DocuSign SOAP API Developer Guide

DocuSign API Developer Guide June 3, 2016

If you have any comments or feedback on our documentation, please send them to us at: Documentation@DocuSign.com.

Summary of changes for this version:

- Added tabOrder property to the Tab element that allows senders to set the order in which tabs are shown to users on a page when signing. Note that this option is not available for all account plans.
- Update the note for the CorrectAndResendEnvelope method to say if automatic reminders are enabled for the envelope, then resending the envelope resets the automatic reminder date and the resend date is used for determining when to send reminder messages.

Table of Contents

DocuSign SOAP API Introduction

The SOAP API Developer's Guide provides information about the DocuSign Service API and Account Management Service API.

DocuSign Security Requirements and API Call Limits

To ensure our customers continue to trust DocuSign for the fastest, easiest, most secure way to get a signature, we have put safeguards in place to protect our multi-tenant data centers. The two safeguards used by DocuSign are Integrator Keys and API Call Limits.

Integrator Keys

DocuSign has introduced Integrator Keys to identify third party applications. The use of Integrator Keys is mandatory for API calls to our production system (www.docusign.net) and for developer sandboxes (demo.docusign.net).

The Integrator Key is used to allow you to send requests with the users passed via the UsernameToken. Users with DocuSign privileges will be allowed to be passed in the UsernameToken as long as a valid integrator key is provided.

> **Important:** Web service calls made without an Integrator Key will receive an exception for every call made. The exception message states, "The specified Integrator Key was not found or is disabled" (error number 3).

Integrator Keys are provided to API developers by DocuSign. If you do not have an Integrator Key, follow this procedure to obtain one:

1. Log on to your demo account.
2. In the DocuSign Console menu bar, click **Preferences**. The Account Preferences page appears.
3. Scroll down and under Account Administration click **API**.
4. Create a new key:
   - Below the Active Integrator Keys table, type a Key Description.
   - Click **Request Key** adjacent to the bolded key information. This key is added to the list of Active Integrator Keys table.
5. Add the Integrator Key to your code for use as described below in Integrator Key Usage and in the Authentication section. The API page also has examples of how to use an Integrator Key. You can test your Integrator Key in the demo environment.
6. When you are ready to certify your code for the production account, return to the API page and review the Integration and Certification Steps. Click the **DocuSign Developer Center** link and find the link to start the DocuSign Certification process. A message with an access code is sent to your listed email address. Follow the instructions to access the envelope and fill out the information.
7. After starting the DocuSign Certification process, you will need to request migration of your Integrator Key to the Production environment. In the Active Integrator Keys table, find the key you are using in your application. Click the **Request Migration to Production** link adjacent to that Integrator Key.

Integrator Key Usage

The integrator key must be placed in front of the user ID that is in the Username node of the UsernameToken. The integrator key must be wrapped with brackets, “[and]”. Example format:

<wsse:Username>**[Integrator Key]**2988541c-4ec7-4245-b520-f2d324062ca3</wsse:Username>

Sample SOAP Header:

```
   <soap:Header>
     <wsa:Action>http://www.docusign.net/API/3.0/GetRecipientEsignList</wsa:Action>
      <wsa:MessageID>uuid:3f9d7626-c088-43b4-b579-2bd5e8026b17</wsa:MessageID>
      <wsa:ReplyTo>

<wsa:Address>http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous</wsa:Address
>
      </wsa:ReplyTo>
      <wsa:To>http://demo.docusign.net/api/3.0/api.asmx</wsa:To>
      <wsse:Security soap:mustUnderstand="1">
        <wsu:Timestamp wsu:Id="Timestamp-8838aa24-9759-4f85-8bf2-26539e14f750">
          <wsu:Created>2006-04-14T14:29:23Z</wsu:Created>
          <wsu:Expires>2006-04-14T14:34:23Z</wsu:Expires>
        </wsu:Timestamp>
        <wsse:UsernameToken xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-utility-1.0.xsd" wsu:Id="SecurityToken-7c7b695e-cef7-463b-b05a-
9e133ea43c41">
          <wsse:Username>[Integrator Key Here]2988541c-4ec7-4245-b520-
f2d324062ca3</wsse:Username>
          <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-1.0#PasswordText">password</wsse:Password>
          <wsse:Nonce>SjlScsL5q3cC1CDWrcMx3A==</wsse:Nonce>
          <wsu:Created>2006-04-14T14:29:23Z</wsu:Created>
        </wsse:UsernameToken>
      </wsse:Security>
    </soap:Header>
```

API Call Rate Limits

To maintain reliability and stability within our demo and production environments, DocuSign operates with certain API call efficiency guidelines. To ensure effective load balance we continually monitor the API calls to our backend systems and we will contact developers that are putting unnecessary burden on the system.

DocuSign has implemented the following API Call Rate Limits to balance loads on the system:

- The demo environment (demo.docusign.net) is limited to a call rate of 1,000 API calls per hour per account.
- The production environment (www.docusign.net) is limited to a call rate of 1,000 API calls per hour per account.

If the API call rate limit is reached, you will receive an exception for each call until the start of the next hour (this can be up to 60 minutes). The exception message states, “The maximum number of hourly API invocations has been exceeded” (error number 207).

Hourly API usage is tracked from the start of one-hour to start of the next hour.

Customers can monitor the number of API calls their application uses by checking the API Limit Headers that are returned in the response to SOAP API requests. The headers are returned in both the production and demo environments.

The headers are only returned in the response to SOAP API requests for calls that are subject to DocuSign API limits. So the API limit headers will not be returned in all the API responses and your code should take this into consideration.

Instances where the API limit headers are not returned include:

- When an error is detected early in an API request. Usually this due to errors in the request body or invalid query string parameters.
- API calls that historically have not been counted as part of the DocuSign legacy API limits strategy.
- Any API calls that do not require user authentication and do not identify a specific account user.

For instances where the API limit headers are not returned, there is no change to the remaining calls available for the account.

The information in the API limit header shows the current API limit for the account (the number of calls that can be made per hour), the number of remaining calls for the current time period, and when the account API limit will reset. The reset time is shown in Unix epoch time.

Example API Limit Headers

```
X-RateLimit-Limit → 1000
X-RateLimit-Remaining → 996
X-RateLimit-Reset → 1425967200
```

For polling compliance DocuSign recommends that you do not exceed 1 status request per unique envelope per 15 minutes for the following methods:

RequestStatus, RequestStatusEx, RequestStatuses, RequestStatusesEx, RequestPDF and RequestDocumentPDFs.

There are a number of ways to minimize API impact, such as:

- using bulk operations for requesting status,
- utilizing DocuSign's event notification feature,
- and refraining from repeatedly requesting information on envelopes that are in terminal state (Completed, Declined or Voided).

If you find your application still requires more than 1,000 calls per hour per account, please contact service@docusign.com for assistance in working on a solution.

If you have any questions, please check our Developer Forum.

Authentication

All DocuSign API methods require authentication. There are two ways to pass member credentials:

- SOAP Header via WS-Security UsernameToken
- HTTP Header via a custom field "X-DocuSign-Authentication"

The Account Management API only supports the HTTP Header authentication method, while all others can support either method.

Additionally, the DocuSign API has two API end points: API.asmx and DSAPI.asmx. The API.asmx end point requires the WS-Security UsernameToken in the SOAP header authentication. The DSAPI.asmx and AccountManagement.asmx end points require the HTTP Header authentication method.

HTTP Header Authentication

For the HTTP header, access to the API must be enabled for the member login that is being used. This is controlled by combinations of user name, password and Integration Key. The valid authentication combinations are:

- User Name, password, Integrator Key

  Note: this is the preferred option since is provides the highest level of protection.

  Where the User Name is the API User Name retrieved from the Credential Login() function, the password is the encrypted password retrieved from the AccountManagement GetEncryptedPassword() function, and the Integrator Key is the key provided by DocuSign.

- User Name, password, Integrator Key

  Where the User Name is the API User Name retrieved from the Credential Login() function, the password is in clear text, and the Integrator Key is the key provided by DocuSign.

- User Name, password, Integrator Key

  Where the User Name is the user's email login, the password is in clear text, and the Integrator Key is the key provided by DocuSign.

Note that when using the HTTP Header form of authentication, the header variable name is: X-DocuSign-Authentication.

One way to provide the HTTP authentication header is to create a subclass from the Web Service and override the GetWebRequest method, as shown in the following example.

Example HTTP Header – C#

```
namespace DSAPI
{
    // override of web service interface is required to insert the HTTP header
authentication.
    public class DocuSignAcctMgmtService :
DSAPI_AcctMgmtWebService.AccountManagementService
    {
        private string myDSUserId = "";
        private string myDSPassword = "";
        private string myDSIntegratorKey = "";


        public string UserName
        {
            get { return myDSUserId; }
            set { myDSUserId = value; }
        }

        public string Password
        {
            get { return myDSPassword; }
            set { myDSPassword = value; }
        }

        public string IntegratorKey
```

```
        {
            get { return myDSIntegratorKey; }
            set { myDSIntegratorKey = value; }
        }
        protected override System.Net.WebRequest GetWebRequest(Uri uri)
        {

            System.Net.HttpWebRequest r = base.GetWebRequest(uri) as
System.Net.HttpWebRequest;
            r.Headers.Add("X-DocuSign-Authentication",
                string.Format("<DocuSignCredentials><Username>YOUR USER EMAIL OR ID GOES
HERE</Username><Password>YOUR USER PASSWORD GOES HERE</Password><IntegratorKey>YOUR
INTEGRATION KEY GOES HERE</IntegratorKey></DocuSignCredentials>",
                UserName,
                Password,
                IntegratorKey));

            return r;
        }
}
```

SOAP Header Authentication

For the WS-Security UsernameToken, the values for the UsernameToken elements can be the same as those used for the HTTP header values.

Example UsernameToken

```
<wsse:Security soap:mustUnderstand="1">
  <wsu:Timestamp wsu:Id="Timestamp-0741d0e0-529f-49bc-bf86-653238d2532b">
  <wsu:Created>2006-01-02T21:26:04Z</wsu:Created>
  <wsu:Expires>2006-01-02T21:31:04Z</wsu:Expires>
  </wsu:Timestamp>
  <wsse:UsernameToken xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd" wsu:Id="SecurityToken-8d4e766e-a8a2-4bb3-a327-89c34bc7f85f">
    <wsse:Username>caa26663-927b-4800-bfdf-d115d1c72f20</wsse:Username>
    <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-1.0#PasswordText">password</wsse:Password>
    <wsse:Nonce>RumCR4p6U4a7hiX9lUlGWA==</wsse:Nonce>
    <wsu:Created>2006-01-02T21:26:04Z</wsu:Created>
  </wsse:UsernameToken>
</wsse:Security>
```

Example HTTP Header – PHP

```
$am_api_endpoint = "https://demo.docusign.net/api/3.0/accountmanagement.asmx";
$am_api_wsdl = "api/AccountManagementService.wsdl";

// build credential xml to add to http header
$ds_auth = "<DocuSignCredentials><Username>" . $UserID . "</Username><Password>"
    . "$Password</Password><IntegratorKey>" . $IntegratorsKey .
     "</IntegratorKey></DocuSignCredentials>";
$ctxStream = stream_context_create(array(
    'http' => array(
        'method' => "GET",
        'header' => "X-DocuSign-Authentication: " . $ds_auth . "\r\n")));

$am_api_options =  array(
    'protocol_version' => "1.0",
    'trace'=>true,
    'stream_context' => $ctxStream);
$am_api = new AccountManagementService($am_api_wsdl, $am_api_options);
```

Optional Authentication Mechanism: XML Signature

In addition to the UsernameToken, any account may elect to enforce that all API requests be signed with a valid third-party X.509 certificate 2. For any RequestRecipientToken and ExportAuthoritativeCopy calls DocuSign requires an XML signature. If enabled, DocuSign Connect API will validate that the SOAP Body of the message is signed. Please contact your DocuSign representative to enable this setting. Supported certificate authorities are VeriSign and Thawte.

For more information about OASIS standards, see:

http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss.

Example SOAP Envelope with XML Signature:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">
    <soap:Header>
     <wsa:Action>http://www.docusign.net/API/3.0/GetRecipientEsignList</wsa:Action>
      <wsa:MessageID>uuid:3f9d7626-c088-43b4-b579-2bd5e8026b17</wsa:MessageID>
      <wsa:ReplyTo>

<wsa:Address>http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous</wsa:Address
>
      </wsa:ReplyTo>
      <wsa:To>http://demo.docusign.net/api/3.0/api.asmx</wsa:To>
      <wsse:Security soap:mustUnderstand="1">
        <wsu:Timestamp wsu:Id="Timestamp-8838aa24-9759-4f85-8bf2-26539e14f750">
          <wsu:Created>2006-04-14T14:29:23Z</wsu:Created>
          <wsu:Expires>2006-04-14T14:34:23Z</wsu:Expires>
        </wsu:Timestamp>
        <wsse:UsernameToken xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-utility-1.0.xsd" wsu:Id="SecurityToken-7c7b695e-cef7-463b-b05a-
9e133ea43c41">
          <wsse:Username>2988541c-4ec7-4245-b520-f2d324062ca3</wsse:Username>
          <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-1.0#PasswordText">password</wsse:Password>
          <wsse:Nonce>SjlScsL5q3cC1CDWrcMx3A==</wsse:Nonce>
          <wsu:Created>2006-04-14T14:29:23Z</wsu:Created>
        </wsse:UsernameToken>
```

```
        <wsse:BinarySecurityToken ValueType="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3"
EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-
security-1.0#Base64Binary" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-utility-1.0.xsd" wsu:Id="SecurityToken-b7825bf5-1d1c-446c-8f99-
88dff075fab8">MIIEdDCCAlygAwIBAgIDAgIOMA0GCSqGSIb3DQEBBQUAMHkxEDAOBgNVBAoTB1Jvb3QgQ0ExHjA
cBgNVBAsTFWh0dHA6Ly93d3cuY2FjZXJ0Lm9yZzEiMCAGA1UEAxMZQ0EgQ2VydCBTaWduaW5nIEF1dGhvcml0eTEh
MB8GCSqGSIb3DQEJARYSc3VwcG9ydEBjYWNlcnQub3JnMB4XDTA2MDMwNzE3NTU0NVoXDTA2MDkwMzE3NTU0NVowJ
jEkMCIGA1UEAxMbZGVtb3NhbXBsZWNlcnQuZG9jdXNpZ24uY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQ
DhgGA7HXHcotoKuVTN5+3yqeL1A4M/bYNLKyFFQYYFANVXcb7D+la2Oh8YI2TBHTgNGccJxeCsefZh1LHLguE4/kM
Mvz62mbdohhnjXeJH55O4YqdILxZ3r1EJbOrSZyHEwn1/PvGwj2cDF0QHnvqfgvsuozlJmRYNKXpnD9QzzwIDAQAB
o4HbMIHYMAwGA1UdEwEB/wQCMAAwNAYDVR0lBC0wKwYIKwYBBQUHAwIGCCsGAQUFBwMBBglghkgBhvhCBAEGCisGA
QQBgjcKAwMwCwYDVR0PBAQDAgWgMDIGCCsGAQUFBwEBBCYwJDAiBggrBgEFBQcwAYYWaHR0cDovL29jc3AuY2FjZX
J0Lm9yZzBRBgNVHREESjBIghtkZW1vc2FtcGxlY2VydC5kb2N1c2lnbi5jb22gKQYIKwYBBQUHCAWgHQwbZGVtb3N
hbXBsZWNlcnQuZG9jdXNpZ24uY29tMA0GCSqGSIb3DQEBBQUAA4ICAQBzgTyhXvDMmt6UwB6ZR8qVGa4Jhch68kS0
X5vQjMa0wXdfJjU3C13rnNujX8wefPOrX79vyS4CVVXW3QRTUp+hyaodwi2ed0msz0vo71VUU6KGUNp3GClY5NjMz
mz9y900SjL74shWiDsbhT/yMGDp8cZt6nFSUabhaBM5NFRvguwI1hAjugT6QnLjeUxblVuS2s2b90Kj+jh/w3gw5f
/0XCxgNBdz66jvxFxk0wKl3xLoLs/a2nh4TOvEdUixZEPvsJyQNFY/+5cmlVM6/Nl/T2TjMcEuvhpBvizDvNykluE
di0R5vftkURlE/InmmGt37eH3xB3YeyFDR7uY6qJREBswNWKd8xqK7cb4XXntY1L5XEIdW/5ZNn3JdAg2Vq05lHF5
+i/i+5Uc4GDcQlxmTbiSvt4z0tobnHUsrsrBbLRlcmzruZbX5OKgJ/r1hO3JTEhFzR5KCogDfsCdAqCC+ClKc0GPS
Sgt6Cu36F5ytE04KTXBltU/Ex3kDDIlm7OX/os7kSIZf9IAejr4Cxx1wdenjy//n89qLjbd0u11e038g41/orQiBY
n7opj3wmbPdRZ+gsMpIxhhnmUMFWMIJMIidzZwRa9n3E+l49ZjElCDuBRnUoaGDhmSnEYJrtv/uaQ5UulvMtMge7F
rqiIDzI1A++nBfGRMS0EHLdfBdg==</wsse:BinarySecurityToken>
        <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
            <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#" xmlns:ds="http://www.w3.org/2000/09/xmldsig#" />
            <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
            <Reference URI="#Id-6c2377a1-a116-444c-8c1e-101543d5d721">
              <Transforms>
                <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
              </Transforms>
              <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
              <DigestValue>auFLuvdCM5tbsrruRJUY7h0vf9E=</DigestValue>
            </Reference>
          </SignedInfo>

<SignatureValue>d21v1CT2UxIus64mXbfCxO6xjJ3gbm+cE14HUnlOJmA9QsB5M5L1ric4cvMcOky0hFUDZrhDn
5FVUXtLKJQgV5wqTtmeHi3NT6HNEh1Rrq/VFj/O4/rFGyc4JrzkVxZpqiwRYOee1Empv6iHq3Uf4PIeAv1Tn/qn/b
09P+D7FTo=</SignatureValue>
          <KeyInfo>
            <wsse:SecurityTokenReference>
              <wsse:Reference URI="#SecurityToken-b7825bf5-1d1c-446c-8f99-88dff075fab8"
ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-
1.0#X509v3" />
            </wsse:SecurityTokenReference>
          </KeyInfo>
        </Signature>
      </wsse:Security>
    </soap:Header>
    <soap:Body wsu:Id="Id-6c2377a1-a116-444c-8c1e-101543d5d721">
      <GetRecipientEsignList xmlns="http://www.docusign.net/API/2.0">
        <UserName>UserName</UserName>
        <SenderEmail>username@email.com</SenderEmail>
        <RecipientEmail>name@email.com</RecipientEmail>
    <AccountId>2184872a-8f6c-4f18-b808-1ec864cec29d</AccountId>
      </GetRecipientEsignList>
    </soap:Body>
  </soap:Envelope>
```

Send On Behalf Of Functionality

The DocuSign *Send On Behalf Of* (SOBO) functionality allows a single user in an account to authenticate for other members of the account. When originally conceived, it was meant for sending fully-defined envelopes only. Since then, most other API functions have been allowed - making this really an "authenticate on behalf of" function instead of just a sending function.

This authentication is especially useful for CRM and other integrations where the API client also maintains an account-with-multiple-members structure similar to DocuSign. In this scenario, having a single user authenticate with DocuSign relieves the API integration from knowing and storing individual user passwords within the client application.

The SOBO user, also known as the authenticating user, authenticates with DocuSign on behalf of an operating user. Once authentication is passed, the operating user's user and member settings are used for all authorization functions.

The SOBO functionality is only available to accounts that use the DocuSign API to send envelopes. To enable SOBO for a user the user's Account-Wide Rights and Send On Behalf Of Rights (API) Permission settings must be enabled. This can be done by a DocuSign administrator through the Classic DocuSign Experience web application or by contacting your DocuSign Account Manager.

If user permissions are being set through the Classic DocuSign Experience web application, the listed settings correspond to the Account-Wide Rights and Send On Behalf Of Rights (API) settings.

The DocuSign SOAP API SOBO function can use the normal email authentication for the DocuSign API. The SOBO identifier can be a properly formatted email address, with an option semicolon delimited user name, or a User ID (UID), if using the Single Sign On (SSO) environment, that can be looked up in the DocuSign SSO configuration.

- If the identifier is a properly formatted email address, the system conducts a look-up of the email address and user name (if provided) to see if that user is a member of the account. If membership in the account is not found, a Partner Authentication Failed exception is thrown.

  The account used for the check is the one associated with the account member's login credentials (User Name email address and password).

- If the identifier is not an email address (the identifier fails the regular expression test for an email), it is assumed that the identifier is a UID. A search is conducted for the UID in the Single Sign On (SSO) customer system and the email address and user name associated with the UID are retrieved from the system. Then the system conducts a look-up of the email address and user name to see if that user is a member of the account. If membership in the account is not found, a Partner Authentication Failed exception is thrown.

  The account used for the Single Sign On check is the one associated with the account member's login credentials (User Name email address and password).

Functions Supported and Not Supported by Send On Behalf Of

The following DocuSign SOAP API calls do not support SOBO:

- Account Management API's (AccountManagement.asmx) functions that authenticate using Distributor Code and Distributor Password (ActivateSalesforceInstance, ChangeAccountPricePlan, GetPlanGroupInformation, GetPlanPricingInformation, GetSuccessorPlanInformation, NewSocialAccount, NewAccount, and ResendAccountActivation) are not supported by SOBO.
- Credential API's (Credential.asmx) functions (Login, Ping, GetAuthenticationToken, and RequestSenderToken) are not supported by SOBO.

All other API.ASMX/DSAPI.ASMX functions (i.e. API calls that require username/password authentication support) are supported by SOBO authentication.

Send On Behalf Of SOAP Examples

The examples in this section show the different ways the SOBO identifier is used for authentication in a SOAP Header. The most common use for SOBO is with one of the DocuSign SOAP API methods for envelope sending methods, but for this example a RequestStatus method is used for brevity.

These examples use an Integrator Key and Send On Behalf Of identifier, both included in separate brackets in the Email or Username objects. The Send On Behalf Of identifier is highlighted in the examples.

Example: Authentication in SOAP Header with Email Address

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Header>
    <wsse:UsernameToken xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-utility-1.0.xsd" wsu:Id="SecurityToken-7c7b695e-cef7-463b-b05a-
9e133ea43c41">
      <wsse:Username>[BILL- 9048-469a-a9e9-211cef79e5f3][john.doe@docusign.com] 2988541c-
4ec7-4245-b520-f2d324062ca3</wsse:Username>
      <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-1.0#PasswordText">password</wsse:Password>
    </wsse:UsernameToken>
  </soap:Header>
  <soap:Body>
    <RequestStatus xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>YYYYYYYY</EnvelopeID>
    </RequestStatus>
  </soap:Body>
</soap:Envelope>
```

Example: Authentication in SOAP Header with Email Address and optional User Name

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Header>
    <wsse:UsernameToken xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-utility-1.0.xsd" wsu:Id="SecurityToken-7c7b695e-cef7-463b-b05a-
9e133ea43c41">
      <wsse:Username>[BILL- 9048-469a-a9e9-211cef79e5f3][john.doe@docusign.com;John Doe]
2988541c-4ec7-4245-b520-f2d324062ca3</wsse:Username>
      <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-1.0#PasswordText">password</wsse:Password>
    </wsse:UsernameToken>
  </soap:Header>
  <soap:Body>
    <RequestStatus xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>YYYYYYYY</EnvelopeID>
    </RequestStatus>
  </soap:Body>
</soap:Envelope>
```

Example: Authentication in SOAP Header with SSO UID

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Header>
```

```
    <wsse:UsernameToken xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-utility-1.0.xsd" wsu:Id="SecurityToken-7c7b695e-cef7-463b-b05a-
9e133ea43c41">
      <wsse:Username>[BILL-9048-469a-a9e9-211cef79e5f3][P134325] 2988541c-4ec7-4245-b520-
f2d324062ca3</wsse:Username>
      <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-1.0#PasswordText">password</wsse:Password>
    </wsse:UsernameToken>
  </soap:Header>
  <soap:Body>
    <RequestStatus xmlns=”http://www.docusign.net/API/3.0”>
      <EnvelopeID>YYYYYYYY</EnvelopeID>
    </RequestStatus>
  </soap:Body>
</soap:Envelope>
```

Using Encrypted Passwords in SOAP

This section provides information on requesting and using a DocuSign encrypted password in the SOAP api.asmx API where the wsse security header is used for authentication.

1. To start the process, use the Credential Login API to find the accounts associated with the user's credentials. This initial call uses the email address and clear text password.

Request

```
POST https://test.docusign.net/api/3.0/credential.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: 458
SOAPAction: "http://www.docusign.net/API/Credential/Login"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Login xmlns="http://www.docusign.net/API/Credential">
      <Email>r.tester@gmail.com</Email>
      <Password>passWordabC2</Password>
      <ReturnBaseUrl>true</ReturnBaseUrl>
    </Login>
  </soap:Body>
</soap:Envelope>
```

Response

```
HTTP/1.1 200 OK
Cache-Control: private, max-age=0
Content-Length: 1895
Content-Type: text/xml; charset=utf-8
Server: Microsoft-IIS/7.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Fri, 11 Oct 2013 13:45:10 GMT

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">
```

```
  <soap:Header>
    <wsa:Action>http://www.docusign.net/API/Credential/LoginResponse</wsa:Action>
    <wsa:MessageID>urn:uuid:6b00bce5-5196-4bf4-bae6-350a3f8c1596</wsa:MessageID>
    <wsa:RelatesTo>urn:uuid:ec87cca1-dd40-48ce-a681-0ea7f3b6076c</wsa:RelatesTo>
    <wsa:To>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
</wsa:To>
    <wsse:Security>
      <wsu:Timestamp wsu:Id="Timestamp-f3df2bf7-9dab-48dd-809e-e07ffb19d801">
        <wsu:Created>2013-10-11T13:45:10Z</wsu:Created>
        <wsu:Expires>2013-10-11T13:50:10Z</wsu:Expires>
      </wsu:Timestamp>
    </wsse:Security>
  </soap:Header>
  <soap:Body>
    <LoginResponse xmlns="http://www.docusign.net/API/Credential">
      <LoginResult>
        <Success>true</Success>
        <ErrorCode>Success</ErrorCode>
        <AuthenticationMessage>Successful authentication</AuthenticationMessage>
        <Accounts>
          <Account>
            <AccountID>1d270976-75ac-4e7b-9ff8-a746b6881f0d</AccountID>
            <AccountName>Rest Tester Account</AccountName>
            <UserID>47c97f72-3f01-4ec4-96f2-75ac3b630bfa</UserID>
            <UserName>Resty Tester</UserName>
            <Email>r.tester@gmail.com</Email>
            <BaseUrl>https://test.docusign.net/api/3.0</BaseUrl>
          </Account>
          <Account>
            <AccountID>db08fde2-740f-4af3-92fa-bc611de2f525</AccountID>
            <AccountName>Rest Account Test2</AccountName>
            <UserID>3100c524-ece7-44a2-bea2-688dcaba9d5d</UserID>
            <UserName>Resty Tester2</UserName>
            <Email>r.tester@gmail.com</Email>
            <BaseUrl>https://test2.docusign.net/api/3.0</BaseUrl>
          </Account>
        </Accounts>
      </LoginResult>
    </LoginResponse>
  </soap:Body>
</soap:Envelope>
```

The response shows that this user has access to two accounts and the BaseUrl shows that one account is on “test” while the second account is on “test2”. In future API calls, you will select the server and endpoint as appropriate for the account you will use. For the example we will use:

Account = `https://test.docusign.net/api/3.0`

UserId = `47c97f72-3f01-4ec4-96f2-75ac3b630bfa`

Note: The email address can be used instead of the UserId in API calls.

2. Next, request the encrypted Password for the user by making a GetEncryptedPassword request to the server where this user’s account is located and providing the credentials in the X-DocuSign-Authentication header.

Request

```
POST https://test.docusign.net/api/3.0/accountmanagement.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetEncryptedPassword"
```

```
X-DocuSign-Authentication:
<DocuSignCredentials><Username>r.tester@gmail.com</Username><Password>passWordabC2</Passw
ord><IntegratorKey>DOCU-KEY-ABC2</IntegratorKey></DocuSignCredentials>
Content-Length: 342

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetEncryptedPassword xmlns="http://www.docusign.net/API/AccountManagement" />
  </soap:Body>
</soap:Envelope>
```

Response

```
HTTP/1.1 200 OK
Cache-Control: private, max-age=0
Content-Length: 1329
Content-Type: text/xml; charset=utf-8
Server: Microsoft-IIS/7.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Fri, 11 Oct 2013 13:37:56 GMT

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">
  <soap:Header>
    <wsa:Action>http://www.docusign.net/API/AccountManagement/
GetEncryptedPasswordResponse</wsa:Action>
    <wsa:MessageID>urn:uuid:ce8dc29c-c2b6-46e4-b474-b15b1cf6523d</wsa:MessageID>
    <wsa:RelatesTo>urn:uuid:d763a3d1-de91-474a-800c-34987b858f6e</wsa:RelatesTo>
    <wsa:To>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/
anonymous</wsa:To>
    <wsse:Security>
      <wsu:Timestamp wsu:Id="Timestamp-48eac7b1-7d1c-4034-a7cf-92e55010579e">
        <wsu:Created>2013-10-11T13:37:56Z</wsu:Created>
        <wsu:Expires>2013-10-11T13:42:56Z</wsu:Expires>
      </wsu:Timestamp>
    </wsse:Security>
  </soap:Header>
  <soap:Body>
    <GetEncryptedPasswordResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <GetEncryptedPasswordResult>
        <EncryptedPassword>SBXBCNHn29VRB8e8cxsMes9LZuk=</EncryptedPassword>
        <Success>true</Success>
      </GetEncryptedPasswordResult>
    </GetEncryptedPasswordResponse>
  </soap:Body>
</soap:Envelope>
```

The response returns the encrypted version of the password, `SBXBCNHn29VRB8e8cxsMes9LZuk=`.

At this point, if client application (securely) stores the BaseUrl, UserId and encrypted password, then the application does not need to repeat the previous calls unless authentication fails on successive calls.

3. The following request shows how the encrypted password is used, along with the Integrator Key and UserId, to authenticate a Request Status call for an envelope. This same pattern would be used in all successive API calls.

   In this case the Integrator Key is `[DOCU-KEY-ABC2]`.

Request

```
POST https://test.docusign.net/api/3.0/api.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
SOAPAction: "http://www.docusign.net/API/3.0/RequestStatus"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">
  <soap:Header>
    <wsa:Action>http://www.docusign.net/API/3.0/RequestStatus</wsa:Action>
    <wsa:MessageID>urn:uuid:be6a8f4a-c4d9-4ae3-a79d-496f207c5c27</wsa:MessageID>
    <wsa:ReplyTo>
      <wsa:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
</wsa:Address>
    </wsa:ReplyTo>
    <wsa:To>https://test.docusign.net/API/3.0/api.asmx</wsa:To>
    <wsse:Security soap:mustUnderstand="1">
      <wsu:Timestamp wsu:Id="Timestamp-8af36607-1d5b-457f-a954-e849cd186c93">
        <wsu:Created>2013-10-11T13:34:18Z</wsu:Created>
        <wsu:Expires>2013-10-11T13:39:18Z</wsu:Expires>
      </wsu:Timestamp>
      <wsse:UsernameToken xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-utility-1.0.xsd" wsu:Id="SecurityToken-289a0ed4-5a95-4653-a800-
f64c692384ae">
        <wsse:Username>[DOCU-KEY-ABC2]47c97f72-3f01-4ec4-96f2-
75ac3b630bfa</wsse:Username>
        <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-1.0#PasswordText">SBXBCNHn29VRB8e8cxsMes9LZuk=</wsse:Password>
        <wsse:Nonce>STapZaoS6MuKXmdh2TU1Qw==</wsse:Nonce>
        <wsu:Created>2013-10-11T13:34:18Z</wsu:Created>
      </wsse:UsernameToken>
    </wsse:Security>
  </soap:Header>
  <soap:Body>
    <RequestStatus xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>D65860E3-D2DF-40D4-8FEF-4F37ABDED5AA</EnvelopeID>
    </RequestStatus>
  </soap:Body>
</soap:Envelope>
```

SOAP DSAPI.ASMX and API.ASMX Endpoints

DocuSign has two SOAP API service endpoints: API.ASMX and DSAPI.ASMX. DocuSign recommends using the DSAPI.ASMX endpoint for your integrations.

Note: The additional endpoints for credential and account administration are covered later in this guide.

The main advantage of the DSAPI.ASMX endpoint is that it performs authentication using the X-DocuSign-Authentication header instead of the WSE3 UsernameToken. This authentication is more straightforward for a caller to implement than the WSE UsernameToken authentication.

The DSAPI.ASMX endpoint is a layered endpoint on top of the API.ASMX endpoint and, calls the API.ASMX WebService public functions after the caller has been authenticated. The DSAPI.ASMX supports X509 message signing, similar to the API.ASMX endpoint.

The DSAPI.ASMX endpoint does not have the following methods that exist in the API.ASMX:

- GetRecipientAuthenticationStatusEx
- RequestEnvelopeHistoryToken

Additionally, the DSAPI.ASMX endpoint does not perform schema validation of input XML. Validation is performed as required for various fields within business logic. The difference in field validation accounts for the differences in behavior between using the API.ASMX and DSAPI.ASMX. The API.ASMX endpoint uses of up-front schema validation, which causes different error messages to be shown to the API caller than the business logic. In some cases (such as the Email subject and blurb) the DSAPI.ASMX endpoint just truncates the fields when they are committed to the database, while the API.ASMX endpoint's schema validation would return an error based on schema validation.

Using DocuSign WSDLs

This section provides information on using Web Services Description Language (WSDL) to implement the DocuSign Service API and Account Management Service API in a .Net or Salesforce environment.

The URLs for the DocuSign Service API and Account Management Service API WSDLs are:

- Service API - https://www.docusign.net/api/3.0/schema/dsapi.wsdl.
- Account Management Service API - https://www.docusign.net/api/3.0/schema/dsapi-accountmanagement.wsdl.

The endpoints locations for these are:

- Service API - https://www.docusign.net/api/3.0/dsapi.asmx
- Account Management Service API - https://www.docusign.net/api/3.0/accountmanagement.asmx

Using DocuSign WSDL's in the .Net Environment

The WSDL files may be used to generate Web Reference proxy classes for use in the .Net environment. Authentication is provided in the HTTP header for functions that require it. The standard method to accomplish this is to create a Web Reference to the WSDL file and wrap the resulting proxy class to provide the HTTP header.

Creating a Web Reference in Visual Studio 2010 for WSDL files:

1. Create a project.
2. In Solution Explorer, right-click on the project and select **Add Service Reference**.

3. In the Address, type the URL to the WSDL file (for example https://demo.docusign.net/api/3.0/dsapi.asmx).
4. Type a Namespace (for example DocuSignDemo) and click **OK**.

HTTP Header:

When you make a call to DocuSign, you need to add an HTTP header to your code. For example:

```
     String auth = "<DocuSignCredentials><Username>" + userName
     + "</Username><Password>" + password
     + "</Password><IntegratorKey>" + integratorKey
     + "</IntegratorKey></DocuSignCredentials>";

     DSAPIServiceSoapClient client = new DSAPIServiceSoapClient();
     using (OperationContextScope scope = new
System.ServiceModel.OperationContextScope(client.InnerChannel))
     {

          HttpRequestMessageProperty httpRequestProperty = new
HttpRequestMessageProperty();
          httpRequestProperty.Headers.Add("X-DocuSign-Authentication", auth);

OperationContext.Current.OutgoingMessageProperties[HttpRequestMessageProperty.Name] =
httpRequestProperty;

          EnvelopeStatus status = client.RequestStatusEx("D3151108-FC4C-4D1A-A168-
86E5233AACDB");
          Console.Out.WriteLine("Subject: " + status.Subject);
     }
```

Using DocuSign WSDL's in the Salesforce Environment

Salesforce provides a mechanism to import WSDL files and create Apex proxy classes. The steps are:

1. Create local copies of the new WSDL files by opening a web browser to each WSDL file (e.g. https://www.docusign.net/API/3.0/Schema/dsapi.wsdl and saving them locally.
2. Navigate to **Develop→Apex Classes** and select the "**Generate from WSDL**" option, the follow the instructions to create proxy classes.

Providing the Authentication Header

Proxy classes generated from WSDL's in Apex include a mechanism to add headers. Before making calls on the API, the DocuSign Authentication header must be added. For example:

```
       DSAPI_Status.APIServiceSoap dsApiStatus = new DSAPI_Status.APIServiceSoap();

   //Setting docusign authorization.
   dsApiStatus.inputHttpHeaders_x = new Map<String, String>();
   dsApiStatus.inputHttpHeaders_x.put('X-DocuSign-Authentication',
            '<DocuSignCredentials><Username>578a282b-9263-4fbe-8c2f-
52ab919da96e</Username><Password>1234567</Password><IntegratorKey>TEST_KEY</IntegratorKey
></DocuSignCredentials>');

   // Make an API call
   DSAPI_Status.EnvelopeStatuses  = dsApiStatus.RequestStatus('7BDF80CA-9CA8-4911-9629-
92DC40A1A34A');
```

Note that although hard-coded here, the user name, password and Integrator Key would normally be stored as variables.

DocuSign Time Track Header

The DocuSign Time Track header is an optional line that can be added to your API request header to return processing times for your requests.

When the X-DocuSignTimeTrack header is included in the API request, DocuSign returns the header in the response with the start-time and the end-time of the associated API request.

For example, when the Time Track header is included in the request:

```
X-DocuSignTimeTrack
```

The response includes the header and times in the response:

```
X-DocuSign-TimeTrack: SOAP_Start,2016-01-05T22:30:52.2522730Z;SOAP_End,2016-01-
05T22:30:52.3182730Z
```

Additionally you must add your own value on the request and that information is preserved in the response.

For example, you could add the time your application starts a process (such as MySendApp,<start time>) and then check the round-trip network time for an action.

Request:

```
X-DocuSignTimeTrack: MySendApp,2016-01-05T22:30:52.2522666Z
```

Response:

```
X-DocuSign-TimeTrack: MySendApp,2016-01-05T22:30:52.2522666Z;SOAP_Start,2016-01-
05T22:30:52.2522730Z;SOAP_End,2016-01-05T22:30:52.3182730Z
```

DocuSign Service API

This section of the Developer's Guide provides information about the DocuSign Service API.

Overview

The DocuSign Service API provides methods that allow partner companies' servers to integrate the DocuSign service into their applications. The service allows partners to build solutions that:

- Submit partially specified envelopes for later completion by the customer.
- Submit completely specified envelopes that are immediately processed for delivery.
- Void an envelope that has been submitted but not yet completed.
- Retrieve the status of an envelope.
- Retrieve the completed PDF of every document in an envelope.
- Retrieve the completed PDF for each separate document in an envelope.
- Retrieve the Electronic Record and Signature Disclosure acceptance status of a recipient.
- Transfer an envelope to another DocuSign user or account.
- Correct recipient information for an existing envelope en route.
- Resend a notification email to an existing recipient.
- Retrieve the Member level permissions for the optional features.
- Purge the envelope contents from the DocuSign system.
- Withdraw an Authoritative Copy of the envelope.
- Retrieve a list of audit events pertaining to a particular envelope.
- Work with address books.
- Upload and download templates.

These methods can be used by themselves or in addition to linking the customer's experience to the DocuSign site to complete any partially completed processes.

Commonly Used Terms

Definitions of some commonly used terms are given here to familiarize the API user with their use in the DocuSign system.

- **Envelope** - This represents a package used to mail documents to recipients. The envelope carries information about the sender and timestamps to indicate the progress of the delivery procedure. It contains collections of Documents, Tabs and Recipients.
- **Document** - A document that is to be delivered, representing the content to be reviewed and/or signed. Documents have names and are always base64 encoded while in the system.
- **Tab** - This represents a DocuSign Tag (also known as a Stick-eTab®) on a document. It is used in several ways. First, it is used to indicate to a recipient where a signature or initials are required. Second, it is used to include various bits of information in a document in a manner similar to Form Fields or Macros. For example, a tab may automatically fill in the Company

Name of a recipient when the document is signed. Third, it is used as editable information fields where signers can add data to a document.

- **Recipient** - Someone who receives the envelope and, optionally signs and initials the documents where indicated by tabs.

Code Samples

Code samples appear throughout this guide to assist API users in implementation. A few notes about the code samples:

- APIServiceSoapClient is configured and set up correctly. To do this, the API user must construct credentials as follows:

C# Setup Code Sample

```
String _userName = "";
String _apiUrl = "https://demo.docusign.net/api/3.0/api.asmx";
String _accountId = "Your API account ID";
String _password = "Your account password";
String _email = "Your login email";
String _integratorKey = "Your integrator key";
if (_integratorKey != null && _integratorKey.Length > 0)
{
    _userName += "[" + _integratorKey + "]";
}
_userName += _email;

DocuSignWeb.APIServiceSoapClient _apiClient =
     new DocuSignWeb.APIServiceSoapClient("APIServiceSoap", _apiUrl);

_apiClient.ClientCredentials.UserName.UserName = _userName;
_apiClient.ClientCredentials.UserName.Password = _password;
```

PHP Setup Code Sample

```
//  credential api service proxy classes and soapclient
include("api/CredentialService.php");
// transaction api service proxy classes and soapclient
include("api/APIService.php");

// TODO: Use Integrator's Key from Docusign DevCenter Account Preferences API
$IntegratorsKey = "your integrator key GUID here";
// TODO: Use your Docusign DevCenter Account email
$UserID = "your login email here";
// TODO: Use your Docusign DevCenter Account password
$Password = "your password here";
// TODO: Use API Account ID from Docusign DevCenter Account Preferences API
$AccountID = "your api account GUID here";
// TODO: put in your timezone or make it null
$TimeZone = 'America/Los_Angeles';

//================================================================================
// Set up the API
//================================================================================
$api_endpoint = "https://demo.docusign.net/api/3.0/api.asmx";
$api_wsdl = "api/APIService.wsdl";
$api_options =  array('location'=>$api_endpoint,'trace'=>true,'features' =>
SOAP_SINGLE_ELEMENT_ARRAYS);
$api = new APIService($api_wsdl, $api_options);
$api->setCredentials("[" . $IntegratorsKey . "]" . $UserID, $Password);
```

- Several of the code samples contain links to previous code samples. The referred code samples set up variables (such as an envelope or an envelope ID) that are necessary for the sample to demonstrate a method.
- Similar to the code samples above, the code samples in this document will often contain placeholder text or variables. Take note of where parameters need correction specific to a scenario or account.
- **PHP Helper Functions:** The following functions are used in some of the PHP samples in this document and are included here to inform the developer what they do.

```
/**
 * Returns xsd format datetime for start of today
 * @return string
 */
function todayXsdDate() {
    global $TimeZone;
    if ($TimeZone != null) {
        date_default_timezone_set($TimeZone);
    }
    return (date("Y") . "-" . date("m") . "-" . date("d") . "T00:00:00.00");
}

/**
 * Returns xsd format datetime for now
 * @return string
 */
function nowXsdDate() {
    global $TimeZone;
    if ($TimeZone != null) {
        date_default_timezone_set($TimeZone);
    }
        return (date("Y") . "-" . date("m") . "-" . date("d") . "T" . date("H") . ":" .
date("i") . ":" . date("s"));
}

/**
 * A guid maker for all seasons (note that com_create_guid() only works on windows
 * @return string
 */
function guid(){
    if (function_exists('com_create_guid')){
        return com_create_guid();
    }else{
        mt_srand((double)microtime()*10000);//optional for php 4.2.0 and up.
        $charid = strtoupper(md5(uniqid(rand(), true)));
        $hyphen = chr(45);// "-"
        $uuid = chr(123)// "{"
                .substr($charid, 0, 8).$hyphen
                .substr($charid, 8, 4).$hyphen
                .substr($charid,12, 4).$hyphen
                .substr($charid,16, 4).$hyphen
                .substr($charid,20,12)
                .chr(125);// "}"
        return $uuid;
    }
}
```

Basic Process Flow

This is a general overview of how the DocuSign system works to familiarize API users with terms and process. It is not a faithful reproduction of how the process is implemented, either on the website or in the API.

A sender has a set of documents that they would like to have signed. The sender supplies the name and email address of each person they want to sign the document, and marks the documents with tabs to indicate where each party should sign or initial. The sender may also choose to let the receiving party free-form sign the document. The sender then places the document into the DocuSign system. The DocuSign system then notifies each recipient, via the supplied email address(es), that they have been asked to sign a document, and provides a link to the envelope.

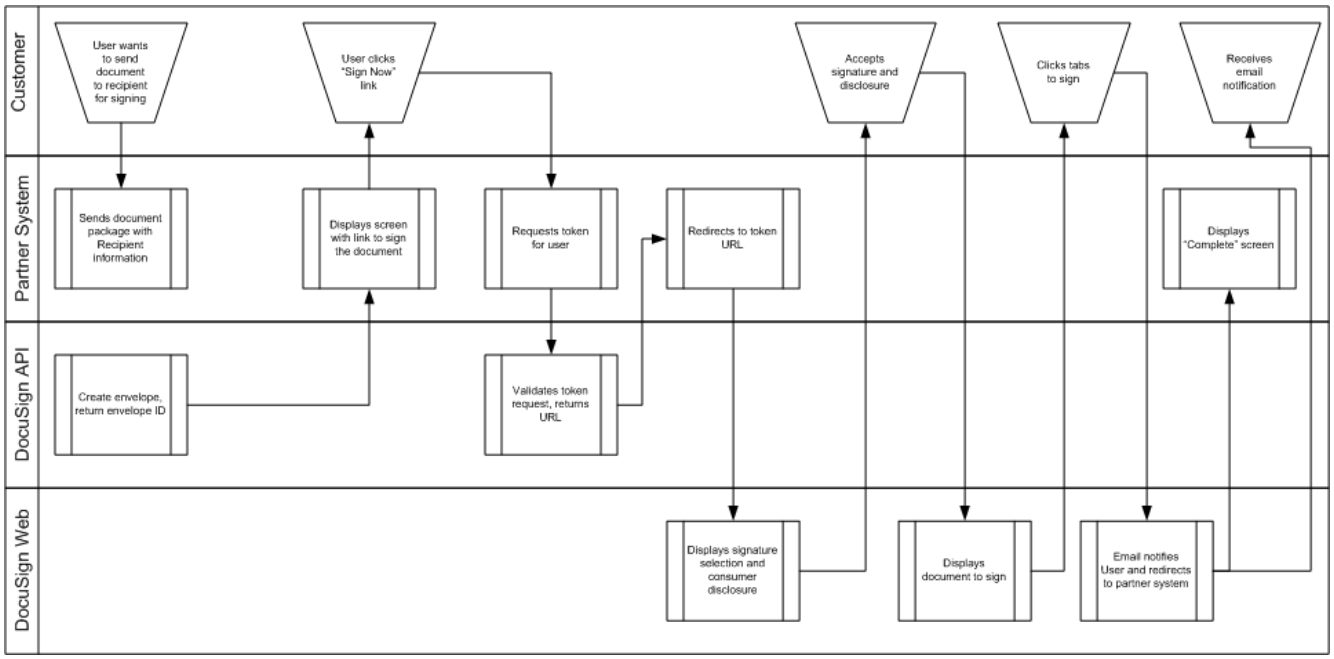
When a recipient clicks the link, they will see the documents with DocuSign Tags (Stick-eTabs) in the locations specified by the sender, representing where they need to click to sign or initial the document. If the sender chooses to use free-form signing the recipient will see the document with a note letting them know to click on the document to sign.

When all recipients have clicked on all of their respective DocuSign Tags (Stick-eTabs) or added their signature via free-form signing, the document is signed.

When this process is initiated via the DocuSign Connect API, the envelope can either be completely specified, which is to say that all Tags (or free-form signing), documents and recipients have been either supplied in the request, or partially specified, meaning that at least one document is present, but recipients and DocuSign Tags may not be. This second case is used to create a 'draft' envelope for the sender and allows him or her to finish sending it online through the web interface.

Example Usage Diagram

The following diagram depicts In-session signing process flow. The diagram shows one of the basic process and meant only to explain how different systems interact with each other.



Placing DocuSign Tags

The location of DocuSign Tags, also known as Stick-eTabs or Tabs, can be specified in one of two ways:

- Passing the explicit x/y coordinates of each tab in the xml when creating the Envelope. This is appropriate for documents with a standard format, where the signature lines and/or other data items collected by DocuSign tabs are always located in the same place on the document. In this scenario, the xml request describes *where* to locate DocuSign tabs.
- Specifying "anchor" strings that exist in the document, against which the tabs should be placed. This is appropriate for documents that have variable content/format, where the tab locations must similarly vary. In this scenario, the xml request describes *how* to locate DocuSign tabs.

File Limitations

DocuSign has the following limitations on files used in envelopes and as attachments:

- DocuSign recommends that you do not add files larger than 25MB to an envelope. Note that, depending on the recipient's internet connection, large files might affect signing performance.
- DocuSign has not imposed a limit on the number of files that can be added to an envelope. However, as with file size, envelopes with a large number of files might affect signing performance.
- For signer-uploaded attachment files, DocuSign supports files sizes up to 25MB for an envelope.
- There is a file size limit of 5MB for attaching completed documents to emails sent by DocuSign to recipients when an envelope is completed. If the completed documents are greater than 5MB, the email still provides a link to the documents on the DocuSign system.

Methods Exposed in the DocuSign Service API

The DocuSign API exposes the following major methods (listed alphabetically):

- **AcknowledgeAuthoritativeCopyExport** – Returns the key to decrypt the documents returned in the ExportAuthoritativeCopy method. Removes the Authoritative Copy from DocuSign. Available only in the 3.0 API.
- **CorrectAndResendEnvelope** - Correct the specified recipients of the envelope, along with envelope reminders and expirations. Allows a sender to modify the name and/or email address of a recipient, change the envelope access authentication information for a recipient, or resend the envelope notification email to a recipient. Supports multiple corrections within an envelope. The return value has a Boolean CorrectionSucceeded for each correction, which indicates the success of the recipient correction.
- **CreateAccountBrands** – Used to upload one or more brand profile files to the account. The Account Branding feature must be enabled for the account to use this.
- **CreateAndSendEnvelope** - Creates the envelope and initiates the delivery process. This method requires an envelope that includes all necessary information; no sender interaction via the DocuSign website is required.
- **CreateEnvelope** – Creates the envelope without sending it. These envelopes do not contain all of the information necessary for immediate processing. Instead, the sender can be directed to the DocuSign web site to complete the preparation of the envelope.

- **CreateEnvelopeFromTemplates** – Creates and sends envelope based on DocuSign Pro templates and envelope information. Available only in the 3.0 API.
- **CreateEnvelopeFromTemplatesAndForms** – Creates envelopes from a combination of a PDF form and a DocuSign Template. Once all the data from the form is overlaid on the template the envelope is passed to the *CreateAndSendEnvelope* or *CreateEnvelope* method.
- **DeleteAccountBrands** – Used to delete one or more brand profiles from an account. The Account Branding feature must be enabled for the account to use this.
- **DeleteCaptiveRecipientsSignatures** - This method deletes the signature for one or more captive recipient records; it is primarily used for testing. This provides a way to reset the signature associated with a ClientUserId so a new signature can be created the next time the ClientUserId is used.
- **DeleteEnvelopes** - Deletes the specified envelopes from the DocuSign System.
- **EnvelopeAuditEvents** - Returns a XML document of the current envelope events. Available only in the 3.0 API.
- **ExportAuthoritativeCopy** – Export an Authoritative Copy envelope from DocuSign. Returns an encrypted collection for all the documents and an export transaction ID for a given envelope. To decrypt the documents and remove the actual Authoritative Copy from DocuSign the method AcknowledgeAuthoritativeCopyExport must be called. Available only in the 3.0 API.
- **GetAccountBrands** – Used to retrieve a list of brand profiles associated with the account and the default brand profile. The Account Branding feature must be enabled for the account to use this.
- **GetAccountMembershipFeaturesList** – Returns the member level permissions for the optional features. The optional features include DocuSign Professional, eOriginal Vaulting, SequentialSigningAPI, SequentialSigningUI and TransactionPoint.
- **GetAccountSettingsList** – Returns the settings for an account.
- **GetAddressBookItems** – These methods are used to manage your server side address book in the DocuSign system.
- **GetAuthenticationToken** – This method is used to get a onetime use URL with an authentication token to launch the DocuSign member system.
- **GetConnectFailures** – This method retrieves a list of Connect post failures for the account and date range specified.
- **GetFolderList** and **GetFolderItems** – The GetFolderList and GetFolderItem methods are used to retrieve the list of folders, including shared folders, and envelopes in the folders. Using these calls is a more efficient way to get a list of envelopes than using RequestStatuses to request individual envelope statuses.
- **GetRecipientAuthenticationStatusEx** – Returns the authentication status of a recipient for the specified envelope. The response returns the applicable authentication results with the authentication status, a date time stamp and authentication failure details (if provided).
- **GetRecipientEsignList** – Return a collection of RecipientEsignRecords for the supplied Sender UserName, Email address and the recipient Email address. The purpose of this function is to allow a sender to determine if an Esign agreement already exists between the sender (as identified by the UserName and Email address) and the recipient (identified by email address only). Each RecipientEsignRecord returned (there may be more than one if

there are several recipients at the same email address) has the Recipient's UserName, Email address and a Boolean indicating whether or not an Esign Agreement exists for that recipient.

- **GetRecipientList** – Return a collection of RecipientRecords for the supplied recipient email address. The purpose of this function is to allow a sender to determine what recipients are available in the system at the given email address.
- **GetSharedAccess** – This requests shared item status for one or more users and types of items.
- **GetStatusInDocuSignConnectFormat** – This method is reserved for future use.
- **MoveEnvelopes** – This method is used to move envelopes between folders.
- **Ping** - Returns true if this method can be reached. Use this method for testing availability. Available only in the 3.0 API.
- **PublishConnectFailures** – This method requests a list of Connect post failures for the set of envelopes included in the request.
- **PurgeDocuments** – Allows a sender to purge the documents from an envelope. Only completed documents can be purged.
- **PurgeDocumentsAndMetaData** – Can be used to purge envelope documents and metadata from the DocuSign system. The metadata that is removed along with the document includes the DocuSign tab data and any attachments that were provided with the envelope creation request. In all other respects, this is identical to PurgeDocuments.
- **RemoveAddressBookItems** – This method is used to remove specified items passed from your address book. This method returns an UpdateAddressBookResult object.
- **RequestCertificate** – Returns the signing certificate, which details the specific attributes of the participants and landmark events of the signing transaction, for an envelope.
- **RequestCertificateWithCertLanguage** - Returns the signing certificate, which details the specific attributes of the participants and landmark events of the signing transaction, for an envelope in the specified language.
- **RequestCorrectToken** – This call returns a token to place a user in a web session in Advanced Correct mode on an envelope.
- **RequestDocumentPDFs** - Returns a collection of all of the documents in an envelope. This method differs from RequestPDF in that this method returns individual documents, while RequestPDF returns all of the documents combined into a single, contiguous PDF. Additionally, the RequestDocumentPDFs method returns the Signing Certificate pdf document, which details the specific attributes of the participants and landmark events of the signing transaction.
- **RequestDocumentPDFsEx** - Returns an extended collection of all of the documents from RequestDocumentPDFs. Extensions include originating document ID and document type. Available only in the 3.0 API.
- **RequestDocumentPDFsRecipientsView** – This method requests the recipient's view of the document PDFs in an envelope.
- **RequestEnvelope** – Returns an API Envelope object containing all the data of an envelope. The envelope must be owned by the API user.

- **RequestEnvelopeWithDocumentFields** - This call is similar to the RequestStatus call. It returns the API Envelope object containing all the envelope data, including the document custom DocumentFields.
- **RequestEnvelopeHistoryToken** – Allows allows the caller to get a view of the history dialog for an envelope. The return URL is the url that the caller wants to return to when the close button is pressed after the history dialog is displayed.
- **RequestPDF** – Query the envelope and return the PDF.
- **RequestPDFNoWaterMark** – Query the envelope and return the PDF without showing the watermark.
- **RequestPDFRecipientView** - Returns the view of the document a recipient would see if the recipient downloaded the document. This allows the sending account to get the recipient view for cases where the documents or tabs the recipient sees are limited, such as when an envelope is sent with document visibility enabled.
- **RequestPDFWithCert** – Query the envelope and return the PDF with the signing certificate.
- **RequestPDFWithOptions** – Returns all of the documents combined into a single, contiguous PDF. It includes the ability to set display options for the PDFs.
- **RequestPDFsWithOptions** – Returns all of the documents in an envelope as an array of DocumentPDF (see DocumentPDF for more about the returned information). If the account has the Highlight Data Changes feature enabled and the ShowChanges option is set to true, any changes one the documents are highlighted.
- **RequestRecipientFaxToken** - Specific to InSession Signing implementations. This token returns a URL to invoke a DocuSign signing session where the signer is required to print out and sign documents. The documents are then returned to DocuSign by fax or upload.
- **RequestRecipientToken** - Specific to InSession Signing implementations. This token returns a URL to invoke a DocuSign signing session where the signer completes the signing process online.
- **RequestSenderToken** – This method is used to get a onetime use login token that allows the user to be placed into the DocuSign sending wizard. Upon sending completion the user is returned to the return URL provided by the API application.
- **RequestStatus** – Query the envelope and return its current status.
- **RequestStatusWithDocumentFields** - This call is similar to the RequestStatus and RequestStatusEx calls. It returns envelope status for the requested envelope along with all the envelope data, including the document custom DocumentFields.
- **RequestStatusChanges** – This method requests the envelope status changes for the envelopes for account on or after the specified date/time.
- **RequestStatusCodes** – This method requests the current state (Delivered, Complete, Voided, etc.) of the specified envelopes.
- **RequestStatusEx** - Query the envelope and return its extended status. Available only in the 3.0 API.
- **RequestStatuses** - Query DocuSign for a collection of EnvelopeStatuses. The request can be filtered by date range, StatusCode, sending user or EnvelopeID.
- **RequestStatusesWithDocumentFields** - This call is similar to the RequestStatuses and RequestStatusesEx calls and follows the same rules for requests if the result size is over 200

envelopes. It returns envelope status for the requested envelopes with all the envelope data, including the document custom DocumentFields.

- **RequestStatusesEx** - Query DocuSign for a collection of extended EnvelopeStatuses. The request can be filtered by date range, StatusCode, sending user or EnvelopeID. Available only in the 3.0 API.
- **RequestTemplate** – This method retrieves a specific template from the server by returning an EnvelopeTemplate upon execution completion.
- **RequestTemplateWithDocumentFields** - This call is similar to the RequestTemplate call. It returns the requested template containing all the template data, including the document custom DocumentFields.
- **RequestTemplateList** - This method requests a list of templates and returns an EnvelopeTemplate list.
- **RequestTemplateListWithDocumentFields** - This call is similar to the RequestTemplateList call. It returns the requested templates containing all the template data, including the document custom DocumentFields.
- **RequestTemplates** – This method returns a list of server side templates available for this account.
- **SaveTemplate** – This method returns the SaveTemplateResult.
- **SendEnvelope** – Send an envelope that is in draft status. Upon sending all the validations that are made with the CreateAndSendEnvelope method apply.
- **SetSharedAccess** – This method is used to set or update the shared item status for one or more users and types of items. The users and item types listed in the SharedAccessFilter. This call can only be used by users with account administration privileges
- **SynchEnvelope** – This method is only useful when the 'Asynchronous' flag is set to true on a CreateAndSendEnvelope call. It will determine when the queued envelope has been processed by the system.
- **TransferEnvelope** - Transfers ownership of the specified envelope to the specified User in the specified Account. Returns Boolean to indicate success of the operation.
- **UpdateAddressBookItems** – This method updates and inserts the specified items passed to your address book and returns an UpdateAddressBookResult object.
- **UploadTemplates** – This method returns SaveTemplateResult. This method currently only supports DocuSign Professional Template XML.
- **VoidEnvelope** - Voids the envelope. Returns Boolean to indicate success of the operation.

DocuSign Service API Function Groups

There are five basic function groups in the DocuSign API:

- Sending
- Embedding
- Status and Managing
- Post Processing

- Administrative

Additionally, there is a Credential API group that covers the methods involved in implementing a DocuSign Credential API. The exposed methods are associated with the different function groups.

Sending Function Group

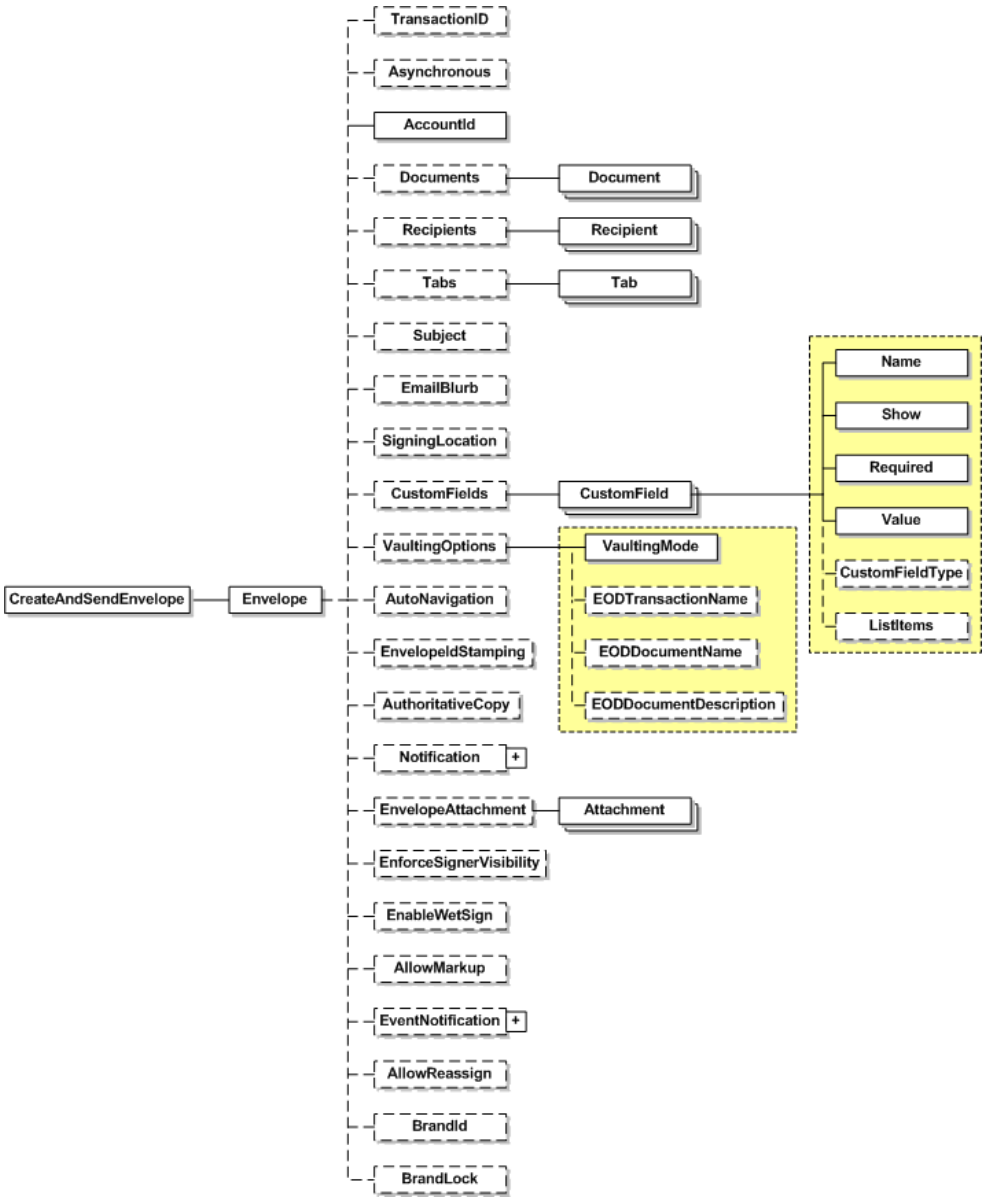
This section outlines the usage rules and behaviors of the methods associated with sending. The functions are presented in alphabetical order.

CreateAndSendEnvelope and CreateEnvelope

This section outlines the usage rules and behaviors of the *CreateEnvelope and CreateAndSendEnvelope* call, which is used to create envelopes, specify recipients, documents and actions on those documents within that envelope.

> **Note:** The *CreateAndSendEnvelope and CreateEnvelope* schemas, which are used to create envelopes, specify recipients, documents and actions on those documents within that envelope, are similar and only one method is detailed.

Schema



Name	Schema Type	Description
TransactionID	LongString	An optional element that can be used to identify an envelope. The id is a sender-generated value and is valid in the DocuSign system for 7 days. This is typically used when calling SynchEnvelope. The transactionId can be used determine an envelope status (i.e. was created or not) for cases where an internet connection was lost before the envelope status could be returned.
Asynchronous	Boolean	If true, will queue the envelope for processing and EnvelopeStatus will have a value of 'Processing'. Use SynchEnvelope to determine when the queued envelope has been processed. Additionally, RequestStatus calls will return 'Processing' until completed.
AccountId	DSXId	Account Id of the user who created the envelope.
Documents	*Document*	Complex element contains the details on the documents in the envelope. See the Document section below for more information.
Recipients	*Recipient*	Specifies the envelope recipients. See the Recipient section below for more information.
Tabs	*Tab*	Specifies information of the tabs affixed on the documents. Please refer the Tab section below for details.
Subject	String	The subject of the email that will be sent to the recipient. This can be a maximum of 100 characters.
EmailBlurb	String	Optional element, if specified is included in email notifications to the envelope recipients. This can be a maximum of 10000 characters.
SigningLocation	SigningLocationCode	Specifies the physical location where the signing takes place. It can have two enumeration values; InPerson and Online.

Name	Schema Type	Description
CustomFields	CustomField	Complex element contains a list of names and values. Element can specify if the name value pair needs to be entered before sending the envelope. It can also specify if the name value pair needs to be shown to user who is sending the envelope. **Note:** Each custom field used in the envelope must have a unique name. Reserved CustomField names: ##SFAccount ##SFContract ##SFOpportunity ##SFCase Reserved field names can be used to link data from DocuSign Connect to Salesforce. **Important:** If custom fields are set here when sending the envelope, only the fields specified in this section are included in the envelope. This overrides any required account level custom fields.
VaultingOptions	VaultingOptions	Specifies that, on document completion, this document is to be sent to an electronic vaulting solution. You MUST have an established relationship with the electronic vault and explicitly have vaulting permissions enabled to use this feature. Contact your DocuSign representative for more information.
AutoNavigation	Boolean	Specifies the Auto Navigation feature. If the Boolean is set to true the auto navigation is enabled.
EnvelopeIDStamping	Boolean	Specifies the Envelope Stamping feature. If the Boolean is set to true the Envelope Stamping is enabled.
AuthoritativeCopy	Boolean	Specifies the Authoritative copy feature. If the Boolean is set to true the Authoritative copy feature is enabled.
EnvelopeAttachment	Attachment	It would be possible to send attachments with envelope. This complex element contains data in base64Binary. It also contains label and type for the attachment.
Notification	Notification	Specifies information of the notification options for the envelope. Please refer the Notification section below for details.
EnforceSignerVisibility	Boolean	When true requires that a Signer have a signature or initial on the document or that the document has no signers in order to view it.

Name	Schema Type	Description
EnableWetSign	Boolean	When true, allows the signer to print the document and sign it on paper.
AllowMarkup	Boolean	When true, enable Document Markup for envelope.
EventNotification	EventNotification	This optional complex element allows a message to be sent a specified URL when the envelope or a recipient changes status. It is similar to DocuSign Connect, but only applies to the envelope. For example, from "Sent" to "Delivered", the URL is sent a message containing the updated envelope status and optionally the documents. When an EventNotification is attached to an envelope using the API, it only applies to the envelope (treating the envelope as the sender). This is different from envelopes created through the console user interface, where the user is treated as the sender. EventNotification consists of: • URL – The endpoint where envelope updates are sent. This will accept XML unless “useSoapInterface” is set to true. • LoggingEnabled – When set to true, logging is turned on for envelope events on the Web Console Connect page. • RequireAcknowledgment – When set to true, the DocuSign Connect service checks that the message was received and retries on failures. • UseSoapInterface – When set to true, this tells the Connect service that the user’s endpoint has implemented a SOAP interface. • SoapNameSpace – This lists the namespace in the SOAP listener provided. • IncludeCertificateWithSoap – When set to true, this tells the Connect service to send the DocuSign signedby certificate as part of the outgoing SOAP xml. This appears in the XML as wsse:BinarySecurityToken. • SignMessageWithX509Cert – When set to true, messages are signed with an X509 certificate. • IncludeDocuments – When set to true, the PDF documents are included in the message along with the update XML. • IncludeEnvelopeVoidReason – When set to true, this tells the Connect Service to include the void reason, as entered by the

Name	Schema Type	Description
		person that voided the envelope, in the message. • IncludeTimeZone – When set to true, the envelope time zone information is included in the message. • IncludeSenderAccountAsCustomField – When set to true, the sender account ID is included as a envelope custom field in the data. • IncludeDocumentFields – When set to true, this tells the Connect Service to include the Document Fields associated with the envelope. Document Fields are optional custom name-value pairs added to documents using the API. • IncludeCertificateOfCompletion – When set to true, this tells the Connect Service to include the Certificate of Completion with completed envelopes. • EnvelopeEvents – The list of envelope-level events statuses that will trigger Connect to send updates to the url. It can be a two-part list with: ○ EnvelopeEventStatusCode – The envelope status, this can be Sent, Delivered, Completed, Declined, or Voided. ○ IncludeDocuments – When set to true, the envelope time zone information is included in the message. • RecipientEvents – The list of recipient event statuses that will trigger Connect to send updates to the url. It can be a two-part list with: ○ RecipientEventStatusCode – The recipient status, this can be Sent, Delivered, Completed, Declined, AuthenticationFailed or AutoResponded. ○ IncludeDocuments – When set to true, the envelope time zone information is included in the message.
AllowReassign	Boolean	If true, the recipient can redirect an envelope to a more appropriate recipient.
BrandId	String	This sets the brand profile format used for the envelope. The value in the string is the BrandId for the profile. Account branding must be enabled for the account to use this option.

Name	Schema Type	Description
BrandLock	Boolean	When true, the brand profile associated with the envelope cannot be changed.
Accessibility	String	Optional element that can only be used if Document Accessibility is enabled for the account. This sets the document reading zones for screen reader applications. **Note:** This information is currently generated from the DocuSign web console by setting the reading zones when creating a template, exporting the reading zone string information, and adding it here.
MessageLock	Boolean	If true, prevents senders from changing the EmailBlurb and Subject for the envelope. Additionally, this prevents users from making changes to the EmailBlurb and Subject when correcting envelopes However, if the messageLock node is set to True and the Subject is empty, senders and correctors are able to add a subject to the envelope.
RecipientsLock	Boolean	If true, prevents senders from changing or correcting the recipient information for the envelope.
UseDisclosure	Boolean	When set to 'false', the Electronic Record and Signature Disclosure is not shown to any envelope recipients. When set to 'true', the disclosure is shown to recipients in accordance with the account's Electronic Record and Signature Disclosure frequency setting. If there is no setting for use UseDisclosure, then the account's normal disclosure setting is used and the UseDisclosure setting is not returned in responses when getting envelope information.
EmailSettings	EmailSettings	This optional complex element allows sender to override some envelope email setting information. This can be used to override the Reply To email address and name associated with the envelope and to override the BCC email addresses to which an envelope is sent. When the EmailSettings information is used for an envelope, it only applies to that envelope. **IMPORTANT:** The EmailSettings information is not returned in the response when an envelope is created or in the RequestEnvelope response. EmailSettings consists of: • ReplyEmailAddressOverride – The Reply To email used for the envelope.

Name	Schema Type	Description
		DocuSign will verify a correct email format is used, but does not verify that the email is active. This can be a maximum of 100 characters. • ReplyEmailNameOverride – The name associated with the Reply To email address. This can be a maximum of 100 characters. • BCCEmailAddresses – Only users with canManageAccount setting can use this option. A list of up to 5 Email addresses the envelope is sent to as a BCC email. DocuSign verifies that the email format is correct, but does not verify that the email is active. This can be a maximum of 100 characters. Using this overrides the BCC for Email Archive information setting for this envelope. Example: if your account has BCC for Email Archive set up for the email address 'archive@mycompany.com' and you send an envelope using the BCC Email Override to send a BCC email to 'salesarchive@mycompany.com', then a copy of the envelope is only sent to the 'salesarchive@mycompany.com' email address.

Request XML Data Structure Outline

```
SOAPAction: "http://www.docusign.net/API/3.0/CreateAndSendEnvelope"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CreateAndSendEnvelope xmlns="http://www.docusign.net/API/3.0">
      <Envelope>
        <TransactionID>string</TransactionID>
        <Asynchronous>boolean</Asynchronous>
        <AccountId>string</AccountId>
        <Documents>
          <Document>
            <ID>positiveInteger</ID>
            <Name>string</Name>
            <PDFBytes>base64Binary</PDFBytes>
            <Password>string</Password>
            <TransformPdfFields>boolean</TransformPdfFields>
            <FileExtension>string</FileExtension>
            <MatchBoxes xsi:nil="true" />
            <AttachmentDescription>string</AttachmentDescription>
            <DocumentFields>
              <DocumentField>
                <Name>
                <Value>
              </DocumentField>
```

```
              <DocumentField>
                <Name>
                <Value>
              </DocumentField>
            </DocumentFields>
          </Document>
          <Document>
            <ID>positiveInteger</ID>
            <Name>string</Name>
            <PDFBytes>base64Binary</PDFBytes>
            <Password>string</Password>
            <TransformPdfFields>boolean</TransformPdfFields>
            <FileExtension>string</FileExtension>
            <MatchBoxes xsi:nil="true" />
            <AttachmentDescription>string</AttachmentDescription>
            <DocumentFields>
              <DocumentField>
                <Name>
                <Value>
              </DocumentField>
              <DocumentField>
                <Name>
                <Value>
              </DocumentField>
            </DocumentFields>
          </Document>
        </Documents>
        <Recipients>
          <Recipient>
            <ID>positiveInteger</ID>
            <UserName>string</UserName>
            <SignerName>string</SignerName>
            <Email>string</Email>
            <Type>Signer or Agent or Editor or CarbonCopy or CertifiedDelivery or
InPersonSigner</Type>
            <AccessCode>string</AccessCode>
            <AddAccessCodeToEmail>boolean</AddAccessCodeToEmail>
            <RequireIDLookup>boolean</RequireIDLookup>
            <IDCheckConfigurationName>string</IDCheckConfigurationName>
            <LiveIDRecipientAuthentication>boolean</LiveIDRecipientAuthentication>
            <FacebookRecipientAuthentication>boolean</FacebookRecipientAuthentication>
            <LinkedinRecipientAuthentication>boolean</LinkedinRecipientAuthentication>
            <GoogleRecipientAuthentication>boolean</GoogleRecipientAuthentication>

<SalesforceRecipientAuthentication>boolean</SalesforceRecipientAuthentication>
            <TwitterRecipientAuthentication>boolean</TwitterRecipientAuthentication>
            <YahooRecipientAuthentication>boolean</YahooRecipientAuthentication>
            <OpenIDRecipientAuthentication>boolean</OpenIDRecipientAuthentication>

<AnySocialIDRecipientAuthentication>boolean</AnySocialIDRecipientAuthentication>
            <PhoneAuthentication xsi:nil="true" />
            <SAMLAuthentication>
              <SAMLAttributes xsi:nil="true" />
            </SAMLAuthentication>
            <SMSAuthentication xsi:nil="true" />
            <SignatureInfo xsi:nil="true" />
            <CaptiveInfo xsi:nil="true" />
            <CustomFields xsi:nil="true" />
            <RoutingOrder>unsignedShort</RoutingOrder>
            <IDCheckInformationInput xsi:nil="true" />
            <AutoNavigation>boolean</AutoNavigation>
            <RecipientAttachment xsi:nil="true" />
            <Note>string</Note>
            <RoleName>string</RoleName>
```

```
            <TemplateLocked>boolean</TemplateLocked>
            <TemplateRequired>boolean</TemplateRequired>
            <TemplateAccessCodeRequired>boolean</TemplateAccessCodeRequired>
            <DefaultRecipient>boolean</DefaultRecipient>
            <EmailNotification xsi:nil="true" />
            <AgentCanEditEmail>boolean</AgentCanEditEmail>
            <AgentCanEditName>boolean</AgentCanEditName>
            <SignInEachLocation>boolean</SignInEachLocation>
            <ExcludedDocuments xsi:nil="true" />
            <RequireSignerCertificate>string</RequireSignerCertificate>

<InheritEmailNotificationConfiguration>boolean</InheritEmailNotificationConfiguration>
          </Recipient>
          <Recipient>
            <ID>positiveInteger</ID>
            <UserName>string</UserName>
            <SignerName>string</SignerName>
            <Email>string</Email>
            <Type>Signer or Agent or Editor or CarbonCopy or CertifiedDelivery or
InPersonSigner</Type>
            <AccessCode>string</AccessCode>
            <AddAccessCodeToEmail>boolean</AddAccessCodeToEmail>
            <RequireIDLookup>boolean</RequireIDLookup>
            <IDCheckConfigurationName>string</IDCheckConfigurationName>
            <LiveIDRecipientAuthentication>boolean</LiveIDRecipientAuthentication>
            <FacebookRecipientAuthentication>boolean</FacebookRecipientAuthentication>
            <LinkedinRecipientAuthentication>boolean</LinkedinRecipientAuthentication>
            <GoogleRecipientAuthentication>boolean</GoogleRecipientAuthentication>

<SalesforceRecipientAuthentication>boolean</SalesforceRecipientAuthentication>
            <TwitterRecipientAuthentication>boolean</TwitterRecipientAuthentication>
            <YahooRecipientAuthentication>boolean</YahooRecipientAuthentication>
            <OpenIDRecipientAuthentication>boolean</OpenIDRecipientAuthentication>

<AnySocialIDRecipientAuthentication>boolean</AnySocialIDRecipientAuthentication>
            <PhoneAuthentication xsi:nil="true" />
            <SMSAuthentication xsi:nil="true" />
            <SignatureInfo xsi:nil="true" />
            <CaptiveInfo xsi:nil="true" />
            <CustomFields xsi:nil="true" />
            <RoutingOrder>unsignedShort</RoutingOrder>
            <IDCheckInformationInput xsi:nil="true" />
            <AutoNavigation>boolean</AutoNavigation>
            <RecipientAttachment xsi:nil="true" />
            <Note>string</Note>
            <RoleName>string</RoleName>
            <TemplateLocked>boolean</TemplateLocked>
            <TemplateRequired>boolean</TemplateRequired>
            <TemplateAccessCodeRequired>boolean</TemplateAccessCodeRequired>
            <DefaultRecipient>boolean</DefaultRecipient>
            <EmailNotification xsi:nil="true" />
            <AgentCanEditEmail>boolean</AgentCanEditEmail>
            <AgentCanEditName>boolean</AgentCanEditName>
            <SignInEachLocation>boolean</SignInEachLocation>
            <ExcludedDocuments xsi:nil="true" />
            <RequireSignerCertificate>string</RequireSignerCertificate>

<InheritEmailNotificationConfiguration>boolean</InheritEmailNotificationConfiguration>
          </Recipient>
        <Tabs>
          <Tab>
            <DocumentID>positiveInteger</DocumentID>
            <RecipientID>positiveInteger</RecipientID>
            <PageNumber>nonNegativeInteger</PageNumber>
```

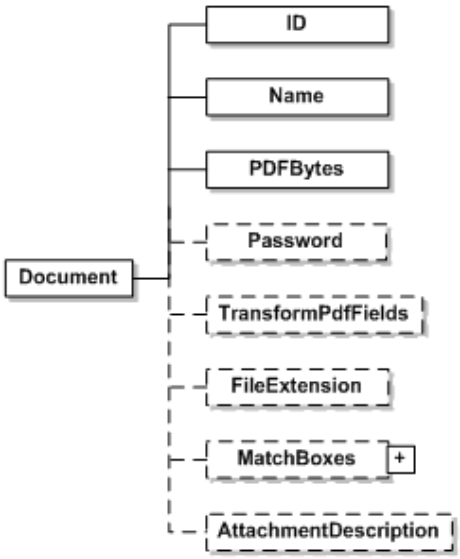
```
            <XPosition>nonNegativeInteger</XPosition>
            <YPosition>nonNegativeInteger</YPosition>
            <ScaleValue>decimal</ScaleValue>
            <AnchorTabItem xsi:nil="true" />
            <Type>InitialHere or SignHere or FullName or FirstName or LastName or
EmailAddress or Company or Title or DateSigned or InitialHereOptional or EnvelopeID or
Custom or SignerAttachment or SignHereOptional or Approve or Decline or
SignerAttachmentOptional</Type>
            <Name>string</Name>
            <TabLabel>string</TabLabel>
            <Value>string</Value>
            <CustomTabType>Text or Checkbox or Radio or List or Date or Number or SSN or
ZIP5 or ZIP5DASH4 or Email or Note</CustomTabType>
            <CustomTabWidth>int</CustomTabWidth>
            <CustomTabHeight>int</CustomTabHeight>
            <CustomTabRequired>boolean</CustomTabRequired>
            <CustomTabLocked>boolean</CustomTabLocked>
            <CustomTabDisableAutoSize>boolean</CustomTabDisableAutoSize>
            <CustomTabListItems>string</CustomTabListItems>
            <CustomTabListValues>string</CustomTabListValues>
            <CustomTabListSelectedValue>string</CustomTabListSelectedValue>
            <CustomTabRadioGroupName>string</CustomTabRadioGroupName>
            <CustomTabValidationPattern>string</CustomTabValidationPattern>
            <CustomTabValidationMessage>string</CustomTabValidationMessage>
            <TemplateLocked>boolean</TemplateLocked>
            <TemplateRequired>boolean</TemplateRequired>
            <ConditionalParentLabel>string</ConditionalParentLabel>
            <ConditionalParentValue>string</ConditionalParentValue>
            <SharedTab>boolean</SharedTab>
            <RequireInitialOnSharedTabChange>boolean</RequireInitialOnSharedTabChange>
            <ConcealValueOnDocument>boolean</ConcealValueOnDocument>
            <Font>Arial or ArialNarrow or Calibri or CourierNew or Garamond or Georgia or
Helvetica or LucidaConsole or Tahoma or TimesNewRoman or Trebuchet or Verdana or MSGothic
or MSMinchoor OcrA</Font>
            <Bold>boolean</Bold>
            <Italic>boolean</Italic>
            <Underline>boolean</Underline>
            <FontColor>Black or BrightBlue or BrightRed or DarkRed or DarkGreen or Gold
or Green or NavyBlue or Purple or White</FontColor>
            <FontSize>Size7 or Size8 or Size9 or Size10 or Size11 or Size12 or Size14 or
Size16 or Size18 or Size20 or Size22 or Size24 or Size26 or Size28 or Size36 or Size48 or
Size72</FontSize>
            <MergeFieldXml>string</MergeFieldXml>
            <IncludeNoteInEmail>boolean</IncludeNoteInEmail>
          </Tab>
          <Tab>
            <DocumentID>positiveInteger</DocumentID>
            <RecipientID>positiveInteger</RecipientID>
            <PageNumber>nonNegativeInteger</PageNumber>
            <XPosition>nonNegativeInteger</XPosition>
            <YPosition>nonNegativeInteger</YPosition>
            <ScaleValue>decimal</ScaleValue>
            <AnchorTabItem xsi:nil="true" />
            <Type>InitialHere or SignHere or FullName or FirstName or LastName or
EmailAddress or Company or Title or DateSigned or InitialHereOptional or EnvelopeID or
Custom or SignerAttachment or SignHereOptional or Approve or Decline or
SignerAttachmentOptional</Type>
            <Name>string</Name>
            <TabLabel>string</TabLabel>
            <Value>string</Value>
            <CustomTabType>Text or Checkbox or Radio or List or Date or Number or SSN or
ZIP5 or ZIP5DASH4 or Email</CustomTabType>
            <CustomTabWidth>int</CustomTabWidth>
            <CustomTabHeight>int</CustomTabHeight>
```

```
            <CustomTabRequired>boolean</CustomTabRequired>
            <CustomTabLocked>boolean</CustomTabLocked>
            <CustomTabDisableAutoSize>boolean</CustomTabDisableAutoSize>
            <CustomTabListItems>string</CustomTabListItems>
            <CustomTabListValues>string</CustomTabListValues>
            <CustomTabListSelectedValue>string</CustomTabListSelectedValue>
            <CustomTabRadioGroupName>string</CustomTabRadioGroupName>
            <CustomTabValidationPattern>string</CustomTabValidationPattern>
            <CustomTabValidationMessage>string</CustomTabValidationMessage>
            <TemplateLocked>boolean</TemplateLocked>
            <TemplateRequired>boolean</TemplateRequired>
            <ConditionalParentLabel>string</ConditionalParentLabel>
            <ConditionalParentValue>string</ConditionalParentValue>
            <SharedTab>boolean</SharedTab>
            <RequireInitialOnSharedTabChange>boolean</RequireInitialOnSharedTabChange>
            <ConcealValueOnDocument>boolean</ConcealValueOnDocument>
            <Font>Arial or ArialNarrow or Calibri or CourierNew or Garamond or Georgia or
Helvetica or LucidaConsole or Tahoma or TimesNewRoman or Trebuchet or Verdana or MSGothic
or MSMincho or OcrA</Font>
            <Bold>boolean</Bold>
            <Italic>boolean</Italic>
            <Underline>boolean</Underline>
            <FontColor>Black or BrightBlue or BrightRed or DarkRed or DarkGreen or Gold
or Green or NavyBlue or Purple or White</FontColor>
            <FontSize>Size7 or Size8 or Size9 or Size10 or Size11 or Size12 or Size14 or
Size16 or Size18 or Size20 or Size22 or Size24 or Size26 or Size28 or Size36 or Size48 or
Size72</FontSize>
            <MergeFieldXml>string</MergeFieldXml>
            <IncludeNoteInEmail>boolean</IncludeNoteInEmail>
          </Tab>
        </Tabs>
        <Subject>string</Subject>
        <EmailBlurb>string</EmailBlurb>
        <SigningLocation>InPerson or Online</SigningLocation>
        <CustomFields>
          <CustomField>
            <Name>string</Name>
            <Show>string</Show>
            <Required>string</Required>
            <Value>string</Value>
            <CustomFieldType>Text or List</CustomFieldType>
            <ListItems>string</ListItems>
          </CustomField>
          <CustomField>
            <Name>string</Name>
            <Show>string</Show>
            <Required>string</Required>
            <Value>string</Value>
            <CustomFieldType>Text or List</CustomFieldType>
            <ListItems>string</ListItems>
          </CustomField>
        </CustomFields>
        <VaultingOptions>
          <VaultingMode>None or EODeStore or EODAuthoritativeCopy</VaultingMode>
          <EODTransactionName>string</EODTransactionName>
          <EODDocumentName>string</EODDocumentName>
          <EODDocumentDescription>string</EODDocumentDescription>
        </VaultingOptions>
        <AutoNavigation>boolean</AutoNavigation>
        <EnvelopeIdStamping>boolean</EnvelopeIdStamping>
        <AuthoritativeCopy>boolean</AuthoritativeCopy>
        <Notification>
          <UseAccountDefaults>boolean</UseAccountDefaults>
          <Reminders>
```

```
            <ReminderEnabled>boolean</ReminderEnabled>
            <ReminderDelay>nonNegativeInteger</ReminderDelay>
            <ReminderFrequency>nonNegativeInteger</ReminderFrequency>
          </Reminders>
          <Expirations>
            <ExpireEnabled>boolean</ExpireEnabled>
            <ExpireAfter>nonNegativeInteger</ExpireAfter>
            <ExpireWarn>nonNegativeInteger</ExpireWarn>
          </Expirations>
        </Notification>
        <EnvelopeAttachment>
          <Attachment>
            <Data>base64Binary</Data>
            <Label>string</Label>
            <Type>string</Type>
          </Attachment>
          <Attachment>
            <Data>base64Binary</Data>
            <Label>string</Label>
            <Type>string</Type>
          </Attachment>
        </EnvelopeAttachment>
        <EnforceSignerVisibility>boolean</EnforceSignerVisibility>
        <EnableWetSign>boolean</EnableWetSign>
        <AllowMarkup>boolean</AllowMarkup>
        <EventNotification>
          <URL>string</URL>
          <LoggingEnabled>boolean</LoggingEnabled>
          <RequireAcknowledgment>boolean</RequireAcknowledgment>
          <UseSoapInterface>boolean</UseSoapInterface>
          <SoapNameSpace>string</SoapNameSpace>
          <IncludeCertificateWithSoap>boolean</IncludeCertificateWithSoap>
          <SignMessageWithX509Cert>boolean</SignMessageWithX509Cert>
          <IncludeDocuments>boolean</IncludeDocuments>
          <IncludeTimeZone>boolean</IncludeTimeZone>
          <IncludeSenderAccountAsCustomField>boolean</IncludeSenderAccountAsCustomField>
          <EnvelopeEvents>
            <EnvelopeEvent xsi:nil="true" />
            <EnvelopeEvent xsi:nil="true" />
          </EnvelopeEvents>
          <RecipientEvents>
            <RecipientEvent xsi:nil="true" />
            <RecipientEvent xsi:nil="true" />
          </RecipientEvents>
        </EventNotification>
        <AllowReassign>boolean</AllowReassign>
        <BrandId>string</BrandId>
        <BrandLock>boolean</BrandLock>
        <Accessibility>boolean</Accessibility>
        <MessageLock>boolean</MessageLock>
        <RecipientsLock>boolean</RecipientsLock>
        <UseDisclosure>boolean</UseDisclosure>
        <EmailSettings>
          <ReplyEmailAddressOverride>string</ReplyEmailAddressOverride>
          <ReplyEmailNameOverride>string</ReplyEmailNameOverride>
          <BCCEmailAddresses>
            <Email xsi:nil="true" />
            <Email xsi:nil="true" />
          </BCCEmailAddresses>
        </EmailSettings>
      </Envelope>
    </CreateAndSendEnvelope>
  </soap:Body>
</soap:Envelope>
```

Document

This element contains the details of the documents in the envelope.

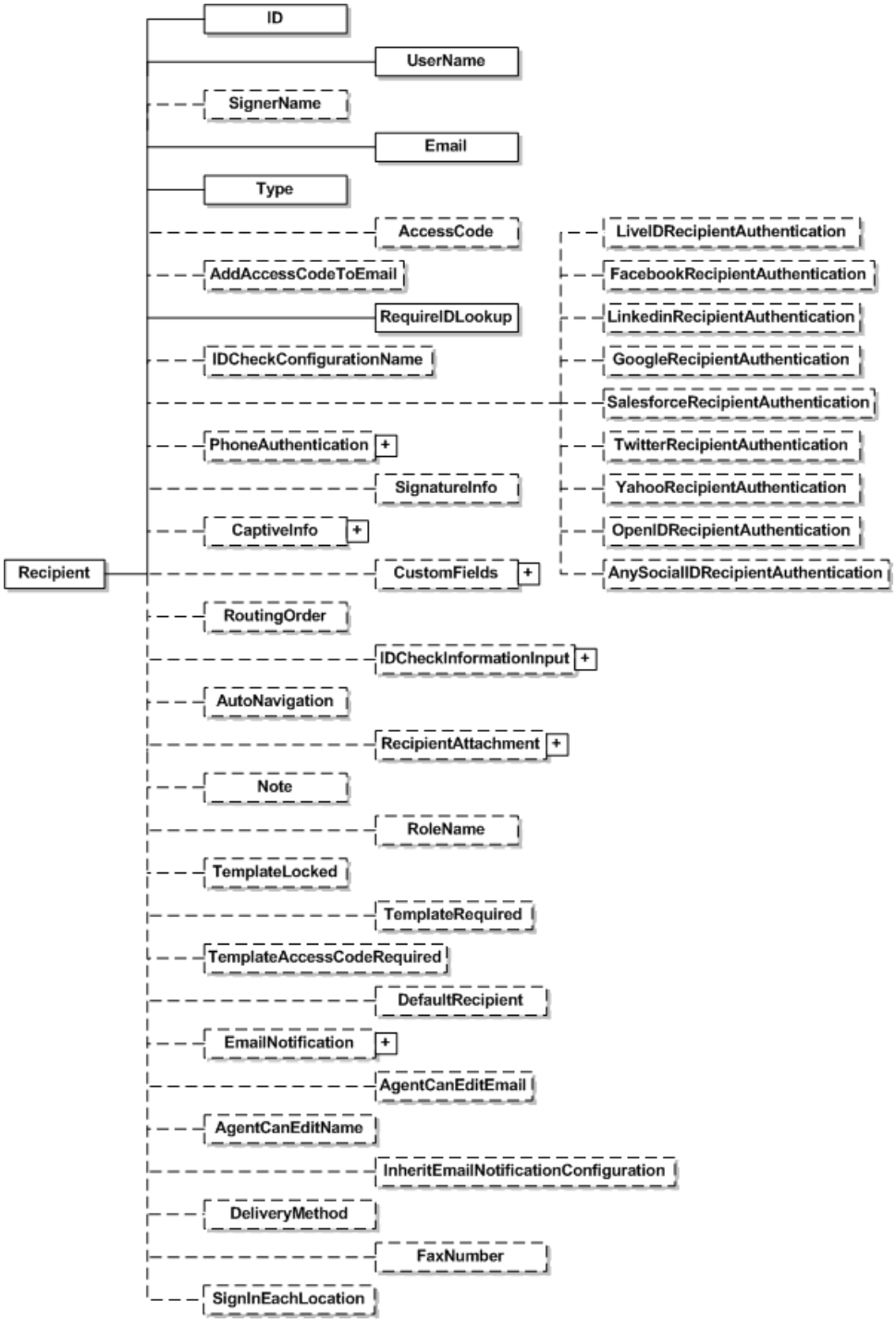


Name	Schema Type	Description
ID	LocalId	The unique Id for the document in the specific envelope.
Name	LongString	The name of the document. This can be a maximum of 100 characters.
PDFBytes	base64Binary	The document byte stream
Password	Password	This is a legacy field and no longer used.
TransformPdfFields	Boolean	Optional element. When set to true PDF form field data will be transformed into document tab values when the PDF form field name matches the DocuSign custom tab TabLabel. The resulting PDF form data will also be returned in the PDF meta data when requesting the document PDF. Note: DocuSign will not transform PDF form fields that have the text "DocuSignIgnoreTransform" or "eSignIgnoreTransform" as part of the name of the PDF form field.
FileExtension	String	Optional element. File extension of the document. If the document is non-PDF it will be converted to PDF.

Name	Schema Type	Description
MatchBoxes	String	Optional element. Only used when uploading and editing templates. Matchboxes are used to define areas in a document for document matching when creating envelopes. The Matchbox schema is defined below in the Request Template section.
AttachmentDescription	LongString	Optional element. If present, indicates signer will be attaching a document here.
DocumentFields	ArrayOfDocumentField	Optional element that allows the sender to add custom data to a document. This information is returned with status and template requests, but otherwise not used by DocuSign. Each DocumentField in the array contains the elements: Name – A string that can be a maximum of 50 characters. Value – A string that can be a maximum of 200 characters.

Recipient

The Recipient element is used to specify envelope recipients.



lame	Schema Type	Description
D	LocalId	Unique for the recipient. It is used by the tab element to indicate which recipient is to sign the Document. It must be a positive integer and cannot be 0 (zero).

lame	Schema Type	Description
serName	UserName	Full legal name of the recipient. If the recipient Type is InPersonSigner, this is the signing host for the envelope. This can be a maximum of 100 characters.
ignerName	SignerName	The full legal name of a signer for an InPersonSigner recipient Type. This can be a maximum of 100 characters.
mail	Email	Email of the recipient. Notification will be sent to this email id. This can be a maximum of 100 characters.
ype	RecipientTypeCode	Specifies the role of the recipient. The role can be Signer, CarbonCopy, Editor, Agent, Intermediary, CertifiedDelivery, or InPersonSigner. If InPersonSigner is used, SignerName is required and is the name of the person signing the tabs. Also, the UserName is the name of the signing host for the envelope.
ccessCode	LongString	This Optional element specifies the access code a recipient has to enter to validate the identity. This can be a maximum of 50 characters.
ddAccessCodeToEmail	Boolean	This Optional attribute indicates that the access code will be added to the email sent to the recipient; this nullifies the Security measure of Access Code on the recipient.
equireIDLookup	Boolean	If set to true, then the recipient is required to complete an ID Check or Phone Authentication check.
DCheckConfigurationName	LongString	Specify ID check configuration by name. Overrides the default.
eliveryMethod	Delivery Method	This sets the delivery method used for the recipient. The two enumerations are Email or Fax. Email is the default value.
axNumber	String	The fax number for the recipient. This is only active if the DeliveryMethod is Fax. This can be a maximum of 100 characters.
iveIDRecipientAuthentication	Boolean	If true, this is the information used for recipient authentication.
acebookRecipientAuthentication	Boolean	If true, this is the information used for recipient authentication.
inkedinRecipientAuthentication	Boolean	If true, this is the information used for recipient authentication.
oogleRecipientAuthentication	Boolean	If true, this is the information used for recipient authentication.
alesforceRecipientAuthentication	Boolean	If true, this is the information used for recipient authentication.
witterRecipientAuthentication	Boolean	If true, this is the information used for recipient authentication.

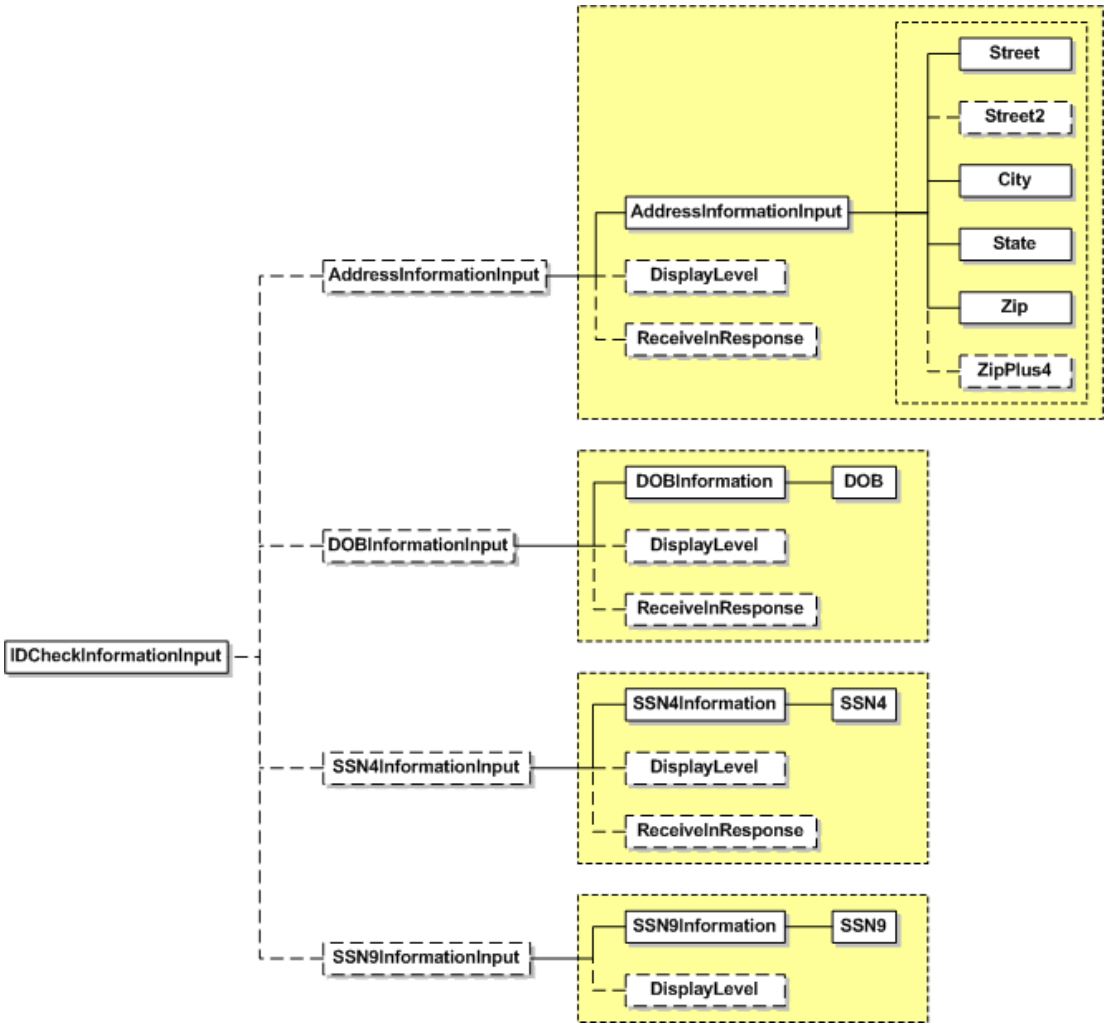
lame	Schema Type	Description
ahooRecipientAuthentication	Boolean	If true, this is the information used for recipient authentication.
penIDRecipientAuthentication	Boolean	If true, this is the information used for recipient authentication.
nySocialIDRecipient uthentication	Boolean	If true, this is the information used for recipient authentication.
honeAuthentication	RecipientPhone Authentication	Optional element. Contains the elements: RecipMayProvideNumbers – Boolean (if true then recipient can use whatever phone number they want) SenderProvideNumbers – ArrayOfString (a list of phone numbers the recipient may use) RecordVoidPrint – Reserved ValidateRecipProvideNumber – Reserved
AMLAuthentication	SAMLAttributes	Optional element, account must be set up to use SSO to use this. Contains an array of the name/value pairs for the SAML assertion attributes: name – The name of the SAML assertion attribute. value – The value associated with the named SAML assertion attribute.
MSAuthentication	RecipientSMS Authentication	Optional element. Contains the element: SenderProvideNumbers – Array with a list of phone numbers the recipient may use for SMS text authentication.
ignatureInfo	SignatureInfo	Allows the sender to pre-specify the signature name, signature initials and signature font used in the signature stamp for the recipient.
aptiveInfo	CaptiveInfo ClientUserId EmbeddedRecipient StartURL	CaptiveInfo is made up of two optional elements that 1) determine if a recipient is embedded or remote, and 2) express a redirect URL used for starting a recipient's signing session. • ClientUserId – This is a sender created value that can be a maximum of 100 characters. If there is a value for this (it is not null) then the recipient is considered to be embedded. Note that if a ClientUserId is used and the account settings SignerMustHaveAccount or SignerMustLoginToSign are true, an error is generated on sending. • EmbeddedRecipientStartURL – This is a sender provided valid URL string for redirecting the recipient. When using this option, the embedded recipient still receives an email from DocuSign, just as a remote recipient would, but when the document link in the email is clicked the recipient is redirected, through DocuSign, to this URL to complete their actions. When routing to the URL, it is up to the sender's system (the server responding to the URL) to then request a recipient token to launch a signing session. If the value SIGN_AT_DOCUSIGN is used for this

lame	Schema Type	Description
		node, the recipient will be directed to an embedded signing or viewing process directly at DocuSign. The signing or viewing action is initiated by the DocuSign system and the transaction activity and Certificate of Completion records will reflect this. In all other ways the process is identical to an embedded signing or viewing operation that would be launched by any partner. It is important to remember that in a typical embedded workflow the authentication of an embedded recipient is the responsibility of the sending application and DocuSign expects that senders will follow their own process for establishing the recipient's identity. In this workflow the recipient goes through the sending application before the embedded signing or viewing process in initiated. However, when the sending application sets EmbeddedRecipientStartURL=SIGN_AT_DOCUSIGN, the recipient goes directly to the embedded signing or viewing process bypassing the sending application and any authentication steps the sending application would use. In this case, DocuSign recommends that one of the normal DocuSign authentication features (Access Code, Phone Authentication, SMS Authentication, etc.) be used to verify the identity of the recipient. If a ClientUserId is NOT provided and an EmbeddedRecipientStartURL is provided, DocuSign will ignore the redirect URL and launch the standard signing process for the email recipient. Information can be appended to the EmbeddedRecipientStartURL using merge fields. The available merge fields items are: EnvelopeId, RecipientId, RecipientName, RecipientEmail, and CustomFields. The CustomFields must be part of the Recipient or Envelope. The merge fields are enclosed in double brackets. *Example:* http://senderHost/[[mergeField1]]/ beginSigningSession?[[mergeField2]]& [[mergeField3]]
ustomFields	Array of CustomField	Allows the sender to send custom data about the recipient. This information is returned in the envelope status but otherwise not used by DocuSign. Each CustomField can be a maximum of 100 characters.
outingOrder	PositiveShort	This element specifies the routing order of the recipient in the envelope
DCheckInformationInput	*IDCheckInformationInput*	This complex element contains information related to recipient ID check. It comprises of Address, DOB, SS4 and SS9. Please refer the IDCheckInformationInput section below for details.
utoNavigation	Boolean	Specifies the auto navigation setting for recipient.

lame	Schema Type	Description
ecipientAttachment	Attachment	It would be possible to send attachments with recipients. This complex element contains data in base64Binary. It also contains label and type for the attachment.
lote	String	A note that will be unique to this recipient. This note will be sent to the recipient via the signing email. This note will display in the signing UI near the upper left corner of the document on the signing screen. This can be a maximum of 1000 characters.
oleName	String	Optional element. Used only when working with template recipients. Role name for the recipient
emplateLocked	Boolean	Optional element. Used only when working with template recipients. If true, the sender cannot change any attributes of the recipient.
emplateRequired	Boolean	Optional element. Used only when working with template recipients. If true, the sender may not remove the recipient.
emplateAccessCodeRequired	Boolean	Optional element. Used only when working with template recipients. If true and TemplateLocked = true, the sender must enter an access code.
efaultRecipient	Boolean	If true, this is the default recipient for the envelope. This option is used with CreateEnvelopeFromTemplatesAndForms method.
mailNotification	RecipientEmailNotification	An optional complex type that has information for setting the language for the recipient's email information. It is composed of three elements: • Subject: a long string with the subject of the email sent to the recipient. • EmailBlurb: a string with the email message sent to the recipient. • Language: the simple type SupportedLanguage enumerating the language used. The supported languages, with the language value shown in parenthesis, are: Arabic (ar), Bahasa Indonesia (id), Bahasa Melayu (ms) Bulgarian (bg), Czech (cs), Chinese Simplified (zh_CN), Chinese Traditional (zh_TW), Croatian (hr), Danish (da), Dutch (nl), English US (en), English UK (en_GB), Estonian (et), Farsi (fa), Finnish (fi), French (fr), French Canada (fr_CA), German (de), Greek (el), Hebrew (he), Hindi (hi), Hungarian (hu), Italian (it), Japanese (ja), Korean (ko), Latvian (lv), Lithuanian (lt), Norwegian (no), Polish (pl), Portuguese (pt), Portuguese Brazil (pt_BR), Romanian (ro),Russian (ru), Serbian (sr), Slovak (sk), Slovenian (sl), Spanish (es),Spanish Latin America (es_MX), Swedish (sv), Thai (th), Turkish (tr), Ukrainian (uk) and Vietnamese (vi). **IMPORTANT:** If this is enabled for one recipient, it overrides the Envelope Subject and EmailBlurb. Also, you must enable EmailNotification for all recipients.

lame	Schema Type	Description
gentCanEditEmail	Boolean	Optional element. If true, the Agent or Editor Recipient associated with this Recipient can change the Recipient's pre-populated Email address. This element is only active if enabled for the account.
gentCanEditName	Boolean	Optional element. If true, the Agent or Editor Recipient associated with this recipient can change the recipient's pre-populated name (UserName). This element is only active if enabled for the account.
ignInEachLocation	Boolean	Optional element. If true, the signer must write their signature in each Signature or initial tab. Can only be used for Signer and InPersonSigner recipient types.
heritEmailNotificationConfiguration	Boolean	Optional element. If true and the recipient creates a DocuSign account after signing, the Manage Account Email Notification settings are used as the default settings for the recipient's account.
xcludedDocuments	Array of positive integers	Optional element, can only be used if Document Visibility is enabled for the account. Accepts an array of Document IDs to set the documents in the envelope that are not visible to the recipient. If a recipient has a tab on a document, it cannot be excluded.
equireSignerCertificate	String	Sets the type of signer certificate required for signing. If left blank, no certificate is required. Only one type of certificate can be set for a signer. The possible values are: • DocusignExpress – Requires a DocuSign Express certificate. • Safe – Requires a SAFE-BioPharma certificate. • OpenTrust – Requires an OpenTrust certificate. **Important:** There are certain rules and restrictions that must be followed when requiring OpenTrust digital signatures. See Rules and Restrictions for OpenTrust Envelopes more information.
igningGroupId	String	The DocuSign generated Signing Group ID. When used, the Email and UserName parameters are not required for this recipient. Note that a Signing Group cannot be used as an In Person Signing host. So a Signing Group ID cannot be used if the Recipient – Type = InPersonSigner Signing Groups cannot use the Fax out for Signature feature. If phone authentication is specified for a signing group recipient, then the RecipMayProvideNumber option must be set to true. DocuSign does not recommend that SMS authentication be used with Signing Groups, since the recipient's SMS phone number must be entered before sending and the group members probably aren't sharing the same phone number.

IDCheckInformationInput



Name	Schema Type	Description
AddressInformationInput	AddressInformation	This Complex type contains the following information; Street1, Street2, City, State and Zip and ZipPlus4. DisplayLevel specifies the display level for the recipient, which is controlled by an enumerator. Display level could be ReadOnly, Editable or DoNotDisplay. Boolean element ReceiveInResponse specifies if the information needs to be returned in the response. The maximum characters or each field are: - Street1 and Street2: 150 characters - City: 50 characters - State: 2 characters - ZIP and ZIP+4: 20 characters

DoBInformationInput	DOBInformation	Contains DOB information; date, Month and Year. This complex element also contains DisplayLevel and ReceiveInResponse elements.
SSN4InformationInput	SSN4Information	The last four digits of the social security number of the recipient. This complex element also contains DisplayLevel and ReceiveInResponse elements.
SSN9InformationInput	SSN9Information	Social security number of the recipient. This element contains the DisplayLevel. SSN9 is never returned in the response.

Rules and Restrictions for OpenTrust Envelopes

The DocuSign API enforces certain rules and restrictions when at least one signer or in person signer is required to use an OpenTrust signer certificate (RequireSignerCertificate=OpenTrust).

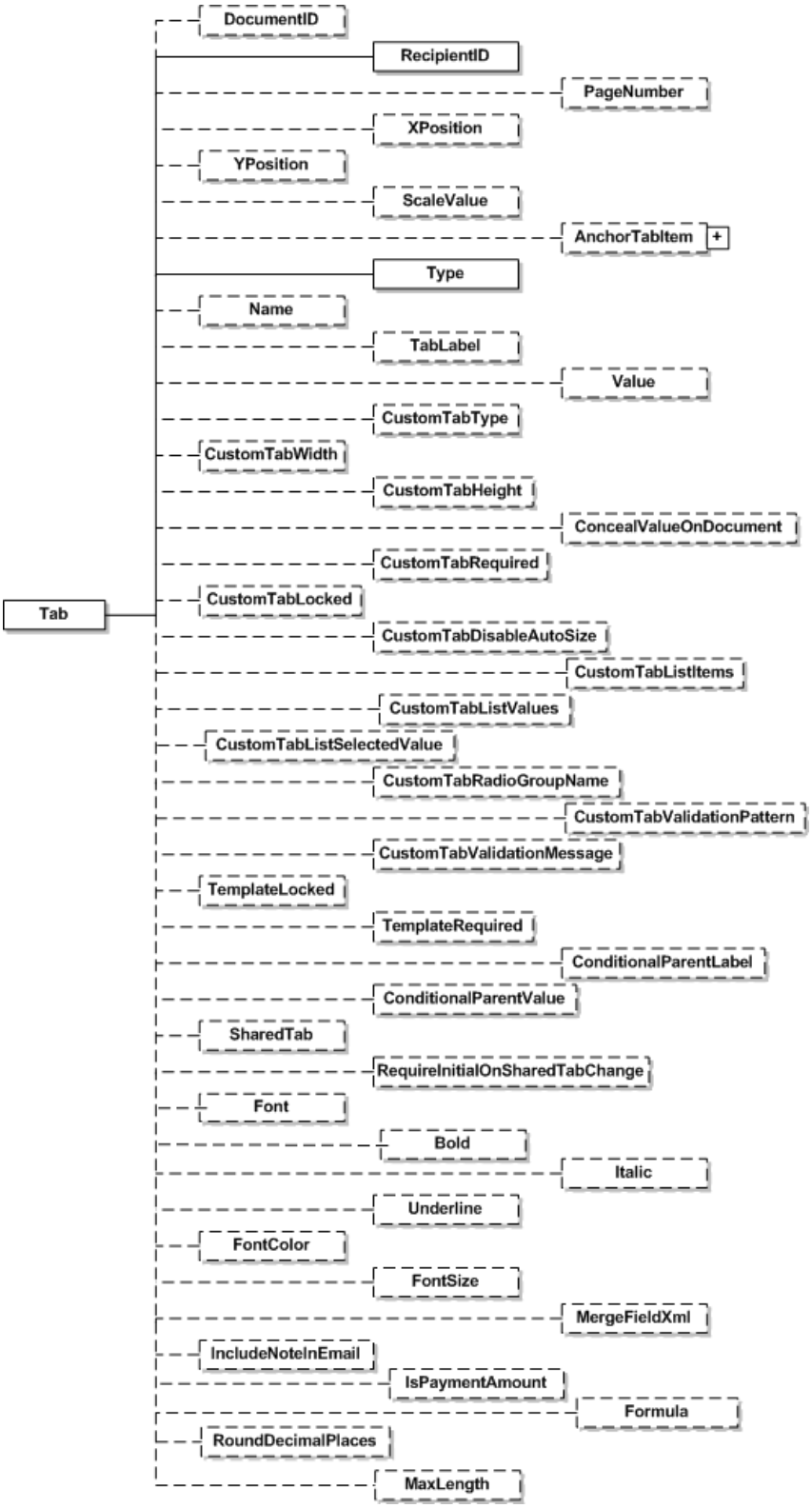
The following restrictions apply when sending an envelope:

- Sign on Paper, Document Markup, Field Markup, watermarks, and Electronic Notary cannot be enabled for the envelope.
- A SMS Authentication or Access Code authentication check must be included for the OpenTrust signer.
- All recipients must be the only recipient in their routing order.
- The OpenTrust signer must have at least one required Signature or Initials tag in the envelope (the envelope cannot be sent for Free-Form signing). Additionally, the signer cannot have more than 5 OpenTrust Signature tabs in the envelope.
- When a signer in the envelope is required to apply a digital signature, then all subsequent signers that have Signature or Initial tags are required to use a digital signature. Note that other recipient types can be used and subsequent signers can use other tab types.
- Downloading documents individually or as a combined PDF is allowed. However, downloading the combined PDF of all documents might cause the digital signatures to appear invalid due to the process of combining the documents. This is a known issue and each document can be downloaded individually to verify the digital signature.

Other restrictions that apply to envelopes:

- Envelopes cannot be corrected if at least one OpenTrust signer has signed the envelope.

Tab



Name	Schema Type	Description

Name	Schema Type	Description
DocumentID	LocalId	This is the document on that the tab is placed on. Must refer to an existing Document's ID attribute.
RecipientID	LocalId	This specifies the recipient associated with the tab. Must refer to an existing recipient's ID attribute.
PageNumber	nonNegativeInteger	This specifies the page number where the tab will be affixed.
XPosition	nonNegativeInteger	This indicates the horizontal offset of the tab on the page, in a coordinate space that has left top corner of the document as origin. DocuSign uses 72 DPI when determining position.
YPosition	nonNegativeInteger	This indicates the vertical offset of the tab on the page, in a coordinate space that has left top corner of the document as origin. DocuSign uses 72 DPI when determining position.
ScaleValue	Decimal	Represents a decimal value of 0.5 to 1.0 position of the webcontrol scaling slider. A value of 1.0 represents full size and 0.5 is 50% size.
AnchorTabItem	AnchorTab	This indicates Anchor tab in the document. See the Anchor Tab section below for details
Type	TabTypeCode	TabTypeCode is a simple type enumerating one of the following possible values: InitialHere, SignHere, FullName, FirstName, LastName, EmailAddress, Company, Title, DateSigned, InitialHereOptional, EnvelopeID, Custom, SignerAttachment, SignHereOptional, Approve, Decline, or SignerAttachmentOptional.
Name	Name	This would be the name of the custom tab. Notes: For Type of "Custom" and CustomTabType of "List" the name is a ";" separated list of drop down list items.

Name	Schema Type	Description
TabLabel	TabLabel	This would be the name of the custom tab. This can be a maximum of 500 characters. Notes: Making custom tab's TabLabel the same will cause the all like fields to update when the user enters data. When using this option, the TabLabel must not contain any spaces. For Type of "Custom" and CustomTabType of "Radio" make the TabLabels the same for any radio buttons you want to group together. The Name value will be what is returned for the radio selected in a group.
Value	Value	This element specifies the value of the custom tab. Notes: For Type of "Custom" and CustomTabType of "List" the Value is the default selected item in the drop down list. For Type of "Custom" and CustomTabType of "Radio" or "Checkbox" place a "X" in this field to check the item by default.
CustomTabType	CustomTabType	It tells what is represented by the tabs that have a TabTypeCode of Custom. Available custom tab types are: Text, Checkbox, Radio, List, Date, Number, SSN, ZIP5, ZIP5DASH4, Email Note and Formula. CustomTabType Notes: • List will be presented to the signer as a drop down list on the signing document. The values for a List are set in the Name field as ";" separated items. • Checkbox cannot be set as a required field. • Number, Date, SSN, ZIP5, ZIP5DASH4, and Email fields will be validated for format before the signer can complete the signing. • Text, List, Date, Number, SSN, ZIP5, ZIP5DASH4, and Email characters will be limited by the size of the field set with CustomTabWidth and CustomTabHeight.
CustomTabWidth	Integer	Width of the custom tab.
CustomTabHeight	Integer	Height of the custom tab.
CustomTabRequired	Boolean	Signer is required to fill out the custom tab if true. If the *CustomTabType* is Checkbox, this must be set to false.

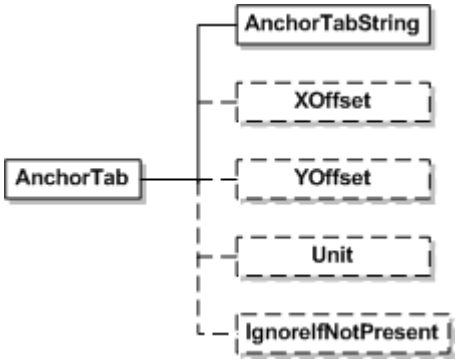
Name	Schema Type	Description
ConcealValueOnDocument	Boolean	Optional element. If true the field appears normally while the recipient is adding or modifying the information in the field, but the data is not visible (the characters are hidden by asterisks) to any other signer or the sender. When an envelope is completed the information is available to the sender through the Form Data link in the DocuSign Console. This setting applies only to text boxes and does not affect list boxes, radio buttons, or check boxes.
CustomTabLocked	Boolean	Signer cannot change the data of the custom tab if true. If the *CustomTabType* is Checkbox, this must be set to false.
CustomTabDisableAutoSize	Boolean	Disables the auto sizing of single line text boxes in the signing screen when the signer enters data. If disabled users will only be able enter as much data as the text box can hold. By default this is false. This property only affects single line text boxes.
CustomTabListItems	String	Semicolon separated items used in the dropdown list for a CustomTabType of “List”. These values may also be specified in the Name node; however, this node will take precedence. This can be a maximum of 2000 characters.
CustomTabListValues	String	For Form Field List Box type this is the semi-colon delimited list of values to go with the displayed list items set in CustomTabListItems or Name. These values are the hidden values of the drop down list display items. This can be a maximum of 2000 characters.
CustomTabListSelectedValue	String	Set this to the default selected item based on the *CustomTabListValues*. If Value is specified that will take precedence.
CustomTabRadioGroupName	String	This is the group name for a set of radio buttons. The CustomTabType must be “Radio”. This may also be specified in the TabLabel node; however, this node will take precedence.

Name	Schema Type	Description
CustomTabValidationPattern	String	For all custom tab types except radio buttons, checkboxes, and list items you may supply a regular expression that will be validated when data is entered in the field. This is optional and if not provided the default DocuSign validation rules will apply. Javascript RegEx object is used for regular expression validation. Regular expressions must be supported by this object to resolve.
CustomTabValidationMessage	String	Message to be displayed to the signer if the validation rule from CustomTabValidationPattern fails. This is optional and if not provided the default DocuSign message will be displayed. This can be a maximum of 250 characters.
TemplateLocked	Boolean	Optional element. Used only when working with template tabs. If true, the attributes of the tab cannot be changed by the sender.
TemplateRequired	Boolean	Optional element. Used only when working with template tabs. If true, the tab cannot be removed by the sender.
ConditionalParentLabel	String	Optional element. For conditional fields this is the TabLabel of the parent tab that controls this tab's visibility.
ConditionalParentValue	String	Optional element. For conditional fields this is the Value of the parent tab that controls this tab's visibility. If the parent tab is a Checkbox, Radio button, Optional Signature, or Optional Initial use "on" as the value to show that the parent tab is active.
SharedTab	Boolean	Optional element for field markup. When set to true, enables field markup for the field.
RequireInitialOnSharedTabChange	Boolean	Optional element for field markup. When set to true it requires the signer to initial when they modify a shared field.
Font	Font	The font type used for the information in the tab. Possible values are: Arial, ArialNarrow, Calibri, CourierNew, Garamond, Georgia, Helvetica, LucidaConsole, Tahoma, TimesNewRoman, Trebuchet, Verdana, OcrA, MSGothic, and MSMincho. **Note:** The MSGothic, and MSMincho font types are supported in requests, but in responses these font types are returned as Arial.
Bold	Boolean	If true, the information in the tab is bold.

Name	Schema Type	Description
Italic	Boolean	If true, the information in the tab is italic.
Underline	Boolean	If true, the information in the tab is underlined.
FontColor	FontColor	The font color used for the information in the tab. Possible values are: Black, BrightBlue, BrightRed, DarkGreen, DarkRed, Gold, Green, NavyBlue, Purple, or White.
FontSize	FontSize	The font size used for the information in the tab. Possible values are: Size7, Size8, Size9, Size10, Size11, Size12, Size14, Size16, Size18, Size20, Size22, Size24, Size26, Size28, Size36, Size48, or Size72.
MergeFieldXml	String	Reserved for future use.
IncludeNoteInEmail	Boolean	If true, the text in the CustomTabType "Note" tab is included in the email message (EmailBlurb) for the recipient.
IsPaymentField	Boolean	When true, sets this as a payment tab. Can only be used with CustomTabType Text, Number, List or Formula. The value of the tab must be a number.
Formula	String	The Formula string contains the TabLabel for the reference tabs used in the formula and calculation operators. Each TabLabel must be contained in brackets. This can be a maximum of 2000 characters. **Example:** Three tabs (TabLabels: Line1, Line2, and Tax) need to be added together. The formula string would be: <Formula>[Line1]+[Line2]+[Tax]</Formula>
RoundDecimalPlaces	Integer	Sets the number of digits after the decimal point in a CustomTabType Formula tab. Can only have the values of 0 or 2.
MaxLength	Integer	Sets the maximum number of characters a signer can enter in the tag. This is only active if CustomTabType is set to Text, Date, Number, SSN, ZIP5, ZIP5DASH4, or Email.
SenderRequired	Boolean	When true for a tab in a template, the sender must populate the tab before an envelope can be sent using the template. The value is only checked if the CustomTabType is List or Text. **Note:** This option does not function with the CreateEnvelopeFromTemplates method. If you want to send from a template that has this option enabled, you must use the CreateEnvelopeFromTemplatesAndForms method.

Name	Schema Type	Description
RequireAll	Boolean	Optional element associated with *SharedTab* element (field markup). When true and *SharedTab* is true, information must be entered in this field to complete the envelope.
TabOrder	nonNegativeInteger	A positive integer that sets the order the tab is navigated to during signing. Tabs on a page are navigated to in ascending Tab order value, starting with the lowest number and moving to the highest. If two or more tabs have the same Tab order number, the normal auto-navigation setting behavior for the envelope is used.

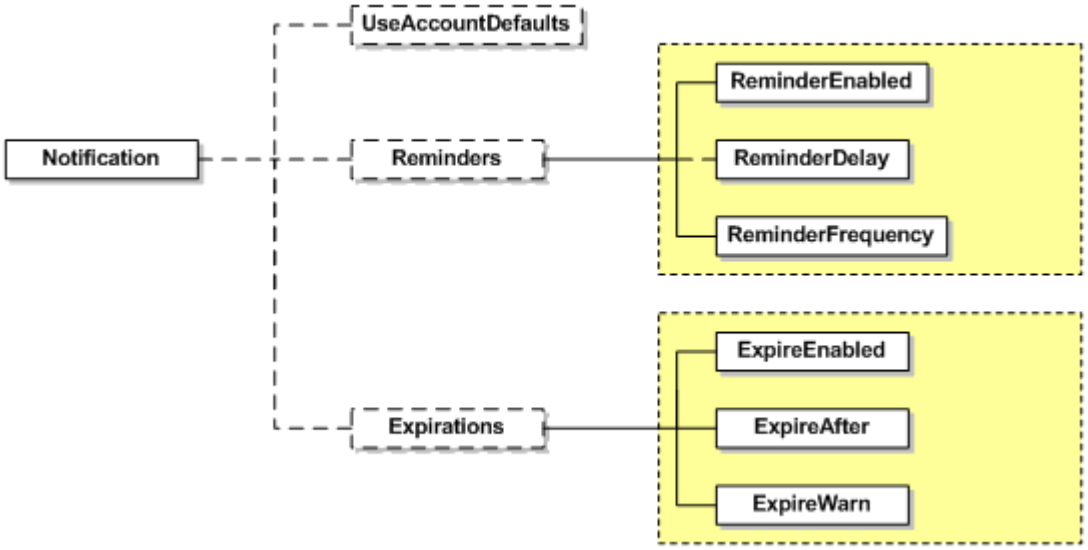
Anchor Tab



Note: When anchor tabs are used, all documents in the envelope are searched for the AnchorTabString.

Name	Schema Type	Description
AnchorTabString	string	Specifies string used as anchor tab in the document.
XOffset	double	This specifies tab location as XOffset position from the anchor tab.
YOffset	double	This specifies tab location as YOffset position from the anchor tab.
Unit	UnitTypeCode	Specifies units of the offset. Unit could be Pixels, Mms, Cms or Inches.
IgnoreIfNotPresent	Boolean	Ignore this tab if anchor is not found in the document.

Notification



Name	Schema Type	Description
UseAccountDefaults	Boolean	Indicates whether the Account defaults for notifications should be used on this envelope.
Reminders	Reminders	ReminderEnabled: Boolean: Whether or not a reminder email will be sent. ReminderDelay: Integer: Number of days AFTER a person receives an envelope, that they will start receiving reminder emails. ReminderFrequency: Integer: Number of days to wait between reminder emails.
Expirations	Expirations	ExpireEnabled: Boolean: Whether or not this envelope is set to expire. If false, the account default setting is used. If the account does not have an expiration setting, the DocuSign default value of 120 days is used. ExpireAfter: Integer: Number of days envelope will be active. ExpireWarn: Integer: Number of days before envelope expiration, that a warning email will be sent. If set to 0 (zero), no warning is sent.

EnvelopeNotification

Note that there is no configuration in the Member or Administrator tools for this feature. It is driven completely from the API and used in CreateEnvelope, CreateAndSendEnvelope, CreateFromTemplate, and CreateAndSendFromTemplate.

Name	Schema Type	Description
URL	URL	URL to the client-supplied listener page that consumes the status events. The listener page receives the envelope status XML messages from DocuSign. An example URL is: https://<serverName>/<path>/Listener.aspx
LoggingEnabled	Boolean	If true, the status events are recorded in a log on the DocuSign site for debugging purposes. Note that only the 10 latest messages are kept.
EnvelopeEvents	EnvelopeEvents	An array that consists of: EnvelopeStatusCode: The status code on which to receive event modifications. Values are Sent, Delivered, Completed, Declined, and Voided. IncludeDocuments: This is not used by the system.

EnvelopeStatus

EnvelopeStatus is the response element for CreateEnvelope and CreateAndSendEnvelope. See EnvelopeStatus in the Status and Managing Group for more information about this response.

Rules for CreateAndSendEnvelope and CreateEnvelope

API user specific rules

- The length of AccountId, document Name should not exceed 100 characters. Document password length should not exceed 50 characters.
- UserName, Email and access code of the recipient should not exceed 100 characters.
- SignatureName in SignatureInfo element should at least have one character. SignatureInitial should have a minimum 1 character and maximum 10 characters.
- Font styles supported for recipient signature are RageItalic, Mistral, BradleyHandITC, KaufmannBT, Freehand575 and LuciaBT.
- Allowed length for State attribute in address information element will be 2 characters. For Zip it will be 5 characters and for ZipPlus4 it will be 4 characters.
- Allowed length for SSN4 will be 4 characters and for SSN9 it will be 9 characters.
- The length of Label and Type for attachment element should not exceed 100 characters.
- CaptiveInfo element is made up of the optional ClientUserId and EmbeddedRecipientStartURL elements that 1) determine if a recipient is embedded or remote, and 2) express a redirect URL used to start an embedded recipient's signing session. ClientUserId is a sender created value that can be a maximum of 100 characters. If there is a value for this (it is not null) then the recipient is considered to be embedded. Note that if a ClientUserId is used and the account settings SignerMustHaveAccount or SignerMustLoginToSign are true, an error is generated on sending. EmbeddedRecipientStartURL is a sender provided valid URL string for redirecting the

recipient. When using this option, the embedded recipient still receives an email from DocuSign, just as a remote recipient would, but when the document link in the email is clicked the recipient is redirected, through DocuSign, to this URL to complete their actions. When routing to the URL, it is up to the sender's system (the server responding to the URL) to then request a recipient token to launch a signing session.

If the value SIGN_AT_DOCUSIGN is used for this node, the recipient will be directed to an embedded signing or viewing process directly at DocuSign. The signing or viewing action is initiated by the DocuSign system and the transaction activity and Certificate of Completion records will reflect this. In all other ways the process is identical to an embedded signing or viewing operation that would be launched by any partner.

It is important to remember that in a typical embedded workflow the authentication of an embedded recipient is the responsibility of the sending application and DocuSign expects that senders will follow their own process for establishing the recipient's identity. In this workflow the recipient goes through the sending application before the embedded signing or viewing process in initiated. However, when the sending application sets EmbeddedRecipientStartURL=SIGN_AT_DOCUSIGN, the recipient goes directly to the embedded signing or viewing process bypassing the sending application and any authentication steps the sending application would use. In this case, DocuSign recommends that one of the normal DocuSign authentication features (Access Code, Phone Authentication, SMS Authentication, etc.) be used to verify the identity of the recipient.

If a ClientUserId is NOT provided and an EmbeddedRecipientStartURL is provided, DocuSign will ignore the redirect URL and launch the standard signing process for the email recipient. Information can be appended to the EmbeddedRecipientStartURL using merge fields. The available merge fields items are: EnvelopeId, RecipientId, RecipientName, RecipientEmail, and CustomFields. The CustomFields must be part of the Recipient or Envelope. The merge fields are enclosed in double brackets.

Example:
http://senderHost/[[mergeField1]]/beginSigningSession?[[mergeField2]]&[[mergeField3]]

Rules for determining the brandId used in an envelope:

- If a brandId is specified in the envelope/template and that brandId is available to the account, that brand is used in the envelope.
- If more than one template is used in an envelope and more than one brandId is specified, the first brandId specified is used throughout the envelope.
- In cases where no brand is specified and the sender belongs to a Group; if there is only one brand associated with the Group, then that brand is used in the envelope. Otherwise, the account's default signing brand is used.
- For envelopes that do not meet any of the previous rules, the account's default signing brand is used in the envelope.

Rules for Exceptions thrown by the API

- AccountId specified while creating an envelope should exist else an exception with error message "Account_Does_Not_Exist_In_System" is thrown.
- DocumentID specified in the tab element should refer to a document in the envelope else exception with error message "Tab_Refers_To_Missing_Document" is thrown.

- RecipientID specified in the tab element should refer to an existing recipients ID attribute. Else exception is thrown with error message “Tab_Refers_To_Missing_Recipient”.
- Tab is affixed to a specific page number of the document. So the document specified by DocumentID in the Tab element should have page number specified by PageNumber. Else exception “Tab_Pagenumber_Is_Not_In_Document” is thrown.
- The document element in the CreateEnvelope and CreateAndSendEnvelope should contain the encoded document. Else exception “No_Document_Received” is thrown.
- The Email specified in the recipient element should exist else an exception is thrown with error message “Invalid_Email_Address_For_Recipient”.
- Recipient specified by UserName and Email pair in the recipient element must exist. Else exception with error message “Unknown_Envelope_Recipient” is thrown.
- If a signer recipient has any tabs available to them in the envelope they will be required to have at least one signable tab in at least one of the documents. Else exception with error message “Signer_Recipient_Has_No_Signable_Tabs” is thrown. If the signer recipient has no tabs available they will be considered a freeform signer and be required to place the tabs themselves.
- Access code is used to validate recipient identity. User has to enter the access code specified in the recipient element in the CreateEnvelope/CreateAndSendEnvelope to prove his/her identity. Recipient will need to validate his/her identity even if access code is not specified in the recipient element if RequireIDLookup attribute is set to true. But in this case the identity is verified not by entering access code, but by answering a questionnaire correctly. If the user fails to validate his/her identity then an exception is thrown with error message “Recipient_Has_Failed_Security_Check”.
- All recipients in an envelope should have a valid routing order else an exception with error message “Recipient_Has_Invalid_RoutingOrder” is thrown.
- There is a provision for creating custom tab. But in case of custom tab both the attributes Name and TabLabel should be specified. If only one of the attribute is specified in the Tab element then an exception “CustomTab_Is_Incomplete” is thrown.
- Account should have API permission to utilize CreateEnvelope and CreateAndSendEnvelope methods. Account needs to have “InSession permission” to send envelopes to InSession recipients. Else an exception with error message “Account_Lacks_Permissions” is thrown.
- AccountID mentioned should not be suspended by DocuSign Administrator; else exception will be thrown with message “Account_Has_Been_Suspended”.
- If the recipient Email specified in CreateAndSendEnvelope/CreateEnvelope is already reserved by another then processor throws an exception with error message “Email_Is_Reserved”.
- In CreateEnvelope and CreateAndSend Envelope, the account Id is validated for pricing plan, if the account does not have a pricing plan then “Account_Lacks_Pricing_Plan” exception is thrown
- InSession carbon copy recipients cannot be specified in CreateAndSendEnvelope. If the recipient Type is CC then ClientUserId has to be null else exception “Captive_Carbon_Copy_Recipient_Not_Supported” will be thrown.

- When angle brackets are detected in the packet submitted to one of the CreateEnvelope APIs an exception will be thrown with message “Invalid_characters_detected.”

Sample Code

CreateEnvelope – C#

```
// Create the recipient
DocuSignWeb.Recipient recipient = new DocuSignWeb.Recipient();
recipient.Email = "Test email";
recipient.UserName = "Testing account";
recipient.Type = DocuSignWeb.RecipientTypeCode.Signer;
recipient.ID = "1";
recipient.RoutingOrder = 1;

// Create the envelope content
DocuSignWeb.Envelope envelope = new DocuSignWeb.Envelope();
envelope.Subject = "Subject";
envelope.EmailBlurb = "Email content";
envelope.Recipients = new DocuSignWeb.Recipient[] { recipient };
envelope.AccountId = accountId;

// Attach the document(s)
envelope.Documents = new DocuSignWeb.Document[1];
DocuSignWeb.Document doc = new DocuSignWeb.Document();
doc.ID = "1";
doc.Name = "Document Name";
doc.PDFBytes = [Location of Document];
envelope.Documents[0] = doc;

// Create a new signature tab
DocuSignWeb.Tab tab = new DocuSignWeb.Tab();
tab.DocumentID = "1";
tab.RecipientID = "1";
tab.Type = DocuSignWeb.TabTypeCode.SignHere;
tab.PageNumber = "1";
tab.XPosition = "100";
tab.YPosition = "100";
envelope.Tabs = new DocuSignWeb.Tab[1];
envelope.Tabs[0] = tab;

// Create the envelope on the account -- it will be a draft
DocuSignWeb.EnvelopeStatus status = _apiClient.CreateEnvelope(envelope);

// An envelope ID indicates that it succeeded
Console.WriteLine("Envelope Id is {0}", status.EnvelopeID);
```

CreateEnvelope - PHP

```
// Create the recipient
$rcp1 = new Recipient();// First recipient to put in recipient array
$rcp1->UserName = "John Doe";
$rcp1->Email = $Recipient1Email;
$rcp1->Type = RecipientTypeCode::Signer;
$rcp1->ID = "1";
$rcp1->RoutingOrder = 1;
$rcp1->RequireIDLookup = FALSE;

// Create the envelope contents
$env = new Envelope();
$env->AccountId = $accountID; // Note: GUID should be used here rather than email
$env->Subject = "Subject";
$env->EmailBlurb = "testing docusign creation services";
```

```
$env->Recipients = array($rcp1);

// Attach the document
$doc = new Document();
$doc->ID = "1";
$doc->Name = "Picture PDF";
$doc->PDFBytes = file_get_contents("docs/picturePdf.pdf");
$env->Documents = array($doc);

// Create a new signature tab
$tab = new Tab();
$tab->DocumentID = "1";
$tab->RecipientID = "1";
$tab->Type = TabTypeCode::SignHere;
$tab->PageNumber = "1";
$tab->XPosition = "100";
$tab->YPosition = "100";
$env->Tabs = array($tab);

// Create a draft envelope on the account
$createEnvelopeparams = new CreateEnvelope();
$createEnvelopeparams->Envelope = $env;
$response = $api->CreateEnvelope($createEnvelopeparams);
```

CreateAndSendEnvelope – C#

```
// Create the recipient
DocuSignWeb.Recipient recipient = new DocuSignWeb.Recipient();
recipient.Email = "Test email";
recipient.UserName = "Testing account";
recipient.Type = DocuSignWeb.RecipientTypeCode.Signer;
recipient.ID = "1";
recipient.RoutingOrder = 1;

// Create the envelope content
DocuSignWeb.Envelope envelope = new DocuSignWeb.Envelope();
envelope.Subject = "Subject";
envelope.EmailBlurb = "Email content";
envelope.Recipients = new DocuSignWeb.Recipient[] { recipient };
envelope.AccountId = accountId;

// Attach the document(s)
envelope.Documents = new DocuSignWeb.Document[1];
DocuSignWeb.Document doc = new DocuSignWeb.Document();
doc.ID = "1";
doc.Name = "Document Name";
doc.PDFBytes = [Location of Document];
envelope.Documents[0] = doc;

// Create a new signature tab
DocuSignWeb.Tab tab = new DocuSignWeb.Tab();
tab.DocumentID = "1";
tab.RecipientID = "1";
tab.Type = DocuSignWeb.TabTypeCode.SignHere;
tab.PageNumber = "1";
tab.XPosition = "100";
tab.YPosition = "100";
envelope.Tabs = new DocuSignWeb.Tab[1];
envelope.Tabs[0] = tab;

// Create and send the envelope
DocuSignWeb.EnvelopeStatus status = _apiClient.CreateAndSendEnvelope(envelope);

// Examine the return status
```

```
Console.WriteLine("Envelope status is {0}", status.Status);
```

CreateAndSendEnvelope – PHP

```
// Create the recipient
$rcp1 = new Recipient();// First recipient to put in recipient array
$rcp1->UserName = "John Doe";
$rcp1->Email = $Recipient1Email;
$rcp1->Type = RecipientTypeCode::Signer;
$rcp1->ID = "1";
$rcp1->RoutingOrder = 1;
$rcp1->RequireIDLookup = FALSE;

// Create the envelope contents
$env = new Envelope();
$env->AccountId = $accountID; // Note: GUID should be used here rather than email
$env->Subject = "Subject";
$env->EmailBlurb = "testing docusign creation services";
$env->Recipients = array($rcp1);

// Attach the document
$doc = new Document();
$doc->ID = "1";
$doc->Name = "Picture PDF";
$doc->PDFBytes = file_get_contents("docs/picturePdf.pdf");
$env->Documents = array($doc);

// Create a new signature tab
$tab = new Tab();
$tab->DocumentID = "1";
$tab->RecipientID = "1";
$tab->Type = TabTypeCode::SignHere;
$tab->PageNumber = "1";
$tab->XPosition = "100";
$tab->YPosition = "100";
$env->Tabs = array($tab);

// Create and send the envelope
$createAndSendEnvelopeparams = new CreateEnvelope();
$createAndSendEnvelopeparams->Envelope = $env;
$response = $api->CreateAndSendEnvelope($createAndSendEnvelopeparams);
```

CreateEnvelopeFromTemplates

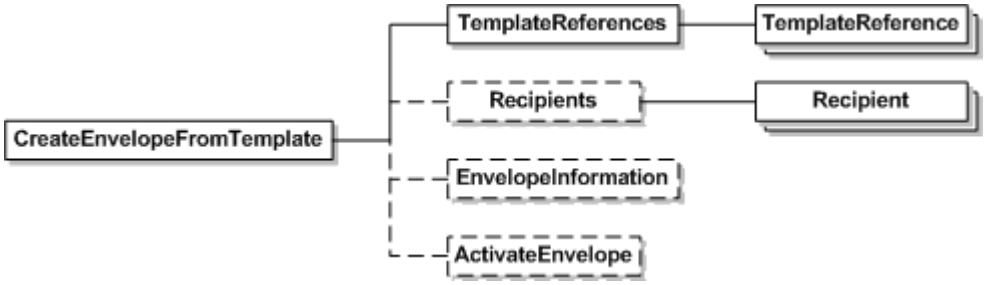
CreateEnvelopeFromTemplates uses templates as the basis for the definition of a signing ceremony. In contrast to the *CreateAndSendEnvelope* method, where the relationship of Tabs-to-Documents and Tabs-to-Recipients is passed in the request, *CreateEnvelopeFromTemplates* derives its Tab assignments from the pre-configured Template; and provides Recipient information to fulfill the Roles that are specified in the Template.

When provided one or more templates, along with their supplemental Document, Recipient and Envelope information, DocuSign will merge the various Template references into a single Envelope and send to the Recipients.

When a template is added or applied to an envelope and the template has a locked email subject and message (MessageLock is true), that subject and message is used for the envelope and cannot be changed even if another locked template is subsequently added or applied to the envelope. If an email subject or message is entered before adding or applying a locked template, the email subject and message will be overwritten with the email subject and message from the locked template.

Note: CreateEnvelopeFromTemplates does not reference the reminder and expiration notification settings from server templates. Reminder and Expiration Notification elements will need to be specified in the call or UseAccountDefaults will need to be used. The CreateEnvelopeFromTemplatesAndForms method does reference notifications from server templates.

Schema



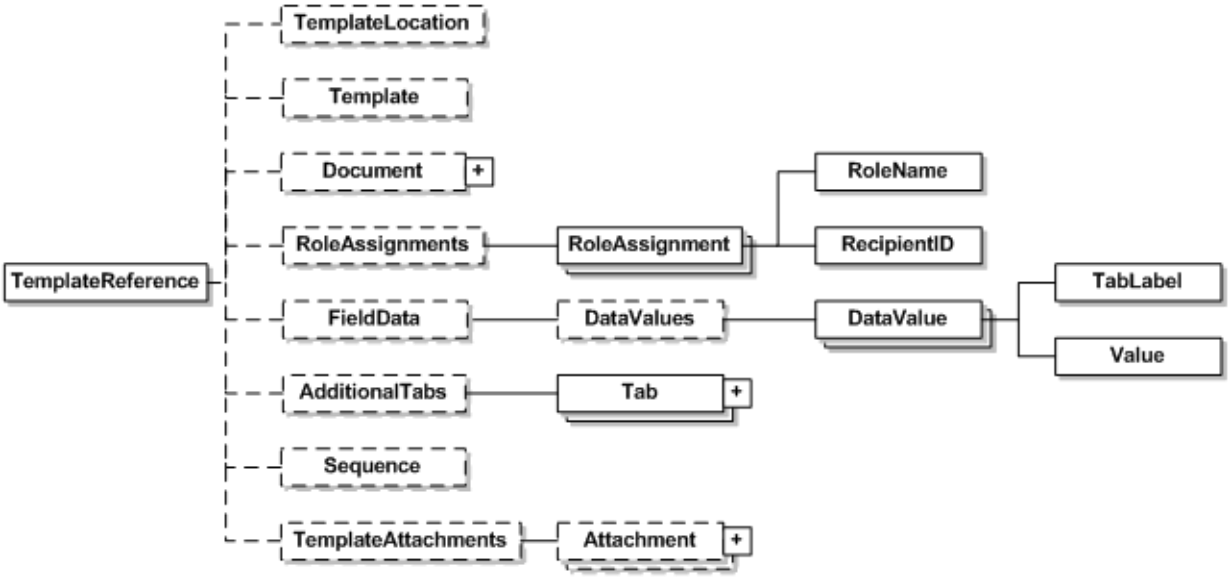
Name	Schema Type	Description
TemplateReferences	TemplateReference	Specifies the Templates that are included in this call. See the TemplateReference section below for more information.
Recipients	Recipient	Defines the Recipients that are included in this Envelope. These will be referenced from the RoleAssignment elements that are in each TemplateReference. The structure of a Recipient in the CreateEnvelopeFromTemplates method is similar to Recipients in other methods of the DocuSign Connect API. However, since these properties may also be specified in Templates that are referenced in the request, override rules are in place to determine the order of precedence of values that are specified in multiple places for like fields. For more information on the override rules of Recipient data over the roles specifications in the Template(s), see the override rules below. For Recipient definition refer to the Recipient section earlier in this document.
EnvelopeInformation	EnvelopeInformation	The structure and usage of these Envelope-level concepts are similar with their counterparts in other methods of the API. However, since these properties may also be specified in Templates that are referenced in the request, override rules are in place to determine the order of precedence of values that are specified in multiple places for like fields. For more information on the override rules see the EnvelopeInformation section below.

Name	Schema Type	Description
ActivateEnvelope	Boolean	Indicates whether the Envelope should be sent immediately or created as a Draft Envelope. Defaults to "True", where the Envelope will be sent.

TemplateReference

TemplateReferences contains an unbounded array of *TemplateReference* objects, each a self-contained structure used to satisfy the business rules of a given template, which will be merged into the parent Envelope.

Schema



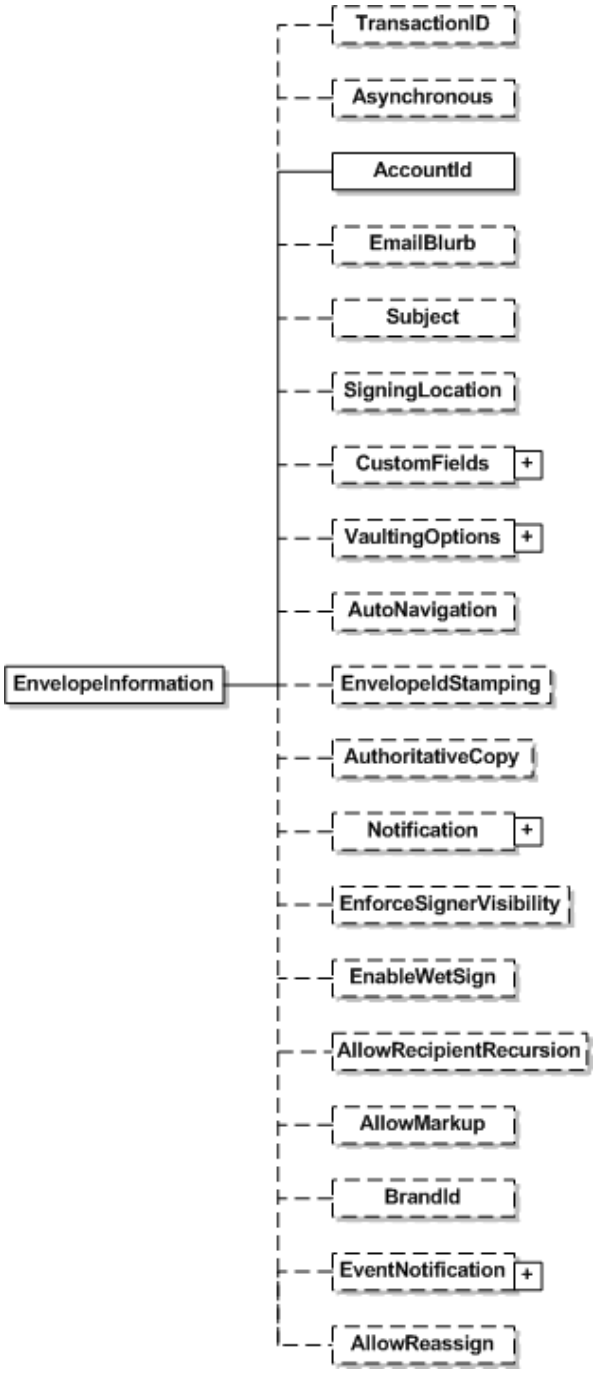
Name	Schema Type	Description
TemplateLocation	TemplateLocation	Enumerated list of options that describe where the Template information resides. Available options include: SOAP (default), PDFMetaData and Server.
Template	String	This element will contain different values, depending on the value of TemplateLocation: "Server": Template is the TemplateID of the template created in the web console. "SOAP": Template will include the Template XML in the format created by DocuSign Professional "PDFMetaData": Template can be left blank. It will be disregarded if provided. In this scenario, the Template xml is contained within the xml portion of the PDF document. In either case, the Document pdfBytes can be removed from the Template xml to optimize the payload size.

Name	Schema Type	Description
Document	Document	Refer to Document earlier in this document for details.
RoleAssignments	RoleAssignment	Specifies the RoleAssignments that are included in this call. RoleName – LongString - The name of the Role in the Template to which this Recipient information will be assigned RecipientID – positiveInteger - The Recipient identifier from the Recipients object that will fulfill this Role. It must be a positive integer and cannot be 0 (zero).
FieldData	FieldData	Provides the rules and/or data values that are used with the Template fields. See the FieldData section below for more information.
AdditionalTabs	Tab	Specifies any Tabs that should be added to a Document. AdditionalTabs enable the request to apply additional Tabs that are not configured in the Template. This would allow for run-time information to be specified as well as the template definitions that are specified at design-time. This could be used with an "open" Template that lacks pre-considered configuration in order to include ad hoc documents to the Envelope. Refer to Tab earlier in this document for details.
Sequence	NonNegativeInteger	Specifies the order that the Document(s) included in the Template should be presented in the Envelope. If not provided, the Documents will be presented in the order of the TemplateReferences.
TemplateAttachments	Attachment	A mechanism to provide the Document object through a SOAP attachment.

EnvelopeInformation

The structure and usage of these Envelope-level concepts are similar with their counterparts in other methods. However, since these properties may also be specified in Templates that are referenced in the request, override rules are in place to determine the order of precedence of values that are specified in multiple places for like fields. Refer to the rules section below for additional information.

Schema



All fields in the EnvelopeInformation schema, except for AllowRecipientRecursion, are defined in the Envelope section earlier in this document. The definition of AllowRecipientRecursion follows.

Name	Schema Type	Description
AllowRecipientRecursion	Boolean	When set to true, this enables the Recursive Recipients feature and allows a recipient to appear more than once in the routing order.

Template Email Subject Merge Fields

This provides the ability to insert recipient name and email address merge fields into the email subject line for envelopes sent using CreateEnvelopeFromTemplates.

The merge fields, based on the recipient's RoleName in the template, are added to the EnvelopeInformation Subject when the template is used to create an envelope. After a template sender adds the name and email information for the recipient and sends the envelope, the recipient information is automatically merged into the appropriate fields in the email subject line.

Both the sender and the recipients will see the information in the email subject line for any emails associated with the template. This provides an easy way for senders to organize their envelope emails by subject without having to open an envelope to check the recipient.

Note: If merging the recipient information into the subject line causes the subject line to exceed 100 characters, then any characters over the 100 character limit are not included in the subject line. For cases where the recipient name or email is expected to be long, you should consider placing the merge field at the start of the email subject.

- To add a recipient's name in the subject line add the following text in the EnvelopeInformation Subject when sending an envelope from a template:

  [[<*RoleName*>_UserName]]

  Example:

```
<EnvelopeInformation>
  <Subject>[[Signer 1_UserName]], Please sign this NDA </Subject>
</EnvelopeInformation>
```

- To add a recipient's email address in the subject line add the following text in the EnvelopeInformation Subject when sending an envelope from a template:

  [[<*RoleName*>_Email]]

  Example:

```
<EnvelopeInformation>
  <Subject>[[Signer 1_Email]], Please sign this NDA </Subject>
</EnvelopeInformation>
```

  In both cases the <*RoleName*> is the recipient's RoleName in the template.

For cases where another recipient (such as an Agent, Editor, or Intermediary recipient) is entering the name and email information for the recipient included in the email subject, then [[<*RoleName*>_UserName]] or [[<*RoleName*>_Email]] is shown in the email subject.

FieldData

FieldData provides information regarding the field data that will be correlated to the fields of the Template.

Name	Schema Type	Description
DataValues	DataValue	Provides explicit TabLabel/Value pairs that should be used for the field values in the Envelope. This value will take precedence over a similarly named PDF field if both are provided.

Rules for CreateEnvelopeFromTemplates

CreateEnvelopeFromTemplates refers to Templates to establish many of the properties of an Envelope. However, the *CreateEnvelopeFromTemplates* schema also overlaps the Template capabilities, so it is necessary to establish rules of usage and precedence to address incidents of data collision between the values in the SOAP request and the Template(s); or between the Template(s) themselves, since they are logically unrelated to each other. The table below describes the data override rules of the *CreateEnvelopeFromTemplates* schema.

Section	Element	Description
Recipient	ID	RecipientID is an identifier that is used to correlate the Recipient instance to the RoleAssignment. It must be a positive integer and cannot be 0 (zero).
Recipient	UserName	If allowable per the Template specification(s), this value will override the corresponding field for any role that this Recipient is assigned to.
Recipient	Email	If allowable per the Template specification(s), this value will override the corresponding field for any role that this Recipient is assigned to.
Recipient	Type	This value cannot override the value specified in the Template(s) for any role that this Recipient is assigned to.
Recipient	AccessCode	If allowable per the Template specification(s), this value will override the corresponding field for any role that this Recipient is assigned to. If conflicting, non-null values are provided in multiple roles to which this Recipient is assigned, an override value must be provided in this structure.
Recipient	RequireIDLookup	If allowable per the Template specifications, this value will override the corresponding field for any role that this Recipient is assigned to. If conflicting values are provided in multiple roles to which this Recipient is assigned, the value will resolve to True if it is not specified in this structure.
Recipient	IDCheckConfigurationName	If allowable per the Template specification(s), this value will override the corresponding field for any role that this Recipient is assigned to. If conflicting, non-null values are provided in multiple roles to which this Recipient is assigned, an override value must be provided in this structure.
Recipient	SignatureInfo .SignatureName .SignatureInitials .FontStyle	If allowable per the Template specification(s), these values will override the corresponding fields for any role that this Recipient is assigned to. If conflicting, non-null values are provided in multiple roles to which this Recipient is assigned, override values must be provided in this structures.

Section	Element	Description
Recipient	CaptiveInfo ClientUserId EmbeddedRecipient StartURL	If allowable per the Template specification(s), these values will override the corresponding field for any role that this Recipient is assigned to. If conflicting, non-null values are provided in multiple roles to which this Recipient is assigned, an override value must be provided in this structure.
Recipient	CustomFields	If allowable per the Template specification(s), this value will override a CustomField of the same name for any role that this Recipient is assigned to. If conflicting, non-null values are provided for the same CustomField name in multiple roles to which this Recipient is assigned, an override value must be provided in this structure.
Recipient	RoutingOrder	If allowable per the Template specification(s), this value will override the corresponding field for any role that this Recipient is assigned to. If conflicting, non-zero values are provided in multiple roles to which this Recipient is assigned, an override value must be provided in this structure.
Recipient	IDCheckInformation Input	This information cannot be pre-specified in a template, so the data must be provided in this structure.
Recipient	AutoNavigation	If allowable per the Template specifications, this value will override the corresponding field for any role that this Recipient is assigned to. If conflicting values are provided in multiple roles to which this Recipient is assigned, the value will resolve to True if it is not specified in this structure.
Recipient	Attachments	If allowable per the Template specification(s), this value will override an attachment of the same name for any role that this Recipient is assigned to. If values are provided for the same Attachment name in multiple roles to which this Recipient is assigned, an override value must be provided in this structure.
Recipient	Note	If allowable per the Template specification(s), this value will override the corresponding field for any role that this Recipient is assigned to. If conflicting, non-null values are provided in multiple roles to which this Recipient is assigned, an override value must be provided in this structure.

Section	Element	Description
EnvelopeInformation	EmailBlurb	If allowable per the Template specification(s), this value will override the corresponding field if provided in a referenced Template. If conflicting, non-null values are provided in multiple referenced Templates, an override value must be provided in this structure.
EnvelopeInformation	Subject	If allowable per the Template specification(s), this value will override the corresponding field if provided in a referenced Template. If conflicting, non-null values are provided in multiple referenced Templates, an override value must be provided in this structure.
EnvelopeInformation	SigningLocation	If allowable per the Template specification(s), this value will override the corresponding field if provided in a referenced Template. If conflicting, non-null values are provided in multiple referenced Templates, an override value must be provided in this structure.
EnvelopeInformation	CustomFields	If allowable per the Template specification(s), this value will override a CustomField of the same name if provided in a referenced Template. If conflicting, non-null values are provided in multiple referenced Templates for the same CustomField name, an override value must be provided in this structure.
EnvelopeInformation	VaultingOptions	If allowable per the Template specification(s), this value will override the corresponding field if provided in a referenced Template. If conflicting, non-null values are provided in multiple referenced Templates, an override value must be provided in this structure.
EnvelopeInformation	AutoNavigation	If allowable per the Template specification(s), this value will override the corresponding field if provided in a referenced Template. If conflicting, non-null values are provided in multiple referenced Templates, the value will resolve to True if it is not specified in this structure.
EnvelopeInformation	EnvelopeIdStamping	If allowable per the Template specification(s), this value will override the corresponding field if provided in a referenced Template. If conflicting, non-null values are provided in multiple referenced Templates, the value will resolve to True if it is not specified in this structure.

Section	Element	Description
EnvelopeInformation	AuthoritativeCopy	If allowable per the Template specification(s), this value will override the corresponding field if provided in a referenced Template. If conflicting, non-null values are provided in multiple referenced Templates, the value will resolve to True if it is not specified in this structure.

Template Execution Rules

The rules below are enforced when sending an Envelope based on a Template specification.

- If a Template role is specified as Optional, remove the signing instructions associated with the role, including all Tabs, if the role is not assigned in the *CreateEnvelopeFromTemplates* request.
- If any of the data override rules are violated (indicated by "must" statements in the Data Override descriptions), an exception is returned that describes the violation.

Rules for Mapping Data from PDF Forms

Data will be mapped from the incoming PDF form based on a direct correlation between the field name of the Form and the TabLabel of the DocuSign Tab. Fields will only be mapped if ***TransformPDFFields*** is set to true in the document node.

> **Note:** DocuSign will not transform PDF form fields that have the text "DocuSignIgnoreTransform" or "eSignIgnoreTransform" as part of the name of the PDF form field.

Mapping must be supported for static XFA-based PDF forms and AcroForms. The order of preference for establishing the form type is:

1. Derived on DocuSign server (transparent to client)
2. Pre-configured in a DocuSign structure (i.e. design-time configuration in a template)
3. Passed in via API request.

Rule #	Description
1	When a PDF form is submitted, the DocuSign mapping component will interrogate the form for any fields and apply the data from those fields to DocuSign Form Fields of the same name.
2	Fields from XFA-based PDF forms will be evaluated by their fully-qualified field name (ex: "form/customer/address/line1[0]")
3	The DocuSign component will map any fields that match (per the requirement above). Any fields that do not have a matching field (i.e. field exists in form but not in Envelope; or field exists in Envelope but not in form) will be disregarded by the mapping component.
4	DocuSign form fields are limited in scope to the document that they pertain to.
5	Edit rules for the fields must be pre-configured for the DocuSign fields (ex: editable vs. locked, required vs. optional, edit masks). Edit rules will not be derived by the DocuSign mapping component.
6	The following field types are supported for data mapping: text boxes, check boxes, radio buttons, drop-down lists, text areas. Regarding drop-down lists and radio buttons, the data mapping is for the selected value only. The list of available values must be pre-configured in the DocuSign fields.

Error Rules

All errors from CreateAndSendEnvelope and CreateEnvelope apply in addition to the following:

- Template_Cannot_Add_Document – error thrown if the template location is set to PDFMetaData and the PDF bytes within the Document node of the template reference are invalid.
- Template_Not_Provided – error thrown if the template XML cannot be found in the PDFMetaData or TemplateReference Template node.
- Template_Unable_To_Load – error thrown if the template provided cannot be loaded.
- Template_Cannot_Add_Document – error thrown if the template and document cannot be loaded.
- Template_Unable_To_Load_FieldData – error thrown if the PDF form data exists in the PDF and it cannot be extracted.
- Template_Unable_To_Process_FieldData – error thrown if the PDF form data exists and it cannot be mapped properly to the recipients DocuSign Tab data.
- Template_Unable_To_Flatten_PDF – error thrown if the PDF form data exists and it cannot be flattened.
- Template_RoleSpecified_Does_Not_Exist – error thrown if a recipient is mapped to a role that does not exist in the template.
- Template_RecipientID_For_Role_NotFound – error thrown if a recipient ID is mapped to a role and the recipient ID does not exist in the template.
- Template_Required_Recip_Not_Satisfied – error thrown if there is a required recipient in the template and no recipient/role mapping satisfied the recipient.
- Template_To_Envelope_Error – error thrown if the template provided cannot be transformed to a DocuSign envelope.
- Template_Override_EnvelopeInformation_Error – error thrown if any items in the EnvelopeInformation data cannot be applied to the envelope.
- Template_AdditionalTabs_Error – error thrown if an items in the AdditionalTabs data cannot be applied to the envelope.
- Template_Override_RecipientData_Error – error thrown if an items in the Recipient data that should override the template data cannot be applied to the envelope.

Sample Code

CreateEnvelopeFromTemplates– C#

```
// Create the recipient information
DocuSignWeb.Recipient recipient = new DocuSignWeb.Recipient();
recipient.Email = "Test email";
recipient.UserName = "Testing account";
recipient.Type = DocuSignWeb.RecipientTypeCode.Signer;
recipient.ID = "1";
recipient.RoutingOrder = 1;
DocuSignWeb.Recipient[] recipients = new DocuSignWeb.Recipient[] { recipient };

// Create the template reference from a server-side template ID
DocuSignWeb.TemplateReference templateReference = new DocuSignWeb.TemplateReference();
```

```
templateReference.Template = "Template ID";
templateReference.TemplateLocation = DocuSignWeb.TemplateLocationCode.Server;

// Construct the envelope information
DocuSignWeb.EnvelopeInformation envelopeInfo = new DocuSignWeb.EnvelopeInformation();
envelopeInfo.AccountId = _accountId;
envelopeInfo.Subject = "Subject";
envelopeInfo.EmailBlurb = "Email content";

// Create draft with all the template information
DocuSignWeb.EnvelopeStatus status = _apiClient.CreateEnvelopeFromTemplates(new
    DocuSignWeb.TemplateReference[] { templateReference }, recipients,
    envelopeInfo, false);

// Confirm that the envelope has been assigned an ID
Console.WriteLine("Status for envelope {0} is {1}", status.EnvelopeID, status.Status);
```

CreateEnvelopeFromTemplates– PHP

```
// Create the recipient
$rcp1 = new Recipient();// First recipient to put in recipient array
$rcp1->UserName = "John Doe";
$rcp1->Email = $Recipient1Email;
$rcp1->Type = RecipientTypeCode::Signer;
$rcp1->ID = "1";
$rcp1->RoutingOrder = 1;
$rcp1->RequireIDLookup = FALSE;

// Use a server-side template
$templateRef = new TemplateReference();
$templateRef->TemplateLocation = TemplateLocationCode::Server;
// TODO: Replace string with the GUID of a template already uploaded to your account
$templateRef->Template = "put in a template GUID here";

// Construct the envelope info
$envInfo = new EnvelopeInformation();
$envInfo->AccountId = $AccountID;
$envInfo->Subject = "create envelope from templates test";
$envInfo->EmailBlurb = "testing docusign create services";

// Send creates draft with all the template info
$createEnvelopeFromTemplatesparams = new CreateEnvelopeFromTemplates();
$createEnvelopeFromTemplatesparams->TemplateReferences = array($templateRef);
$createEnvelopeFromTemplatesparams->Recipients = array($rcp1)
$createEnvelopeFromTemplatesparams->EnvelopeInformation = $envInfo;
$createEnvelopeFromTemplatesparams->ActivateEnvelope = false;
$response = $api->CreateEnvelopeFromTemplates($createEnvelopeFromTemplatesparams);
```

CreateEnvelopeFromTemplatesAndForms

This method is used to create envelopes from a combination of a PDF form and a DocuSign Template. Once all the data from the form is overlaid on the template the envelope is passed to the *CreateAndSendEnvelope* or *CreateEnvelope* method. Refer to *CreateAndSendEnvelope* method for the sending rules that are applied before the envelope can be sent.

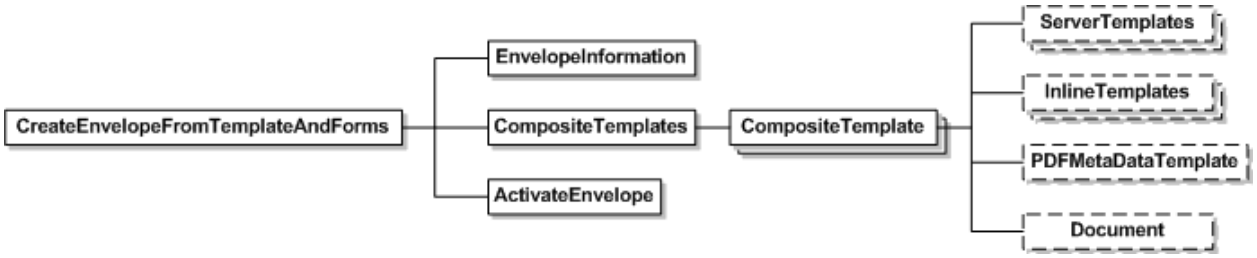
This method has some similarities to the *CreateEnvelopeFromTemplates* method, in that both merge documents and templates, with the primary differences to the *CreateEnvelopeFromTemplatesAndForms* method being:

- DocuSign Tabs are extrapolated from the form itself. Instead of having to define the fields on a Template in advance, they can be derived from the form on the fly.

- Templates, which are really just partially-defined Envelopes, can be stacked over each other, creating a *Composite* Template.
- Templates do not need to be fully defined. For example, a template can be created to define only those items that cannot be derived from the form itself.

When a template is added or applied to an envelope and the template has a locked email subject and message (MessageLock is true), that subject and message is used for the envelope and cannot be changed even if another locked template is subsequently added or applied to the envelope. If an email subject or message is entered before adding or applying a locked template, the email subject and message will be overwritten with the email subject and message from the locked template.

Schema



Name	Schema Type	Description
EnvelopeInformation	EnvelopeInformation	The envelope information structure defining envelope level attributes to be used. This data will override any data that is supplied in the templates. This information is unchanged from the current schema.

Name	Schema Type	Description
CompositeTemplates	CompositeTemplate	Each CompositeTemplate adds a new document and template overlays into the envelope. There can be any number of CompositeTemplate structures. Each CompositeTemplate consists of: ServerTemplates - 0 or more server-side templates and their position in the overlay. If supplied they are overlaid into the envelope in the order of their Sequence value. InlineTemplates - 0 or more inline templates and their position in the overlay. If supplied they are overlaid into the envelope in the order of their Sequence value. PDFMetaDataTemplate – 0 or 1 embedded template in the PDF meta data. If supplied the PDF meta data template will be overlaid into the envelope in the order of their Sequence value. Document - 0 or 1 document. The document is used to specify a different document than the ones in the overlay templates. The PDF fields from this document will be transformed to DocuSign Secure Fields if the TransformPdfFields option is set to true. **Note:** DocuSign will not transform PDF form fields that have the text "DocuSignIgnoreTransform" or "eSignIgnoreTransform" as part of the name of the PDF form field. If PDFMetaDataTemplate is supplied this is the document the template is extracted from. This information is unchanged from the current schema. For rules associated with the CompositeTemplate see the following section.
ActivateEnvelope	Boolean	Indicates whether the envelope should be sent immediately after being created or if it is created as a Draft Envelope. When set to “True” the envelope will be sent after being created.

Rules for Composite Template Usage

Each CompositeTemplate adds a new document and templates overlay into the envelope. For each CompositeTemplate these rules are applied:

- Templates are overlaid in the order of their Sequence value.

- If Document is not passed into the system, the first template's document (based on template's Sequence value) is used.
- Last in wins in all cases except for the document (i.e. envelope information, recipient information, secure field information). This was done to keep things simple. There is no special casing.

  For example, if you want higher security on a tab, then that needs to be specified in the last template in which the tab is included. If you want higher security on a role recipient it needs to be in the last template in which that role recipient is specified.
- Recipient matching is based on Recipient Role and Routing Order. If there are matches, the recipient information is merged together. A final pass is done on all CompositeTemplates, after all template overlays have been applied, to collapse recipients with the same email, username and routing order. This prevents having the same recipients at the same routing order.
- If you specify in a template that a recipient is locked, once that recipient is overlaid the recipient attributes can no longer be changed. The only items that can be changed for the recipient in this case are the email, username, access code and IDCheckInformationInput..
- Tab matching is based on Tab Labels, Tab Types and Documents. If a Tab Label matches but the Document is not supplied, the Tab is overlaid for all the Documents.

  For example, if you have a simple inline template with only one tab in it with a label and a value, the Signature, Initial, Company, Envelope ID, UserName tabs will only be matched and collapsed if they fall in the exact same X and Y locations.
- RoleName and TabLabel matching is case sensitive.
- A DefaultRecipient node has been introduced so that you can specify which recipient the generated tabs from the PDF form are mapped to. You can also set PDF form generated tabs to a recipient other than the DefaultRecipient by specifying the mapping of the tab label that is created to one of the template recipients (see the following example).

  You can also use tab label wild carding to map a series of tabs from the PDF form. To use this you must end a tab label with "*" and then the system matches tabs that start with the label (see the following example).
- If no DefaultRecipient is specified, tabs must be explicitly mapped to recipients in order to be generated from the form. Unmapped form objects will not be generated into their DocuSign equivalents. (In the case of Signature/Initials, the tabs will be disregarded entirely; in the case of pdf text fields, the field data will be flattened on the Envelope document, but there will not be a corresponding DocuSign data tab.)
- Only the FieldTypes and FieldProperties listed below are extrapolated from the forms:

  FieldTypes that are extrapolated are: CheckBox, DateTime, ListBox, Numeric, Radio, Text, Signature, and Password.

Note:

When extrapolating Adobe Digital Signatures, the following Adobe names correspond to DocuSign names:

Adobe name contains DocuSignSignHere or eSignSignHere = DocuSign Signature

Adobe name contains DocuSignSignHereOptional or eSignSignHereOptional = DocuSign Optional Signature

Adobe name contains DocuSignInitialHere or eSignInitialHere = DocuSign Initials

Adobe name contains DocuSignInitialHereOptional or eSignInitialHereOptional = DocuSign Optional Initials

Any other Adobe name will default to DocuSign Signature.

When extrapolating Adobe text fields, the following Adobe names correspond to DocuSign names:

Adobe name contains DocuSignSignHere or eSignSignHere = DocuSign Signature

Adobe name contains DocuSignSignHereOptional or eSignSignHereOptional = DocuSign Optional Signature

Adobe name contains DocuSignInitialHere or eSignInitialHere = DocuSign Initials

Adobe name contains DocuSignInitialHereOptional or eSignInitialHereOptional = DocuSign Optional Initials

Adobe name contains DocuSignEnvelopeID or eSignEnvelopeID = DocuSign EnvelopeID

Adobe name contains DocuSignCompany or eSignCompany = DocuSign Company

*Adobe name contains DocuSignDateSigned or eSignDateSigned = DocuSign DateSigned

Adobe name contains DocuSignTitleor eSignTitle = DocuSign Title

Adobe name contains DocuSignFullNameor eSignFullName = DocuSign FullName

Adobe name contains DocuSignSignerAttachmentOptional or eSignSignerAttachmentOptional = DocuSign Optional Signer Attachment

Any other name will default to a DocuSign data (text) field

Note: DocuSign will not transform PDF form fields that have the text "DocuSignIgnoreTransform" or "eSignIgnoreTransform" as part of the name of the PDF form field.

* Adobe date fields can be transformed to DocuSignDateSigned fields using the same naming scheme.

FieldProperties that are extrapolated are: ReadOnly, Required, MaxLength, Positions, and Initial Data.

Example:

This example XML posted to our API would create an envelope draft where:

- The document included is the document used. The form fields are extracted from the document and turned into secure fields.
- The server template is applied first.
- The 2 inline templates are applied 2nd and 3rd.

  In the first inline template, the tabs fld1 and fld2* are matched against the tabs and the form fields that were converted and their values are placed into the matching tabs. The tab fld2* finds all tabs that have a label that starts with fld2.
- The PDF metadata template is applied 4th.

```
<CreateEnvelopeFromTemplatesAndForms xmlns="http://www.docusign.net/API/3.0">
  <EnvelopeInformation>
    <TransactionID></TransactionID>
    <Asynchronous>false</Asynchronous>
    <AccountId>ACCOUNT ID</AccountId>
    <EmailBlurb>Test CreateEnvelopeFromTemplatesAndForms</EmailBlurb>
    <Subject>Test CreateEnvelopeFromTemplatesAndForms</Subject>
    <SigningLocation>Online</SigningLocation>
    <CustomFields>
      <CustomField>
        <Name>FromEnvelope</Name>
        <Show>true</Show>
        <Required>false</Required>
        <Value>FromEnvelope</Value>
        <CustomFieldType>Text</CustomFieldType>
      </CustomField>
    </CustomFields>
    <AutoNavigation>true</AutoNavigation>
    <EnvelopeIdStamping>true</EnvelopeIdStamping>
    <AuthoritativeCopy>false</AuthoritativeCopy>
    <EnforceSignerVisibility>false</EnforceSignerVisibility>
    <EnableWetSign>false</EnableWetSign>
    <AllowRecipientRecursion>true</AllowRecipientRecursion>
    <AllowMarkup>false</AllowMarkup>
  </EnvelopeInformation>
  <CompositeTemplates>
    <CompositeTemplate>
      <ServerTemplates>
        <ServerTemplate>
          <Sequence>1</Sequence>
          <TemplateID>TEMPLATE ID</TemplateID>
        </ServerTemplate>
      </ServerTemplates>
      <InlineTemplates>
        <InlineTemplate>
          <Sequence>2</Sequence>
          <Envelope>
            <Recipients>
              <Recipient>
                <ID>1</ID>
                <UserName>User Name 1</UserName>
                <Email>user.name_1@myemail.com </Email>
                <Type>Signer</Type>
                <AccessCode />
                <RequireIDLookup>false</RequireIDLookup>
                <RoutingOrder>1</RoutingOrder>
                <Note />
                <RoleName>SignerOne</RoleName>
                <DefaultRecipient>true</DefaultRecipient>
              </Recipient>
              <Recipient>
                <ID>2</ID>
                <UserName>User Name 2</UserName>
                <Email>user.name_2@myemail.com</Email>
                <Type>Signer</Type>
                <AccessCode />
                <RequireIDLookup>false</RequireIDLookup>
                <RoutingOrder>2</RoutingOrder>
                <Note />
                <RoleName>SignerTwo</RoleName>
              </Recipient>
              <Recipient>
                <ID>3</ID>
                <UserName>User Name 3</UserName>
```

```
            <Email>user.name_3@myemail.com</Email>
            <Type>Signer</Type>
            <AccessCode />
            <RequireIDLookup>false</RequireIDLookup>
            <RoutingOrder>1</RoutingOrder>
            <Note />
            <RoleName>SignerThree</RoleName>
          </Recipient>
          <Recipient>
            <ID>4</ID>
            <UserName>User Name 4</UserName>
            <Email>user.name_4@myemail.com </Email>
            <Type>Signer</Type>
            <AccessCode />
            <RequireIDLookup>false</RequireIDLookup>
            <RoutingOrder>2</RoutingOrder>
            <Note />
            <RoleName>SignerOne</RoleName>
          </Recipient>
        </Recipients>
        <Tabs>
          <Tab>
            <DocumentID>1</DocumentID>
            <RecipientID>1</RecipientID>
            <PageNumber>1</PageNumber>
            <XPosition>12</XPosition>
            <YPosition>10</YPosition>
            <ScaleValue>1.00000</ScaleValue>
            <Type>SignHere</Type>
            <Name>SignHere</Name>
            <TabLabel>Sign Here 1</TabLabel>
            <Value />
            <CustomTabHeight>0</CustomTabHeight>
            <CustomTabRequired>false</CustomTabRequired>
            <CustomTabLocked>false</CustomTabLocked>
            <CustomTabDisableAutoSize>false</CustomTabDisableAutoSize>
            <TemplateLocked>false</TemplateLocked>
            <TemplateRequired>false</TemplateRequired>
          </Tab>
          <Tab>
            <RecipientID>2</RecipientID>
            <TabLabel>fld1</TabLabel>
            <Value>From Inline Template</Value>
            <Type>Custom</Type>
          </Tab>
          <Tab>
            <RecipientID>2</RecipientID>
            <TabLabel>fld2\*</TabLabel>
            <Value>Wild card from Inline Template</Value>
            <Type>Custom</Type>
          </Tab>
        </Tabs>
        <Subject>Sample Form</Subject>
        <EmailBlurb />
        <SigningLocation>Online</SigningLocation>
        <CustomFields>
          <CustomField>
            <Name>Custom Field</Name>
            <Show>True</Show>
            <Required>True</Required>
            <Value />
          </CustomField>
          <CustomField>
            <Name>Custom Field List</Name>
```

```
            <Show>True</Show>
            <Required>True</Required>
            <Value />
            <CustomFieldType>List</CustomFieldType>
            <ListItems>One; Two; Three</ListItems>
          </CustomField>
        </CustomFields>
        <AutoNavigation>false</AutoNavigation>
        <EnvelopeIdStamping>false</EnvelopeIdStamping>
        <AuthoritativeCopy>false</AuthoritativeCopy>
        <Notification>
          <Reminders>
            <ReminderEnabled>false</ReminderEnabled>
            <ReminderDelay>0</ReminderDelay>
            <ReminderFrequency>0</ReminderFrequency>
          </Reminders>
          <Expirations>
            <ExpireEnabled>true</ExpireEnabled>
            <ExpireAfter>120</ExpireAfter>
            <ExpireWarn>0</ExpireWarn>
          </Expirations>
        </Notification>
        <EnableWetSign>true</EnableWetSign>
      </Envelope>
    </InlineTemplate>
    <InlineTemplate>
      <Sequence>3</Sequence>
      <Envelope>
        <Recipients>
          <Recipient>
            <ID>1</ID>
            <UserName>User Name 1</UserName>
            <Email>user.name_1@email.com</Email>
            <Type>Signer</Type>
            <AccessCode />
            <RequireIDLookup>false</RequireIDLookup>
            <RoutingOrder>1</RoutingOrder>
            <Note />
            <RoleName>SignerOne</RoleName>
            <DefaultRecipient>true</DefaultRecipient>
          </Recipient>
        </Recipients>
        <Tabs>
          <Tab>
            <DocumentID>1</DocumentID>
            <RecipientID>1</RecipientID>
            <PageNumber>2</PageNumber>
            <XPosition>12</XPosition>
            <YPosition>10</YPosition>
            <ScaleValue>1.00000</ScaleValue>
            <Type>SignHere</Type>
            <Name>SignHere</Name>
            <TabLabel>Sign Here 1</TabLabel>
            <Value />
            <CustomTabHeight>0</CustomTabHeight>
            <CustomTabRequired>false</CustomTabRequired>
            <CustomTabLocked>false</CustomTabLocked>
            <CustomTabDisableAutoSize>false</CustomTabDisableAutoSize>
            <TemplateLocked>false</TemplateLocked>
            <TemplateRequired>false</TemplateRequired>
          </Tab>
        </Tabs>
      </Envelope>
    </InlineTemplate>
```

```
      </InlineTemplates>
      <PDFMetaDataTemplate>
        <Sequence>4</Sequence>
      </PDFMetaDataTemplate>
      <Document>
        <ID>1</ID>
        <Name>Form Document</Name>
        <PDFBytes>PDF_BYTES_GO_HERE</PDFBytes>
        <TransformPdfFields>true</TransformPdfFields>
        <FileExtension>pdf</FileExtension>
      </Document>
    </CompositeTemplate>
    <!--
    Add another composite template to the mix
    <CompositeTemplate>
      <ServerTemplates>
      ....
      <InlineTemplates>
      ...
      </InlineTemplates>
      <PDFMetaDataTemplate>
      ...
      </PDFMetaDataTemplate>
      <Document>
        ...
      </Document>
    </CompositeTemplate>
    -->
  </CompositeTemplates>
  <ActivateEnvelope>false</ActivateEnvelope>
</CreateEnvelopeFromTemplatesAndForms>
```

EnvelopeInformation

This is the Envelope-level information that overlays the information from the Templates. The structure and usage of these Envelope-level concepts are unchanged from the current schema.

Template Email Subject Merge Fields

This provides the ability to insert recipient name and email address merge fields into the email subject line for envelopes sent using CreateEnvelopeFromTemplatesAndForms.

The merge fields, based on the recipient's RoleName in the template, are added to the EnvelopeInformation Subject when the template is used to create an envelope. After a template sender adds the name and email information for the recipient and sends the envelope, the recipient information is automatically merged into the appropriate fields in the email subject line.

Both the sender and the recipients will see the information in the email subject line for any emails associated with the template. This provides an easy way for senders to organize their envelope emails by subject without having to open an envelope to check the recipient.

Note: If merging the recipient information into the subject line causes the subject line to exceed 100 characters, then any characters over the 100 character limit are not included in the subject line. For cases where the recipient name or email is expected to be long, you should consider placing the merge field at the start of the email subject.

- To add a recipient's name in the subject line add the following text in the EnvelopeInformation Subject when sending an envelope from a template:

  [[<*RoleName*>_UserName]]

Example:

```
<EnvelopeInformation>
  <Subject>[[Signer 1_UserName]], Please sign this NDA </Subject>
</EnvelopeInformation>
```

- To add a recipient's email address in the subject line add the following text in the EnvelopeInformation Subject when sending an envelope from a template:

  [[<*RoleName*>_Email]]

  Example:

```
<EnvelopeInformation>
  <Subject>[[Signer 1_Email]], Please sign this NDA </Subject>
</EnvelopeInformation>
```

  In both cases the <*RoleName*> is the recipient's RoleName in the template.

For cases where another recipient (such as an Agent, Editor, or Intermediary recipient) is entering the name and email information for the recipient included in the email subject, then [[<*RoleName*>_UserName]] or [[<*RoleName*>_Email]] is shown in the email subject.

Document

The document is used to specify a different document than the ones in the overlay templates. This structure and usage of the Document-level concepts are unchanged from the current schema.

Recipient

The Recipient element is used to specify envelope recipients. This structure and usage of the Recipient-level concepts are similar to the current schema with the addition off the DefaultRecipient element.

Name	Schema Type	Description
DefaultRecipient	Boolean	This specifies which recipient the generated tabs from the PDF from would be mapped to. When set to "True" this recipient is the default recipient and generated tabs are mapped to this recipient.

ServerTemplate

Sets the server-side templates that are overlaid in the envelope and the order in which the template is used.

Name	Schema Type	Description
Sequence	positiveInteger	Sets the order in which the template is applied.
TemplateID	String	Unique identifier of the template.

InlineTemplate

Sets the inline template envelopes that are applied and the order in which it is used.

Name	Schema Type	Description

Name	Schema Type	Description
Sequence	positiveInteger	Sets the order in which the template is applied.
Envelope	Envelope	The inline template envelope information.

PDFMetaDataTemplate

Sets the PDF embedded templates that are overlaid into the envelope and the order in which it is used.

Name	Schema Type	Description
Sequence	positiveInteger	Sets the order in which the template is applied.

The naming convention for PDF Metadata is:

- PDF Metadata Prefix = DocuSign
- PDF Metadata Schema = Envelope

The embedded template needs to be in DocuSign server template format.

For more information about using the XMP layer to include MetaData, refer to the Adobe XMP Specification document Adding Intelligence to Media.

Sample Code

CreateEnvelopeFromTemplatesAndForms – C#

```
// Configure the envelope information
DocuSignWeb.EnvelopeInformation envelopeInfo = new DocuSignWeb.EnvelopeInformation();
envelopeInfo.AccountId = _accountId;
envelopeInfo.EmailBlurb = "testing docusign creation services";
envelopeInfo.Subject = "create envelope from templates and forms test";
DocuSignWeb.CompositeTemplate template = new DocuSignWeb.CompositeTemplate();

DocuSignWeb.Recipient recipient1 = new DocuSignWeb.Recipient();
recipient1.UserName = "SignerOne";
// TODO: replace the email string with an actual email
recipient1.Email = "test email one";
recipient1.Type = DocuSignWeb.RecipientTypeCode.Signer;
recipient1.RequireIDLookup = false;
recipient1.RequireIDLookupSpecified = true;
recipient1.RoutingOrder = 1;
recipient1.RoutingOrderSpecified = true;
recipient1.RoleName = "One";
recipient1.ID = "1";

DocuSignWeb.Recipient recipient2 = new DocuSignWeb.Recipient();
recipient2.UserName = "Signer2";
// TODO: replace the email string with an actual email
recipient2.Email = "test email two";
recipient2.Type = DocuSignWeb.RecipientTypeCode.Signer;
recipient2.RequireIDLookup = false;
recipient2.RequireIDLookupSpecified = true;
recipient2.RoutingOrder = 2;
recipient2.RoutingOrderSpecified = true;
recipient2.RoleName = "Two";
recipient2.ID = "2";

DocuSignWeb.Recipient[] signers = { recipient1, recipient2 };
```

```
// Configure the inline templates
DocuSignWeb.InlineTemplate inlineTemplate = new DocuSignWeb.InlineTemplate();
inlineTemplate.Sequence = "1";
inlineTemplate.Envelope = new DocuSignWeb.Envelope();
inlineTemplate.Envelope.Recipients = signers;
inlineTemplate.Envelope.AccountId = _accountId;

// This tab matches the DateSigned tab assigned to recipient one
DocuSignWeb.Tab tab1 = new DocuSignWeb.Tab();
tab1.RecipientID = "1";
tab1.TabLabel = "DocuSignDateSignedOne";
tab1.Type = DocuSignWeb.TabTypeCode.DateSigned;

// This tab matches the SignHere tabs assigned to recipient two
DocuSignWeb.Tab tab2 = new DocuSignWeb.Tab();
tab2.RecipientID = "2";
tab2.TabLabel = "SignTwo\\*";
tab2.Type = DocuSignWeb.TabTypeCode.SignHere;

// This tab matches the SignHere tabs assigned to recipient one
DocuSignWeb.Tab tab3 = new DocuSignWeb.Tab();
tab3.RecipientID = "1";
tab3.TabLabel = "SignOne\\*";
tab3.Type = DocuSignWeb.TabTypeCode.SignHere;

// This tab matches the DateSigned tab assigned to recipient two
DocuSignWeb.Tab tab4 = new DocuSignWeb.Tab();
tab4.RecipientID = "2";
tab4.TabLabel = "DocuSignDateSignedTwo";
tab4.Type = DocuSignWeb.TabTypeCode.DateSigned;

// This tab matches nothing -- but that's okay!
// It will just get discarded
DocuSignWeb.Tab tab5 = new DocuSignWeb.Tab();
tab5.RecipientID = "1";
tab5.TabLabel = "asdf";
tab5.Type = DocuSignWeb.TabTypeCode.FullName;

inlineTemplate.Envelope.Tabs = new DocuSignWeb.Tab[] { tab1, tab2, tab3, tab4, tab5 };

template.InlineTemplates = new DocuSignWeb.InlineTemplate[] { inlineTemplate };

// Configure the document
template.Document = new DocuSignWeb.Document();
template.Document.ID = "1";
template.Document.Name = "Form Document";
template.Document.PDFBytes = <Test Document>;
template.Document.TransformPdfFields = true;
template.Document.FileExtension = "pdf";

// Create draft with all the composite template information
DocuSignWeb.EnvelopeStatus status =
    _apiClient.CreateEnvelopeFromTemplatesAndForms(envelopeInfo,
    new DocuSignWeb.CompositeTemplate[] { template }, false);
```

CreateEnvelopeFromTemplatesAndForms – PHP

```
// Configure and envelope information
$envInfo = new EnvelopeInformation();
$envInfo->AccountId = $AccountID;
$envInfo->EmailBlurb = "testing docusign creation services";
$envInfo->Subject = "create envelope from templates and forms sample";
```

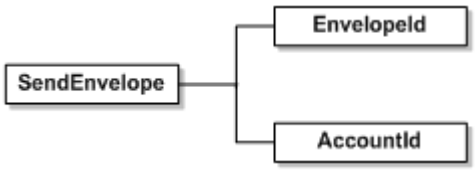
```
$recipient1 = new Recipient();
$recipient1->UserName = "SignerOne";
// TODO: replace email string with actual email
$recipient1->Email = "test email 1";
$recipient1->Type = RecipientTypeCode::Signer;
$recipient1->RequireIDLookup = FALSE;
$recipient1->RoutingOrder = 1;
$recipient1->RoleName = "One";
$recipient1->ID = "1";

$recipient2 = new Recipient();
$recipient2->UserName = "SignerTwo";
// TODO: replace email string with actual email
$recipient2->Email = "test email 2";
$recipient2->Type = RecipientTypeCode::Signer;
$recipient2->RequireIDLookup = FALSE;
$recipient2->RoutingOrder = 2;
$recipient2->RoleName = "Two";
$recipient2->ID = "2";

$signers = array($recipient1, $recipient2);

// Build template
$inlineTemplate = new InlineTemplate();
$inlineTemplate->Sequence = "1";
$env = new Envelope();
$env->Recipients = $signers;
$env->AccountId = $AccountID;

// This tab matches the DateSigned tab assigned to recipient one
$tab1 = new Tab();
$tab1->RecipientID = "1";
$tab1->TabLabel = "DocuSignDateSignedOne";
$tab1->Type = TabTypeCode::DateSigned;

// This tab matches the SignHere tabs assigned to recipient two
$tab2 = new Tab();
$tab2->RecipientID = "2";
$tab2->TabLabel = "SignTwo\\*";
$tab2->Type = TabTypeCode::SignHere;

// This tab matches the SignHere tabs assigned to recipient one
$tab3 = new Tab();
$tab3->RecipientID = "1";
$tab3->TabLabel = "SignOne\\*";
$tab3->Type = TabTypeCode::SignHere;

// This tab matches the DateSigned tab assigned to recipient two
$tab4 = new Tab();
$tab4->RecipientID = "2";
$tab4->TabLabel = "DocuSignDateSignedTwo";
$tab4->Type = TabTypeCode::DateSigned;

// This tab matches nothing -- but that's okay!
// It will just get discarded
$tab5 = new Tab();
$tab5->RecipientID = "1";
$tab5->TabLabel = "asdf";
$tab5->Type = TabTypeCode::FullName;

$env->Tabs = array($tab1, $tab2, $tab3, $tab4, $tab5);

$inlineTemplate->Envelope = $env;
$template = new CompositeTemplate();
```

```
$template->InlineTemplates = array($inlineTemplate);

// Configure the document
$doc = new Document();
$doc->ID = "1";
$doc->Name = "Form Document";
$doc->PDFBytes = file_get_contents("docs/LoremIpsum.pdf");
$doc->TransformPdfFields = true;
$doc->FileExtension = "pdf";
$template->Document = $doc;

// Send
$params = new CreateEnvelopeFromTemplatesAndForms();
$params->EnvelopeInformation = $envInfo;
$params->CompositeTemplates = array($template);
$params->ActivateEnvelope = false;
$response = $api->CreateEnvelopeFromTemplatesAndForms($params);
```

SendEnvelope

This method is used to send draft envelopes. Refer to *CreateAndSendEnvelope* method for the sending rules that will be applied before the envelope can be sent.

Schema



Name	Schema Type	Description
EnvelopeId	DSXId	Envelope Id to be sent for signature.
AccountId	DSXId	Account Id of the user who created the envelope

Sample Code

SendEnvelope – C#

```
// Create envelope as shown in linked code
DocuSignWeb.Envelope envelope = <create envelope>;

// Send the envelope
DocuSignWeb.EnvelopeStatus sendStatus = _apiClient.SendEnvelope(status.EnvelopeID,
_accountId);
// Examine the return status
Console.WriteLine("Envelope status is {0}", sendStatus.Status);
```

SendEnvelope – PHP

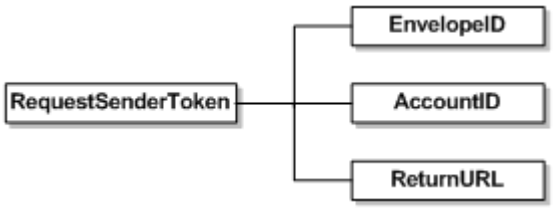
```
// Create envelope as shown in linked code
$response = <create envelope>;
$envStatus = $response->CreateEnvelopeResult;

// Send
$sendEnvelopeparams = new SendEnvelope();
$sendEnvelopeparams->AccountId = $AccountID;
```

```
$sendEnvelopeparams->EnvelopeId = $envStatus->EnvelopeID;
$response = $api->SendEnvelope($sendEnvelopeparams);
```

RequestSenderToken

This method is used to get a onetime use login token that allows the user to be placed into the DocuSign sending wizard. Upon sending completion the user is returned to the return URL provided by the API application.

Schema



Name	Schema Type	Description
EnvelopeID	string	Optional envelope ID. If provided the user will be placed into the envelope in process. If left blank the user will be placed into a new envelope for sending.
AccountID	String	DocuSign account ID to map the envelope to.
ReturnURL	String	URL to send the user to upon completion of the sending process. The URL will have an event passed to it as a query parameter. The parameter will be named “event”, see “In-session sending events” below for description of the valid events. The DocuSign Envelope Id will also be returned in the “envelopeId” parameter. **Important** – You must include HTTPS:// in the URL or the redirect might be blocked by some browsers.

In-Session Sending Events

Events returned on the ReturnURL as a query string parameter. The event is sent in the parameter named “event”. The ReturnURL is called with the event on any completion state of the envelope send. Valid events can be found in the schema type InSessionSendEvent in the DocuSign API XSD.

Event	Description
Send	User sends the envelope
Save	User saves a draft of the envelope
Cancel	User cancels the sending transaction. No envelope ID returned.
Error	Error performing the send.
SessionEnd	Sending session ended before the user completed.

Rules and Exceptions RequestSenderToken

- User_Does_Not_Exist_In_System – API user requesting the sending token URL does not have access to the system.
- Account_Does_Not_Exist_In_System – Account is not valid.
- Account_Lacks_Permissions – API user requesting the sending token URL does not have valid API permissions.
- User_Lacks_Membership – user is not a valid member of the account.
- User_Not_Envelope_Sender – user is not a valid sender in the account.
- Envelope_Does_Not_Exist – if envelope ID is provided, envelope ID does not exist.
- Account_Not_Authorized_For_Envelope – if envelope ID is provided, account is not valid for the envelope.

Anchor Based Tagging

This section outlines the usage rules and behaviors of the DocuSign Anchor-Based Tagging feature. Anchor-Based Tagging enables a DocuSign customer to send documents for signature that do not have a fixed layout or format. In these documents, you can not anticipate the absolute location of the tabs when you design your API client application because the tabs must move with text. As an alternative to sending X and Y coordinates for tabs, the DocuSign Signing Service derives anchor tab locations by correlating XML instructions from the SOAP request to data within the document.

Using Anchor Tabs

Anchor tab solutions require a coordinated map between the Anchor data and the document(s) in the CreateEnvelope/CreateAndSendEnvelope SOAP request. When the DocuSign Signing Service receives a request that contains anchor tabs, it searches the document for instances of the AnchorTabString. When found, it places a tab of the specified type for the designated recipient. Tab positions are established by committing the Anchor offset values relative to the lower left point of the AnchorTabString.

> **Note:** When anchor tabs are used, all documents in the envelope are searched for the AnchorTabString.

When you apply tabs to the document, DocuSign does not remove or replace the AnchorTabString. You can hide codified anchors by using the same font color as the background of the document. While the anchor can be used by DocuSign processes, it will not be visible on the document.

To create an anchor tab:

1. Identify the location in the document by text string. Use either a preexisting text string or add a new one.

   For example, “Borrower’s Signature” might already exist in the document. If not, you could add the text string, “BorrowersSignHere”.

2. Reference the anchor through the AnchorTabString element of the SOAP request of the CreateEnvelope/CreateAndSendEnvelope request.
3. Determine the offset from the AnchorTabString location to where the tab should be placed. The bottom-left of the AnchorTabString is equivalent to position (0,0), and the bottom-left of

the tab graphic is placed relative to that. Positive XOffset values move the tab right on the page and positive YOffset values move the tab down the page.

DocuSign does not currently provide tools to derive the offset values. Determination of the proper offset will likely require some trial-and-error.

Rules for Anchor Tagging

There are both API user specific rules and rules for the exceptions thrown by the processor.

API user specific rules

- The Tab node must contain the RecipientID which refers to an existing recipient and Tab Type (signature, initial etc). Else, the XML validation fails and a Validation error exception is thrown.
- If the Anchor node is initialized, then the AnchorTabString must be specified. Else, the XML validation fails and a Validation error exception is thrown.
- The AnchorTabString specified in the SOAP envelope must be included at least once in the corresponding document of the envelope.
- API users who do not use the anchor-based tabs do not make any changes.
- The user can decide the format of the AnchorTabString. DocuSign tabs are created for each instance of the AnchorTabString within the document, so special care must be taken to establish unique AnchorTabStrings that do not result in unintentional tabs.
- The API user only needs to list the anchor tab in the XML once and map it to a recipient and tab type. The tab is placed for the recipient in each instance that the anchor tab appears in the document(s).
- The API user cannot use the same anchor tab for different recipients for the same document.
- The API user cannot use a text string in an Image PDF.
- The allowed offset units are: pixels, millimeters, centimeters or inches. If not specified, the default unit is pixels.
- X or Y offsets supplied for a tab apply to all instances of the tab in the document.
- To use different offsets at different locations in the document for the same recipient, create multiple, unique anchor tabs.

Rules for Exceptions thrown by the API

- The anchor tab specified by the user must exist in the corresponding document of the envelope. Else, the processor throws an exception with the error message “Anchor_Tab_String_Not_Found”. The error message includes the list of anchor tabs that are not found.
- If the Y offset value would force a tab outside of the page boundaries, the tag will be placed at the page boundary. If the X offset value places a tab outside of the page boundaries, the error message “Invalid_User_Offset” is sent. The error message includes the X offset that resulted in the error.
- The API user must not use the same tab for different recipients for the same document. Nor can the API user use the same tab for different tab types. After the tab is used once,

the processor throws an exception with the error “Invalid_Anchor_Tab_String”. The error message indicates that the user can not map an anchor tab for different recipients or tab types.

Known Limitations

- The processor cannot search text that is embedded in an image.
- The processor does not support an AnchorTabString embedded in the form of a PDF X-object in the document.
- The processor does not re-flow the text that surrounds the anchor tabs. It is the responsibility of the document author to provide sufficient whitespace to contain the potential width of the ultimate tab value.

Tips and Tricks

The following are tips for effective use of Anchor Based Tagging:

- In order to avoid unintentional conflicts between AnchorTabStrings and the text that naturally exists in documents, establish a codified syntax for AnchorTabStrings that is unlikely to otherwise appear in a document.
- Develop an extensible and consistent syntax that can be used across multiple document types. i.e. [RecipientIdentifier.TabType.TabQualifier]. Examples of this syntax might include [R1.InitialHere] or [CoSign1.Custom.SSN].
- Especially for documents that have variable numbers of tabs or signers (ex: optional Co-signers), author the source document to include hidden anchor tabs for all potential signers/permutations. Then, control the tabs that are actually placed by including/excluding the Anchors in the XML request. This approach allows a single document to be used for all use cases instead of maintaining separate documents for each scenario.

Embedding Function Group

This section describes the principles and methods involved in implementing a DocuSign Connect API integration using the Embedded Signing interaction pattern for recipients. In this section, we specify methods, processes and exceptions that deviate from typical API integration concepts.

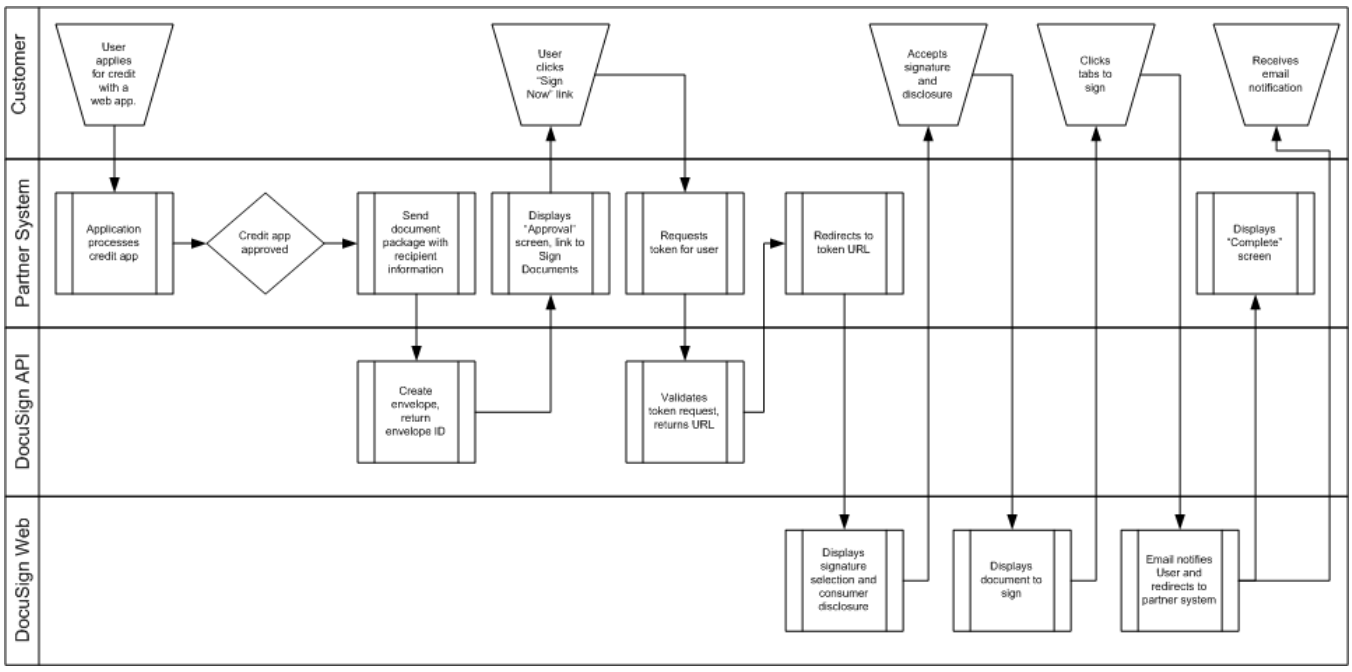
Embedded Signing is a departure from the default “Remote Signer” pattern, in which DocuSign brokers the communication with envelope Recipients via system-generated emails containing links to activate the viewing and signing process. Embedded Signing, in contrast, enables a API client application to maintain its connection with envelope Recipients by incorporating the DocuSign Signing Service directly into its process flow. The functional result is a more fluid document transaction and more transparent feature extension of the client application.

This tighter integration with DocuSign imposes additional technical and functional burdens on the client application, including addressing security, legal, and user experience requirements that are engineered into the DocuSign application in the Remote Signer pattern.

Important: iFrames should not be used for embedded signing on mobile devices due to screen space issues. For iOS devices DocuSign recommends using a WebView.

Embedded Signing Functional Process Flow

This overview of a typical embedded document-signing transaction should familiarize DocuSign Connect API users with terms and process. It is not a faithful reproduction of how the process is implemented on the web site or in the API. In this use case, a web user applies for credit from an online lending source that provides an instant credit decision and electronic signature services.



Captive Recipients

The Embedded Signing pattern introduces a new class of recipients referred to as Captive Recipients. *Captive* refers to the exclusive relationship between the recipient and the sending account. In contrast, *non-Captive Recipients* are DocuSign-global entities that can have recipient relationships with an unlimited number of senders. Characteristics of Captive Recipients are that they:

- Are identified by their FirstName, LastName, Email, sending account, and optionally a sender-provided ClientUserId.
- Have an exclusive relationship with their sending account. If another DocuSign customer has sent documents to the same person, either as a Remote or Captive Recipient, the recipient information is not related in any way.
- Can access DocuSign content only via their sending account's application; they cannot login directly to DocuSign.
- Do not have access to the DocuSign Member Console.
- Do not receive email notifications from DocuSign, except in cases specifically called out in this document.
- Cannot be aware that they have an account in DocuSign.

Captive Recipients can co-exist with Remote Recipients on a single envelope, but they cannot switch modalities after they are created. A Captive Recipient cannot be converted to a Remote Recipient at any point in the process. This is important to recognize during the client solution design phase, as DocuSign must know how a Recipient will interact with the client application at the time the envelope is created.

DocuSign Integration

Three distinct operational areas characterize Embedded Signing API integration:

- **Pre-DocuSign Operations** – Operations within the client application relating to creating the Envelope and navigating the Captive Recipient into the DocuSign-hosted pages.

  This section includes information on requesting and using recipient tokens (RequestRecipientToken and RequestRecipientFaxToken).
- **DocuSign Operations** – Operations include displaying the Electronic Record and Signature Disclosure, Signature display and selection.
- **Post-DocuSign Operations** – Client-hosted pages that the Captive Recipient is redirected to upon reaching a terminal state in the DocuSign user experience.

Pre-DocuSign Operations and Requesting Recipient Tokens

Creating the envelope

Creating envelops for Embedded Recipients parallels the Remote Recipient model. But the sender needs to specify if the recipient is Remote or Embedded. Sender can do this by using ClientUserId. A non-null ClientUserId implies that user is Embedded. A single envelope can contain both Embedded and Remote Recipients.

Requesting Recipient Tokens (RequestRecipientToken and RequestRecipientFaxToken)

DocuSign constrains access to the Embedded signature process via short-lived Recipient Tokens. Successful token requests return an URL that will invoke the DocuSign signing wizard for a particular envelope and Embedded Recipient. Envelopes must be sent before a recipient token can be requested (recipient tokens will not work with draft envelopes).

Important: iFrames should not be used for embedded operations on mobile devices due to screen space issues. For iOS devices DocuSign recommends using a WebView.

There are two types of recipient tokens available:

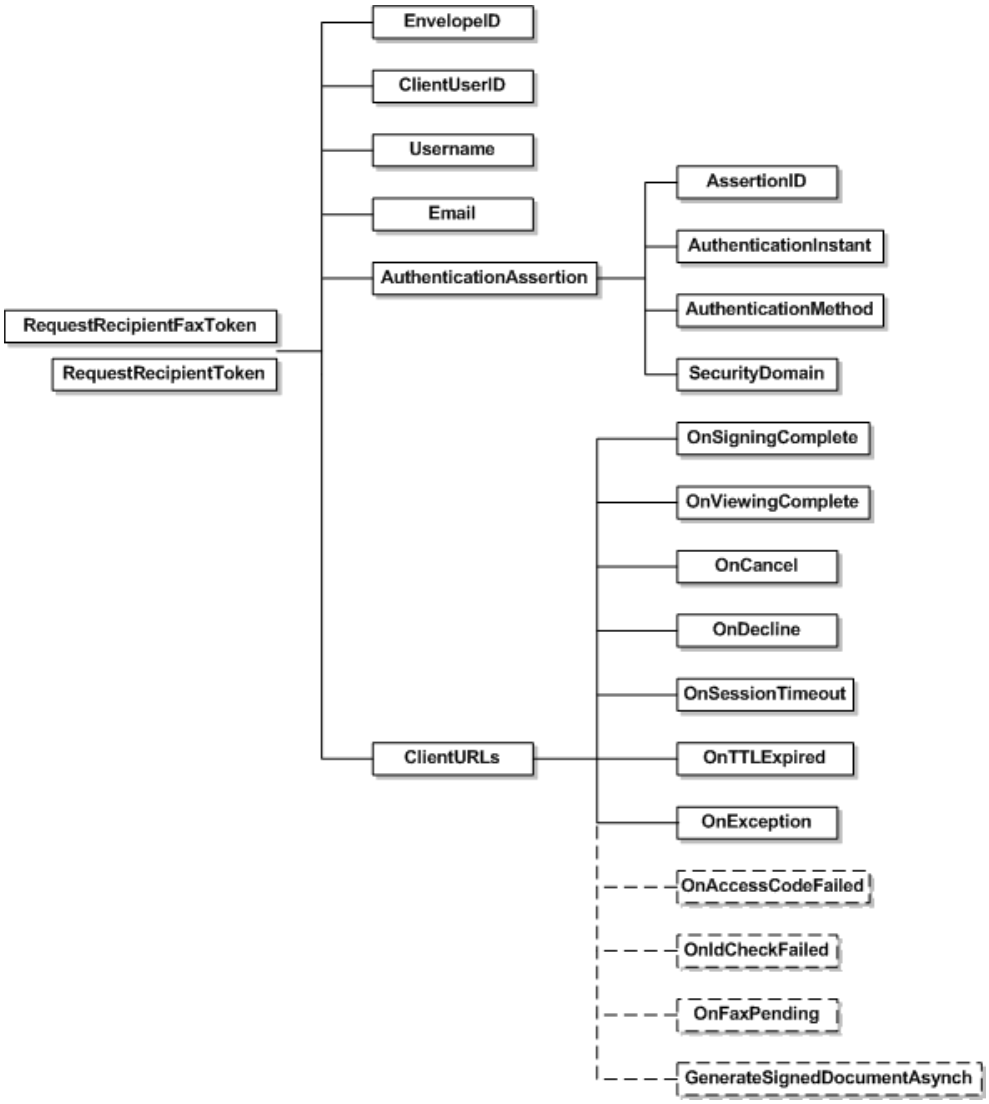
- RequestRecipientToken: This token returns a URL to invoke a DocuSign signing session where the signer completes the signing process online.
- RequestRecipientFaxToken: This token returns a URL to invoke a DocuSign signing session where the signer is required to print out and sign documents. The documents are then returned to DocuSign by fax or upload.

> **Note:** An alternative to using RequestRecipientFaxToken is to use RequestRecipientFaxPDF to return a PDF version of a fax cover page and documents in an envelope rather than returning a URL for a signing session. For more information about this option see RequestRecipientFaxPDF.

This section describes the token methods and the behaviors.

The RequestRecipientToken and RequestRecipientFaxToken methods use the format and rules, and are divided into three sections: Entity Identification, Authentication Assertions and InSessionCallBackURL.

Schema



Entity Identification

The following parameters identify the Envelope and Captive Recipient for which the Recipient Token is requested:

Note: When requesting a Recipient Token, the ID in ClientUserID must use upper-case letters. If lower-case letters are used (ClientUserId) an error will occur.

- **EnvelopeID** – The DocuSign-generated value contained in the EnvelopeStatus object returned from the CreateAndSendEnvelope method. This parameter identifies the Envelope.
- **ClientUserID** – The value that specifies if the recipient is Remote or Embedded
- **UserName** – Specifies the username of the recipient
- **Email** - Specifies the Email of the recipient.

The Username and Email identifies a unique recipient and ClientUserID specifies if the recipient is Embedded or Remote. All the three parameter together identifies a Captive Recipient. All values must exist and correlate to a valid Envelope/Embedded Recipient entity relationship in order for the RequestRecipientToken request to succeed.

Authentication Assertions

DocuSign acts as a Single Sign-On consumer to the API client application in the Embedded Signing model. That is, DocuSign trusts that the client application authenticated the end-user sufficiently to establish their identity and offer documents to sign or view. This is critical to the seamless integration between the two applications, and it is necessary since DocuSign has no interaction with the end-user prior to displaying the documents. However, even though the trust relationship does not include a DocuSign-verification of the user identity, DocuSign requires the client application to assert that it has authenticated the end-user through some means. This assertion is communicated through the following parameters:

- **AssertionID** – A unique identifier of the authentication event executed by the client application.
- **AuthenticationInstant** – The date/time that the end-user was authenticated.
- **AuthenticationMethod** – The convention used to authenticate the end-user.
- **SecurityDomain** – The domain to which the user authenticated.

An entry is added into the Security Level section of the DocuSign Signing Certificate that reflects the “SecurityDomain – AuthenticationMethod” used to verify the user identity. Additional identity verification within DocuSign is optionally available through the ID Check and Access Code features.

ClientURLs

Client application will have the option of setting Client URLs to be called on specific DocuSign events. DocuSign redirects the Recipient to the URL.

If you want an “event” parameter or any other parameters, you MUST provide them in the callback URL. No parameters are added by DocuSign. The valid values are listed in Embedded Callback Event Codes.

XML Signing

It is recommended that embedded clients sign the body of the RequestRecipientToken message with a valid X.509 certificate. This process supports the Single Sign-On trust relationship between

DocuSign and the client application. Client organizations must provide DocuSign with the certificate's Common Name (CN) during the implementation phase so that DocuSign can validate the XML signature. Supported certificate authorities are VeriSign and Thawte.

Expiration

- Recipient Tokens expire five minutes after they are issued by DocuSign. If a Recipient Token URL is invoked after it is expired, the user is re-directed to the Callback URL specified in the RequestRecipientToken request with event code TTLExpired.
- Recipient Tokens expire upon being successfully invoked.
- Active Recipient Tokens expire if the envelope is voided.
- Five minutes is the default "Time to Live" for Recipient tokens. This is a configurable setting.

Signing vs. Viewing Documents

Recipient Tokens must be requested (and returned) in order to sign or to view envelope documents. The DocuSign web application renders the view that is appropriate to the state of the envelope for the Recipient. The logic is as follows:

- If Recipient has uncompleted signing actions, DocuSign will present documents in Sign mode.
- If Recipient has completed all signing actions, DocuSign will present documents in View mode.
- If Recipient has uncompleted signing actions and the envelope has been declined by another Recipient, then the remaining recipients will not be able to view the envelope.
- If the sender has voided the envelope no recipient will be able to view or sign document.

Sample Code

RequestRecipientToken – C#

```
// Create envelope as shown in linked code
DocuSignWeb.Envelope envelope = <create envelope>;

// Need to specify captive info for these recipients
envelope.Recipients[0].CaptiveInfo = new DocuSignWeb.RecipientCaptiveInfo();
envelope.Recipients[0].CaptiveInfo.ClientUserId = "User4521";

DocuSignWeb.EnvelopeStatus status = _apiClient.SendEnvelope(envelope);

// Construct the recipient token authentication assertion
// Specify ID, start time, method and domain
DocuSignWeb.RequestRecipientTokenAuthenticationAssertion assertion
    = new DocuSignWeb.RequestRecipientTokenAuthenticationAssertion();
assertion.AssertionID = new Guid().ToString();
assertion.AuthenticationInstant = DateTime.Now;
assertion.AuthenticationMethod
    =
DocuSignWeb.RequestRecipientTokenAuthenticationAssertionAuthenticationMethod.Password;

assertion.SecurityDomain = "Request Recipient Token Test";

// Construct the URLs based on username
DocuSignWeb.Recipient recipient = envelope.Recipients[0];
DocuSignWeb.RequestRecipientTokenClientURLs urls = new
DocuSignWeb.RequestRecipientTokenClientURLs();
String urlBase = baseURL;
urls.OnSigningComplete = urlBase + "?event=SignComplete&uname=" + recipient.UserName;
```

```
urls.OnViewingComplete = urlBase + "?event=ViewComplete&uname=" + recipient.UserName;
urls.OnCancel = urlBase + "?event=Cancel&uname=" + recipient.UserName;
urls.OnDecline = urlBase + "?event=Decline&uname=" + recipient.UserName;
urls.OnSessionTimeout = urlBase + "?event=Timeout&uname=" + recipient.UserName;
urls.OnTTLExpired = urlBase + "?event=TTLExpired&uname=" + recipient.UserName;
urls.OnIdCheckFailed = urlBase + "?event=IDCheck&uname=" + recipient.UserName;
urls.OnAccessCodeFailed = urlBase + "?event=AccessCode&uname=" + recipient.UserName;
urls.OnException = urlBase + "?event=Exception&uname=" + recipient.UserName;

// Request the token for a specific recipient
String token = _apiClient.RequestRecipientToken(status.EnvelopeID,
    recipient.CaptiveInfo.ClientUserId, recipient.UserName,
    recipient.Email, assertion, urls);

// Display token
Console.WriteLine("Recipient token is {0}", token);
```

RequestRecipientToken: PHP

```
// Create envelope as shown in linked code
$env = <create envelope>;

// Specify captive info for recipients
$captiveInfo = new RecipientCaptiveInfo();
$captiveInfo->ClientUserId = "User4521";
$env->Recipients[0]->CaptiveInfo = $captiveInfo;

// Now send the envelope
$sendEnvelopeparams = new SendEnvelope();
$sendEnvelopeparams->Envelope = $env;
$createResult = $api->SendEnvelope($sendEnvelopeparams)->SendEnvelopeResult;

// Construct the recipient token authentication assertion and specify
// ID, start time, method, and domain
$assertion = new RequestRecipientTokenAuthenticationAssertion();
$assertion->AssertionID = guid();
$assertion->AuthenticationInstant = nowXsdDate();
$assertion->AuthenticationMethod =
RequestRecipientTokenAuthenticationAssertionAuthenticationMethod::Password;
$assertion->SecurityDomain = "Request Recipient Token Test";

// Construct the URLs based on UserName
$recip = $env->Recipients[0];
$urls = new RequestRecipientTokenClientURLs();
$urlbase = "https://127.0.0.1/";
$urls->OnSigningComplete = $urlbase . "?event=SignComplete&uname=" . $recip->UserName;
$urls->OnViewingComplete = $urlbase . "?event=ViewComplete&uname=" . $recip->UserName;
$urls->OnCancel = $urlbase . "?event=Cancel&uname=" . $recip->UserName;
$urls->OnDecline = $urlbase . "?event=Decline&uname=" . $recip->UserName;
$urls->OnSessionTimeout = $urlbase . "?event=Timeout&uname=" . $recip->UserName;
$urls->OnTTLExpired = $urlbase . "?event=TTLExpired&uname=" . $recip->UserName;
$urls->OnIdCheckFailed = $urlbase . "?event=IDCheck&uname=" . $recip->UserName;
$urls->OnAccessCodeFailed = $urlbase . "?event=AccesssCode&uname=" . $recip->UserName;
$urls->OnException = $urlbase . "?event=Exception&uname=" . $recip->UserName;

// Send
$requestRecipientTokenparams = new RequestRecipientToken();
$requestRecipientTokenparams->EnvelopeID = $createResult->EnvelopeID;
$requestRecipientTokenparams->ClientUserID = $recip->CaptiveInfo->ClientUserID;
$requestRecipientTokenparams->Username = $recip->UserName;
$requestRecipientTokenparams->Email = $recip->Email;
$requestRecipientTokenparams->AuthenticationAssertion = $assertion;
$requestRecipientTokenparams->ClientURLs = $urls;
$response = $api->RequestRecipientToken($requestRecipientTokenparams);
```

RequestRecipientFaxPDF

This is an alternative to using RequestRecipientFaxToken. Rather than returning a URL for a signing session, RequestReceipientFaxPDF returns PDF versions of a fax cover page and documents in an envelope for printing and signing. The recipient must still return the documents to DocuSign by fax.

> **IMPORTANT:** Depending on the disclosure settings for your account, you might need to use RequestRecipientFaxToken to let the signer accept the Electronic Record and Signature Disclosure before using RequestRecipientFaxPDF.
>
> *Example:* If your account requires a signer to accept the disclosure and your Electronic Record and Signature Disclosure frequency account setting is "Always" or it is the first time a signer is signing with your organization, then you must use RequestRecipientFaxToken so the signer can accept the disclosure before using RequestRecipientFaxPDF.

Using this method can prevent cross-domain browser issues that can occur when using RequestRecipientFaxToken by allowing API users to take the PDFBytes from the responses and streaming the PDF to the client from their servers.

The parameters for RequestRecipientFaxPDF are the same as those used when requesting recipient tokens.

> **Note:** When requesting a Recipient Token, the ID in ClientUserID must use upper-case letters. If lower-case letters are used (ClientUserId) an error will occur.

Sample Request XML

```
POST /api/3.0/api.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "https://www.docusign.net/API/3.0/RequestRecipientFaxPDF"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestRecipientFaxPDF xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
      <ClientUserID>string</ClientUserID>
      <Username>string</Username>
      <Email>string</Email>
      <AuthenticationAssertion>
        <AssertionID>string</AssertionID>
        <AuthenticationInstant>dateTime</AuthenticationInstant>
        <AuthenticationMethod>string</AuthenticationMethod>
        <SecurityDomain>string</SecurityDomain>
      </AuthenticationAssertion>
    </RequestRecipientFaxPDF>
  </soap:Body>
</soap:Envelope>
```

The response returns the Envelope ID for the envelope and PDFBytes for the documents in the envelope.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestRecipientFaxPDFResponse xmlns="http://www.docusign.net/API/3.0">
      <RequestRecipientFaxPDFResult>
        <EnvelopeID>string</EnvelopeID>
        <PDFBytes>base64Binary</PDFBytes>
      </RequestRecipientFaxPDFResult>
    </RequestRecipientFaxPDFResponse>
  </soap:Body>
</soap:Envelope>
```

DocuSign Operations

Once the authentication is complete user is directed to DocuSign-hosted pages. Here user will be able to view the Electronic Record and Signature Disclosure, Select signature, and complete signing by clicking on Tabs. When signing is complete notification email is sent and user is directed to partner system.

Post-DocuSign Landing Pages

The client application must create, host, and manage landing pages for each of the events that are detailed in the “InSessionCallBackURL” topic. The DocuSign web application will re-direct Recipients to the respective URLs upon reaching a terminal state in the signing/viewing process.

Addenda

Additional Features and Behaviors

This section describes new behaviors that are specific to Embedded Recipients, as well as product enhancements that are currently only available for Embedded Recipients.

- Embedded Recipients will receive an email on behalf of the sending account after they have completed the signing or viewing process. This is intended as a fraud-detection measure and is inherent to the Embedded process. The text of the email is configurable by the sending account.
- This is the only DocuSign-generated email that Embedded recipients will receive. They will not receive emails inviting them to sign, reporting that the Envelope has been voided, or informing them of Envelope completion because the Embedded client application is responsible for these communication tasks.
- Embedded clients have the ability to pass in default SignatureInfo (Signature Name, Signature Initials, and Signature Font), streamlining the user set-up process.
- Embedded clients have the ability to establish Envelope-specific SignatureInfo that is not necessarily the same as the original Recipient Username or the original signature created by the user. This allows the signature to conform to the Recipient’s printed name on a document, specifically in cases where it is represented differently from one document to the next (i.e.,

initials are included in one Envelope instance and not the next). The SignatureInfo, however, cannot vary in separate documents within a single Envelope.

- The Electronic Record and Signature Disclosure agreement is consolidated into the “About You” page for Embedded Recipients. This action streamlines the signing process in relation to the Remote Recipient convention of a separate disclosure page.
- Embedded clients have the ability to create their own branded Help content.

Suppressed Features/Behaviors

This section describes features and behaviors that are characteristic of Remote Recipients but not applicable to Embedded Recipients

- Carbon-Copy Recipients cannot interact as Embedded Recipients.

  Embedded Recipients are not presented with the option to create a Password in DocuSign. Consequently, Captive Recipients cannot enter the DocuSign web application via the login screen at https://www.docusign.net/member/memberlogin.aspx. Captive Recipients can only enter DocuSign via client applications using Recipient Tokens.
- Embedded Recipients do not see the DocuSign Member Console.
- Embedded Recipients do not receive DocuSign-generated email notification for any events except for the Signing/Viewing confirmation upon completion of the respective task, described in the Additional Features/Benefits section.

Legal Considerations

The DocuSign Remote Recipient process flow has been designed to comply with the federal Electronic Signatures in Global and National Commerce Act (ESIGN) and the model form of the Uniform Electronic Transaction Act (UETA). Most of the elements of the signing process such as authentication, disclosure, and document delivery are performed within the DocuSign environment when executing the standard Remote Signing process. However, DocuSign's Embedded signing process provides the Client the flexibility for performing, within the Client’s environment, certain authentication and signing process steps typically performed within the DocuSign environment. Accordingly, while DocuSign will work with the Client to recommend an overall business process for electronic signatures, DocuSign does not assume responsibility or liability for the steps taken outside of the DocuSign Environment and the effect such steps could have on the enforceability of electronic records signed electronically utilizing an Embedded signing process. Additional provisions dealing with an allocation of risk in the Embedded signing process are contained in the relevant sections of the DocuSign Terms of Use for the Embedded process.

GetAuthenticationToken

This methods can be used to get a onetime use URL with an authentication token to launch the DocuSign member system.

Schema

Name	Schema Type	Description
GoToEnvelopeID	string	Optional envelope ID. If provided the URL returned will send the user directly to viewing the envelope. If left blank the user will be taken to the member console.

GetAuthenticationToken rules and exceptions

- User_Does_Not_Exist_In_System – API user requesting the authentication token URL does not have access to the system.
- Account_Lacks_Permissions – API user requesting the authentication token URL does not have valid API permissions.

Sample Code

GetAuthenticationToken – C#

```
// To take advantage of the optional navigation to an envelope,
// create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Now, call the function with the envelope ID of the envelope
// to which you wish to navigate
String token = _apiClient.GetAuthenticationToken(status.EnvelopeID);

// Display token
Console.WriteLine("Token retrieved is {0}", token);
```

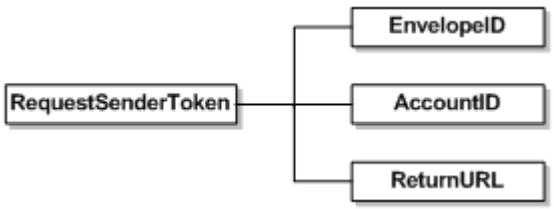
GetAuthenticationToken – PHP

```
// Create and send envelope as shown in linked code
$result = $api->CreateAndSendEnvelope($createAndSendEnvelopeparams)
$status = $result->CreateAndSendEnvelopeResult;

// Send
$getAuthenticationTokenparams = new GetAuthenticationToken();
$getAuthenticationTokenparams->GoToEnvelopeID = $status->EnvelopeID;
$response = $api->GetAuthenticationToken($getAuthenticationTokenparams);
```

RequestSenderToken

This method is used to get a onetime use login token that allows the user to be placed into the DocuSign sending wizard. Upon sending completion the user is returned to the return URL provided by the API application.

Schema



Name	Schema Type	Description
EnvelopeID	string	Optional envelope ID. If provided the user will be placed into the envelope in process. If left blank the user will be placed into a new envelope for sending.
AccountID	String	DocuSign account ID to map the envelope to.
ReturnURL	String	URL to send the user to upon completion of the sending process. The URL will have an event passed to it as a query parameter. The parameter will be named "event", see "In-session sending events" below for description of the valid events. The DocuSign Envelope Id will also be returned in the "envelopeId" parameter. **Important** – You must include HTTPS:// in the URL or the redirect might be blocked by some browsers.

In-session sending events

Events returned on the ReturnURL as a query string parameter. The event is sent in the parameter named "event". The ReturnURL is called with the event on any completion state of the envelope send. Valid events can be found in the schema type InSessionSendEvent in the DocuSign API XSD.

Event	Description
Send	User sends the envelope.
Save	User saves a draft of the envelope.
Cancel	User cancels the sending transaction. No envelope ID returned.
Error	Error performing the send.
SessionEnd	Sending session ended before the user completed.

Rules and exceptions for RequestSenderToken

- User_Does_Not_Exist_In_System – API user requesting the sending token URL does not have access to the system.
- Account_Does_Not_Exist_In_System – Account is not valid.
- Account_Lacks_Permissions – API user requesting the sending token URL does not have valid API permissions.
- User_Lacks_Membership – user is not a valid member of the account.
- User_Not_Envelope_Sender – user is not a valid sender in the account.
- Envelope_Does_Not_Exist – if envelope ID is provided, envelope ID does not exist.
- Account_Not_Authorized_For_Envelope – if envelope ID is provided, account is not valid for the envelope.

Sample Code

RequestSenderToken – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Request the token for the sender
String result = _apiClient.RequestSenderToken(status.EnvelopeID, _accountId, <baseURL>);

// Display token
Console.WriteLine("Sender token is {0}", result);
```

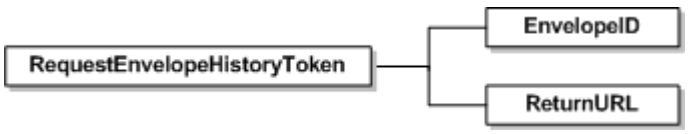
RequestSenderToken – PHP

```
// Create and send an envelope as shown in linked code
$response = <create and send envelope>;
$createResult = $response->CreateAndSendEnvelopeResult;

// Send
$requestSenderTokenparams = new RequestSenderToken();
$requestSenderTokenparams->EnvelopeID = $createResult->EnvelopeID;
$requestSenderTokenparams->AccountID = $AccountID;
$requestSenderTokenparams->ReturnURL = "https://127.0.0.1/";
$response = $api->RequestSenderToken($requestSenderTokenparams);
```

RequestEnvelopeHistoryToken

This method allows the caller to get a view of the history dialog for an envelope. The return URL is the url that the caller wants to return to when the close button is pressed after the history dialog is displayed.

Schema



Name	Schema Type	Description
EnvelopeID	String	Envelope ID for the envelope for which the history view is being requested.
ReturnURL	String	The URL send the user is directed to when the user clicks the close button on the history view. **Important** – You must include HTTPS:// in the URL or the redirect might be blocked by some browsers.

DeleteCaptiveRecipientsSignatures

This method deletes the signature for one or more captive recipient records; it is primarily used for testing. This provides a way to reset the signature associated with a ClientUserId so a new signature can be created the next time the ClientUserId is used.

Schema

Name	Schema Type	Description
ClientUserId	String	The sender created value associated with the captive recipient.
UserName	String	The user name associated with the captive recipient.
Email	String	The email address associated with the captive recipient.

Sample Request XML

```
POST /api/3.0/dsapi.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/3.0/DeleteCaptiveRecipientsSignatures"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  </soap:Body>
    <DeleteCaptiveRecipientsSignatures xmlns="http://www.docusign.net/API/3.0">
      <AccountId>string</AccountId>
      <Arg>
        <CaptiveRecipients>
          <CaptiveRecipient>
            <ClientUserId>string</ClientUserId>
            <UserName>string</UserName>
            <Email>string</Email>
          </CaptiveRecipient>
          <CaptiveRecipient>
            <ClientUserId>string</ClientUserId>
            <UserName>string</UserName>
            <Email>string</Email>
          </CaptiveRecipient>
        </CaptiveRecipients>
      </Arg>
    </DeleteCaptiveRecipientsSignatures>
  </soap:Body>
</soap:Envelope>
```

The response returns the captive recipient information and either a success or failure. If the call fails, error information is provided.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <DeleteCaptiveRecipientsSignaturesResponse xmlns="http://www.docusign.net/API/3.0">
      <DeleteCaptiveRecipientsSignaturesResult>
        <CaptiveRecipients>
          <BrandResultItem>
```

```
            <ClientUserId>string</ClientUserId>
            <UserName>string</UserName>
            <Email>string</Email>
            <ErrorDetails xsi:nil="true" />
          </BrandResultItem>
          <BrandResultItem>
            <ClientUserId>string</ClientUserId>
            <UserName>string</UserName>
            <Email>string</Email>
            <ErrorDetails xsi:nil="true" />
          </BrandResultItem>
        </CaptiveRecipients>
      </DeleteCaptiveRecipientsSignaturesResult>
    </DeleteCaptiveRecipientsSignaturesResponse>
  </soap:Body>
</soap:Envelope>
```

Status and Managing Function Group

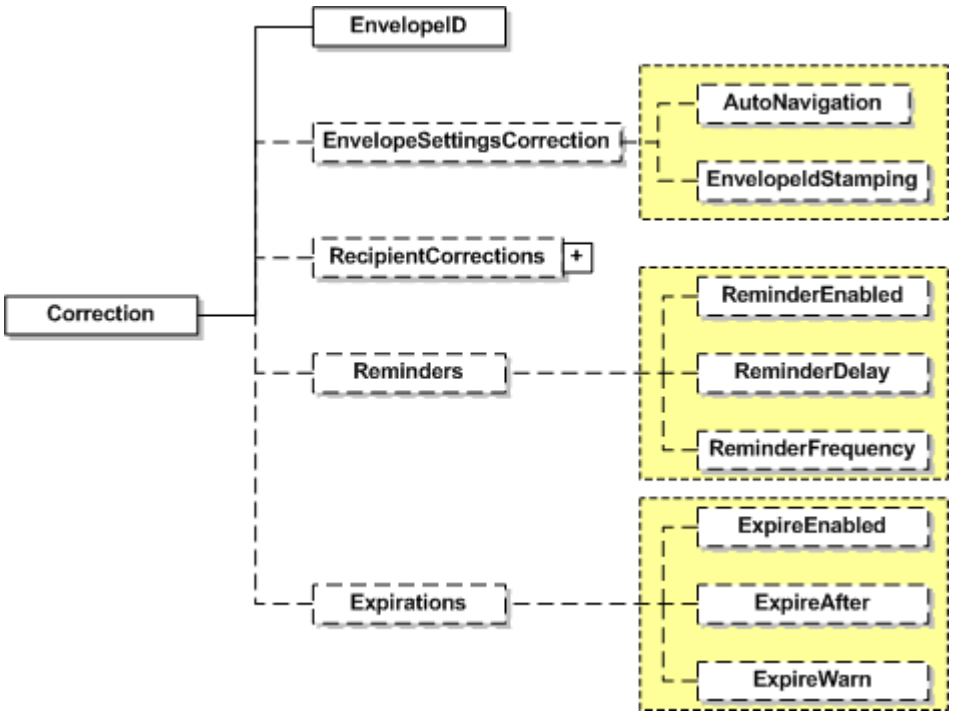
This function group is used to show the status of envelopes and manage envelopes. The Status methods are described first and then the managing methods.

CorrectAndResendEnvelope

The *CorrectAndResendEnvelope* API method enables API users to modify the attributes of envelope reminders, expirations and recipients, and then resend the envelope activation emails.

> **Note:** If automatic reminders are enabled for the envelope, then resending or correcting and resending an envelope resets the time sent for automatic reminder notifications for the envelope. For example, if your automatic reminders are set for 3 days and you correct and resend an envelope, the automatic reminder will be sent 3 days after the correction, not 3 days after the envelope was originally sent.

Schema



Name	Schema Type	Description
Envelope ID	DSXId	Envelope ID of the envelope which is corrected through API call.
EnvelopeSettingCorrection	*EnvelopeSettingCorrection*	Complex element consists of two elements *AutoNavigation* and *EnvelopeIdStamping. AutoNavigation* is a Boolean element and if set to true, auto navigation feature will be enabled. *EnvelopeIdStamping* if set to true, Envelope stamping feature will be enabled.

Name	Schema Type	Description
RecipientCorrections	RecipientCorrection	Contains all the attributes of the recipient that could be corrected. See the RecipientCorrection section below for more information.
Reminders	Reminders	Corrected envelope reminder settings: ReminderEnabled: Boolean: Whether or not a reminder email will be sent. ReminderDelay: Integer: Number of days AFTER a person receives an envelope, that they will start receiving reminder emails. ReminderFrequency: Integer: Number of days to wait between reminder emails.
Expirations	Expirations	Corrected envelope expriation settings: ExpireEnabled: Boolean: Whether or not this envelope is set to expire. ExpireAfter: Integer: Number of days envelope will be active. ExpireWarn: Integer: Number of days before envelope expiration, that a warning email will be sent.

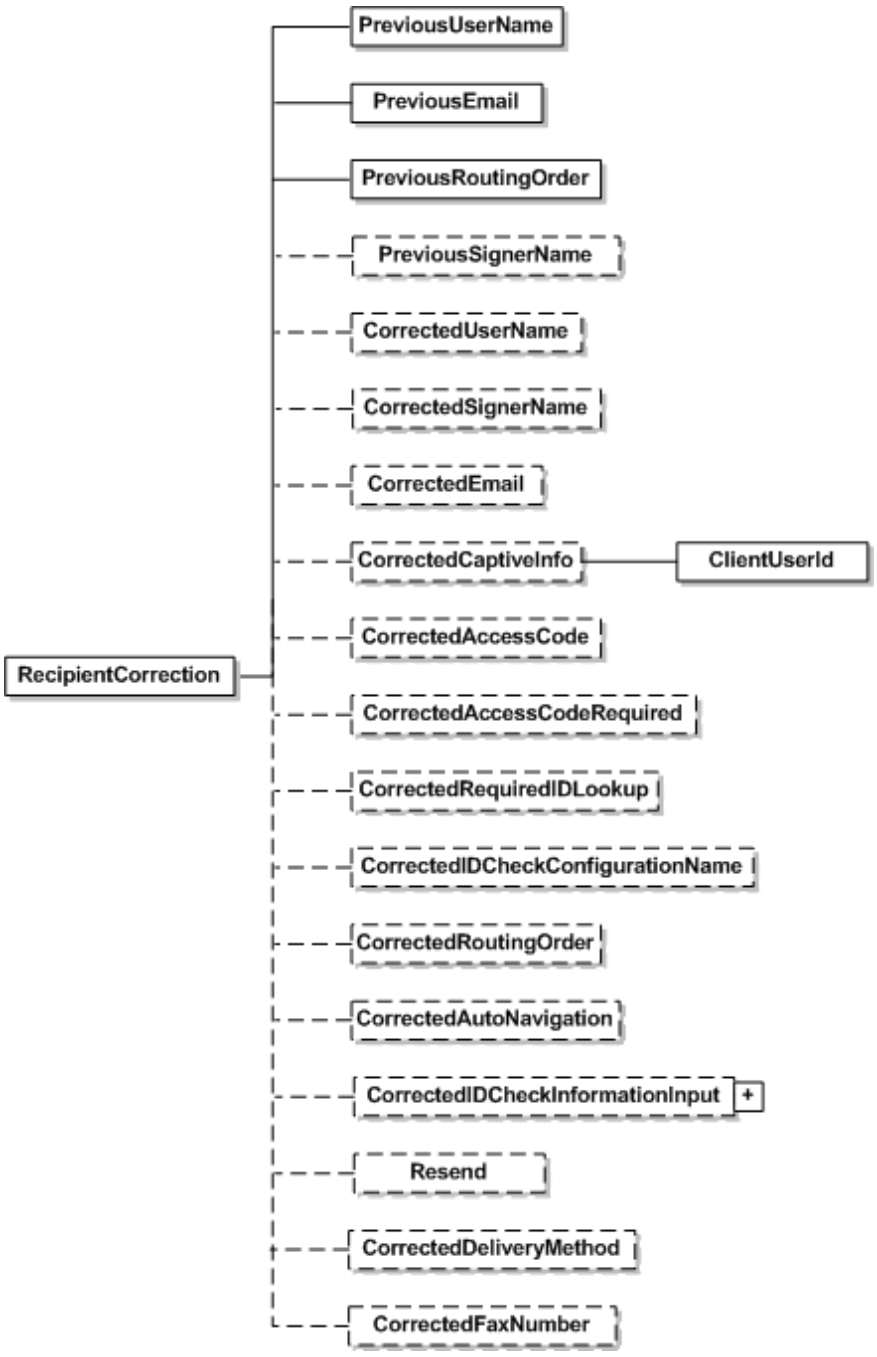
Sample Request XML:

```
SOAPAction: "http://www.docusign.net/API/3.0/CorrectAndResendEnvelope"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CorrectAndResendEnvelope xmlns="http://www.docusign.net/API/3.0">
      <Correction>
        <EnvelopeID>string</EnvelopeID>
        <EnvelopeSettingsCorrection>
          <AutoNavigation>boolean</AutoNavigation>
          <EnvelopeIdStamping>boolean</EnvelopeIdStamping>
        </EnvelopeSettingsCorrection>
        <RecipientCorrections>
          <RecipientCorrection>
            <PreviousUserName>string</PreviousUserName>
            <PreviousEmail>string</PreviousEmail>
            <PreviousRoutingOrder>unsignedShort</PreviousRoutingOrder>
            <PreviousSignerName>string</PreviousSignerName>
            <CorrectedUserName>string</CorrectedUserName>
            <CorrectedSignerName>string</CorrectedSignerName>
            <CorrectedEmail>string</CorrectedEmail>
            <CorrectedCaptiveInfo xsi:nil="true" />
            <CorrectedAccessCode>string</CorrectedAccessCode>
            <CorrectedAccessCodeRequired>boolean</CorrectedAccessCodeRequired>
            <CorrectedRequireIDLookup>boolean</CorrectedRequireIDLookup>
            <CorrectedIDCheckConfigurationName>string</CorrectedIDCheckConfigurationName>
            <CorrectedRoutingOrder>unsignedShort</CorrectedRoutingOrder>
            <CorrectedAutoNavigation>boolean</CorrectedAutoNavigation>
            <CorrectedIDCheckInformationInput xsi:nil="true" />
```

```
            <Resend>boolean</Resend>
            <CorrectedDeliveryMethod>Email or Fax</CorrectedDeliveryMethod>
            <CorrectedFaxNumber>string</CorrectedFaxNumber>
          </RecipientCorrection>
          <RecipientCorrection>
            <PreviousUserName>string</PreviousUserName>
            <PreviousEmail>string</PreviousEmail>
            <PreviousRoutingOrder>unsignedShort</PreviousRoutingOrder>
            <PreviousSignerName>string</PreviousSignerName>
            <CorrectedUserName>string</CorrectedUserName>
            <CorrectedSignerName>string</CorrectedSignerName>
            <CorrectedEmail>string</CorrectedEmail>
            <CorrectedCaptiveInfo xsi:nil="true" />
            <CorrectedAccessCode>string</CorrectedAccessCode>
            <CorrectedAccessCodeRequired>boolean</CorrectedAccessCodeRequired>
            <CorrectedRequireIDLookup>boolean</CorrectedRequireIDLookup>
            <CorrectedIDCheckConfigurationName>string</CorrectedIDCheckConfigurationName>
            <CorrectedRoutingOrder>unsignedShort</CorrectedRoutingOrder>
            <CorrectedAutoNavigation>boolean</CorrectedAutoNavigation>
            <CorrectedIDCheckInformationInput xsi:nil="true" />
            <Resend>boolean</Resend>
            <CorrectedDeliveryMethod>Email or Fax</CorrectedDeliveryMethod>
            <CorrectedFaxNumber>string</CorrectedFaxNumber>
          </RecipientCorrection>
        </RecipientCorrections>
        <Reminders>
          <ReminderEnabled>boolean</ReminderEnabled>
          <ReminderDelay>nonNegativeInteger</ReminderDelay>
          <ReminderFrequency>nonNegativeInteger</ReminderFrequency>
        </Reminders>
        <Expirations>
          <ExpireEnabled>boolean</ExpireEnabled>
          <ExpireAfter>nonNegativeInteger</ExpireAfter>
          <ExpireWarn>nonNegativeInteger</ExpireWarn>
        </Expirations>
      </Correction>
    </CorrectAndResendEnvelope>
  </soap:Body>
</soap:Envelope>
```

RecipientCorrection

Schema



Name	Schema Type	Description
PreviousUserName	UserName	User name of the recipient specified in the envelope which needs correction.
PreviousEmail	Email	Email of the recipient specified in the envelope which needs correction.

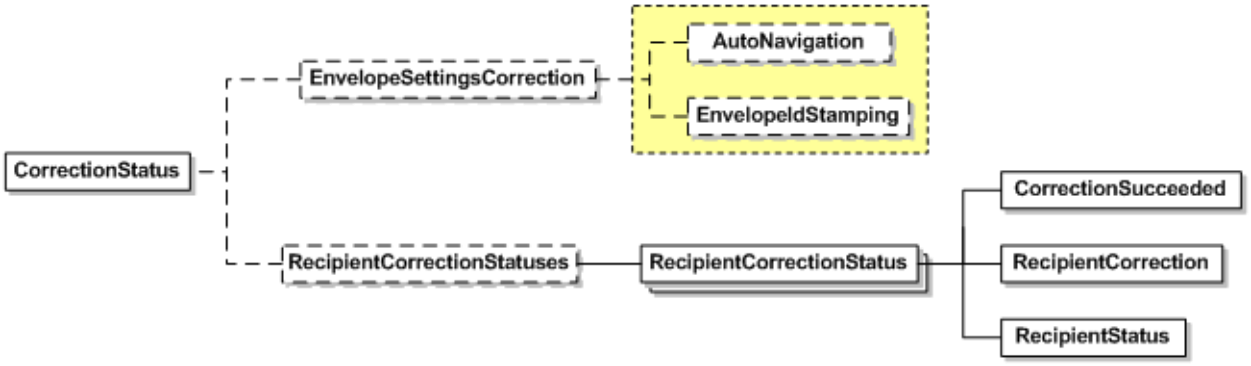
Name	Schema Type	Description
PreviousRoutingOrder	PositiveShort	Routing order of the recipient specified in the envelope which needs correction.
PreviousSignerName	String	Signer name specified in the envelope, which needs correcting.
CorrectedUserName	UserName	Corrected user name of the recipient.
CorrectedSignerName	String	Correct signer name in the envelope.
CorrectedEmail	Email	Corrected Email of the recipient.
CorrectedCaptiveInfo	CorrectedCaptiveInfo	CaptiveInfo is made up of two optional elements that 1) determine if a recipient is embedded or remote, and 2) express a redirect URL used for starting a recipient's signing session. • ClientUserId – This is a sender created value that can be a maximum of 100 characters. If there is a value for this (it is not null) then the recipient is considered to be embedded. Note that if a ClientUserId is used and the account settings SignerMustHaveAccount or SignerMustLoginToSign are true, an error is generated on sending. • EmbeddedRecipientStartURL – This is a sender provided valid URL string for redirecting the recipient. When using this option, the embedded recipient still receives an email from DocuSign, just as a remote recipient would, but when the document link in the email is clicked the recipient is redirected, through DocuSign, to this URL to complete their actions. When routing to the URL, it is up to the sender's system (the server responding to the URL) to then request a recipient token to launch a signing session. If the value SIGN_AT_DOCUSIGN is used for this node, the recipient will be directed to an embedded signing or viewing process directly at DocuSign. The signing or viewing action is initiated by the DocuSign system and the transaction activity and Certificate of Completion records will reflect this. In all other ways the process is identical to an embedded signing or viewing operation that would be launched by any partner. It is important to remember that in a typical embedded workflow the authentication of an embedded recipient is the responsibility of the sending application and DocuSign expects that senders will follow their own

Name	Schema Type	Description
		process for establishing the recipient's identity. In this workflow the recipient goes through the sending application before the embedded signing or viewing process in initiated. However, when the sending application sets EmbeddedRecipientStartURL=SIGN_AT_DOCUSIGN, the recipient goes directly to the embedded signing or viewing process bypassing the sending application and any authentication steps the sending application would use. In this case, DocuSign recommends that one of the normal DocuSign authentication features (Access Code, Phone Authentication, SMS Authentication, etc.) be used to verify the identity of the recipient. If a ClientUserId is NOT provided and an EmbeddedRecipientStartURL is provided, DocuSign will ignore the redirect URL and launch the standard signing process for the email recipient. Information can be appended to the EmbeddedRecipientStartURL using merge fields. The available merge fields items are: EnvelopeId, RecipientId, RecipientName, RecipientEmail, and CustomFields. The CustomFields must be part of the Recipient or Envelope. The merge fields are enclosed in double brackets. Example: http://senderHost/[[mergeField1]]/beginSigningSession?[[mergeField2]]&[[mergeField3]]
CorrectedAccessCode	String	Corrected access for the recipients
CorrectedAccessCodeRequired	Boolean	If set to true, then access code check will be required for the recipient.
CorrectedRequireIDLookup	Boolean	If set to true, then ID check will be required for the recipient.
CorrectedIDCheckConfigurationName	String	Corrected ID check configuration by name.
CorrectedRoutingOrder	PositiveShort	Corrected routing order of the recipient.
CorrectedAutoNavigation	Boolean	Corrected setting for the auto navigation feature. If Boolean is set to true auto navigation feature will be enabled for the recipient
CorrectedIDCheckInformationInput	CorrectedIDCheckInformationInput	Specified the corrected IDCheck information. For details please refer IDCheckInformationInput under CreateAndSend method
Resend	Boolean	Specifies if the email notifying correction needs to be sent to the recipient.

Name	Schema Type	Description
CorrectedDeliveryMethod	Delivery Method	This sets the corrected delivery method used for the recipient. The two enumerations are Email or Fax.
CorrectedFaxNumber	String	The corrected fax number for the recipient.

CorrectionStatus

Schema



Name	Schema Type	Description
EnvelopeSettingsCorrection	EnvelopeSettingsCorrection	Consists of AutoNavigation and EnvelopeIdStamping. A true value indicates that the feature is enabled.
RecipientCorrectionStatuses	RecipientCorrectionStatuses	See the RecipientCorrectionStatus section below for more information.

RecipientCorrectionStatus

Name	Schema Type	Description
CorrectionSuccededed	Boolean	Return true if the envelope correction is successful
RecipientCorrection	RecipientCorrection	Returns the corrected information. For more details check the RecipientCorrection section above
RecipientStatus	RecipientStatus	Returns current status of the recipient. For more details check RecipientStatus section in above.

Rules for CorrectAndResendEnvelope

API user specific rules

- The length of any of the email addresses (specified as *CorrectedEmail*) must not exceed 100 characters. Else, the XML validation fails and the processor throws a Validation error.
- The email addresses (specified as *CorrectedEmail*) must conform to the appropriate email format. Else, the XML validation fails and the processor throws a Validation error.
- The length of any of the EnvelopeID must not exceed 100 characters. Else, the XML validation fails and the processor throws a Validation error.
- If the *CorrectedAccessCode* field is not null and not empty then regardless of whether the value of *CorrectedAccessCode*Required is 'true' or 'false' the *CorrectedAccessCode* is considered as the new Access Code for the specified recipient.
- If the *CorrectedAccessCode* field is null or empty:
    - If the value of CorrectedAccessCodeRequired is 'false', the existing Access Code for the specified recipient is cancelled.
    - If the CorrectedAccessCodeRequired is 'true' then the existing access code is left as it is.

Rules for Exceptions thrown by the API

- An envelope having the specified EnvelopeID must exist in the system. Else, an exception with the error message "Envelope_Does_Not_Exist" is thrown.
- The status of the envelope with the specified EnvelopeID must be 'Sent' or 'Delivered'. Else, an exception with the error message "Envelope_Cannot_Correct_Invalid_State" is thrown.
- The user making the API calls must have permissions to correct the Envelope. Else, an exception with the error message "User_Lacks_Permissions" is thrown. The API user can correct the Envelope only if the user is the sender of the envelope
- User specified by PreviousUserName and PreviousEmail should be the owner of the envelope. Else an exception with error message "User_Not_Envelope_Sender" is thrown.
- The envelope with the specified EnvelopeID must not have duplicate recipients. Else an exception is thrown with the error message "Envelope_Has_Duplicate_Recipients". An envelope with duplicate recipients cannot be corrected, because if a correction is specified for a user who exists as more than one recipient, then it raises an ambiguity on which instance of the user to be corrected.
- More than one correction must not be specified for a given recipient as it raises an ambiguity on which correction to apply. Else, an exception with the error message "Correction_Has_Duplicate_Recipients" thrown.
- The *CorrectedUserName* of any correction (after trimming of leading and following spaces) must not be empty. Else an exception with the error message "Correction_Has_A_Blank_Username" is thrown.
- The final set of recipients of the envelope (assuming that the corrections that succeed are applied) must not contain duplicates. Else an exception is thrown with the error message "Correction_Results_In_Duplicate_Recipients".
- Recipient specified by CorrectedUserName and CorrectedEmail should exist else exception with error message "Recipient_Not_Found_For_Correction" is thrown.

- Correcting Carbon copy recipient is not allowed. If tried the exception with error message “Captive_Carbon_Copy_Recipient_Not_Supported” will be thrown.
- If the corrected Email specified has already been reserved by some other account then an exception with error message “Corrected_Email_Is_Reserved” is thrown.

Rules for return value of the API

- The status of a recipient (for whom a correction is specified) must not be ‘Signed’, ‘Declined’ or ‘Completed’. Else, correction is not permitted for this recipient and *CorrectionSucceeded* value is set to ‘false’. In other words a *CorrectionSucceeded* value of ‘false’ indicates that the correction could not be applied, as the status of the recipient does not permit corrections.
- The *RecipientCorrection* object in the *CorrectAndResendEnvelope* the same as the *RecipientCorrection* passed as the argument to the API.

Sample Code

CorrectAndResendEnvelope – C#

```
// Create envelope as shown in linked code
DocuSignWeb.Envelope envelope = <create envelope>;

// Send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <send envelope>;

// Create a new correction, and make it a recipient correction
DocuSignWeb.Correction correction = new CodeSnippets.DocuSignWeb.Correction();
correction.EnvelopeID = status.EnvelopeID;
correction.RecipientCorrections = new DocuSignWeb.RecipientCorrection[1];

// We're just copying recipient data into the corrected data
// We could also change it entirely
for (int i = 0; i < correction.RecipientCorrections.Length; i++)
{
    DocuSignWeb.RecipientCorrection recipient = correction.RecipientCorrections[i];
    correction.RecipientCorrections[i] = new DocuSignWeb.RecipientCorrection();
    correction.RecipientCorrections[i].PreviousEmail
        = correction.RecipientCorrections[i].CorrectedEmail
        = envelope.Recipients[i].Email;
    correction.RecipientCorrections[i].PreviousUserName
        = correction.RecipientCorrections[i].CorrectedUserName
        = envelope.Recipients[i].UserName;
    correction.RecipientCorrections[i].PreviousRoutingOrder
        = correction.RecipientCorrections[i].CorrectedRoutingOrder
        = envelope.Recipients[i].RoutingOrder;}

// Go ahead and make the correction
DocuSignWeb.CorrectionStatus correctionStatus =
    _apiClient.CorrectAndResendEnvelope(correction);

// Confirm that the call succeeded
Console.WriteLine("Correction status succeeded? {0}",
correctionStatus.RecipientCorrectionStatuses[0].CorrectionSucceeded);
```

CorrectAndResendEnvelope – PHP

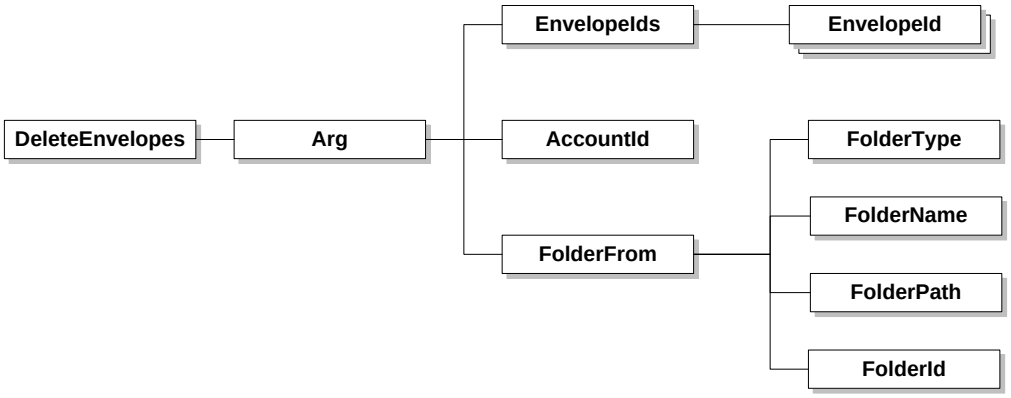
```
// Create envelope as shown in linked code
$env = <create envelope>;

// Send envelope as shown in linked code
```

```
$response = <send envelope>;
$createResponse =$response->CreateAndSendEnvelopeResult;

// Create a new recipient correction
$recip = $env->Recipients[0];
$correction = new Correction();
$correction->EnvelopeID = $createResponse->EnvelopeID;
$recipCorrection = new RecipientCorrection();
//Just reuse the same data
$recipCorrection->PreviousEmail = $recipCorrection->CorrectedEmail = $recip->Email;
$recipCorrection->PreviousUserName = $recipCorrection->CorrectedUserName = $recip-
>UserName;
$recipCorrection->PreviousRoutingOrder = $recipCorrection->CorrectedRoutingOrder =
$recip->RoutingOrder;
$correction->RecipientCorrections = array($recipCorrection);

$correctAndResendEnvelopeparams = new CorrectAndResendEnvelope();
$correctAndResendEnvelopeparams->Correction = $correction;

// Send
$response = $api->CorrectAndResendEnvelope($correctAndResendEnvelopeparams);
```

DeleteEnvelopes

DeleteEnvelopes deletes the specified envelopes from a folder.

Schema:



Name	Schema Type	Description
EnvelopeIds	Array of EnvelopeIds	An array of EnvelopeIds for the envelopes being deleted.
AccountId	String	The account ID associated with the envelopes being deleted.
FolderFrom	FolderTypeInfo	The folder with the envelopes to be deleted. Refer to the FolderTypeInfo section for more information.

This method returns DeleteEnvelopeResponse with the result of the request.

Sample Request XML:

```
SOAPAction: "http://www.docusign.net/API/3.0/DeleteEnvelopes"

<?xml version="1.0" encoding="utf-8"?>
```

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <DeleteEnvelopes xmlns="http://www.docusign.net/API/3.0">
      <Arg>
        <EnvelopeIds>
          <EnvelopeId>string</EnvelopeId>
          <EnvelopeId>string</EnvelopeId>
        </EnvelopeIds>
        <AccountId>string</AccountId>
        <FolderFrom>
          <FolderType>RecycleBin or Draft or Inbox or SentItems or Normal</FolderType>
          <FolderName>string</FolderName>
          <FolderPath>string</FolderPath>
          <FolderId>string</FolderId>
        </FolderFrom>
      </Arg>
    </DeleteEnvelopes>
  </soap:Body>
</soap:Envelope>
```

EnvelopeAuditEvents

The *EnvelopeAuditEvent* API method enables API users to request the events performed on an envelope. The events are returned in an XML document.

Schema

Name	Schema Type	Description
EnvelopeId	DSXId	Envelope ID of the envelope to return the events for.

Sample Request XML:

```
SOAPAction: "http://www.docusign.net/API/3.0/EnvelopeAuditEvents"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <EnvelopeAuditEvents xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeId>string</EnvelopeId>
    </EnvelopeAuditEvents>
  </soap:Body>
</soap:Envelope>
```

Rules for accessing envelope events.

API user specific rules

- The envelope ID passed to the API must be valid.
- The envelope ID passed to the API must be accessible by the API user.

- The user and account must be configured properly by DocuSign in order to enable this API.

Rules for Exceptions thrown by the EnvelopeAuditEvents API

- Envelope_Does_Not_Exist – invalid envelope ID passed.
- User_Lacks_Permissions - The user making the API calls must have permissions to export Authoritative Copy Documents.

Return XML for EnvelopeAuditEvents API

Below is a sample of the XML returned:

```
<DocuSignEnvelopeEvents EnvelopeId="xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" xmlns="">
       <Event>
       <logTime>2008-02-12T16:52:00.6430736-08:00</logTime>
       <Source>Web</Source>
              <UserName>User name here</UserName>
       <Action>Registered</Action>
       <Message>The envelope was created by SS UI and API</Message>
       <EnvelopeStatus>Created</EnvelopeStatus>
       <ClientIPAddress>127.0.0.1</ClientIPAddress>
       <Information />
       </Event>
       <Event>
       <logTime>2008-02-12T16:52:52.0519556-08:00</logTime>
       <Source>Web</Source>
       <UserName>The User</UserName>
       <Action>Sent Invitations</Action>
       <Message>The User sent an invitation to User
email[the.user@docusign.com]</</Message>
       <EnvelopeStatus>Sent</EnvelopeStatus>
       <ClientIPAddress>127.0.0.1</ClientIPAddress>
       <Information>User email[the.user@docusign.com]</Information>
       </Event>
</DocuSignEnvelopeEvents>
```

Sample Code

EnvelopeAuditEvents – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Request the audit history
XmlNode result = _apiClient.EnvelopeAuditEvents(status.EnvelopeID);

// Display audit events
StringWriter stringWriter = new StringWriter();
XmlTextWriter xmlTextWriter = new XmlTextWriter(stringWriter);
xmlTextWriter.Formatting = Formatting.Indented;
result.WriteTo(xmlTextWriter);
xmlTextWriter.Flush();
Console.Write(stringWriter.ToString());
```

EnvelopeAuditEvents – PHP

```
// Create and send an envelope as shown in linked code
$response = <create and send envelope>;
$createResult = $response->CreateAndSendEnvelopeResult;
```

```
// Send and request audit history
$envelopeAuditEventsparams = new EnvelopeAuditEvents();
$envelopeAuditEventsparams->EnvelopeId = $createResult->EnvelopeID;
$response = $api->EnvelopeAuditEvents($envelopeAuditEventsparams);
```

GetConnectFailures

GetConnectFailures retrieves a list of Connect post failures for the account and date range specified.

Schema

Name	Schema Type	Description
AccountId	String	The account ID for the requested Connect post failures.
DateFrom	dateTime	The start date and time for the Connect post failures search.
DateTo	dateTime	The end date and time for the Connect post failure search.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/GetConnectFailures"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetConnectFailures xmlns="http://www.docusign.net/API/3.0">
      <ConnectFailuresFilter>
        <AccountId>string</AccountId>
        <DateFrom>dateTime</DateFrom>
        <DateTo>dateTime</DateTo>
      </ConnectFailuresFilter>
    </GetConnectFailures>
  </soap:Body>
</soap:Envelope>
```

This method returns GetConnectFailureResults with an array of ConnectFailures. Each ConnectFailure instance has the elements shown below.

Name	Schema Type	Description
EnvelopeId	String	The envelope ID of the envelope status that failed to post.
AccountId	String	The account ID associated with the envelope.
Failed	dateTime	The date and time the original Connect post failed.

Name	Schema Type	Description
Retried	dateTime	The date and time the last attempt to post.
RetryCount	Integer	The number of times the Connect post has been retried.
EnvelopeStatus		The new envelope status for the failed Connect post. The possible values are: Any, Voided, Created, Deleted, Sent, Delivered, Signed, Completed, Declined, TimedOut, Template, or Processing
Sender	String	The name of the envelope sender.
Email	String	The email that sent the envelope.
Subject	String	The envelope subject.
Error	String	The error that caused the Connect post to fail.
ConfigId	String	The identifier for the Connect configuration that failed. If an account has multiple Connect configurations, this value is used to to look up the Connect configuration for the failed post.
ConfigUrl	String	The web address of the listener or Retrieving Service end point for Connect.

GetStatusInDocuSignConnectFormat

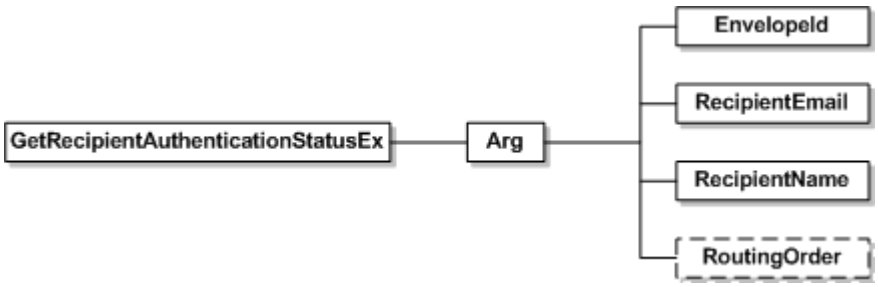
GetStatusInDocuSignConnectFormat API is reserved for future use.

Name	Schema Type	Description
EnvelopeId	DSXId	Envelope ID of the envelope to return the status for.

GetRecipientAuthenticationStatusEx

This method requests the details of authentication checks for a recipient.

Schema



Name	Schema Type	Description
EnvelopeId	DSXId	The envelope ID associated with the recipient authentication check.
RecipientEmail	String	The recipient's email address.
RecipientName	String	The recipient's name

Name	Schema Type	Description
RoutingOrder	Integer	Optional element. The recipient's RoutingOrder in the envelope.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/GetRecipientAuthenticationStatusEx"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetRecipientAuthenticationStatusEx xmlns="http://www.docusign.net/API/3.0">
      <Arg>
        <EnvelopeId>string</EnvelopeId>
        <RecipientName>string</RecipientName>
        <RecipientEmail>string</RecipientEmail>
        <RoutingOrder>int</RoutingOrder>
      </Arg>
    </GetRecipientAuthenticationStatusEx>
  </soap:Body>
</soap:Envelope>
```

This method returns AuthenticationStatus with elements listed below. The response only returns the applicable authentication result. The result contains the authentication status, a date time stamp and authentication failure details (if provided).

Name	Schema Type	Description
AccessCodeResult	EventResult	Result for an access code authentication check, it returns: Status – Pass or Fail dateTime – The date/time of the event FailureDescription – A string with the details of a failed authentication, if provided. This is only provided in case of a failed authentication. VendorFailureStatusCode – A string with a vendor status code, if provided. This is only provided in case of a failed authentication.
IDQuestionsResult	EventResult	Result for an ID question authentication check. It returns the same elements as shown in the AccessCodeResult above.
IDLookupResult	EventResult	Result for an ID lookup authentication check. It returns the same elements as shown in the AccessCodeResult above.
AgeVerifyResult	EventResult	Result for an age verification authentication check. It returns the same elements as shown in the AccessCodeResult above.

Name	Schema Type	Description
STANPinResult	EventResult	Result for a Student Authentication Network (STAN) check. It returns the same elements as shown in the AccessCodeResult above.
OFACResult	EventResult	Result for an Office of Foreign Asset Control (OFAC) check. It returns the same elements as shown in the AccessCodeResult above.
PhoneAuthResult	EventResult	Result for a phone authentication check. It returns the same elements as shown in the AccessCodeResult above.
LiveIDResult	EventResult	Result for a Live ID authentication check. It returns the same elements as shown in the AccessCodeResult above.
FacebookResult	EventResult	Result for a Facebook authentication check. It returns the same elements as shown in the AccessCodeResult above.
GoogleResult	EventResult	Result for a Google authentication check. It returns the same elements as shown in the AccessCodeResult above.
LinkedinResult	EventResult	Result for a LinkedIn authentication check. It returns the same elements as shown in the AccessCodeResult above.
SalesforceResult	EventResult	Result for a Salesforce authentication check. It returns the same elements as shown in the AccessCodeResult above.
TwitterResult	EventResult	Result for a Twitter authentication check. It returns the same elements as shown in the AccessCodeResult above.
OpenIDResult	EventResult	Result for an Open ID authentication check. It returns the same elements as shown in the AccessCodeResult above.
AnySocialIDResult	EventResult	Result for any other social ID authentication check. It returns the same elements as shown in the AccessCodeResult above.
YahooResult	EventResult	Result for a Yahoo authentication check. It returns the same elements as shown in the AccessCodeResult above.

GetSharedAccess

GetSharedAccess requests shared item status for one or more users and types of items.

Schema

Name	Schema Type	Description
AccountId	String	Account Id.

Name	Schema Type	Description
SharedAccessFilters	Array of SharedAccessFilter	An array of SharedAccessFilter that defines the requested user and shared item information.

SharedAccessFilter Schema

Name	Schema Type	Description
UserIds	Array of UserId	An array of User Ids for whom the shared item information is being requested.
ItemType	ItemType	Specifies the type of shared item being requested. The accepted values are: • Envelopes: returns information about envelope sharing between users.
SearchText	String	Optional field. This can be used to filter user names in the response. The wild-card '%' can be used around the string. If no value is specified, then there is no filtering of user names.
Shared	Shared	Optional field. Specifies which users should be included in the response by setting the sharing status for the requested user (by UserId) and item (by ItemType). If no value is specified and the requester has account administrator privileges, then the request returns item sharing status for all active account users. If no value is specified and the requestor does not have account administrator privileges, the SharedTo value is used. Requestors that do not have account administrator privileges can only use the SharedTo, any other setting will result in an error. The accepted values are: • NotShared: Returns account users that the specified item type is not being shared with and that are not sharing the specified item type with the user. User ✕ (Share) ✕ Account user • SharedTo: Returns account users that the specified item type is not being shared with and who are sharing the specified item type with the user (only shared to the user). User ◂◂ (Share) Account user • SharedFrom: Returns account users that the specified item type is being shared with and who are not sharing the specified item type with the user (only shared from the

Name	Schema Type	Description
		user). User (Share) ⏩ Account user • SharedToAndFrom: Returns account users that the specified item type is being shared with and who are sharing the specified item type with the user. User ⏪ (Share) ⏩ Account user
StartAtIndex	nonNegativeInteger	Optional Field. If a shared access response set exceeds Count, this element can be used to specify that the method should return users starting at the specified index. The first index is 0, and should be used in the first GetSharedAccess call. Typically this number is a multiple of Count. If no value is specified, this defaults to be 0.
Count	nonNegativeInteger	Optional Field. Specifies how many sharedItems results are included in the response. If no value is specified, this defaults to 1000.

Sample Request XML

```
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
    <GetSharedAccess xmlns="http://www.docusign.net/API/3.0">
      <AccountId>b9ddeca5-2eb0-448b-b304-11e8a595e740</AccountId>
      <SharedAccessFilter>
        <UserIds>
          <UserId>4af36b6c-8475-491f-8be9-4212e419a78c</UserId>
          <UserId>dcde4bbd-d8f1-442b-a1be-171bdcb6974d</UserId>
          <UserId>d1375465-3980-4be3-8b59-4c0dad592dfa</UserId>
        </UserIds>
        <ItemType>Envelopes</ItemType>
        <SearchText>%John%</SearchText>
        <Shared>SharedTo</Shared>
        <StartAtIndex>0</StartAtIndex>
        <Count>4</Count>
      </SharedAccessFilter>
    </GetSharedAccess>
  </env:Body>
</env:Envelope>
```

The response provides information about the result set, an array of the shared items, and either a success or failure. If the call fails an error code is provided.

A description of the information in the response is given below.

Name	Schema Type	Description
AccountId	String	Account Id.

Name	Schema Type	Description
ResultSetSize	nonNegativeInteger	The number of results returned in this response.
TotalSetSize	nonNegativeInteger	The total number of results from the call. This will always be greater than or equal to ResultSetSize.
StartAtIndex	nonNegativeInteger	The index of the first element in the response set.
EndAtIndex	nonNegativeInteger	The index of the last element in the response set.
SharedItems	Array of SharedItems objects	The shared access information for the users specified in the request. See description below for more information.

Description of *SharedItems* objects

Name	Schema Type	Description
User	User object	The information for the user associated with the shared item information in this object.
SharedItem	Array of SharedItem objects	Shared item information for the user. See description below for more information.
ErrorDetails	ErrorDetails object	Error information related to this user.

Description of *SharedItem* objects

Name	Schema Type	Description
User	User object	The information for the account user associated with this shared status. See description below for more information.
Shared	Shared status	Specifies the current sharing status between the requested user and other account users for the specified item type. • NotShared: The specified item type is not being shared with this account user and the account user not sharing the specified item type with the user. User ✕ (Share) ✕ Account user • SharedTo: The specified item type is not being shared with the account user and the account user is sharing the specified item type with the user (only shared to the user). User ◂◂ (Share) Account user • SharedFrom: The specified item type is being shared with the account user and the account user is not sharing the

Name	Schema Type	Description
		specified item type with the user (only shared from the user). User (Share) ⏩ Account user • SharedToAndFrom: The specified item type is being shared with the account user and account user is sharing the specified item type with the user. User ⏪ (Share) ⏩ Account user
ItemType	ItemType	The type of the shared item. Currently the only ItemType supported is Envelopes.

Description of *User* objects

Name	Schema Type	Description
UserId	UserId	User Id of the account user.
Email	String	Email for the account user.
UserName	String	User name for the account user.

Sample Response XML

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetSharedAccessResponse xmlns=" http://www.docusign.net/API/3.0">
      <GetSharedAccessResult>
        <AccountId>b9ddeca5-2eb0-448b-b304-11e8a595e740</AccountId>
        <ResultSetSize>4</ResultSetSize>
        <TotalSetSize>50</TotalSetSize>
        <StartAtIndex>0</StartAtIndex>
        <EndAtIndex>3</EndAtIndex>
        <SharedItems>
          <User>
            <UserId>4af36b6c-8475-491f-8be9-4212e419a78c</UserId>
            <Email>some.name@docusign.com</Email>
            <UserName>Some Name</UserName>
          </User>
          <SharedItem>
            <User>
              <UserId>1272b5d0-f7d1-4295-bcb0-ce8ccafdffd4</UserId>
              <Email>elton.john@docusign.co</Email>
              <UserName>Elton John</UserName>
            </User>
            <Shared>NotShared</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>1f2289df-01d7-4bd7-b581-bb70362ea662</UserId>
              <Email>john.doe@docusign.com</Email>
              <UserName>John Doe</UserName>
            </User>
```

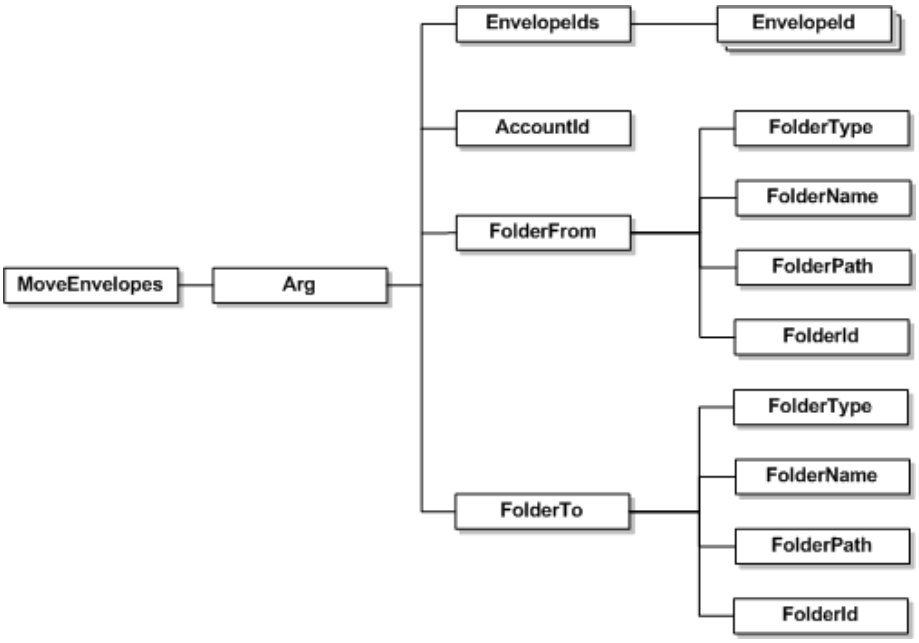
```
            <Shared>NotShared</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
        </SharedItems>
        <SharedItems>
          <User>
            <UserId>dcde4bbd-d8f1-442b-a1be-171bdcb6974d</Userid>
            <Email>another.name@docusign.com</Email>
            <UserName>Another Name</UserName>
          </User>
          <SharedItem>
            <User>
              <UserId>1272b5d0-f7d1-4295-bcb0-ce8ccafdffd4</UserId>
              <Email>calamity.jane@docusign.com</Email>
              <UserName>Calamity Jane</UserName>
            </User>
            <Shared>SharedTo</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>1f2289df-01d7-4bd7-b581-bb70362ea662</UserId>
              <Email>jane.doe@docusign.com</Email>
              <UserName>Jane Doe</UserName>
            </User>
            <Shared>SharedToAndFrom</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
        </SharedItems>
        <Error>
          <ErrorCode>string</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetSharedAccessResult>
    </GetSharedAccessResponse>
  </soap:Body>
</soap:Envelope>
```

MoveEnvelopes

MoveEnvelopes is used to move envelopes between folders.

> **Note:** This can be used to delete envelopes by using “RecycleBin” as the FolderType. Placing an in process envelope (envelope status of sent or delivered) in the RecycleBin folder will void the envelope.

Schema



Name	Schema Type	Description
EnvelopeIds	Array of EnvelopeIds	An array of EnvelopeIds for the envelopes being moved.
AccountId	String	The account ID associated with the envelopes.
FolderFrom	FolderTypeInfo	The folder where the envelopes are currently located. Refer to the FolderTypeInfo section for more information.
FolderTo	FolderTypeInfo	The folder to which the envelopes are being moved. Refer to the FolderTypeInfo section for more information.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/MoveEnvelopes"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <MoveEnvelopes xmlns="http://www.docusign.net/API/3.0">
      <Arg>
        <EnvelopeIds>
          <EnvelopeId>string</EnvelopeId>
          <EnvelopeId>string</EnvelopeId>
        </EnvelopeIds>
        <AccountId>string</AccountId>
        <FolderFrom>
          <FolderType>RecycleBin or Draft or Inbox or SentItems or Normal</FolderType>
          <FolderName>string</FolderName>
```

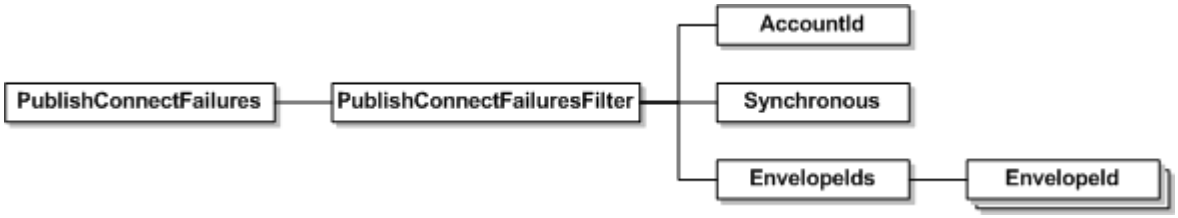
```
          <FolderPath>string</FolderPath>
          <FolderId>string</FolderId>
        </FolderFrom>
        <FolderTo>
          <FolderType>RecycleBin or Draft or Inbox or SentItems or Normal</FolderType>
          <FolderName>string</FolderName>
          <FolderPath>string</FolderPath>
          <FolderId>string</FolderId>
        </FolderTo>
      </Arg>
    </MoveEnvelopes>
  </soap:Body>
</soap:Envelope>
```

This method returns MoveEnvelopeResponse with the result of the request.

PublishConnectFailures

PublishConnectFailures requests a list of Connect post failures for the set of envelopes included in the request.

Schema



Name	Schema Type	Description
AccountId	String	The account ID associated with the requested Connect post failures.
Synchronous	Boolean	If true, the system attempts to publish failed posts again and returns the status of the post attempt.
EnvelopeIds	Array of EnvelopeIds	The array of EnvelopeIds for the envelopes included in the request.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/PublishConnectFailures"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <PublishConnectFailures xmlns="http://www.docusign.net/API/3.0">
      <PublishConnectFailuresFilter>
        <AccountId>string</AccountId>
        <Synchronous>boolean</Synchronous>
        <EnvelopeIds>
          <EnvelopeId>string</EnvelopeId>
          <EnvelopeId>string</EnvelopeId>
        </EnvelopeIds>
      </PublishConnectFailuresFilter>
```

```
    </PublishConnectFailures>
  </soap:Body>
</soap:Envelope>
```

This method returns PublishConnectFailuresResponse with an array of PublishConnectFailuresResult with the results shown below.

Name	Schema Type	Description
EnvelopeId	EnvelopeId	The envelope ID associated with the Connect post.
ConfigId	String	The identifier for the Connect configuration that failed. If an account has multiple Connect configurations, this value is used to to look up the Connect configuration for the failed post.
ConfigUrl	String	The web address of the listener or Retrieving Service end point for Connect.
Status	ConnectPublishStatus	The status of the Connect post. The possible values are: Queued, Success, or Fail.
StatusMessage	String	If the ConnectPublishStatus is Queued or Success, the StatusMessage is empty. If the ConnectPublishStatus is Fail and the envelope is not part of the account, the StatusMessage is "Envelope Not In Account."

RequestCorrectToken

This call returns a token to place a user in a web session in Advanced Correct mode on an envelope.

Name	Schema Type	Description
EnvelopeID	String	Identifies the envelope to be corrected.
SuppressNavigation	Boolean	Sets whether the window is displayed with or without dressing.
ReturnURL	String	Identifies the return point after correcting the envelope. The URL will have an event passed to it as a query parameter. The parameter will be named “event”, see “In-session sending events” below for description of the valid events. The DocuSign Envelope Id will also be returned in the “envelopeId” parameter. **Important** – You must include HTTPS:// in the URL or the redirect might be blocked by some browsers.

In-Session Sending Events

Events returned on the ReturnURL as a query string parameter. The event is sent in the parameter named “event”. The ReturnURL is called with the event on any completion state of the envelope send. Valid events can be found in the schema type InSessionSendEvent in the DocuSign API XSD.

Event **Description**

Send	User sends the envelope.
Save	User saves a draft of the envelope.
Cancel	User cancels the sending transaction. No envelope ID returned.
Error	Error performing the send.
SessionEnd	Sending session ended before the user completed.

Sample Code

RequestCorrectToken – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Request the token with the envelope ID and a return URL
String token = _apiClient.RequestCorrectToken(status.EnvelopeID, true, baseURL);

// Display token
Console.WriteLine("The token returned is {0}", token);
```

RequestCorrectToken – PHP

```
// Create and send an envelope s shown in linked code
$response = <create and send envelope>;
$createResult = $response->CreateAndSendEnvelopeResult;

// Request the token with the envelope ID and a return URL
$requestCorrectTokenparams = new RequestCorrectToken();
$requestCorrectTokenparams->EnvelopeID = $createResult->EnvelopeID;
$requestCorrectTokenparams->ReturnURL = $baseURL;
$requestCorrectTokenparams->SuppressNavigation = true;
$response = $api->RequestCorrectToken($requestCorrectTokenparams);
```

RequestStatus and RequestStatusEx

The RequestStatus and RequestStatusEx methods can be used to query the status of existing envelopes.

The RequestStatus method takes the element EnvelopeID for the envelope for which the status is requested. This Envelope ID must refer to an envelope created via the *Create Envelope/CreateAndSendEnvelope* method call.

Schema

RequestStatus

RequestStatusEx

Name	Schema Type	Description

Name	Schema Type	Description
EnvelopeID	DSXId	The Envelope ID for which to retrieve status.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/RequestStatus"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestStatus xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
    </RequestStatus>
  </soap:Body>
</soap:Envelope>
```

Sample Code

RequestStatus – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus eStatus = <create and send envelope>;

// Request the status of that envelope
DocuSignWeb.EnvelopeStatus status = _apiClient.RequestStatus(eStatus.EnvelopeID);

// Display the status for the envelope
Console.WriteLine("Status for envelope ID {0} is {1}", eStatus.EnvelopeID,
status.Status);
```

RequestStatus – PHP

```
// Create and send envelope as shown in linked code
$response = <create and send envelope>;
$createResult = $response->CreateAndSendEnvelopeResult;

// Request the status of that envelope
$requestStatusparams = new RequestStatus();
$requestStatusparams->EnvelopeID = $createResult->EnvelopeID;
$result = $api->RequestStatus($requestStatusparams);
```

RequestStatusEx – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus eStatus = <create and send envelope>;

// Request the status of that envelope
DocuSignWeb.EnvelopeStatus status = _apiClient.RequestStatusEx(eStatus.EnvelopeID);

// Confirm that the status of the envelope is sent
Console.WriteLine("Status for envelope ID {0} is {1}", eStatus.EnvelopeID,
status.Status);
Console.WriteLine("Account status is {0}", status.RecipientStatuses[0].AccountStatus);
```

RequestStatusEx – PHP

```
// Create and send envelope as shown in linked code
$response = <create and send envelope>;
```

```
$createResult = $response->CreateAndSendEnvelopeResult;

// Request the status of that envelope
$requestStatusExparams = new RequestStatusEx();
$requestStatusExparams->EnvelopeID = $createResult->EnvelopeID;
$result = $api->RequestExStatus($requestStatusparams);
```

RequestStatusWithDocumentFields

The RequestStatusWithDocumentFields call is similar to the RequestStatus and RequestStatusEx calls. It returns envelope status for the requested envelope along with all the envelope data, including the document custom DocumentFields.

Schema

Name	Schema Type	Description
EnvelopeID	DSXId	The Envelope ID for the envelope being requested.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/AddMembersToAccount"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestStatusWithDocumentFields xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
    </RequestStatusWithDocumentFields>
  </soap:Body>
</soap:Envelope>
```

The response returns the requested Envelope information, including the document custom DocumentFields.

RequestStatuses and RequestStatusesEx

This method is used to request the status of multiple envelopes in a single call. Up to 1,000 envelopes can be retrieved within a single call. If more results are available, subsequent calls can be used to retrieve the next 1,000, etc.

Request Envelope Statuses Notes

The SOAP calls for RequestStatuses(Ex) and RequestStatusCodes use certain filters to find results. In some cases requests are check for “any status change” instead of the just the single status requested. In these cases, more envelopes might be returned by the request than otherwise would be. For example, for a request with the begin date is set to Jan 1st, an end date set to Jan 7th and the StatusQualifier set to “Delivered” – the response set might contain envelopes that were created during that time period, but not delivered during the time period.

To avoid unnecessary database queries, the DocuSign system checks requests to ensure that the added filters will not result in a zero-size response before acting on the request. The following table shows the valid envelope statuses (in the Valid Current Statuses column) for the status qualifiers in the request. If the status and status qualifiers in the API request do not contain any of the values shown in the valid current statuses column, then an empty list is returned.

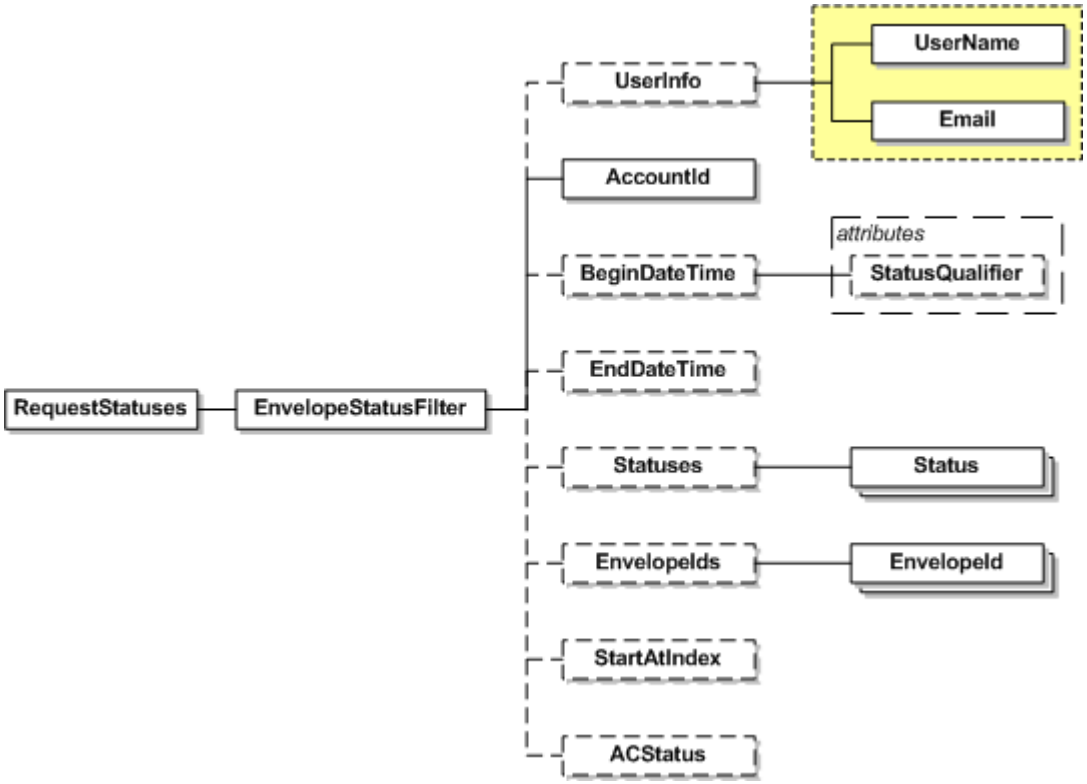
For example, a request with a StatusQualifier of “Delivered” and a status of “Created,Sent”, DocuSign will always return an empty list. This is because the request essentially translates to: find the envelopes that were Delivered between the begin and end dates that have a current status of Created or Sent, and since an envelope that has been delivered can never have a status of Created or Sent, a zero-size response would be generated. In this case, DocuSign does not run the request, but just returns the empty list.

Client applications should check that the statuses they are requesting make sense for a given status qualifier.

BeginDateTime - StatusQualifier	Effective Status Qualifier	Valid Current Statuses
Any (changed)	StatusChanged	Any, Created, Sent, Delivered, Signed, Completed, Declined, Voided, Deleted
Created	Created	Any, Created, Sent, Delivered, Signed, Completed, Declined, Voided, Deleted
Sent	Sent	Any, Sent, Delivered, Signed, Completed, Declined, Voided, Deleted
Delivered	StatusChanged	Any, Delivered, Signed, Completed, Declined, Voided, Deleted
Signed	StatusChanged	Any, Signed, Completed, Declined, Voided, Deleted
Completed	Completed	Any, Completed, Declined, Voided, Deleted
Declined	StatusChanged	Any, Declined, Deleted
TimedOut - Always return zero results	StatusChanged	Any, Voided, Deleted
Voided	Voided	Any, Voided, Deleted
Deleted	StatusChanged	Any, Deleted

Schema

RequestStatuses



RequestStatusesEx (EnvelopeStatusFilter is the same as above)

Name	Schema Type	Description
UserInfo	UserInfo	This would be the username Email pair of the person whose envelope details are requested.
AccountId	DSXId	Account Id for which envelope status is requested.
BeginDateTime	dateTime	Specifies the start date of the date range for which envelopes are requested. The date range also can have a status qualifier attribute, which specifies the envelope statuses being requested. The StatusQualifier options are: Completed, Created (the default and what is used if Template or Processing is passed in), Sent, and Voided; any other value becomes StatusChanged.
EndDateTime	dateTime	Specifies the end date of the date range for which envelopes are requested.

Name	Schema Type	Description
Statuses	EnvelopeStatusCode	Specifies the status of the envelope at the time of request. This is enumerator controlled.
EnvelopeIds	DSXId	Specifies the envelope Ids of envelopes for which status request. There is a maximum of 200 different EnvelopeIds for each request.
StartAtIndex	nonNegativeInteger	If envelope status result sets exceed 1000 envelopes, this element can be used to specify that the method should return envelopes at the specified index. The first index is 0, and should be used in the first call to RequestStatuses. Typically this number is a multiple of 1000.
ACStatus	EnvelopeACStatusCode	Specifies the Authoritative Copy Status for the envelopes. The possible values are: Unknown, Original, Transferred, AuthoritativeCopy, AuthoritativeCopyExportPending, AuthoritativeCopyExported, DepositPending, Deposited, DepositedEO, or DepositFailed.

Sample Code

RequestStatuses – C#

```
// Create a filter using account ID and today as a start time
DocuSignWeb.EnvelopeStatusFilter filter = new DocuSignWeb.EnvelopeStatusFilter();
filter.AccountId = _accountId;
filter.BeginDateTime = new DocuSignWeb.EnvelopeStatusFilterBeginDateTime();
filter.BeginDateTime.Value = DateTime.Today;

// Request all envelopes that match the filter
DocuSignWeb.FilteredEnvelopeStatuses statuses = _apiClient.RequestStatuses(filter);

Console.WriteLine("We have {0} statuses that match account ID {1}",
        statuses.EnvelopeStatuses.Length, _accountId);

foreach (DocuSignWeb.EnvelopeStatus eStatus in statuses.EnvelopeStatuses)
{
    Console.WriteLine("\tEnvelope with ID {0} has status {1}", eStatus.EnvelopeID,
        eStatus.Status.ToString());
}
```

RequestStatuses – PHP

```
// Create a filter using account ID and today as a start time
$envStatusFilter = new EnvelopeStatusFilter();
$envStatusFilter->AccountId = $AccountID;
$beginDateTime = new EnvelopeStatusFilterBeginDateTime();
$beginDateTime->_ = todayXsdDate(); // note that this helper function
                                    // is in CodeSnippets/include/utils.php
                                    // in the PHP SDK
$envStatusFilter->BeginDateTime = $beginDateTime;

// Send
$requestStatusesparams = new RequestStatuses();
```

```
$requestStatusesparams->EnvelopeStatusFilter = $envStatusFilter;
$response = $api->RequestStatuses($requestStatusesparams);
```

RequestStatusesEx – C#

```
// Create a filter using account ID and today as a start time
DocuSignWeb.EnvelopeStatusFilter filter
    = new DocuSignWeb.EnvelopeStatusFilter();

filter.AccountId = _accountId;
DocuSignWeb.EnvelopeStatusFilterBeginDateTime begin
    = new DocuSignWeb.EnvelopeStatusFilterBeginDateTime();

begin.Value = DateTime.Today;
filter.BeginDateTime = begin;

// Request all envelopes that match the filter
DocuSignWeb.FilteredEnvelopeStatuses statuses = _apiClient.RequestStatusesEx(filter);

// Display information about the statuses
Console.WriteLine("We have {0} statuses that match account ID {1}",
        statuses.EnvelopeStatuses.Length, statuses.EnvelopeStatusFilter.AccountId);

foreach (DocuSignWeb.EnvelopeStatus eStatus in statuses.EnvelopeStatuses)
{
    Console.WriteLine("\tEnvelope with ID {0}", eStatus.EnvelopeID);
    foreach (DocuSignWeb.DocumentStatus dStatus in eStatus.DocumentStatuses)
    {
        Console.WriteLine("\t\tDocument with ID {0} has name {1}", dStatus.ID,
            dStatus.Name);
    }
}
```

RequestStatusesEx – PHP

```
// Create a filter using account ID and today as a start time
$envStatusFilter = new EnvelopeStatusFilter();
$envStatusFilter->AccountId = $AccountID;
$beginDateTime = new EnvelopeStatusFilterBeginDateTime();
$beginDateTime->_ = todayXsdDate(); // link to Helper Functions
$envStatusFilter->BeginDateTime = $beginDateTime;

// Send
$RequestStatusesExparams = new RequestStatusesEx();
$RequestStatusesExparams->EnvelopeStatusFilter = $envStatusFilter;
$response = $api->RequestStatusesEx($RequestStatusesExparams);
```

RequestStatusesWithDocumentFields

The RequestStatusesWithDocumentFields call is similar to the RequestStatuses and RequestStatusesEx calls and follows the same rules for requests if the result size is over 200 envelopes. It returns envelope status for the requested envelopes with all the envelope data, including the document custom DocumentFields.

Schema

EnvelopeStatusFilter information:

Name	Schema Type	Description

Name	Schema Type	Description
UserInfo	UserInfo	This would be the username Email pair of the person whose envelope details are requested.
AccountId	DSXId	Account Id for which envelope status is requested.
BeginDateTime	dateTime	Specifies the start date of the date range for which envelopes are requested. The date range also can have a status qualifier attribute, which specifies the envelope statuses being requested. The StatusQualifier options are: Completed, Created (the default and what is used if Template or Processing is passed in), Sent, and Voided; any other value becomes StatusChanged.
EndDateTime	dateTime	Specifies the end date of the date range for which envelopes are requested.
Statuses	EnvelopeStatusCode	Specifies the status of the envelope at the time of request. This is enumerator controlled.
EnvelopeIds	DSXId	Specifies the envelope Ids of envelopes for which status needs to be known.
StartAtIndex	nonNegativeInteger	If envelope status result sets exceed 200 envelopes, this element can be used to specify that the method should return envelopes at the specified index. The first index is 0, and should be used in the first call to RequestStatusesWithDocumentFields. Typically this number is a multiple of 200.
ACStatus	EnvelopeACStatusCode	Specifies the Authoritative Copy Status for the envelopes. The possible values are: Unknown, Original, Transferred, AuthoritativeCopy, AuthoritativeCopyExportPending, AuthoritativeCopyExported, DepositPending, Deposited, DepositedEO, or DepositFailed.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/AddMembersToAccount"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestStatusesWithDocumentFields xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeStatusFilter>
        <UserInfo>
          <UserName>string</UserName>
          <Email>string</Email>
```

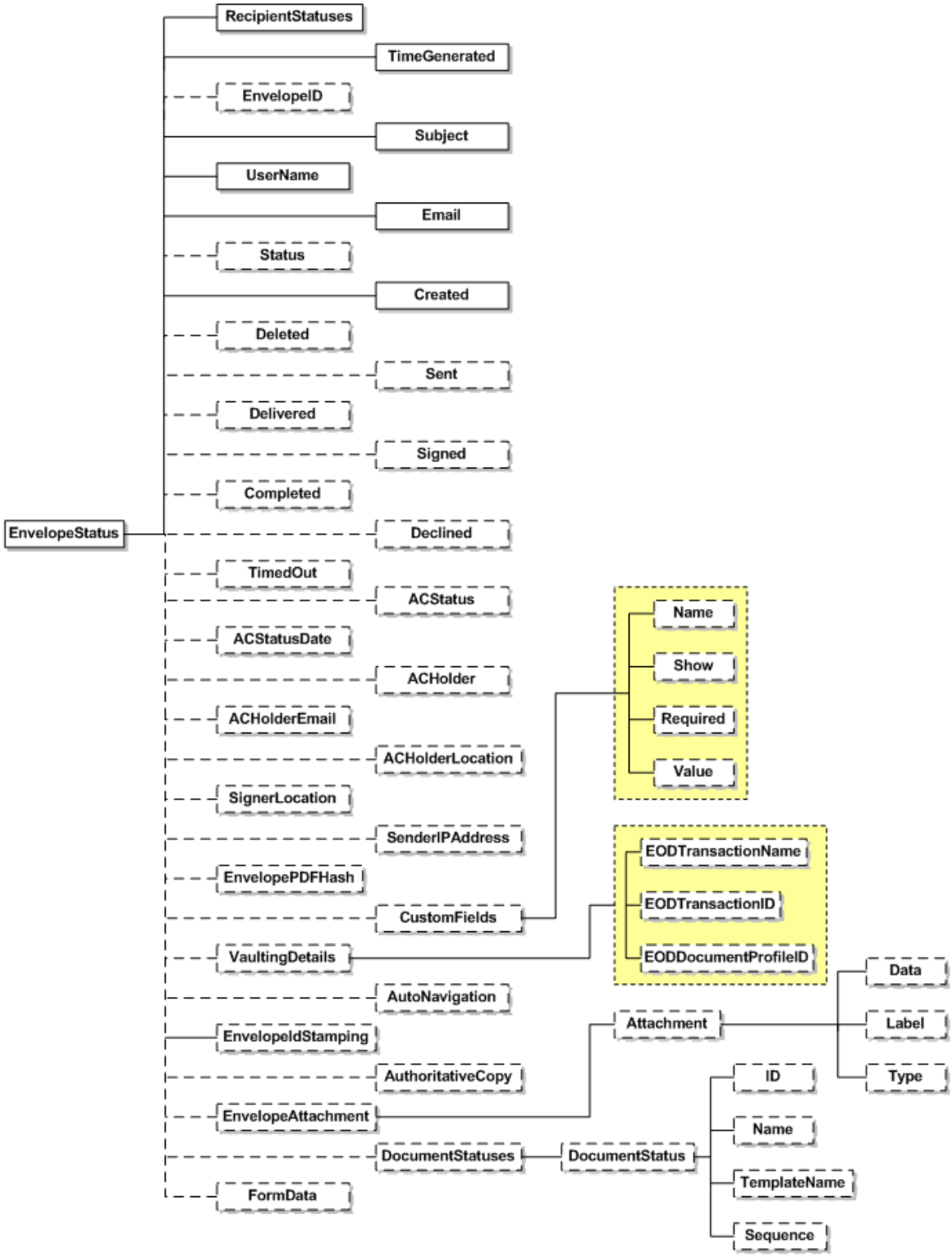
```
        </UserInfo>
        <AccountId>string</AccountId>
        <BeginDateTime d5p1:statusQualifier="string"
xmlns:d5p1="http://www.docusign.net/API/3.0" />
        <EndDateTime>dateTime</EndDateTime>
        <Statuses>
          <Status>Any or Voided or Created or Deleted or Sent or Delivered or Signed or
Completed or Declined or TimedOut or Template or Processing</Status>
          <Status>Any or Voided or Created or Deleted or Sent or Delivered or Signed or
Completed or Declined or TimedOut or Template or Processing</Status>
        </Statuses>
        <EnvelopeIds>
          <EnvelopeId>string</EnvelopeId>
          <EnvelopeId>string</EnvelopeId>
        </EnvelopeIds>
        <StartAtIndex>nonNegativeInteger</StartAtIndex>
        <ACStatus>Unknown or Original or Transferred or AuthoritativeCopy or
AuthoritativeCopyExportPending or AuthoritativeCopyExported or DepositPending or
Deposited or DepositedEO or DepositFailed</ACStatus>
      </EnvelopeStatusFilter>
    </RequestStatusesWithDocumentFields>
  </soap:Body>
</soap:Envelope>
```

The response returns the requested Envelopes, including the document custom DocumentFields.

EnvelopeStatus

Shown below is the schema for envelope status, which is the response schema to the RequestStatus.

Schema



The RequestStatus response contains a snapshot of the status of the envelope at the time the call was issued. It contains the following attributes and elements:

Name	Schema Type	Description
RecipientStatuses	RecipientStatus	Includes the status of each of the envelope recipients. See the RecipientStatus section below for more information.
TimeGenerated	dateTime	Specifies the time of the status change.
EnvelopeID	DSXId	The Envelope ID of the envelope who's status is provided.
Subject	String	The envelope subject, as provided by the envelope creator.
UserName	UserName	Contains the user name of the person who created the envelope
Email	Email	Contains the email of the person who created the envelope
Status	EnvelopeStatusCode	Status of the envelope at that specific time. Enumeration values could be Created, Deleted, Sent, Delivered, Signed, Completed, Declined, Voided, TimedOut, AuthoritativeCopy, TransferCompleted, Template, and Correct. See Envelope Status Code Descriptions for information about the codes. The <any> element has been included to enable document author to extend his/her document with elements not specified by the schema.
Created	dateTime	Date and time when the envelope was created.
Deleted	dateTime	Date and time when the envelope was deleted.
Sent	dateTime	Date and time when the envelope was sent.
Delivered	dateTime	Last time the envelope was viewed by a signer.
Signed	dateTime	Last time a signer signed the document.
Completed	dateTime	Date and time when the envelope is completed.
Declined	dateTime	Date and time when the envelope was declined by the recipient.
TimedOut	dateTime	Date and time with the envelope is timed out
ACStatus	String	Returns any one of the following Authoritative copy status; Unknown, Original, Transferred, AuthoritativeCopy, AuthoritativeCopyExportPending, AuthoritativeCopyExported, DepositPending, Deposited, DepositedEO, or DepositFailed.
ACStatusDate	dateTime	The time at with authoritative copy was created
ACHolder	String	Contains the user name of the authoritative copy holder.

Name	Schema Type	Description
ACHolderEmail	LongString	Contains the email of the authoritative copy holder.
ACHolderLocation	String	Contains the location of the authoritative copy holder.
SigningLocation	SigningLocationCode	Specifies the physical location of the signer. Enumeration values are InPerson and Online
SenderIPaddress	String	IP address of the user who sent the envelope
CustomField	CustomField	Each custom field defined for the envelope is returned. In addition to the name and value of the field, it will include information as to whether the field was required to be provided. Also it mentions if it is shown in the DocuSign user interface.
VaultingDetails	VaultingDetails	This complex type contains information about EOD Transaction name, transaction ID and document profile ID.
EnvelopeIdStamping	Boolean	Specifies if the envelope stamping feature is enabled or not.
AuthoritativeCopy	Boolean	Return value true if the authoritative copy is created else returns false.
EnvelopeAttachment	Attachment	It would be possible to return attachments with envelope. This complex element contains data in base64Binary. It also contains label and type for the attachment.
DocumentStatuses	DocumentStatus	Only returned when calling RequestStatusEx and RequestStatusesEx. DocumentStatus includes: ID – this is the ID from the source system. Name – name of the document. TemplateName – template the document originated from if the envelope was created using CreateEnvelopeFromTemplates. Sequence – order in which the document appears in the envelope.
FormData	FormData	Reserved for future use. Will only be returned with the GetStatusInDocuSignConnectFormat API. This will return all the DocuSign Secure Fields and DocuSign PowerForms fields as XML.

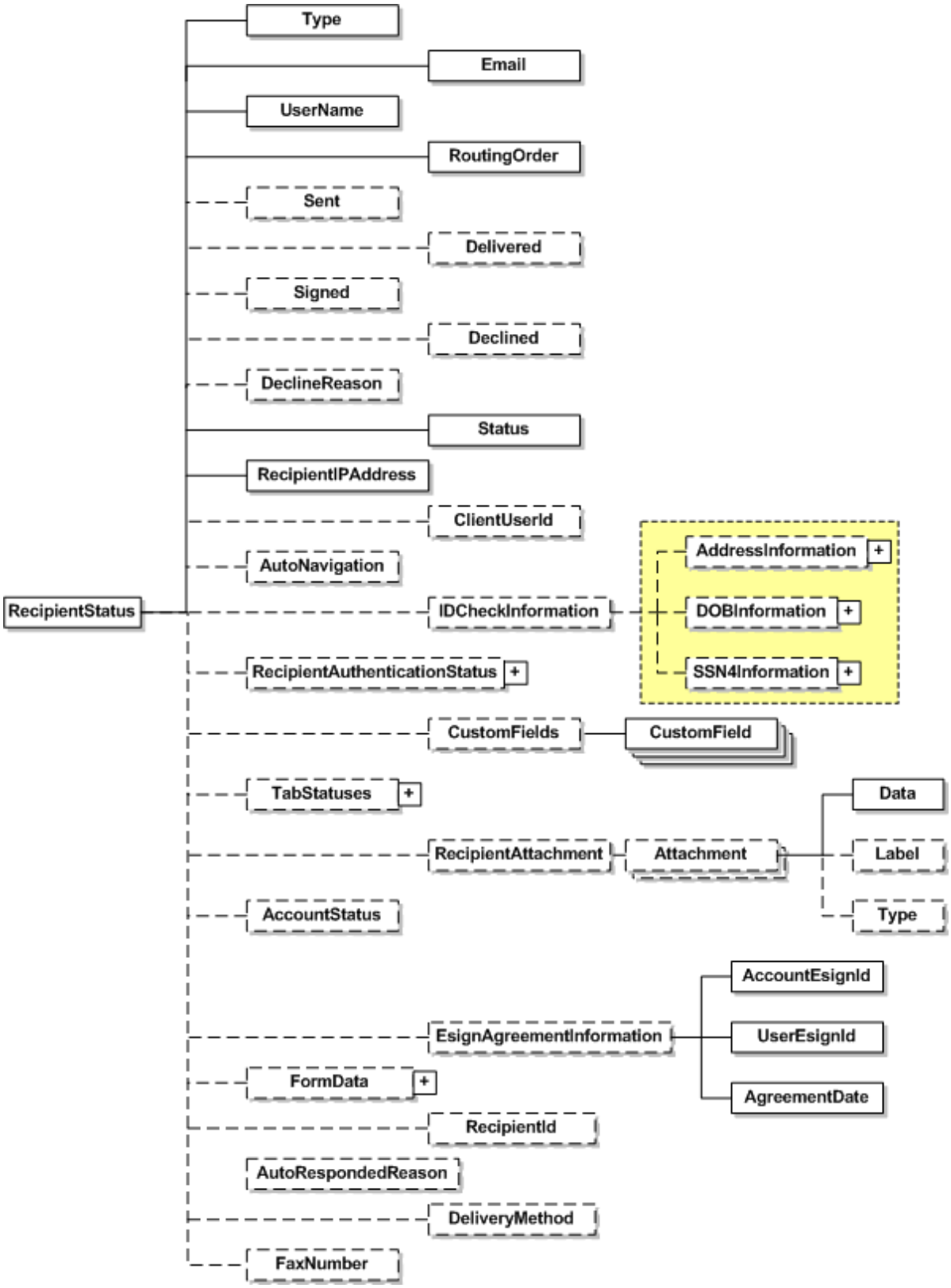
Envelope Status Code Descriptions

The table below provides descriptions of the Envelope Status Codes.

Code	Description
Created	The envelope is in a draft state and has not been sent out for signing.
Deleted	This is a legacy status and is no longer used.

Code	Description
Sent	An email notification with a link to the envelope has been sent to at least one recipient. The envelope remains in this state until all recipients have viewed it at a minimum.
Delivered	All recipients have viewed the document(s) in an envelope through the DocuSign signing web site. This is not an email delivery of the documents in an envelope.
Signed	The envelope has been signed by all the recipients. This is a temporary state during processing, after which the envelope is automatically moved to Completed status.
Completed	The envelope has been completed by all the recipients.
Declined	The envelope has been declined for signing by one of the recipients.
Voided	The envelope has been voided by the sender
TimedOut	This is a legacy status and is no longer used.
AuthoritativeCopy	The envelope is in an Authoritative state. Only “Copy” views of the documents will be shown.
TransferCompleted	The envelope has been transferred out of DocuSign to another authority.
Template	The envelope is a Template
Correct	The envelope has been opened by the sender for correction. The signing is stopped for envelopes with this status.

RecipientStatus



Name	Schema Type	Description
Type	RecipientTypeCode	This element controlled by enumerator tells the role of the recipient. It could be Signer, CarbonCopy, CertifiedDelivery, InPersonSigner, Agent, Editor or Intermediary. For InPersonSigner: The Email element should be that of the "Signing Host" (the person who will receive the email and assist the signer with the in-person signing process. The UserName element should be that of the "Signing Host" (the Email and UserName combination must match an active DocuSign user).
Email	Email	The recipient's email address.
UserName	UserName	User name of recipient.
RoutingOrder	PositiveShort	Routing order of the recipient in the envelope.
Sent	dateTime	Date and time when the envelope was sent.
Delivered	dateTime	Date and time when the envelope was viewed by the recipient.
Signed	dateTime	Date and time when the envelope was signed by the recipient.
Declined	dateTime	Date and time when the envelope was declined by the recipient.
DeclineReason	String	The reason given by the recipient while declining the envelope.
Status	RecipientStatusCode	Specifies the status of the recipient at the time of request. It could be Created, Sent, Delivered, Signed, Declined, Completed, FaxPending or AutoResponded. See Recipient Status Code Descriptions for information about the codes.
AutoRespondedReason	String	Only active if the Status is AutoResponded. Contains the SMTP message text returned from a recipient's automatic response.
Recipient IPAddress	String	The IP address of the recipient's most recent access of this envelope.
ClientUserId	LongString	Tells if the recipient is captive or not. If the ClientUserId is not null then the recipient is captive.

Name	Schema Type	Description
CustomField	LongString	Each custom field defined for the envelope is returned. In addition to the name and value of the field, it will include information as to whether the field was required to be provided. Also it mentions if it is shown in the DocuSign user interface.
AutoNavigation	Boolean	If returns true then autonavigation feature is enabled for the recipient.
IDCheckInformation	IDCheckInformationInput	Returns the following information about the recipient; Address (i.e. Street1, Street2, City, State, Zip and ZipPlus4), DOB and last four digits of SSN.
RecipientAuthenticationStatus	AuthenticationStatus	Returns the details about recipient authentication. It includes information regarding the IdCheck, Access Code and IDLookUp results. Tells if the status of the above check is passed or failed and also does a timestamp.
TabStatuses	TabStatus	Returns the status of the tabs. See the TabStatus section below for more information.
RecipientAttachment	Attachment	This complex element contains data in base64Binary. It also contains label and type for the attachment.
AccountStatus	String	Only returned when calling RequestStatusEx and RequestStatusesEx. Returns the status of the recipient's account.
EsignAgreementInformation	EsignAgreementInformation	Only returned when calling RequestStatusEx and RequestStatusesEx. Only returned if the recipient agreed. Returns the agreement information of the recipient. Returned items: AccountEsignID – account agreement ID. AgreementDate – date accepted. UserEsignID – user agreement ID.
FormData	FormData	Reserved for future use. Will only be returned with the GetStatusInDocuSignConnectFormat API. This will return all the DocuSign Secure Fields for the recipient.
RecipientId	String	Only returned when calling RequestStatusEx and RequestStatusesEx. Returns the DocuSign recipient Id mapped to the recipient.

Name	Schema Type	Description
DeliveryMethod	Delivery Method	This shows the delivery method used for the recipient. The two enumerations are Email or Fax.
FaxNumber	String	The fax number for the recipient. This is only active if the DeliveryMethod is Fax.

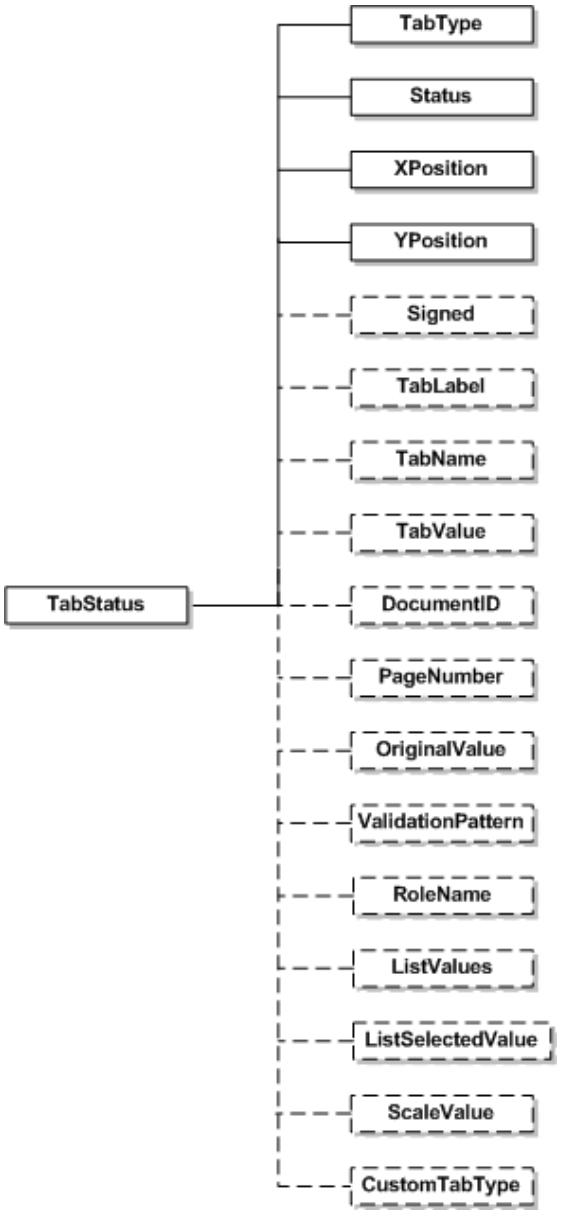
Recipient Status Code Descriptions

The table below provides descriptions of the Recipient Status Codes.

Code	Description
Created	The recipient is in a draft state. This is only associated with draft envelopes (envelopes with a Created status).
Sent	The recipient has been sent an email notification that it is their turn to sign and envelope.
Delivered	The recipient has viewed the document(s) in an envelope through the DocuSign signing web site. This is not an email delivery of the documents in an envelope.
Signed	The recipient has completed (signed) all required tags in an envelope. . This is a temporary state during processing, after which the recipient is automatically moved to Completed.
Declined	The recipient declined to sign the document(s) in the envelope.
Completed	The recipient has completed their actions (signing or other required actions if not a signer) for an envelope.
FaxPending	The recipient has finished signing and the system is waiting a fax attachment by the recipient before completing their signing step.
AutoResponded	The recipient's email system auto-responded (bounced-back) to the email from DocuSign. This status is used in the web console to inform senders about the bounced-back email. This is only used if "Send-on-behalf-of" is turned off for the account.

TabStatus

The TabStatus element contains information on tabs in the document.



Name	Schema Type	Description
TabType	TabTypeCode	Returns the tab type. This element is enumerator controlled and can have following values; InitialHere, SignHere, FullName, Company, Title, DateSigned, InitialHereOptional, EnvelopeID, Custom, SignerAttachment, SignHereOptional, Approve, or Decline. Note that FirstName, LastName, EmailAdress, and SignerAttachmentOptional tab types are not returned in this SOAP version.
Status	String	The status of individual tabs. It can have the enumeration values; Active, Signed, Declined and N/A

Name	Schema Type	Description
XPosition/YPosition	double	X position and Y position of the field. The X and Y positions originate at the upper left of the page and indicate the bottom left-most point of the field. For text fields, this is the upper left of the first letter of the text string. For signature fields, this is the upper left of the first letter of the signature stamp and does NOT include signature stamp image decoration which can surround the signature text.
Signed	dateTime	**Note:** The Signed information is not returned in the response, even though it is included in the SOAP WSDL. Date and time when the document was signed by the recipient.
TabLabel	String	This would be the label of the custom tab.
TabName	String	This would be the name of the custom tab.
TabValue	String	This element contains the value of the custom tab.
DocumentID	PositiveInteger	Only returned when calling RequestStatusEx and RequestStatusesEx. Returns the originating document ID.
PageNumber	PositiveInteger	Only returned when calling RequestStatusEx and RequestStatusesEx. Returns the page number the tab is on.
OriginalValue	String	Only returned when calling RequestStatusEx and RequestStatusesEx. Original value before signing.
ValidationPattern	String	Only returned when calling RequestStatusEx and RequestStatusesEx. Validation pattern that was originally passed in, if any.
RoleName	String	Only returned when calling RequestStatusEx and RequestStatusesEx. Only returned when the envelope originated from CreateEnvelopeFromTemplates. RoleName that this tab is associated with in the originating template.
ListValues	String	Only returned when calling RequestStatusEx and RequestStatusesEx. Returns the list values passed in when the envelope was created.
ListSelectedValue	String	Only returned when calling RequestStatusEx and RequestStatusesEx. Returns the selected value from ListValues.
ScaleValue	Decimal	Represents a decimal value of 0.0 to 1.0 position of the webcontrol scaling slider. A value of 1.0 represents full size.
CustomTabType	CustomTabType	Returns the custom tab type.

AuthenticationStatus

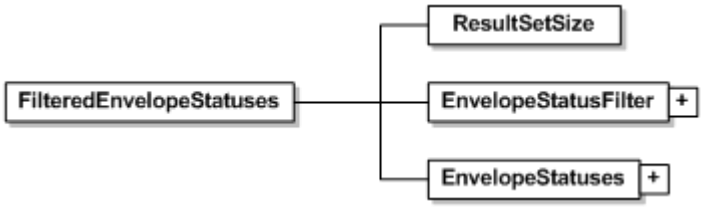
The AuthenticationStatus element contains information on the recipient authentication status. This only returns the applicable authentication result.

Name	Schema Type	Description
AccessCodeResult	EventResult	Result for an access code authentication check, it returns: Status – Pass or Fail dateTime – The date/time of the event FailureDescription – A string with the details of a failed authentication, if provided. This is only provided in case of a failed authentication. VendorFailureStatusCode – A string with a vendor status code, if provided. This is only provided in case of a failed authentication.
IDQuestionsResult	EventResult	Result for an ID question authentication check. It returns the same elements as shown in the AccessCodeResult above.
IDLookupResult	EventResult	Result for an ID lookup authentication check. It returns the same elements as shown in the AccessCodeResult above.

Name	Schema Type	Description
AgeVerifyResult	EventResult	Result for an age verification authentication check. It returns the same elements as shown in the AccessCodeResult above.
STANPinResult	EventResult	Result for a Student Authentication Network (STAN) check. It returns the same elements as shown in the AccessCodeResult above.
OFACResult	EventResult	Result for an Office of Foreign Asset Control (OFAC) check. It returns the same elements as shown in the AccessCodeResult above.
PhoneAuthResult	EventResult	Result for a phone authentication check. It returns the same elements as shown in the AccessCodeResult above.
LiveIDResult	EventResult	Result for a Live ID authentication check. It returns the same elements as shown in the AccessCodeResult above.
FacebookResult	EventResult	Result for a Facebook authentication check. It returns the same elements as shown in the AccessCodeResult above.
GoogleResult	EventResult	Result for a Google authentication check. It returns the same elements as shown in the AccessCodeResult above.
LinkedinResult	EventResult	Result for a LinkedIn authentication check. It returns the same elements as shown in the AccessCodeResult above.
SalesforceResult	EventResult	Result for a Salesforce authentication check. It returns the same elements as shown in the AccessCodeResult above.
TwitterResult	EventResult	Result for a Twitter authentication check. It returns the same elements as shown in the AccessCodeResult above.
OpenIDResult	EventResult	Result for an Open ID authentication check. It returns the same elements as shown in the AccessCodeResult above.
AnySocialIDResult	EventResult	Result for any other social ID authentication check. It returns the same elements as shown in the AccessCodeResult above.
YahooResult	EventResult	Result for a Yahoo authentication check. It returns the same elements as shown in the AccessCodeResult above.

FilteredEnvelopeStatuses

Schema



Name	Schema Type	Description
ResultSetSize	Integer	This element contains the total number of envelopes in the total result set. If this number exceeds 200 then at most 200 results is returned in this request.
EnvelopeStatusFilter	EnvelopeStatusFilter	This element contains the filter criteria exactly as passed in the request. See the EnvelopeStatusFilter information in the RequestStatuses and RequestStatuesEx section above for more information.
EnvelopeStatus	EnvelopeStatus	An EnvelopeStatus element (exactly as defined by the EnvelopeStatus method response above) is returned for each of the envelopes in the result set, up to a maximum size of 200 elements. See the EnvelopeStatus section above for more information.

Rules for RequestStatus, RequestStatuses, RequestStatusEx, RequestStatusesEx, EnvelopeStatus and FilteredEnvelopeStatuses

API user specific rules

- The EnvelopeID, UserName and Email specified in the RequestStatus, EnvelopeStatus RecipientStatus, and RequestStatuses methods shall not exceed 100 characters.
- AccountID specified in RequestStatuses should not exceed 100 characters.

Rules for Exceptions thrown by the API

- An envelope having the specified Envelope*ID* must exist in the system. Else, an exception with the error message “Envelope_Does_Not_Exist” is thrown.
- In RequestStatus method User defined by UserName and Email pair should have the permission to request for envelope status. Only sender recipient or a user in the same account as sender with account wide permission will be able to request for the status. Else an exception is thrown with error message “User_Lacks_Permissions”.
- In the RequestStatuses method, user defined by UserName and Email pair should belong to the account specified by AccountID. Else an exception with error message “User_Does_Not_Belong_To_Specified_Account” thrown.

- In the RequestStatuses method the envelopes for which the status is requested should owned by the account in the specified AccountID. Else an exception is thrown with error message "Account_Not_Authorized_For_Envelope".

GetFolderList and GetFolderItems

The GetFolderList and GetFolderItem methods are used to retrieve the list of folders, including shared folders, and envelopes in the folders. Using these calls is a more efficient way to get a list of envelopes than using RequestStatuses to request individual envelope statuses.

GetFolderList

GetFolderList requests the list of all folders, including shared folders, available for the account.

Schema

Name	Schema Type	Description
AccountId	DSXId	The account ID associated with the requested folders.
IncludeHierarchy	Boolean	If true, the folder's hierarchical positions are returned.

AvailableFolders

This is the response to a GetFolderList request and provides the list of folders, including shared folders, for the requestor.

Name	Schema Type	Description
Folders	Folder	An array of Folder objects with the FolderOwner and FolderTypeInfo.

Folder

Provides the Folder Owner and Folder Type information.

Name	Schema Type	Description
FolderOwner	String	The name of the user that currently owns folder.
FolderTypeInfo	FolderTypeInfo	The type of folder and in some cases the name of the folder. Values can be RecycleBin, Draft, Inbox, SentItems or Normal.

FolderTypeInfo

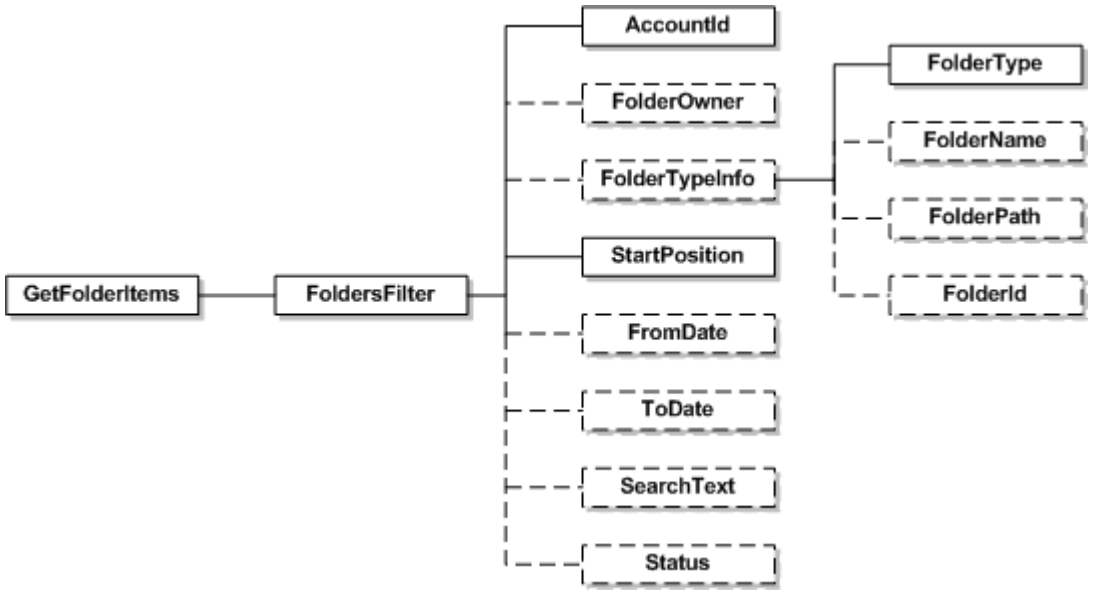
Provides the Folder Type information and, if the Folder Type is Normal, the Folder Name.

Name	Schema Type	Description
FolderType	FolderType	The type of folder based on the GetFolders call. Values can be RecycleBin, Draft, Inbox, SentItems or Normal.
FolderName	String	The name of the folder. This only appears if the FolderType is Normal.
FolderPath	String	the hierarchical path to the folder, comprised of bar (|) delimited FolderIds (for example: grandparent_folder|parent_folder|folder)
FolderId	String	The folder's (Guid) identifier.

GetFolderItems

GetFolderItems requests all the envelopes in a specified folder for the requestor or a Folder Owner.

Schema



Name	Schema Type	Description
AccountId	DSXId	The account ID about which the call is requesting folder information. This is used in conjunction with the user passed in the API request.
FolderOwner	String	Optional field. The name of the user that currently owns the requested folders.

Name	Schema Type	Description
FolderTypeInfo	FolderTypeInfo	Optional FolderTypeInfo object. The type of folder and in some cases the name of the folder. Values can be RecycleBin, Draft, Inbox, SentItems or Normal. If no value is entered, the default search is all folders. See FolderTypeInfo for more information.
StartPosition	Integer	The position of the folder items to return. This is used for repeated calls, when the number of envelopes returned is too much for one return (calls return 100 envelopes at a time). The default value is 0.
FromDate	dateTime	Optional field. Only return items on or after this date. If no value is provided, the default search is the previous 30 days.
ToDate	dateTime	Optional field. Only return items up to this date. If no value is provided, the default search is to the current date.
SearchText	String	Optional field. The search text used to search the items of the envelope. The search looks at recipient names and emails, envelope custom fields, sender name, and subject.
Status	String	Optional field. The current status of the envelope. If no value is provided, the default search is all/any status.

FolderResults

This is the response for a GetFolderItems request and provides the list of envelopes in the requested folder(s) for the requestor.

Name	Schema Type	Description
ResultSetSize	Integer	The total number of envelopes in the folder for the dates requested.
StartPosition	Integer	The starting position of the folder items returned (only 100 results are returned at a time).
EndPosition	Integer	The end position of the folder items returned.
FolderItems	FolderItem	Provides information about the envelopes in the specified folder.
FolderTypeInfo	FolderTypeInfo	The type of folder and in some cases the name of the folder. Values can be RecycleBin, Draft, Inbox, SentItems or Normal. See FolderTypeInfo for more information.

FolderItem

FolderItem provides the information about each envelope in a folder.

Name	Schema Type	Description
Owner	String	The name of the user that currently owns the envelope.
EnvelopeID	LocalId	The unique ID for the envelope.
Status	String	The current status of the envelope.
SenderName	String	The name of the sender of the envelope.
SenderEmail	String	The email for the sender of the envelope.
SenderCompany	String	The company name of the sender of the envelope.
RecipientStatuses	RecipientStatus	A complex element that contains the status of each of the envelope recipients
CustomFields	CustomField	A complex element that contains a list of names and values for any custom fields.
Created	dateTime	Date and time when the envelope was created.
Sent	dateTime	Date and time when the envelope was sent.
Completed	dateTime	Date and time when the envelope was completed.
Subject	String	The envelope subject, as provided by the envelope creator.

Code Samples

GetFolderList – C#

```
// Create the folders filter with an account ID
DocuSignWeb.FoldersFilter filter = new DocuSignWeb.FoldersFilter();
filter.AccountId = _accountId;

// Now, call the method to get a list of the folders on the
// specified account
DocuSignWeb.AvailableFolders folders = _apiClient.GetFolderList(filter);

// Everyone will at least have the folders:
// Draft, Sent Items, Inbox and Deleted Items
// Print out some information about the returned folders
Console.WriteLine("The folders available on account {0} are:", _accountId);

foreach (DocuSignWeb.Folder folder in folders.Folders)
{
    Console.WriteLine("\t{0} owns folder {1}",
        folder.FolderOwner.UserName, folder.FolderTypeInfo.FolderName);
}
```

GetFolderList – PHP

```
// Create the folders filter with an account ID
$filter = new FoldersFilter();
$filter->AccountId = $AccountID;

// Send
$getFolderListparams = new GetFolderList();
```

```
$getFolderListparams->FoldersFilter = $filter;
$response = $api->GetFolderList($getFolderListparams);
```

GetFolderItems – C#

```
// Create the folder filter to specify the scope of your search
// Here, we are limiting the item search to the inbox
// You can also limit by owner, date, status and position
DocuSignWeb.FolderFilter filter = new DocuSignWeb.FolderFilter();
filter.AccountId = _accountId;
filter.FolderTypeInfo = new DocuSignWeb.FolderTypeInfo();
filter.FolderTypeInfo.FolderType = DocuSignWeb.FolderType.Inbox;

// Now, call the method with the filter we created
DocuSignWeb.FolderResults results = _apiClient.GetFolderItems(filter);

// If there are results, print out the details
if (results.ResultSetSize > 0)
{
    // Loop through and print out some information about the results
    Console.WriteLine("Envelopes in the inbox are:");
    foreach (DocuSignWeb.FolderItem item in results.FolderItems)
    {
        Console.WriteLine("\tEnvelope {0} has status of: {1}",
            item.EnvelopeId, item.Status);
    }
}
```

GetFolderItems - PHP

```
// Create the folder filter to specify the scope of your search
// Here, we are limiting the item search to the inbox
// You can also limit by owner, date, status and position
$filter = new FolderFilter();
$filter->AccountId = $AccountID;
$filter->StartPosition = 0;
$filterTypeInfo = new FolderTypeInfo();
$filterTypeInfo->FolderType = FolderType::Inbox;
$filter->FolderTypeInfo = $filterTypeInfo;

// Send
$getFolderItemsparams = new GetFolderItems();
$getFolderItemsparams->FolderFilter = $filter;
$response = $api->GetFolderItems($getFolderItemsparams);
```

Ping

The Ping API method enables API users make a simple call to the API service to determine its active state.

Schema

Sample Request XML:

```
SOAPAction: "http://www.docusign.net/API/3.0/Ping"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Ping xmlns="http://www.docusign.net/API/3.0" />
```

```
  </soap:Body>
</soap:Envelope>
```

Return XML

This method simply returns “true” in the result XML if the calling application was able to reach it.

PurgeDocuments

This method can be used to purge envelope documents from the DocuSign system. When the PurgeDocuments method is used, the envelope documents are placed in a purge queue for deletion in 14 days. A warning email notification is sent to the sender and recipients associated with the envelope notifying them that the envelope document will be deleted in 14 days and providing a link to the documents. Another email is sent 7 days later with the same message. At the end of the 14-day period, the envelope documents are deleted from the system. The envelope information for the document will remain, but the documents are removed.

> **Note:** If you set up a Document Retention policy for your account, by specifying the number of days to retain documents in the Account Preferences Features section, at the end of the retention period the documents are placed on that 14-day purge queue and the warning emails are sent. So in effect, setting Document Retention policy is the same as setting a schedule for calling PurgeDocuments.

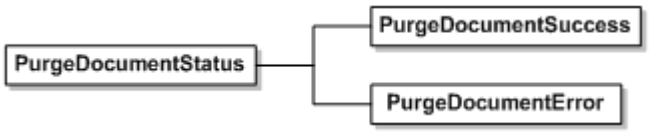
Schema

Name	Schema Type	Description
EnvelopeId	DSXId	Envelope ID to purge the documents from.

This method returns PurgeDocumentStatus upon execution completion.

PurgeDocumentsStatus

Schema



Name	Schema Type	Description
PurgeDocumentSuccess	Boolean	True if successful.
PurgeDocumentError	String	If PurgeDocumentSuccess is false an error description is returned.

Rules and Exceptions for PurgeDocuments

- Envelope_Does_Not_Exist – invalid envelope ID passed. This exception will be thrown.
- User_Lacks_Permissions - The user making the API calls must have permissions to export Authoritative Copy Documents. This exception will be thrown.

- If PurgeDocumentSuccess returns false one of the following error conditions occurred:
  - Only the sender may remove the envelope documents.
  - Envelope is an authoritative copy, documents cannot be removed.
  - Envelope is not complete, documents cannot be removed.

Sample Code

PurgeDocuments – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Purge the documents from the envelope
DocuSignWeb.PurgeDocumentStatus pStatus = _apiClient.PurgeDocuments(status.EnvelopeID);

// Confirm that the purge succeeded
Console.WriteLine("Purging the envelope worked? {0}", pStatus.PurgeDocumentSuccess);
```

PurgeDocuments – PHP

```
// Create and send envelope as shown in linked code
$response = <create and send envelope>;
$createResult = $response->CreateAndSendEnvelopeResult;

// Purge the documents from the envelope
$purgeDocumentsparams = new PurgeDocuments();
$purgeDocumentsparams->EnvelopeID = $createResult->EnvelopeID;
$response = $api->PurgeDocuments($purgeDocumentsparams);
```

PurgeDocumentsAndMetaData

This method can be used to purge envelope documents and metadata from the DocuSign system. The metadata that is removed along with the document includes the DocuSign tab data and any attachments that were provided with the envelope creation request. In all other respects, this is identical to PurgeDocuments.

When this method is used, the envelope documents are placed in a purge queue for deletion in 14 days. A warning email notification is sent to the sender and recipients associated with the envelope notifying them that the envelope document will be deleted in 14 days and providing a link to the documents. Another email is sent 7 days later with the same message. At the end of the 14-day period, the envelope documents and metadata are deleted from the system.

Schema

Name	Schema Type	Description
EnvelopeID	DSXId	Envelope ID to purge from the system.

Sample Request XML

```
POST /api/3.0/dsapi.asmx HTTP/1.1
Host: demo.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
```

```
SOAPAction: "http://www.docusign.net/API/3.0/PurgeDocumentsAndMetaData"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <PurgeDocumentsAndMetaData xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
    </PurgeDocumentsAndMetaData>
  </soap:Body>
</soap:Envelope>
```

The response either a success or failure. If the call fails an error code is provided.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <PurgeDocumentsAndMetaDataResponse xmlns="http://www.docusign.net/API/3.0">
      <PurgeDocumentsAndMetaDataResult>
        <PurgeDocumentSuccess>boolean</PurgeDocumentSuccess>
        <PurgeDocumentError>string</PurgeDocumentError>
      </PurgeDocumentsAndMetaDataResult>
    </PurgeDocumentsAndMetaDataResponse>
  </soap:Body>
</soap:Envelope>
```

RequestEnvelope

The RequestEnvelope call returns an API Envelope object containing all the data of an envelope. The envelope must be owned by the API user.

Name	Schema Type	Description
EnvelopeID	String	The envelope ID of the user's envelope
IncludeDocumentBytes	Boolean	If set to "true" the document bytes are returned with the envelope information.

The response to the RequestEnvelope call is:

Name	Schema Type	Description
Envelope	complexType	A complex API Envelope object

Additional Errors for RequestEnvelope

In addition to the normal authentication check errors, the following error is also added.

- User_Lacks-Permissions – The user does not own have permission to access the envelope.
- Envelope_Does_Not_Exist – Invalid envelope ID passed.

Schema for RequestEnvelope

```
  <xs:element name="RequestEnvelope">
    <xs:complexType>
      <xs:all>
        <xs:element name="EnvelopeID" type="dsx:DSXId" />
        <xs:element name="IncludeDocumentBytes" type="xs:boolean" />
      </xs:all>
    </xs:complexType>
  </xs:element>
```

Sample Code

RequestEnvelope – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Call to request the envelope with the envelope's ID
DocuSignWeb.Envelope returnEnvelope = _apiClient.RequestEnvelope(status.EnvelopeID,
false);

// Display information about the return envelope
Console.WriteLine("Return envelope has a subject of \"{0}\"", returnEnvelope.Subject);
```

RequestEnvelope: PHP

```
// Create and send an envelope as shown in linked code
$response = <create amd send envelope>;
$createResult = $response->CreateAndSendEnvelopeResult;

// Request envelope with envelope id
$requestEnvelopeparams = new RequestEnvelope();
$requestEnvelopeparams->EnvelopeID = $createResult->EnvelopeID;
$requestEnvelopeparams->IncludeDocumentBytes = false;
$response = $api->RequestEnvelope($requestEnvelopeparams);
```

RequestEnvelopeWithDocumentFields

The RequestEnvelopeWithDocumentFields call is similar to the RequestStatus call. It returns the API Envelope object containing all the envelope data, including the document custom DocumentFields. The envelope must be owned by the API user.

Schema

Name	Schema Type	Description
EnvelopeID	String	The Envelope ID for the requested envelope.
IncludeDocumentBytes	Boolean	If set to “true” the document bytes are returned with the envelope information.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/AddMembersToAccount"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestEnvelopeWithDocumentFields xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
      <IncludeDocumentBytes>boolean</IncludeDocumentBytes>
    </RequestEnvelopeWithDocumentFields>
  </soap:Body>
</soap:Envelope>
```

The response returns the requested Envelope object, including the document custom DocumentFields.

RequestStatusChanges

The RequestStatusChanges method requests the envelope status changes for the envelopes for account on or after the specified date/time.

> **Note:** The SOAP API calls RequestStatusChanges and RequestStatusCodes were created to provide a high-efficiency way to determine envelope status changes. These calls are the preferred methods for determining status.
>
> Generally, customers should poll as infrequently as possible. But for customers that generate a large number of envelopes, frequent polling using these calls is an acceptable option. However, when polling at frequent intervals, the call should include an appropriately short time range for *StatusChangedSince*. For example, if you are checking for status changes every 15 minutes, do not use a *StatusChangedSince* of 7 days ago.

If no *UserInfo* is specified and if the *StatusChangedSince* is more than 24 hours ago, this call will return information for all users, but only for the previous 24 hours. If no *UserInfo* is specified and *StatusChangedSince* is less than 24 hours ago, this call will return information for all users for the specified time.

Schema

Name	Schema Type	Description
AccountID	DSXId	The account ID for the envelope status changes being requested.
StatusChangedSince	DateTime	Specifies the date/time when the request begins checking for status changes for envelopes in the account.

Name	Schema Type	Description
Statuses	EnvelopeStatusChange	Optional array of envelope status codes. If provided, this indicates the envelope status code changes being requested. Possible values are: Voided, Created, Deleted, Sent, Delivered, Signed, Completed, Declined, TimedOut and Processing.
UserInfo	UserInfo	Optional UserInfo object. If provided, this indicates the user whose envelopes are being requested. UserInfo contains: UserName – The user's name as listed in the account. Email – The email associated with the user name.

FilteredEnvelopeStatusChanges

This is the response for RequestStatusChanges. It returns the current state of the requested envelopes in an array.

Name	Schema Type	Description
EnvelopeStatusChanges	EnvelopeStatusChanges	An array of EnvelopeStatusChange objects. Each EnvelopeStatusChange contains: EnvelopeID – The envelope ID. Status – The EnvelopeStatusCode for the envelope. StatusChanged – The date and time the status change occurred.
ResultSetSize	Integer	This is the total number of items returned in EnvelopeStatusChanges.

Sample Code

RequestStatusChanges – C#

```
// Create the status change filter to specify the scope of your search
// Here, we are limiting the search to envelopes changed today
// You can also limit by user and status
DocuSignWeb.EnvelopeStatusChangeFilter filter =
    new DocuSignWeb.EnvelopeStatusChangeFilter();

filter.AccountId = _accountId;

DocuSignWeb.EnvelopeStatusFilterBeginDateTime begin =
    new DocuSignWeb.EnvelopeStatusFilterBeginDateTime();

begin.Value = DateTime.Today;
filter.StatusChangedSince = begin.Value;

// Now, make the call with the filter we created
DocuSignWeb.FilteredEnvelopeStatusChanges changes =
    _apiClient.RequestStatusChanges(filter);
```

```
if (changes.ResultSetSize > 0)
{
    // Loop through and print out some information about the results
    Console.WriteLine("Changes since today are:");
    foreach (DocuSignWeb.EnvelopeStatusChange change
        in changes.EnvelopeStatusChanges)
    {
        Console.WriteLine("\tEnvelope {0} has status: {1}\n",
            change.EnvelopeID, change.Status);
    }
}
```

RequestStatusChanges – PHP

```
// Create the status change filter to specify the scope of your search
// Here, we are limiting the search to envelopes changed today
// You can also limit by user and status
$filter = new EnvelopeStatusChangeFilter();
$filter->AccountId = $AccountID;
$filter->StatusChangedSince = todayXsdDate();

// Send
$requestStatusChangesparams = new RequestStatusChanges();
$requestStatusChangesparams->EnvelopeStatusChangeFilter = $filter;
$response = $api->RequestStatusChanges($requestStatusChangesparams);
```

RequestStatusCodes

The RequestStatusCodes method requests the current state (Delivered, Complete, Voided, etc.) of the specified envelopes.

Note: The SOAP API calls RequestStatusChanges and RequestStatusCodes were created to provide a high-efficiency way to determine envelope status changes. These calls are the preferred methods for determining status.

Generally, customers should poll as infrequently as possible. But for customers that generate a large number of envelopes, frequent polling using these calls is an acceptable option. However, when polling at frequent intervals, the call should include an appropriately short time range for the *BeginDateTime* and *SendDateTime*. For example, if you are checking for status changes every 15 minutes, do not use a *BeginDateTime* and *SendDateTime* with a difference of 7 days.

Request Envelope Statuses Notes

The SOAP calls for RequestStatuses(Ex) and RequestStatusCodes use certain filters to find results. In some cases requests are check for “any status change” instead of the just the single status requested. In these cases, more envelopes might be returned by the request than otherwise would be. For example, for a request with the begin date is set to Jan 1st, an end date set to Jan 7th and the StatusQualifier set to “Delivered” – the response set might contain envelopes that were created during that time period, but not delivered during the time period.

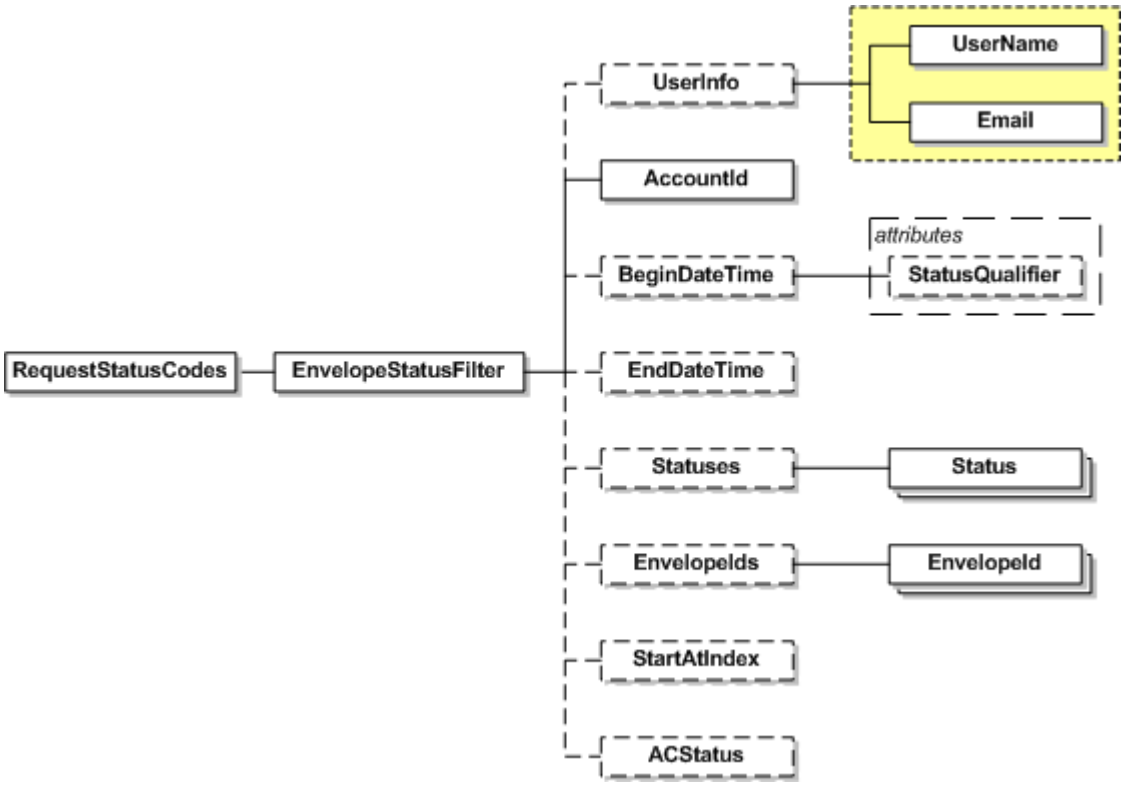
To avoid unnecessary database queries, the DocuSign system checks requests to ensure that the added filters will not result in a zero-size response before acting on the request. The following table shows the valid envelope statuses (in the Valid Current Statuses column) for the status qualifiers in the request. If the status and status qualifiers in the API request do not contain any of the values shown in the valid current statuses column, then an empty list is returned.

For example, a request with a StatusQualifier of “Delivered” and a status of “Created,Sent”, DocuSign will always return an empty list. This is because the request essentially translates to: find the envelopes that were Delivered between the begin and end dates that have a current status of Created or Sent, and since an envelope that has been delivered can never have a status of Created or Sent, a zero-size response would be generated. In this case, DocuSign does not run the request, but just returns the empty list.

Client applications should check that the statuses they are requesting make sense for a given status qualifier.

BeginDateTime - StatusQualifier	Effective Status Qualifier	Valid Current Statuses
Any (changed)	StatusChanged	Any, Created, Sent, Delivered, Signed, Completed, Declined, Voided, Deleted
Created	Created	Any, Created, Sent, Delivered, Signed, Completed, Declined, Voided, Deleted
Sent	Sent	Any, Sent, Delivered, Signed, Completed, Declined, Voided, Deleted
Delivered	StatusChanged	Any, Delivered, Signed, Completed, Declined, Voided, Deleted
Signed	StatusChanged	Any, Signed, Completed, Declined, Voided, Deleted
Completed	Completed	Any, Completed, Declined, Voided, Deleted
Declined	StatusChanged	Any, Declined, Deleted
TimedOut - Always return zero results	StatusChanged	Any, Voided, Deleted
Voided	Voided	Any, Voided, Deleted
Deleted	StatusChanged	Any, Deleted

Schema



Name	Schema Type	Description
UserInfo	UserInfo	Optional field. The UserName-Email pair of the person whose envelope details are requested.
AccountId	DSXId	Account ID for which envelope status is requested.
BeginDateTime	dateTime	Optional Field. Specifies the start date of the date range for which envelopes are requested. If no value is provided, the default search is the previous 30 days. The date range also can have a status qualifier attribute, which specifies the envelope statuses being requested. The StatusQualifier options are: Completed, Created (the default and what is used if Template or Processing is passed in), Sent, and Voided; any other value becomes StatusChanged.
EndDateTime	dateTime	Optional Field. Specifies the end date of the date range for which envelopes are requested. If no value is provided, the default search is to the current date.
Statuses	EnvelopeStatusCode	Optional Field. Specifies the status of the envelope at the time of request. This is enumerator controlled.

Name	Schema Type	Description
EnvelopeIds	DSXId	Optional Field. Specifies the envelope IDs of envelopes for which status is requested.
StartAtIndex	nonNegativeInteger	Optional Field. If envelope status result sets exceed 200 envelopes, this element can be used to specify that the method should return envelopes at the specified index. The first index is 0, and should be used in the first call to RequestStatuses. Typically this number is a multiple of 200.
ACStatus	EnvelopeACStatusCode	Specifies the Authoritative Copy Status for the envelopes. The possible values are: Unknown, Original, Transferred, AuthoritativeCopy, AuthoritativeCopyExportPending, AuthoritativeCopyExported, DepositPending, Deposited, DepositedEO, or DepositFailed.

FilteredEnvelopeStatusChanges

This is the response for RequestStatusCodes. It returns the current state of the requested envelopes in an array.

Name	Schema Type	Description
EnvelopeStatusChanges	EnvelopeStatusChanges	An array of EnvelopeStatusChange objects. Each EnvelopeStatusChange contains: EnvelopeID – The envelope ID. Status – The EnvelopeStatusCode for the envelope. StatusChanged – The date and time the status change occurred.
ResultSetSize	Integer	This is the total number of items returned in EnvelopeStatusChanges.

Code Samples

RequestStatusCodes – C#

```
// Create the status change filter to specify the scope of your search
// Here, we are limiting the search to envelopes changed today
// You can also limit by user and status
DocuSignWeb.EnvelopeStatusFilter filter =
    new DocuSignWeb.EnvelopeStatusFilter();

filter.AccountId = _accountId;

DocuSignWeb.EnvelopeStatusFilterBeginDateTime begin =
    new DocuSignWeb.EnvelopeStatusFilterBeginDateTime();

begin.Value = DateTime.Today;
filter.BeginDateTime = begin;

// Now, make the call with the filter we created
```

```
DocuSignWeb.FilteredEnvelopeStatusChanges codes =
    _apiClient.RequestStatusCodes(filter);

if (codes.ResultSetSize > 0)
{
    // Loop through and print out some information about the results
    Console.WriteLine("Changes since today are:");
    foreach (DocuSignWeb.EnvelopeStatusChange code
        in codes.EnvelopeStatusChanges)
    {
        Console.WriteLine("\tEnvelope {0} has status: {1}\n",
            code.EnvelopeID, code.Status);
    }
}
```

RequestStatusCodes - PHP

```
// Create the status change filter to specify the scope of your search
// Here, we are limiting the search to envelopes changed today
// You can also limit by user and status
$filter = new EnvelopeStatusFilter();
$filter->AccountId = $AccountID;
$filter->BeginDateTime = todayXsdDate();

// Send
$requestStatusCodesparams = new RequestStatusCodes();
$requestStatusCodesparams->EnvelopeStatusFilter = $filter;
$response = $api->RequestStatusCodes($requestStatusCodesparams);
```

SetSharedAccess

SetSharedAccess is used to set or update the shared item status for one or more users and types of items. The users and item types listed in the SharedAccessFilter. This call can only be used by users with account administration privileges.

Changes to the shared items status are not additive; the change always replaces the current status.

Schema

Name	Schema Type	Description
AccountId	String	Account Id
SharedItems	Array of SharedItems objects	An array of shared item information to be updated. See the description below for more information.

Description of *SharedItems* objects

Name	Schema Type	Description
User	User object	The UserId of the user for which the shared item status is being changed.
SharedItem	Array of SharedItem objects	Array of SharedItem objects. A SharedItem object is required for each shared item access status being changed. See the description below for more information.

Description of *SharedItem* objects

Name	Schema Type	Description
User	User object	The UserId of the account user for whom the shared access status is being changed. The status change is relative to the user set by the UserId enclosed in the SharedItems object previously described.
Shared	Shared status	Specifies the change to the sharing status between the user and other account users for the specified item type. The accepted values are: • NotShared: This removes all sharing between the user and the account user for this item type. • SharedTo: This only shares access from the item type for the account user to the user. There is no shared access from the user to the account user (only shared to the user). • SharedFrom: This only shares access to the item type from the user to the account user. There is no shared access to the user from the account user with this status (only shared from the user). • SharedToAndFrom: This shares access to and from the item type for both the user and the account user.
ItemType	ItemType	The type of the shared item. Currently the only ItemType supported is Envelopes.

Sample Request XML

```
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
    <SetSharedAccess xmlns="http://www.docusign.net/API/3.0">
      <SetSharedAccessRequest>
        <AccountId>string</AccountId>
        <SharedItems>
          <User>
            <UserId>be27e2d2-01b6-48bb-be5b-24933f750e73</UserId>
          </User>
          <SharedItem>
            <User>
              <UserId>1f2289df-01d7-4bd7-b581-bb70362ea662</UserId>
            </User>
            <Shared>SharedTo</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>1272b5d0-f7d1-4295-bcb0-ce8ccafdffd4</UserId>
            </User>
```

```
            <Shared>SharedToAndFrom</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>284ede51-fa8c-445b-af09-141f9bc22eda</UserId>
            </User>
            <Shared>NotShared</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>fefc71c2-cede-49ff-946a-d6198df3909d</UserId>
            </User>
            <Shared>SharedFrom</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
        </SharedItems>
        <SharedItems>
          <User>
            <UserId>0af4e014-f7ff-4c5f-9a3e-00d6f4157223</UserId>
          </User>
          <SharedItem>
            <User>
              <UserId>1f2289df-01d7-4bd7-b581-bb70362ea662</UserId>
            </User>
            <Shared>SharedTo</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>1272b5d0-f7d1-4295-bcb0-ce8ccafdffd4</UserId>
            </User>
            <Shared>SharedToAndFrom</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>284ede51-fa8c-445b-af09-141f9bc22eda</UserId>
            </User>
            <Shared>NotShared</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>fefc71c2-cede-49ff-946a-d6198df3909d</UserId>
            </User>
            <Shared>SharedFrom</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
        </SharedItems>
      </SetSharedAccessRequest>
    </SetSharedAccess>
  </env:Body>
</env:Envelope>
```

The response returns the information from the changes and either a success or failure. If the call fails an error code is provided.

Sample Response XML

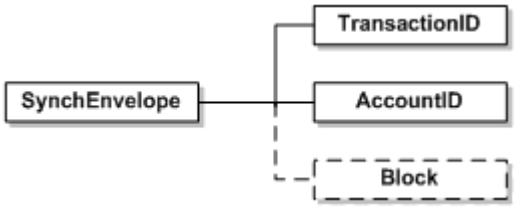
```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SetSharedAccessResponse xmlns="http://www.docusign.net/API/3.0">
      <SetSharedAccessResult>
        <AccountId>string</AccountId>
        <SharedItems>
          <User>
            <UserId>be27e2d2-01b6-48bb-be5b-24933f750e73</UserId>
          </User>
          <ErrorDetails>
            <ErrorCode>error code</ErrorCode>
            <Message>error message</Message>
          </ErrorDetails>
        </SharedItems>
        <SharedItems>
          <User>
            <UserId>0af4e014-f7ff-4c5f-9a3e-00d6f4157223</UserId>
            <Email>john.doe@docusign.com</Email>
            <UserName>John Doe</UserName>
          </User>
          <SharedItem>
            <User>
              <UserId>1f2289df-01d7-4bd7-b581-bb70362ea662</UserId>
              <Email>jane.doe@docusign.com</Email>
              <UserName>Jane Doe</UserName>
            </User>
            <Shared>SharedTo</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>1272b5d0-f7d1-4295-bcb0-ce8ccafdffd4</UserId>
              <Email>juan.doe@docusign.com</Email>
              <UserName>Juan Doe</UserName>
            </User>
            <Shared>SharedToAndFrom</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>284ede51-fa8c-445b-af09-141f9bc22eda</UserId>
              <Email>ted.doe@docusign.com</Email>
              <UserName>Ted Doe</UserName>
            </User>
            <Shared>NotShared</Shared>
            <ItemType>Envelopes</ItemType>
          </SharedItem>
          <SharedItem>
            <User>
              <UserId>fefc71c2-cede-49ff-946a-d6198df3909d</UserId>
            </User>
            <ErrorDetails>
              <ErrorCode>error code</ErrorCode>
              <Message>error message</Message>
            </ErrorDetails>
          </SharedItem>
        </SharedItems>
      </SetSharedAccessResult>
    </SetSharedAccessResponse>
  </soap:Body>
```

```
</soap:Envelope>
```

SynchEnvelope

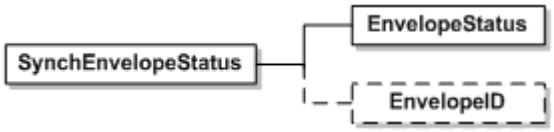
This method is only useful when the 'Asynchronous' flag is set to true on a *CreateAndSendEnvelope* call. It will determine when the queued envelope has been processed by the system.

Schema



Name	Schema Type	Description
TransactionID	LongString	External ID that is passed in CreateAndSendEnvelope when the 'Asynchronous' flag is set to true.
AccountID	DSXId	Account Id of the user who created the envelope
Block	Boolean	If true, this call will wait to return until the queued envelope is done processing. The max time it will wait is 10 minutes.

SynchEnvelopeStatus



Name	Schema Type	Description
EnvelopeStatus	EnvelopeStatusCode	Status of the envelope at that specific time. Enumeration values could be Voided, Created, Deleted, Sent, Delivered, Signed, Completed, Declined, TimedOut and Processing.
EnvelopeID	LocalId	The unique Id for the envelope.

Sample Code

SynchEnvelope – C#

```
// Create envelope as shown in linked code
DocuSignWeb.Envelope envelope = <create envelope>;

// Assign it a transaction ID and make it asynchronous send
envelope.TransactionID = System.Guid.NewGuid().ToString();
envelope.Asynchronous = true;
```

```
// Go ahead and send the envelope
DocuSignWeb.EnvelopeStatus status = _apiClient.SendEnvelope(envelope);

// Request a synch using the transaction ID and account ID
// but don't block it from processing
DocuSignWeb.SynchEnvelopeStatus synchStatus =
    _apiClient.SynchEnvelope(envelope.TransactionID, _accountId, false);

// Display the status we got back
Console.WriteLine("Synch envelope status is {0}", synchStatus.EnvelopeStatus);
```

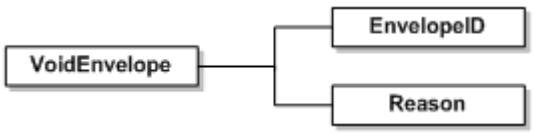
SynchEnvelope – PHP

```
// Create an envelope as shown in linked code
$env = <create envelope>;

// Assign a transaction ID and make it an asynchronous send
$env->TransactionID = guid(); //Helper function link
$env->Asynchronous = true;

// Send the envelope
$sendEnvelopeparams = new SendEnvelope();
$sendEnvelopeparams->Envelope = $env;
$status = $api->SendEnvelope($sendEnvelopeparams)->CreateAndSendEnvelopeResult;


// Request a sych without blocking
$synchEnvelopeparams = new SynchEnvelope();
$synchEnvelopeparams->AccountID = $AccountID;
$synchEnvelopeparams->Block = false;
$synchEnvelopeparams->TransactionID = $env->TransactionID;
$response = $api->SynchEnvelope($synchEnvelopeparams);
```

VoidEnvelope

This section describes the method available to void envelopes. Only incomplete envelopes can be voided. VoidEnvelope method is used to void an envelope.

Schema



Name	Schema Type	Description
EnvelopeID	DSXId	This is the envelope ID of the envelope which needs to be voided.
Reason	Reason	The reason for voiding the envelope. Envelope recipients, when notified of the void, are shown this reason.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/VoidEnvelope"

<?xml version="1.0" encoding="utf-8"?>
```

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <VoidEnvelope xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
      <Reason>string</Reason>
    </VoidEnvelope>
  </soap:Body>
</soap:Envelope>
```

VoidEnvelopeStatus

VoidEnvelopeStatus is the response method for VoidEnvelopes.

Schema

Name	Schema Type	Description
VoidSuccess	Boolean	Set to true if the void operation succeeded.

Rules for VoidEnvelope

- The length of any of the EnvelopeID must not exceed 100 characters. Else, the XML validation fails and the processor throws a Validation error.
- The length of reason should not exceed 200 characters and should have a minimum of 1 character.
- EnvelopeID specified in the VoidEnvelope request should exist else an exception is thrown with error message “Envelope_Does_Not_Exist” .
- Only envelopes in the 'Sent' or 'Delivered' states may be voided. Else “Envelope_Cannot_Void_Invalid_State” exception will be thrown.
- Only the sender of envelope can void the envelope. If a user other than sender tries to void an envelope then exception “User_Not_Envelope_Sender” will be thrown.

Sample Code

VoidEnvelope – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Void the envelope
DocuSignWeb.VoidEnvelopeStatus voidStatus = _apiClient.VoidEnvelope(status.EnvelopeID,
    "voiding reason");

// Confirm that the envelope was voided
Console.WriteLine("Voiding the envelope worked? {0}", voidStatus.VoidSuccess);
```

VoidEnvelope – PHP

```
// Create and send an envelope as shown in linked code
$response = <create and send envelope>;
$status = $response->CreateAndSendEnvelopeResult;
```

```
// Void the envelope
$voidEnvelopeparams = new VoidEnvelope();
$voidEnvelopeparams->EnvelopeID = $status->EnvelopeID;
$voidEnvelopeparams->Reason = "void envelope sample";
$response = $api->VoidEnvelope($voidEnvelopeparams);
```

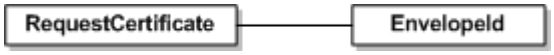
Post Processing Function Group

This section describes the methods that can be used to retrieve envelope documents from the DocuSign system. DocuSign maintains a separate certificate document with each envelope document that contains important origination and landmark events pertinent to the document. RequestDocumentsPDFs and RequestPDF methods are used to request documents. DocumentPDFs and EnvelopePDF are the responses for these methods.

RequestCertificate

RequestCertificate returns the signing certificate, which details the specific attributes of the participants and landmark events of the signing transaction, for an envelope.

Schema



Name	Schema Type	Description
EnvelopeID	DSXId	This is the envelope ID of the envelope, as returned by the CreateEnvelope/CreateAndSendEnvelope. Only documents within the specified envelope are returned.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/RequestDocumentPDFs"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestDocumentPDFs xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
    </RequestDocumentPDFs>
  </soap:Body>
</soap:Envelope>
```

RequestCertificateWithCertLanguage

RequestCertificateWithCertLanguage returns the signing certificate for an envelope in the specified language.

Schema

Name	Schema Type	Description
EnvelopeID	DSXId	This is the envelope ID of the envelope, as returned by the CreateEnvelope/CreateAndSendEnvelope.

Name	Schema Type	Description
CertLanguage	String	Sets the language for the Certificate of Completion in the response. The supported languages, with the language value shown in parenthesis, are: Chinese Simplified (zh_CN), Chinese Traditional (zh_TW), Dutch (nl), English US (en), French (fr), German (de), Italian (it), Japanese (ja), Korean (ko), Portuguese (pt), Portuguese (Brazil) (pt_BR), Russian (ru), Spanish (es).

Sample Request XML:

```
SOAPAction: "http://www.docusign.net/API/3.0/RequestCertificateWithCertLanguage"

<?xml version="1.1" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestCertificateWithCertLanguage xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
      <CertLanguage>string</CertLanguage>
    </RequestCertificateWithCertLanguage>
  </soap:Body>
</soap:Envelope>
```

RequestDocumentPDFs

Schema

Name	Schema Type	Description
EnvelopeID	DSXId	This is the envelope ID of the envelope, as returned by the CreateEnvelope/CreateAndSendEnvelope. Only documents within the specified envelope are returned.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/RequestDocumentPDFs"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestDocumentPDFs xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
    </RequestDocumentPDFs>
  </soap:Body>
</soap:Envelope>
```

This method returns a response with an array of DocumentPDF. See DocumentPDF for more information about the returned information.

Sample Code

RequestDocumentPDFs – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Call to request the PDFs with the envelope's ID
DocuSignWeb.DocumentPDFs pdf = _apiClient.RequestDocumentPDFs(status.EnvelopeID);

// Display number of PDFs returned
Console.WriteLine("Envelope has {0} pdfs", pdf.DocumentPDF.Length);
```

RequestDocumentPDFs – PHP

```
// Create and send an envelope as shown in linked code
$response = <create and send envelope>;
$status = $response->CreateAndSendEnvelopeResult;

// Request PDFs
$requestDocumentPDFsparams = new RequestDocumentPDFs();
$requestDocumentPDFsparams->EnvelopeID = $status->EnvelopeID;
$response = $api->RequestDocumentPDFs($requestDocumentPDFsparams);
```

RequestDocumentPDFsEx

Schema

Name	Schema Type	Description
EnvelopeID	DSXId	This is the envelope ID of the envelope, as returned by the CreateEnvelope/CreateAndSendEnvelope. Only documents within the specified envelope are returned.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/RequestDocumentPDFs"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestDocumentPDFsEx xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
    </RequestDocumentPDFsEx>
  </soap:Body>
</soap:Envelope>
```

This method returns a response with an array of DocumentPDF. See DocumentPDF for more information about the returned information.

Sample Code

RequestDocumentPDFsEx – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Call to request the PDFs with the envelope's ID
DocuSignWeb.DocumentPDFs pdf = _apiClient.RequestDocumentPDFsEx(status.EnvelopeID);

// Display number of PDFs returned
Console.WriteLine("Envelope has {0} pdfs", pdf.DocumentPDF.Length);
```

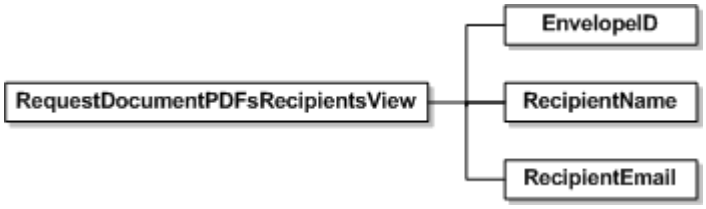
RequestDocumentPDFsEx – PHP

```
// Create and send an envelope as shown in linked code
$response = <create and send envelope>;
$status = $response->CreateAndSendEnvelopeResult;

// Request PDFs
$requestDocumentPDFsExparams = new RequestDocumentPDFsEx();
$requestDocumentPDFsExparams->EnvelopeID = $status->EnvelopeID;
$response = $api->RequestDocumentPDFsEx($requestDocumentPDFsExparams);
```

RequestDocumentPDFsRecipientsView

RequestDocumentPDFsRecipientsView requests the recipient's view of the document PDFs in an envelope.

Schema



Name	Schema Type	Description
EnvelopeId	EnvelopeId	The envelope ID for the document being requested.
RecipientName	String	The name of the recipient for the document being requested.
RecipientEmail	String	The email for the recipient for the document being requested.

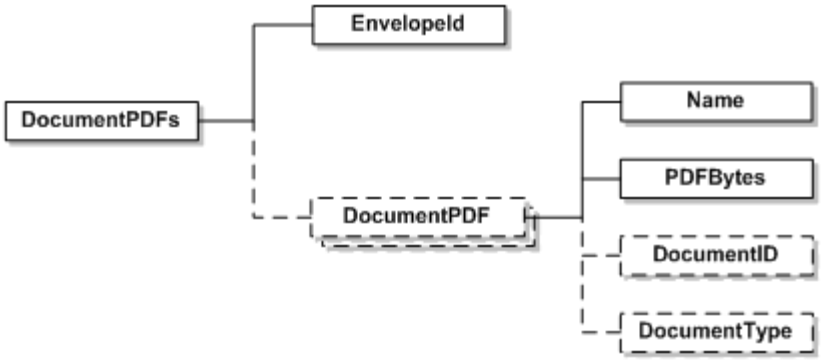
```
SOAPAction: "http://www.docusign.net/API/3.0/RequestDocumentPDFsRecipientsView"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
```

```
  <soap:Body>
    <RequestDocumentPDFsRecipientsView xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
      <RecipientName>string</RecipientName>
      <RecipientEmail>string</RecipientEmail>
    </RequestDocumentPDFsRecipientsView>
  </soap:Body>
</soap:Envelope>
```

This method returns a response with an array of DocumentPDF. See DocumentPDF for more information about the returned information.

DocumentPDF

Schema



Name	Schema Type	Description
EnvelopeID	DSXId	Returns EnvelopeID of the envelope.
DocumentPDF	DocumentPDF	This complex type contains: • Name - name of the document. • PDFBytes – the document byte stream. • DocumentID – the originating document ID. Only returned when calling RequestDocumentPDFsEx. • DocumentType – type of the document. Only returned when calling RequestDocumentPDFsEx.

RequestPDF

This method returns all of the documents combined into a single, contiguous PDF. If the watermark feature is enabled for your account and envelope signing is not complete, the watermark is displayed in the PDF.

Schema

Name	Schema Type	Description

Name	Schema Type	Description
EnvelopeID	DSXId	This is the envelope ID of the envelope, as returned by the CreateEnvelope/CreateAndSendEnvelope.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/RequestPDF"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestPDF xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
    </RequestPDF>
  </soap:Body>
</soap:Envelope
```

Sample Code

RequestPDF – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Call to request the PDF with the envelope ID
DocuSignWeb.EnvelopePDF pdf =
    _apiClient.RequestPDF(status.EnvelopeID);

// Display the size of the PDF
Console.WriteLine("Pdf has {0} bytes", pdf.PDFBytes.Length);
```

RequestPDF – PHP

```
// Create and send an envelope as shown in linked code
$response = <create and send envelope>;
$status = $response->CreateAndSendEnvelopeResult;

// Request PDFs
$requestPDFparams = new RequestPDF();
$requestPDFparams->EnvelopeID = $status->EnvelopeID;
$response = $api->RequestPDF($requestPDFparams);
```

RequestPDFNoWaterMark

If your account has the watermark feature enabled, this method returns all of the documents combined into a single, contiguous PDF without displaying the watermark.

Schema

Name	Schema Type	Description

Name	Schema Type	Description
EnvelopeID	DSXId	This is the envelope ID of the envelope, as returned by the CreateEnvelope/CreateAndSendEnvelope.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/RequestPDFNoWaterMark"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestPDFNoWaterMark xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
    </RequestPDFNoWaterMark>
  </soap:Body>
</soap:Envelope
```

Sample Code

RequestPDFNoWaterMark – C#

```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Call to request the PDF with the envelope ID
DocuSignWeb.EnvelopePDF pdf =
    _apiClient.RequestPDFNoWaterMark(status.EnvelopeID);

// Display the size of the PDF
Console.WriteLine("Pdf has {0} bytes", pdf.PDFBytes.Length);
```

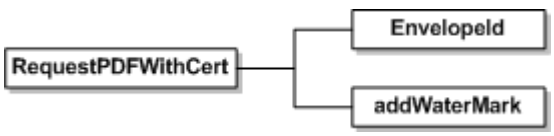
RequestPDFNoWaterMark – PHP

```
// Create and send envelope as shown in linked code
$result = $api->CreateAndSendEnvelope($createAndSendEnvelopeparams);
$status = $result->CreateAndSendEnvelopeResult;

// Send
$requestPDFNoWaterMarkparams = new RequestPDFNoWaterMark();
$requestPDFNoWaterMarkparams->EnvelopeID = $status->EnvelopeID;
$response = $api->RequestPDFNoWaterMark($requestPDFNoWaterMarkparams);
```

RequestPDFWithCert

This method returns all of the documents combined into a single, contiguous PDF. Additionally it returns the Signing Certificate PDF document, which details the specific attributes of the participants and landmark events of the signing transaction. It also returns the Electronic Record and Signature Disclosure, per account settings, associated with the envelope.

Schema



Name	Schema Type	Description
EnvelopeID	DSXId	This is the envelope ID of the envelope, as returned by the CreateEnvelope/CreateAndSendEnvelope.
addWaterMark	Boolean	If your account has the watermark feature enabled and when true, the watermark for the account is shown on documents for envelopes that are not completed.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/RequestPDFWithCert"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestPDFWithCert xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
      <addWaterMark>boolean</addWaterMark>
    </RequestPDFWithCert>
  </soap:Body>
</soap:Envelope
```

Sample Code

RequestPDFWithCert – C#

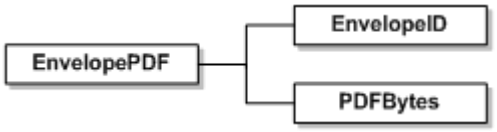
```
// Create and send envelope as shown in linked code
DocuSignWeb.EnvelopeStatus status = <create and send envelope>;

// Call to request the PDF with the envelope ID
// Here, we chose to suppress the watermark
DocuSignWeb.EnvelopePDF pdf =
    _apiClient.RequestPDFWithCert(status.EnvelopeID, false);

// Display the size of the PDF
Console.WriteLine("Pdf has {0} bytes", pdf.PDFBytes.Length);
```

RequestPDFWithCert – PHP

```
// Create and send envelope as shown in linked code
$result = $api->CreateAndSendEnvelope($createAndSendEnvelopeparams);
$status = $result->CreateAndSendEnvelopeResult;

// Send
$requestPDFWithCertparams = new RequestPDFWithCert();
$requestPDFWithCertparams->EnvelopeID = $status->EnvelopeID;
    $requestPDFWithCertparams->AddWaterMark = false;
    $response = $api->RequestPDFWithCert($requestPDFWithCertparams);
```

EnvelopePDF

Schema



Name	Schema Type	Description
EnvelopeID	DSXId	Returns EnvelopeID of the envelope.
PDFBytes	base64Binary	The document bytestream

Rules for using RequestDocumentPDFs, RequestDocumentPDFsEx, RequestPDF, RequestPDFNoWaterMark, RequestPDFWithCert, DocumentPDF and EnvelopePDF

- In the methods mentioned in this section, length of the EnvelopeID must not exceed 100 characters. Else, the XML validation fails and the processor throws a Validation error.
- In the request methods envelope with specified Id must exist. Else an exception “Envelope_Does_Not_Exist” is thrown.
- In the DocumentPDF method returns all the PDF documents in the envelope as base64binary with document names. But DocumentPDF element is optional. If the envelope with mentioned EnvelopeID is in the draft stage then Name or PDFBytes is not returned.
- DocumentPDFs also return certificate document with PDF documents in the envelope
- Only sender or recipient can retrieve the documents else exception, “User_Not_Envelope_Sender_Or_Recipient” will be thrown.
- You may have your account setup by DocuSign to include DocuSign PDF meta data in the returned PDF bytes.

  The envelope level PDF will contain:

  - Tracking information that contains the Envelope ID.
  - EnvelopeStatus XML.

  The document level PDF will contain:

  - Tracking information that contains the Envelope ID and originating Document ID.
  - EnvelopeStatus XML with the tab information removed that does not pertain to the document.
  - Document PDF form data if the document originated with TransformPDFFields enabled.

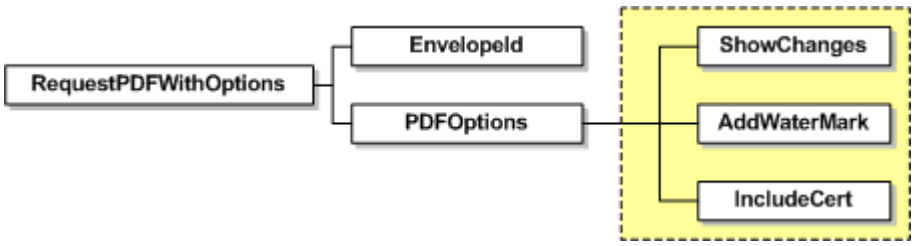
  The envelope summary document will contain:

  - Tracking information that contains the Envelope ID.
  - EnvelopeStatus XML with all tab information removed.

RequestPDFWithOptions

This method returns all of the documents combined into a single, contiguous PDF. It includes the ability to set display options for the PDFs.

Schema



Name	Schema Type	Description
EnvelopeId	DSXId	The envelope ID for the PDF document being requested.
PDFOptions		Optional display elements for the PDF: • ShowChanges – If the account has the Highlight Data Changes feature enabled and this option is set to true, any changed fields for the returned PDF are highlighted in yellow and optional signatures or initials outlined in red. • AddWaterMark – If the account has the watermark feature enabled and the envelope is not complete, the watermark for the account is added to the PDF document. When this is set to false, the watermark is removed from the PDF. • IncludeCert – When set to true, this option removes the envelope signing certificate from the download. • CertificateLanguage – Sets the language for the Certificate of Completion in the response. The supported languages, with the language value shown in parenthesis, are: Chinese Simplified (zh_CN), Chinese Traditional (zh_TW), Dutch (nl), English US (en), French (fr), German (de), Italian (it), Japanese (ja), Korean (ko), Portuguese (pt), Portuguese (Brazil) (pt_BR), Russian (ru), Spanish (es).

Sample Request XML

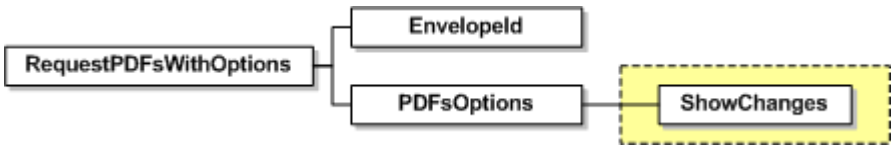
```
SOAPAction: "http://www.docusign.net/API/3.0/RequestPDFWithOptions"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestPDFWithOptions xmlns="http://www.docusign.net/API/3.0">
```

```
      <EnvelopeID>string</EnvelopeID>
      <PDFOptions>
        <ShowChanges>boolean</ShowChanges>
        <AddWaterMark>boolean</AddWaterMark>
        <IncludeCert>boolean</IncludeCert>
        <CertificateLanguage>string</CertificateLanguage>
      </PDFOptions>
    </RequestPDFWithOptions>
  </soap:Body>
</soap:Envelope>
```

RequestPDFsWithOptions

This method returns one document or all the documents in an envelope as an array of DocumentPDF (see DocumentPDF for more about the returned information). If the account has the Highlight Data Changes feature enabled and the ShowChanges option is set to true, any changes on the documents are highlighted.

Schema



Name	Schema Type	Description
EnvelopeID	DSXId	The envelope ID for the PDF documents being requested.
PDFsOptions		Optional display elements for the PDFs: • ShowChanges – When set to true, any changed fields for the returned PDF are highlighted in yellow and optional signatures or initials outlined in red. • DocumentID – The DocumentID of the document that should be returned by the call. • CertificateLanguage – Sets the language for the Certificate of Completion in the response. The supported languages, with the language value shown in parenthesis, are: Chinese Simplified (zh_CN), Chinese Traditional (zh_TW), Dutch (nl), English US (en), French (fr), German (de), Italian (it), Japanese (ja), Korean (ko), Portuguese (pt), Portuguese (Brazil) (pt_BR), Russian (ru), Spanish (es).

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/RequestPDFsWithOptions"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestPDFsWithOptions xmlns="http://www.docusign.net/API/3.0">
```

```
      <EnvelopeID>string</EnvelopeID>
      <PDFsOptions>
        <ShowChanges>boolean</ShowChanges>
        <DocumentID>positiveInteger</DocumentID>
        <CertificateLanguage>string</CertificateLanguage>
      </PDFsOptions>
    </RequestPDFsWithOptions>
  </soap:Body>
</soap:Envelope>
```

RequestPDFRecipientView

This method returns the view of the document a recipient would see if the recipient downloaded the document. This allows the sending account to get the recipient view for cases where the documents or tabs the recipient sees are limited, such as when an envelope is sent with document visibility enabled. It includes the ability to set display options for the PDFs.

Schema

Name	Schema Type	Description
EnvelopeID	DSXId	The envelope ID for the PDF document being requested.
RecipientName	String	The name of the recipient for the documents being requested. This is the same as the UserName for the envelope.
RecipientEmail	String	The email address of the recipient for the documents being requested. This is the same as Email for the envelope.
PDFOptions		Optional display elements for the PDF: • ShowChanges – If the account has the Highlight Data Changes feature enabled and this option is set to true, any changed fields for the returned PDF are highlighted in yellow and optional signatures or initials outlined in red. • AddWaterMark – If the account has the watermark feature enabled and the envelope is not complete, the watermark for the account is added to the PDF document. When this is set to false, the watermark is removed from the PDF. • IncludeCert – When set to true, this option removes the envelope signing certificate from the download.

Sample Request XML

```
POST /api/3.0/dsapi.asmx HTTP/1.1
Host: demo.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/3.0/RequestPDFRecipientsView"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestPDFRecipientsView xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
      <RecipientName>string</RecipientName>
      <RecipientEmail>string</RecipientEmail>
      <PDFOptions>
        <ShowChanges>boolean</ShowChanges>
        <AddWaterMark>boolean</AddWaterMark>
        <IncludeCert>boolean</IncludeCert>
      </PDFOptions>
    </RequestPDFRecipientsView>
  </soap:Body>
</soap:Envelope>
```

The response returns the recipient's view of the PDF documents in the envelope, with selected display options.

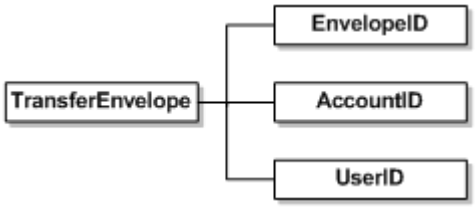
Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestPDFRecipientsViewResponse xmlns="http://www.docusign.net/API/3.0">
      <RequestPDFRecipientsViewResult>
        <EnvelopeID>string</EnvelopeID>
        <PDFBytes>base64Binary</PDFBytes>
      </RequestPDFRecipientsViewResult>
    </RequestPDFRecipientsViewResponse>
  </soap:Body>
</soap:Envelope>
```

TransferEnvelope

The TransferEnvelope method effectively transfers all documents in the specified envelope to the new owner.

Schema



Name	Schema Type	Description
EnvelopeID	DSXId	This is the envelope ID of the envelope, as specified by the request.
AccountID	DSXId	This specifies the account ID of the person to whom ownership is transferred.
UserID	DSXId	The user ID of the person to whom the envelope ownership is being transferred.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/TransferEnvelope"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <TransferEnvelope xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeID>string</EnvelopeID>
      <AccountID>string</AccountID>
      <UserID>string</UserID>
    </TransferEnvelope>
  </soap:Body>
</soap:Envelope>
```

TransferEnvelopeStatus

TransferEnvelopeStatus is the response method for TransferEnvelope.

Schema

Name	Schema Type	Description
TransferEnvelopeSuccess	Boolean	This element is true if the transfer succeeded.

Rules for using TransferEnvelope

- The length of any of the EnvelopeID, AccountID, and UserID must not exceed 100 characters. Else, the XML validation fails and the processor throws a Validation error.
- An envelope having the specified Envelop*eID* must exist in the system. Else, an exception with the error message “Envelope_Does_Not_Exist” is thrown.
- AccountID must exist in the system; else exception “Account_Does_Not_Exist_In_System” will be thrown.
- UserID must exist in the system; else exception “User_Does_Not_Exist_In_System” will be thrown.
- Transfer of envelope will be allowed only if the envelope status is complete. Else “Envelope_Is_Incomplete” will be thrown.

- Only sender would be able to transfer the ownership to someone else. Else error "User_Not_Envelope_Sender" will be thrown
- User cannot transfer the ownership of the envelope to self. If a user tries to transfer envelope to self "Envelope_Transferee_Already_Owns_Envelope" exception will be thrown.
- If Authoritative Copy status is set then envelope cannot be transferred. It tried to transfer the resulting error will be "Envelope_Cannot_Transfer_Invalid_ACStatus".

Sample Code

TransferEnvelope – C#

```
// Request the envelope specified to be transferred to the account specified
DocuSignWeb.TransferEnvelopeStatus status = _apiClient.TransferEnvelope("envelope ID",
    "account ID as GUID", "user ID as GUID");

// Confirm that the transfer succeeded
Console.WriteLine("Transfer succeeded? {0}", status.TransferEnvelopeSuccess);
```

TransferEnvelope – PHP

```
// Request the envelope specified to be transferred to the account specified
$transferEnvelopeparams = new TransferEnvelope();
// TODO: replace string with account ID GUID that you will transfer the envelope to
$transferEnvelopeparams->AccountID = "someone else's account ID GUID here";
// TODO: replace string with envelope ID GUID that will be transferred
$transferEnvelopeparams->EnvelopeID = "your envelopeID GUID here";
// TODO: replace string with user ID GUID that you will transfer the envelope to
$transferEnvelopeparams->UserID = "someone else's user ID as GUID here";
$response = $api->TransferEnvelope($transferEnvelopeparams);
```

ExportAuthoritativeCopy

This section presents the principles and methods involved in implementing a DocuSign Connect API integration using the ExportAuthoritativeCopy and AcknowledgeAuthoritativeCopyExport API methods. The ExportAuthoritativeCopy API method enables API users to extract Authoritative Copy envelopes from DocuSign. The AcknowledgeAuthoritativeCopyExport API method is used to indicate success of the extract call and to obtain the key to unlock the envelope's documents.

Schema

Name	Schema Type	Description
EnvelopeID	DSXId	Envelope ID of the envelope which is to be exported.

Sample Request XML:

```
SOAPAction: "http://www.docusign.net/API/3.0/ExportAuthoritativeCopy"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
```

```
    <ExportAuthoritativeCopy xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeId>string</EnvelopeId>
    </ExportAuthoritativeCopy>
  </soap:Body>
</soap:Envelope>
```

AuthoritativeCopyExportDocuments

Schema

Name	Schema Type	Description
EnvelopeId	DSXId	Envelope ID of the envelope which is to be exported.
TransactionId	DSXId	Transaction ID for this export process.
Count	Integer	Number of documents in the DocumentPDF array.
DocumentPDF	DocumentPDF	Array of encrypted document PDFs in the envelope.

AcknowledgeAuthoritativeCopyExport

Schema

Name	Schema Type	Description
EnvelopeId	DSXId	Envelope ID of the envelope which is to be exported.
TransactionId	DSXId	Transaction ID for this export process. Obtained from the ExportAuthoritativeCopy API call.
checkSumHash	base64Binary	Total hash of all the documents returned in the ExportAuthoritativeCopy API call.

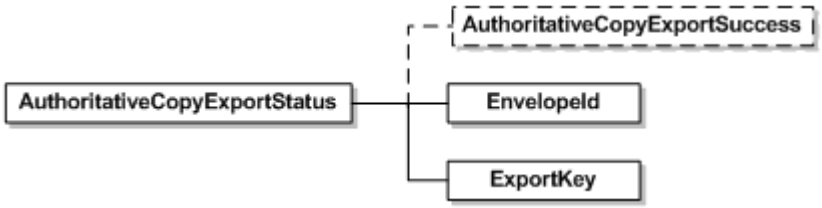
Sample Request XML:

```
SOAPAction: "http://www.docusign.net/API/3.0/AcknowledgeAuthoritativeCopyExport"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <AcknowledgeAuthoritativeCopyExport xmlns="http://www.docusign.net/API/3.0">
      <EnvelopeId>string</EnvelopeId>
      <TransactionId>string</TransactionId>
      <checkSumHash>base64Binary</checkSumHash>
    </AcknowledgeAuthoritativeCopyExport>
  </soap:Body>
</soap:Envelope>
```

AuthoritativeCopyExportStatus

Schema



Name	Schema Type	Description
AuthoritativeCopyExportSuccess	Boolean	Return true if the envelope export is successful
EnvelopeId	DSXId	Envelope ID of the envelope which is to be exported.
ExportKey	DSXId	Key used to decrypt the Rijndael encrypted documents returned in the ExportAuthoritativeCopy API call. Encryption mode is CBC. Encryption padding is PKCS7.

Rules for exporting Authoritative Copy envelopes

API user specific rules

- The envelope ID passed to the API must be valid.
- The envelope ID passed to the API must be in the Authoritative Copy status in DocuSign.
- The user and account must be configured properly by DocuSign in order to enable this API.

Rules for Exceptions thrown by the ExportAuthoritativeCopy API

- Envelope_Does_Not_Exist – invalid envelope ID passed.
- User_Lacks_Permissions - The user making the API calls must have permissions to export Authoritative Copy Documents.
- Envelope_AC_Export_Completed – the envelope being requested has already been exported from DocuSign.

- Envelope_AC_Export_Not_AC_Copy – the envelope being requested is not an Authoritative Copy.

Rules for Exceptions thrown by the AcknowledgeAuthoritativeCopyExport API

- Envelope_Does_Not_Exist – invalid envelope ID passed.
- User_Lacks_Permissions - The user making the API calls must have permissions to export Authoritative Copy Documents.
- Envelope_AC_Export_Invalid_Status – envelope is not in the valid export status.
- Envelope_AC_Export_Invalid_TransID – transaction ID passed is invalid.

Rules for return value of the AcknowledgeAuthoritativeCopyExport API

- AuthoritativeCopyExportSuccess – True if export was successfully completed.
- ExportKey – key to decrypt the Rijndael encrypted documents. This key is only provided if AuthoritativeCopyExportSuccess is true. Encryption mode is CBC. Encryption padding is PKCS7.

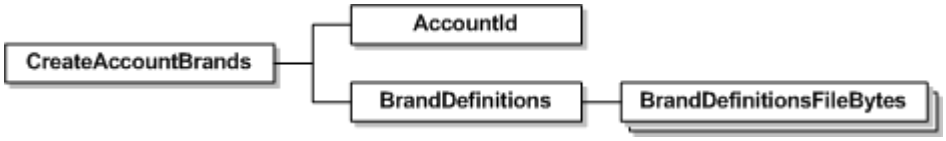
Administrative Function Group

This section describes administrative methods.

CreateAccountBrands

CreateAccountBrands is used to upload one or more brand profile files to the account. The Account Branding feature must be enabled for the account to use this.

Schema



Name	Schema Type	Description
AccountID	DSXId	The AccountId for the account to which the brand profiles are added.
BrandDefinitions	BrandDefinitionFileBytes	The base64Binary bytes of the brand profile xml file being added.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/CreateAccountBrands"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CreateAccountBrands xmlns="http://www.docusign.net/API/3.0">
      <AccountId>string</AccountId>
      <BrandDefinitions>
        <BrandDefinitionsFileBytes>base64Binary</BrandDefinitionsFileBytes>
      </BrandDefinitions>
    </CreateAccountBrands>
  </soap:Body>
</soap:Envelope>
```

DeleteAccountBrands

DeleteAccountBrands is used to delete one or more brand profiles from an account. The Account Branding feature must be enabled for the account to use this.

Schema

Name	Schema Type	Description
AccountID	DSXId	The AccountId for the account from which the brand profiles are being deleted.
BrandDeleteRequest	BrandRequestItems	An array of BrandRequestItem that each contain the BrandId for the brand profile to be deleted.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/DeleteAccountBrands"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <DeleteAccountBrands xmlns="http://www.docusign.net/API/3.0">
      <AccountId>string</AccountId>
      <BrandDeleteRequest>
        <BrandRequestItems>
          <BrandRequestItem>
            <BrandId>string</BrandId>
          </BrandRequestItem>
          <BrandRequestItem>
            <BrandId>string</BrandId>
          </BrandRequestItem>
        </BrandRequestItems>
      </BrandDeleteRequest>
    </DeleteAccountBrands>
  </soap:Body>
</soap:Envelope>
```

GetAccountBrands

GetAccountBrands is used to retrieve a list of brand profiles associated with the account and the default brand profile. The Account Branding feature must be enabled for the account to use this.

Schema

Name	Schema Type	Description
SenderAccountID	DSXId	The AccountId for the account.

Sample Request XML

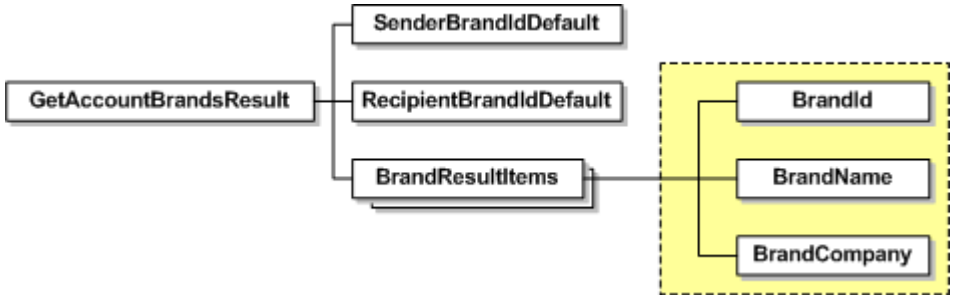
```
SOAPAction: "http://www.docusign.net/API/3.0/GetAccountBrands"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetAccountBrands xmlns="http://www.docusign.net/API/3.0">
      <SenderAccountId>string</SenderAccountId>
```

```
    </GetAccountBrands>
  </soap:Body>
</soap:Envelope>
```

GetAccountBrandsResult

This is the response for GetAccountBrands and contains the default brand profile and list of brand profiles associated with the account.

Schema



Name	Schema Type	Description
RecipientBrandIdDefault	String	The BrandId of the default signing brand profile.
SenderBrandIdDefault	String	The BrandId of the default sending brand profile.
BrandResultItems	BrandResultItem	A complex element with the BrandId, BrandName and BrandCompany information for all the brand profiles associated with the account.

GetAccountMembershipFeatureList

GetAccountMembershipFeaturesList is used to retrieve the features available to the membership.

Schema

Name	Schema Type	Description
AccountID	DSXId	Account Id of the user whose membership permissions for optional features are requested.

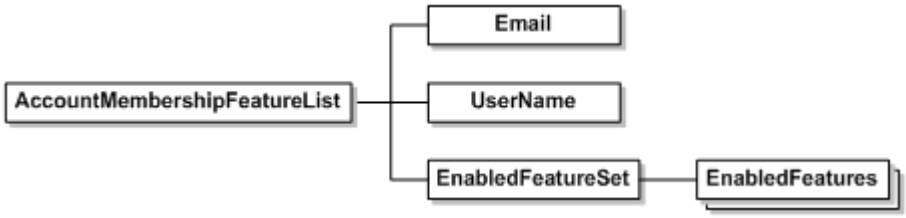
Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/GetAccountMembershipFeaturesList"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
```

```
  <soap:Body>
    <GetAccountMembershipFeaturesList xmlns="http://www.docusign.net/API/3.0">
      <AccountId>string</AccountId>
    </GetAccountMembershipFeaturesList>
  </soap:Body>
</soap:Envelope>
```

AccountMembershipFeatureList

Schema



Name	Schema Type	Description
Email	Email	Email of the user whose membership feature list is requested.
UserName	UserName	UserName of the user whose membership feature list is requested.
EnabledFeaturesSet	String	Returns the set of DocuSign Professional features in membership.

Usage rules for GetAccountMembershipFeatureList and AccountMembershipFeatureList

- The length of AccountId, Email and UserName must not exceed 100 characters.
- The EnabledFeatures would be unbounded. Values for this will be DocuSignProfessional, eOriginalVaultResponse, SequentialSigningAPI, SequentialSigningUI, TransactionPoint, CanSetDocumentSignerVisibility and AccountForcesSignerVisibility.

Sample Code

GetAccountMembershipFeaturesList – C#

```
// This is simple -- just input your account ID
DocuSignWeb.AccountMembershipFeaturesList list =
    _apiClient.GetAccountMembershipFeaturesList(_accountId);
```

GetAccountMembershipFeaturesList – PHP

```
$getAccountMembershipFeaturesListparams = new GetAccountMembershipFeaturesList();
$getAccountMembershipFeaturesListparams->AccountID = $AccountID;
$response = $api-
>GetAccountMembershipFeaturesList($getAccountMembershipFeaturesListparams);
```

GetAccountSettingsList

GetAccountSettingsList is used to retrieve the settings available to the account.

Schema

Name	Schema Type	Description
AccountID	DSXId	AccountId of the account settings to be returned.

Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/GetAccountSettingsList "

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetAccountSettingsList xmlns="http://www.docusign.net/API/3.0">
      <AccountId>string</AccountId>
    </ GetAccountSettingsList >
  </soap:Body>
</soap:Envelope>
```

AccountSettingsList

Schema

Name	Schema Type	Description
AccountSetting	AccountSetting	Name: string: the name of the account setting. Value: string: the value of the account setting. Type: string: the type of the account setting (ie. Integer, Boolean, String)

Usage rules for GetAccountSettingsList and AccountSettingsList

- The length of AccountId, must not exceed 100 characters.
- The AccountSetting is unbounded. Current “Name” values are: UserOverrideEnabled, ReminderEnabled, ReminderFrequency, ReminderDelay, ExpireEnabled, ExpireAfter, and ExpireWarnBuffer.

Sample Code

GetAccountSettingsList – C#

```
// This is simple -- just input your account ID
DocuSignWeb.AccountSetting[] settings =
    _apiClient.GetAccountSettingsList(_accountId);

// Display some information about the settings
Console.WriteLine("We have {0} account settings:", settings.Length);

foreach (DocuSignWeb.AccountSetting setting in settings)
{
    Console.WriteLine("\t{0}: {1}", setting.Name, setting.Value);
}
```

GetAccountSettingsList – PHP

```
$getAccountSettingsListparams = new GetAccountSettingsList();
$getAccountSettingsListparams->AccountID = $AccountID;
$response = $api->GetAccountSettingsList($getAccountSettingsListparams);
```

GetAddressBookItems

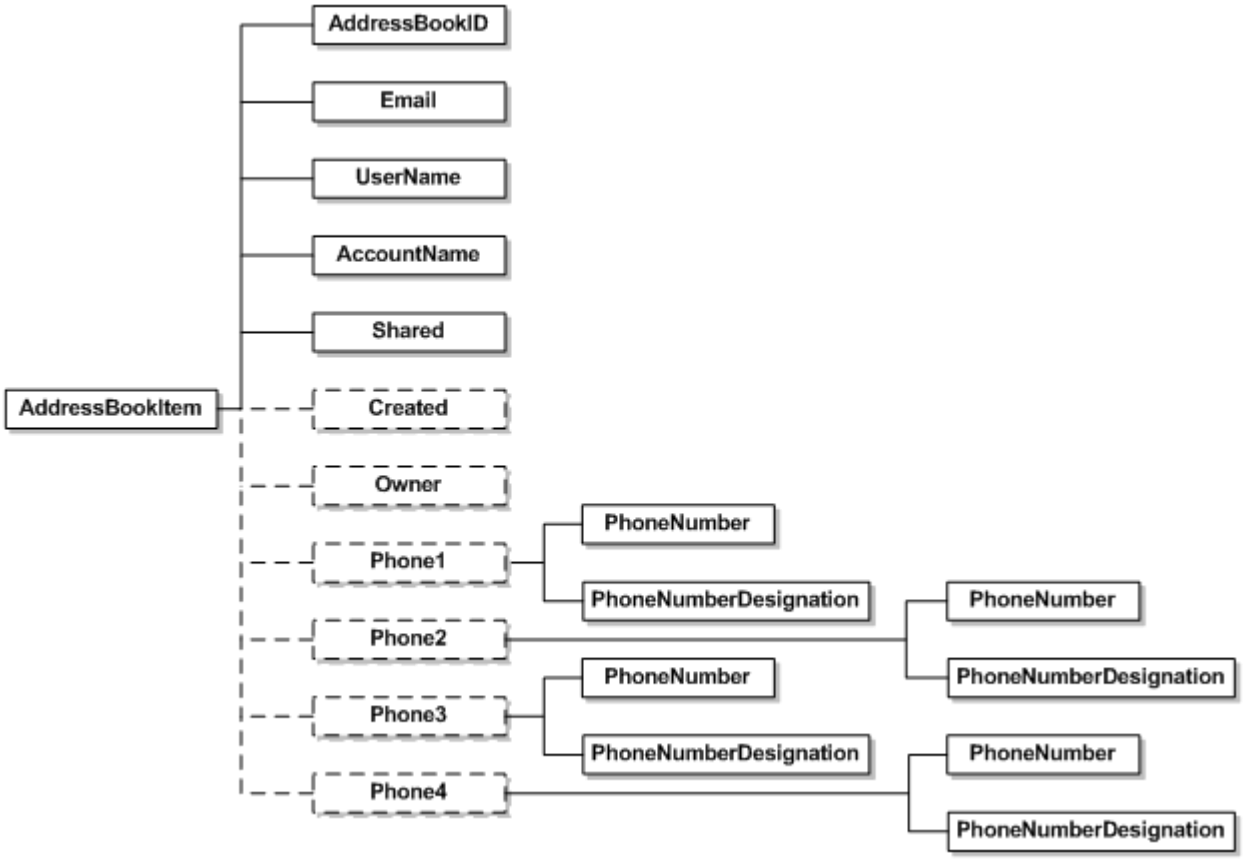
These methods can be used to manage your server side address book in the DocuSign system. You will be able to retrieve address book entries.

Schema

Name	Schema Type	Description
AccountID	DSXId	The DocuSign account ID.

Retrieves your address book and account shared address book. This method returns an array of AddressBookItem objects.

AddressBookItem



Name	Schema Type	Description
AddressBookID	DSXId	Unique identifier of the address book item. This can be a maximum of 100 characters.
Email	String	Email associated with the address book item. This can be a maximum of 100 characters.
UserName	String	Name associated with the address book item. This can be a maximum of 100 characters.
AccountName	String	Account name associated with the address book item. This can be a maximum of 100 characters.
Shared	Boolean	True if the address book item is shared with the API user's account.
Created	DateTime	Date the address book item was created.
Owner	Boolean	True if the API user account is the owner of the address book item.
Phone1	AddressBookPhone	PhoneNumber: string: phone number, maximum length is 20. PhoneNumberDesignation: value: one of the following: Home, Mobile, Work, Fax, Other.

Name	Schema Type	Description
Phone2	AddressBookPhone	PhoneNumber: string: phone number, maximum length is 20. PhoneNumberDesignation: value: one of the following: Home, Mobile, Work, Fax, Other.
Phone3	AddressBookPhone	PhoneNumber: string: phone number, maximum length is 20. PhoneNumberDesignation: value: one of the following: Home, Mobile, Work, Fax, Other.
Phone4	AddressBookPhone	PhoneNumber: string: phone number, maximum length is 20. PhoneNumberDesignation: value: one of the following: Home, Mobile, Work, Fax, Other.

Rules and exceptions for GetAddressBookItems

- Account_Lacks_Permissions – account does not have permissions to use this API.
- User_Lacks_Permissions – user does not have permissions to use this API.
- Address book items that are passed to this API that the user does not own will not be removed.

Sample Code

GetAddressBookItems – C#

```
// This is simple -- just input your account ID
DocuSignWeb.AddressBookItem[] items =
    _apiClient.GetAddressBookItems(_accountId);

// Display information about the address book items
Console.WriteLine("There there are {0} items in the address book:", items.Length);

foreach (DocuSignWeb.AddressBookItem item in items)
{
    Console.WriteLine("\t{0}: {1}", item.UserName, item.Email);
}
```

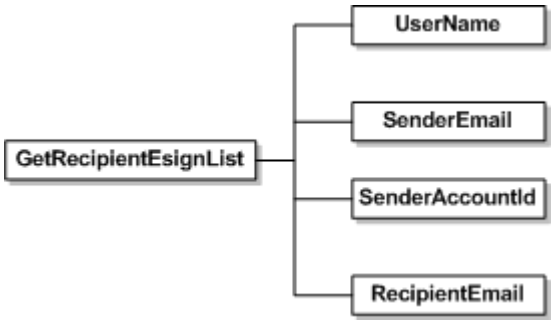
GetAddressBookItems – PHP

```
$getAddressBookItemsparams = new GetAddressBookItems();
$getAddressBookItemsparams->AccountID = $AccountID;
$getAddressBookItemsResponse = $api->GetAddressBookItems($getAddressBookItemsparams);
```

GetRecipientEsignList

Senders can use this method to determine if an Esign agreement pre-exists between the sender and recipient.

Schema



Name	Schema Type	Description
UserName	UserName	Sender's name.
SenderEmail	Email	Sender's email address.
SenderAccountId	DSXId	Sender's account Id
RecipientEmail	Email	Recipient's email address.

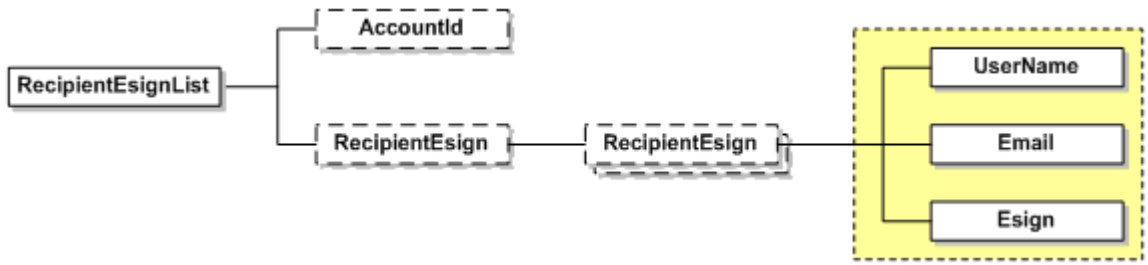
Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/GetRecipientEsignList"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetRecipientEsignList xmlns="http://www.docusign.net/API/3.0">
      <UserName>string</UserName>
      <SenderEmail>string</SenderEmail>
      <SenderAccountId>string</SenderAccountId>
      <RecipientEmail>string</RecipientEmail>
    </GetRecipientEsignList>
  </soap:Body>
</soap:Envelope>
```

RecipientEsignList

This element returns a list of RecipientEsignRecords for a designated sender and recipient combination. For each record, the return Boolean value indicates whether or not an Esign agreement exists between the sender and that particular recipient.

Schema



Name	Schema Type	Description
AccountId	DSXId	The account holding the ESign agreement.
UserName	UserName	The recipient's name.
Email	Email	The recipient's email.
ESign	boolean	If true, an Esign agreement exists between the sender and the recipient

Rules for using GetRecipientEsignList and RecipientEsignList

- The UserName should be valid.
- The SenderEmail should be valid; else "Invalid_Email_Address_For_Sender" execution will be thrown.
- The length of UserName, SenderEmail, SenderAccountId and RecipientEmail must not exceed 100 characters.
- User defined by UserName and SenderEmail pair belong to SenderAccountId specified, else "User_Does_Not_Belong_To_Specified_Account" exception is thrown.

Sample Code

GetRecipientEsignList – C#

```
// Enter account holder's username, email and accountID,
// and the email of the user that you wish to determine if
// an esign agreement exists
DocuSignWeb.RecipientEsignList recipients =
    _apiClient.GetRecipientEsignList(_userName, _email, _accountId, "<email>");

// Display the number of people in the list
Console.WriteLine("There are {0} recipients", recipients.RecipientEsign.Length);

// Examine each record -- does the esign agreement exist?
foreach (DocuSignWeb.RecipientEsign esign in recipients.RecipientEsign)
{
    Console.WriteLine("\t{0}: {1}", esign.Email, esign.Esign);
}
```

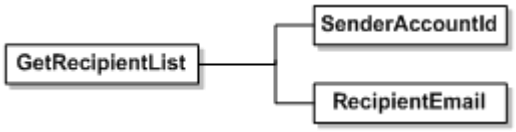
GetRecipientEsignList – PHP

```
// Enter account holder's username, email and accountID, and the email of
// the user that you wish to determine if an esign agreement exists
$getRecipientEsignListparams = new GetRecipientEsignList();
$getRecipientEsignListparams->SenderAccountId = $AccountID;
$getRecipientEsignListparams->SenderEmail = $UserID;
$getRecipientEsignListparams->UserName = $UserName;
$getRecipientEsignListparams->RecipientEmail = $Recipient1Email;
$response = $api->GetRecipientEsignList($getRecipientEsignListparams);
```

GetRecipientList

Senders can use this method to determine which recipients are available at the given email address.

Schema



Name	Schema Type	Description
SenderAccountId	DSXId	Sender's account Id
RecipientEmail	Email	Recipient's email address.

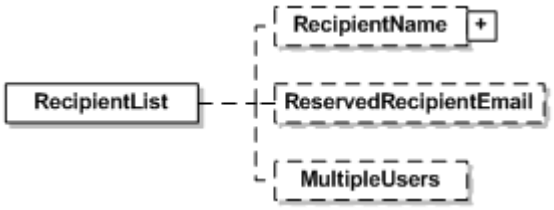
Sample Request XML

```
SOAPAction: "http://www.docusign.net/API/3.0/GetRecipientList"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetRecipientList xmlns="http://www.docusign.net/API/3.0">
      <SenderAccountId>string</SenderAccountId>
      <RecipientEmail>string</RecipientEmail>
    </GetRecipientList>
  </soap:Body>
</soap:Envelope>
```

RecipientList

This element returns a list of RecipientRecords for a designated recipient email. For each record, the return Boolean value indicates whether or not an Esign agreement exists between the sender and that particular recipient.

Schema



Name	Schema Type	Description
RecipientName	LongString	List of recipient Signature Names available to sign at this email.
ReservedRecipientEmail	Boolean	Whether or not the user has reserved this email to disallow new recipient names at this email address.
MultipleUsers	Boolean	If true, more than one physical person is using this email address to sign with.

Rules for using GetRecipientList and RecipientList

- The length of SenderAccountId and RecipientEmail must not exceed 100 characters.

- If no active signatures are available at the given email address, then no records will be returned.

Sample Code

GetRecipientList – C#

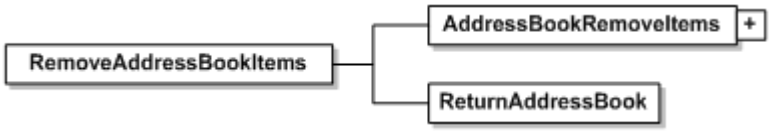
```
// Enter the account holder's ID and the target email
// to find all recipients at the target email address
DocuSignWeb.RecipientList recipients =
    _apiClient.GetRecipientList(_accountId, "<email>");

// Display the number of recipients
Console.WriteLine("There are {0} recipients", recipients.RecipientName.Length);

// Examine each record -- does the esign agreement exist?
foreach(String recipient in recipients.RecipientName)
{
    Console.WriteLine("\t{0}", recipient);
}
```

GetRecipientList – PHP

```
// Enter the account holder's ID and the target email to find all recipients
// at the target email address
$getRecipientListparams = new GetRecipientList();
$getRecipientListparams->SenderAccountId = $AccountID;
$getRecipientListparams->RecipientEmail = $Recipient1Email;
$response = $api->GetRecipientList($getRecipientListparams);
```

RemoveAddressBookItems

Schema



Name	Schema Type	Description
AddressBookRemoveItems	AddressBookRemoveItem Array	Array of address book item IDs to remove.
ReturnAddressBook	Boolean	If true, return the address book item list in the UpdateAddressBookResult object.

Removes all the specified items passed from your address book. This method returns an UpdateAddressBookResult object described earlier.

AddressBookRemoveItem

Schema

Name	Schema Type	Description
AddressBookID	DSXId	Address book ID to remove.

Rules and exceptions for RemoveAddressBookItems

- Account_Lacks_Permissions – account does not have permissions to use this API.
- User_Lacks_Permissions – user does not have permissions to use this API.
- Address book items that are passed to this API that the user does not own will not be removed.
- Invalid address book IDs will be ignored.

Sample Code

RemoveAddressBookItems – C#

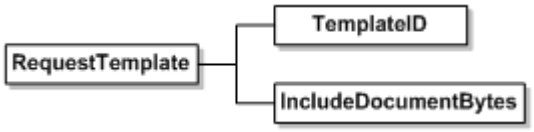
```
// Grab all the address book items from the specified account
DocuSignWeb.AddressBookItem[] items = _apiClient.GetAddressBookItems(_accountId);

if(items.Length >= 1)
{
    // Use the address book ID of an item to remove it
    DocuSignWeb.AddressBookRemoveItem toRemove = new DocuSignWeb.AddressBookRemoveItem();
    toRemove.AddressBookID = items[0].AddressBookID;

    // Construct an array with all the items you want to remove
    DocuSignWeb.AddressBookRemoveItem[] remove = { toRemove };
    DocuSignWeb.UpdateAddressBookResult result =
_apiClient.RemoveAddressBookItems(remove,
        false);

    // Confirm that the removal succeeded
    Console.WriteLine("Removed item {0} succeeded? {1}", toRemove.AddressBookID,
        result.Success);
}
```

RemoveAddressBookItems – PHP

```
// Grab all the address book items from the specified account
$getAddressBookItemsparams = new GetAddressBookItems();
$getAddressBookItemsparams->AccountID = $AccountID;
$addBookItems = $api->GetAddressBookItems($getAddressBookItemsparams)-
>GetAddressBookItemsResult->AddressBookItem;

if (count($addBookItems) >= 1) {
    $addToRemove = new AddressBookRemoveItem();
    // Build an array of the first address book item to remove
    $addToRemove->AddressBookID = $addBookItems[0]->AddressBookID;
    $removeAddressBookItemsparams = new RemoveAddressBookItems();
    $removeAddressBookItemsparams->AddressBookRemoveItems = array($addToRemove);
    $removeAddressBookItemsparams->ReturnAddressBook = false;
    $response = $api->RemoveAddressBookItems($removeAddressBookItemsparams);
}
else {
    $response = "No address book items to remove.";
}
```

RequestTemplate

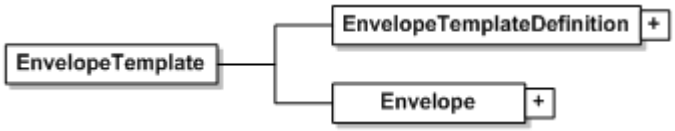
Schema



Name	Schema Type	Description
TemplateId	DSXId	Template ID to return.
IncludeDocumentBytes	Boolean	If true, include the document bytes of the template within the Envelope object returned.

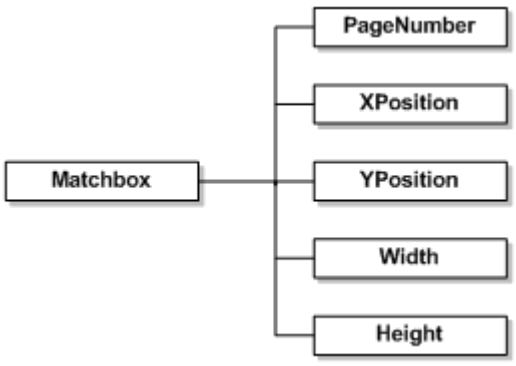
Retrieves a specific template from the server. This method returns EnvelopeTemplate upon execution completion.

EnvelopeTemplate



Name	Schema Type	Description
EnvelopeTemplateDefinition	EnvelopeTemplateDefinition	Defines the attributes of a template. See the EnvelopeTemplateDefinition section below for more information.
Envelope	Envelope	Template is contained within an Envelope object defined earlier in this document.

MatchBox

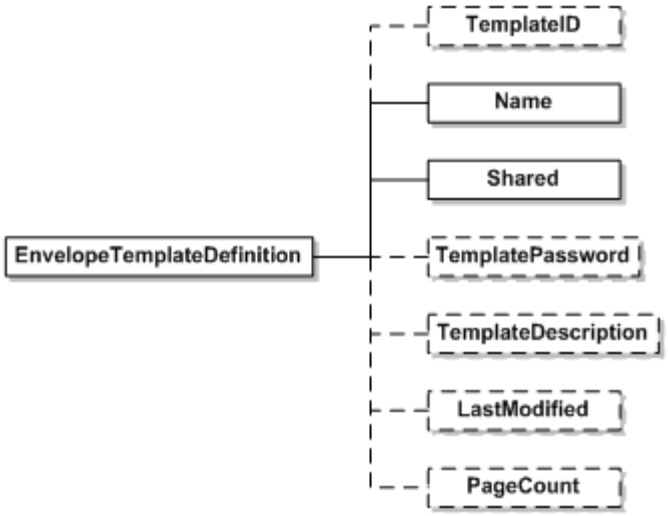


Name	Schema Type	Description
PageNumber	Integer	Page to put the matchbox on.
XPosition	Integer	X position of the matchbox.
YPosition	Integer	Y position of the matchbox.
Width	Integer	Width of the matchbox.

Name	Schema Type	Description
Height	Integer	Height of the matchbox.

Matchboxes can be supplied with the document structure within the Envelope. They are used in the matching process when documents are uploaded for creating templates.

EnvelopeTemplateDefinition



Name	Schema Type	Description
TemplateID	DSXId	Unique identifier of the template.
Name	String	Name of the template.
Shared	Boolean	If true, the template is shared with the Everyone group in the account. If false, the template is only shared with the Administrator group.
TemplatePassword	String	Password if the template is locked. NOTE: This will only be returned if the requesting user is the creator of the template.
TemplateDescription	String	Description of the template.
LastModified	DateTime	Last time the template was modified.
PageCount	Integer	Number of pages in document of the template.

Rules and exceptions for RequestTemplate

- Account_Lacks_Permissions – account does not have permissions to use this API.
- User_Lacks_Permissions – user does not have permissions to use this API.
- Template_Unable_To_Load – template ID provided was invalid.

Sample Code

RequestTemplate – C#

```
// Request all the templates as shown in linked code
DocuSignWeb.EnvelopeTemplateDefinition[] response = <templates request>;

// Request information about each template definition
foreach (DocuSignWeb.EnvelopeTemplateDefinition def in response)
{
    Console.WriteLine("Retrieved template definition with ID {0}", def.TemplateID);

    // Request the template specified by the template ID
    DocuSignWeb.EnvelopeTemplate template = _apiClient.RequestTemplate(def.TemplateID,
        false);

    // Display information about the returned template
    Console.WriteLine("\tRetrieved template with name \"{0}\" and with {1} pages",
        template.EnvelopeTemplateDefinition.Name,
        template.EnvelopeTemplateDefinition.PageCount);
}
```

RequestTemplate – PHP

```
// Request all the templates this account has on file
$requestTemplatesparams = new RequestTemplates();
$requestTemplatesparams->AccountID = $AccountID;
$requestTemplatesparams->IncludeAdvancedTemplates = false;
$rtsResponses = $api->RequestTemplates($requestTemplatesparams)->RequestTemplatesResult-
>EnvelopeTemplateDefinition;

// Request info for each template and put into array of $responses
$requestTemplateparams = new RequestTemplate();
$requestTemplateparams->IncludeDocumentBytes = false;
$responses = array();
foreach ($rtsResponses as $envTemplateDef) {
    $requestTemplateparams->TemplateID = $envTemplateDef->TemplateID;
    $response = $api->RequestTemplate($requestTemplateparams);
    array_push($responses, $response);
}
```

RequestTemplateWithDocumentFields

The RequestTemplateWithDocumentFields call is similar to the RequestTemplate call. It returns the requested template containing all the template data, including the document custom DocumentFields.

Schema

Name	Schema Type	Description
TemplateIDs	DSXId	Template ID being requested
IncludeDocumentBytes	Boolean	If true, include the document bytes of the template within the Envelope object returned.

Sample Request XML

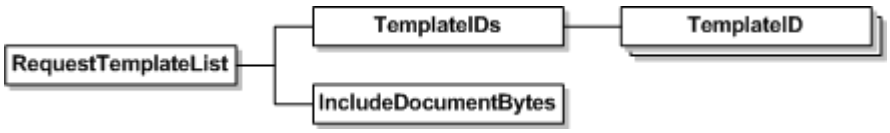
```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/AddMembersToAccount"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestTemplateWithDocumentFields xmlns="http://www.docusign.net/API/3.0">
      <TemplateID>string</TemplateID>
      <IncludeDocumentBytes>boolean</IncludeDocumentBytes>
    </RequestTemplateWithDocumentFields>
  </soap:Body>
</soap:Envelope>
```

The response returns the requested Template definition, including the document custom DocumentFields.

RequestTemplateList

This method requests a list of templates.

Schema



Name	Schema Type	Description
TemplateIDs	ArraryofTemplateID	An array of Template IDs for the templates being requested.
IncludeDocumentBytes	Boolean	If true, the document bytes of the requested templates are included in the response.

The method returns a list of EnvelopeTemplates for the requested templates.

RequestTemplateListWithDocumentFields

The RequestTemplateListWithDocumentFields call is similar to the RequestTemplateList call. It returns the requested templates containing all the template data, including the document custom DocumentFields.

Schema

Name	Schema Type	Description
TemplateIDs	ArraryofTemplateID	An array of Template IDs for the templates being requested.
IncludeDocumentBytes	Boolean	If true, the document bytes for the requested template documents are included in the response.

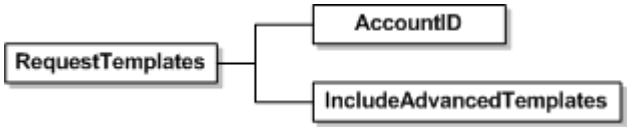
Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/AddMembersToAccount"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RequestTemplateListWithDocumentFields xmlns="http://www.docusign.net/API/3.0">
      <TemplateIDs>
        <string>string</string>
        <string>string</string>
      </TemplateIDs>
      <IncludeDocumentBytes>boolean</IncludeDocumentBytes>
    </RequestTemplateListWithDocumentFields>
  </soap:Body>
</soap:Envelope>
```

The method returns a list of Envelope Templates for the requested templates, including the document custom DocumentFields.

RequestTemplates

Schema



Name	Schema Type	Description
AccountId	DSXId	Account to request the templates for. If left blank the API user's default account will be used.
IncludeAdvancedTemplates	Boolean	If true, this will return templates with advanced features, (advanced features are those not supported by DSPro). If false, only templates without advanced features are returned.

This method returns a list of server side templates available for this account. The list format is in EnvelopeTemplateDefinition section above.

Rules and exceptions for RequestTemplates

- Account_Lacks_Permissions – account does not have permissions to use this API.
- User_Lacks_Permissions – user does not have permissions to use this API.

Sample Code

RequestTemplates – C#

```
// Request all the templates that this account has on file
```

```
DocuSignWeb.EnvelopeTemplateDefinition[] response =
_apiClient.RequestTemplates(_accountId,
    false);

// Print out information about the templates
foreach (DocuSignWeb.EnvelopeTemplateDefinition def in response)
{
    Console.WriteLine("Retrieved template definition with ID {0} with name \"{1}\"",
        def.TemplateID, def.Name);
}
```

RequestTemplates – PHP

```
// Request all the templates this account has on file
$requestTemplatesparams = new RequestTemplates();
$requestTemplatesparams->AccountID = $AccountID;
$requestTemplatesparams->IncludeAdvancedTemplates = false;
$rtsResponses = $api->RequestTemplates($requestTemplatesparams)->RequestTemplatesResult-
>EnvelopeTemplateDefinition;
```

SaveTemplate

Schema

Name	Schema Type	Description
EnvelopeTemplate	EnvelopeTemplate	Template to save. EnvelopeTemplate object was described earlier.

This method returns SaveTemplateResult.

Input to SaveTemplate is the EnvelopeTemplate object described earlier.

Rules and exceptions for SaveTemplate

- Account_Lacks_Permissions – account does not have permissions to use this API.
- User_Lacks_Permissions – user does not have permissions to use this API or the user does not have permissions to manage templates for the account.
- If the template ID is not provided in EnvelopeTemplate a new template will be created.

Sample Code

SaveTemplate – C#

```
// Request all the templates as shown in linked code
DocuSignWeb.EnvelopeTemplateDefinition[] response = <templates request>;

foreach (DocuSignWeb.EnvelopeTemplateDefinition def in response)
{
    Console.WriteLine("Retrieved template definition with ID {0}", def.TemplateID);

    // Request the template specified by the template ID
    DocuSignWeb.EnvelopeTemplate template = _apiClient.RequestTemplate(def.TemplateID,
        true);

    // Modify the name of the template
```

```
    template.EnvelopeTemplateDefinition.Name = "Modified name";

    // Save the template back to the account
    DocuSignWeb.SaveTemplateResult result = _apiClient.SaveTemplate(template);

    // Confirm that the save operation succeeded
    Console.WriteLine("\tSaved template? {0}", result.Success);
}
```

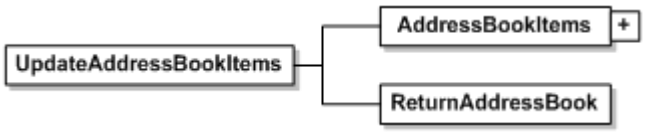
SaveTemplate – PHP

```
// Request all the templates this account has on file
$requestTemplatesparams = new RequestTemplates();
$requestTemplatesparams->AccountID = $AccountID;
$requestTemplatesparams->IncludeAdvancedTemplates = false;
$rtsResponses = $api->RequestTemplates($requestTemplatesparams)->RequestTemplatesResult-
>EnvelopeTemplateDefinition;

// Make a new template from each template on file and save it
$requestTemplateparams = new RequestTemplate();
$requestTemplateparams->IncludeDocumentBytes = true;
$responses = array();
foreach ($rtsResponses as $envTemplateDef) {
    $requestTemplateparams->TemplateID = $envTemplateDef->TemplateID;
    $template = $api->RequestTemplate($requestTemplateparams)->RequestTemplateResult;
    $template->EnvelopeTemplateDefinition->Name = "Modified Name";
    $saveTemplateparams = new SaveTemplate();
    $saveTemplateparams->EnvelopeTemplate = $template;
    $response = $api->SaveTemplate($saveTemplateparams);
    array_push($responses, $response);
}
```

UpdateAddressBookItems

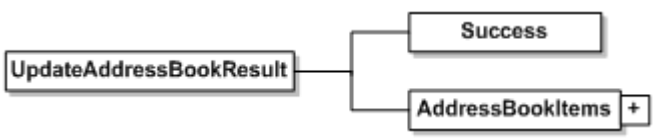
Updates and inserts all the specified items passed to your address book. This method returns an UpdateAddressBookResult object.

Schema



Name	Schema Type	Description
AddressBookItems	AddressBookItem Array	Array of address book items to update or insert.
ReturnAddressBook	Boolean	If true, return the address book item list in the UpdateAddressBookResult object.

UpdateAddressBookResult



Name	Schema Type	Description
Success	Boolean	True if successful.
AddressBookItems	AddressBookItem Array	Array of AddressBookItem after the updates are completed. AddressBookItem object described earlier.

Rules and exceptions for UpdateAddressBookItems

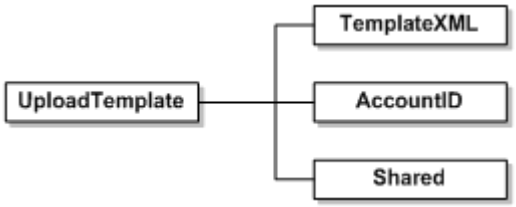
- Account_Lacks_Permissions – account does not have permissions to use this API.
- User_Lacks_Permissions – user does not have permissions to use this API.
- Address book items that are passed to this API that the user does not own will not be removed.
- If the address book ID is left blank in the AddressBookItem object a new address book entry will be added.
- If the AddressBookItem share is changed and the user does not have address book sharing privileges, the address book item share will not be updated.
- AddressBook_Email_Invalid – if there is an invalid email in the address book items passed to the API this error will be returned and no updates will be done.

UploadTemplate

This method returns SaveTemplateResult.

Note: this method currently only supports DocuSign Professional Template XML.

Schema



Name	Schema Type	Description
TemplateXML	String	XML data to be transformed into a DocuSign server side template. NOTE: only DocuSign Professional template XML is supported.
AccountID	DSXId	Account ID to add the template to. If blank, the API user's default account will be used.
Shared	Boolean	If true, the template is shared with the Everyone group in the account. If false, the template is only shared with the Administrator group.

Rules and exceptions for UploadTemplate

- Account_Lacks_Permissions – account does not have permissions to use this API.
- User_Lacks_Permissions – user does not have permissions to use this API or the user does not have permissions to manage templates for the account.

Sample Code

UploadTemplate – C#

```
// Read the template that you want to upload
String templateXML = System.IO.File.ReadAllText("path to template");

// Upload the specified template to the account
DocuSignWeb.SaveTemplateResult result = _apiClient.UploadTemplate(templateXML,
    _accountId, false);

// Confirm that the template saved successfully
Console.WriteLine("The template was successfully uploaded? {0}", result.Success);
```

UploadTemplate – PHP

```
// Read the template that you want to upload
$templateXML = file_get_contents("path to template");

// Send
$uploadTemplateparams = new UploadTemplate();
$uploadTemplateparams->AccountID = $AccountID;
$uploadTemplateparams->TemplateXML = $templateXML;
$uploadTemplateparams->Shared = true;
$response = $api->UploadTemplate($uploadTemplateparams);
```

Embedded Callback Event Codes

When DocuSign redirects the *Recipient* to the Embedded client application, it will append a URL parameter "event" to the *CallbackURL* to indicate which terminal state the *Recipient* reached. The valid values are:

Event	Description
OnSigningComplete	The Recipient successfully completed the signing the document.
OnViewingComplete	The Recipient successfully viewed the document.
OnCancel	The Recipient canceled the document while viewing the envelope.
OnDecline	The recipient declined the envelope.
OnSessionTimeout	The recipient session has timed out.
OnTTLExpired	If a Recipient Token URL is invoked after it is expired.
OnAccessCodeFailed	The Recipient failed to provide the correct Access Code.
OnIdCheckFailed	The Recipient failed to pass the ID Check authentication questions.
OnException	If an exception is thrown.
OnFaxPending	If a faxed in document is pending.

Asynchronous Document Generation

The following value can be set to cause the system to report that the envelope has been completed before the actual final documents have been generated.

Name	Schema Type	Description
GenerateSignedDocumentAsynch	Boolean	Optional flag. If true, the final document generation will happen after the envelope status has been set to 'Completed'. Default is false.

Credential API

This section covers the methods involved in implementing a DocuSign Credential API. The DocuSign Credential API can be used to authenticate with an email and password to acquire your API credentials. The credentials can then be used to access the DocuSign Connect API methods described earlier. If you have an integrators key from DocuSign you can place it in front of the email argument bracketed, "[" and "]".

The DocuSign Credential API exposes the following methods:

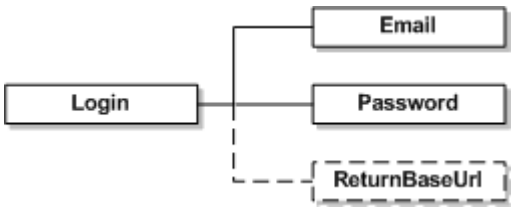
Login – Allows you to authenticate with an email and password to retrieve your API credentials required to access the DocuSign Connect API.

Ping – Allows you to validate that the Credential API is alive.

Login

The Login method can be used to authenticate and retrieve DocuSign Connect API credentials based on an email and password. This method will return an array of valid DocuSign Connect API accounts that match the email and password provided.

Schema



Name	Schema Type	Description
Email	String	Email to authenticate with. Place your bracketed integrators key in front of the email if you have received one from DocuSign. Sample: [integrator key]email
Password	String	Password to authenticate with.
ReturnBaseUrl	Boolean	When true, the response will return the BaseUrl string. This contains information about the server location of your account. The string is should be used in the header for future API calls.

Sample Request XML:

SOAP 1.1: The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

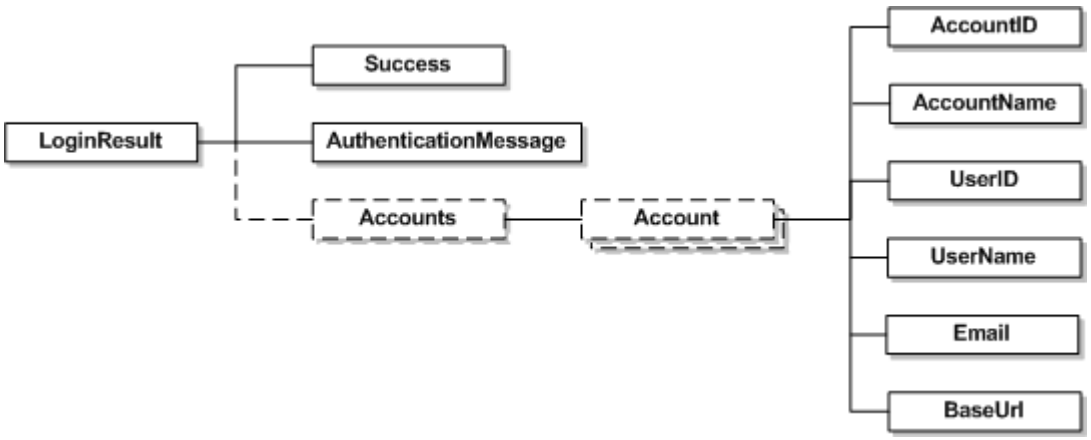
```
POST /api/3.0/credential.asmx HTTP/1.1
Host: www.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/Credential/Login"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Login xmlns="http://www.docusign.net/API/Credential">
      <Email>string</Email>
      <Password>string</Password>
      <ReturnBaseUrl>boolean</ReturnBaseUrl>
    </Login>
  </soap:Body>
</soap:Envelope>
```

SOAP 1.2: The following is a sample SOAP 1.2 request and response. The placeholders shown need to be replaced with actual values.

```
POST /api/3.0/credential.asmx HTTP/1.1
Host: www.docusign.net
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <Login xmlns="http://www.docusign.net/API/Credential">
      <Email>string</Email>
      <Password>string</Password>
      <ReturnBaseUrl>boolean</ReturnBaseUrl>
    </Login>
  </soap12:Body>
```

LoginResult



Name	Schema Type	Description
AccountID	DSXId	Account ID of the valid DocuSign Connect API user.
AccountName	String	Account Name of the valid DocuSign Connect API user.
UserID	DSXId	User ID of the valid DocuSign Connect API user.

Name	Schema Type	Description
UserName	String	Username of the valid DocuSign Connect API user.
Email	String	Email of the valid DocuSign Connect API user.
BaseUrl	String	The BaseUrl string for your account. This contains information about the server location of your account. The string is should be used in the header for future API calls.

Ping

Ping can be used to do a quick check to see if the DocuSign Credential service is available.

Schema

Sample Request XML:

```
SOAPAction: http://www.docusign.net/API/Credential/Ping

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Ping xmlns="http://www.docusign.net/API/Credential" />
  </soap:Body>
</soap:Envelope>
```

PingResult

Returns true if successful.

Sample Code

Ping – C#

```
// Use this function to validate that the Credential API is alive
bool retval = _apiClient.Ping();

// Confirm that we got an appropriate reply
if (retval)
{
    Console.WriteLine("Ping succeeded.");
}
```

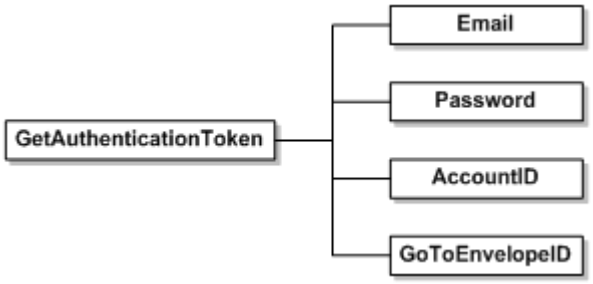
Ping – PHP

```
$pingParams = new Ping();
$response = $am_api->Ping($pingParams);
```

GetAuthenticationToken

This method can be used to get a onetime use URL with an authentication token to launch the DocuSign member system.

Schema



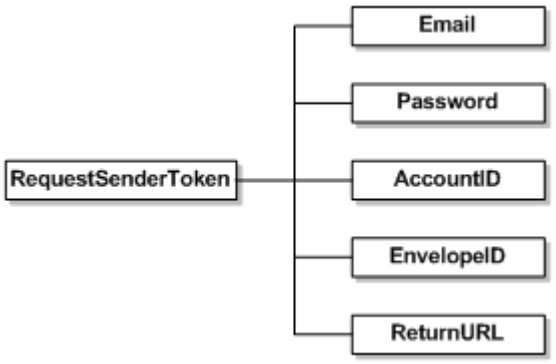
Name	Schema Type	Description
Email	string	Email to authenticate with. Place your bracketed integrators key in front of the email if you have received one from DocuSign. Sample: [integrator key]email
Password	string	Password to authenticate with.
AccountID	string	Optional DocuSign account ID for the user. If left blank the default account ID will be used.
GoToEnvelopeID	String	Optional envelope ID. If provided the URL returned will send the user directly to viewing the envelope. If left blank the user will be taken to the member console.

This method returns a string that is the URL to access the DocuSign member system. This URL can be placed in a browser and you will be automatically logged into the member system.

RequestSenderToken

This method is used to get a onetime use login token that allows the user to be placed into the DocuSign sending wizard. Upon sending completion the user is returned to the return URL provided by the API application.

Schema



Name	Schema Type	Description
Email	String	Email to authenticate with. Place your bracketed integrators key in front of the email if you have received one from DocuSign. Sample: [integrator key]email
Password	String	Password to authenticate with.
EnvelopeID	String	Optional envelope ID. If provided the user will be placed into the envelope in process. If left blank the user will be placed into a new envelope for sending.
AccountID	String	DocuSign account ID to map the envelope to.
ReturnURL	String	URL to send the user to upon completion of the sending process. The URL will have an event passed to it as a query parameter. The parameter will be named “event”, see “In-session sending events” below for description of the valid events. The DocuSign Envelope Id will also be returned in the “envelopeId” parameter. **Important** – You must include HTTPS:// in the URL or the redirect might be blocked by some browsers.

In-session sending events

Events returned on the ReturnURL as a query string parameter. The event is sent in the parameter named “event”. The ReturnURL is called with the event on any completion state of the envelope send. Valid events can be found in the schema type InSessionSendEvent in the DocuSign API XSD.

Event	Description
Send	User sends the envelope.
Save	User saves a draft of the envelope.
Cancel	User cancels the sending transaction. No envelope ID returned.
Error	Error performing the send.
SessionEnd	Sending session ended before the user completed.

Rules and exceptions for RequestSenderToken

- User_Does_Not_Exist_In_System – API user requesting the sending token URL does not have access to the system.
- Account_Does_Not_Exist_In_System – Account is not valid.
- Account_Lacks_Permissions – Account does not have valid permissions.
- User_Lacks_Permissions – user does not have valid permissions.
- User_Authentication_Failed – user/password is not valid.

Account Management Service API

This section of the Developer's Guide provides information about the different methods in the DocuSign Account Management Service API.

Methods Exposed in the DocuSign Account Management Service API

The DocuSign Account Management Service API exposes the following major methods (listed alphabetically):

- **ActivateSalesforceInstance** - This method is used by the DocuSign for Salesforce package to connect Salesforce to DocuSign.
- **AddMembersToAccount** - This method is used to add new members to an existing account.
- **AuthenticateMember** and **AuthenticateMemberEx** - These methods requests the status of a member in an account.
- **ChangeAccountPricePlan** - This method is used to change the billing plan for an account.
- **ChangePassword** – This method is used to change the password for an account member.
- **CheckAccountMember** - This method is used to determine if user email is a member of the specified account.
- **CloseMembers** – This method is used to close member accounts in the same manner as closing an account through the DocuSign web console.
- **CloseSignature** - This method is used to remove a signature for a user.
- **GetAccountCustomFields** - This method is used to retrieve a list of the envelope custom fields associated with the account.
- **GetAccountDistributorCode** - This method requests the Distributor Code for an account.
- **GetAccountInformation** - This method requests the account level attributes for an account.
- **GetAccountSettings** - This method requests the settings for an account.
- **GetConnectCredentials** - This method is used by the DocuSign for Salesforce package to retrieve the Salesforce connect settings.
- **GetEncryptedPassword** - This method requests the encrypted form of the user's password. This encrypted password may be used for authentication in other API function calls along with a User Name or User ID and an Integrator Key.
- **GetMemberSettings** - This method requests some attributes about the specified member of the account.
- **GetMembershipSummary** – This is reserved for DocuSign use. This function returns a list of summary information about the accounts and users associated with the supplied email address.
- **GetPlanGroupInformation** - This function returns the plan group names and plans associated with each plan group for the given distributor code. This information is used to assign billing plans to an account.
- **GetPlanPricingInformation** - This function returns the plan pricing information associated with the given distributor code.

- **GetPlanType** - This function requests the plan payment type information for an account. The payment types use the same terminology as the payment method.
- **GetProvisioningInformation** - This function returns account provisioning information, the DistributorCode, DistributorPassword, and other plan information, for the given Token.
- **GetSignatures** – This method is reserved for DocuSign use only.
- **GetSuccessorPlanInformation** - This function returns the current plan information and a list of successor plans, plans that the current plan could be rolled into, associated with the given distributor code and account.
- **GetUserProfile** - This method retrieves the privacy settings and personal information (address, phone number, etc.) from a user ID card.
- **GetUserProfileImage** - This method retrieves the image from a user ID card.
- **InitializeClientAccount** – This method is reserved for DocuSign use only.
- **NewAccount** - This method creates a new DocuSign account with a single member, who is the account owner.
- **NewSocialAccount** – This method is reserved for DocuSign use only.
- **Ping** - This method determines if the Account Management API service can be contacted.
- **ResendAccountActivation** - This method sends another account activation message to an account user.
- **SetConnectCredentials** - This method is used by the DocuSign for Salesforce package to set the connection used by DocuSign to update Salesforce data.
- **SetSignatureImages** – This method is reserved for DocuSign use only.
- **SetUserProfile** - This method is used to set the privacy settings and personal information (address, phone number, etc.) from a user ID card.
- **SetUserProfileImage** - Thsis method is used to upload an image to a user ID card.
- **UpdateAccountSettings** - This method is used to modify account level settings.
- **UpdateMemberSettings** - This method is used to modify settings for a member of an account.
- **UpgradeRecipientAccount** - This function is used to upgrade a user from a free account, where the user has few privileges, to a paid account plan.

Account Management Service API Methods

This section outlines the usage rules and behaviors of the Account Management Service methods. The functions are presented in alphabetical order.

ActivateSalesforceInstance

This method is used by the DocuSign for Salesforce package to connect Salesforce to DocuSign.

Schema

Name	Schema Type	Description
AccountId	String	The DocuSign account ID to which you are trying to connect.
ExternalInstanceId	String	The Salesforce session ID.
ConnectUsername	String	The Salesforce login user name.
ConnectPassword	String	The Saleforce login password
SalesforceEnvironment	String	The Salesforce environment being accessed (i.e: Production or Test).
Member	Member	A complex type that contains information about the member being activated. See the Member section for more information.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/ActivateSalesforceInstance"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ActivateSalesforceInstance xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
      <ExternalInstanceId>string</ExternalInstanceId>
      <ConnectUsername>string</ConnectUsername>
      <ConnectPassword>string</ConnectPassword>
      <SalesforceEnvironment>string</SalesforceEnvironment>
      <Member>
        <MemberEmailAddress>string</MemberEmailAddress>
        <MemberUserName>string</MemberUserName>
        <MemberPassword>string</MemberPassword>
        <MemberForgottenPasswordQuestion>string</MemberForgottenPasswordQuestion>
        <MemberForgottenPasswordAnswer>string</MemberForgottenPasswordAnswer>
        <MemberTitle>string</MemberTitle>
        <MemberFirstName>string</MemberFirstName>
        <MemberMiddleName>string</MemberMiddleName>
        <MemberLastName>string</MemberLastName>
        <MemberSuffix>string</MemberSuffix>
        <EnableConnectForUser>boolean</EnableConnectForUser>
        <MemberSettings>
          <CanManageAccount>boolean</CanManageAccount>
          <CanSendEnvelope>boolean</CanSendEnvelope>
          <CanSendAPIRequests>boolean</CanSendAPIRequests>
          <APIAccountWideAccess>boolean</APIAccountWideAccess>
          <EnableVaulting>boolean</EnableVaulting>
          <VaultingMode>bytes</VaultingMode>
          <EnableTransactionPoint>boolean</EnableTransactionPoint>
          <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
          <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
          <EnableDSPro>boolean</EnableDSPro>
          <PowerFormAdmin>boolean</PowerFormAdmin>
```

```
          <PowerFormUser>boolean</PowerFormUser>
          <CanEditSharedAddressBook>bytes</CanEditSharedAddressBook>
          <CanManageTemplates>bytes</CanManageTemplates>
          <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
          <EnableSignerAttachments>boolean</EnableSignerAttachments>
        </MemberSettings>
        <ReturnEncryptedPassword>boolean</ReturnEncryptedPassword>
      </Member>
    </ActivateSalesforceInstance>
  </soap:Body>
</soap:Envelope>
```

The result contains the information below and returns either a success or failure. If the call fails an error code is provided.

Name	Schema Type	Description
AccountId	String	The DocuSign account ID to which you are connected.
SiteId	String	The DocuSign account number.
UserId	String	The User ID for the connected account.
MembershipId	String	The membership ID for the connected account.
EncryptedPassword	String	The encrypted password for the authenticated user.

Sample Return XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ActivateSalesforceInstanceResponse
xmlns="http://www.docusign.net/API/AccountManagement">
      <ActivateSalesforceInstanceResult>
        <AccountId>string</AccountId>
        <SiteId>string</SiteId>
        <UserId>string</UserId>
        <MembershipId>string</MembershipId>
        <EncryptedPassword>string</EncryptedPassword>
        <Success>boolean</Success>
        <Error>
```

```
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </ActivateSalesforceInstanceResult>
    </ActivateSalesforceInstanceResponse>
  </soap:Body>
</soap:Envelope>
```

Member

This element contains general information for a member.

Name	Schema Type	Description
MemberEmailAddress	String	The member's email address for the associated account. This can be a maximum of 100 characters.
MemberUserName	String	The member's name for the associated account. This can be a maximum of 100 characters.
MemberPassword	String	The member's password for the associated account. This can be a maximum of 50 characters.
MemberForgottenPasswordQuestion	String	The member's forgotten password question for the associated account.
MemberForgottenPasswordAnswer	String	The answer to the member's forgotten password question for the associated account.
MemberTitle	String	The member's title for the associated account. This can be a maximum of 10 characters.
MemberFirstName	String	The member's first name. This can be a maximum of 50 characters.
MemberMiddileName	String	The member's middle name. This can be a maximum of 50 characters.

Name	Schema Type	Description
MemberLastName	String	The member's last name. This can be a maximum of 50 characters.
MemberSuffix	String	The suffix for the member's name. This can be a maximum of 50 characters.
EnableConnectForUser	Boolean	When true, the user is enabled for connect.
MemberSettings	MemberSettings	A complex type that contains the account settings for the member. See MemberSettings for more information.
ReturnEncryptedPassword	Boolean	When true, returns the encrypted DocuSign password sent in the authentication header.

MemberSettings

This element contains details about the account settings for a member.

Name	Schema Type	Description
CanManageAccount	Boolean	When true, this user can manage account settings, manage user settings, add users, and remove users.
CanSendEnvelope	Boolean	When true, this user can send envelopes though the DocuSign Console.
CanSendAPIRequests	Boolean	When true, this user can send and manage envelopes using the DocuSign API.
APIAccountWideAccess	Boolean	When true, this user can send and manage envelopes for the entire account using the DocuSign API.
EnableVaulting	Boolean	When true, this user can use electronic vaulting for documents.
VaultingMode	String	This element sets the electronic vaulting mode for the user. Enumeration values are: None, eStored and electronicOriginal.
EnableTransactionPoint	Boolean	When true, this user can select an envelope from their member console and upload the envelope documents to TransactionPoint.
EnableSequentialSigningAPI	Boolean	When true, this user can define the routing order of recipients for envelopes sent using the DocuSign API.
EnableSequentialSigningUI	Boolean	When true, this user can define the routing order of recipients while sending documents for signature.
EnableDSPro	Boolean	When true, this user can send and manage envelopes from the DocuSign Desktop Client.

Name	Schema Type	Description
PowerFormAdmin	Boolean	When true, this user can create, manage and download the PowerForms documents.
PowerFormUser	Boolean	When true, this user can view and download PowerForms documents.
CanEditSharedAddressBook	String	This element sets the address book usage and management rights for the user. Enumeration values are: None, UseOnlyShared, UsePrivateAndShared, Share.
CanManageTemplates	String	This element sets the template usage and management rights for the user. Enumeration values are: None, Use, Create, Share.
EnableSignOnPaperOverride	Boolean	When true, this user can override the account setting that determines if signers may sign their documents on paper as an option to signing electronically.
EnableSignerAttachments	Boolean	When true, this user can add requests for attachments from signers while sending documents.
Locale	String	This sets the default language for the user. The supported languages, with the language value shown in parenthesis, are: Chinese Simplified (zh_CN), Chinese Traditional (zh_TW), Dutch (nl), English US (en), French (fr), German (de), Italian (it), Japanese (ja), Korean (ko), Portuguese (pt), Portuguese (Brazil) (pt_BR), Russian (ru), Spanish - (es).

AddMembersToAccount

This method is used to add new members to an existing account.

Schema

Name	Schema Type	Description
AccountId	String	The account ID of the account to which the members are being added.
Members	ArrayOfMember	An array of Member elements for members that are being added to the account. See the Member section for more information.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/AddMembersToAccount"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <AddMembersToAccount xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
      <Members>
        <Member>
          <MemberEmailAddress>string</MemberEmailAddress>
          <MemberUserName>string</MemberUserName>
          <MemberPassword>string</MemberPassword>
          <MemberForgottenPasswordQuestion>string</MemberForgottenPasswordQuestion>
          <MemberForgottenPasswordAnswer>string</MemberForgottenPasswordAnswer>
          <MemberTitle>string</MemberTitle>
          <MemberFirstName>string</MemberFirstName>
          <MemberMiddleName>string</MemberMiddleName>
          <MemberLastName>string</MemberLastName>
          <MemberSuffix>string</MemberSuffix>
          <EnableConnectForUser>boolean</EnableConnectForUser>
          <MemberSettings>
            <CanManageAccount>boolean</CanManageAccount>
            <CanSendEnvelope>boolean</CanSendEnvelope>
            <CanSendAPIRequests>boolean</CanSendAPIRequests>
            <APIAccountWideAccess>boolean</APIAccountWideAccess>
            <EnableVaulting>boolean</EnableVaulting>
            <VaultingMode>bytes</VaultingMode>
            <EnableTransactionPoint>boolean</EnableTransactionPoint>
            <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
            <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
            <EnableDSPro>boolean</EnableDSPro>
            <PowerFormAdmin>boolean</PowerFormAdmin>
            <PowerFormUser>boolean</PowerFormUser>
            <CanEditSharedAddressBook>bytes</CanEditSharedAddressBook>
            <CanManageTemplates>bytes</CanManageTemplates>
            <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
            <EnableSignerAttachments>boolean</EnableSignerAttachments>
          </MemberSettings>
          <ReturnEncryptedPassword>boolean</ReturnEncryptedPassword>
        </Member>
        <Member>
          <MemberEmailAddress>string</MemberEmailAddress>
          <MemberUserName>string</MemberUserName>
          <MemberPassword>string</MemberPassword>
          <MemberForgottenPasswordQuestion>string</MemberForgottenPasswordQuestion>
          <MemberForgottenPasswordAnswer>string</MemberForgottenPasswordAnswer>
          <MemberTitle>string</MemberTitle>
          <MemberFirstName>string</MemberFirstName>
          <MemberMiddleName>string</MemberMiddleName>
          <MemberLastName>string</MemberLastName>
          <MemberSuffix>string</MemberSuffix>
          <EnableConnectForUser>boolean</EnableConnectForUser>
          <MemberSettings>
            <CanManageAccount>boolean</CanManageAccount>
            <CanSendEnvelope>boolean</CanSendEnvelope>
            <CanSendAPIRequests>boolean</CanSendAPIRequests>
            <APIAccountWideAccess>boolean</APIAccountWideAccess>
            <EnableVaulting>boolean</EnableVaulting>
            <VaultingMode>bytes</VaultingMode>
            <EnableTransactionPoint>boolean</EnableTransactionPoint>
            <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
            <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
            <EnableDSPro>boolean</EnableDSPro>
```

```
            <PowerFormAdmin>boolean</PowerFormAdmin>
            <PowerFormUser>boolean</PowerFormUser>
            <CanEditSharedAddressBook>bytes</CanEditSharedAddressBook>
            <CanManageTemplates>bytes</CanManageTemplates>
            <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
            <EnableSignerAttachments>boolean</EnableSignerAttachments>
          </MemberSettings>
          <ReturnEncryptedPassword>boolean</ReturnEncryptedPassword>
        </Member>
      </Members>
    </AddMembersToAccount>
  </soap:Body>
</soap:Envelope>
```

The response contains an array of MemberResults, with the information below, and either a success or failure. If the call fails an error code is provided.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <AddMembersToAccountResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <AddMembersToAccountResult>
        <Members>
          <Member>
            <UserId>string</UserId>
            <EncryptedPassword>string</EncryptedPassword>
            <MembershipId>string</MembershipId>
            <Success>boolean</Success>
            <Error xsi:nil="true" />
          </Member>
          <Member>
            <UserId>string</UserId>
            <EncryptedPassword>string</EncryptedPassword>
            <MembershipId>string</MembershipId>
            <Success>boolean</Success>
            <Error xsi:nil="true" />
          </Member>
        </Members>
        <Success>boolean</Success>
        <Error>
```

```
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </AddMembersToAccountResult>
    </AddMembersToAccountResponse>
  </soap:Body>
</soap:Envelope>
```

MemberResult

Name	Schema Type	Description
UserId	String	The User ID for the new member.
EncryptedPassword	String	The encrypted password for the authenticated user.
MembershipId	String	The membership ID for the new member.

AuthenticateMember and AuthenticateMemberEx

These method requests the status of a member in an account.

Schema

Name	Schema Type	Description
AccountId	String	The account ID associated with the member you are authenticating.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/AuthenticateMember"

<?xml version="1.0" encoding="utf-8"?>
```

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <AuthenticateMember xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
    </AuthenticateMember>
  </soap:Body>
</soap:Envelope>
```

The response contains the information below and either a success or failure. If the call fails an error code is provided.

Name	Schema Type	Description
UserId	String	The user ID for the member being authenticated.
EncryptedPassword	String	The encrypted password for the authenticated user.
UsesAPI	Boolean	When true, the user can use the DocuSign API.
MemberSettings	MemberSettings	A complex type that contains the account settings for the member. See MemberSettings for more information.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <AuthenticateMemberResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <AuthenticateMemberResult>
        <UserId>string</UserId>
        <AccountId>string</AccountId>
        <UsesAPI>boolean</UsesAPI>
        <MemberSettings>
          <CanManageAccount>boolean</CanManageAccount>
          <CanSendEnvelope>boolean</CanSendEnvelope>
          <CanSendAPIRequests>boolean</CanSendAPIRequests>
          <APIAccountWideAccess>boolean</APIAccountWideAccess>
          <EnableVaulting>boolean</EnableVaulting>
          <VaultingMode>bytes</VaultingMode>
          <EnableTransactionPoint>boolean</EnableTransactionPoint>
          <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
          <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
          <EnableDSPro>boolean</EnableDSPro>
          <PowerFormAdmin>boolean</PowerFormAdmin>
          <PowerFormUser>boolean</PowerFormUser>
          <CanEditSharedAddressBook>bytes</CanEditSharedAddressBook>
          <CanManageTemplates>bytes</CanManageTemplates>
          <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
          <EnableSignerAttachments>boolean</EnableSignerAttachments>
        </MemberSettings>
        <Success>boolean</Success>
```

```
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </AuthenticateMemberResult>
    </AuthenticateMemberResponse>
  </soap:Body>
</soap:Envelope>
```

ChangeAccountPricePlan

This method is used to change the billing plan for an account

Schema

Name	Schema Type	Description
DistributorCode	String	The Distributor Code that identifies the billing plan groups and plans.
DistributorPassword	String	The Distributor Password for the DistributorCode.
AccountId	String	The account ID for the account you want to change.
Pgp	String	The plan group plan for the account. The Pgp uniquely identifies a plan/plan group and is used to set plans in other functions.
EnableSupport	Boolean	If true the plan has support enabled.
IncludedSeats	Integer	The number of seats included in the plan
CreditCardInformation	CreditCardInformation	This complex type has information about the credit card used to pay for this account. It included the elements: ccNumber, ccExpirationMonth, ccExpirationYear, ccUserName, and ccType.

Name	Schema Type	Description
AddressInformation	AddressInformation	This complex type contains the following information: Street1, Street2, City, State, Zip, Phone, and Fax. The maximum characters or each field are: - Street1, Street2, City, and State: 100 characters - ZIP, Phone, and Fax: 20 characters

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/ChangeAccountPricePlan"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ChangeAccountPricePlan xmlns="http://www.docusign.net/API/AccountManagement">
      <DistributorCode>string</DistributorCode>
      <DistributorPassword>string</DistributorPassword>
      <AccountId>string</AccountId>
      <Pgp>string</Pgp>
      <EnableSupport>boolean</EnableSupport>
      <IncludedSeats>int</IncludedSeats>
      <CreditCardInformation>
        <ccNumber>string</ccNumber>
        <ccExpirationMonth>string</ccExpirationMonth>
        <ccExpirationYear>string</ccExpirationYear>
        <ccUserName>string</ccUserName>
        <ccType>string</ccType>
      </CreditCardInformation>
      <AddressInformation>
        <Address1>string</Address1>
        <Address2>string</Address2>
        <City>string</City>
        <State>string</State>
        <Zip>string</Zip>
        <Phone>string</Phone>
        <Fax>string</Fax>
      </AddressInformation>
    </ChangeAccountPricePlan>
  </soap:Body>
</soap:Envelope>
```

The response returns a success or a success or failure. If the call fails an error code is provided.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
```

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ChangeAccountPricePlanResponse
xmlns="http://www.docusign.net/API/AccountManagement">
      <ChangeAccountPricePlanResult>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </ChangeAccountPricePlanResult>
    </ChangeAccountPricePlanResponse>
  </soap:Body>
</soap:Envelope>
```

ChangePassword

This method is used to change the password for an account member. It uses ChangePasswordArg with the following information to change the account member's password.

Schema

Name	Schema Type	Description
EmailAddress	String	The email address for the account member.
OldPassword	String	The account member's old password.
NewPassword	String	The new password for the account member.
ForgottenPasswordQuestion	String	The member's forgotten password question. Note: The number of forgotten password questions and answers required is set in the Password Strength Settings for your account.
ForgottenPasswordAnswer	String	The answer to the member's forgotten password question.
ForgottenPasswordQuestion2	String	The member's second forgotten password question.

Name	Schema Type	Description
ForgottenPasswordAnswer2	String	The answer to the member's second forgotten password question.
ForgottenPasswordQuestion3	String	The member's forgotten third password question.
ForgottenPasswordAnswer3	String	The answer to the member's third forgotten password question.
ForgottenPasswordQuestion4	String	The member's fourth forgotten password question.
ForgottenPasswordAnswer4	String	The answer to the member's fourth forgotten password question.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/ChangePassword"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ChangePassword xmlns="http://www.docusign.net/API/AccountManagement">
      <ChangePasswordArg>
        <EmailAddress>string</EmailAddress>
        <OldPassword>string</OldPassword>
        <NewPassword>string</NewPassword>
        <ForgottenPasswordQuestion>string</ForgottenPasswordQuestion>
        <ForgottenPasswordAnswer>string</ForgottenPasswordAnswer>
        <ForgottenPasswordQuestion2>string</ForgottenPasswordQuestion2>
        <ForgottenPasswordAnswer2>string</ForgottenPasswordAnswer2>
        <ForgottenPasswordQuestion3>string</ForgottenPasswordQuestion3>
        <ForgottenPasswordAnswer3>string</ForgottenPasswordAnswer3>
        <ForgottenPasswordQuestion4>string</ForgottenPasswordQuestion4>
        <ForgottenPasswordAnswer4>string</ForgottenPasswordAnswer4>
      </ChangePasswordArg>
    </ChangePassword>
  </soap:Body>
</soap:Envelope>
```

The response contains either a success or failure. If the call fails an error code is provided.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ChangePasswordResponse xmlns="http://www.docusign.net/API/AccountManagement">
```

```
      <ChangePasswordResult>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed or
Hourly_API_Invocation_Limit_Exceeded or Initialize_Client_Account_Error</ErrorCode>
          <Description>string</Description>
        </Error>
      </ChangePasswordResult>
    </ChangePasswordResponse>
  </soap:Body>
</soap:Envelope>
```

CheckAccountMember

This method is used to determine if user email is a member of the specified account.

Schema

Name	Schema Type	Description
AccountId	String	The account ID associated with the member.
Email	String	The user email to be checked.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/CheckAccountMember"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CheckAccountMember xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
      <Email>string</Email>
    </CheckAccountMember>
  </soap:Body>
```

```
</soap:Envelope>
```

The response contains the information below and either a success or failure. If the call fails an error code is provided.

Name	Schema Type	Description
UserId	String	The ID of the user associated with the account.
Status	String	If true, the user email is a member of the account. If false, an error message is provided.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CheckAccountMemberResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <CheckAccountMemberResult>
        <UserId>string</UserId>
        <Status>string</Status>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </CheckAccountMemberResult>
    </CheckAccountMemberResponse>
  </soap:Body>
</soap:Envelope>
```

CloseMembers

This function closes member accounts in the same manner as closing an account through the DocuSign web console.

CloseMembers takes the argument:

Name	Schema Type	Description
AccountId	String	The account ID associated with the member accounts being closed.
UserIds	ArrayofString	An array with the User IDs for the member accounts being closed.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: demo.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/CloseMembers"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CloseMembers xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
      <UserIds>
        <string>string</string>
        <string>string</string>
      </UserIds>
    </CloseMembers>
  </soap:Body>
</soap:Envelope>
```

The response contains the overall success or failure of the call. If successful, the response contains the Members result information below. If the call fails an error code is provided.

Name	Schema Type	Description
Members	ArrayOfMemberResult	An array with member information for the accounts being closed. See MemberResult for more information.
DateStamp	dateTime	The data time stamp for when the member account was closed.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CloseMembersResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <CloseMembersResult>
        <Success>boolean</Success>
        <Members>
          <Member>
```

```
            <UserId>string</UserId>
            <EncryptedPassword>string</EncryptedPassword>
            <MembershipId>string</MembershipId>
            <Success>boolean</Success>
            <Error xsi:nil="true" />
          </Member>
          <Member>
            <UserId>string</UserId>
            <EncryptedPassword>string</EncryptedPassword>
            <MembershipId>string</MembershipId>
            <Success>boolean</Success>
            <Error xsi:nil="true" />
          </Member>
        </Members>
        <DateStamp>dateTime</DateStamp>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed or
Hourly_API_Invocation_Limit_Exceeded or Initialize_Client_Account_Error</ErrorCode>
          <Description>string</Description>
        </Error>
      </CloseMembersResult>
    </CloseMembersResponse>
  </soap:Body>
</soap:Envelope>
```

CloseSignature

This method is used to remove a signature for a user.

CloseSignature take the argument:

Name	Schema Type	Description
SignatureName	String	The name of the signature file that should be closed.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
```

```
SOAPAction: "http://www.docusign.net/API/AccountManagement/CloseSignature"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CloseSignature xmlns="http://www.docusign.net/API/AccountManagement">
      <SignatureName>string</SignatureName>
    </CloseSignature>
  </soap:Body>
</soap:Envelope>
```

The response contains the overall success or failure of the request.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CloseSignatureResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <CloseSignatureResult>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed or
Hourly_API_Invocation_Limit_Exceeded or Initialize_Client_Account_Error or
Max_Members_Exceeded or Invalid_File_Or_Too_Large</ErrorCode>
          <Description>string</Description>
        </Error>
      </CloseSignatureResult>
    </CloseSignatureResponse>
  </soap:Body>
</soap:Envelope>
```

GetAccountCustomFields

This retrieves a list of the envelope custom fields associated with the account. These fields can be used in the envelopes for your account to record information about the envelope, help search for

envelopes and track information. The envelope custom fields are shown in the Envelope Settings section when a user is creating an envelope in the DocuSign member console. The envelope custom fields are not seen by the envelope recipients.

There are two types of envelope custom fields (shown by CustomFieldType), text and list. A text custom field lets the sender enter the value for the field. The list custom field lets the sender select the value of the field from a premade list.

Schema

Name	Schema Type	Description
AccountId	String	The account ID of the account for which information is requested.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetAccountCustomFields"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetAccountCustomFields xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
    </GetAccountCustomFields>
  </soap:Body>
</soap:Envelope>
```

This returns a list of envelope custom fields associated with the account and either a success or failure. If the call fails, an error code and message is provided. The following information is included in the result:

Name	Schema Type	Description
AccountCustomFields	AccountCustomField	The list of information about the envelope custom field. Each custom field has the following information: • Name – (String) The name of the envelope custom field. • CustomFieldType – The type of custom field, the values can be List or Text. • Show – (Boolean) When true, the custom field is shown to senders in the DocuSign console during sending. • Required – (Boolean) When true, information must be entered into the custom field to send the envelope. • ListItems – An array of items if the CustomFieldType is List.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetAccountCustomFieldsResponse
xmlns="http://www.docusign.net/API/AccountManagement">
      <GetAccountCustomFieldsResult>
        <AccountCustomFields>
          <AccountCustomField>
            <Name>string</Name>
            <CustomFieldType>Text or List</CustomFieldType>
            <Show>boolean</Show>
            <Required>boolean</Required>
            <ListItems xsi:nil="true" />
          </AccountCustomField>
          <AccountCustomField>
            <Name>string</Name>
            <CustomFieldType>Text or List</CustomFieldType>
            <Show>boolean</Show>
            <Required>boolean</Required>
            <ListItems xsi:nil="true" />
          </AccountCustomField>
        </AccountCustomFields>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>string</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetAccountCustomFieldsResult>
    </GetAccountCustomFieldsResponse>
  </soap:Body>
</soap:Envelope>
```

GetAccountDistributorCode

This function requests the Distributor Code for an account.

GetAccountDistributorCode takes the argument:

Name	Schema Type	Description
AccountId	String	The account ID for the account requested.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetAccountDistributorCode"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
```

```
    <GetAccountDistributorCode xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
    </GetAccountDistributorCode>
  </soap:Body>
</soap:Envelope>
```

The function returns the Distributor Code for the account and either a success or failure. If the call fails, an error code is provided. The following information included in the result:

Name	Schema Type	Description
DistributorCode	String	The Distributor Code that identifies the billing plan groups and plans.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetAccountDistributorCodeResponse
xmlns="http://www.docusign.net/API/AccountManagement">
      <GetAccountDistributorCodeResult>
        <DistributorCode>string</DistributorCode>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetAccountDistributorCodeResult>
    </GetAccountDistributorCodeResponse>
  </soap:Body>
</soap:Envelope>
```

GetAccountInformation

This method requests the account level attributes for an account.

Schema

Name	Schema Type	Description
AccountId	String	The account ID for the account being checked.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetAccountInformation"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetAccountInformation xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
    </GetAccountInformation>
  </soap:Body>
</soap:Envelope>
```

The response contains the account information below and either a success or failure. If the call fails an error code is provided.

Name	Schema Type	Description
CurrentPlanId	String	The Plan ID of the requested account.
CurrentPlanName	String	The Plan Name of the requested account.
CurrentPlanStartDate	dateTime	The Plan start date of the requested account.
CurrentPlanEndDate	dateTime	The Plan end date of the requested account.
CurrentBillingPeriodStartDate	dateTime	The billing period start date of the requested account.
CurrentBillingPeriodEndDate	dateTime	The billing period end date of the requested account.
CurrentBillingPeriodEnvelopesSent	Integer	The total number envelopes sent by this account in the current billing period.
CurrentBillingPeriodEnvelopesAllowed	Integer	The total number of envelopes that the account can send during the current billing period.
AccountSuspensionStatus	String	Blank if account is not suspended. Otherwise, it indicates the reason why the account was suspended.

Name	Schema Type	Description
AccountSuspensionDate	dateTime	The date that the account was suspended.
AccountName	String	The account name associated with the requested account.
ExternalAccountId	String	The DocuSign account number.
ConnectPermission	String	Shows the account permission for using Connect. Enumeration values are: None, Full or Sendonly.
DocusignLandingUrl	String	The URL of the landing page associated with the account.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetAccountInformationResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <GetAccountInformationResult>
        <CurrentPlanId>string</CurrentPlanId>
        <CurrentPlanName>string</CurrentPlanName>
        <CurrentPlanStartDate>dateTime</CurrentPlanStartDate>
        <CurrentPlanEndDate>dateTime</CurrentPlanEndDate>
        <CurrentBillingPeriodStartDate>dateTime</CurrentBillingPeriodStartDate>
        <CurrentBillingPeriodEndDate>dateTime</CurrentBillingPeriodEndDate>
        <CurrentBillingPeriodEnvelopesSent>integer</CurrentBillingPeriodEnvelopesSent>

<CurrentBillingPeriodEnvelopesAllowed>integer</CurrentBillingPeriodEnvelopesAllowed>
        <AccountSuspensionStatus>string</AccountSuspensionStatus>
        <AccountSuspensionDate>dateTime</AccountSuspensionDate>
        <AccountName>string</AccountName>
        <ExternalAccountId>string</ExternalAccountId>
        <ConnectPermission>string</ConnectPermission>
        <DocusignLandingUrl>string</DocusignLandingUrl>
        <Success>boolean</Success>
        <Error>
```

```
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetAccountInformationResult>
    </GetAccountInformationResponse>
  </soap:Body>
</soap:Envelope>
```

GetAccountSettings

This method requests the settings for an account.

Schema

Name	Schema Type	Description
AccountId	String	The account ID of the account being checked.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetAccountSettings"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetAccountSettings xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
    </GetAccountSettings>
  </soap:Body>
</soap:Envelope>
```

The response contains the AccountSettings information given below and either a success or failure. If the call fails an error code is provided.

AccountSettings

Name	Schema Type	Description
UsesAPI	Boolean	When true, the account can use the DocuSign API.
EnableDSPro	Boolean	When true, this account can send and manage envelopes from the DocuSign Desktop Client.
EnableSendToManage	Boolean	When true, this account can use the Editor Recipient Type.
EnableSendToAgent	Boolean	When true, this account can use the Agent Recipient Type.
EnableVaulting	Boolean	When true, this account can use electronic vaulting for documents.
EnableEnvelopeStampingByAccountAdmin	Boolean	When true, senders for this account can choose to have the envelope ID stamped in the document margins.
EnvelopeStamplingDefaultValue	Boolean	When true, envelopes sent by this account automatically have the envelope ID stamped in the margins, unless the sender selects not to have them stamped.
SignerMustHaveAccount	Boolean	When true, senders can only send an envelope to a recipient that has a DocuSign account.
SignerMustLoginToSign	Boolean	When true, an envelope signer must log in to the DocuSign console to sign an envelope.
SignerCanCreateAccount	Boolean	When true, the signer without a DocuSign account can create a DocuSign account after signing.
AllowInPerson	Boolean	When true, the account allows In Person Signing.
EnablePowerForm	Boolean	When true, PowerForm access is enabled for the account.
AllowSignerReassign	Boolean	When true, the account allows signers to reassign an envelope.
EnableReservedDomain	Boolean	When true, an account administrator can reserve web domain and users.
EnableSequentialSigningAPI	Boolean	When true, the account can define the routing order of recipients for envelopes sent using the DocuSign API.
EnableSequentialSigningUI	Boolean	When true, the account can define the routing order of recipients for envelopes sent using the DocuSign console.
EnableAutoNav	Boolean	When true, the auto-navigation is enabled for the account.

Name	Schema Type	Description
AutoNavRule	String	The auto-navigation rule for the account. Enumeration values are: Off, RequiredFields, RequiredAndBlankFields, AllFields, PageThenRequiredFields, PageThenRequiredAndBlankFields, PageThenAllFields.
EnableTransactionPoint	Boolean	When true, Transaction Point is enabled for this account.
EnvelopeIntegrationAllowed	String	Shows the envelope integration rule for the account. Enumeration values are: NotAllowed, Full, IntegrationSendOnly.
EnvelopeIntegrationEnabled	Boolean	When true, envelope integration is enabled for the account.
CanSelfBrandSend	Boolean	When true, account administrators can self-brand their sending console through the DocuSign Console.
CanSelfBrandSign	Boolean	When true, account administrators can self-brand their signing console through the DocuSign Console.
IDCheckRequired	String	Indicates if authentication is required by envelope signers. Enumeration values are: Always, Never, or Optional. Optional means the authentication is determined by the sender.
IDCheckExpire	String	Indicates when a user's authentication expires. Enumeration values are: Always, Never, Variable. Variable uses the value in IDCheckExpireDays.
IDCheckExpireDays	Integer	The number of days before a user's authentication expires. This is only active if the IDCheckExpire value is Variable.
SignDateFormat	String	The date/time format applied to Date Signed fields for the account.
PKISignDownloadedPDFDocs	String	The policy for adding a digital certificate to downloaded, printed and emailed documents. Enumeration values are: NoSign, NoSignAllowUserOverride, YesSign
InPersonIDCheckQuestion	String	The default question used by the In Person signing host for an In Person signing session.

Name	Schema Type	Description
SessionTimeout	Integer	The amount of idle activity time, in minutes, before a user is automatically logged out of the system. The minimum setting is 20 minutes and the maximum setting is 120 minutes.
AttachCompletedEnvelope	Boolean	When true, envelope documents are included as a PDF file attachment for signing completed emails.
SignerCanSignOnMobile	Boolean	When true, signer can use the DocuSign mobile signing user interface.
SignerShowSecureFieldInitialValues	Boolean	When true, the initial value of all SecureFields is written to the document when sent.
SignerAttachCertificateToEnvelopePDF	Boolean	When true, the Certificate of Completion is included in the envelope documents PDF when it is downloaded.
EnableSignOnPaper	Boolean	When true, a user can allow signers to use the sign on paper option.
EnableSignOnPaperOverride	Boolean	When true, a user can override the default account setting for the sign on paper option.
EnableSignerAttachments	Boolean	When true, a user can request attachments from a signer.
UseAccountLevelEmail	Boolean	When true, the content of notification emails is determined at the account level.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetAccountSettingsResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <GetAccountSettingsResult>
        <AccountSettings>
          <UsesAPI>boolean</UsesAPI>
          <EnableDSPro>boolean</EnableDSPro>
          <EnableSendToManage>boolean</EnableSendToManage>
          <EnableSendToAgent>boolean</EnableSendToAgent>
          <EnableVaulting>boolean</EnableVaulting>

<EnableEnvelopeStampingByAccountAdmin>boolean</EnableEnvelopeStampingByAccountAdmin>
          <EnvelopeStampingDefaultValue>boolean</EnvelopeStampingDefaultValue>
          <SignerMustHaveAccount>boolean</SignerMustHaveAccount>
          <SignerMustLoginToSign>boolean</SignerMustLoginToSign>
          <SignerCanCreateAccount>boolean</SignerCanCreateAccount>
          <AllowInPerson>boolean</AllowInPerson>
```

```
          <EnablePowerForm>boolean</EnablePowerForm>
          <AllowSignerReassign>boolean</AllowSignerReassign>
          <EnableReservedDomain>boolean</EnableReservedDomain>
          <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
          <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
          <EnableAutoNav>boolean</EnableAutoNav>
          <AutoNavRule>bytes</AutoNavRule>
          <EnableTransactionPoint>boolean</EnableTransactionPoint>
          <EnvelopeIntegrationAllowed>bytes</EnvelopeIntegrationAllowed>
          <EnvelopeIntegrationEnabled>boolean</EnvelopeIntegrationEnabled>
          <CanSelfBrandSend>boolean</CanSelfBrandSend>
          <CanSelfBrandSign>boolean</CanSelfBrandSign>
          <IDCheckRequired>bytes</IDCheckRequired>
          <IDCheckExpire>bytes</IDCheckExpire>
          <IDCheckExpireDays>integer</IDCheckExpireDays>
          <SignDateFormat>string</SignDateFormat>
          <PKISignDownloadedPDFDocs>bytes</PKISignDownloadedPDFDocs>
          <InPersonIDCheckQuestion>string</InPersonIDCheckQuestion>
          <SessionTimeout>integer</SessionTimeout>
          <AttachCompletedEnvelope>boolean</AttachCompletedEnvelope>
          <SignerCanSignOnMobile>boolean</SignerCanSignOnMobile>

<SignerShowSecureFieldInitialValues>boolean</SignerShowSecureFieldInitialValues>

<SignerAttachCertificateToEnvelopePDF>boolean</SignerAttachCertificateToEnvelopePDF>
          <EnableSignOnPaper>boolean</EnableSignOnPaper>
          <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
          <EnableSignerAttachments>boolean</EnableSignerAttachments>
          <UseAccountLevelEmail>boolean</UseAccountLevelEmail>
        </AccountSettings>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetAccountSettingsResult>
    </GetAccountSettingsResponse>
  </soap:Body>
</soap:Envelope>
```

GetConnectCredentials

This method is used by the DocuSign for Salesforce package to retrieve the Salesforce connect settings.

Schema

Name	Schema Type	Description
AccountId	String	The DocuSign account ID.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetConnectCredentials"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetConnectCredentials xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
    </GetConnectCredentials>
  </soap:Body>
</soap:Envelope>
```

The response contains the information below and either a success or failure. If the call fails an error code is provided.

Name	Schema Type	Description
ConnectUserName	String	The Salesforce login user name.
ConnectConfig	String	The XML configuration.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetConnectCredentialsResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <GetConnectCredentialsResult>
        <ConnectUserName>string</ConnectUserName>
        <ConnectConfig>string</ConnectConfig>
        <Success>boolean</Success>
        <Error>
```

```
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetConnectCredentialsResult>
    </GetConnectCredentialsResponse>
  </soap:Body>
</soap:Envelope>
```

GetEncryptedPassword

This method requests the encrypted form of the user's password. This encrypted password may be used for authentication in other API function calls along with a User Name or User ID and an Integrator Key.

Sample Request XML:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetEncryptedPassword xmlns="http://www.docusign.net/API/AccountManagement" />
  </soap:Body>
</soap:Envelope>
```

The function returns encrypted password for the authenticated user that can be used in successive API calls.

Name	Schema Type	Description
EncryptedPassword	String	The encrypted password for the authenticated user.

The call returns either a success or failure. If the call fails an error code is provided.

Sample Response XML:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetEncryptedPasswordResponse xmlns="http://www.docusign.net/API/AccountManagement">
```

```
      <GetEncryptedPasswordResult>
        <EncryptedPassword>string</EncryptedPassword>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetEncryptedPasswordResult>
    </GetEncryptedPasswordResponse>
  </soap:Body>
</soap:Envelope>
```

Sample Code

GetEncryptedPassword – C#

```
DSAPI_AcctMgmtWebService.GetEncryptedPasswordResponseGetEncryptedPasswordResult result =
            dsAcctMgmtApi.GetEncryptedPassword();
       if (result.Success)
       {
           txtEncryptedPW.Text = result.EncryptedPassword;
           txtResults.Text = "Encrypted Password successfully retrieved.";
           return;
       }
       txtResults.Text = "Encrypted Password request failed: " +
       result.Error.Description;
```

GetMemberSettings

This method requests some attributes about the specified member of the account

Schema

Name	Schema Type	Description
AccountId	String	The account ID associated with the member.
UserId	String	The User ID for the member.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
```

```
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetMemberSettings"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetMemberSettings xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
      <UserId>string</UserId>
    </GetMemberSettings>
  </soap:Body>
</soap:Envelope>
```

The response provides the MemberSettings for the user and either a success or failure. If the call fails an error code is provided. See the MemberSettings section for more information.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetMemberSettingsResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <GetMemberSettingsResult>
        <MemberSettings>
          <CanManageAccount>boolean</CanManageAccount>
          <CanSendEnvelope>boolean</CanSendEnvelope>
          <CanSendAPIRequests>boolean</CanSendAPIRequests>
          <APIAccountWideAccess>boolean</APIAccountWideAccess>
          <EnableVaulting>boolean</EnableVaulting>
          <VaultingMode>bytes</VaultingMode>
          <EnableTransactionPoint>boolean</EnableTransactionPoint>
          <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
          <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
          <EnableDSPro>boolean</EnableDSPro>
          <PowerFormAdmin>boolean</PowerFormAdmin>
          <PowerFormUser>boolean</PowerFormUser>
          <CanEditSharedAddressBook>bytes</CanEditSharedAddressBook>
          <CanManageTemplates>bytes</CanManageTemplates>
          <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
          <EnableSignerAttachments>boolean</EnableSignerAttachments>
        </MemberSettings>
        <Success>boolean</Success>
        <Error>
```

```
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetMemberSettingsResult>
    </GetMemberSettingsResponse>
  </soap:Body>
</soap:Envelope>
```

GetMembershipSummary

This is reserved for DocuSign use and you must have DocuSign Administrator rights to use this method.

This method requests a summary list of information about the accounts and users associated with the supplied email address.

This method uses the UserName/Password/IntegratorKey authentication and requires System Administrator privileges. It takes the argument:

Name	Schema Type	Description
Email	String	The email address for the user whose account and user information is being requested.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetMembershipSummary"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetMembershipSummary xmlns="http://www.docusign.net/API/AccountManagement">
      <Email>string</Email>
```

```
    </GetMembershipSummary>
  </soap:Body>
</soap:Envelope>
```

This method returns a repeating list of membership summaries with the following information:

Name	Schema Type	Description
AccountName	String	The Account Name associated with the requested user.
AccountId	String	The Account ID associated with the requested user.
UserName	String	The name of the user associated with the account.
UserId	String	The ID of the user associated with the account.
UserType	String	Works in combination with UserStatus to show user membership information. There are two possible values for this element: Recipient and CompanyUser. See the table below for a description of the meaning of different UserId and UserStatus combinations.
UserStatus	String	Works in combination with UserType to show user membership information. There are three possible values for this element: Created, Active, and Closed. See the table below for a description of the meaning of different UserId and UserStatus combinations.
PlanName	String	The plan name for the account.
Pgp	String	The plan group plan for the account. The Pgp uniquely identifies a plan/plan group and is used to set plans in other functions.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetMembershipSummaryResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <GetMembershipSummaryResult>
        <MembershipSummaries>
          <MembershipSummary>
            <AccountName>string</AccountName>
            <AccountId>string</AccountId>
            <UserName>string</UserName>
            <UserId>string</UserId>
```

```
            <UserType>CompanyUser or Recipient</UserType>
            <UserStatus>Created or Active or Closed</UserStatus>
            <PlanName>string</PlanName>
            <Pgp>string</Pgp>
          </MembershipSummary>
          <MembershipSummary>
            <AccountName>string</AccountName>
            <AccountId>string</AccountId>
            <UserName>string</UserName>
            <UserId>string</UserId>
            <UserType>CompanyUser or Recipient</UserType>
            <UserStatus>Created or Active or Closed</UserStatus>
            <PlanName>string</PlanName>
            <Pgp>string</Pgp>
          </MembershipSummary>
        </MembershipSummaries>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetMembershipSummaryResult>
    </GetMembershipSummaryResponse>
  </soap:Body>
</soap:Envelope>
```

UserType and UserStatus Combinations

UserType	UserStatus	Description
Recipient	Created	The user has received documents for signing and may have signed them, but does not have a DocuSign account used to log into the console.
Recipient	Active	The user has received documents for signing and has signed up for free DocuSign account. The user has console access to view and sign documents in the console, but cannot send documents.
CompanyUser	Active	The user has a DocuSign account that allows access to view, sign, and send documents. The user also has a billing plan.
CompanyUser	Created	This combination is not used.
Recipeint or	Closed	The user account has been closed. Note that the UserType does

CompanyUser		not matter if the UserStatus is Closed.

GetPlanGroupInformation

This function requests the plan group names and plans associated with each plan group for the given distributor code. This information is used to assign billing plans to an account.

GetPlanGroupInformation takes the arguments:

Name	Schema Type	Description
DistributorCode	String	The Distributor Code that identifies the billing plan groups and plans.
DistributorPassword	String	The Distributor Password for the DistributorCode.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetPlanGroupInformation"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetPlanGroupInformation xmlns="http://www.docusign.net/API/AccountManagement">
      <DistributorCode>string</DistributorCode>
      <DistributorPassword>string</DistributorPassword>
    </GetPlanGroupInformation>
  </soap:Body>
</soap:Envelope>
```

The function returns a list of the plan groups for the distributor code. A distributor code may have multiple plan groups associated with it and each of those plan groups may have multiple plans. The following information is included in the result:

Name	Schema Type	Description
PlanGroupName	String	The name of the billing plan group.
PlanGroupPlans	PlanGroupPlan	A repeating list of the Plan Name and the Plan Group Plan ID for each plan group.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetPlanGroupInformationResponse
xmlns="http://www.docusign.net/API/AccountManagement">
```

```
      <GetPlanGroupInformationResult>
        <PlanGroups>
          <PlanGroup>
            <PlanGroupName>string</PlanGroupName>
            <PlanGroupPlans xsi:nil="true" />
          </PlanGroup>
          <PlanGroup>
            <PlanGroupName>string</PlanGroupName>
            <PlanGroupPlans xsi:nil="true" />
          </PlanGroup>
        </PlanGroups>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetPlanGroupInformationResult>
    </GetPlanGroupInformationResponse>
  </soap:Body>
</soap:Envelope>
```

GetPlanPricingInformation

This function requests the plan pricing information associated with the given distributor code.

GetPlanPricingInformation takes the arguments:

Name	Schema Type	Description
DistributorCode	String	The Distributor Code that identifies the billing plan groups and plans.
DistributorPassword	String	The Distributor Password for the DistributorCode.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetPlanPricingInformation"

<?xml version="1.0" encoding="utf-8"?>
```

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetPlanPricingInformation xmlns="http://www.docusign.net/API/AccountManagement">
      <DistributorCode>string</DistributorCode>
      <DistributorPassword>string</DistributorPassword>
      <Pgp>string</Pgp>
    </GetPlanPricingInformation>
  </soap:Body>
</soap:Envelope>
```

The function returns the plan pricing information for the distributor code. The following information is included in the result:

Name	Schema Type	Description
PlanName	String	The name of the plan.
PaymentCycle		The payment cycle associated with the plan. The possible values are: Monthly or Annually.
PaymentMethod		The payment method used with the plan. The possible values are: CreditCard, PurchaseOrder, Premium, or Freemium.
PerSeatPrice	Decimal	The per seat price for the plan.
OtherDiscountPercent	Decimal	Any other percentage discount for the plan.
SupportIncidentFee	Decimal	The support incident fee charged for each support incident.
SupportPlanFee	Decimal	The support plan fee charged for this plan.
IncludedSeats	Integer	The number of seats included in the plan.
EnableSupport	Boolean	If “true” the plan has support enabled.
Pgp	String	The plan group plan for the account. The Pgp uniquely identifies a plan/plan group and is used to set plans in other functions.
SeatDiscounts	SeatDiscount	A complex type that return any seat discount information. It contains the information: BeginSeatCount, EndSeatCount and SeatDiscountPercent.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetPlanPricingInformationResponse
xmlns="http://www.docusign.net/API/AccountManagement">
      <GetPlanPricingInformationResult>
        <Plan>
```

```
          <PlanName>string</PlanName>
          <PaymentCycle>Monthly or Annually</PaymentCycle>
          <PaymentMethod>CreditCard or PurchaseOrder or Premium or
Freemium</PaymentMethod>
          <PerSeatPrice>decimal</PerSeatPrice>
          <OtherDiscountPercent>decimal</OtherDiscountPercent>
          <SupportIncidentFee>decimal</SupportIncidentFee>
          <SupportPlanFee>decimal</SupportPlanFee>
          <IncludedSeats>integer</IncludedSeats>
          <EnableSupport>boolean</EnableSupport>
          <PgpId>string</PgpId>
          <SeatDiscounts>
            <SeatDiscount xsi:nil="true" />
            <SeatDiscount xsi:nil="true" />
          </SeatDiscounts>
        </Plan>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetPlanPricingInformationResult>
    </GetPlanPricingInformationResponse>
  </soap:Body>
</soap:Envelope>
```

GetPlanType

This function requests the plan payment type information for an account. The payment types use the same terminology as the payment method.

GetPlanType takes the argument:

Name	Schema Type	Description
AccountId	String	The account ID for the account requested.

```
Sample Request XML
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetPlanType"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetPlanType xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
    </GetPlanType>
  </soap:Body>
</soap:Envelope>
```

The function returns the account plan payment type (the PaymentMethod value on the Plan table for the PlanId that is associated with the specified AccountId) and either a success or failure. If the call fails an error code is provided. The following information included in the result:

Name	Schema Type	Description
LogicalPlanType	String	The payment method used with the plan. The possible values are: CreditCard, PurchaseOrder, or Freemium.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetPlanTypeResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <GetPlanTypeResult>
        <LogicalPlanType>string</LogicalPlanType>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetPlanTypeResult>
    </GetPlanTypeResponse>
  </soap:Body>
```

```
</soap:Envelope>
```

GetProvisioningInformation

This function requests account provisioning information, the DistributorCode, DistributorPassword, and other plan information, for the given Token.

GetProvisioningInformation takes the arguments:

Name	Schema Type	Description
AppToken	String	The token associated with the account provisioning information. This is provided by the group provisioning the account.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetProvisioningInformation"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetProvisioningInformation xmlns="http://www.docusign.net/API/AccountManagement">
      <AppToken>string</AppToken>
    </GetProvisioningInformation>
  </soap:Body>
</soap:Envelope>
```

The function returns account provisioning information, the DistributorCode, DistributorPassword, and other plan information. The following information included in the result:

Name	Schema Type	Description
DistributorCode	String	The Distributor Code that identifies the billing plan groups and plans.
DistributorPassword	String	The Distributor Password for the DistributorCode.
Pgp	String	The plan group plan for the account. The Pgp uniquely identifies a plan/plan group and is used to set plans in other functions.
DefaultConnectConfigurationId	Integer	Reserved.
PasswordRuleText	String	A text string describing the rules for setting an account password.
PurchaseOrderOrPromoAllowed	Boolean	Shows if a Purchase Order or Promotion is associated with the account.
Success	Boolean	This is True if the API succeeded. If False the ErrorCode and Description fields will contain the error information.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetProvisioningInformationResponse
xmlns="http://www.docusign.net/API/AccountManagement">
      <GetProvisioningInformationResult>
        <DistributorCode>string</DistributorCode>
        <DistributorPassword>string</DistributorPassword>
        <Pgp>string</Pgp>
        <PlanPromoText>string</PlanPromoText>
        <DefaultConnectConfigurationId>integer</DefaultConnectConfigurationId>
        <PasswordRuleText>string</PasswordRuleText>
        <PurchaseOrderOrPromoAllowed>boolean</PurchaseOrderOrPromoAllowed>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetProvisioningInformationResult>
    </GetProvisioningInformationResponse>
  </soap:Body>
</soap:Envelope>
```

GetSuccessorPlanInformation

This function requests the current plan information and a list of successor plans, plans that the current plan could be rolled into, associated with the given distributor code and account.

GetSuccessorPlanInformation takes the arguments:

Name	Schema Type	Description
DistributorCode	String	The Distributor Code that identifies the billing plan groups and plans.

Name	Schema Type	Description
DistributorPassword	String	The Distributor Password for the DistributorCode.
AccountId	String	The Account ID for the account for which information is being requested.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetSuccessorPlanInformation"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetSuccessorPlanInformation xmlns="http://www.docusign.net/API/AccountManagement">
      <DistributorCode>string</DistributorCode>
      <DistributorPassword>string</DistributorPassword>
      <AccountId>string</AccountId>
    </GetSuccessorPlanInformation>
  </soap:Body>
</soap:Envelope>
```

The function returns the current plan information and a list of successor plans for the distributor code and account. The following information included in the result:

Name	Schema Type	Description
CurrentPlan	CurrentPlan	A complex type that returns the pricing plan information for the current plan. The information in this complex type is the same as the return from the GetPricingPlanInformation call. See the return information above for more information.
SuccessorPlan	SucessorPlan	The list of SuccessorPlans. One of the plans in the list could be selected to succeed the current plan. The SuccessorPlan information in this complex type is the same as the return from the GetPricingPlanInformation call. See the return information above for more information.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
```

```
    <GetSuccessorPlanInformationResponse
xmlns="http://www.docusign.net/API/AccountManagement">
      <GetSuccessorPlanInformationResult>
        <CurrentPlan>
          <PlanName>string</PlanName>
          <PaymentCycle>Monthly or Annually</PaymentCycle>
          <PaymentMethod>CreditCard or PurchaseOrder or Premium or
Freemium</PaymentMethod>
          <PerSeatPrice>decimal</PerSeatPrice>
          <OtherDiscountPercent>decimal</OtherDiscountPercent>
          <SupportIncidentFee>decimal</SupportIncidentFee>
          <SupportPlanFee>decimal</SupportPlanFee>
          <IncludedSeats>integer</IncludedSeats>
          <EnableSupport>boolean</EnableSupport>
          <PgpId>string</PgpId>
          <SeatDiscounts>
            <SeatDiscount xsi:nil="true" />
            <SeatDiscount xsi:nil="true" />
          </SeatDiscounts>
        </CurrentPlan>
        <SuccessorPlans>
          <SuccessorPlan>
            <PlanName>string</PlanName>
            <PaymentCycle>Monthly or Annually</PaymentCycle>
            <PaymentMethod>CreditCard or PurchaseOrder or Premium or
Freemium</PaymentMethod>
            <PerSeatPrice>decimal</PerSeatPrice>
            <OtherDiscountPercent>decimal</OtherDiscountPercent>
            <SupportIncidentFee>decimal</SupportIncidentFee>
            <SupportPlanFee>decimal</SupportPlanFee>
            <IncludedSeats>integer</IncludedSeats>
            <EnableSupport>boolean</EnableSupport>
            <PgpId>string</PgpId>
            <SeatDiscounts xsi:nil="true" />
          </SuccessorPlan>
          <SuccessorPlan>
            <PlanName>string</PlanName>
            <PaymentCycle>Monthly or Annually</PaymentCycle>
            <PaymentMethod>CreditCard or PurchaseOrder or Premium or
Freemium</PaymentMethod>
            <PerSeatPrice>decimal</PerSeatPrice>
            <OtherDiscountPercent>decimal</OtherDiscountPercent>
            <SupportIncidentFee>decimal</SupportIncidentFee>
            <SupportPlanFee>decimal</SupportPlanFee>
            <IncludedSeats>integer</IncludedSeats>
            <EnableSupport>boolean</EnableSupport>
            <PgpId>string</PgpId>
            <SeatDiscounts xsi:nil="true" />
          </SuccessorPlan>
        </SuccessorPlans>
        <Success>boolean</Success>
        <Error>
```

```
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetSuccessorPlanInformationResult>
    </GetSuccessorPlanInformationResponse>
  </soap:Body>
</soap:Envelope>
```

GetUserProfile

This method retrieves the privacy settings and personal information (address, phone number, etc.) from a user ID card.

GetUserProfile takes the argument:

Name	Schema Type	Description
UserId	String	The User ID for the user for which you want to retrieve information.
AccountId	String	The Account ID for the account associated with the user.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetUserProfile"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetUserProfile xmlns="http://www.docusign.net/API/AccountManagement">
      <UserId>string</UserId>
```

```
      <AccountId>string</AccountId>
    </GetUserProfile>
  </soap:Body>
</soap:Envelope>
```

The response returns the user profile information from the user's ID card. The User Profile contains the following information.

Name	Schema Type	Description
AccountId	String	The Account ID for the account associated with the user.
UserId	String	User ID for the user.
ShowDocuSignID	Boolean	When true, the user's ID card can be viewed from signed documents and envelope history.
DisplayCompanyAndTitle	Boolean	When true, the user's company and title information are shown on the ID card.
DisplayAddressAndPhone	Boolean	When true, the user's Address and Phone number are shown on the ID card.
DisplayUsageHistory	Boolean	When true, the user's usage information is shown on the ID card.
Company	String	The user's Company information.
Title	String	The user's Title information.
Address	Address	A complex element consisting of the optional elements: • Address1 • Address2 • City • State • Zip • Phone • Fax • Country
AuthenticationHistory	AuthenticationMethod	Shows the authentication methods used by the user.
UsageHistory	Usage History	A complex element consisting of: • SignedCount – the number of enevelopes the user has signed. • LastSignedTimestamp – the dateand time the user last signed an envelope. • SentCount – the number of envelopes the user has sent. • LastSentTimestamp – the date and time the user last sent an envelope.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetUserProfileResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <GetUserProfileResult>
        <UserProfile>
          <AccountId>string</AccountId>
          <UserId>string</UserId>
          <ShowDocuSignID>boolean</ShowDocuSignID>
          <DisplayCompanyAndTitle>boolean</DisplayCompanyAndTitle>
          <DisplayAddressAndPhone>boolean</DisplayAddressAndPhone>
          <DisplayUsageHistory>boolean</DisplayUsageHistory>
          <Company>string</Company>
          <Title>string</Title>
          <Address>
            <Address1>string</Address1>
            <Address2>string</Address2>
            <City>string</City>
            <State>string</State>
            <Zip>string</Zip>
            <Phone>string</Phone>
            <Fax>string</Fax>
            <Country>string</Country>
          </Address>
          <AuthenticationHistory>
            <AuthenticationMethod xsi:nil="true" />
            <AuthenticationMethod xsi:nil="true" />
          </AuthenticationHistory>
          <UsageHistory>
            <SignedCount>int</SignedCount>
            <LastSignedTimestamp>dateTime</LastSignedTimestamp>
            <SentCount>int</SentCount>
            <LastSentTimestamp>dateTime</LastSentTimestamp>
          </UsageHistory>
        </UserProfile>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed or
Hourly_API_Invocation_Limit_Exceeded or Initialize_Client_Account_Error or
Max_Members_Exceeded or Invalid_File_Or_Too_Large</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetUserProfileResult>
```

```
    </GetUserProfileResponse>
  </soap:Body>
</soap:Envelope>
```

GetUserProfileImage

This method retrieves the image from a user ID card.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/GetUserProfileImage"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetUserProfileImage xmlns="http://www.docusign.net/API/AccountManagement" />
  </soap:Body>
</soap:Envelope>
```

The request returns the image file from the user ID card

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetUserProfileImageResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <GetUserProfileImageResult>
        <UserProfileImage>
          <Image>
            <MimeType>string</MimeType>
            <ImageData>base64Binary</ImageData>
          </Image>
        </UserProfileImage>
        <Success>boolean</Success>
        <Error>
```

```
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed or
Hourly_API_Invocation_Limit_Exceeded or Initialize_Client_Account_Error or
Max_Members_Exceeded or Invalid_File_Or_Too_Large</ErrorCode>
          <Description>string</Description>
        </Error>
      </GetUserProfileImageResult>
    </GetUserProfileImageResponse>
  </soap:Body>
</soap:Envelope>
```

NewAccount

This method creates a new DocuSign account with a single member, who is the account owner.

Name	Schema Type	Description
AccountName	String	The account name for the new account.
DistributorCode	String	The Distributor Code that identifies the billing plan groups and plans for the new account.
DistributorPassword	String	The Distributor Password for the DistributorCode.
Pgp	String	The plan group plan for the account. The Pgp uniquely identifies a plan/plan group and is used to set plans in other functions.
CreditCardInformation	CreditCardInformation	A complex type that has information about the credit card used to pay for this account. It included the elements: ccNumber, ccExpirationMonth, ccExpirationYear, ccUserName, and ccType.
ReferralInformation	ReferralInformation	A complex type that contains the following information: EnableSupport, IncludedSeats, ReferrerName, ReferralCode, AdvertisementID, PublisherID, ShopperID, PromoCode, GroupMemberID, IdType, and Industry.
AccountSettings	AccountSettings	A complex type with the account settings information for the new account. See the AccountSettings section for more information

Name	Schema Type	Description
Member	Member	A complex type with the member information for the new account. See Member section for more information.
AddressInformation	AddressInformation	A complex type that contains the following information for the new account: Street1, Street2, City, State, Zip, Phone, and Fax.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/NewAccount"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <NewAccount xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountName>string</AccountName>
      <DistributorCode>string</DistributorCode>
      <DistributorPassword>string</DistributorPassword>
      <Pgp>string</Pgp>
      <CreditCardInformation>
        <ccNumber>string</ccNumber>
        <ccExpirationMonth>string</ccExpirationMonth>
        <ccExpirationYear>string</ccExpirationYear>
        <ccUserName>string</ccUserName>
        <ccType>string</ccType>
      </CreditCardInformation>
      <ReferralInformation>
        <EnableSupport>boolean</EnableSupport>
        <IncludedSeats>positiveInteger</IncludedSeats>
        <SaleDiscountPercent>decimal</SaleDiscountPercent>
        <PlanStartMonth>positiveInteger</PlanStartMonth>
        <ReferrerName>string</ReferrerName>
        <ReferralCode>string</ReferralCode>
        <AdvertisementID>string</AdvertisementID>
        <PublisherID>string</PublisherID>
        <ShopperID>string</ShopperID>
        <PromoCode>string</PromoCode>
        <GroupMemberID>string</GroupMemberID>
        <IdType>string</IdType>
        <Industry>string</Industry>
      </ReferralInformation>
      <AccountSettings>
        <UsesAPI>boolean</UsesAPI>
        <EnableDSPro>boolean</EnableDSPro>
        <EnableVaulting>boolean</EnableVaulting>

<EnableEnvelopeStampingByAccountAdmin>boolean</EnableEnvelopeStampingByAccountAdmin>
        <EnvelopeStampingDefaultValue>boolean</EnvelopeStampingDefaultValue>
        <SignerMustHaveAccount>boolean</SignerMustHaveAccount>
        <SignerMustLoginToSign>boolean</SignerMustLoginToSign>
        <SignerCanCreateAccount>boolean</SignerCanCreateAccount>
        <AllowInPerson>boolean</AllowInPerson>
        <EnablePowerForm>boolean</EnablePowerForm>
```

```
        <AllowSignerReassign>boolean</AllowSignerReassign>
        <EnableReservedDomain>boolean</EnableReservedDomain>
        <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
        <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
        <EnableAutoNav>boolean</EnableAutoNav>
        <AutoNavRule>bytes</AutoNavRule>
        <EnableTransactionPoint>boolean</EnableTransactionPoint>
        <EnvelopeIntegrationAllowed>bytes</EnvelopeIntegrationAllowed>
        <EnvelopeIntegrationEnabled>boolean</EnvelopeIntegrationEnabled>
        <CanSelfBrandSend>boolean</CanSelfBrandSend>
        <CanSelfBrandSign>boolean</CanSelfBrandSign>
        <IDCheckRequired>bytes</IDCheckRequired>
        <IDCheckExpire>bytes</IDCheckExpire>
        <IDCheckExpireDays>integer</IDCheckExpireDays>
        <SignDateFormat>string</SignDateFormat>
        <PKISignDownloadedPDFDocs>bytes</PKISignDownloadedPDFDocs>
        <InPersonIDCheckQuestion>string</InPersonIDCheckQuestion>
        <SessionTimeout>integer</SessionTimeout>
        <AttachCompletedEnvelope>boolean</AttachCompletedEnvelope>
        <SignerCanSignOnMobile>boolean</SignerCanSignOnMobile>
        <SignerShowSecureFieldInitialValues>boolean</SignerShowSecureFieldInitialValues>

<SignerAttachCertificateToEnvelopePDF>boolean</SignerAttachCertificateToEnvelopePDF>
        <EnableSignOnPaper>boolean</EnableSignOnPaper>
        <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
        <EnableSignerAttachments>boolean</EnableSignerAttachments>
        <UseAccountLevelEmail>boolean</UseAccountLevelEmail>
      </AccountSettings>
      <Member>
        <MemberEmailAddress>string</MemberEmailAddress>
        <MemberUserName>string</MemberUserName>
        <MemberPassword>string</MemberPassword>
        <MemberForgottenPasswordQuestion>string</MemberForgottenPasswordQuestion>
        <MemberForgottenPasswordAnswer>string</MemberForgottenPasswordAnswer>
        <MemberTitle>string</MemberTitle>
        <MemberFirstName>string</MemberFirstName>
        <MemberMiddleName>string</MemberMiddleName>
        <MemberLastName>string</MemberLastName>
        <MemberSuffix>string</MemberSuffix>
        <EnableConnectForUser>boolean</EnableConnectForUser>
        <MemberSettings>
          <CanManageAccount>boolean</CanManageAccount>
          <CanSendEnvelope>boolean</CanSendEnvelope>
          <CanSendAPIRequests>boolean</CanSendAPIRequests>
          <APIAccountWideAccess>boolean</APIAccountWideAccess>
          <EnableVaulting>boolean</EnableVaulting>
          <VaultingMode>bytes</VaultingMode>
          <EnableTransactionPoint>boolean</EnableTransactionPoint>
          <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
          <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
          <EnableDSPro>boolean</EnableDSPro>
          <PowerFormAdmin>boolean</PowerFormAdmin>
          <PowerFormUser>boolean</PowerFormUser>
          <CanEditSharedAddressBook>bytes</CanEditSharedAddressBook>
          <CanManageTemplates>bytes</CanManageTemplates>
          <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
          <EnableSignerAttachments>boolean</EnableSignerAttachments>
        </MemberSettings>
        <ReturnEncryptedPassword>boolean</ReturnEncryptedPassword>
      </Member>
      <AddressInformation>
        <Address1>string</Address1>
        <Address2>string</Address2>
        <City>string</City>
```

```
        <State>string</State>
        <Zip>string</Zip>
        <Phone>string</Phone>
        <Fax>string</Fax>
      </AddressInformation>
    </NewAccount>
  </soap:Body>
</soap:Envelope>
```

The response contains the information below and either a success or failure. If the call fails an error code is provided.

Name	Schema Type	Description
AccountId	String	The DocuSign account ID for the new account.
SiteId	String	The Site ID for the new account.
UserId	String	The User ID for the new account.
MembershipId	String	The membership ID for the new account.
EncryptedPassword	String	The encrypted password for the new account. .

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <NewAccountResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <NewAccountResult>
        <AccountId>string</AccountId>
        <SiteId>string</SiteId>
        <UserId>string</UserId>
        <MembershipId>string</MembershipId>
        <EncryptedPassword>string</EncryptedPassword>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
```

```
          <Description>string</Description>
        </Error>
      </NewAccountResult>
    </NewAccountResponse>
  </soap:Body>
</soap:Envelope>
```

Ping

This method determines if the AccountManagement API service can be contacted.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/Ping"
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Ping xmlns="http://www.docusign.net/API/AccountManagement" />
  </soap:Body>
</soap:Envelope>
```

This method simply returns “true” in the result XML if the calling application was able to reach it.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <PingResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <PingResult>boolean</PingResult>
    </PingResponse>
  </soap:Body>
</soap:Envelope>
```

ResendAccountActivation

This method sends another account activation message to an account user.

Name	Schema Type	Description
DistributorCode	String	The Distributor Code that identifies the billing plan groups and plans.
DistributorPassword	String	The Distributor Password for the DistributorCode.
MemberEmailAddress	String	The member’s email address for the associated account.

Name	Schema Type	Description
MemberUserName	String	The member's name for the associated account.
MemberPassword	String	The member's password for the associated account.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/ResendAccountActivation"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ResendAccountActivation xmlns="http://www.docusign.net/API/AccountManagement">
      <DistributorCode>string</DistributorCode>
      <DistributorPassword>string</DistributorPassword>
      <MemberEmailAddress>string</MemberEmailAddress>
      <MemberUserName>string</MemberUserName>
      <MemberPassword>string</MemberPassword>
    </ResendAccountActivation>
  </soap:Body>
</soap:Envelope>
```

The response returns a success or failure result

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ResendAccountActivationResponse
xmlns="http://www.docusign.net/API/AccountManagement">
      <ResendAccountActivationResult>
        <Success>boolean</Success>
        <Error>
```

```
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed or
Hourly_API_Invocation_Limit_Exceeded or Initialize_Client_Account_Error or
Max_Members_Exceeded or Invalid_File_Or_Too_Large</ErrorCode>
          <Description>string</Description>
        </Error>
      </ResendAccountActivationResult>
    </ResendAccountActivationResponse>
  </soap:Body>
</soap:Envelope>
```

SetConnectCredentials

This method is used by the DocuSign for Salesforce package to set the connection used by DocuSign to update Salesforce data.

Name	Schema Type	Description
AccountId	String	The DocuSign account ID.
ConnectUsername	String	The Salesforce login user name.
ConnectPassword	String	The Salesforce login password.
ConnectConfigurationId	Int	The default XML used when setting up Connect for Salesforce.
SalesforceEnvironment	String	The Salesforce environment being accessed (i.e: Production or Test).

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/SetConnectCredentials"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SetConnectCredentials xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
```

```
      <ConnectUsername>string</ConnectUsername>
      <ConnectPassword>string</ConnectPassword>
      <ConnectConfigurationId>int</ConnectConfigurationId>
      <SalesforceEnvironment>string</SalesforceEnvironment>
    </SetConnectCredentials>
  </soap:Body>
</soap:Envelope>
```

This method returns a ConnectTestResult string and either a success or failure. If the call fails an error code is provided.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SetConnectCredentialsResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <SetConnectCredentialsResult>
        <ConnectTestResult>string</ConnectTestResult>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </SetConnectCredentialsResult>
    </SetConnectCredentialsResponse>
  </soap:Body>
</soap:Envelope>
```

SetUserProfile

This method is used to set the privacy settings and personal information (address, phone number, etc.) from a user ID card.

Name	Schema Type	Description

Name	Schema Type	Description
AccountId	String	The Account ID for the account associated with the user.
UserId	String	User ID for the user.
ShowDocuSignID	Boolean	When true, the user's ID card can be viewed from signed documents and envelope history.
DisplayCompanyAndTitle	Boolean	When true, the user's company and title information are shown on the ID card.
DisplayAddressAndPhone	Boolean	When true, the user's Address and Phone number are shown on the ID card.
DisplayUsageHistory	Boolean	When true, the user's usage information is shown on the ID card.
Company	String	The user's Company information.
Title	String	The user's Title information.
Address	Address	A complex element consisting of the optional elements: • Address1 • Address2 • City • State • Zip • Phone • Fax • Country
AuthenticationHistory	AuthenticationMethod	A complex element consisting of: • AuthenticationMethodType (PhoneAuth, STAN, ISCheck, OFAC, AccessCode, AgeVerify, or SSOAuth) • TotalCount – the number of times used. • LastTimestamp – the data and time the user last used the authentication method. • LastProvider – the last provider that authenticated the user.
UsageHistory	Usage History	A complex element consisting of: • SignedCount – the number of enevelopes the user has signed. • LastSignedTimestamp – the dateand time the user last signed an envelope. • SentCount – the number of envelopes the user has sent. • LastSentTimestamp – the date and time the user last sent an envelope.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
```

```
SOAPAction: "http://www.docusign.net/API/AccountManagement/SetUserProfile"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SetUserProfile xmlns="http://www.docusign.net/API/AccountManagement">
      <UserProfile>
        <AccountId>string</AccountId>
        <UserId>string</UserId>
        <ShowDocuSignID>boolean</ShowDocuSignID>
        <DisplayCompanyAndTitle>boolean</DisplayCompanyAndTitle>
        <DisplayAddressAndPhone>boolean</DisplayAddressAndPhone>
        <DisplayUsageHistory>boolean</DisplayUsageHistory>
        <Company>string</Company>
        <Title>string</Title>
        <Address>
          <Address1>string</Address1>
          <Address2>string</Address2>
          <City>string</City>
          <State>string</State>
          <Zip>string</Zip>
          <Phone>string</Phone>
          <Fax>string</Fax>
          <Country>string</Country>
        </Address>
        <AuthenticationHistory>
          <AuthenticationMethod Type="PhoneAuth or STAN or IDCheck or OFAC or AccessCode
or AgeVerify or SSOAuth">
            <TotalCount>int</TotalCount>
            <LastTimestamp>dateTime</LastTimestamp>
            <LastProvider>string</LastProvider>
          </AuthenticationMethod>
          <AuthenticationMethod Type="PhoneAuth or STAN or IDCheck or OFAC or AccessCode
or AgeVerify or SSOAuth">
            <TotalCount>int</TotalCount>
            <LastTimestamp>dateTime</LastTimestamp>
            <LastProvider>string</LastProvider>
          </AuthenticationMethod>
        </AuthenticationHistory>
        <UsageHistory>
          <SignedCount>int</SignedCount>
          <LastSignedTimestamp>dateTime</LastSignedTimestamp>
          <SentCount>int</SentCount>
          <LastSentTimestamp>dateTime</LastSentTimestamp>
        </UsageHistory>
      </UserProfile>
    </SetUserProfile>
  </soap:Body>
</soap:Envelope>
```

The response returns the user Membership ID and the success of failure of the information post.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
```

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SetUserProfileResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <SetUserProfileResult>
        <MembershipId>string</MembershipId>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed or
Hourly_API_Invocation_Limit_Exceeded or Initialize_Client_Account_Error or
Max_Members_Exceeded or Invalid_File_Or_Too_Large</ErrorCode>
          <Description>string</Description>
        </Error>
      </SetUserProfileResult>
    </SetUserProfileResponse>
  </soap:Body>
</soap:Envelope>
```

SetUserProfileImage

Thsis method is used to upload an image file to a user ID card.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: test.docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/SetUserProfileImage"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SetUserProfileImage xmlns="http://www.docusign.net/API/AccountManagement">
      <UserProfileImage>
        <Image>
          <MimeType>string</MimeType>
          <ImageData>base64Binary</ImageData>
        </Image>
      </UserProfileImage>
    </SetUserProfileImage>
```

```
  </soap:Body>
</soap:Envelope>
```

The response returns the user ID for the user and a success or failure of the post.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SetUserProfileImageResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <SetUserProfileImageResult>
        <UserId>string</UserId>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed or
Hourly_API_Invocation_Limit_Exceeded or Initialize_Client_Account_Error or
Max_Members_Exceeded or Invalid_File_Or_Too_Large</ErrorCode>
          <Description>string</Description>
        </Error>
      </SetUserProfileImageResult>
    </SetUserProfileImageResponse>
  </soap:Body>
</soap:Envelope>
```

UpdateAccountSettings

This method is used to modify account level settings.

Name	Schema Type	Description
AccountId	String	The account ID for the account to be modified.
AccountSettings	AccountSettings	A complex type with the account settings information being modified. See the AccountSettings section for more information

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/UpdateAccountSettings"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <UpdateAccountSettings xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
      <AccountSettings>
        <UsesAPI>boolean</UsesAPI>
        <EnableDSPro>boolean</EnableDSPro>
        <EnableSendToManage>boolean</EnableSendToManage>
        <EnableSendToAgent>boolean</EnableSendToAgent>
        <EnableVaulting>boolean</EnableVaulting>

<EnableEnvelopeStampingByAccountAdmin>boolean</EnableEnvelopeStampingByAccountAdmin>
        <EnvelopeStampingDefaultValue>boolean</EnvelopeStampingDefaultValue>
        <SignerMustHaveAccount>boolean</SignerMustHaveAccount>
        <SignerMustLoginToSign>boolean</SignerMustLoginToSign>
        <SignerCanCreateAccount>boolean</SignerCanCreateAccount>
        <AllowInPerson>boolean</AllowInPerson>
        <EnablePowerForm>boolean</EnablePowerForm>
        <AllowSignerReassign>boolean</AllowSignerReassign>
        <EnableReservedDomain>boolean</EnableReservedDomain>
        <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
        <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
        <EnableAutoNav>boolean</EnableAutoNav>
        <AutoNavRule>bytes</AutoNavRule>
        <EnableTransactionPoint>boolean</EnableTransactionPoint>
        <EnvelopeIntegrationAllowed>bytes</EnvelopeIntegrationAllowed>
        <EnvelopeIntegrationEnabled>boolean</EnvelopeIntegrationEnabled>
        <CanSelfBrandSend>boolean</CanSelfBrandSend>
        <CanSelfBrandSign>boolean</CanSelfBrandSign>
        <IDCheckRequired>bytes</IDCheckRequired>
        <IDCheckExpire>bytes</IDCheckExpire>
        <IDCheckExpireDays>integer</IDCheckExpireDays>
        <SignDateFormat>string</SignDateFormat>
        <PKISignDownloadedPDFDocs>bytes</PKISignDownloadedPDFDocs>
        <InPersonIDCheckQuestion>string</InPersonIDCheckQuestion>
        <SessionTimeout>integer</SessionTimeout>
        <AttachCompletedEnvelope>boolean</AttachCompletedEnvelope>
        <SignerCanSignOnMobile>boolean</SignerCanSignOnMobile>
        <SignerShowSecureFieldInitialValues>boolean</SignerShowSecureFieldInitialValues>

<SignerAttachCertificateToEnvelopePDF>boolean</SignerAttachCertificateToEnvelopePDF>
        <EnableSignOnPaper>boolean</EnableSignOnPaper>
        <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
        <EnableSignerAttachments>boolean</EnableSignerAttachments>
        <UseAccountLevelEmail>boolean</UseAccountLevelEmail>
      </AccountSettings>
    </UpdateAccountSettings>
  </soap:Body>
</soap:Envelope>
```

The method returns either a success or failure. If the call fails an error code is provided.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <UpdateAccountSettingsResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <UpdateAccountSettingsResult>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </UpdateAccountSettingsResult>
    </UpdateAccountSettingsResponse>
  </soap:Body>
</soap:Envelope>
```

UpdateMemberSettings

This method is used to modify settings for a member of an account

Name	Schema Type	Description
AccountId	String	The account ID associated with the member.
UserId	String	The User ID of the user that is being modified.
MemberSettings	MemberSettings	A complex type with the member settings that are being modified. See the MemberSettings section for more information.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/UpdateMemberSettings"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <UpdateMemberSettings xmlns="http://www.docusign.net/API/AccountManagement">
      <AccountId>string</AccountId>
      <UserId>string</UserId>
      <MemberSettings>
        <CanManageAccount>boolean</CanManageAccount>
        <CanSendEnvelope>boolean</CanSendEnvelope>
        <CanSendAPIRequests>boolean</CanSendAPIRequests>
        <APIAccountWideAccess>boolean</APIAccountWideAccess>
        <EnableVaulting>boolean</EnableVaulting>
        <VaultingMode>bytes</VaultingMode>
        <EnableTransactionPoint>boolean</EnableTransactionPoint>
        <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
        <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
        <EnableDSPro>boolean</EnableDSPro>
        <PowerFormAdmin>boolean</PowerFormAdmin>
        <PowerFormUser>boolean</PowerFormUser>
        <CanEditSharedAddressBook>bytes</CanEditSharedAddressBook>
        <CanManageTemplates>bytes</CanManageTemplates>
        <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
        <EnableSignerAttachments>boolean</EnableSignerAttachments>
      </MemberSettings>
    </UpdateMemberSettings>
  </soap:Body>
</soap:Envelope>
```

The method returns either a success or failure. If the call fails an error code is provided.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <UpdateMemberSettingsResponse xmlns="http://www.docusign.net/API/AccountManagement">
      <UpdateMemberSettingsResult>
        <Success>boolean</Success>
        <Error>
```

```
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </UpdateMemberSettingsResult>
    </UpdateMemberSettingsResponse>
  </soap:Body>
</soap:Envelope>
```

UpgradeRecipientAccount

This function is used to upgrade a user from a free account, where the user has few privileges, to a paid account plan. The function changes the UserType/UserStatus of the account from Recipient/Active to CompanyUser/Active.

This function uses the UserName/Password/IntegratorKey authentication and requires System Administrator or Account Administrator privileges. It takes the arguments:

Name	Schema Type	Description
DistributorCode	String	The Distributor Code that identifies the billing plan groups and plans.
DistributorPassword	String	The Distributor Password for the DistributorCode.
AccountId	String	The Account ID associated with the user.
UserId	String	The ID of the user associated with the account.
Pgp	String	The plan group plan for the user.
CreditCardInformation	CreditCardInformation	This complex type has information about the credit card used to pay for this account. It included the elements: ccNumber, ccExpirationMonth, ccExpirationYear, ccUserName, and ccType.

Name	Schema Type	Description
ReferralInformation	ReferralInformation	This complex type contains the following information: EnableSupport, IncludedSeats, ReferrerName, ReferralCode, AdvertisementID, PublisherID, ShopperID, PromoCode, GroupMemberID, IdType, and Industry.
AccountSettings	AccountSettings	This complex type contains the following information: attachCompletedEnvelope, envelopeIntegrationAllowed, envelopeIntegrationEnabled and UseAccountLevelEmail. Note, the complete AccountSettings structure exists for this, but only the above fields are used within this API.
AddressInformation	AddressInformation	This complex type contains the following information: Street1, Street2, City, State, Zip, Phone, and Fax.

Sample Request XML

```
POST /api/3.0/accountmanagement.asmx HTTP/1.1
Host: docusign.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.docusign.net/API/AccountManagement/UpgradeRecipientAccount"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <UpgradeRecipientAccount xmlns="http://www.docusign.net/API/AccountManagement">
      <DistributorCode>string</DistributorCode>
      <DistributorPassword>string</DistributorPassword>
      <AccountId>string</AccountId>
      <UserId>string</UserId>
      <Pgp>string</Pgp>
      <CreditCardInformation>
        <ccNumber>string</ccNumber>
        <ccExpirationMonth>string</ccExpirationMonth>
        <ccExpirationYear>string</ccExpirationYear>
        <ccUserName>string</ccUserName>
        <ccType>string</ccType>
      </CreditCardInformation>
      <ReferralInformation>
        <EnableSupport>boolean</EnableSupport>
        <IncludedSeats>positiveInteger</IncludedSeats>
        <SaleDiscountPercent>decimal</SaleDiscountPercent>
        <PlanStartMonth>positiveInteger</PlanStartMonth>
        <ReferrerName>string</ReferrerName>
        <ReferralCode>string</ReferralCode>
        <AdvertisementID>string</AdvertisementID>
        <PublisherID>string</PublisherID>
        <ShopperID>string</ShopperID>
        <PromoCode>string</PromoCode>
        <GroupMemberID>string</GroupMemberID>
        <IdType>string</IdType>
```

```
        <Industry>string</Industry>
      </ReferralInformation>
      <AccountSettings>
        <UsesAPI>boolean</UsesAPI>
        <EnableDSPro>boolean</EnableDSPro>
        <EnableSendToManage>boolean</EnableSendToManage>
        <EnableSendToAgent>boolean</EnableSendToAgent>
        <EnableVaulting>boolean</EnableVaulting>

<EnableEnvelopeStampingByAccountAdmin>boolean</EnableEnvelopeStampingByAccountAdmin>
        <EnvelopeStampingDefaultValue>boolean</EnvelopeStampingDefaultValue>
        <SignerMustHaveAccount>boolean</SignerMustHaveAccount>
        <SignerMustLoginToSign>boolean</SignerMustLoginToSign>
        <SignerCanCreateAccount>boolean</SignerCanCreateAccount>
        <AllowInPerson>boolean</AllowInPerson>
        <EnablePowerForm>boolean</EnablePowerForm>
        <AllowSignerReassign>boolean</AllowSignerReassign>
        <EnableReservedDomain>boolean</EnableReservedDomain>
        <EnableSequentialSigningAPI>boolean</EnableSequentialSigningAPI>
        <EnableSequentialSigningUI>boolean</EnableSequentialSigningUI>
        <EnableAutoNav>boolean</EnableAutoNav>
        <AutoNavRule>bytes</AutoNavRule>
        <EnableTransactionPoint>boolean</EnableTransactionPoint>
        <EnvelopeIntegrationAllowed>bytes</EnvelopeIntegrationAllowed>
        <EnvelopeIntegrationEnabled>boolean</EnvelopeIntegrationEnabled>
        <CanSelfBrandSend>boolean</CanSelfBrandSend>
        <CanSelfBrandSign>boolean</CanSelfBrandSign>
        <IDCheckRequired>bytes</IDCheckRequired>
        <IDCheckExpire>bytes</IDCheckExpire>
        <IDCheckExpireDays>integer</IDCheckExpireDays>
        <SignDateFormat>string</SignDateFormat>
        <PKISignDownloadedPDFDocs>bytes</PKISignDownloadedPDFDocs>
        <InPersonIDCheckQuestion>string</InPersonIDCheckQuestion>
        <SessionTimeout>integer</SessionTimeout>
        <AttachCompletedEnvelope>boolean</AttachCompletedEnvelope>
        <SignerCanSignOnMobile>boolean</SignerCanSignOnMobile>
        <SignerShowSecureFieldInitialValues>boolean</SignerShowSecureFieldInitialValues>

<SignerAttachCertificateToEnvelopePDF>boolean</SignerAttachCertificateToEnvelopePDF>
        <EnableSignOnPaper>boolean</EnableSignOnPaper>
        <EnableSignOnPaperOverride>boolean</EnableSignOnPaperOverride>
        <EnableSignerAttachments>boolean</EnableSignerAttachments>
        <UseAccountLevelEmail>boolean</UseAccountLevelEmail>
      </AccountSettings>
      <AddressInformation>
        <Address1>string</Address1>
        <Address2>string</Address2>
        <City>string</City>
        <State>string</State>
        <Zip>string</Zip>
        <Phone>string</Phone>
        <Fax>string</Fax>
      </AddressInformation>
    </UpgradeRecipientAccount>
  </soap:Body>
</soap:Envelope>
```

The user being upgraded must have a UserType/UserStatus of Recipient/Active or the call will result in an error. The call returns either a success or failure. If the call fails an error code is provided.

Sample Response XML

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <UpgradeRecipientAccountResponse
xmlns="http://www.docusign.net/API/AccountManagement">
      <UpgradeRecipientAccountResult>
        <Success>boolean</Success>
        <Error>
          <ErrorCode>Unspecified_Error or Invalid_Account_ID or
Account_Requires_User_Name_And_Password_For_Activation or
Account_Is_Already_Activated_For_Salesforce or Invalid_Distributor_For_Account or
Invalid_User_ID or Invalid_Account or User_Is_Not_An_Account_Manager or Invalid_Login or
Invalid_Member_User_Name or Invalid_Member_Email or Member_Email_Not_Allowed or
Member_Email_And_User_Name_Awaiting_Activation or
Member_Email_And_User_Name_Already_Exists_For_This_Account or Member_Password_Blank or
Member_Forgotten_Password_Question_Blank or Member_Forgotten_Password_Answer_Blank or
Invalid_Password_Format or Invalid_Member_Data or
Member_Email_And_User_Name_Already_Exists or Not_Authorized or
Invalid_Distributor_Selected or Invalid_PGP_For_Distributor or Invalid_Credit_Card_Type
or CreditCard_Auth_Failed or Invalid_PGP or Invalid_Plan_Retired or
Invalid_Successor_Plan or Invalid_Credit_Card or Credit_Card_Expiration or
Invalid_AppToken or Distributor_Not_Enabled_For_AppToken or
Plan_Group_Not_Enabled_For_Distributor or Invalid_Configuration_Number or
Invalid_Salesforce_Credentials or Invalid_Salesforce_External_Instance_ID or
Invalid_DocuSign_Connect_Configuration_For_Account or Invalid_User or Invalid_Membership
or Invalid_Account_Member or Invalid_Edit_User or Invalid_Edit_Membership or
Invalid_CanEditSharedAddressBook_Value or Invalid_CanManageTemplates_Value or
Invalid_Membership_ID or Invalid_Request or Partner_Authentication_Failed</ErrorCode>
          <Description>string</Description>
        </Error>
      </UpgradeRecipientAccountResult>
    </UpgradeRecipientAccountResponse>
  </soap:Body>
</soap:Envelope>
```

Error Codes and Associated Messages

The following table lists all of the error numbers and associated messages. The list is presented in error number order.

DocuSign's SOAP and REST APIs use the same error list. The SOAP API shows the error number and the message text, while the REST API shows the upper-case error code and the message text. In many cases additional information is added to the message string.

Note: There are no errors for numbers 125, 131, 132, and 147.

Number	Message	Error Code
1	An Error Occurred.	UNSPECIFIED_ERROR
2	The method specified is not implemented.	METHOD_NOT_IMPLEMENTED
3	The specified Integrator Key was not found or is disabled.	PARTNER_AUTHENTICATION_FAILED
4	The API Version specified is not valid.	INVALID_API_VERSION
100	The envelope specified either does not exist or you have no rights to it.	ENVELOPE_DOES_NOT_EXIST
101	The email address for the recipient is invalid. The recipient Id follows.	INVALID_EMAIL_ADDRESS_FOR_RECIPIENT
102	The email address for the sender is invalid.	INVALID_EMAIL_ADDRESS_FOR_SENDER
103	The UserName and Email did not identify a sender in the system.	SENDER_DOES_NOT_EXIST_IN_SYSTEM
104	The document element did not contain the encoded document, or there is a problem with the encoding.	NO_DOCUMENT_RECEIVED
105	The pagenumber specified in the tab element is not in the document that the tab refers to.	TAB_PAGENUMBER_IS_NOT_IN_DOCUMENT
106	The sender specified is not part of a group that the Partner is authorized to send for.	PARTNER_NO_AUTHORITY_FOR_SENDER
107	The DocumentId specified in the tab element does not refer to a document in this envelope.	TAB_REFERS_TO_MISSING_DOCUMENT
108	The RecipientId specified in the tab element does not refer to a recipient of this envelope.	TAB_REFERS_TO_MISSING_RECIPIENT
109	The Envelope is not Complete. A Complete Envelope Requires Documents, Recipients, Tabs, and a Subject Line.	ENVELOPE_IS_INCOMPLETE
110	A Custom Tab is not Complete. A Custom Tab requires both a Name and a TabLabel.	CUSTOMTAB_IS_INCOMPLETE
111	This Account lacks sufficient permissions.	ACCOUNT_LACKS_PERMISSIONS
112	This User lacks sufficient permissions.	USER_LACKS_PERMISSIONS
113	One or both of Username and Password are invalid.	USER_AUTHENTICATION_FAILED
114	The UserID does not have a valid membership in this Account.	USER_LACKS_MEMBERSHIP
115	The Account ID did not identify an Account in the system.	ACCOUNT_DOES_NOT_EXIST_IN_SYSTEM
116	The UserID did not identify a User in the system.	USER_DOES_NOT_EXIST_IN_SYSTEM
117	The security token format does not conform to expected schema.	INVALID_TOKEN_FORMAT

Number	Message	Error Code
118	This account is not authorized to access the requested envelope.	ACCOUNT_NOT_AUTHORIZED_FOR_ENVELOPE
119	This user is not the sender of the envelope. Only the sender of the envelope may perform the requested operation.	USER_NOT_ENVELOPE_SENDER
120	This user is not the sender or a recipient of the envelope. Only the sender or a recipient of the envelope may perform the requested operation.	USER_NOT_ENVELOPE_SENDER_OR_RECIPIENT
121	Only envelopes in the 'Sent' or 'Delivered' states may be voided.	ENVELOPE_CANNOT_VOID_INVALID_STATE
122	The envelope you are attempting to access has been voided by the sender. For more information, please contact the sender:	ENVELOPE_HAS_BEEN_VOIDED
123	The recipient you have identified is not a valid recipient of the specified envelope.	UNKNOWN_ENVELOPE_RECIPIENT
124	The specified recipient is not a captive recipient. This operation requires a captive recipient.	RECIPIENT_NOT_CAPTIVE
126	The requested user has failed the security check for the requested envelope. Correct the envelope to continue.	RECIPIENT_HAS_FAILED_SECURITY_CHECK
127	A security check was specified for the recipient but the appropriate callback URL was not provided.	REQUIRED_SECURITY_CHECK_URL_MISSING
128	A carbon copy recipient has been specified as captive. This operation is not supported.	CAPTIVE_CARBON_COPY_RECIPIENT_NOT_SUPPORTED
129	One or more Signer type recipients have not been assigned any signable tabs.	SIGNER_RECIPIENT_HAS_NO_SIGNABLE_TABS
130	The specified User is not a member of the specified Account.	USER_DOES_NOT_BELONG_TO_SPECIFIED_ACCOUNT
133	The specified transfer recipient of the envelope is already the owner of the envelope.	ENVELOPE_TRANSFEREE_ALREADY_OWNS_ENVELOPE
134	The specified envelope is not in a terminal state and cannot be transferred.	ENVELOPE_CANNOT_TRANSFER_INVALID_STATE
135	The specified Anchor Tab string was not found in the document.	ANCHOR_TAB_STRING_NOT_FOUND
136	The specified User Offset exceeds the page boundaries.	INVALID_USER_OFFSET
137	The Anchor Tab String Format is invalid.	INVALID_ANCHOR_TAB_STRING
138	Only envelopes in the 'Sent' or 'Delivered' states may be corrected.	ENVELOPE_CANNOT_CORRECT_INVALID_STATE
139	The specified User is not a recipient of the specified envelope.	RECIPIENT_NOT_FOUND_FOR_CORRECTION
140	The specified envelope has duplicate recipients.	ENVELOPE_HAS_DUPLICATE_RECIPIENTS
141	The specified envelope corrections have duplicate recipients.	CORRECTION_HAS_DUPLICATE_RECIPIENTS
142	The specified envelope corrections result in duplicate recipients.	CORRECTION_RESULTS_IN_DUPLICATE_RECIPIENTS

Number	Message	Error Code
143	The specified envelope correction has a blank username.	CORRECTION_HAS_A_BLANK_USERNAME
144	The specified envelope correction has a username with invalid characters.	CORRECTION_HAS_A_USERNAME_WITH_INVALID_CHARACTERS
145	Email sending failed.	FAILED_EMAIL_SENDING
146	The specified envelope has In-Session recipients.	ENVELOPE_HAS_INSESSION_RECIPIENTS
148	Envelopes with AC Status - Deposit Pending or Deposited or Deposited As Authoritative Copy or Deposit Failed cannot be transferred.	ENVELOPE_CANNOT_TRANSFER_INVALID_ACSTATUS
149	The specified Account is suspended	ACCOUNT_HAS_BEEN_SUSPENDED
150	The specified Account lacks an Account Pricing Plan	ACCOUNT_LACKS_PRICING_PLAN
151	The specified recipient has invalid Routing Order.	RECIPIENT_HAS_INVALID_ROUTINGORDER
152	The requested email domain is reserved in the DocuSign system. Please contact your company's DocuSign account administrator for more information.	EMAIL_IS_RESERVED
153	The corrected email (s)are reserved.	CORRECTED_EMAIL_IS_RESERVED
154	The user name for the recipient is invalid.	INVALID_USERNAME_FOR_RECIPIENT
155	The user name is invalid.	INVALID_USERNAME
156	The token for an out of sequence recipient cannot be generated.	RECIPIENT_NOT_IN_SEQUENCE
157	Envelope is not in the correct export status.	ENVELOPE_AC_EXPORT_INVALID_STATUS
158	Invalid transaction ID.	ENVELOPE_AC_EXPORT_INVALID_TRANSID
159	Envelope already exported.	ENVELOPE_AC_EXPORT_COMPLETED
160	Envelope not authoritative copy.	ENVELOPE_AC_EXPORT_NOT_AC_COPY
161	Tab is placed off of the page.	TAB_OUT_OF_BOUNDS
162	The token for a recipient that has failed activation cannot be generated.	ACTIVATION_FAILED
163	Cannot add the document for the template.	TEMPLATE_CANNOT_ADD_DOCUMENT
164	Freeform signing is not allowed for your account because it conflicts with other settings, please place signing tabs for each signer.	ONESIGNALLSIGN_NOT_SATISFIED
165	Required recipient in the template has not been provided.	TEMPLATE_REQUIRED_RECIP_NOT_SATISFIED
166	Template was not provided.	TEMPLATE_NOT_PROVIDED
167	Unable to load template.	TEMPLATE_UNABLE_TO_LOAD
168	Unable to load field data from PDF.	TEMPLATE_UNABLE_TO_LOAD_FIELDDATA
169	Unable to process field data for the Template.	TEMPLATE_UNABLE_TO_PROCESS_FIELDDATA
170	Unable to flatting PDF for the Template.	TEMPLATE_UNABLE_TO_FLATTEN_PDF

Number	Message	Error Code
171	Unable to transform Template and data to Envelope.	TEMPLATE_TO_ENVELOPE_ERROR
172	Unable to override Envelope data from EnvelopeInformation.	TEMPLATE_OVERRIDE_ENVELOPEINFORMATION_ERROR
173	Unable to add AdditionalTabs from Templates to Envelope.	TEMPLATE_ADDITIONALTABS_ERROR
174	Unable to override recipient data from Recipients to Envelope.	TEMPLATE_OVERRIDE_RECIPIENTDATA_ERROR
175	Unable to load the document.	UNABLE_TO_LOAD_DOCUMENT
176	Role does not exist in the template.	TEMPLATE_ROLESPECIFIED_DOES_NOT_EXIST
177	Recipient ID not found in the Recipient list for the role.	TEMPLATE_RECIPIENTID_FOR_ROLE_NOTFOUND
178	The regular expression provided is not valid.	REGULAREXPRESSION_IS_INVALID
179	The routing order of the recipients is not valid.	TEMPLATE_MERGE_INVALID_ROUTE_ORDER
180	Unable to purge documents.	UNABLE_TO_PURGE_DOCUMENTS
181	Invalid template ID.	TEMPLATE_ID_INVALID
182	Not authorized to change template.	TEMPLATE_AUTHENTICATION_FAILED
183	Invalid ID for the address book entry.	ADDRESSBOOK_ID_INVALID
184	Invalid Email for the address book entry.	ADDRESSBOOK_EMAIL_INVALID
185	The individual attempting to access the address book entry does not own it.	ADDRESSBOOK_NOTOWNER
186	The specified username(s) and email(s) are not in our system and the email does not allow new users.	USERNAME_IS_NOT_ALLOWED
187	This address book entry cannot be shared.	ADDRESSBOOK_CANTSHARE
188	Captive Recipient cannot be of type In Person Signer.	CAPTIVE_IN_PERSON_SIGNER_RECIPIENT_NOT_SUPPORTED
189	The In Person Signing Host must be a valid and active DocuSign user.	IN_PERSON_SIGNING_HOST_MUST_BE_VALID_USER
190	For In Person Signer type, the Recipient Signer Name cannot be blank.	IN_PERSON_SIGNER_NAME_CANNOT_BE_BLANK
191	Only three custom fields allowed per envelope.	ENVELOPE_ONLYTHREECUSTOMFIELDS_ALLOWED
192	The document ID has already been used.	TEMPLATE_DOCUMENTID_MAPPING_INVALID
193	The transaction ID has already been submitted.	TRANSACTIONID_REDUNDANT
194	The transaction ID does not exist.	TRANSACTIONID_DOES_NOT_EXIST
195	Unable to submit the transaction for processing.	TRANSACTIONID_UNABLE_TO_SUBMIT
196	The maximum number of tabs was exceeded	MAX_TABS_EXCEEDED
197	Account settings indicate the recipient must be an active DocuSign user.	RECIPIENT_MUST_BE_VALID_USER
198	An Error Occurred during anchor tag processing.	ANCHOR_TAG_PROCESSING_FAILURE
199	A conditional tab references an invalid parent. Parent label must match another tab. Only one parent allowed. Signature tabs may not be parent tabs.	CONDITIONALTAB_HAS_INVALID_PARENT

Number	Message	Error Code
200	Schema validation Failed. Additional Errors follow.	FAILED_SCHEMA_VALIDATION
201	Authentication is not setup correctly for the recipient.	INVALIDAUTHENTICATIONSETUP
202	A signer attachment tab is placed on a non-signer attachment document (a document without an AttachmentDescription).	SATABONNONSADOCUMENT
203	The requested envelope is not a draft.	ENVELOPE_NOT_DRAFT
204	More than one signer attachment tab is placed on a signer attachment document for a recipient.	MULTIPLESATABSONDOCUMENT
205	System was unable to convert this document to a PDF.	UNABLE_TO_CONVERT_DOCUMENT
206	Page number not specified in tab element.	TAB_PAGE_NUMBER_NOT_SPECIFIED
207	The maximum number of hourly API invocations has been exceeded.	HOURLY_APIINVOCATION_LIMIT_EXCEEDED
208	The user does not own the folder	USER_NOT_FOLDER_OWNER
209	The envelope does not exist in the folder	ENVELOPE_NOT_IN_FOLDER
210	Envelopes may not be deleted from the RecycleBin	CANNOT_DELETE_FROM_RECYCLEBIN
211	Invalid characters detected	INVALID_CHARACTERS_PROVIDED
212	Account or user does not have permission to set recipient email notifications.	USER_LACKS_RECIPIENTEMAILNOTIFICATION_PERMISSION
213	Account does not have permission to send to Certified Delivery recipient type.	CANNOT_SEND_TO_CERTIFIED_DELIVERY
214	Account does not have permission to send to Agent recipient type.	CANNOT_SEND_TO_AGENT
215	Account does not have permission to send to Editor recipient type.	CANNOT_SEND_TO_EDITOR
216	Invalid envelope expiration and/or reminder information provided.	INVALID_EXPIRATION_REMINDER_DATA
217	Account does not have permission to set Allow Markup.	CANNOT_ALLOW_MARKUP
218	A Mergefield value for a Tab could not be retrieved.	TAB_MERGEFIELD_VALUE_NOT_RETRIEVED
219	Account does not have permission to send to Intermediary recipient type.	CANNOT_SEND_TO_INTERMEDIARY
220	Send to Manage Role (Agent, Editor, Intermediary) cannot have tabs assigned.	CANNOT_ASSIGN_TAB_TO_MANAGER ROLE
221	Multiple Send to Manage Roles (Agent, Editor, Intermediary) cannot be assigned same routing order.	CANNOT_ASSIGN_MANAGERS_SAME_ROUTING_ORDER
222	The request contained at least one invalid parameter.	INVALID_REQUEST_PARAMETER
223	The DocumentId provided is invalid.	INVALID_DOCUMENT_ID
224	The HTTP method specified is not allowed with this resource.	METHOD_NOT_ALLOWED
225	The URL provided does not resolve to a resource.	RESOURCE_NOT_FOUND
226	Content Type specified is not supported.	INVALID_CONTENT_TYPE
227	The maximum number of members for the account has been exceeded.	MAX_MEMBERS_EXCEEDED

Number	Message	Error Code
228	An error was found while parsing the multipart request.	INVALID_MULTI_PART_REQUEST
229	No recipients were found in the request.	RECIPIENTS_NOT_PROVIDED
230	The Tab specified is not valid for the requested operation.	INVALID_TAB_OPERATION
231	The Tab formula is not valid.	INVALID_TAB_FORMULA
232	Recipient Calculated Fields have a circular reference.	FORMULA_CIRCULAR_REFERENCE
233	The document specified was not found.	DOCUMENT_DOES_NOT_EXIST
234	A recipient ID is missing or invalid.	INVALID_RECIPIENT_ID
235	Functionality not supported.	NOT_SUPPORTED
236	A 'Send on Behalf Of' user is not permitted for this operation.	SOBO_USER_NOT_ALLOWED
237	AppToken is invalid or distributor not found for AppToken	INVALID_APPTOKEN
238	The PlanId specified is not valid.	INVALID_PLANID
239	The Distributor specified is not valid.	INVALID_DISTRIBUTOR
240	The Social Provider information specified is not valid.	INVALID_SOCIAL_INFORMATION
241	This social provider account is already mapped in the system.	SOCIAL_PROVIDER_ALREADY_MAPPED
242	Social Provider User Name is not valid.	INVALID_SOCIAL_USER_NAME
243	Social Provider is not valid.	INVALID_SOCIAL_PROVIDER
244	The Url override format is not supported. Use supported formats such as .json or .xml.	INVALID_REQUEST_RESPONSE_OVERRIDE_FORMAT
245	The credit card information supplied was not valid.	INVALID_CREDIT_CARD_INFORMATION
246	The billing plan is retired and cannot be used.	INVALID_BILLING_PLAN_RETIRED
247	The request body is missing or improperly formatted.	INVALID_REQUEST_BODY
248	Invalid Successor Billing Plan.	INVALID_BILLING_PLAN_SUCCESSOR
249	The billing plan could not be changed.	BILLING_PLAN_ERROR
250	Username and email combination already exists for this account.	USER_ALREADY_EXISTS_IN_ACCOUNT
251	Invalid UserId.	INVALID_USERID
252	User is not an account administrator.	USER_NOT_ACCOUNT_ADMIN
253	Invalid email address.	INVALID_EMAIL_ADDRESS
254	The email address is not shared.	EMAIL_NOT_SHARED
255	This user name and email are awaiting activation.	USER_AWAITING_ACTIVATION
256	The password is invalid.	INVALID_PASSWORD
257	Invalid forgotten password challenge.	INVALID_PASSWORD_CHALLENGE
258	Fax Number is invalid.	INVALID_FAXNUMBER
259	Fax Recipient must be the only recipient at a routing order.	FAX_RECIPIENT_ROUTING
260	Fax Recipient Type must be a signer.	FAX_RECIPIENT_TYPE_NOT_SUPPORTED
261	Fax Recipient cannot be corrected after sent.	FAX_RECIPIENT_CORRECT_WAS_SENT

Number	Message	Error Code
262	Invalid operation on captive recipient.	INVALID_CAPTIVE_RECIPIENT_OPERATION
263	The email address does not specify an active user.	EMAIL_NOT_ACTIVE
264	The specified envelope has no recipients.	ENVELOPE_HAS_NO_RECIPIENTS
265	The recipient could not be updated.	RECIPIENT_UPDATE_FAILED
266	The recipient Editor must be an existing DocuSign account member.	EDITOR_MUST_HAVE_ACCOUNT
267	EnableWetSign must be set on the envelope.	MUST_ENABLE_SIGNONPAPER
268	The signature specified was not found.	SIGNATURE_NOT_FOUND
269	Sign on paper has not been enabled for this envelope.	ENVELOPE_NOT_ENABLED_WETSIGN
270	The value for 'country' is not supported.	INVALID_ADDRESS_COUNTRY
271	The value for 'state' is not supported.	INVALID_ADDRESS_STATE
272	Fax Recipient Token must be of type Signer.	FAX_BACK_MUST_BE_SIGNER
273	Purchased envelopes not added.	PURCHASED_ENVELOPES_NOT_ADDED
274	A requested plan item is not enabled for this account.	PLAN_ITEM_NOT_ENABLED
275	The folder type specified is not valid.	INVALID_FOLDER_TYPE
276	This recipientId already exists.	RECIPIENT_ALREADY_EXISTS_IN_ENVELOPE
277	Janrain request Error.	JANRAIN_REQUEST_ERROR
278	Invalid recipient status for delete.	INVALID_RECIPIENT_STATUS_FOR_DELETE
279	Unable to delete document.	UNABLE_TO_DELETE_DOCUMENT
280	The envelope allowance for the account has been exceeded.	ENVELOPE_ALLOWANCE_EXCEEDED
281	Recipient must not have an active account.	RECIPIENT_NOT_ACCOUNTLESS
282	Recipient is not a signer type.	RECIPIENT_NOT_SIGNER
283	The data could not be converted.	FORMAT_CONVERSION_ERROR
284	PDF URL Inaccessible	PDF_INACCESSIBLE
285	Mobile notifier configuration failed.	MOBILE_NOTIFIER_CONFIG_FAILED
286	Invalid 'content transfer encoding' for envelope.	INVALID_CONTENT_TRANSFER_ENCODING
287	Invalid envelope status.	ENVELOPE_INVALID_STATUS
288	Invalid file type chosen for conversion.	FORMAT_CONVERSION_ERROR_INVALID_FILE_TYPE
289	Document name not specified.	DOCUMENT_NAME_NOT_SPECIFIED
290	Invalid Document Order.	INVALID_DOCUMENT_ORDER
291	Social login not enabled on account.	SOCIAL_LOGIN_NOT_ENABLED
292	The MemberGroupId provided is invalid.	INVALID_GROUP_ID
293	The SettingsTemplateId provided is invalid.	INVALID_PERMISSION_PROFILE_ID
294	The plan is not offered in the requested currency.	INVALID_PLAN_CURRENCY
295	This document cannot be excluded for this recipient.	CANNOT_EXCLUDE_DOCUMENT
296	The feature set is not configured for the requested plan.	INVALID_PLAN_FEATURESET

Number	Message	Error Code
297	Authorization has been revoked.	AUTHORIZATION_REVOKED
298	Unable to revoke access.	AUTHORIZATION_REVOKED_FAILED
299	Maximum number of access tokens exceeded.	AUTHORIZATION_MAX_ACCESS_TOKENS_EXCEEDED
300	Invalid Token.	INVALID_TOKEN
301	The currency type cannot be changed.	INVALID_CURRENCY_REQUEST
302	The specified setting was not found.	CUSTOM_SETTING_NOT_FOUND
303	The maximum length for custom settings was exceeded	CUSTOM_SETTINGS_MAX_EXCEEDED
304	Invalid Grant	INVALID_OAUTH2_SAML_ASSERTION
305	Custom settings were not found.	CUSTOM_SETTINGS_NOT_FOUND
306	Invalid Brand for this Account.	INVALID_BRAND
307	The page image specified was not found.	PAGE_IMAGE_NOT_FOUND
308	The envelope is no longer in a status to be signed.	ENVELOPE_DECLINE_TO_SIGN_INVALID_STATUS
309	Brands could not be created.	BRAND_CREATE_FAILED
310	Brands could not be deleted.	BRAND_DELETE_FAILED
311	Social account information provided is invalid.	INVALID_SOCIAL_ACCOUNT
312	Accessibility format is invalid.	INVALID_ACCESSIBILITY_FORMAT
313	Document sequence is invalid.	INVALID_DOCUMENT_SEQUENCE
314	Fax Pending Failed.	FAX_START_PENDING_FAILED
315	Invalid connect log id.	INVALID_CONNECT_LOGID
316	Invalid connect id.	INVALID_CONNECT_ID
317	Consumer disclosure acceptance is required.	CONSUMER_DISCLOSURE_ACCEPTANCE_REQUIRED
318	Recipient has not passed authentication steps.	RECIPIENT_AUTHENTICATION_REQUIRED
319	Brand is locked and cannot be reset.	BRAND_LOCKED
320	Maximum number of connect custom configuration exceeded.	MAX_CONNECT_CUSTOM_CONFIGURATION_EXCEEDED
321	Recipient setting value is not an allowed value.	INVALID_RECIPIENT_SETTING_VALUE
322	Connect Error.	CONNECT_ERROR
323	Custom Field already exists.	CUSTOM_FIELD_ALREADY_EXISTS
324	Invalid Custom Field Id.	INVALID_CUSTOM_FIELD_ID
325	Invalid request for Fax token with a PKI signer certificate requirement.	INVALID_FAX_RECIPIENT_SIGNERCERTREQUIREMENT
326	You must move all of the envelopes before you can delete this folder.	MUST_MOVE_ENVELOPES_BEFORE_DELETE
327	You must delete all sub-folders first before you can delete this folder.	MUST_DELETE_SUBFOLDERS
328	You cannot delete predefined system folders.	CANNOT_DELETE_PREDEFINED_FOLDERS
329	The specified folder id was not found for the user.	FOLDER_NOT_FOUND
330	The name was not specified for the folder.	FOLDER_NAME_NOT_DEFINED
331	The folder could not be created/updated.	FOLDER_UPDATE_FAILED

Number	Message	Error Code
332	Invalid parent folder id.	INVALID_PARENT_FOLDER_ID
333	The folder id must not be defined when adding a folder.	FOLDER_ID_DEFINED_DURING_ADD
334	The folder id was not found.	FOLDER_ID_NOT_FOUND
335	An OAuth2 error occurred.	OAUTH_ERROR
336	Invalid Custom Setting Name.	INVALID_CUSTOM_SETTING_NAME
337	Documents are required for offline signing.	OFFLINE_SIGNING_DOCUMENTS_REQUIRED
338	ClientUserId is required for offline signer.	OFFLINE_SIGNING_CLIENTUSERID_REQUIRED
339	Draft envelopes are not allowed for offline signing.	OFFLINE_SIGNING_INVALID_DRAFT_ENVELOPE
340	DeliveryMethod of offline is only allowed for signer recipients.	OFFLINE_SIGNING_DELIVERY_METHOD_MUST_BE_SIGNER
341	Invalid offline signer routing order.	OFFLINE_SIGNING_INVALID_SIGNER_ROUTING_ORDER
342	Offline signer must have at least one signature or initials tab.	OFFLINE_SIGNING_SIGNER_TABS_REQUIRED
343	Signature image not provided for offline signature or initials tab.	OFFLINE_SIGNING_TAB_IMAGE_REQUIRED
344	Offline signing not allowed.	OFFLINE_SIGNING_NOT_ALLOWED
345	'signedDateTime' must be specified for offline recipients.	OFFLINE_SIGNING_SIGNEDDATETIME_REQUIRED
346	Must have distinct email and username for all offline recipients.	OFFLINE_SIGNING_DUPLICATE_RECIPIENTS
347	Invalid tab type specified for offline signing.	OFFLINE_SIGNING_INVALID_TAB_TYPE
348	Invalid value specified for offline DateTime.	OFFLINE_SIGNING_DATETIME_INVALID
349	Invalid value specified for clientUserId.	INVALID_CLIENTUSERID
350	Cannot use an envelope id for a template operation.	TEMPLATE_CANNOT_USE_ENVELOPEID_FOR_TEMPLATE_OPERATION
351	Cannot use a template id for an envelope operation.	ENVELOPE_CANNOT_USE_TEMPLATEID_FOR_ENVELOPE_OPERATION
352	Recipient does not have permission to access document.	RECIPIENT_NO_ACCESS
353	User is not a recipient.	USER_NOT_RECIPIENT
354	User does not match recipient.	DECLINE_TO_SIGN_USER_IS_NOT_RECIPIENT
355	Recipient correct succeed, but Decline to Sign failed.	RECIPIENT_CORRECT_SUCCESS_DECLINE_TO_SIGN_FAILED
356	Failed to create the response for the operation.	FAILED_TO_CREATE_RESPONSE
357	Failed to create the request for the operation.	FAILED_TO_CREATE_REQUEST
358	The maximum sale discount period has been exceeded.	MAX_SALE_DISCOUNT_PERIODS_EXCEEDED
359	Invalid Envelope Subject.	INVALID_ENVELOPE_SUBJECT
360	Invalid Envelope Email Blurb.	INVALID_ENVELOPE_EMAILBLURB

Number	Message	Error Code
361	Invalid Recipient Subject.	INVALID_RECIPIENT_SUBJECT
362	Invalid Recipient Email Blurb.	INVALID_RECIPIENT_EMAILBLURB
363	This user lacks sufficient permissions to access this resource. A password is required to edit this template.	TEMPLATE_PASSWORD_REQUIRED
364	This user lacks sufficient permissions to access this resource. The user is not the owner of the envelope.	USER_NOT_ENVELOPE_OWNER
365	This user lacks sufficient permissions to access this resource. The user is not the owner of the template.	USER_NOT_TEMPLATE_OWNER
366	Draft envelopes can only be moved to the RecycleBin.	DRAFT_ENVELOPES_CAN_ONLY_BE_MOVED_TO_RECYCLEBIN
367	Deleted draft envelopes can only be moved back to the drafts folder.	DELETED_DRAFTS_CAN_ONLY_BE_MOVED_TO_DRAFTS
368	Invalid recipient status for correct.	INVALID_RECIPIENT_STATUS_FOR_CORRECT
369	An error occurred in the external doc service.	EXTERNAL_DOC_SERVICE_ERROR
370	External doc service not authenticated.	EXTERNAL_DOC_SERVICE_NOT_AUTHENTICATED
371	Included seats must be greater than zero and not less than the minimum included seats.	INVALID_INCLUDED_SEATS_VALUE
372	Only 'true' or 'false' are valid values for 'shared'.	ADDRESSBOOK_INVALID_SHARING_OPTIO
373	Attempting to update the address book failed.	ADDRESSBOOK_ATTEMPTED_UPDATE_FAILED
374	The user name provided was null or empty.	ADDRESSBOOK_USERNAME_INVALID
375	The account id provided does not exist, or is improperly formatted.	ACCOUNT_SPECIFIED_INVALID
376	The account name provided does not exist or is null or empty.	ADDRESSBOOK_ACCOUNTNAME_INVALID
377	The contact id provided does not exist in the address book, or is improperly formatted.	ADDRESSBOOK_CONTACTID_INVALID
378	The search criteria provided are invalid.	ADDRESSBOOK_INVALID_SEARCH_CRITERIA
379	The address book action resulted in an invalid state.	ADDRESSBOOK_INVALID_RESULT_STATE
380	The update criteria provided are invalid.	ADDRESSBOOK_INVALID_UPDATE_CRITERIA
381	Invalid value specified for the folder filter.	INVALID_FOLDER_FILTER
382	Tab specified was not found.	TAB_NOT_FOUND
383	Invalid TabId specified.	INVALID_TABID
384	No Enterprise Report data could be returned because one or more query/filter parameters was invalid.	REPORT_QUERY_INVALID
385	Message is locked.	MESSAGE_LOCKED
386	Recipients are locked and cannot be modified.	RECIPIENTS_LOCKED
387	The Enterprise Report failed to execute. See additional information for more details.	REPORT_EXECUTION_FAILED
388	A folder ID is missing or invalid.	INVALID_FOLDER_ID
389	Document field name to long.	DOCUMENTFIELDNAMETOLONG

Number	Message	Error Code
390	Document field value to long.	DOCUMENTFIELDVALUETOLONG
391	SMS Authentication not allowed.	SMS_AUTHENTICATION_NOT_ALLOWED
392	SAML Authentication not allowed.	SAML_AUTHENTICATION_NOT_ALLOWED
393	The Enterprise Report being requested does not exist.	REPORT_DOES_NOT_EXIST
394	The Enterprise Report could not be executed because more than one report match the query/filter provided.	REPORT_MULTIPLE_MATCHES_FOUND
395	Invalid status specified for signature.	INVALID_SIGNATURE_STATUS
396	Invalid attribute specified for updating recipient signature.	INVALID_PUT_RECIPIENT_SIGNATURE_ATTRIBUTE
397	The specified captive recipient was not found.	CAPTIVE_RECIPIENT_NOT_FOUND
398	You have already saved a report by this name.	REPORT_DUPLICATE_NAME_EXISTS
399	The cache object size has been exceeded.	CACHE_OBJECT_SIZE_EXCEEDED
400	The initialization string was null or empty.	CACHE_INVALID_INIT_STRING
401	You have reached the maximum number of saved reports. Please deactivate one and try again.	REPORT_MAX_SAVED_REPORTS_EXCEEDED
402	You have reached the maximum number of scheduled reports. Please deactivate one and try again.	REPORT_MAX_SCHEDULED_EXCEEDED
403	The timezone offset specified is invalid.	REPORT_TIMEZONE_INVALID
404	No date value was provided or the date provided was null.	REPORT_DATE_INVALID
405	App Store Error.	APPSTORE_ERROR
406	Receipt verification failed.	APPSTORE_RECEIPT_VERIFICATION_FAILED
407	Receipt not provided	APPSTORE_RECEIPT_MISSING
408	Product Id not provided	APPSTORE_PRODUCTID_MISSING
409	Product Id is Invalid.	APPSTORE_PRODUCTID_INVALID
410	Receipt Data not provided	APPSTORE_RECEIPT_DATA_MISSING
411	Payment method error.	APPSTORE_PAYMENT_METHOD_ERROR
412	Signer certificate type not enabled for account.	SIGNER_CERTIFICATE_TYPE_NOT_ENABLED
413	Signer certificate type not supported.	SIGNER_CERTIFICATE_TYPE_NOT_SUPPORTED
414	RequireSignerCertificate can only be used for Recipient Type Signer and InPersonSigner	SIGNER_CERTIFICATE_RECIPIENT_TYPE_INVALID
415	RequireSignerCertificate conflicts with another envelope setting.	SIGNER_CERTIFICATE_SETTINGS_CONFLICT
416	RequireSignerCertificate specified but no required SignHere or InitialHere tabs are present.	SIGNER_CERTIFICATE_REQUIRED_TAB_ERROR
417	When RequireSignerCertificate is specified, there must not be another recipient at the same routing order.	SIGNER_CERTIFICATE_ROUTING_ORDER_ERROR
418	PKI Certificate Provider is not set.	SIGNER_CERTIFICATE_PROVIDER_NOT_SET

Number	Message	Error Code
419	Billing Invoices could not be retrieved.	BILLING_INVOICES_NOT_RETRIEVED
420	Authorization header is required.	AUTHORIZATION_HEADER_REQUIRED
421	The request requires higher privileges than provided by the access token.	AUTHORIZATION_INSUFFICIENT_SCOPE
422	The access token provided is expired, revoked or malformed.	AUTHORIZATION_INVALID_TOKEN
423	The authorization request is malformed.	AUTHORIZATION_INVALID_REQUEST
424	Login to billing system failed.	BILLING_SYSTEM_LOGIN_FAILED
425	The Address Id did not identify an Address in the system.	ADDRESS_DOES_NOT_EXIST_IN_SYSTEM
426	The Billing setup is not found in the system.	BILLING_SETUP_DOES_NOT_EXIST_IN_SYSTEM
427	Billing not configured in the system.	BILLING_NOT_CONFIGURED
428	First Name is required.	REQUIRED_FIRSTNAME
429	Last Name is required.	REQUIRED_LASTNAME
430	Email is required.	REQUIRED_EMAIL
431	First Name length exceeded.	LENGTH_EXCEEDED_FIRSTNAME
432	Last Name length exceeded.	LENGTH_EXCEEDED_LASTNAME
433	Email length exceeded.	LENGTH_EXCEEDED_EMAIL
434	Phone length exceeded.	LENGTH_EXCEEDED_PHONE
435	Fax length exceeded.	LENGTH_EXCEEDED_FAX
436	Address Line 1 is required.	REQUIRED_ADDRESSLINE1
437	City is required.	REQUIRED_CITY
438	State/Province is required.	REQUIRED_STATE
439	Postal Code is required.	REQUIRED_POSTALCODE
440	Country is required.	REQUIRED_COUNTRY
441	Credit Card Expiration is not valid.	INVALID_CREDIT_CARD_EXPIRATION
442	Credit Card is expired.	CREDIT_CARD_EXPIRED
443	Credit Card Expiration Year is not valid.	INVALID_CREDIT_CARD_EXPIRATION_YEAR
444	The Current Account Plan does not exist in the system.	ACCOUNT_PLAN_DOES_NOT_EIXST_IN_SYSTEM
445	The Current Account Billing Period does not exist in the system.	BILLING_PERIOD_DOES_NOT_EXIST_IN_SYSTEM
446	Billing changes are pending.	BILLING_CHANGES_PENDING
447	A payment processing error occurred in the billing system.	UNSPECIFIED_ERROR_PROCESSING_PAYMENT
448	An error occurred in the billing system.	UNSPECIFIED_ERROR_IN_BILLING_SYSTEM
449	The Credit Card Type is not valid in the system.	INVALID_CREDIT_CARD_TYPE
450	Credit Card Cardholder Name is required.	REQUIRED_CREDIT_CARD_NAME
451	Credit Card Number is not valid.	INVALID_CREDIT_CARD_NUMBER
452	The Current Account Plan Start Date does not exist in the system.	PLAN_START_DATE_DOES_NOT_EXIST_IN_SYSTEM
453	The Payment is not valid.	PAYMENT_NOT_VALID

Number	Message	Error Code
454	The Payment does not exist in the system.	PAYMENT_DOES_NOT_EXIST_IN_SYSTEM
455	Address Line 1 length exceeded.	LENGTH_EXCEEDED_ADDRESSLINE1
456	Address Line 2 length exceeded.	LENGTH_EXCEEDED_ADDRESSLINE2
457	City length exceeded.	LENGTH_EXCEEDED_CITY
458	Country length exceeded.	LENGTH_EXCEEDED_COUNTRY
459	State/Province length exceeded.	LENGTH_EXCEEDED_STATE
460	Postal Code length exceeded.	LENGTH_EXCEEDED_POSTALCODE
461	Credit Card Cardholder Name length exceeded.	LENGTH_EXCEEDED_CREDIT_CARD_NAME
462	The Invoice PDF is not available.	INVOICE_PDF_NOT_AVAILABLE
463	The InvoiceID did not identify an Invoice in the system.	INVOICE_DOES_NOT_EXIST_IN_SYSTEM
464	The State/Province is not valid in the system.	INVALID_STATE
465	The Payment Id is not valid	INVALID_PAYMENT_ID
466	Invoice Id is not valid	INVALID_INVOICE_ID
467	The Country is not valid in the system.	INVALID_COUNTRY
468	Filter Start Date cannot be greater than Filter End Date.	FILTER_START_EXCEEDS_END
469	Filter Start Date is less than minimum value allowed.	FILTER_START_LESS_THAN_MIN_ALLOWED
470	Filter Start Date is greater than maximum value allowed.	FILTER_START_GREATER_THAN_MAX_ALLOWED
471	Filter End Date is less than minimum value allowed.	FILTER_END_LESS_THEN_MIN_ALLOWED
472	Filter End Date is greater than maximum value allowed.	FILTER_END_GREATER_THAN_MAX_ALLOWED
473	Credit Card not authorized.	CREDIT_CARD_NOT_AUTHORIZED
474	The Credit Card does not exist in the system.	CREDIT_CARD_DOES_NOT_EXIST_IN_SYSTEM
475	The address update failed.	ADDRESS_UPDATE_FAILED
476	The maximum number of BCC email addresses has been exceeded.	MAX_BCC_ADDRESSES_EXCEEDED
477	Shared Templates can only be moved by the creator of the template.	SHARED_TEMPLATE_NOT_CREATOR
478	Unable to clone the specified envelope.	UNABLE_TO_CLONE_ENVELOPE
479	Not eligible for renewal cancellation.	NOT_ELIGIBLE_FOR_RENEWAL_CANCELLATION
480	The Credit Card is not allowed for the specified currency.	CREDIT_CARD_NOT_ALLOWED_FOR_CURRENCY
481	This email settings object already exists.	EMAIL_SETTINGS_ALREADY_EXISTS
482	No email settings were found.	EMAIL_SETTINGS_NOT_FOUND
483	Duplicate BCC Email Address Id specified.	DUPLICATE_BCC_EMAIL_ADDRESS_ID
484	BCC Email Address Id not found.	BCC_EMAIL_ADDRESS_ID_NOT_FOUND
485	Duplicate BCC email address.	DUPLICATE_BCC_EMAIL_ADDRESS

Number	Message	Error Code
486	Credit Card update failed.	CREDIT_CARD_UPDATE_FAILED
487	Credit Card has invalid address.	CREDIT_CARD_INVALID_ADDRESS
488	Envelope has already been sent.	ENVELOPE_ALREADY_SENT
489	The maximum number of accounts was exceeded.	MAX_ACCOUNTS_EXCEEDED
490	No User was found for given criteria.	USER_NOT_FOUND
491	UserCustomTabId is invalid.	INVALID_USERCUSTOMTABID
492	Custom Tab Type is invalid.	INVALID_CUSTOM_TAB_TYPE
493	Custom Tab User Id is invalid.	INVALID_CUSTOM_TAB_USER_ID
494	Custom Tab Property is invalid.	INVALID_CUSTOM_TAB_PROPERTY
495	Custom Tab delete failed.	CUSTOM_TAB_DELETE_FAILED
496	This Account lacks sufficient permissions to use custom tab.	ACCOUNT_LACKS_CUSTOM_TAB_PERMISSION
497	Fax error occurred.	FAX_ERROR_OCCURRED
498	The fax could not be read as a PDF.	FAX_FORMAT_INVALID
499	The POST request for submitting a fax was invalid or not present.	FAX_ERROR_POST_INVALID
500	The domain name specified was invalid. This field is required.	FAX_ERROR_DOMAIN_INVALID
501	The envelope fax id provided is invalid (zero or less).	FAX_ERROR_SAVE_FAX_FAILED
502	The fax entry could not be saved.	FAX_ERROR_INVALID_ENVELOPE_ID
503	The brand identifier is unknown or invalid.	INVALID_BRAND_ID
504	The type of logo is unrecognized.	INVALID_BRAND_LOGOTYPE
505	The branding logo is unsupported or invalid.	INVALID_BRAND_LOGO
506	The CSS color is not regognized as a valid and supported W3C-compliant format.	INVALID_CSS_COLOR
507	The URI is not valid.	INVALID_URI
508	One or more brands could not be updated.	BRAND_UPDATE_FAILED
509	The fax was attempted, but failed to send.	FAX_FAILED_TO_SEND
510	The fax provider is not authorized to send or receive faxes.	FAX_INTEGRATOR_NOT_AUTHORIZED
511	Error adding documents to the purge queue.	DOCUMENT_PURGE_FAILED
512	The fax error request was null or empty.	FAX_ERROR_STATUS_REQUEST_INVALID
513	A corresponding fax status could not be found.	FAX_ERROR_NO_MATCHING_FAX_RECORD
514	Address Book Designation is not valid.	ADDRESSBOOK_DESIGNATION_INVALID
515	Invalid login status.	INVALID_LOGIN_STATUS
516	Report XML is invalid.	REPORT_XML_INVALID
517	Report data is invalid.	REPORT_DATA_INVALID
518	Report is missing a required parameter.	REPORT_MISSING_REQUIRED_PARAMETER
519	A parameter is not set for the report.	REPORT_REQUIRED_PARAMETER_NOT_SET

Number	Message	Error Code
520	Too many query parameters specified.	REPORT_QUERY_PARAMETERS_EXCEED_EXPECTED
521	Query parameter types are not consistent.	REPORT_QUERY_PARAMETER_TYPES_INCONSISTENT
522	Invalid option for report schedule.	REPORT_SCHEDULE_INVALID_OPTION
523	Report schedule failed.	REPORT_SCHEDULE_SCHEDULE_FAILED
524	Report schedule could not be created.	REPORT_SCHEDULE_CREATION_FAILED
525	Report subscription failed.	REPORT_SUBSCRIPTION_FAILED
526	Report schedule already exists.	REPORT_SCHEDULE_ALREADY_EXISTS
527	The billing plan could not be previewed.	BILLING_PLAN_PREVIEW_ERROR
528	Value specified is not part of the list item values.	INVALID_LIST_VALUE
529	The Session Timeout value is not valid.	INVALID_SESSION_TIMEOUT_VALUE
530	Document upload not allowed.	DOCUMENT_UPLOAD_NOT_ALLOWED
531	Only PDF document upload allowed.	DOCUMENT_UPLOAD_ONLY_PDF_ALLOWED
532	PDF document upload not allowed.	DOCUMENT_UPLOAD_PDF_NOT_ALLOWED
533	Template upload not allowed.	TEMPLATE_UPLOAD_NOT_ALLOWED
534	Not all 'sender required fields' have been populated.	SENDER_REQUIRED_FIELD_NOT_SATISFIED
535	Sender required fields cannot be deleted from envelope.	SENDER_REQUIRED_FIELD_CANNOT_BE_DELETED
536	Property 'sender required' can only be set/modified on 'Text' or 'Drop Down' tags on a template.	SENDER_REQUIRED_FIELD_NOT_ALLOWED
537	A required envelope custom field is missing.	ENVELOPE_CUSTOM_FIELD_MISSING
538	The account is not managed by the distributor.	ACCOUNT_IS_NOT_MANAGED_BY_DISTRIBUTOR
539	Recipient is not a bulk recipient.	BULK_ENVELOPE_RECIPIENT_NOT_BULK_RECIPIENT
540	Field not supported for bulk recipient.	BULK_ENVELOPE_FIELD_NOT_SUPPORTED
541	Required field is missing.	BULK_ENVELOPE_REQUIRED_FIELD_MISSING
542	Maximum number of bulk recipients exceeded.	BULK_ENVELOPE_MAX_RECIPIENTS_EXCEEDED
543	Invalid Bulk Recipient CSV list.	BULK_ENVELOPE_INVALID_CSV_LIST
544	Invalid line in Bulk Recipient CSV list.	BULK_ENVELOPE_INVALID_CSV_ROW
545	Invalid characters.	BULK_ENVELOPE_INVALID_CHARACTERS
546	Invalid empty value.	BULK_ENVELOPE_EMPTY_VALUE
547	Maximum Length Exceeded for value.	BULK_ENVELOPE_MAX_VALUE_LENGTH_EXCEEDED
548	Invalid Email.	BULK_ENVELOPE_INVALID_EMAIL

Number	Message	Error Code
549	Recipient Phone Authentication not enabled for account.	BULK_ENVELOPE_INVALID_PERMISSION_PHONE_AUTH
550	Invalid phone number.	BULK_ENVELOPE_INVALID_PHONE,
551	Recipient Social Authentication not enabled for account.	BULK_ENVELOPE_INVALID_PERMISSION_SOCIAL_AUTH
552	Invalid authentication identifier.	BULK_ENVELOPE_INVALID_AUTH_ID
553	Bulk Recipient must have an account.	BULK_ENVELOPE_MUST_HAVE_ACCOUNT
554	Bulk Recipient duplicates found.	BULK_ENVELOPE_DUPLICATE_RECIPIENTS
555	Fail to delete bulk send file.	BULK_ENVELOPE_FILE_DELETE_FAILED
556	The requested envelope is not a draft or template.	BULK_ENVELOPE_NOT_DRAFT_OR_TEMPLATE
557	Template that is tied to a powerform is not allowed to bulk send.	BULK_ENVELOPE_TEMPLATE_IS_TIED_TO_POWERFORM
558	Bulk recipient envelope doesn't have bulk recipients.	BULK_ENVELOPE_HAS_NO_BULK_RECIPIENTS
559	Recipient SMS Authentication not enabled for account.	BULK_ENVELOPE_INVALID_PERMISSION_SMS_AUTH
560	Recipient SAML Authentication not enabled for account.	BULK_ENVELOPE_INVALID_PERMISSION_SAML_AUTH
561	Maximum number of bulk envelope exceeded.	BULK_ENVELOPE_MAX_ENVELOPE_EXCEEDED
562	Fail to send bulk envelopes.	BULK_ENVELOPE_SEND_FAILED
563	Batch Id is invalid.	BULK_ENVELOPE_INVALID_BATCHID
564	This Account lacks bulk send permissions.	BULK_ENVELOPE_ACCOUNT_LACKS_PERMISSIONS
565	This User lacks bulk send permissions.	BULK_ENVELOPE_USER_LACKS_PERMISSIONS
566	The watermark is invalid.	INVALID_WATERMARK
567	Watermark could not be updated.	WATERMARK_UPDATE_FAILED
568	Watermark is disabled.	WATERMARK_DISABLED
569	There is an error generating Watermark preview.	INVALID_WATERMARK_PREVIEW
570	There is an error selecting the specified discount.	DISCOUNT_FAILED
571	The subscription was not found.	SUBSCRIPTION_NOT_FOUND
572	The subscription failed.	SUBSCRIPTION_FAILED
573	There is an error with the subscription.	SUBSCRIPTION_ERROR
574	The currency is not valid in the system.	INVALID_CURRENCY
575	Recipient denied access to documents.	POWERFORMS_RECIPIENT_DENIED_DOCUMENTS
576	Invalid envelope for recipient.	POWERFORMS_INVALID_ENVELOPEID_FOR_RECIPIENT
577	Shared tags are not allowed because a digital certificate is required for a signer.	POWERFORMS_DIGITALCERTS_SHARED_TABS_NOT_ALLOWED

Number	Message	Error Code
578	Signers that are required to use a digital certificate must have at least one required, non-conditional Signature or Initials tab.	POWERFORMS_DIGITALCERTS_FREE_FORM_TABS_NOT_ALLOWED
579	Signers that are required to use a digital certificate must be the only recipient in a routing order. Please edit the routing order or remove the digital certificate requirement.	POWERFORMS_DIGITALCERTS_MULTIPLE_RECIPIENTS_ROUTING_ORDER
580	Document markup is not allowed because a digital certificate is required for a signer.	POWERFORMS_DIGITALCERTS_MARKUP_NOT_ALLOWED
581	Recipient UserName, Email or Role not set.	POWERFORMS_INCOMPLETE_RECIPIENT
582	PowerFormId is required.	POWERFORMS_POWERFORMID_REQUIRED
583	PowerFormId mismatch.	POWERFORMS_POWERFORMID_MISMATCH
584	TemplateId is required.	POWERFORMS_TEMPLATEID_REQUIRED
585	The PowerForm specified was not found	POWERFORMS_POWERFORMID_NOT_FOUND
586	PowerForms is not enabled on the account.	POWERFORMS_NOT_ENABLED
587	User lacks permission, not a PowerForms admin.	POWERFORMS_NOT_ADMIN
588	Direct signing mode is not enabled for PowerForms.	POWERFORMS_DIRECT_NOT_ENABLED
589	User lacks permission, not a PowerForms user.	POWERFORMS_NOT_USER
590	Signing mode is required.	POWERFORMS_SIGNINGMODE_REQUIRED
591	Uses remaining is required when max use is enabled.	POWERFORMS_USESREMAINING_REQUIRED
592	Fax usage could not be recorded.	FAX_USAGE_NOT_RECORDED
593	The email domain change was not initiated.	DOMAIN_CHANGE_FAILED
594	The domain change was not successful.	DOMAIN_CHANGE_INTERRUPTED
595	The domain change may have been partially successful.	DOMAIN_CHANGE_PARTIALLY_APPLIED
596	The address is required.	REQUIRED_ADDRESS
597	The role does not exist.	ROLE_DOES_NOT_EXIST
598	Invalid Salesforce credentials.	INVALID_SALESFORCE_CREDENTIALS
599	Invalid Connect ID.	INVALID_CONNECT_ID
600	This is already any existing Salesforce Connect integration.	EXISTING_SALESFORCE_CONNECT
601	Connect configuration cannot change type after creation.	CONNECT_CANNOT_CHANGE_TYPE
602	CustomField1 maximum length Exceeded.	LENGTH_EXCEEDED_RECIPIENT_CUSTOMFIELD1
603	CustomField2 Maximum length Exceeded.	LENGTH_EXCEEDED_RECIPIENT_CUSTOMFIELD2
604	CustomField3 maximum length Exceeded.	LENGTH_EXCEEDED_RECIPIENT_CUSTOMFIELD3

Number	Message	Error Code
605	ExternalUserName maximum length Exceeded.	LENGTH_EXCEEDED_RECIPIENT_EXTERNALUSERNAME
606	The envelope is already locked.	EDIT_LOCK_ENVELOPE_ALREADY_LOCKED
607	The template is already locked.	EDIT_LOCK_TEMPLATE_ALREADY_LOCKED
608	Failed to acquire the lock.	EDIT_LOCK_FAILED_TO_ACQUIRE_LOCK
609	Failed to acquire the lock because the envelope status was not 'created', 'sent' or 'delivered'.	EDIT_LOCK_INVALID_STATE_FOR_LOCK
610	Invalid lock type.	EDIT_LOCK_INVALID_LOCK_TYPE
611	The envelope is not locked.	EDIT_LOCK_ENVELOPE_NOT_LOCKED
612	The template is not locked.	EDIT_LOCK_TEMPLATE_NOT_LOCKED
613	The user is not the owner of the lock.	EDIT_LOCK_NOT_LOCK_OWNER
614	Invalid lock duration value.	EDIT_LOCK_INVALID_LOCK_DURATION
615	Failed to update the lock.	EDIT_LOCK_FAILED_TO_UPDATE_LOCK
616	Invalid LockedByApp value.	EDIT_LOCK_INVALID_LOCKEDBYAPP
617	Locking for edit is not enabled.	EDIT_LOCK_NOT_ENABLED
618	Edit lock error.	EDIT_LOCK_ERROR
619	Error creating the scratchpad.	EDIT_SCRATCHPAD_CREATE_ERROR
620	Scratchpad not enabled.	EDIT_SCRATCHPAD_NOT_ENABLED
621	The envelope is locked.	EDIT_LOCK_ENVELOPE_LOCKED
622	The template is locked.	EDIT_LOCK_TEMPLATE_LOCKED
623	Document update is not allowed.	DOCUMENT_UPDATE_NOT_ALLOWED
624	The recipient could not be deleted.	RECIPIENT_DELETE_FAILED
625	Account does not have permission to send to InPersonSigner recipient type.	CANNOT_SEND_TO_INPERSONSIGNER
626	Cannot add the document for the envelope.	CANNOT_ADD_DOCUMENT
627	The account lacks sufficient permissions to correct tabs.	ACCOUNT_LACKS_PERMISSION_TO_CORRECT_TABS
628	Bulk envelope only allows one bulk recipient.	BULK_ENVELOPE_ONLY_ONE_BULK_RECIPIENT_ALLOWED
629	SMS authentication phone number cannot be empty.	BULK_ENVELOPE_SMS_AUTH_PHONE_MISSING
630	Either permission profile Id or permission profile name is invalid is invalid.	PERMISSION_PROFILE_ID_DOES_NOT_MATCH_PERMISSION_PROFILE_NAME
631	Insufficient Account Provisioning Distributor Seats.	ACCOUNT_PROVISIONER_INSUFFICIENT_SEATS
632	Invalid Admin NRDS ID	ACCOUNT_PROVISIONER_INVALID_ADMIN_NRDS
633	Invalid template, the first recipient in the routing order must be a role recipient.	POWERFORMS_TEMPLATE_FIRST_RECIPIENT
634	Access Code provided is not in the correct format.	ACCESS_CODE_IN_WRONG_FORMAT

Number	Message	Error Code
635	Invalid Member Group ID	INVALID_MEMBER_GROUP_ID
636	NAR not enabled for user.	NAR_NOT_ENABLED_FOR_USER
637	NAR not able to validate NRDs ID and NRDs Last Name.	NAR_UNABLE_TO_VALIDATE_CREDENTIALS
638	NAR user not active.	NAR_USER_NOT_ACTIVE
639	There are users still associated to this permission profile.	USERS_IN_PERMISSION_PROFILE
640	The permission profile name is invalid or already in use.	INVALID_PERMISSION_PROFILE_NAME
641	The settings for the account role are invalid.	INVALID_ACCOUNT_ROLE_SETTINGS
642	A Required field is incomplete.	REQUIRED_TAB_INCOMPLETE
643	Error saving the scratchpad changes to the database.	EDIT_SCRATCHPAD_SAVE_ERROR
644	CSV file contains blank header fields.	BULK_ENVELOPE_BLANK_CSV_HEADER_FIELD
645	Allow Shared Tabs is not enabled on account.	REQUIRE_ALL_SHARED_TAB_NOT_ENABLED
646	Shared must be set to 'true' when setting 'Require All' to 'true'.	REQUIRE_ALL_SHARED_TAB_NOT_SET
647	Shared view links not allowed.	SHARED_VIEW_LINK_NOT_ALLOWED
648	Invalid Client Id provided.	PARTNER_INVALID_CLIENT_ID
649	Partner application creation not allowed in this environment.	PARTNER_APP_CREATION_NOT_ALLOWED
650	Partner application Name is required.	PARTNER_APP_NAME_REQUIRED
651	Partner application Redirect URI is invalid.	PARTNER_APP_REDIRECT_URI_INVALID
652	The group name is invalid.	INVALID_GROUP_NAME
653	SMS authentication phone number cannot be usersupplied.	BULK_ENVELOPE_SMS_AUTH_PHONE_INVALID
654	No Security Appliances configured on account.	NO_SECURITY_APPLIANCES_CONFIGURED
655	The document could not be added/updated.	DOCUMENT_UPDATE_FAILED
656	The page image is not available.	PAGE_IMAGE_NOT_RETRIEVED
657	File is invalid.	API_DOCGEN_INVALIDFILE
658	File does not exist.	API_DOCGEN_FILE_NOT_EXIST
659	File could not be read.	API_DOCGEN_FILE_READ
660	File could not be created.	API_DOCGEN_FILE_CREATE
661	File does not contain API documentation.	API_DOCGEN_FILE_NO_VALUE
662	The workspace does not exist or permission denied.	WORKSPACE_DOES_NOT_EXIST
663	Workspace name is invalid.	INVALID_WORKSPACE_NAME
664	Invalid signing group supplied.	SIGNING_GROUP_INVALID
665	Captive recipient not allowed to be a signing group.	SIGNING_GROUP_CAPTIVE_RECIPIENT_NOT_ALLOWED
666	Maximum signing groups reached.	SIGNING_GROUP_MAX_REACHED
667	Maximum users in a signing group reached.	SIGNING_GROUP_MAX_USERS_REACHED

Number	Message	Error Code
668	Fax delivery is not allowed with signing groups.	SIGNING_GROUP_FAX_DELIVERY_NOT_ALLOWED
669	SMS authentication is not allowed with signing groups.	SIGNING_GROUP_SMS_AUTH_NOT_ALLOWED
670	The user merge failed	USER_MERGE_FAILED
671	Invalid MergeUserId.	INVALID_MERGEUSERID
672	The Page TransactionId is invalid.	INVALID_PAGE_TRANSACTIONID
673	PDF was not found.	PDF_NOT_FOUND
674	Get page image failed.	GET_PAGE_IMAGE_FAILED
675	The API Credential is null.	CUSTOM_TAB_APICREDENTIAL_IS_NULL
676	Custom Tab update failed.	CUSTOM_TAB_UPDATE_FAILED
677	Custom Tab create failed.	CUSTOM_TAB_CREATE_FAILED
678	Invalid related field sequence.	INVALID_RELATED_FIELD
679	Unable to connect to the service.	FAILED_TO_CONNECT_TO_SERVICE
680	The Custom Tab is null.	CUSTOM_TAB_IS_NULL
681	Invalid RequestRecipientTokenClientURLs value. The clientUrls argument is null.	CAPTIVE_RECIPIENT_TOKEN_CLIENTURLS_NULL

Appendix 1: Time Zones and Date/Time Format Information

This appendix provides information on how time zone and date/time format information is set for an account and how it appears in the DocuSign web console, Certificate of Completion, and API responses. The appendix also has information on the expected date/time formats for API requests.

Account Settings

There are a number of account settings that determine the time zone and date/time format used for an account. These options are set in the DocuSign web console Preferences – Features; some can be set through the API account settings.

- **Default Account Time Zone:** This option selects the default time zone to be used in the User Interface display. A member of the account can set their own time zone preference to override the account default if the Allow account member to set their own unique time zone option (see below) is enabled.
- **Time Zone Used For API:** This option selects the default time zone used for DocuSign SOAP API operations.
- **Format for Date Signed:** This option selects the date format applied to the Date Signed tags.
- **Format for Time Signed:** This option selects the time format applied to the Date Signed tags. You can also select to **Include AM/PM designation** for the time format.
- **Allow account member to set their own unique time zone:** When selected, account members can set their own personal time zone, which can be different that the default account time zone.

If the **Allow account member to set their own unique time zone** option is selected, there are two options the account members can set by a user:

- **My local time zone:** This option selects the default time zone to be used in the account member's DocuSign web console.
- **Format my local date and time:** This option selects the date/time format used in the account member's DocuSign web console and the format used when the account member signs envelopes.

Time Zone and Date/Time Format Appearance

How the time zone and date/time format appears depends on what is being viewed, who is viewing the information, and the settings for the account.

Signing user interface and PDF documents: The time zone and date/time format information shown in Date tags and form data in the signing user interface and PDF documents depends on the signer:

- If the signer has a DocuSign account, then the signer's time zone and the sender's date/time format settings are used for the information. If the sender's account does not allow users to set their own date/time format, then the date/time format setting for the account is used.
- If the signer does not have a DocuSign account, then the sender's time zone and date/time format settings are used. If the sender's account does not allow users to set their own time zones or date/time format, then the account settings are used.

Certificate of Completion: The sender's time zone and date/time format settings are used. If the sender's account does not allow users to set their own time zone and date/time format, then the account time zone and date/time format settings are used.

Envelope History: When viewing History through the web console or signing user interface, the time zone and date/time format used depends on the viewer:

- If the sender is viewing the History, the sender's time zone and date/time format settings are used. If the sender's account does not allow users to set their own time zones or date/time format, then the account settings are used.
- If the viewer has a DocuSign account that allows users to set their own time zone and date/time format, then the viewer's time zone and date/time format settings are used.
- If the viewer has a DocuSign account that does not allow users to set their own time zone and date/time format, then the account time zone and date/time format settings are used.
- If the viewer does not have a DocuSign account, then the sender's time zone and date/time format are used. If the sender's account does not allow users to set their own time zones or date/time format, then the account settings are used.

REST API: When getting information from the REST API, all non-PDF time zone responses are returned in ISO 8601 date/time format using GMT as the time zone. PDF items retrieved through the REST API, such as a Certificate of Completion or documents with date tags, use Certificate of Completion and PDF documents settings previously described.

SOAP API: When getting information from the SOAP API, all non-PDF items use the time zone set by the **Time Zone Used For API** option. PDF items retrieved through the SOAP API, such as a Certificate of Completion or documents with date tags, use Certificate of Completion and PDF documents settings previously described.

Date/Time Formats for API Calls

All DocuSign SOAP and REST API requests must use ISO 8601 date/time formats. The REST API assumes that all values passed represent UTC date/times.

When providing a date/time format for the DocuSign REST API, the preferred formats are:

- "yyyy-MM-dd | HH:mm"
- "MMMM d, yyyy | HH:mm"
- "MMM-dd-yyyy | HH:mm"
- "dd-MM-yyyy | HH:mm"