

ADOBE SIGN COMPONENT

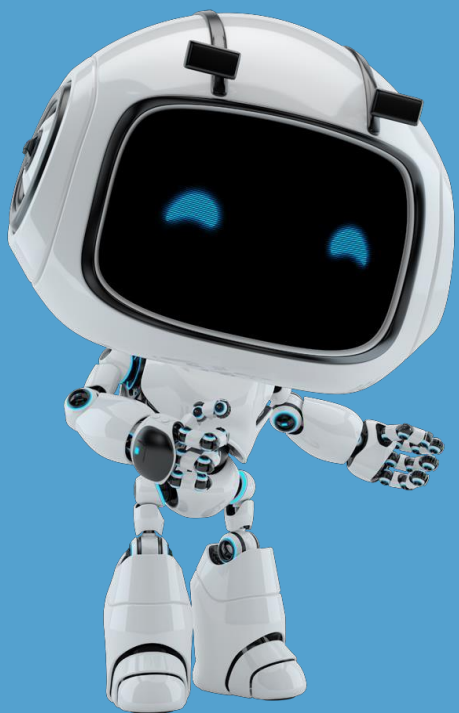


TABLE OF CONTENTS

Contents

TABLE OF CONTENTS	2
1 INTRODUCTION	4
1.2 ACTIVITIES	11
1.2.1 Get Refresh Token	12
1.2.2 Adobe Sign Scope	13
1.2.3 Revoke Token	14
1.2.4 Add Document	14
1.2.5 Send Document	15
1.2.6 Get Agreements	15
1.2.7 Get Agreement	17
1.2.8 Send Reminder	25
1.2.9 Share Agreement	26
1.2.10 Get Signing URL	27
1.2.11 Download Agreement	28
1.2.12 Update Agreement Visibility	28
1.2.13 Update Agreement State	29
1.2.14 Delete Agreement	29
1.2.15 Get All Users	30
1.2.16 Get User	30
1.2.17 Get Groups of Users	32
1.2.18 Update User Details	33
1.2.19 Update User State	34
1.2.20 Get All Groups	34

1.2.21	Get Group.....	35
1.2.22	Get Users of Group	35
1.2.23	Get Message Template.....	37
1.2.24	Get Account Information	37
1.2.25	Get All Workflows	38
1.2.26	Get Workflow	39
1.2.27	Get All Library Documents	45
1.2.28	Get Library Document.....	46
1.2.29	Update Document Visibility.....	48
1.2.30	Update Document State	48

1 INTRODUCTION

1.1 PURPOSE

Adobe Sign, an Adobe Document Cloud solution is a cloud-based, enterprise-class e-signature service that lets you replace paper and ink signature processes with fully automated electronic signature workflows. With it, you can easily send, sign, track, and manage signature processes using a browser or mobile device. And you can use turnkey integrations and APIs to include e-signature workflows in your enterprise apps and systems of record.

The activities in the Adobe Sign component developed for UiPath Studio allows the user to interact seamlessly with Adobe Sign APIs. **The activities are present in 5 different languages which include English, Japanese, Chinese, Korean and French.**

The Adobe Sign component contains **30 activities** (will add more activities in future) that allows user to perform various operations like Creating a signed document, sending the document for eSign, tracking the document status, sending recurring reminders to participants, get the signing URL etc.

To be able to use the component, you first need to setup your account on adobe website.

Following are the steps to get your developer account

- 1 **Developer Sign Up:** Log on to <https://acrobat.adobe.com/in/en/sign/developer-form.html>

Sign up to create your account.

<p>First name</p> <input type="text" value="John"/>	<p>Last name</p> <input type="text" value="Smith"/>
<p>Business email</p> <input type="text" value="name@companyemail.com"/>	<p>Phone</p> <input type="text" value="+1 888.555.1000"/>
<p>Company</p> <input type="text" value="Company name"/>	<p>Job title</p> <input type="text" value="Title"/>
<p>Industry</p> <input type="text" value="Select_"/>	<p>Select company size</p> <input type="text" value="Select_"/>
<p>Company website</p> <input type="text" value="www.yoursite.com"/>	<p>Company HQ country/region</p> <input type="text" value="Select_"/>

By clicking "Continue" I agree that:

- I have read and accepted the [Terms of Use](#).
- The Adobe family of companies may keep me informed with [personalized](#) emails about products and services.

See our [Privacy Policy](#) for more details or to opt-out at any time.

Please note: Partner communications are currently only available in English.

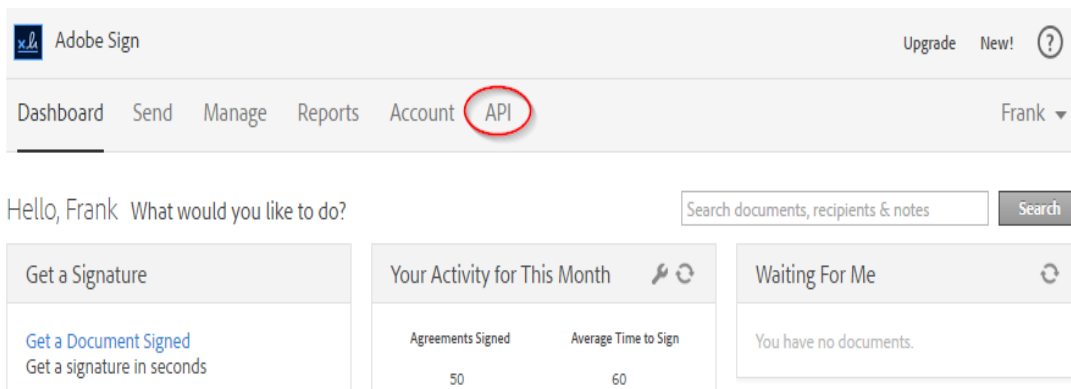
[Continue](#)

Fill in the required information and create your account. You will also need to confirm you email account by clicking on the link given in the email

2 Create an Application: To use Adobe sign APIs, you first need to create an application

Adobe sign uses the OAuth authentication protocol to authorize request for any sign API endpoint. So, you need to get the Application ID and Application Secret from the web UI of Adobe Sign.

- 1) Log in to Adobe Sign.
- 2) Select API from the top menu.



If you are already an Enterprise customer, you may not see the API link. In that case, click Account to proceed.

3) Select API Applications.

The screenshot shows the Adobe Sign API Information page. On the left is a navigation sidebar with a search bar and several menu items: Personal Preferences, Users, Groups, Account Settings, Workflows, Adobe Sign API, API Information, API Request Log, and API Applications (which is circled in red). The main content area is titled 'API Information' and includes an 'API Introduction' section with text about integrating with Adobe Sign REST API, instructions on creating an application, and a 'Documentation' section with links to REST API Methods Documentation and Downloadable REST API samples.

4) Select the Create (+) icon and provide details about your app.

The 'Create' dialog box prompts the user to provide a name for the application. It contains two text input fields, both containing 'My First App', for 'Name' and 'Display Name'. Below these is a 'Domain' section with two radio button options: 'CUSTOMER (This application will only have access to data within your account)' and 'PARTNER (This application will have access to any authorized Adobe Sign account)'. At the bottom right are 'Cancel' and 'Save' buttons.

When you create a new app, you need to choose the right domain:

- 2.1.1 CUSTOMER:** Apps for internal use and testing. Use this domain if you need your app to access data only from your account.
- 2.1.2 PARTNER:** Apps for production and public use. Use this domain if you need your app to access data in any Adobe Sign account.

3 Create OAuth for the Application

In the previous section, you learned to create an application. Now, you must configure OAuth for your application so that the client-side applications that you build can invoke the Sign APIs.

- (a) Select API Applications to view the list of apps you have created; then select your app to view its action menu.
- (b) Select View/Edit to get the Application ID and secret.

View / Edit ✕

Application ID:
[REDACTED]

Client Secret:
[REDACTED]

Created:
05/14/2018 03:52

Created By:
Matthew Lusher ([REDACTED])

Last Updated:
05/14/2018 03:52

Name:

Display Name:

Domain:
CUSTOMER (This application will only have access to data within your account)

Status:
ACTIVE

Note down the App's Application ID and Client Secret. You will be using this information to issue access tokens in the Adobe Sign API.

(c) Select Configure OAuth for Application

Modifier	Description
self	Perform the specified action on behalf of the authorizing user. This is the default: for example, "agreement_send:self" is the same scope as "agreement_send"
group	Perform the specified action on behalf of any user in the same group as the authorizing user. The authorizing user must be a group admin in order to grant this scope, and must have the Business or Enterprise edition of Adobe Sign.
account	Perform the specified action on behalf of any user in the same account as the authorizing user. The authorizing user must be an account admin in order to grant this scope, and must have the Business or Enterprise edition of Adobe Sign.

4 Get Authorization Code

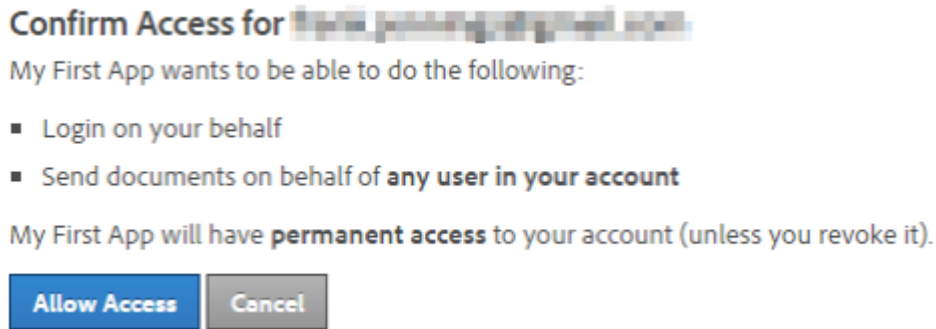
Paste the below URL in browser to get authorization code

https://secure.in1.echosign.com/public/oauth?redirect_uri=yourRedirectURL&response_type=code&client_id=yourclientid&scope=user_login:account agreement_send:account agreement_write:account agreement_read:account user_read:account user_write:account workflow_read:account library_read:account library_write:account

This URL contains following parameters

- Redirect URI: Redirect URL from step 3
- Response Type: it should always be set to "Code"
- Client id: Client ID from step 3
- Scope: Keep the value as it is

Once you submit this URL, you will get the following screen (if you are not logged in then you need to login first using admin account)



Click on Allow Access Button and you will be redirected to a new URL

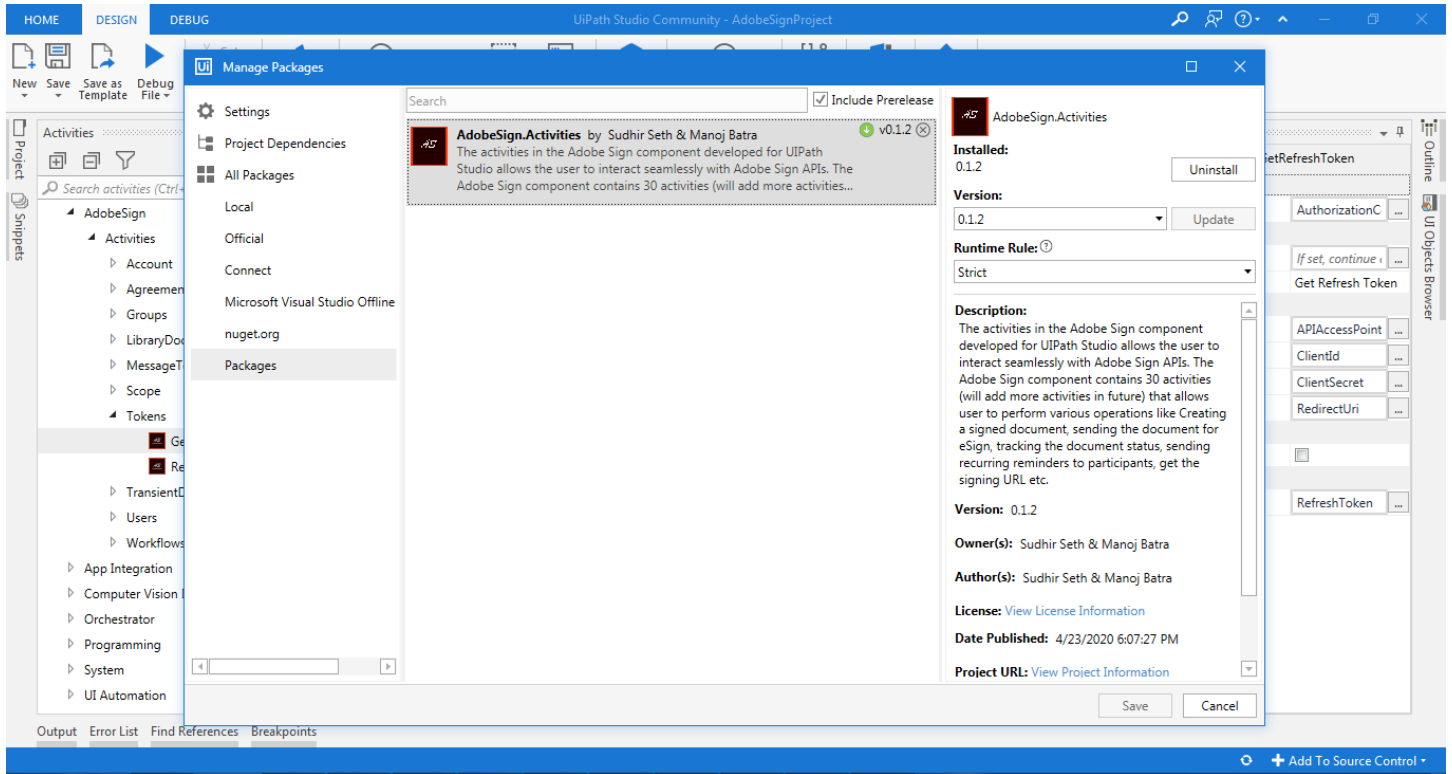
```
https://mysite.com/?code=xyzhdfkhdkfdjskhfdsyiehkdsjdkshfkdskfdhkdsf&
api_access_point=https://api.in1.echosign.com&
web_access_point=https://secure.in1.echosign.com
```

Copy the code (also abbreviated as authorization code), api_access_point and save it somewhere. These are your secure codes and are valid for only 5 minutes. These values will act as input for fetching the refresh token to work with APIs. If you failed to generate the refresh token within 5 minutes then you need to generate the authorize code again.

1.2 ACTIVITIES

To use the Adobe Sign Component in UiPath one should add the package from Manage Packages window in UiPath Studio.

For the same open UiPath Studio, click on Manage Packages button and add the package named **AdobeSign.Activities**.



The adobe sign activities are further categorized into 10 different groups and these groups are following:

1. Account
2. Agreements
3. Groups
4. Library Documents
5. Message Templates
6. Scope
7. Tokens
8. Transient Documents
9. Users
10. Workflows

1.2.1 Get Refresh Token

This activity is used to get the refresh token to work on the Adobe Sign Activities. **This is one-time activity**. The refresh token is generated from this activity is valid for lifetime but it may get expired if the refresh token is not used for 60 days. You need to regenerate the authentication code again to get this token again. Following are the input and output parameters

Input:

- API Access Point* (String): API access point received from Get Authorization Code section (point 4)

- Authorization Code* (String): Authorization code received from Get Authorization code section (point 4)
- Client Id* (String): Client Id from OAuth step (point 3)
- Client Secret* (String): Client Secret from OAuth step (point 3)
- Redirect URL* (String): Redirect URL from OAuth (point 3)

Output

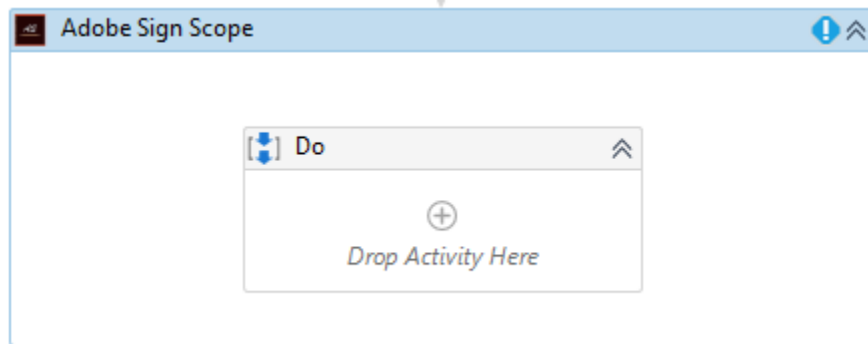
- Refresh Token (String): This property will contain the refresh token. You should save it somewhere to work with Adobe Sign Activities. This refresh token will act as input parameter for generating the access token. Only one refresh token can be generated with one authorization code. In order to generate the refresh token again, you need to generate the authorization code again.

1.2.2 Adobe Sign Scope

This activity is the parent activity of all Adobe Sign activities. Following are the input parameters

Input:

- API Access Point* (String): API access code received from Get Authorization Code section (point 4)
- Refresh Token* (String): Refresh token generated from the **Get Refresh Token** activity
- Client Id* (String): Client Id from OAuth step (point 3)
- Client Secret* (String): Client Secret from OAuth step (point 3)
- Redirect URL* (String): Redirect URL from OAuth (point 3)

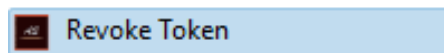


1.2.3 Revoke Token

Access as well as refresh token can be revoked. If an access token is revoked and it has a corresponding refresh token, the refresh token will also be revoked. When a refresh token is revoked, all the access token issued from that refresh token also gets revoked. Following are the parameters for this activity

Output:

- Status (Boolean): Status of the token revoked



1.2.4 Add Document

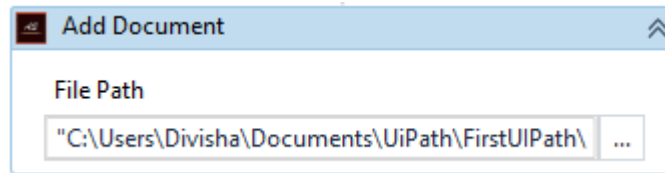
This activity uploads the document and returns the document's ID. The document uploaded through this call is referred to as transient since it is available only for 7 days after the upload. The returned transient document ID can be used in the API calls where the uploaded file needs to be referred. Following are the parameters for this activity

Input:

- File Path* (String): The file part of the multipart request for document upload. You can upload only one file at a time

Output:

- Transient Document Id (String): The unique identifier of the uploaded document that can be used in an agreement.



1.2.5 Send Document

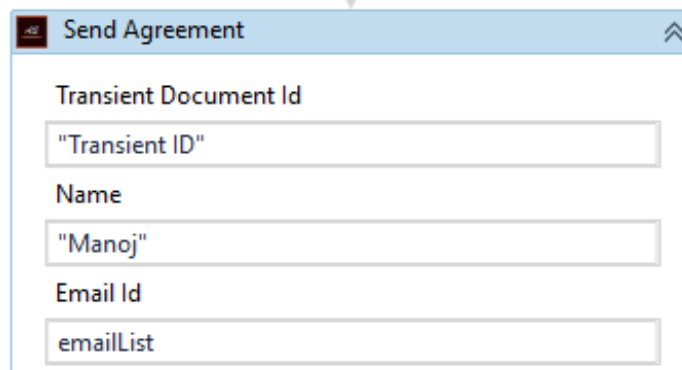
This activity creates an agreement. Sends it out for signatures and returns the Agreement Id in response. Following are the parameters for this activity

Input:

- Transient Document Id* (String): The unique identifier of the uploaded document that can be used in an agreement
- Name of Agreement* (String): The name of the agreement that will be used to identify it, in emails, website and other places
- Email Id* (String []): Email Ids of recipients
- Role* (String): The possible values are 'SIGNER' or 'APPROVER' or 'ACCEPTOR' or 'CERTIFIED_RECIPIENT' or 'FORM_FILLER' or 'DELEGATE_TO_SIGNER' or 'DELEGATE_TO_APPROVER' or 'DELEGATE_TO_ACCEPTOR' or 'DELEGATE_TO_CERTIFIED_RECIPIENT' or 'DELEGATE_TO_FORM_FILLER' or 'SHARE'
- Signature Type* (String): Specifies the type of signature you would like to request - written or e-signature. The possible values are
ESIGN: Agreement needs to be signed electronically
WRITTEN: Agreement will be signed using handwritten signature and signed document will be uploaded into the system
- State* (String): The state in which the agreement should land. 'AUTHORING' or 'DRAFT' or 'IN_PROCESS' are valid values

Output:

- Id (String): The Unique identifier of the agreement.



1.2.6 Get Agreements

This activity is used to retrieve all agreements for the user in JSON format. Following is the parameter for this activity

Output:

- UserAgreements (String): All the agreements in the following JSON format

```
{
  "page": {
    "nextCursor": ""
  },
  "userAgreementList": [
    {
      "displayDate": "",
      "displayParticipantSetInfos": [
        {
          "displayUserSetMemberInfos": [
            {
              "email": "",
              "company": "",
              "fullName": ""
            }
          ],
          "displayUserSetName": ""
        }
      ],
      "esign": false,
      "groupId": "",
      "hidden": false,
      "latestVersionId": "",
      "name": "",
      "id": "",
      "parentId": "",
      "status": "",
      "type": ""
    }
  ]
}
```


UserAgreement {

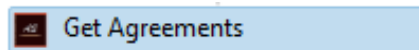
displayDate (string): The display date for the agreement. Format would be yyyy-MM-dd'T'HH:mm:ssZ. For example, e.g 2016-02-25T18:46:19Z represents UTC time,
displayParticipantSetInfos ([DisplayParticipantSetInfo\[\]](#)): The most relevant current user set for the agreement. It is typically the next signer if the agreement is from the current user, or the sender if received from another user,
esign (boolean): True if this is an e-sign document,
groupId (string): Unique identifier of the group,
hidden (boolean): True if agreement is hidden for the user,
latestVersionId (string): A version ID which uniquely identifies the current version of the agreement,
name (string): Name of the Agreement,
id (string, *optional*): The unique identifier of the agreement.If provided in POST, it will simply be ignored,
parentId (string, *optional*): The parent id of a megaSign child *OR* the originating id of a widget agreement instance,
status (string, *optional*) = ['WAITING_FOR_MY_SIGNATURE' or 'WAITING_FOR_MY_APPROVAL' or 'WAITING_FOR_MY_DELEGATION' or 'WAITING_FOR_MY_ACKNOWLEDGEMENT' or 'WAITING_FOR_MY_ACCEPTANCE' or 'WAITING_FOR_MY_FORM_FILLING' or 'OUT_FOR_SIGNATURE' or 'OUT_FOR_APPROVAL' or 'OUT_FOR_DELIVERY' or 'OUT_FOR_ACCEPTANCE' or 'OUT_FOR_FORM_FILLING' or 'SIGNED' or 'APPROVED' or 'FORM_FILLED' or 'DELIVERED' or 'ACCEPTED' or 'ARCHIVED' or 'CANCELLED' or 'EXPIRED' or 'WAITING_FOR_AUTHORIZING' or 'WAITING_FOR_PREFILL' or 'DRAFT' or 'DOCUMENTS_NOT_YET_PROCESSED' or 'WAITING_FOR_MY_VERIFICATION' or 'WAITING_FOR_VERIFICATION']: This is a server generated attribute which provides the detailed status of an agreement with respect to the apiCaller,
type (string, *optional*) = ['AGREEMENT' or 'MEGASIGN_CHILD' or 'WIDGET_INSTANCE']: The kind of agreement

DisplayParticipantSetInfo {

displayUserSetMemberInfos ([DisplayParticipantInfo\[\]](#)): Displays the info about user set,
displayUserSetName (string, *optional*): The name of the display user set. Returned only, if the API caller is the sender of agreement.

DisplayParticipantInfo {

email (string): Displays the email of the user,
company (string, *optional*): Displays the name of the company of the user, if available,
fullName (string, *optional*): Displays the full name of the user, if available.



1.2.7 Get Agreement

This Activity returns the current status of an agreement. Following are the input and output parameters of the activity

Input:

- Agreement Id* (String): Unique Agreement Id

Output:

- Agreement Information (String): Contains all Information related to agreement in the following JSON format

```
{  
  "name": "",  
  "participantSetsInfo": [  

```

```

{
  "memberInfos": [
    {
      "email": "",
      "securityOption": {
        "authenticationMethod": "",
        "password": "",
        "phoneInfo": {
          "countryCode": "",
          "countryIsoCode": "",
          "phone": ""
        }
      }
    }
  ],
  "order": 1,
  "role": "",
  "label": "",
  "name": "",
  "privateMessage": "",
  "visiblePages": [
    ""
  ]
}
],
"signatureType": "",
"status": "",
"ccs": [
  {
    "email": "",
    "label": "",
    "visiblePages": [
      ""
    ]
  }
]
},
"createdDate": "",
"deviceInfo": {
  "applicationDescription": "",
  "deviceDescription": "",
  "deviceTime": ""
},

```

```
"documentVisibilityEnabled": false,
"emailOption": {
  "sendOptions": {
    "completionEmails": "",
    "inFlightEmails": "",
    "initEmails": ""
  }
},
"expirationTime": "",
"externalId": {
  "id": ""
},
"fileInfos": [
  {
    "document": {
      "id": "",
      "label": "",
      "numPages": 1,
      "mimeType": "",
      "name": ""
    },
    "label": "",
    "libraryDocumentId": "",
    "transientDocumentId": "",
    "urlFileInfo": {
      "mimeType": "",
      "name": "",
      "url": ""
    }
  }
],
"firstReminderDelay": 1,
"formFieldLayerTemplates": [
  {
    "document": {
      "id": "",
      "label": "",
      "numPages": 1,
      "mimeType": "",
      "name": ""
    },
    "label": "",
```

```

    "libraryDocumentId": "",
    "transientDocumentId": "",
    "urlFileInfo": {
      "mimeType": "",
      "name": "",
      "url": ""
    }
  },
  "groupId": "",
  "hasFormFieldData": false,
  "hasSignerIdentityReport": false,
  "id": "",
  "isDocumentRetentionApplied": false,
  "locale": "",
  "mergeFieldInfo": [
    {
      "defaultValue": "",
      "fieldName": ""
    }
  ],
  "message": "",
  "parentId": "",
  "postSignOption": {
    "redirectDelay": 1,
    "redirectUrl": ""
  },
  "reminderFrequency": "",
  "securityOption": {
    "openPassword": ""
  },
  "senderEmail": "",
  "state": "",
  "type": "",
  "vaultingInfo": {
    "enabled": false
  },
  "workflowId": ""
}

```

JSON Description

Key	Description
-----	-------------

name (string)	The name of the agreement that will be used to identify it, in emails, website and other places
participantSetInfo (ParticipantSetInfo[])	A list of one or more participant set. A participant set may have one or more participant. If any member of the participant set takes the action that has been assigned to the set(Sign/Approve/Acknowledge etc), the action is considered as the action taken by whole participation set. For regular (non-MegaSign) documents, there is no limit on the number of electronic signatures in a single document. Written signatures are limited to four per document
signatureType (string)	['ESIGN' or 'WRITTEN']: Specifies the type of signature you would like to request - written or e-signature. The possible values are ESIGN : Agreement needs to be signed electronically , WRITTEN : Agreement will be signed using handwritten signature and signed document will be uploaded into the system
status (string)	['OUT_FOR_SIGNATURE' or 'OUT_FOR_DELIVERY' or 'OUT_FOR_ACCEPTANCE' or 'OUT_FOR_FORM_FILLING' or 'OUT_FOR_APPROVAL' or 'AUTHORING' or 'CANCELLED' or 'SIGNED' or 'APPROVED' or 'DELIVERED' or 'ACCEPTED' or 'FORM_FILLED' or 'EXPIRED' or 'ARCHIVED' or 'PREFILL' or 'WIDGET_WAITING_FOR_VERIFICATION' or 'DRAFT' or 'DOCUMENTS_NOT_YET_PROCESSED' or 'WAITING_FOR_FAXIN' or 'WAITING_FOR_VERIFICATION']: This is a server generated attribute which provides the detailed status of an agreement.
createdDate (string, optional)	Date when agreement was created. This is a server generated attributed and cannot be provided in POST/PUT calls. Format would be yyyy-MM-dd'T'HH:mm:ssZ. For example, e.g 2016-02-25T18:46:19Z represents UTC time
createdDate (string, optional)	Date when agreement was created
deviceInfo (OfflineDeviceInfo, optional)	Device info of the offline device. It should only be provided in case of offline agreement creation
documentVisibilityEnabled (boolean, optional)	If set to true, enable limited document visibility
emailOption (EmailOption, optional)	Email configurations for the agreement
expirationTime (string, optional)	Time after which Agreement expires and needs to be signed before it
externalId (ExternalId, optional)	An arbitrary value from your system, which can be specified at sending time and then later returned or queried. Should not be provided in offline agreement creation.
fileInfos (FileInfo[], optional)	A list of one or more files (or references to files) that will be sent out for signature
firstReminderDelay	Integer which specifies the delay in hours before sending the first reminder

(integer, optional)	
formFieldLayerTemplates (FileInfo[], optional)	Specifies the form field layer template or source of form fields to apply on the files in this transaction
groupId (string, optional)	The unique identifier of the group to which the agreement belongs to
hasFormFieldData (boolean, optional)	True if form field data is present.
hasSignerIdentityReport (boolean, optional)	True if agreement has signer identity report available
id (string, optional)	The unique identifier of the agreement
isDocumentRetentionApplied (boolean, optional)	Is document retention applied for this resource
locale (string, optional)	The locale associated with this agreement - specifies the language for the signing page and emails
mergeFieldInfo (MergefieldInfo[], optional)	Optional default values for fields to merge into the document
message (string, optional)	An optional message to the participants, describing what is being sent or why their signature is required
parentId (string, optional)	The parent id of a megaSign child *OR* the originating id of a widget agreement instance
postSignOption (PostSignOption, optional)	URL and associated properties for the success page the user will be taken to after completing the signing process
reminderFrequency (string, optional)	['DAILY_UNTIL_SIGNED' or 'WEEKDAILY_UNTIL_SIGNED' or 'EVERY_OTHER_DAY_UNTIL_SIGNED' or 'EVERY_THIRD_DAY_UNTIL_SIGNED' or 'EVERY_FIFTH_DAY_UNTIL_SIGNED' or 'WEEKLY_UNTIL_SIGNED' or 'ONCE']
securityOption (SecurityOption, optional)	Optional secondary security parameters for the agreement
senderEmail (string, optional)	Email of agreement sender
state (string, optional)	['AUTHORING' or 'DRAFT' or 'IN_PROCESS']: The state in which the agreement should land
type (string, optional)	['AGREEMENT' or 'MEGASIGN_CHILD' or 'WIDGET_INSTANCE']
vaultingInfo (VaultingInfo, optional)	Vaulting properties that allows Adobe Sign to securely store documents with a vault provider
workflowId (string, optional)	The identifier of custom workflow which defines the routing path of an agreement

ParticipantSetInfo Description

ParticipantSetInfo {

memberInfos ([ParticipantInfo\[\]](#)): Array of ParticipantInfo objects, containing participant-specific data (e.g. email). All participants in the array belong to the same set, **order** (integer): Index indicating position at which signing group needs to sign. Signing group to sign at first place is assigned a 1 index. Different signingOrder specified in input should form a valid consecutive increasing sequence of integers. Otherwise signingOrder will be considered invalid. No signingOrder should be specified for SHARE role,

role (string) = ['SIGNER' or 'APPROVER' or 'ACCEPTOR' or 'CERTIFIED_RECIPIENT' or 'FORM_FILLER' or 'DELEGATE_TO_SIGNER' or 'DELEGATE_TO_APPROVER' or 'DELEGATE_TO_ACCEPTOR' or 'DELEGATE_TO_CERTIFIED_RECIPIENT' or 'DELEGATE_TO_FORM_FILLER' or 'SHARE']: Role assumed by all participants in the set (signer, approver etc.),

label (string, *optional*): The unique label of a participant set.

For custom workflows, label specified in the participation set should map it to the participation step in the custom workflow.,

name (string, *optional*): Name of the participant set (it can be empty, but needs not to be unique in a single agreement). Maximum no of characters in participant set name is restricted to 255,

privateMessage (string, *optional*): Participant set's private message - all participants in the set will receive the same message,

visiblePages (string[], *optional*): When you enable limited document visibility (documentVisibilityEnabled), you can specify which file (fileInfo) should be made visible to which specific participant set.

Specify one or more label values of a fileInfos element.

Each signer participant sets must contain at least one required signature field in at least one visible file included in this API call; if not a page with a signature field is automatically appended for any missing participant sets. If there is a possibility that one or more participant sets do not have a required signature field in the files included in the API call, all signer participant sets should include a special index value of '0' to make this automatically appended signature page visible to the signer. Not doing so may result in an error. For all other roles, you may omit this value to exclude this page.

ParticipantInfo {

email (string): Email of the participant. In case of creating new Agreements(POST/PUT), this is a required field. In case of GET, this is the required field and will always be returned unless it is a fax workflow(legacy agreements) that were created using fax as input,

securityOption ([ParticipantSecurityOption](#), *optional*): Security options that apply to the participant

ParticipantSecurityOption {

authenticationMethod (string) = ['NONE' or 'PASSWORD' or 'PHONE' or 'KBA' or 'WEB_IDENTITY' or 'ADOBE_SIGN' or 'GOV_ID']: The authentication method for the participants to have access to view and sign the document. When replacing a participant that has PASSWORD or PHONE authentication specified, you must supply a password or phone number for the new participant, and you cannot change the authentication method,

password (string, *optional*): The password required for the participant to view and sign the document. Note that AdobeSign will never show this password to anyone, so you will need to separately communicate it to any relevant parties. The password will not be returned in GET call. When replacing a participant that has PASSWORD authentication specified, you must supply a password for the new participant,

phoneInfo ([PhoneInfo](#), *optional*): The phoneInfo required for the participant to view and sign the document

PhoneInfo {

countryCode (string, *optional*): The numeric country calling code (ISD code) required for the participant to view and sign the document if authentication type is PHONE,

countryIsoCode (string, *optional*): The country ISO Alpha-2 code required for the participant to view and sign the document if authentication method is PHONE,

phone (string, *optional*): The phone number required for the participant to view and sign the document if authentication method is PHONE. When replacing a participant that has PHONE authentication specified, you must supply a phone number for the new participant.

CcInfo {

email (string): Email of the CC participant of the agreement,

label (string, *optional*): Label of the CC list as returned in workflow description,

visiblePages (string[], *optional*): When you enable limited document visibility (documentVisibilityEnabled), you can specify which file (fileInfo) should be made visible to which specific participant set.

Specify one or more label values of a fileInfos element.

Each signer participant sets must contain at least one required signature field in at least one visible file included in this API call; if not a page with a signature field is automatically appended for any missing participant sets. If there is a possibility that one or more participant sets do not have a required signature field in the files included in the API call, all signer participant sets should include a special index value of '0' to make this automatically appended signature page visible to the signer. Not doing so may result in an error. For all other roles, you may omit this value to exclude this page.

OfflineDeviceInfo {

applicationDescription (string): Application Description,

deviceDescription (string): Device Description,

deviceTime (string, *optional*): The device local time. The device time provided should not be before 30 days of current date. Format should be yyyy-MM-dd'T'HH:mm:ssZ. For example, e.g 2016-02-25T18:46:19Z represents UTC time

EmailOption {

sendOptions ([SendOptions](#), *optional*): Specify emails to be sent to different participants at different steps of the agreement process. Note: ALL means emails for the events will be sent to all participants. NONE means emails for the events will not be sent to any participant

SendOptions {

completionEmails (string, *optional*) = ['ALL' or 'NONE']: Control notification mails for agreement completion events - COMPLETED, CANCELLED, EXPIRED and REJECTED,

inFlightEmails (string, *optional*) = ['ALL' or 'NONE']: Control notification mails for agreement-in-process events - DELEGATED, REPLACED,

initEmails (string, *optional*) = ['ALL' or 'NONE']: Control notification mails for Agreement initiation events - ACTION_REQUESTED and CREATED

ExternalId {

id (string, *optional*): An arbitrary value from your system, which can be specified at sending time and then later returned or queried

FileInfo {

document ([Document](#), *optional*): A document that is associated with the agreement. This field cannot be provided in POST call. In case of GET call, this is the only field returned in the response,

label (string, *optional*): The unique label value of a file info element. In case of custom workflow this will map a file to corresponding file element in workflow definition. This must be specified in case of custom workflow agreement creation request,

libraryDocumentId (string, *optional*): ID for an existing Library document that will be added to the agreement,

transientDocumentId (string, *optional*): ID for a transient document that will be added to the agreement,

urlFileInfo ([URLFileInfo](#), *optional*): URL for an external document to add to the agreement

Document {

id (string): ID of the document. In case of PUT call, this is the only field that is accepted in Document structure. Name and mimeType are ignored in case of PUT call,

label (string): Label of the document,

numPages (integer): Number of pages in the document,

mimeType (string, *optional*): mimeType of the original file. This is returned in GET but not accepted back in PUT,

name (string, *optional*): Name of the original document uploaded. This is returned in GET but not accepted back in PUT

URLFileInfo {

mimeType (string): The mime type of the referenced file, used to determine if the file can be accepted and the necessary conversion steps can be performed,
name (string): The original system file name of the document being sent,
url (string): A publicly accessible URL for retrieving the raw file content

MergefieldInfo {

defaultValue (string): The default value of the field,
fieldName (string): The name of the field

PostSignOption {

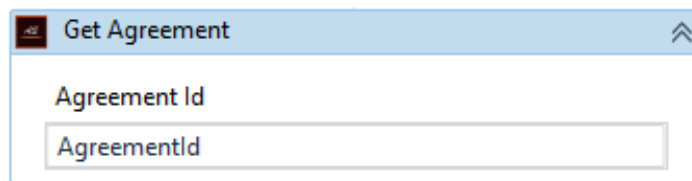
redirectDelay (integer, *optional*): The delay (in seconds) before the user is taken to the success page. If this value is greater than 0, the user will first see the standard Adobe Sign success message, and then after a delay will be redirected to your success page,
redirectUrl (string, *optional*): A publicly accessible url to which the user will be sent after successfully completing the signing process

SecurityOption {

openPassword (string, *optional*): The secondary password that will be used to secure the PDF document. Note that AdobeSign will never show this password to anyone, so you will need to separately communicate it to any relevant parties

VaultingInfo {

enabled (boolean, *optional*): For accounts set up for document vaulting and the option to enable per agreement, this determines whether the document is to be vaulted



1.2.8 Send Reminder

This activity creates a reminder on the specified participants of an agreement identified by agreement id in the path

Input:

- Agreement Id* (String): This parameter will contain the agreement id
- Participant Ids* (String []): This parameter will contain the participant id in comma separated format
- Frequency* (String): The frequency at which reminder will be sent until the agreement is completed. This is an option control with DAILY_UNTIL_SIGNED, WEEKDAILY_UNTIL_SIGNED, ONCE, NEVER.

Output:

- Reminder Creation Result (String): The parameter will contain response in JSON format

1.2.9 Share Agreement

This activity is used to share the agreement with the other users. Following are the parameters for this activity

Input:

- Agreement Id* (String): This parameter will contain the agreement id
- email* (String []): This parameter will contain email accounts where the reminder needs to be sent
- message (String): This parameter will contain the message. This field is optional

Output:

- Share Creation Response List (String): The parameter will contain response in JSON format

```
{
  "shareCreationResponseList": [
    {
      "email": "",
      "participantId": ""
    }
  ]
}
```

ShareCreationResponseList {

shareCreationResponseList ([ShareCreationResponse\[\]](#)): List of ShareCreationResponse

ShareCreationResponse {

email (string): The email address that was requested,

participantId (string): The unique identifier of the participant

1.2.10 Get Signing URL

This activity Retrieves the URL for the e-sign page for the current signer(s) of an agreement. Following are the parameters for this activity.

Input:

- Agreement Id* (String): The agreement identifier, as returned by the agreement creation API or retrieved from the API to fetch agreements

Output:

- Signing URL Response (String): The parameter will contain response in JSON format

```
{
  "signingUrlSetInfos": [
    {
      "signingUrls": [
        {
          "email": "",
          "esignUrl": ""
        }
      ],
      "signingUrlSetName": ""
    }
  ]
}
```

[-] **SigningUrlResponse {**

signingUrlSetInfos ([SigningUrlSetInfo\[\]](#)): An array of urls for signer sets involved in this agreement.

}

[-] **SigningUrlSetInfo {**

signingUrls ([SigningUrl\[\]](#)): An array of urls for current signer set,

signingUrlSetName (string, *optional*): The name of the current signer set. Returned only, if the API caller is the sender of agreement

}

[-] **SigningUrl {**

email (string): The email address of the signer associated with this signing url,

esignUrl (string): The email address of the signer associated with this signing url

}

Get Signing URL

Agreement Id

AgreementId

1.2.11 Download Agreement

This activity is used to download the agreement based on agreement id. Following are the parameters for this activity.

Input:

- Agreement Id* (String): This parameter will contain the agreement id of the agreement
- File Name* (String): File Name of the downloaded file
- File Path* (String): Location where the file will be downloaded

Output:

- Downloaded (Boolean): Response in boolean

Download Agreement

Agreement Id

AgreementId

File Name

"File to be downloaded Name"

File Path

"File path"

1.2.12 Update Agreement Visibility

This activity is used to set the visibility of the agreement. Following are the parameters for this activity.

Input:

- Agreement Id* (String): This parameter will contain the user id
- Visibility State* (String): This is a dropdown where you can set the visibility of the agreement

Output:

- Response (Boolean): Response in boolean

Update Agreement Visibility

Agreement Id

AgreementId

1.2.13 Update Agreement State

This activity is used to set the state of the agreement. Following are the parameters for this activity.

Input:

- Agreement Id* (String): This parameter will contain the user id
- State* (String): This is a dropdown where you can set the status of the agreement
- Notify Others (Boolean): set to true if you want to notify other users on the updated status

Output:

- Response (Boolean): Response in boolean

Update Agreement State

Agreement Id

AgreementId

Notify Others

True

1.2.14 Delete Agreement

This activity is used to delete the agreement from the database. Following are the parameters for this activity.

Input:

- Agreement Id* (String): This parameter will contain the agreement id

Output:

- Response (Boolean): Response in boolean

Delete Agreement

Agreement Id

AgreementId

1.2.15 Get All Users

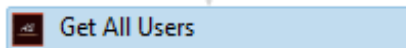
This activity retrieves all the users in an account in an account in JSON format. Following are the parameters for this activity.

Output:

- Users Information (String): The parameter will contain response in JSON format

```
{
  "page": {
    "nextCursor": ""
  },
  "userInfoList": [
    {
      "email": "",
      "id": "",
      "isAccountAdmin": false,
      "accountId": "",
      "company": "",
      "firstName": "",
      "lastName": ""
    }
  ]
}
```

Key	Description
email (string)	The email address of the user
id (string)	A unique identifier of the user
isAccountAdmin (boolean)	True if user is account admin
accountId (string, optional)	The account id of the user
Company (string, optional)	The name of company of the user
firstName (string, optional)	The first name of the user,
lastName (string, optional)	The last name of the user



1.2.16 Get User

This activity Retrieves detailed information about the user in a the caller account in JSON format.

Following are the parameters for this activity.

Input:

- User Id* (String): This parameter will contain the user id

Output:

- Detailed User Information (String): The parameter will contain response in JSON format

```
{
  "accountType": "",
  "email": "",
  "id": "",
  "isAccountAdmin": false,
  "status": "",
  "accountId": "",
  "company": "",
  "firstName": "",
  "initials": "",
  "lastName": "",
  "locale": "",
  "phone": "",
  "title": ""
}
```

Key	Description
accountType (string)	Type of account to which the user belongs (null if no account)
email (string)	The email address of the user
id (string)	A unique identifier of the user
isAccountAdmin (boolean)	True if the user is account admin
status (string)	Status of the user
accountId (string, optional)	The account id of the user
Company (string, optional)	The name of company of the user
firstName (string, optional)	The first name of the user
initials (string, optional)	The initials of the user
lastName (string, optional)	The last name of the user
locale (string, optional)	The UI locale of the user
phone (string, optional)	The phone number of the user
title (string, optional)	The job title of the user

1.2.17 Get Groups of Users

This activity is used to retrieve the information for the group for a user in JSON format. Following are the parameters for this activity.

Input:

- User Id* (String): This parameter will contain the user id

Output:

- User Groups Information (String): Output in following JSON format

```
{
  "groupInfoList": [
    {
      "id": "",
      "isGroupAdmin": false,
      "isPrimaryGroup": false,
      "status": "",
      "name": "",
      "settings": {
        "libraryDocumentCreationVisible": {
          "value": false,
          "inherited": false
        },
        "userCanSend": {
          "value": false,
          "inherited": false
        }
      }
    }
  ]
}
```

Key	Description
id (string)	Unique identifier of the group
isGroupAdmin (boolean)	True if user is group admin
isPrimaryGroup (boolean)	True if the group is the user's primary group

status (string)	Status of the group membership
name (string, optional)	Name of the group.
settings (SettingsInfo, optional)	List of some group membership settings.

[-] **SettingsInfo {**

libraryDocumentCreationVisible ([BooleanSettingInfo](#), *optional*): Can user create library documents?,

userCanSend ([BooleanSettingInfo](#), *optional*): Can user send agreements for signature to others?

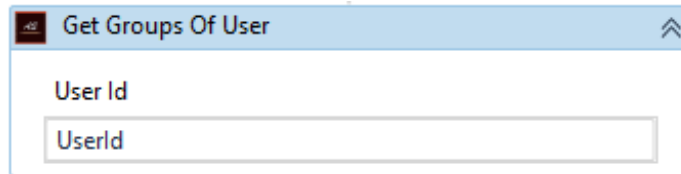
}

[-] **BooleanSettingInfo {**

value (boolean): Value of the setting,

inherited (boolean, *optional*): True if setting is inherited from its group or account value.

}



1.2.18 Update User Details

This activity is used to update the user details based on user id. Following are the parameters for this activity.

Input:

- User Id* (String): This parameter will contain the user id
- Phone (String): This parameter will contain the phone number. This field is optional
- Title (String): This parameter will contain the title of the user. This field is optional
- IsAccountAdmin* (Boolean): Set it to true if you want to make user as admin

Output:

- Response (Boolean): Response in boolean

The screenshot shows a configuration window titled "Update User Details". It contains two input fields. The first field is labeled "User Id" and contains the text "UserId". The second field is labeled "Is Account Admin" and contains the text "True".

1.2.19 Update User State

This activity is used to activate/deactivate the user from account. Following are the parameters for this activity.

Input:

- User Id* (String): This parameter will contain the user id
- State* (String): This is a dropdown where you can set the status of the user

Output:

- Response (Boolean): Response in boolean

The screenshot shows a configuration window titled "Update User State". It contains one input field labeled "User Id" with the value "UserId".

1.2.20 Get All Groups

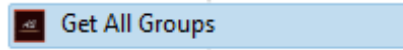
This activity is used to retrieve all groups related to account in JSON format. Following are the parameters for this activity.

Output:

- Groups Information (String): Retrieves all groups in the following JSON format

```
{
  "groupInfoList": [
    {
      "groupId": "",
      "groupName": ""
    }
  ],
  "page": {
    "nextCursor": ""
  }
}
```

Key	Description
groupId (string)	Unique identifier of the group
groupName (string)	Name of the group



1.2.21 Get Group

This activity is used to retrieve the group information based on group id JSON format. Following are the parameters for this activity.

Input:

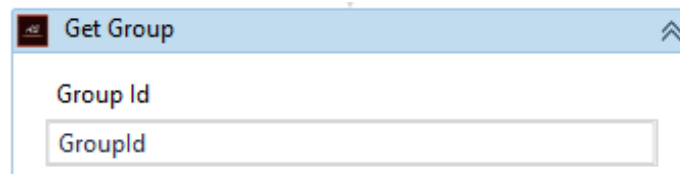
- Group Id* (String): This parameter will contain the group id

Output:

- Detailed Group Information (String): Output in following JSON format

```
{
  "created": "",
  "name": "",
  "id": ""
}
```

Key	Description
created (string)	Date of creation of the group. Format would be yyyy-MM-dd'T'HH:mm:ssZ. For example, e.g 2016-02-25T18:46:19Z represents UTC time,
name (string)	Name of the group
id (string, optional)	he unique identifier of the Group



1.2.22 Get Users of Group

This activity is used to retrieve all users in a group JSON format. Following are the parameters for this

activity.

Input:

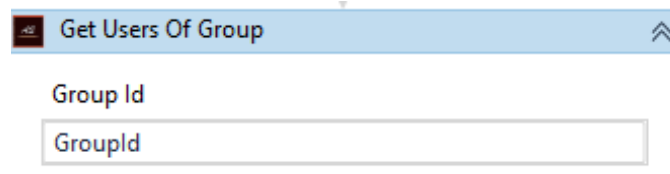
- Group Id* (String): This parameter will contain the group id

Output:

- Group User Information (String): Output in following JSON format

```
{
  "page": {
    "nextCursor": ""
  },
  "userInfoList": [
    {
      "email": "",
      "id": "",
      "isGroupAdmin": false,
      "company": "",
      "firstName": "",
      "lastName": ""
    }
  ]
}
```

Key	Description
email (string)	The email address of the user
id (string)	A unique identifier of the user
isGroupAdmin (boolean)	True if user is group admin
company (string, optional)	The name of company of the user,
firstName (string, optional)	The first name of the user
lastName (string, optional)	The last name of the user



1.2.23 Get Message Template

This activity is used to retrieve a list of message templates applicable to the current user based on locale in JSON format. Following are the parameters for this activity.

Input:

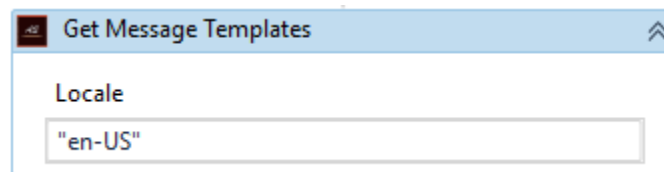
- **Locale** (String): This parameter will contain the locale of the user

Output:

- **MessageTemplates** (String): Output in following JSON format

```
{
  "messageTemplateList": [
    {
      "locale": "",
      "messageTemplateId": "",
      "name": "",
      "text": ""
    }
  ],
  "page": {
    "nextCursor": ""
  }
}
```

Key	Description
locale (string)	Message template locale
messageTemplateId (string)	Unique identifier of the message template
name (string)	Message template name
text (string)	Message template text



1.2.24 Get Account Information

This activity is used to retrieve the information for an account in JSON format. Following are the parameters for this activity.

Input:

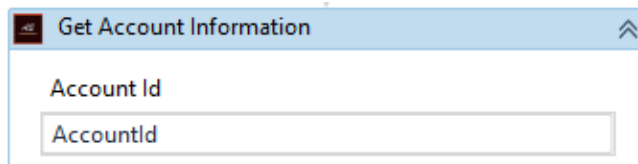
- Account Id* (String): This parameter will contain the group id

Output:

- Account (String): Output in following JSON format

```
{  
  "accountType": "",  
  "company": "",  
  "created": "date",  
  "id": "",  
  "modified": "date",  
  "name": ""  
}
```

Key	Description
accountType (string, optional)	The provisioned type of the account
company (string, optional)	The company associated with the account.
created (date, optional)	Date when the account was created
id (string, optional)	Identifier of the account
modified (date, optional)	The last date that the account information was updated
name (string, optional)	The name which the account is known



1.2.25 Get All Workflows

This activity is used to retrieve workflows for a user in JSON format. Following are the parameters for this activity.

Output:

- Workflows (String): Output in following JSON format

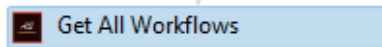
```
{  
  "userWorkflowList": [  
    {  
      "created": "",  
      "description": "",
```

```

    "displayName": "",
    "id": "",
    "name": "",
    "scope": "",
    "status": "",
    "scopeId": ""
  }
]
}

```

Key	Description
created (string)	The date on which the workflow was created
description (string)	Description provided for this workflow at the time of its creation
displayName (string)	The display name of the workflow
id (string)	The unique identifier of a workflow
name (string)	The name of the workflow.
scope (string)	The workflow scope (ACCOUNT or GROUP or OTHER)
status (string)	The workflow status (ACTIVE or DRAFT or INACTIVE or OTHER)
scopeId (string, optional)	Identifier of scope



1.2.26 Get Workflow

This activity is used to download the agreement based on agreement id. Following are the parameters for this activity.

Input:

- Workflow Id* (String): This parameter will contain the workflow id of the

Output:

- Workflow Description (String): Output in following JSON format

```

{
  "agreementNameInfo": {
    "defaultValue": "",
    "editable": false,
    "required": false,
    "visible": false
  },
  "authoringInfo": {
    "defaultValue": "",
    "editable": false,
    "required": false,

```

```

    "visible": false
  },
  "ccsListInfo": [
    {
      "defaultValues": [
        ""
      ],
      "editable": false,
      "label": "",
      "maxListCount": 1,
      "minListCount": 1,
      "name": "",
      "required": false,
      "visible": false
    }
  ],
  "created": "date",
  "description": "",
  "displayName": "",
  "expirationInfo": {
    "defaultValue": "",
    "editable": false,
    "maxDays": 1,
    "required": false,
    "visible": false
  },
  "fileInfos": [
    {
      "label": "",
      "name": "",
      "required": false,
      "workflowLibraryDocumentSelectorList": [
        {
          "label": "",
          "modifiedDate": "",
          "sharingMode": "",
          "templateTypes": [
            ""
          ],
          "workflowLibDoc": ""
        }
      ]
    }
  ],
  "localeInfo": {

```



```
"availableLocales": [
  ""
],
"defaultValue": "",
"editable": false,
"required": false,
"visible": false
},
"mergeFieldsInfo": [
  {
    "defaultValue": "",
    "displayName": "",
    "editable": false,
    "fieldName": "",
    "required": false,
    "visible": false
  }
],
"messageInfo": {
  "defaultValue": "",
  "editable": false,
  "required": false,
  "visible": false
},
"modified": "date",
"name": "",
"passwordInfo": {
  "defaultValue": "",
  "editable": false,
  "label": "",
  "name": "",
  "required": false,
  "visible": false
},
"recipientsListInfo": [
  {
    "allowSender": false,
    "allowfax": false,
    "authenticationMethod": "",
    "defaultValue": "",
    "editable": false,
    "label": "",
    "maxListCount": "int",
    "minListCount": "int",
    "name": "",
```

```

        "visible": false
    }
],
"scope": "",
"status": "",
"scopeId": ""
}

```

Key	Description
agreementNameInfo (WorkflowDefaultParams)	Information about name field in DocumentCreationInfo input field in the agreement creation request
authoringInfo (WorkflowDefaultParams)	Information about authoringRequested field in SendDocumentInteractiveOptions input field in the agreement creation request
ccsListInfo (CCsListInfoDescription[])	Information about CCList input field in the agreement creation request
created (date)	The day on which the workflow was created
description (string)	Description provided for this workflow at the time of its creation
displayName (string)	The display name of the workflow
expirationInfo (ExpirationFieldInfoDescription)	Information about daysUntilSigningDeadline field in DocumentCreationInfo input field in the agreement creation request
fileInfos (FileInfosDescription[])	Information about FileInfo input field in the agreement creation request
localeInfo (LocaleFieldInfoDescription)	Information about locale field in DocumentCreationInfo input field in the agreement creation request
mergeFieldsInfo (MergeFieldInfoDescription[])	Information about customFieldInfos in DocumentCreationInfo input field in the agreement creation request
messageInfo (WorkflowDefaultParams)	Information about message field in DocumentCreationInfo input field in the agreement creation request
modified (date)	The day on which the workflow was last modified
name (string)	The name of the workflow
passwordInfo (PasswordFieldInfoDescription)	Information about openPassword field in SecurityOptions input field in the agreement creation request
recipientsListInfo (RecipientsListInfoDescription[])	Information about RecipientsInfo input field in the agreement creation request
scope (string)	The workflow scope (ACCOUNT or GROUP or OTHER)
status (string)	The workflow status (ACTIVE or DRAFT or INACTIVE or OTHER)
scopeId (string, optional)	Identifier of scope

WorkflowDefaultParams {

defaultValue (string): default value of the field if input for this field is not provided and this field is required,
editable (boolean): Whether current field can be edited. If editable attribute for this field is false then this field should not be provided in the agreement creation request and default value of this field will be used in agreement creation,
required (boolean): Whether this field is required or optional,
visible (boolean): Whether current field is visible or not. If visible attribute for this field is false then this field should not be visible in the agreement creation UI using this workflow to user

CCsListInfoDescription {

defaultValues (string[]): An array of default emails that will be used if no input is provided for this list element,
editable (boolean): Whether current field can be edited. If editable attribute for this field is false then this field should not be provided in the agreement creation request and default value of this field will be used in agreement creation,
label (string): A display text for the workflow user that can be used for the current CC list,
maxListCount (integer): maximum number of entries allowed in the current CC list,
minListCount (integer): minimum number of entries allowed in the current CC list,
name (string): Name of the current CC list,
required (boolean): Whether this field is required or optional,
visible (boolean): Whether current field is visible or not. If visible attribute for this field is false then this field should not be shown in the agreement creation page using this workflow

ExpirationFieldInfoDescription {

defaultValue (string): default value of the field if input for this field is not provided and this field is required,
editable (boolean): Whether current field can be edited. If editable attribute for this field is false then this field should not be provided in the agreement creation request and default value of this field will be used in agreement creation,
maxDays (integer): Maximum number of days for agreement expiration,
required (boolean): Whether this field is required or optional,
visible (boolean): Whether current field is visible or not. If visible attribute for this field is false then this field should not be visible in the agreement creation UI using this workflow to user

FileInfosDescription {

label (string): Display label of this field for the external users,
name (string): Name of the fileInfo element,
required (boolean): Whether this field is required or optional,
workflowLibraryDocumentSelectorList ([WorkflowLibraryDocument\[\]](#)): A list of workflow library documents out of which one workflow library document can be selected with this fileInfo object

WorkflowLibraryDocument {

label (string): A display text for this form for workflow users,

modifiedDate (string): The date on which the library document was last modified. Format would be yyyy-MM-dd'T'HH:mm:ssZ.

For example, e.g 2016-02-25T18:46:19Z represents UTC time,

sharingMode (string) = ['USER' or 'GROUP' or 'ACCOUNT' or 'GLOBAL']: Specifies who should have access to this library document. GLOBAL sharing is a restricted operation,

templateTypes (string[]) = ['DOCUMENT' or 'FORM_FIELD_LAYER']: A list of one or more library template types,

workflowLibDoc (string): An id of the workflow library document that can be provided as an input file in the custom workflow agreement creation request

LocaleFieldInfoDescription {

availableLocales (string[]): Which locales can be chosen for this workflow,

defaultValue (string): default value of the field if input for this field is not provided and this field is required,

editable (boolean): Whether current field can be edited. If editable attribute for this field is false then this field should not be provided in the agreement creation request and default value of this field will be used in agreement creation,

required (boolean): Whether this field is required or optional,

visible (boolean): Whether current field is visible or not. If visible attribute for this field is false then this field should not be visible in the agreement creation UI using this workflow to user

MergeFieldInfoDescription {

defaultValue (string): default value of the field if input for this field is not provided and this field is required,

displayName (string): The display text that can be shown for this custom field,

editable (boolean): Whether current field can be edited. If editable attribute for this field is false then this field should not be provided in the agreement creation request and default value of this field will be used in agreement creation,

fieldName (string): Name of the custom field in this workflow,

required (boolean): Whether this field is required or optional,

visible (boolean): Whether current field should be visible on agreement creation page. If visible attribute for this field is false then this field should not be shown on the agreement creation page using this workflow

PasswordFieldInfoDescription {

defaultValue (string): Default value of the password info field,

editable (boolean): Default value of the password info field,

label (string): Label of password field,

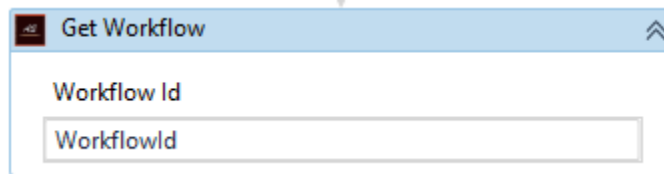
name (string): Name of password field,

required (boolean): Whether this field is required or optional,

visible (boolean): Whether password info field is visible on agreement creation page

RecipientsListInfoDescription {

- allowSender** (boolean): whether sender is allowed as a recipient,
- allowfax** (boolean): whether fax is allowed or not,
- authenticationMethod** (string) = ['NONE' or 'KBA' or 'PASSWORD' or 'WEB_IDENTITY']: authentication method for the current recipient list to have access to view and sign the document,
- defaultValue** (string): A default email or fax number,
- editable** (boolean): Whether current field can be edited. If editable attribute for this field is false then this field should not be provided in the agreement creation request and default value of this field will be used in agreement creation,
- label** (string): A display text for the workflow user that can be used for the current recipients list,
- maxListCount** (int): maximum number of entries allowed in the current recipient list,
- minListCount** (int): minimum number of entries allowed in the current recipient list,
- name** (string): Name of the current RecipientInfo list,
- visible** (boolean): Whether current field is visible. If visible attribute for this field is false then this field should not be shown in the agreement creation UI using this workflow to user



1.2.27 Get All Library Documents

This activity is used to retrieve library document for a user in JSON format. Following are the parameters for this activity.

Output:

- Library Documents (String): output in JSON format

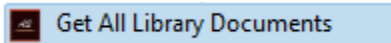
```
{
  "libraryDocumentList": [
    {
      "hidden": false,
      "id": "",
      "modifiedDate": "",
      "name": "",
      "sharingMode": "",
      "templateTypes": [
        ""
      ],
      "creatorEmail": "",
      "groupId": "",
      "status": ""
    }
  ],
  "page": {
```

```

    "nextCursor": ""
  }
}

```

Key	Description
hidden (boolean)	True if Library Document is hidden
id (string)	The unique identifier that is used to refer to the library template
modifiedDate (string)	The date on which the library document was last modified
name (string)	The name of the library document
sharingMode (string)	Specifies who should have access to this library document. GLOBAL sharing is a restricted operation
templateTypes (string [])	A list of one or more library template types
creatorEmail (string, optional)	Email address of the library document creator.
groupId (string, optional)	The unique identifier of the group to which the library template belongs to.
status (string, optional)	Status of the library document



1.2.28 Get Library Document

This activity is used to retrieve the details of a library document based on document id. Following are the parameters for this activity.

Input:

- Library Document Id* (String): This parameter will contain the library document id of the

Output:

- Library Document Information (String): Output in following JSON format

```

{
  "fileInfos": [
    {
      "transientDocumentId": "",
      "urlFileInfo": {
        "mimeType": "",
        "name": "",
        "url": ""
      }
    }
  ],
  "name": "",
  "sharingMode": "",

```

```

    "state": "",
    "templateTypes": [
      ""
    ],
    "createdDate": "",
    "creatorEmail": "",
    "creatorName": "",
    "groupId": "",
    "id": "",
    "isDocumentRetentionApplied": false,
    "modifiedDate": "",
    "status": ""
  }
}

```

Key	Description
fileInfos (FileInfo[])	A list of one or more files (or references to files) that will be used to create the template.
name (string)	The name of the library template that will be used to identify it, in emails and on the website
sharingMode (string)	Specifies who should have access to this library document
state (string)	State of the library document
templateTypes (string [])	A list of one or more library template types
createdDate (string, optional)	Date when library document was created
creatorEmail (string, optional)	Email address of the library document creator
creatorName (string, optional)	Name of the library document creator
groupId (string, optional)	The unique identifier of the group to which the library template belongs to. If not provided during creation, primary group of the creator will be used
id (string, optional)	The unique identifier that is used to refer to the library template
isDocumentRetentionApplied (boolean, optional)	Is document retention applied for this resource
modifiedDate (string, optional)	Date when library document was last modified
status (string, optional)	Status of the library document

[-] **FileInfo {**

transientDocumentId (string, *optional*): ID for a transient document that will be added to the library document,
urlFileInfo ([URLFileInfo](#), *optional*): URL for an external document to add to the library document

}

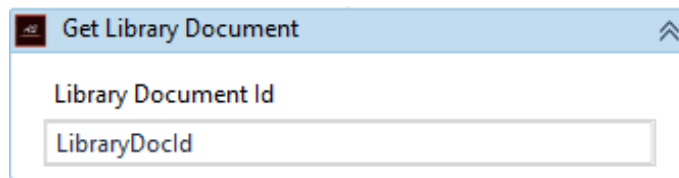
[-] **URLFileInfo {**

contentType (string): The mime type of the referenced file, used to determine if the file can be accepted and the necessary conversion steps can be performed,

name (string): The original system file name of the document being sent,

url (string): A publicly accessible URL for retrieving the raw file content

}



1.2.29 Update Document Visibility

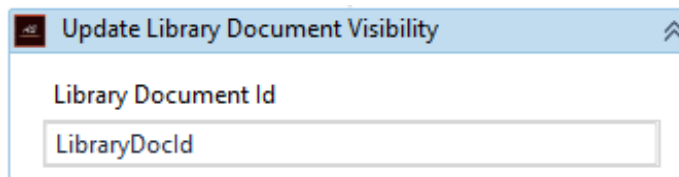
This activity is used to update the library document's state. Following are the parameters for this activity.

Input:

- Library Document Id* (String): This parameter will contain the library document id
- Visibility Status* (String): This is a dropdown where you can set the visibility of the user

Output:

- Response (Boolean): Response in Boolean



1.2.30 Update Document State

This activity is used to update the library document's state. Following are the parameters for this activity.

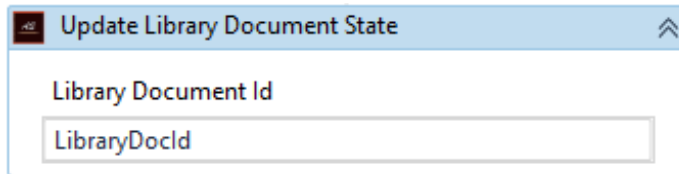
Input:

- Library Document Id* (String): This parameter will contain the library document id

- State* (String): This is a dropdown where you can set the status of the user

Output:

- Response (Boolean): Response in Boolean



The image shows a screenshot of a REST client window titled "Update Library Document State". The window has a light blue header bar with a small icon on the left and an upward-pointing arrow on the right. Below the header, the text "Library Document Id" is displayed. Underneath, there is a text input field containing the value "LibraryDocId".