# How to Get Inbox Rules in Office 365

## Introduction

## Currently, most of you manage email messages by inbox rules. Especially, when you become an owner of a shared mailbox, you find the former owner created a lot of inbox rules to manage email messages efficiently. But you need to modify these inbox rules to meet the new business needs. Before changing these inbox rules, you want to find a solution to document these inbox rules in case something goes wrong. But you don't have an out-of-box solution.

In this application, we will demonstrate how to get Inbox rules in Office 365:

1. Get the accounts that users input

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

3. Get Inbox rules of the accounts.

## Running the Sample

Press F5 to run the sample.

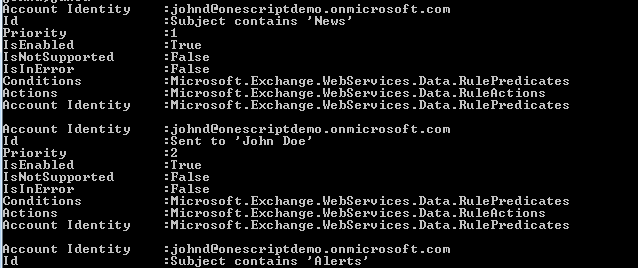
First, we use our account to connect to Exchange Online.



Then, we can get Inbox rules of multi accounts if we have the impersonation permission.



The following is the result of Inbox rules:



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

## Using the Code

If we have impersonation permission, we can get Inbox rules 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);  GetAccountGetInboxRules(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)  GetAccountGetInboxRules(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 Inbox rules of the login account.

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

We get the Inbox rules of the specific account.

|  |
| --- |
| -Code block start-  --C# code snippet start--  static void GetAccountGetInboxRules(ExchangeService service, String emailAddress,  params UserId[] userIds)  {  var emailBox = new Mailbox(emailAddress);  RuleCollection rules=service.GetInboxRules(emailAddress);  if (rules.Count <= 0)  {  Console.WriteLine("There's no rule for the account {0}.", emailAddress);  Console.WriteLine();  }  else  {  foreach (Rule rule in rules)  {  Console.WriteLine("{0,-20}:{1}", "Account Identity", emailAddress);  Console.WriteLine("{0,-20}:{1}", "Id", rule.DisplayName);  Console.WriteLine("{0,-20}:{1}", "Priority", rule.Priority);  Console.WriteLine("{0,-20}:{1}", "IsEnabled", rule.IsEnabled);  Console.WriteLine("{0,-20}:{1}", "IsNotSupported", rule.IsNotSupported);  Console.WriteLine("{0,-20}:{1}", "IsInError", rule.IsInError);  Console.WriteLine("{0,-20}:{1}", "Conditions", rule.Conditions);  Console.WriteLine("{0,-20}:{1}", "Actions", rule.Actions);  Console.WriteLine("{0,-20}:{1}", "Account Identity", rule.Exceptions);  Console.WriteLine();  }  }  }  --C# code snippet end--  --VB code snippet start--  Private Shared Sub GetAccountGetInboxRules(ByVal service As ExchangeService,  ByVal emailAddress As String,  ByVal ParamArray userIds() As UserId)  Dim emailBox = New Mailbox(emailAddress)  Dim rules As RuleCollection = service.GetInboxRules(emailAddress)  If rules.Count <= 0 Then  Console.WriteLine("There's no rule for the account {0}.", emailAddress)  Console.WriteLine()  Else  For Each rule As Rule In rules  Console.WriteLine("{0,-20}:{1}", "Account Identity", emailAddress)  Console.WriteLine("{0,-20}:{1}", "Id", rule.DisplayName)  Console.WriteLine("{0,-20}:{1}", "Priority", rule.Priority)  Console.WriteLine("{0,-20}:{1}", "IsEnabled", rule.IsEnabled)  Console.WriteLine("{0,-20}:{1}", "IsNotSupported", rule.IsNotSupported)  Console.WriteLine("{0,-20}:{1}", "IsInError", rule.IsInError)  Console.WriteLine("{0,-20}:{1}", "Conditions", rule.Conditions)  Console.WriteLine("{0,-20}:{1}", "Actions", rule.Actions)  Console.WriteLine("{0,-20}:{1}", "Account Identity", rule.Exceptions)  Console.WriteLine()  Next rule  End If  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)