# How to Set Retention Policy for Email Messages in Office 365 Exchange Online

## Introduction

Currently, you can easily set email messages with Retention Policy enabled in Outlook and Outlook Web App. But you will find that it is not very convenient to set retention policies for email messages in a specific time range. In this application, we will demonstrate how to set retention policies for email messages in office 365:

1. Select the email messages you want to set the retention policy for;

2. Choose the retention policy you want to set for the email messages;

3. Set the retention policy for the email messages.

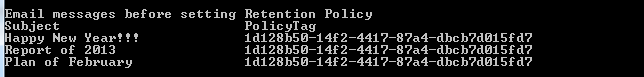
## Running the Sample

Press F5 to run the sample.

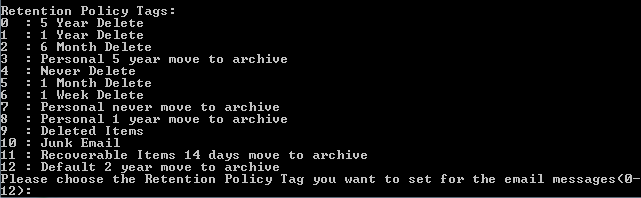
First, you can use your account to connect to the Exchange Online.



Then the emails in recent 30 days will be displayed in the screen.



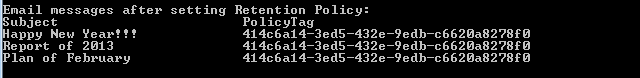
After that, you need to choose the retention policy that you want to set.



You can press a number to choose the policy and the application will set it for the email messages.



At last, you can view the result.



## Using the Code

1. Search the Email Messages

Before setting retention policy, you need to get the email messages first.

|  |
| --- |
| -Code block start-  --C# code snippet start--  private static List<Item> SearchEmailMessages(ExchangeService service,String subject,  DateTime startDate,DateTime endDate,String from,String displayTo,String displayCc)  {  PropertySet itemPropertySet =  new PropertySet(BasePropertySet.FirstClassProperties, EmailMessageSchema.PolicyTag);  SearchFilter.SearchFilterCollection searchFilterCollection =  new SearchFilter.SearchFilterCollection(LogicalOperator.And);  SearchFilter startDateFilter =  new SearchFilter.IsGreaterThanOrEqualTo(EmailMessageSchema.DateTimeCreated, startDate);  SearchFilter endDateFilter =  new SearchFilter.IsLessThanOrEqualTo(EmailMessageSchema.DateTimeCreated,endDate);  // Just select the email message items.  SearchFilter itemClassFilter =  new SearchFilter.IsEqualTo(EmailMessageSchema.ItemClass, "IPM.Note");  searchFilterCollection.Add(startDateFilter);  searchFilterCollection.Add(endDateFilter);  searchFilterCollection.Add(itemClassFilter);  if (!String.IsNullOrWhiteSpace(subject))  {  SearchFilter subjectFilter =  new SearchFilter.ContainsSubstring(EmailMessageSchema.Subject, subject);  searchFilterCollection.Add(subjectFilter);  }  if (!String.IsNullOrWhiteSpace(from))  {  SearchFilter fromFilter =  new SearchFilter.ContainsSubstring(EmailMessageSchema.From, from);  searchFilterCollection.Add(fromFilter);  }  if (!String.IsNullOrWhiteSpace(displayTo))  {  SearchFilter displayToFilter =  new SearchFilter.ContainsSubstring(EmailMessageSchema.DisplayTo, displayTo);  searchFilterCollection.Add(displayToFilter);  }  if (!String.IsNullOrWhiteSpace(displayCc))  {  SearchFilter displayCcFilter =  new SearchFilter.ContainsSubstring(EmailMessageSchema.DisplayCc, displayCc);  searchFilterCollection.Add(displayCcFilter);  }  List<Item> items = GetItems(service, searchFilterCollection, WellKnownFolderName.Inbox, itemPropertySet);  Console.WriteLine("{0,-30}{1}", "Subject", "PolicyTag");  foreach (Item item in items)  {  Console.WriteLine("{0,-30}{1}", item.Subject, item.PolicyTag);  }  Console.WriteLine();  return items;  }  --C# code snippet end--  --VB code snippet start--  Private Shared Function SearchEmailMessages(ByVal service As ExchangeService,  ByVal subject As String, ByVal startDate As Date,  ByVal endDate As Date, ByVal [from] As String,  ByVal displayTo As String, ByVal displayCc As String) As List(Of Item)  Dim itemPropertySet As New PropertySet(BasePropertySet.FirstClassProperties,  EmailMessageSchema.PolicyTag)  Dim searchFilterCollection As New SearchFilter.SearchFilterCollection(LogicalOperator.And)  Dim startDateFilter As SearchFilter =  New SearchFilter.IsGreaterThanOrEqualTo(EmailMessageSchema.DateTimeCreated, startDate)  Dim endDateFilter As SearchFilter =  New SearchFilter.IsLessThanOrEqualTo(EmailMessageSchema.DateTimeCreated, endDate)  ' Just select the email message items.  Dim itemClassFilter As SearchFilter =  New SearchFilter.IsEqualTo(EmailMessageSchema.ItemClass, "IPM.Note")  searchFilterCollection.Add(startDateFilter)  searchFilterCollection.Add(endDateFilter)  searchFilterCollection.Add(itemClassFilter)  If Not String.IsNullOrWhiteSpace(subject) Then  Dim subjectFilter As SearchFilter =  New SearchFilter.ContainsSubstring(EmailMessageSchema.Subject, subject)  searchFilterCollection.Add(subjectFilter)  End If  If Not String.IsNullOrWhiteSpace([from]) Then  Dim fromFilter As SearchFilter =  New SearchFilter.ContainsSubstring(EmailMessageSchema.From, [from])  searchFilterCollection.Add(fromFilter)  End If  If Not String.IsNullOrWhiteSpace(displayTo) Then  Dim displayToFilter As SearchFilter =  New SearchFilter.ContainsSubstring(EmailMessageSchema.DisplayTo, displayTo)  searchFilterCollection.Add(displayToFilter)  End If  If Not String.IsNullOrWhiteSpace(displayCc) Then  Dim displayCcFilter As SearchFilter =  New SearchFilter.ContainsSubstring(EmailMessageSchema.DisplayCc, displayCc)  searchFilterCollection.Add(displayCcFilter)  End If  Dim items As List(Of Item) =  GetItems(service, searchFilterCollection, WellKnownFolderName.Inbox, itemPropertySet)  Console.WriteLine("{0,-30}{1}", "Subject", "PolicyTag")  For Each item As Item In items  Console.WriteLine("{0,-30}{1}", item.Subject, item.PolicyTag)  Next item  Console.WriteLine()  Return items  End Function  --VB code snippet end--  -Code block end- |

2. Get Retention Policy

In Office 365 Exchange Online, there’re many retention policies. You can use one of them in subsequent email messages.

|  |
| --- |
| -Code block start-  --C# code snippet start--  private static RetentionPolicyTag GetRetentionPolicyTag(ExchangeService service, String userAddress)  {  service.GetPasswordExpirationDate(userAddress);  if (service.ServerInfo.MajorVersion < 15)  {  Console.WriteLine("This version of Exchange don't support PolicyTag.");  return null;  }  GetUserRetentionPolicyTagsResponse getUserRetentionPolicyTagsResponse =  service.GetUserRetentionPolicyTags();  if (getUserRetentionPolicyTagsResponse.ErrorCode != ServiceError.NoError)  {  Console.WriteLine("Error:{0}", getUserRetentionPolicyTagsResponse.ErrorMessage);  return null;  }  RetentionPolicyTag[] retentionPolicyTags=  getUserRetentionPolicyTagsResponse.RetentionPolicyTags;  do  {  Int32 policyTagsCount = -1;  foreach (RetentionPolicyTag retentionPolicyTag in retentionPolicyTags)  {  policyTagsCount++;  Console.WriteLine("{0,-3}: {1}", policyTagsCount, retentionPolicyTag.DisplayName);  }  Console.Write("Please choose the Retention Policy Tag you want to set for the email messages(0-{0}):",  policyTagsCount);  String selectedPolicyTag = Console.ReadLine();  Int32 selectedPolicyTagNum = -1;  if (Int32.TryParse(selectedPolicyTag, out selectedPolicyTagNum) &&  selectedPolicyTagNum >= 0 && selectedPolicyTagNum <= policyTagsCount)  {  return retentionPolicyTags[selectedPolicyTagNum];  }  } while (true);  }  --C# code snippet end--  --VB code snippet start--  Private Shared Function GetRetentionPolicyTag(ByVal service As ExchangeService,  ByVal userAddress As String) As RetentionPolicyTag  service.GetPasswordExpirationDate(userAddress)  If service.ServerInfo.MajorVersion < 15 Then  Console.WriteLine("This version of Exchange don't support PolicyTag.")  Return Nothing  End If  Dim getUserRetentionPolicyTagsResponse As GetUserRetentionPolicyTagsResponse =  service.GetUserRetentionPolicyTags()  If getUserRetentionPolicyTagsResponse.ErrorCode <> ServiceError.NoError Then  Console.WriteLine("Error:{0}", getUserRetentionPolicyTagsResponse.ErrorMessage)  Return Nothing  End If  Dim retentionPolicyTags() As RetentionPolicyTag =  getUserRetentionPolicyTagsResponse.RetentionPolicyTags  Do  Dim policyTagsCount As Int32 = -1  For Each retentionPolicyTag As RetentionPolicyTag In retentionPolicyTags  policyTagsCount += 1  Console.WriteLine("{0,-3}: {1}", policyTagsCount, retentionPolicyTag.DisplayName)  Next retentionPolicyTag  Console.Write("Please choose the Retention Policy Tag you want to set for the email messages(0-{0}):", policyTagsCount)  Dim selectedPolicyTag As String = Console.ReadLine()  Dim selectedPolicyTagNum As Int32 = -1  If Int32.TryParse(selectedPolicyTag, selectedPolicyTagNum) AndAlso  selectedPolicyTagNum >= 0 AndAlso selectedPolicyTagNum <= policyTagsCount Then  Return retentionPolicyTags(selectedPolicyTagNum)  End If  Loop While True  End Function  --VB code snippet end--  -Code block end- |

3. Set Retention Policies

After getting the email messages and the retention policy, the application will set the specified retention policy in the email messages.

|  |
| --- |
| -Code block start-  --C# code snippet start--  if (retentionPolicyTag != null)  {  Console.WriteLine("Setting the Retention Policy...");  PolicyTag policyTag = new PolicyTag(true, retentionPolicyTag.RetentionId);  foreach (Item item in items)  {  item.PolicyTag = policyTag;  item.Update(ConflictResolutionMode.AlwaysOverwrite);  }  }  --C# code snippet end--  --VB code snippet start--  If retentionPolicyTag IsNot Nothing Then  Console.WriteLine("Setting the Retention Policy...")  Dim policyTag As New PolicyTag(True, retentionPolicyTag.RetentionId)  For Each item As Item In items  item.PolicyTag = policyTag  item.Update(ConflictResolutionMode.AlwaysOverwrite)  Next item  --VB code snippet end--  -Code block end- |

## More Information

[Windows PowerShell Advanced Function](http://technet.microsoft.com/en-us/library/dd315326.aspx)  
[RetentionPolicyTag class](http://msdn.microsoft.com/en-us/library/exchange/microsoft.exchange.webservices.data.retentionpolicytag%28v=exchg.80%29.aspx)