# How to Get Delegate Information in Office 365 Exchange Online

## Introduction

Currently, you can easily get delegates in Outlook. But you will find this feature is not available in Outlook Web App (OWA). In this application, we will demonstrate how get the delegate information of multi accounts.

1. Get the accounts that users input

2. Set the ImpersonatedUserId property if the login account has the impersonation permission.

3. Get all the delegate information of the accounts.

## Running the Sample

Press F5 to run the sample, you will get the following result.

First, we use our account to connect to the Exchange Online.



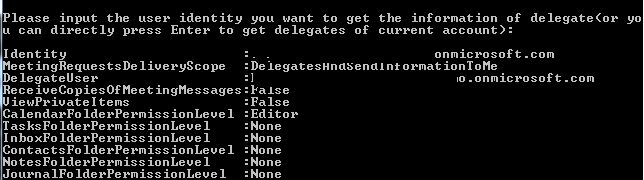
Then, we can get delegate information of multi accounts if we have the impersonation permission.



The following is the result of delegate information:



If we don’t have the impersonation permission, we can directly press Enter to get the delegate information of the login account.



## Using the Code

If we have impersonation permission, we can get delegate information of multi accounts by setting the ImpersonatedUserId property.

|  |
| --- |
| -Code block start-  --C# code snippet start--  foreach (String identity in identities)  {  // If the user identity is valid, we will set it as the ImpersonatedUserId.  NameResolutionCollection nameResolutions =  service.ResolveName(identity, ResolveNameSearchLocation.DirectoryOnly, true);  if (nameResolutions.Count != 1)  {  Console.WriteLine("{0} is invalid user identity.", identity);  }  else  {  String emailAddress = nameResolutions[0].Mailbox.Address;  service.ImpersonatedUserId =  new ImpersonatedUserId(ConnectingIdType.SmtpAddress, emailAddress);  GetAccountDelegates(service, emailAddress);  }  }  --C# code snippet end--  --VB code snippet start--  For Each identity As String In identities  ' If the user identity is valid, we will set it as the ImpersonatedUserId.  Dim nameResolutions As NameResolutionCollection =  service.ResolveName(identity, ResolveNameSearchLocation.DirectoryOnly, True)  If nameResolutions.Count <> 1 Then  Console.WriteLine("{0} is invalid user identity.", identity)  Else  Dim emailAddress As String = nameResolutions(0).Mailbox.Address  service.ImpersonatedUserId =  New ImpersonatedUserId(ConnectingIdType.SmtpAddress, emailAddress)  GetAccountDelegates(service, emailAddress)  End If  Next identity  --VB code snippet end--  -Code block end- |

If we don’t have the impersonation permission, we can only get the delegate information of the login account.

|  |
| --- |
| -Code block start-  --C# code snippet start--  else  {  // We can also directly press Enter to get the delegate information of the  // login account.  GetAccountDelegates(service, currentAddress);  }  --C# code snippet end--  --VB code snippet start--  Else  ' We can also directly press Enter to get the delegate information of the  ' login account.  GetAccountDelegates(service, currentAddress)  End If  --VB code snippet end--  -Code block end- |

We get the delegate information of the specific account.

|  |
| --- |
| -Code block start-  --C# code snippet start--  static void GetAccountDelegates(ExchangeService service, String emailAddress, params UserId[] userIds)  {  var emailBox = new Mailbox(emailAddress);  DelegateInformation result = service.GetDelegates(emailBox, true, userIds);  foreach (var response in result.DelegateUserResponses)  {  if (response.ErrorCode != ServiceError.NoError)  {  Console.WriteLine(response.ErrorMessage);  }  else  {  Console.WriteLine("{0,-30}:{1}", "Identity", emailAddress);  Console.WriteLine("{0,-30}:{1}", "MeetingRequestsDeliveryScope",  result.MeetingRequestsDeliveryScope);  Console.WriteLine("{0,-30}:{1}", "DelegateUser",  response.DelegateUser.UserId.PrimarySmtpAddress);  Console.WriteLine("{0,-30}:{1}", "ReceiveCopiesOfMeetingMessages",  response.DelegateUser.ReceiveCopiesOfMeetingMessages);  Console.WriteLine("{0,-30}:{1}", "ViewPrivateItems",  response.DelegateUser.ViewPrivateItems);  Console.WriteLine("{0,-30}:{1}", "CalendarFolderPermissionLevel",  response.DelegateUser.Permissions.CalendarFolderPermissionLevel);  Console.WriteLine("{0,-30}:{1}", "TasksFolderPermissionLevel",  response.DelegateUser.Permissions.TasksFolderPermissionLevel);  Console.WriteLine("{0,-30}:{1}", "InboxFolderPermissionLevel",  response.DelegateUser.Permissions.InboxFolderPermissionLevel);  Console.WriteLine("{0,-30}:{1}", "ContactsFolderPermissionLevel",  response.DelegateUser.Permissions.ContactsFolderPermissionLevel);  Console.WriteLine("{0,-30}:{1}", "NotesFolderPermissionLevel",  response.DelegateUser.Permissions.NotesFolderPermissionLevel);  Console.WriteLine("{0,-30}:{1}", "JournalFolderPermissionLevel",  response.DelegateUser.Permissions.JournalFolderPermissionLevel);  Console.WriteLine();  }  }  }  --C# code snippet end--  --VB code snippet start--  Private Shared Sub GetAccountDelegates(ByVal service As ExchangeService,  ByVal emailAddress As String,  ByVal ParamArray userIds() As UserId)  Dim emailBox = New Mailbox(emailAddress)  Dim result As DelegateInformation = service.GetDelegates(emailBox, True, userIds)  For Each response In result.DelegateUserResponses  If response.ErrorCode <> ServiceError.NoError Then  Console.WriteLine(response.ErrorMessage)  Else  Console.WriteLine("{0,-30}:{1}", "Identity", emailAddress)  Console.WriteLine("{0,-30}:{1}", "MeetingRequestsDeliveryScope",  result.MeetingRequestsDeliveryScope)  Console.WriteLine("{0,-30}:{1}", "DelegateUser",  response.DelegateUser.UserId.PrimarySmtpAddress)  Console.WriteLine("{0,-30}:{1}", "ReceiveCopiesOfMeetingMessages",  response.DelegateUser.ReceiveCopiesOfMeetingMessages)  Console.WriteLine("{0,-30}:{1}", "ViewPrivateItems",  response.DelegateUser.ViewPrivateItems)  Console.WriteLine("{0,-30}:{1}", "CalendarFolderPermissionLevel",  response.DelegateUser.Permissions.CalendarFolderPermissionLevel)  Console.WriteLine("{0,-30}:{1}", "TasksFolderPermissionLevel",  response.DelegateUser.Permissions.TasksFolderPermissionLevel)  Console.WriteLine("{0,-30}:{1}", "InboxFolderPermissionLevel",  response.DelegateUser.Permissions.InboxFolderPermissionLevel)  Console.WriteLine("{0,-30}:{1}", "ContactsFolderPermissionLevel",  response.DelegateUser.Permissions.ContactsFolderPermissionLevel)  Console.WriteLine("{0,-30}:{1}", "NotesFolderPermissionLevel",  response.DelegateUser.Permissions.NotesFolderPermissionLevel)  Console.WriteLine("{0,-30}:{1}", "JournalFolderPermissionLevel",  response.DelegateUser.Permissions.JournalFolderPermissionLevel)  Console.WriteLine()  End If  Next response  End Sub  --VB code snippet end--  -Code block end- |

## More Information

[EWS Managed API 2.0](http://msdn.microsoft.com/en-us/library/dd633709(v=exchg.80).aspx)

[Working with impersonation by using the EWS Managed API](http://msdn.microsoft.com/en-us/library/dd633680(v=exchg.80).aspx)