PowerTCP Mail for .NET 4.0 Upgrade Guide

This guide provides a 1:1 conversion for the 3 major classes (Pop, Imap and Smtp) and enumerations of PowerTCP Mail for .NET from version 3.2.1.0 to version 4.0 (and generally applies to prior versions as well). For classes not covered in this guide, please see the Code Examples page in our help documentation for examples of common usage scenarios.

IMAP

Properties

Imap.AutoList - Removed. Imap.Mailboxes is now populated automatically when Imap.Authenticate() is called. To set the active mailbox to inbox, set Imap.SelectedMailbox to Imap.Mailboxes[“INBOX”].

Imap.AutoPurge - Removed. Messages marked for deletion may be automatically removed by the server (depending upon its configuration) when the selected mailbox is changed or closed (mailbox is closed on logout). Some servers may delete the message immediately when a message is marked for deletion. If your server deletes messages marked for deletion when changing or closing the mailbox, Imap.Abort() or Imap.Connection.Close() might circumvent it. Call Mailbox.Purge() to immediately delete the marked messages.

Imap.Busy - Removed. Thread safety has been implemented internally for discrete commands; methods will be executed in the order they were called in. Use Lock (C#) or SyncLock (VB.NET) to ensure that sets of commands are executed before those on other threads.

Imap.Certificate - Removed. For most cases, just add a certificate to the Imap.Session.Security.Certificates collection. If the certificate does not pass Microsoft's validation checks, implement Imap.Session.Security.SelectionCallback to force the use of the returned certificate. See the ‘Security Implementation’ help topic in the help documentation for more information and example code.

Imap.Connected – Imap.Connection.State

Imap.CurrentMailbox - Imap.SelectedMailbox

Imap.DoEvents - Removed. No longer used; call methods on a worker thread to not block the UI.

Imap.Editor - Removed. The included Imap Client sample may be used similarly.

Imap.EnableMailBoxNameEncoding - Imap.MailBoxNameEncodingEnabled

Imap.Idle – Now a method: Imap.Idle(). This method will block until the IDLE state is finished; the Update event (or Alert event for alerts) will be raised when unsolicited messages are received. Best used on a worker thread, as it would otherwise block UI message processing.

Imap.Security - Imap.Session.Security.Encrypt

Imap.Timeout - Imap.Connection.SocketOption.ReceiveTimeout. Imap.Connection.Socket.ReceiveTimeout is also available, but only when connected to a server.

Imap.UseAuthentication – Implement Imap.Session.Security.ValidationCallback. If UseAuthentication was false, return true. If UseAuthentication was true, implement your validation within the callback. See the 'Security Implementation' help topic in the help documentation for more information and example code.

Imap.UseMemoryStreams - Static bool Attachment.DecodeToMemory

Methods

Imap.Begin*() - Removed. Call Imap.*() on a worker thread. See the 'Designed for Modern Applications' help topic in the help documentation for more information and code examples.

Imap.Login() - Imap.Connect() and Imap.Authenticate():

Imap.Login(server, username, password) and Imap.Login(server, serverPort, client, clientPort, username, password):
(omit lines that contain unused parameters)

```
imap1.Session.RemoteEndPoint.HostNameOrAddress = server;
imap1.Session.RemoteEndPoint.Port = serverPort;
imap1.Session.LocalEndPoint.HostNameOrAddress = client;
imap1.Session.LocalEndPoint.Port = clientPort;
imap1.Session.Username = username;
imap1.Session.Password = password;
imap1.Connect();
imap1.Authenticate();
```

Imap.Logout() - Imap.Close()

Events

Imap.BusyChanged – Removed. See the Imap.Busy property in this guide for more information.

Imap.CertificateReceived – Imap.Session.Security.ValidationCallback. See the 'Security Implementation' help topic in the help documentation for more information and example code.

Imap.CertificateRequested – Imap.Session.Security.SelectionCallback. See the 'Security Implementation' help topic in the help documentation for more information and example code.

Imap.ConnectedChangedEx – Imap.Connection.StateChanged

Imap.End* - Removed. Begin* methods have been removed; methods should be called on a worker thread to make them non-blocking. Either call the code in these End* events after the blocking call on the worker thread, or move the code contained in these events into the UserState event if it interacts with the UI thread.

Imap.LineReceived – Removed. To view incoming data, use the Imap.Connection.Log event; when e.Data.Direction equals DataDirection.In, e.Data is data received from the server.

Imap.Mailbox – Removed. Examine the Mailbox array returned by Imap.List() instead.

Imap.Trace – Imap.Connection.Log

POP

Properties

Pop.AutoDelete – Removed. Set PopMessage.Deleted to true for any messages you want deleted on logout.

Pop.AutoGet – Removed.
MessageSection.None is the default behavior of the component.
MessageSection.Headers may be replicated by calling PopMessage.Get(0) on each PopMessage in Pop.Messages after Pop.Authenticate().
For MessageSection.Complete, do the same, but call PopMessage.Get() instead.

Pop.AutoLogout – Removed. Call Pop.Close() when finished.

Pop.AutoSize – Removed. Now the first parameter of Pop.Authenticate(getSizes, getUids).

Pop.AutoUid – Removed. Now the second parameter of Pop.Authenticate(getSizes, getUids).

Pop.Busy - Removed. Thread safety has been implemented internally for discrete commands; methods will be executed in the order they were called in. Use Lock (C#) or SyncLock (VB.NET) to ensure that sets of commands are executed before those on other threads.

Pop.Certficate - Removed. For most cases, just add a certificate to the Pop.Session.Security.Certificates collection. If the certificate does not pass Microsoft's validation checks, use Pop.Session.Security.SelectionCallback to force the use of the returned certificate. See the ‘Security Implementation’ help topic in the help documentation for more information and example code.

Pop.Connected – Pop.Connection.State

Pop.DoEvents - Removed. No longer used; call methods on a worker thread to not block the UI.

Pop.Editor – Removed. The included Pop Client sample may be used similarly.

Pop.Security - Pop.Session.Security.Encrypt

Pop.Size – Pop.Messages.Size

Pop.Timeout - Pop.Connection.SocketOption.ReceiveTimeout. Pop.Connection.Socket.ReceiveTimeout is also available, but only when connected to a server.

Pop.UseAuthentication - Implement Pop.Session.Security.ValidationCallback. If UseAuthentication was false, return true. If UseAuthentication was true, implement your validation within the callback. See the 'Security Implementation' help topic in the help documentation for more information and example code.

Pop.UseMemoryStreams - Static bool Attachment.DecodeToMemory

Methods

Pop.Begin*() – Removed. Call Pop.*() on a worker thread. See the 'Designed for Modern Applications' help topic in the help documentation for more information and code examples.

Pop.Login() - Pop.Connect() and Pop.Authenticate():

Pop.Login(server, username, password)
Pop.Login(server, username, password, loginMethod)
Pop.Login(server, serverPort, client, clientPort, username, password)
Pop.Login(server, serverPort, client, clientPort, username, password, loginMethod)
(omit lines that contain unused parameters)

```
pop1.Session.RemoteEndPoint.HostNameOrAddress = server;
pop1.Session.RemoteEndPoint.Port = serverPort;
pop1.Session.LocalEndPoint.HostNameOrAddress = client;
pop1.Session.LocalEndPoint.Port = clientPort;
pop1.Session.Username = username;
pop1.Session.Password = password;
pop1.Session.Authentication = loginMethod;
pop1.Connect();
pop1.Authenticate();
```

Pop.Logout() - Pop.Close()

Events

Pop.BusyChanged – Removed. See the Pop.Busy property in this guide for more information.

Pop.CertificateReceived – Pop.Session.Security.ValidationCallback. See the ‘Security Implementation’ help topic in the help documentation for more information and example code.

Pop.CertificateRequested – Pop.Session.Security.SelectionCallback. See the ‘Security Implementation’ help topic in the help documentation for more information and example code.

Pop.ConnectedChangedEx – Pop.Connection.StateChanged

Pop.End* - Removed. Begin* methods have been removed; methods should be called on a worker thread to make them non-blocking. Either call the code in these End* events after the blocking call on the worker thread, or move the code contained in these events into the UserState event if it interacts with the UI thread.

Pop.Trace – Pop.Connection.Log

SMTP

Properties

Smtp.Busy - Removed. Thread safety has been implemented internally for discrete commands; methods will be executed in the order they were called in. Use Lock (C#) or SyncLock (VB.NET) to ensure that sets of commands are executed before those on other threads.

Smtp.Certificate - Removed. For most cases, just add a certificate to the Smtp.Session.Security.Certificates collection. If the certificate does not pass Microsoft's validation checks, use Smtp.Session.Security.SelectionCallback to force the use of a specific certificate. See the ‘Security Implementation’ help topic in the help documentation for more information and example code.

Smtp.Client – Smtp.Session.LocalEndPoint.HostNameOrAddress

Smtp.ClientPort – Smtp.Session.LocalEndPoint.Port

Smtp.Connected – Smtp.Connection.State

Smtp.DnsServers – Removed. For manual DNS resolution, use the DNS class included in PowerTCP Sockets for .NET. Since the domain name is not used when connecting by IP, when using SSL, Smtp.Session.Security.TargetHost should be set for automatic certificate CN validation (when valid, sslPolicyErrors will not contain a value of RemoteCertificateNameMismatch).

Smtp.DoEvents - Removed. No longer used; call methods on a worker thread to not block the UI.

Smtp.DNS – Smtp.DeliveryStatusNotification.Options. May be configured further with Smtp.DeliveryStatusNotification.

Smtp.Editor - Removed. The included Smtp Client sample may be used similarly.

Smtp.LoginMethod – Smtp.Session.Authentication

Smtp.MailFrom – The ‘from’ parameter of Smtp.Send(message, from, recipients) or Smtp.Send(encodedMessage, from, recipients)

Smtp.Password – Smtp.Session.Password

Smtp.Recipients – The ‘recipients’ parameter of Smtp.Send(message, from, recipients) or Smtp.Send(encodedMessage, from, recipients)

Smtp.Security - Smtp.Session.Security.Encrypt

Smtp.Server – Smtp.Session.RemoteEndPoint.HostNameOrAddress

Smtp.ServerPort – Smtp.Session.RemoteEndPoint.Port

Smtp.Timeout - Smtp.Connection.SocketOption.ReceiveTimeout. Smtp.Connection.Socket.ReceiveTimeout is also available, but only when connected to a server.

Smtp.UseAuthentication - Implement Smtp.Session.Security.ValidationCallback. If UseAuthentication was false, return true. If UseAuthentication was true, implement your validation within the callback. See the ‘Security Implementation’ help topic in the help documentation for more information and example code.

Smtp.UsePipelining – Smtp.AutoPipeline

Smtp.Username – Smtp.Session.Username

Methods

Smtp.BeginSend() - Removed. Call Smtp.Send() on a worker thread. See the ‘Designed for Modern Applications’ help topic in the help documentation for more information and code examples.

Events

Smtp.BusyChanged – Removed. See the Smtp.Busy property in this guide for more information.

Smtp.CertificateReceived – Smtp.Session.Security.ValidationCallback. See the ‘Security Implementation’ help topic in the help documentation for more information and example code.

Smtp.CertificateRequested – Smtp.Session.Security.SelectionCallback. See the ‘Security Implementation’ help topic in the help documentation for more information and example code.

Smtp.ConnectedChangedEx – Smtp.Connection.StateChanged

Smtp.EndSend - Removed. Begin* methods have been removed; methods should be called on a worker thread to make them non-blocking. Either call the code in these End* events after the blocking call on the worker thread, or move the code contained in these events into the UserState event if it interacts with the UI thread.

Smtp.Trace – Smtp.Connection.Log

Misc

Enumerations

Certificate* - Removed. The .NET Framework is now used for certificate handling.

ContentEncoding – TransferEncoding

ContentEncoding.Yencode – Removed. Not supported.

ContentType – Functionally replaced with the .NET System.Net.Mime.ContentType class.

DSNType – Functionally replaced with the .NET System.Net.Mail.DeliveryNotificationOptions class. Configured on Dart.Mail.DeliveryStatusNotification.Options.

EncodingMethod – SecureEncoding

EncodingMethod.SignEncrypt – Call MailMessage.SecureSign(), then MailMessage.SecureEncrypt(). For more information on creating signed and encrypted emails, see RFC3851 section 3.6.

EncryptingMethod – EncryptingAlgorithm

HeaderLabelType – Removed. Strings are now used for header labels. The HeaderKey class contains a set of commonly used header labels for quick reference.

KeyUsage – Removed. The .NET Framework is used for certificate handling.

MessagePriority – Priority

MessageSection – Removed; Pop.AutoGet removed. See Pop.AutoGet in this guide for more information.

PopLoginMethod – Authentication

ProxyType.HttpConnect – ProxyType.Http

SecureProtocol – System.Security.Authentication.SslProtocols

Security – Encrypt

Security.ExplicitOptional – Removed. To replicate this functionality (demonstrated with Imap here):

```
imap1.Session.Security.Encrypt = Encrypt.Explicit;
imap1.Connect();
try
{
   imap1.Authenticate();
}
catch
{
   imap1.Session.Security.Encrypt = Encrypt.None;
   imap1.Authenticate();
}
```

SigningMethod – DigestAlgorithm

SigningMethod.Sha – DigestAlgorithm.Sha1

SmtpLoginMethod – Authentication

Classes

MessageStream – MailMessage. See the Code Examples topic in the help documentation for common usage scenarios.

MessageStream.ToMime() – MailMessage.SecureDecode()

MessageStream.ToSMime() – MailMessage.SecureEncrypt() or MailMessage.SecureSign(). See EncodingMethod.SignEncrypt in this guide for more information.

MessageStream.Store() – MailMessage.Save()