



docmail

Online mail management system

Web Service Help

Version 1.1

22nd June 2009

© 2009 CFH Total Document Management Limited
St Peters Park
Wells Road
Radstock
BA3 3UP



Ownership & Confidentiality

No part of this document may be disclosed orally or in writing, including by reproduction, to any third party without the prior written consent of CFH Total Document Management Limited. This document its associated appendices and any attachments remain the property of CFH Total Document Management Limited and shall be returned upon request.

docmail Web Service

Overview.....	4
Set up a docmail account to use the web service.....	4
Referencing the web service	5
Service Reference size limit	5
Service Reference timeout limit.....	5
Details of services and classes.....	6
SaveMailing service	6
Mailing class.....	6
Mail Pack class.....	7
Template class	8
Variable class	9
Mailing List class	9
Address class	10
Pre-formatted streams.....	10
MailingResult class.....	11
GetProof service.....	11
Proof class.....	11
PlaceOrder service	13
PlaceOrderResult class	13
DeleteMailing service	13
Error codes.....	14
Example code.....	15
Add a reference to the web service	15
Security	15
Creating a new mailing	16
Using saved mail pack and mailing list.....	16
Save a Mailing.....	16
Get a proof	16
Place an order	17
Validating an order	17
Delete a mailing.....	18
Full example code with validation.....	18

Getting file data to byte array	20
Template options	20
Address options.....	21
Full excel mailing list example	21
Adding addresses example	22
Proof file options.....	22
Variables example	23
Web references in VB .NET	24
Glossary of terms	25

Overview

The following help is for the docmail web service version 1.0.

There are four operations available from the web service:

SaveMailing
Saves a mailing.

GetProof
Get a proof object for a mailing, containing cost of order and a proof file (if requested).

PlaceOrder
Approve and pay for a mailing.

DeleteMailing
Delete a mailing, note that a mailing cannot be deleted once payment has been received.

If you are not familiar with the docmail system then please refer to the “Glossary of terms” at the back of the help guide, or read the docmail help guide on the website to familiarise yourselves with docmail terminology.

Please note the docmail web service does not currently support the greeting card functionality available though the docmail web site.

Set up a docmail account to use the web service

Using the docmail website (see “Referencing the web service” for the relevant URL) either create an account or login to an existing account. From the “My Account” screen select “My Users”, and click the “Add new user” button. Enter a login name and password for the web service user, ensuring that the “Can use web service” is checked, along with the other required permissions.

Note: In order to use the “SendToSelf” option when placing a mailing you must login to the docmail website with the new user and enter the address details.

Referencing the web service

The docmail API is a SOAP-based web service and in order to connect you will need the WSDL (service description) page URL, these can be found at:

For testing:

<https://www.cfhdocmail.com/TestAPI/DMWS.asmx>

The testing API has an associated test website, please use this website to set-up a test account and web service user:

<https://www.cfhdocmail.com/Test>

For live:

<https://www.cfhdocmail.com/LiveAPI/DMWS.asmx>

For live API use the main docmail website to set-up an account and web service user:

<https://www.cfhdocmail.com>

Service Reference size limit

By default .NET 2005 and 2008 will limit the amount of information that can be exchanged with the web service. In the binding section of the app.config or web.config increase the message size as required, or set to a large default as shown below:

```
maxBufferSize="2147483647" maxBufferPoolSize="2147483647"  
maxReceivedMessageSize="2147483647"
```

Service Reference timeout limit

By default .NET 2005 and 2008 will limit the time of a connection to 1 minute. This is sufficient for most operations, but for very large address lists or large files this may need increasing in the binding section of the app.config or web.config. The example below shows increasing the timeout to 5 minutes:

```
receiveTimeout="00:05:00" sendTimeout="00:05:00"
```

Details of services and classes

Please note that the description of class properties include only the properties which can be used with the web service, other properties may be listed in the class description, for example the "CreatedOn" date, but setting these properties that are not listed below will either have no affect on the mailing or may stop the mailing saving.

SaveMailing service

Takes a Mailing object and returns a MailingResult object.

Mailing class

The mailing class contains the following properties that can be set:

Property	Data Type / Max length	Description
AddressNameFormat	String (1)	Address name format, e.g. D = Full Name, F = Firstname Surname, S = Title Surname, T = Title FirstName Surname.
AddressPrefix	String (30)	Text to prefix addresses with, e.g. To the Parent/Guardian of.
DespatchDate	Date	Date the mailing is to be despatched, leave null for the earliest available despatch date.
DespatchCode	String (1)	Despatch code, e.g. 1 = Retail first class, 2 = standard Class
DiscountCode	String (20)	Discount code for savings on the mailing price.
GenerateNewProof	Boolean	If false docmail will use a previous proof for a save mail pack and template if one is available. Using a previous proof is the fastest option, but the output will not include address details from the current order.
IsBlackAndWhite	Boolean	Print the mailing in black and white? If false the mailing is produced in colour.
IsDuplex	Boolean	Print the mailing double-sided (Duplex)?
MailingDescription	String (50)	Description of the mailing.
MailingName	String (50)	The mailing name, defaults to the mailing reference is not supplied.
ProofFile	String (10)	Proof File, options are Yes / No / Use Previous. No by default. Use Previous returns a proof file from a previous mailing that used the same mail pack, if not available then a new proof will be generated. Use Previous always generates a new proof for mailings with variables.
SendToSelf	Boolean	Set to true to send a copy of the mailing to the current user. If set to True then DO NOT add the current user's address to the mailing list. Cannot be set to true if the mailing contains a template of type pre-formatted stream.

The mailing requires a mail pack and mailing list to be defined. There are a number of ways of defining these as outlined in the sections below.

Mail Pack class

The Mailing class has a number of options for adding mail pack data, select one of the following options for each mailing:

Option 1 – Use a stored mail pack

Property	Data Type / Max length	Description
MailPackName	String (25)	A mail pack is a saved collection of templates on the docmail website. To use a saved Mail Pack specify the mail pack name property.

Option 2 – Use a single stored template

Property	Data Type / Max length	Description
TemplateName	String (30)	Select from Template to use from stored templates. The stored template must be flagged as an addressed document.
TemplateVariables	Array	Array of Variables. Set the template variables to replace data in the template file.

Option 3 – Upload a single file

Property	Data Type / Max length	Description
TemplateFileData	Byte Array	File data for a single file for your mailing. This template will be treated as an address document.
TemplateFileName	String (255)	Set the name of the single file (must include file extension)
TemplateVariables	Array	Array of Variables. Set the template variables to replace data in the template file.
TemplateAddressFontCode	String (3)	Address Font Code. From the list in the docmail website take the first letter of font name and the font size number, e.g. Arial 10pt = A10. A10 by default.
TemplateBackgroundName	String (120)	Background name to use from saved backgrounds on the docmail website.
TemplateType	String (1)	“D” for a standard template Document, or “S” for a pre-formatted Stream file (a document containing content for multiple addresses).

Option 4 – Add multiple templates (stored or new files)

Property	Data Type / Max length	Description
Templates	Array	Array of template objects to include on the mailing

Tip:

When using the same file or set of files, use stored templates or mail packs wherever possible, as they not only save time on uploading file data but docmail can also use

the data already calculated for previous mailings. Stored templates and mail packs can be added via the docmail website.

Template class

There are 2 options for defining a template:

Option 1

Supply the template name only. The template will be copied from the library with all the saved properties. Please note that the properties of saved templates cannot be overwritten on an order. If the template is the first template in the template array it must stored with the “addressed letter” checkbox ticked (this equates to the template’s Addressed Document property being set to true).

Option 2

Create a new template including as a minimum the FileData and FileName.

For Templates of type “Document” (D), files must be of type .doc, .docx, .rtf or .pdf. If files are of type .pdf then they cannot contain any place holders. Place holders are replaced by data in the address list, e.g. “<<custom1>>” would be replaced by the text in the address object’s “Custom1” property; see the docmail help for a full explanation of place holder tags.

For templates of type “Pre-formatted Stream” (S) the file must be of type .pdf.

If this template is the first template in the template array it will automatically be classed as an addressed document.

Template class properties:

Property	Data Type / Max length	Description
TemplateName	String (30)	The name for the template, defaults to the file name if not supplied, if only the template name is supplied then copies from the store.
AddressedDocument	Boolean	Add an address to the first page of the template. Always true for the first template in a mailing.
AddressFontCode	String (3)	Address Font Code, e.g. from the list in the docmail website take the first letter of font name and font size, e.g. Arial 10pt = A10. A10 by default.
BackgroundName	String (120)	Background name to use from saved backgrounds on the docmail website.
Description	String (100)	Description of the template.
FileName	String (255)	Set the name of the single file (must include file extension)
FileData	Byte Array	File data for a single file for your mailing. This template will be treated as an address document.
PadAfter	Boolean	Stop another template from beginning on the back of this template? (Affects duplex printing only)
PadBefore	Boolean	Stop this template from beginning on the back of another template? (Affects duplex printing only)
TemplateType	String (1)	“D” for a standard template Document, or “S” for a

		pre-formatted Stream file (a PDF document containing content for multiple addresses).
Variables	Array	Array of Variables. Set the template variables to replace data in the template file.

Variable class

Using Templates can reduce the need to upload new documents for each mailing. Template files can contain customisable tags to be replaced by variables. The variable tags are replaced with the same text for all recipients, for example “<<Variable1>> Special Offers”, could become “Spring Special Offers” or “Summer Special Offers”. Variables can be used for single words or whole paragraphs of text.

Property	Data Type / Max length	Description
ConstantName	String (60)	The name for the variable, representing by a tag in the template file, e.g. <<ConstantName>>.
ConstantValue	String (2,147,483,647)	The text to replace the name tag in the document with.
MaximumLength	Long	The Maximum length of the constant value. Note: This property is designed for stored templates only, and there is no need to set this property for Mailing templates as it will automatically be set to the length of the ConstantValue property.

Mailing List class

The Mailing class has two options for adding mailing data, select one of the options for each mailing:

Option 1 – Mailing list file

If a mailing list file has headers these should match the column headers in the “Excel CSV template” on the docmail website (downloadable from the import address file screen).

Property	Data Type / Max length	Description
MailingListFileData	Byte Array	A file containing the mailing list, must be Excel or CSV format, if Excel the Sheet Name must be supplied.
MailingListFileName	String (35)	Mailing List File Name, must include file extension
MailingListFileSheetName	String (32)	If the MailingListFileData is of type Excel then specify the sheetname, defaults to “Sheet1”
MailingListFileNoColumnHeaders	Boolean	Is the mailing list supplied without column headers, if set to true the Field Mapping must be supplied.
MailingListFieldMappingName	String (50)	Select a stored Field Mapping to use

Option 2 – Add addresses

Property	Data Type / Max length	Description
Addresses	Array	Array of address objects to include on the mailing

Address class

Addressed must have either the full name supplied or the first name and surname along with Address line 1. The address class has the following properties that match the fields available on the Edit Addresses screen on the docmail website:

Property	Data Type / Max length
Address1	String (50)
Address2	String (50)
Address3	String (50)
Address4	String (50)
Address5	String (50)
CompanyName	String (50)
Custom1	String (50)
Custom2	String (50)
Custom3	String (50)
Custom4	String (50)
Custom5	String (50)
Custom6	String (50)
Custom7	String (50)
Custom8	String (50)
Custom9	String (50)
Custom10	String (50)
DirectLine	String (20)
Email	String (70)
ExtraInfo	String (50)
Facsimile	String (20)
FirstName	String (30)
FullName	String (100)
JobTitle	String (60)
Mobile	String (20)
Notes	String (100)
Surname	String (30)
Telephone	String (20)
Title	String (15)
StreamPages1	Integer
StreamPages2	Integer
StreamPages3	Integer

Pre-formatted streams

The Stream Pages fields for an address are used in conjunction with Template of type Pre-formatted Stream (S) stating how many pages from the file stream goes to the address.

Each address must have content from at least 1 template file, and the first template for the address must have the “AddressedDocument” flag set as true (this is true by

default for the first template in a mailing). For example if you have 2 stream files, then the StreamPages2 property could be set to 0 if the second stream only contains data for some of the addresses from the first stream.

MailingResult class

The MailingResult object is returned from the SaveMailing web service and includes the following properties:

Property	Data Type	Description
Success	Boolean	Was the proof generation successful?
FailureMessage	String	If proof was unsuccessful the failure message will include reasons why the proof failed.
MailingGUID	Unique Identifier	The unique identification for the mailing; requires passing through to the GetProof service.

When proofing an order, if a MailingGUID is returned but the success is false, then a mailing has been partially created in docmail. This happens if there is a problem saving new templates, mailing list file, or addresses. If you want to destroy the partially created order then call the DeleteOrder web service, alternatively you can go to the docmail website and review your order.

GetProof service

Takes a MailingGUID and a ReturnProofFile Boolean flag, returns a Proof object.

The MailingGUID is a property of the MailingResult class returned from the SaveMailing Service.

If the ReturnProofFile variable is set to true then proof PDF file will be returned in the ProofFileData property.

Proof class

The proof object is returned from the GetProof web service and includes the following properties:

Property	Data Type	Description
Success	Boolean	Was the get proof operation successful?
ProofReady	Boolean	Has docmail completed generating a proof? If this is false then no other proof data will be returned. Unless there is an error the Success flag will be true when the ProofReady flag is false.
FailureMessage	String	If a proof was unsuccessful the failure message will include reasons why the proof failed.

Property	Data Type	Description
ErrorCodeID	Integer	If a proof is unsuccessful the error will be less than 0. The error code ID will be 0 if successful.
ErrorCode	String	A description of the error code ID. The error code will be a zero length string if the error code ID is 0.
ReferencedFontList	String	A list of referenced fonts that are not embedded. NOTE: If there are fonts listed then docmail will substitute these fonts unless they are embedded in the document. It is therefore important to ensure that this list is blank or the proof file is reviewed. For details on how to embed fonts in a document please see the main docmail help document.
MailingName	String	The name of the mailing.
MailingGUID	Unique Identifier	The unique identification for the mailing; requires passing through to the Place Order function.
MailingPackSheets	Integer	The number of sheets for the mailing (per address), based on the sample proof (this may vary if the mailing contains pre-formatted stream templates).
MailingPackTemplates	Integer	The number of templates for the mailing.
MailingPrintColour	Boolean	Is the mailing to be printed in colour?
MailingPrintDuplex	Boolean	Is the mailing to be printed double-sided (duplex)?
MailingListTotalAddresses	Integer	The total number of addresses on the mailing.
MailingListTotalUKCheapAddresses	Integer	The number of addresses that qualify to the cheapest postal rates.
MailingListTotalUKSurchargeAddresses	Integer	The number of UK address that have a surcharge due to the address not being recognised (e.g. invalid or missing post code).
MailingListTotalOverseasAddresses	Integer	The number of overseas envelopes. Overseas envelopes are charged based upon weight.
MailingSendToSelf	Boolean	Is the current user included in the mailing?
MailingDespatchType	String	A description of the type of delivery specified by the despatch code on the mailing.
MailingDespatchDate	Date	When the mailing will be sent.
NetCost	Double	Total mailing cost excluding VAT.
VAT	Double	VAT charged for the mailing.
TotalCost	Double	Total mailing cost including VAT.
OrderRef	Long	The docmail order reference number.
ProofFileData	Byte Array	The proof file data in PDF format (if requested).
CustomChargeAmount	Double	Any custom charges applicable to the order, these are only added in discussion with the docmail support team for non-standard orders.
CustomChargeDescription	String	A description of the custom charge.
DocmailURL	String	An http address to view the order in the docmail website.

PlaceOrder service

Takes a Mailing GUID (from the proof object), and your purchase order reference (string max length 30), and returns a PlaceOrderResult object. The PlaceOrder service approves and pays for the order in docmail using existing top-up credit or account credit. Top-up credit has to be purchased via the docmail website before placing an order.

Note: If the user account has “can approve” permissions but not “can spend credit” then this call can be used to approve a mailing, but it will return as a failure due to not being able to spend credit.

PlaceOrderResult class

The web service returns a PlaceOrderResult class object which contains the following properties:

Property	Data Type	Description
Success	Boolean	Was the order placed successfully?
FailureMessage	String	Details of the failure if paying for the order was unsuccessful.
ErrorCodeID	Integer	If placing the order is unsuccessful the error will be less than 0. The error code ID will be 0 if successful.
ErrorCode	String	A description of the error code ID. The error code will be a zero length string if the error code ID is 0.
OrderRef	Long	The docmail Order Reference.
DocmailURL	String	HTTP address linking to the order in docmai.
BalanceMessage	String	Details of remaining balance.

DeleteMailing service

Takes a Mailing GUID and returns a string of “Success” if it succeeds or a failure message if it fails.

Error codes

The following error codes may be returned by docmail:

Error Code ID	Error Code	Description
-1	ValidationError	The validation of the data was unsuccessful.
-2	RecordNotFound	A requested record could not be found.
-3	ConcurrencyError	Data has been updated since the data was loaded, preventing the save from being able to be performed.
-4	NoRecordsAffected	An update or deletion of data affected no records; therefore the data may have already been deleted or an incorrect reference may have been supplied.
-5	UnexpectedError	A general failure has occurred; the failure message will include a unique error number for tracking a problem through our support team.
-6	PermissionDenied	The web service user does not have permission to perform an operation.
-21	PropertyValidationError	The validation of the data in a particular property was unsuccessful.
-22	PropertyRequired	Data for a required property was not supplied.
-23	PropertyDataType	Incorrect type of data was supplied.
-24	PropertyDataTypeLength	The numeric property length was outside the valid range.
-25	PropertyScale	Too many decimal places after the decimal point for the decimal property.
-26	PropertyPrecision	Too many numbers in a decimal property.
-27	PropertyMaxLength	The string property was over the maximum length.

Example code

The following code examples are for a VB.NET console application. For use within a web site or from adding a Web Service in .NET, please refer to the “Web references in VB .NET” section at the end of the example code.

Add a reference to the web service

Right click on your project and select “Add Service Reference”. *Please note that earlier versions of .NET may not have the “Add Service Reference” option available. If it is not available please follow the “Web references in VB .NET” section at the end of the example code.*

For live enter <https://www.cfhdocmail.com/LiveAPI/DMWS.asmx>
For test enter <https://www.cfhdocmail.com/TestAPI/DMWS.asmx>

Enter the namespace as “DMWS” if you want to follow the code examples.

Click OK to add the service reference.

Referencing the web service in code:

```
Dim oService As New DMWS.DMWSSoapClient()
```

Security

The user name and password are passed through the soap header. This information is kept secure through the SSL encryption.

```
Dim oServiceHeader As New DMWS.ServiceAuthHeader  
oServiceHeader.Username = "USERNAME"  
oServiceHeader.Password = "PASSWORD"
```

Please note that an account will be locked if the password is supplied incorrectly 10 times in a row. Please contact docmail support to get an account reactivated.

Creating a new mailing

```
Dim oMailing As New DMWS.Mailing()
```

Using saved mail pack and mailing list

```
oMailing.MailPackName = "Mail Pack Name"  
oMailing.MailingListName = "Mailing List Name"
```

Save a Mailing

```
Dim oMailingResult As DMWS.MailingResult  
oMailingResult = oService.SaveMailing(oServiceHeader, oMailing)
```

Get a proof

The generation of proofs and address validation process are set running when a mailing is saved. You can poll the GetProof service to see if the proof is available. Once it is available the ProofReady flag will return as true. Please ensure your loop has a wait time between each request of at least 1 second otherwise the request will fail. It is also advisable to add a timeout loop. The length of the timeout is dependent upon the size of the document being uploaded and the number of addresses required. The example below shows a timeout of 3 minutes.

```
Dim oMailingResult As DMWS.MailingResult  
oMailingResult = oService.SaveMailing(oServiceHeader, oMailing)  
  
' Get a proof, trying every second until 3 minutes have elapsed  
Dim oProof As DMWS.Proof  
oProof = oService.GetProof(oServiceHeader, _  
                           oMailingResult.MailingGUID, False)  
  
Dim dStart As Date = Date.Now ' Timeout variable  
  
While oProof.Success And _  
    oProof.ProofReady = False And _  
    DateDiff(DateInterval.Minute, dStart, Date.Now) < 3  
    System.Threading.Thread.Sleep(1000) ' Wait a second  
    oProof = oService.GetProof(oServiceHeader, _  
                              oMailingResult.MailingGUID, False)  
End While  
  
If oProof.Success And oProof.ProofReady = False Then  
    Console.WriteLine("Failed as proof not returned within 3 minutes.")  
End If
```

Place an order

If you pay on account then the last argument of PlaceOrder must contain your Purchase Order reference. If you normally pay by account but wish to pay for this order by credit card or top-up then you will need to approve and pay for the order via the docmail website.

```
Dim oPlaceOrderResult As DMWS.PlaceOrderResult
oPlaceOrderResult = oService.PlaceOrder(oServiceHeader, _
                                         oProof.MailingGUID, "")
```

Validating an order

The MailingResult, Proof and PlaceOrderResult classes have a "Success" boolean flag and a FailureMessage string property. The Proof object also contains the Referenced Font List, the results of address validation and costing information.

Example validation code:

```
If oProof.Success = True And oProof.ProofReady = True Then
    If oProof.ReferencedFontList <> "" Then
        Console.WriteLine("The document contains the following " & _
                           " fonts that require embedding: " & _
                           oProof.ReferencedFontList)
    End If

    If oProof.MailingListTotalOverseasAddresses > 0 Or _
        oProof.MailingListTotalUKSurchargeAddresses > 0 Then
        Console.WriteLine("Failed as we only want cheap postal codes")
    End If

    If oProof.MailingListTotalAddresses <> 100 Then
        Console.WriteLine("Failed as we were expecting 100 addresses")
    End If

    If oProof.TotalCost > 100 Then
        Console.WriteLine("Do not place order if over £100")
    End If

    ' Place an order using top-up credit
    Dim oPlaceOrderResult As DMWS.PlaceOrderResult
    oPlaceOrderResult = oService.PlaceOrder(oServiceHeader, _
                                             oProof.MailingGUID, "")

    If oPlaceOrderResult.Success = False Then
        Console.WriteLine(oPlaceOrderResult.FailureMessage)
    End If
Else
    ' Return failure message
    Console.WriteLine(oProof.FailureMessage)
End If
```

Delete a mailing

The delete mailing option is available for when a partial order is created and fails during adding templates and addresses, or during the payment. Once an order has been paid it cannot be deleted.

```
If Not oProof.MailingGUID.Equals(Guid.Empty) Then
    oService.DeleteMailing(oServiceHeader, _
        oProof.MailingGUID)
End If
```

Full example code with validation

```
' Referencing the web service
Dim oService As New DMWS.DMWSSoapClient()

' Security
Dim oServiceHeader As New DMWS.ServiceAuthHeader
oServiceHeader.Username = "USERNAME"
oServiceHeader.Password = "PASSWORD"

' Create mailing
Dim oMailing As New DMWS.Mailing()

' Use saved mail pack and mailing list
oMailing.MailPackName = "Mail Pack Name"
oMailing.MailingListName = "Mailing List Name"

' Save a mailing
Dim oMailingResult As DMWS.MailingResult
oMailingResult = oService.SaveMailing(oServiceHeader, oMailing)

' Get a proof, trying every second until 3 minutes have elapsed
Dim oProof As DMWS.Proof
oProof = oService.GetProof(oServiceHeader, _
    oMailingResult.MailingGUID, False)

Dim dStart As Date = Date.Now ' Timeout variable

While oProof.Success And _
    oProof.ProofReady = False And _
    DateDiff(DateInterval.Minute, dStart, Date.Now) < 3
    System.Threading.Thread.Sleep(1000) ' Wait a second
    oProof = oService.GetProof(oServiceHeader, _
        oMailingResult.MailingGUID, False)
End While

If oProof.Success And oProof.ProofReady = False Then
    Console.WriteLine("Failed as proof not returned within 3 minutes.")
End If

' Validate data
If oProof.Success = True And oProof.ProofReady = True Then
    If oProof.ReferencedFontList <> "" Then
        Console.WriteLine("The document contains the following " & _
            " fonts that require embedding: " & _
            oProof.ReferencedFontList)
```

```
End If

If oProof.MailingListTotalOverseasAddresses > 0 Or _
    oProof.MailingListTotalUKSurchargeAddresses > 0 Then
    Console.WriteLine("Failed as we only want cheap postal codes")
End If

If oProof.MailingListTotalAddresses <> 100 Then
    Console.WriteLine("Failed as we were expecting 100 addresses")
End If

If oProof.TotalCost > 100 Then
    Console.WriteLine("Do not place order if over £100")
End If

' Place an order using top-up credit
Dim oPlaceOrderResult As DMWS.PlaceOrderResult
oPlaceOrderResult = oService.PlaceOrder(oServiceHeader, _
    oProof.MailingGUID, "")

If oPlaceOrderResult.Success = False Then
    Console.WriteLine(oPlaceOrderResult.FailureMessage)
End If
Else
    ' Delete mailing if mailing GUID created
If Not oMailingResult.MailingGUID.Equals(Guid.Empty) Then
    oService.DeleteMailing(oServiceHeader, _
        oMailingResult.MailingGUID)
End If

    ' Return failure message
Console.WriteLine(oProof.FailureMessage)
End If
```

Getting file data to byte array

Template and Mailing List files require setting as a byte array. The following code is an example of loading a document into a byte array.

```
' Getting file data to byte array
oFileStream = System.IO.File.OpenRead("C:\Temp\Test.doc")
iBytes = CType(oFileStream.Length, Integer)
Dim oByteArray(iBytes - 1) As Byte
oFileStream.Read(oByteArray, 0, iBytes)
oFileStream.Close()
```

Template options

Template files can be in .doc, .docx, rtf, or pdf format.

Adding a single template

```
oMailing.TemplateFileName = "Test.doc"
oMailing.TemplateFileData = oByteArray

' Template name is optional,
' if left out will result to file name without file extension.
oMailing.TemplateName = "Test document"
```

Adding multiple templates

Note: A stored template can be used by just setting the Template Name property to that of an existing template and not including file data.

```
Dim oTemplate As New DMWS.Template()

' Template from file data
oTemplate.FileData = oByteArray
oTemplate.FileName = "Test.doc"

' Use stored template
Dim oTemplate2 As New DMWS.Template()
oTemplate2.TemplateName = "My saved template"

' Add template classes to the Mailing's Templates array
ReDim oMailing.Templates(1)
oMailing.Templates.SetValue(oTemplate, 0)
oMailing.Templates.SetValue(oTemplate2, 1)
```

Adding pre-formatted stream templates

Stream files must be of type "PDF". Set the Template Type property of the mailing or template to "S" for "Stream".

```
oMailing.TemplateType = "S"
or
oTemplate.TemplateType = "S"
```

Address options

Mailing List with valid column headers (combine with “Getting file data to byte array”)

```
oMailing.MailingListFileData = oByteArray
```

Specifying an Excel sheet (default is ‘Sheet1’)

```
oMailing.MailingListFileSheetName = "Sheet2"
```

Setting that the sheet does not have column headers

```
oMailing.MailingListFileNoColumnHeaders = True
```

If the sheet does not have column headers then a field mapping must be specified

```
oMailing.MailingListFieldMappingName = "Field Mapping Name"
```

Full excel mailing list example

```
' Referencing the web service
Dim oService As New DMWS.DMWSSoapClient()

' Security
Dim oServiceHeader As New DMWS.ServiceAuthHeader
oServiceHeader.Username = "USERNAME"
oServiceHeader.Password = "PASSWORD"

' Create mailing
Dim oMailing As New DMWS.Mailing()

' Mailing XLS, XLSX or CSV file
oMailing.MailingListFileName = "Test.xls"

' Getting file data to byte array
Dim oFileStream As System.IO.FileStream
Dim iBytes As Integer

oFileStream = System.IO.File.OpenRead("C:\Test.xls")
iBytes = CType(oFileStream.Length, Integer)
Dim oByteArray(iBytes - 1) As Byte
oFileStream.Read(oByteArray, 0, iBytes)
oFileStream.Close()

' Set the file data
oMailing.MailingListFileData = oByteArray

' Specify the sheet name if an Excel file and not 'Sheet1'
oMailing.MailingListFileSheetName = "Sheet2"

' If file does not have headers set No Column Headers to true
oMailing.MailingListFileNoColumnHeaders = True

' Specify a saved field mapping
oMailing.MailingListFieldMappingName = "Field Mapping Name"
```

Adding addresses example

If you wish to add address data manually rather than in a CSV or Excel file, then address objects can be created and added to the addresses array as follows:

```
Dim oAddress As New DMWS.Address()  
oAddress.FullName = "CFH docmail"  
oAddress.Address1 = "St Peter's Park"  
oAddress.Address2 = "Wells Road"  
oAddress.Address3 = "Radstock"  
oAddress.Address4 = "BA3 3UP"  
  
' Add addresses class to the mailing's Addresses array  
ReDim oMailing.Addresses(0)  
oMailing.Addresses.SetValue(oAddress, 0)
```

Proof file options

Generate a new proof from the supplied address details:

```
oMailing.GenerateNewProof = True
```

Use a previous proof if available, if there is a previous proof then this will be used and the address data in the proof may not be for the current order. Note: Setting this option to false will only make a difference when a mailing contains only stored mail packs and templates, and the stored templates do not contain variables:

```
oMailing.GenerateNewProof = False
```

To return a proof file as a byte array, ensure the ReturnProofFile flag is set to true when calling the GetProof service:

```
oProof = oService.GetProof(oServiceHeader, _  
                           oMailingResult.MailingGUID, True)
```

Saving a returned proof PDF file to disk:

```
Dim oFileStream As New System.IO.FileStream("C:\Temp\Test.pdf", _  
System.IO.FileMode.Create)  
oFileStream.Write(oProof.ProofFileData, 0,  
oProof.ProofFileData.Length)  
oFileStream.Close()  
oFileStream.Dispose()' Note: Dispose not available for .NET 1
```

Variables example

Template files can contain customisable tags to be replaced by variables. Template Variables can be added to a Template's "Variables" property or the Mailing's "TemplateVariables" property. The following example will replace the tag "<<Variable1>>" with the text "Replaced Example Text":

```
' Creating a variable
Dim oVariable1 As New DMWS.Variable()
oVariable1.ConstantName = "Variable1"
oVariable1.ConstantValue = "Replaced Example Text"

' Using a single template on the mailing:
ReDim oMailing.TemplateVariables(0)
oMailing.TemplateVariables.SetValue(oVariable1, 0)

' For a template in the template array:
ReDim oTemplate.Variables(0)
oTemplate.Variables.SetValue(oVariable1, 0)
```

Web references in VB .NET

Right click on your project and select “Add Web Reference”.

For live enter <https://cfhdocmail.com/LiveAPI/DMWS.asmx>

For test enter <https://cfhdocmail.com/TestAPI/DMWS.asmx>

Click on the Go button, enter the web reference name as “DMWS” if you want to follow the code examples, and then click the “Add Reference” button.

When using the web service within a website the declaration code is slightly different, with the Service Header object only needing to be declared once and not passed into the web service calls.

```
' Web service reference example
Dim oServiceHeader As DMWS.ServiceAuthHeader
Dim oDMWS As DMWS.DMWS
Dim oMailing As DMWS.Mailing
Dim oMailingResult As DMWS.MailingResult
Dim oProof As DMWS.Proof
Dim oPlaceOrderResult As DMWS.PlaceOrderResult

oServiceHeader = New DMWS.ServiceAuthHeader
oServiceHeader.Username = txtUserID.Text
oServiceHeader.Password = txtPassword.Text

oDMWS = New DMWS.DMWS
oDMWS.ServiceAuthHeaderValue = oServiceHeader

oMailing = New DMWS.Mailing()
oMailing.MailPackName = " Mail Pack Name "
oMailing.MailingListName = "Mailing List Name"

oMailingResult = oDMWS.SaveMailing(oMailing)

If oMailingResult.Success Then
    oProof = oDMWS.GetProof(oMailingResult.MailingGUID, False)

    Dim dStart As Date = Date.Now ' Timeout variable

    While oProof.Success And _
        oProof.ProofReady = False And _
        DateDiff(DateInterval.Minute, dStart, Date.Now) < 3
        System.Threading.Thread.Sleep(1000) ' Wait a second
        oProof = oDMWS.GetProof(oMailingResult.MailingGUID, False)
    End While

    If oProof.Success And oProof.ProofReady Then
        oPlaceOrderResult = oDMWS.PlaceOrder(oProof.MailingGUID, _
            "")
    End If
End If
```

Glossary of terms

Address

The data relating to an individual person in the mailing. This includes their name, address and any additional information to be merged in a Template.

Addressed Document

A template that contains a space for the address to be added. Please refer to the main docmail help guide for details of the address positioning.

Background

A background is a PDF file that appears behind text entered in a main document, for example a letter head.

Mailing

One of more templates that require sending to one or more addresses.

Mailing List

The list of addresses that will receive the Mail Pack. A mailing list can be saved via the docmail website to be used over and over again.

Mail Pack

The collection of templates that are to be used in the mailing. Think of a mail pack as a paperclip, holding different templates that you may want to use together. Each mail pack will therefore always contain at least one template, and possibly more. A mail pack can be saved via the docmail website to be used over and over again.

Template

A document file that will be sent to mailing list.

Template Type "Document" (D)

A template that can include place holders that will be replaced by data in the address list. This template will be sent to each address in the address list.

Template Type "Pre-formatted Stream" (S)

A PDF file containing pre-formatted data for multiple addresses. The only changes to template stream documents are the adding of addresses.

Variables

A customisable tag in the template document that can be replaced with the same text for all recipients, for example <<Variable1>> Special Offers, could become "Spring Special Offers", or "Summer Special Offers". Variables can be used for single words or whole paragraphs of text.