. A "Run Tests" button is visible at the bottom of the article.

Click on the *Run Tests* button

The screenshot shows the Microsoft 365 Admin Center with the search bar set to "Diag: Enable Basic Auth in EXO". The "Run diagnostics" section contains a message about updating basic authentication settings and a "Run Tests" button. The "User management" sidebar is visible on the right.

Wait for few seconds. Once the research is completed, open the drop down list to see the protocols or features you want to support with the *Basic Authentication*.
Select the *Exchange Web services* value (used by the *EWS-Calendar* connector).

The screenshot shows the Microsoft 365 Admin Center with the search bar set to "Diag: Enable Basic Auth in EXO". The "Run diagnostics" section now displays the current basic authentication settings and a dropdown menu for selecting the protocol to opt out. The "Protocol to Opt Out" dropdown is set to "Exchange Web Services (EWS)". A checkbox for acknowledging changes is also present.

Check the option *I acknowledge clicking 'Update settings' will make the change(s) described above to the tenant configuration* checkbox then click on the *Update settings* button. A confirmation message is displayed showing that basic authentication is now activated for the selected Exchange Web services feature.

Innes

Microsoft 365 admin center

Search

How can we help?

Diag: Enable Basic Auth in EXO

User management

Add cards

Microsoft Teams

Support remote workers with Teams

Learn how to manage Teams for remote work, with setup guidance, short videos, and tips.

Teams is on for your organization

Check setup status for new Teams users

Guest access is on

Add user

Office apps

Run diagnostics

These are the current Basic authentication settings:

Microsoft has not blocked Basic authentication for any protocol or feature for your tenant. You can use the drop-down below to indicate which protocols you wish to exclude from being secured. Basic authentication will be disabled for those protocols at a later date.

Protocol to Opt Out *

Exchange Web Services (EWS)

I acknowledge clicking 'Update settings' will make the change(s) described above to the tenant configuration.

Update Settings

Wait for few minutes and restart your Briva calendar server.