



Conga Sign

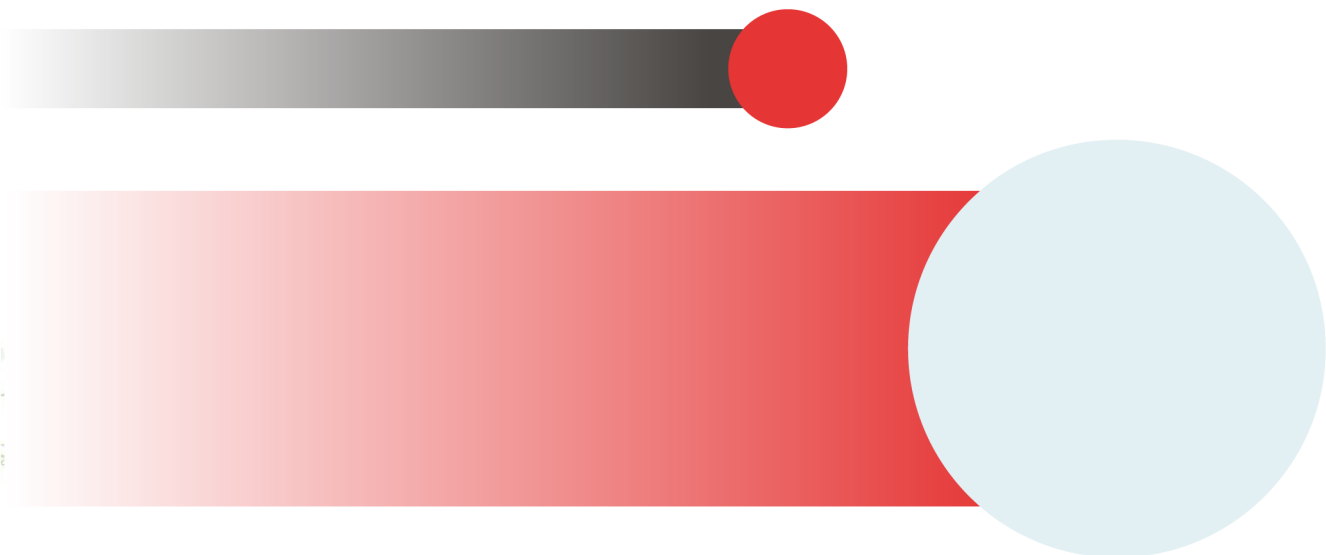


Table of Contents

Conga Sign Release Notes	11
September 22.11.09 Release Notes	11
Packages.....	11
System Requirements and Supported Platforms.....	12
New Features.....	12
Enhancements.....	12
Fixed Issues.....	12
Known Issues	13
September '22 Release Notes	13
Packages.....	13
System Requirements and Supported Platforms.....	14
New Features.....	14
Enhancements.....	14
Fixed Issues.....	14
Known Issues	14
About Conga Sign.....	15
Key Terminology	16
What's New in Conga Sign Documentation.....	18
Conga Sign for Administrators.....	23
Conga Sign Technical Requirements	24
Salesforce® Edition:.....	24
Salesforce® Products:	24
Internet Browsers (latest version):	24
Best Practice Guidelines	25
Template Software	25
Mobile.....	26
SMS Authentication Feature	26
Installing Conga Sign Package.....	26
To install Conga Sign.....	26

Post-Installation Configuration	27
Configuring Conga Sign.....	28
Permissions Needed to Use Conga Sign.....	29
Settings for Conga Sign Connected App	31
Conga Sign Email Change.....	32
What's in the Conga Sign package?.....	32
Adding Sign Button and Transactions Related List to Page Layout	33
Adding Conga Sign Button to a Page Layout Manually.....	34
Configuring an Object for Conga Sign Manually.....	34
Creating a Conga Sign button Manually.....	36
Business Units.....	37
Setting Final Document Delivery Options.....	44
Setting Guided Signing.....	45
Adding Reassign Signer Button.....	46
Remove Reassign Signer	47
Removing Conga Logo from Signature and Initial.....	48
Configuring Require Drawn Signature Setting.....	48
License Provisioning Tips	49
My sandbox license has expired, how do I extend it?	49
Why does my Sandbox have an expiration countdown, but my production org doesn't?	49
I don't see "manage licenses" in my sandbox as I do in production, why?.....	49
How do I assign users licenses?	49
I just renewed, but I am still seeing an expiration warning.	50
Does the Conga use License Keys?	50
Assigning User Licenses for Conga Sign	50
Customizing Your Logo.....	51
To add a logo	51
Customizing Emails	51
To customize emails.....	52
Conga Sign Parameters	53
Parameters.....	53
Conga Sign Redirect URL.....	65
Defining a Conga Sign Redirect URL	66

Automatically Upload a Conga Sign Document to a Third Party Repository.....	67
To automatically upload a completed Conga Sign document to an online repository	67
Using Conga Sign in Salesforce Communities.....	68
To create and add a Conga Sign Button to a Salesforce Community	69
To create and add a Conga Sign-integrated Composer Button to a Salesforce Community.....	69
To create and add a Conga Sign-integrated Trigger solution to a Salesforce Community	69
Using Conga Sign with Conga CLM	70
CLM-Conga Sign Custom Settings.....	70
Using Conga Sign with Conga Composer	71
Using Conga Sign with Conga Contracts for Salesforce	72
Setting up a Process Builder for Conga Sign Transaction	72
To set up a Process Builder for Conga Sign Transaction	72
Conga Sign for Users	74
Conga Sign Product Guide	75
Learn Conga Sign.....	75
Install Conga Sign.....	75
Configure Conga Sign	75
Use Conga Sign.....	75
Use Signature Tags with PDF Forms.....	75
Conga Sign Product Limits.....	75
Available Features for Conga Sign Integrations.....	76
List of Conga Sign Signature Tags	78
List of recognized Adobe Sign formats for signature tags.....	80
List of recognized DocuSign formats for signature tags.....	81
Using Signature and Custom Tags.....	82
Custom tags.....	82
Use Conga Sign Signature Tags with PDF Files	83
Formatting Conga Sign Date of Recipient Tag.....	84
Write Back tags.....	85
Attachment Tags.....	90
Resizing Conga Sign Tags.....	94
Setting the Text Size of Conga Sign Tags.....	95
Setting the Width of a Conga Sign Tag	99

Conditionally Display a Conga Sign Tag in Word	101
Text Wrapping Conga Sign Tags.....	102
Creating a Document to Sign.....	106
To upload the document and add tags.....	106
Creating New Recipients in Create Transaction UI.....	107
To add a New Recipient in the Create Transaction User Interface	107
Sending Documents for eSignature.....	108
To send documents for eSignature	108
Sending Transactions with Multiple Documents.....	110
Sending Transactions using Send On Behalf Of	111
Sending Transactions from Conga CLM.....	113
Signing a Document	115
To sign a document	116
Using Signature Image Upload.....	117
In-Person Signing.....	119
In-Person Signing Functionality.....	119
In-Person Signing Roles.....	119
In-Person Signer	120
Facilitator	120
Signer (optional).....	120
Canceling a Conga Sign Transaction	123
To cancel a Conga Sign Transaction as a Salesforce User.....	123
To cancel a Conga Sign Transaction as a Signer.....	124
Frequently Asked Questions.....	125
Cancel Conga Sign Transactions in Bulk	125
Conga Sign Link Expiration	133
Conga Sign Recipient Object.....	134
Recipient Fields.....	134
Recipient Status	135
Reassigning Signers as a Recipient.....	136
To reassign a designated signer's Conga Sign tags to a different signer.....	136
Reassigning Signers from Salesforce.....	137
To reassign a signer from Salesforce.....	137

Audit Trail.....	139
Access the Audit Trail for a Transaction.....	139
SMS Authentication	140
SMS Authentication Setup.....	140
Enable SMS Authentication for Recipients	141
Completing SMS Authentication as a Recipient.....	142
Tracking SMS Authentication in the Audit Trail.....	143
Reassigning Signers with SMS Authentication.....	143
Frequently Asked Questions.....	143
Supported Languages in Conga Sign.....	144
Selecting a Language for a Recipient.....	145
Selecting a Language as a Recipient	145
Selecting a Language for the Audit Trail.....	145
Troubleshooting Topics.....	146
Browser tabbed navigation issues.....	146
Transactions cannot be linked to Opportunity Product.....	147
Why aren't My Documents displayed on the Send with Conga Sign Screen?.....	147
Write Back tags FAQs and Tips.....	148
Conga Sign for REST API Developers	150
Minimum Software Requirements.....	151
Conga Sign REST APIs	152
Obtain the JWT access token	152
Example Response	153
Authentication Tokens.....	154
User Authentication Tokens.....	155
Sender Authentication Tokens	155
Signer Authentication Tokens.....	156
Using the Signer Experience	157
Signing Documents as a Transaction Owner	158
Signing Documents as a Signer	158
Uploading Attachments.....	159
Accessible Signing.....	160
Changing Signers	160

Declining to Sign	161
Completing Signing.....	161
Downloading Signed Documents	161
Reviewing Signed Documents	162
Storing Signing Session Data	163
Supported Languages	164
Configuring Languages	165
Creating and Sending a Transaction	167
How to Create a Package	172
Checking Transaction Status and Downloading Documents.....	175
Changing a Recipient.....	178
Adding Signatures	180
Managing Signers' Attachments	183
Delivering Signed Documents.....	189
Creating a Signer Workflow	191
Downloading Signed Documents and Audit Trails.....	193
Adding Fields.....	197
Configuring Optional Signatures.....	200
Adding, Updating, Removing Signers	202
Adding Signers to a Transaction.....	202
Updating an Existing Signer	203
Removing Signers From a Transaction.....	204
Document Extraction.....	205
Configuring PDF Form Fields.....	206
Configuring Document Extraction	206
Feature Overview	210
Uploading & Deleting Documents	212
Uploading Documents	212
Uploading Multiple Documents.....	214
Replacing an Existing Document.....	215
Updating Document Metadata	217
Deleting a Document.....	217
Customizing Invitation Emails	218

Customizing emails by transaction.....	218
Customizing emails by Signer	218
Injecting Field Values	221
Form Building Fields	224
Supported Fonts.....	228
Using PDF/A Documents	230
Managing Groups	230
Creating Groups	231
Retrieving Groups	234
Adding a Group signer	236
Authenticating Signers.....	238
Using General Authentication.....	239
Manually sending an SMS code.....	240
Using KBA.....	240
Enforcing Signature Capture	242
Enforcing Capture Signature	242
Custom Transaction Data	245
Custom Document Data	250
Mobile Signing Ceremony	253
Prerequisites.....	254
Signing Packages	254
Signer Options	255
Position Extraction	256
Adding Text Tags	256
Feature Overview	259
Text Tag Types.....	262
Text Tag Parameters	263
Signer with Multiple Signatures	265
Signer Experience Settings	267
Document Package Settings	267
Customizing the Signer Experience	271
Text Anchors	272
Adding Text Anchors	272

Text Anchor Parameters 272

Best Practices for Using Text Anchors 273

Conga Sign Features by Release278

 Features by Release..... 278

Conga Customer Community & Learning Center Resources279

 Ready to get started? 279

Quickly and easily send documents for eSignature using a friendly user interface. For more options, consider a seamless integration with the leading document generation application on the AppExchange, Conga Composer.


Conga Sign Release Notes

Review the latest features delivered in Conga Sign.

- [September22.11.09 Release Notes](#)
- [September '22 Release Notes](#)

September22.11.09 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, fixed issues, and known issues for the Conga Sign September22.11.09 release. For documentation updates, see [What's New in Conga Sign Documentation](#).

 This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

All Conga customers have FREE access to getting started content and release training in the Conga Learning Center. To take your training further, ensure your organization has access to the Conga Learning Pass, which is a training subscription service. [Click here](#) to learn more.

Packages

The following packages and dependencies are required to upgrade to this release and use its new features. These are the *minimum* required versions; later versions are also supported. Prerequisites for each feature can be found in its documentation. Packages marked as **(New)** are new.

Product	Latest Certificate Version
Conga Sign (New)	1.82
Conga Sign API	1.0.0

System Requirements and Supported Platforms

For requirements and recommendations you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see [System Requirements and Supported Platforms Matrix](#).

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

There are no enhancements new to this release.

Fixed Issues

The following table lists the issues fixed in this release. If any actions are required, they will be listed in this table.


Case Number	Conga Internal ID	Description
N/A	DCS-5015	When the Enable JsonAccess Annotation Validation for the Visualforce JavaScript Remoting API critical update is enabled within Salesforce, an error occurs when creating a transaction.
N/A	DCS-5022	When the Sign & Composer solution set to background mode (DS7=1142) and the routing type is set to SERIAL, the signing order is not honored and the documents are sent out in a random order.

Known Issues

There are no known issues in this release.

September '22 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, fixed issues, and known issues for the Conga Sign September '22 release. For documentation updates, see [What's New in Conga Sign Documentation](#).

 This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

All Conga customers have FREE access to getting started content and release training in the Conga Learning Center. To take your training further, ensure your organization has access to the Conga Learning Pass, which is a training subscription service. [Click here](#) to learn more.

Packages

The following packages and dependencies are required to upgrade to this release and use its new features. These are the *minimum* required versions; later versions are also supported. Prerequisites for each feature can be found in its documentation. Packages marked as **(New)** are new.

Product	Latest Certificate Version
Conga Sign	1.81.1
Conga Sign API (New)	1.0.0

System Requirements and Supported Platforms

For requirements and recommendations you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see [System Requirements and Supported Platforms Matrix](#).

New Features

The following feature is new to this release.

Conga Sign REST API

Conga Sign now offers API for developers to customize and fully utilize Conga Sign integrations and features. See [Conga Sign for Rest API Developers](#).

Enhancements

There are no enhancements new to this release.

Fixed Issues

The following table lists the issue fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
N/A	DCS-4808	Creating Transactions on Mobile devices causes the Expiration and Reminder text and entry fields to overlap each other.

Known Issues

There are no known issues in this release.

About Conga Sign

Conga Sign is an eSignature application made specifically for Salesforce customers. Users can initiate an eSignature process from any object in Salesforce using pre-defined templates or by adding signature tags in real-time. Then they select up to 10 recipients, set the signing order, and deliver the document to be signed. All signing activity is tracked and updated in Salesforce, the result being a legally valid electronic signature.

To use Conga Sign, administrators must have a Salesforce license to configure Conga Sign and send documents for eSignature. However, recipients do not need a Salesforce license to sign documents sent through Sign. Conga Sign supports Word and PDF file types and allows users to store files in Salesforce. There is also a streamlined integration with Conga Composer, allowing Composer customers to leverage all of the features of Conga Composer with their Conga Sign eSignature processes.

A typical signing process with Conga Sign:

1. Click Send with Conga Sign on the record you're working on.
2. Select or upload a document to sign.
3. Select recipients and signing order.
4. Add tags to the document (if the template does not already have signature tags).
5. Send for Signature or facilitate in-person signing on your device.
6. Recipient(s) sign.
7. Final document is rendered and uploaded back into Salesforce and emailed to all parties.

Conga Sign allows you to perform the following tasks:

- Install and Upgrade Conga Sign from the Salesforce AppExchange.
- Set up and configure Conga Sign on Salesforce objects with Conga Sign Setup.
 - Use the Automatic Configuration in Conga Sign Setup to create the Send with Conga Sign button on specific objects.
 - Customize your Business Unit and Conga Sign emails.
 - Configure basic tags and write back tags.
- Use and integrate Conga Sign with Conga Composer, Conga Contracts for Salesforce, CLM, CPQ, Conga Contracts, and Conga Collaborate.
- Manually create custom Conga Sign buttons or links on specific objects in Salesforce Setup.
- Send Word and PDF files documents for eSignature from Salesforce through email.
 - Pre-tag documents with signature tags, or drag and drop signature tags in the Conga Sign user interface.
 - Create a Conga Sign Transaction when sending a document for eSignature.
- View the Audit Trail to see the history of a Conga Sign Transaction.

- Use SMS Authentication to require signers to enter a one-time passcode delivered through SMS, prior to viewing a Conga Sign document.

The following table lists the tasks that administrators and users can perform using Conga Sign.

Administrator	Administrator/User
<ul style="list-style-type: none"> • Conga Sign Technical Requirements • Installing Conga Sign Package • Post Installation Configuration • License Provisioning Tips • Customizing your Logo • Customizing Emails • Automatically Upload a Conga Sign Document to a Third Party Repository • Using Conga Sign in Salesforce Communities • Using Conga Sign with Conga Composer • Using Conga Sign with Conga Contracts for Salesforce • Setting up a Process Builder for Conga Sign Transaction 	<ul style="list-style-type: none"> • Using Signature and Custom Tags • Creating a Document to Sign • Creating New Recipients in Create Transaction UI • Sending a Document to Sign • Signing a Document • In Person Signing • Canceling a Conga Sign Transaction • Conga Sign Link Expiration • Conga Sign Recipient Object • Reassigning Signers as Recipient • Reassigning Signers from Salesforce • SMS Authentication

Key Terminology

It is important to understand how terms are used when working with Conga Sign.

Term	Definition
Email notification	The means by which signers are informed that they have an electronic document to sign. The email notification includes a link that takes the signer directly to the document.
Sender	Users who can create, send and manage Conga Sign documents.
Recipient	A Recipient is someone who receives a Conga Sign document. They can be designated as a Signer, In Person signer, or CCd individual.

Term	Definition
Signer	A Conga Sign document recipient is required to take action on the document.
Reminders and Expiration	Enables the sender to set the default validity and frequency of an eSignature request.
Tags	Locations in a document where the signer needs to take action and provide information.
In Person Signer	The In Person Signer is the recipient who signs a Conga Sign document in person.
Facilitator	The facilitator is the Salesforce user that creates the In Person Signing Transaction and is responsible for conducting the in person signing.
Transaction	A record created in Salesforce which tracks statuses, recipients, and documents being signed.

For more information about terms used with Conga products, see [Conga Product Glossary](#).


What's New in Conga Sign Documentation

This section lists changes in the documentation to support each release.

September '22

Document	Publication Date	Topic	Description
September '22	 19 Oct 2022	Permission Needed to Use Conga Sign	Updated Topic. Removed Apex Class APXT_CongaSign__apxt_servicedirect.
	 06 Apr 2022	Conga Sign for REST API Developers	New Topic.

May '22




Document	Publication Date	Topic	Description
May '22	 06 Apr 2022	Transaction Expirations and Reminders	New Topic.
		Recipient Email Notifications	Updated Topic.
		SMS Authentication Configuration Parameters	Updated Topic.
		Supported Languages in Conga Sign	Updated Topic.

December '21

Document	Publication Date	Topic	Description
December '21	📅 10 Nov 2021	Supported Languages in Conga Sign	Updated Topic.
		Audit Trail	Updated Topic.
		Sending Transactions using Send on Behalf Of	New Topic.
	📅 03 Nov 2021	Signing a Document	Updated Topic.
		List of Conga Sign Signature Tags	Updated Topic.
		Supported Languages in Conga Sign	Updated Topic.
		Write Back Tags	Updated Topic.
		Using Conga Sign with Conga CLM	New Topic.
		Conga Sign-CLM Custom Settings	New Topic.
		Sending Transactions from Conga CLM	New Topic.

Summer '21

Document	Publication Date	Topic	Description
Summer '21	📅 09 Sep 2021	Supported Languages in Conga Sign	Updated topic.
	📅 15 Jul 2021	Remove Reassign Signer	Modified topic. Corrected applied.

Document	Publication Date	Topic	Description
	 13 Jul 2021	Business Units	Modified topic. Removed warning notification.
	 09 Jun 2021	Attachment Tags	Modified topic. Updated size limitation of documents.
	 07 Jun 2021	Facilitator Notifications	Modified topic. Added content regarding Facilitator email notifications.
		Audit Trail	Modified topic. Added clarification of Signer and CC audit designations.
		Business Units Subject and Email Line	Updated topic.
		Remove Reassign Signer	New topic.
		Conga Sign Parameter	Updated topic. "reassignAllowed" added.
		Configure Email Notification Frequency	New topic.

Spring '21

Document	Topic	Description
Spring '21	Setting for Conga Sign Connected App	Modified topic. Added content about Refresh Token Policy.

Document	Topic	Description
	What's in the Conga Sign Package?	Modified topic. Updated custom object and tabs list.
	Using Conga Sign with Conga Composer	Modified topic. Added a note about sending multiple documents.
	Sending documents for eSignature	Modified topic. Added content about sending multiple documents for eSignature.
	Signing a Document	Modified topic. Added content about signing multiple documents in a single transaction.
	Setting Guided Signing	New topic.
	Removing Conga Logo from Signature and Initial	New topic. Added missing feature.
	Configuring Require Drawn Signature Setting	New topic. Added missing feature.
	Signing a Document	Modified topic. Added content about Guided Signing.
	In Person Signing	Modified topic. Added content about Guided Signing.

Winter '20

Document	Topic	Description
Winter '20	Attachment Tags	New Topic
	List of Conga Sign Signature Tags	Updated Topic
	Audit Trail	Updated Topic
	Business Units	New Topic
	Configuring Business Units	New Topic
	Choosing a Default Business Unit	New Topic
	Customizing Your Logo	Updated Topic
	Customizing Emails	Updated Topic
	Creating New Recipients in Create Transaction UI	New Topic
	Sending Documents for eSignature	Updated Topic
	Conga Sign Parameters	Updated Topic

Conga Sign for Administrators

With the Conga Sign Administrator Guide, you can find out how Conga Sign works and how to manage your organization's and your customers' eSignature workflows.

Topic	Description
What's Covered	Conga Sign Administrator Guide is designed to provide administrators with information on configuring and setting up Conga Sign.
Primary Audience	Contract Manager, Conga Sign Administrator
IT Environment	Refer to the latest Release Notes for information on System Requirements and Supported Platforms.
Updates	For a comprehensive list of updates to this guide for each release, refer to What's New in Conga Sign Documentation topic.
Other Resources	<ul style="list-style-type: none"> • <i>Conga Sign for Users</i>: Refer to this guide to set-up the entire Conga Sign process.

Select one of the following topics for more information:

- [Conga Sign Technical Requirements](#)
- [Installing Conga Sign Package](#)
- [Post-Installation Configuration](#)
- [License Provisioning Tips](#)
- [Customizing Your Logo](#)
- [Customizing Emails](#)
- [Conga Sign Parameters](#)
- [Conga Sign Redirect URL](#)
- [Automatically Upload a Conga Sign Document to a Third Party Repository](#)
- [Using Conga Sign in Salesforce Communities](#)
- [Using Conga Sign with Conga CLM](#)
- [Using Conga Sign with Conga Composer](#)
- [Using Conga Sign with Conga Contracts for Salesforce](#)
- [Setting up a Process Builder for Conga Sign Transaction](#)

Conga Sign Technical Requirements

To install and successfully use Conga Sign, you must have one of the following from each category:

Salesforce® Edition:


- Performance / Unlimited
- Enterprise
- Developer
- Professional

Salesforce® Products:

- Sales Cloud
- Service Cloud
- [Force.com](#) / App Cloud
- Health Cloud
- Financial Services Cloud
- Quote-to-Cash

Internet Browsers (latest version):

- Google Chrome®
- Mozilla Firefox®
 - Mozilla Firefox Settings: to ensure Salesforce works optimally with Firefox, [update the browser's configuration settings](#).
- [Supported Browsers for Salesforce Classic](#): Salesforce Classic is supported with Microsoft® Internet Explorer® version 11 or higher, and Apple® Safari® version 11.x on macOS. The most recent stable versions of Microsoft Edge, Mozilla® Firefox®, and Google Chrome™ are also supported. There are some limitations.
- [Supported Browsers for Salesforce Lightning Experience](#): See the supported browsers and limitations for Lightning Experience.

 Conga Sign does not support Salesforce Lightning in Microsoft® Internet Explorer® version 11. Some Conga Sign functions may continue to work, but it is highly recommended to use a different browser or to use Salesforce Classic when using Microsoft® Internet Explorer® version 11. If you use Internet Explorer, we recommend using the latest version that Salesforce supports. Apply all Microsoft software updates.

Best Practice Guidelines

For the best experience and security, [follow these guidelines](#).

- For all browsers, enable JavaScript, cookies, and TLS 1.2. If TLS 1.2 isn't available, enable TLS 1.1. Browsers that do not support TLS 1.1 or TLS 1.2 won't be able to access Salesforce after we deactivate TLS 1.0. Deactivation has already occurred in sandbox orgs and concludes with production orgs on July 22, 2017.
- Some features use iframes to work in Lightning Experience, so you must have the appropriate browser security settings for iframes to load correctly. Follow your browser's instructions to enable working with iframes and third-party cookies. For example: In Internet Explorer, set the Launching programs and files in an IFRAME security setting to Enable or Prompt. In Safari, we recommend changing your privacy settings for cookies to Allow from websites I visit. In Chrome, make sure that Block third-party cookies and site data isn't selected in the Content settings.
- When using Salesforce Professional Edition, Conga Sign will work as an end to end product, but does not support uploading documents in the Create Transaction screen due to API limitations put in place by Salesforce.
- The minimum screen resolution required to support all Salesforce features is 1024 x 768. Lower screen resolutions don't always properly display Salesforce features such as Report Builder and Page Layout Editor.
- For Mac OS users on Apple Safari or Chrome, make sure the system setting Show scroll bars is set to \ Always.
- Some third-party browser plug-ins and extensions can interfere with the functionality of Chatter. If you experience malfunctions or inconsistent behavior with Chatter, disable the browser's plug-ins and extensions and try again.

Template Software

- Microsoft Office® 2016 (Mac or PC)
- Adobe Acrobat® XI, Acrobat Reader

 Adobe LiveCycle PDFs are not supported.

Mobile

- Compatible with Salesforce and all mobile browsers when configured following the documentation.

SMS Authentication Feature

To use the SMS Authentication feature, Conga Sign users must have a minimum of read access to the fields below on the Contact, Lead, and User objects.

- **Contact:** MobilePhone and MailingCountry
- **Lead:** MobilePhone and Country
- **User:** MobilePhone and Country

Installing Conga Sign Package

You can install the Conga Sign package into your Salesforce instance from Salesforce AppExchange.

To install Conga Sign

1. Go to [Salesforce AppExchange](#).
2. Click **Get It Now** and follow the on-screen prompts to install Conga Sign.
3. Log in to the AppExchange. You must provide System Administrator credentials to install an application from the AppExchange.
4. Specify where you want to install Conga Sign:
 - Click **Install in Production** to install in your production instance.
 - Click **Install in a Sandbox** to install in a sandbox.
5. Read and accept the terms and conditions.
 - Select the I have read and agree to the terms and conditions box.
 - Click **Confirm and Install**.

As a best practice, and to avoid access issues for end users, Conga recommends selecting Install for All Users.

6. Click **Install**.

7. Approve Third-Party Access for all 3 websites selected.
 - Check **Yes**, grant access to these third-party websites.
 - Click **Continue**.
8. Click **Done**. If the package takes too long to install, you will receive an email letting you know when it is done.
9. After clicking Done, you will be on the Installed Packages page in Salesforce Setup. Click on **Manage Licenses** next to the Conga Sign managed package link.
10. Administrators are assigned a license by default. Continue [assigning licenses](#) to other users who need access to Conga Sign.

✓ Conga Sign document recipients do not need a Conga Sign license. Only users who will be preparing documents to send for signatures need a license.

After assigning licenses, you must [Configuring Conga Sign](#) for use in your organization.

Post-Installation Configuration

After installing the Conga Sign package, you must configure the following Conga Sign Admin Settings.


- [Configuring Conga Sign](#)
- [Permissions Needed to Use Conga Sign](#)
- [Settings for Conga Sign Connected App](#)
- [Conga Sign Email Change](#)
- [What's in the Conga Sign package?](#)
- [Adding Sign Button and Transactions Related List to Page Layout](#)
- [Adding Conga Sign Button to a Page Layout Manually](#)
- [Configuring an Object for Conga Sign Manually](#)
- [Creating a Conga Sign button Manually](#)
- [Business Units](#)
- [Setting Final Document Delivery Options](#)
- [Setting Guided Signing](#)
- [Adding Reassign Signer Button](#)
- [Remove Reassign Signer](#)
- [Removing Conga Logo from Signature and Initial](#)
- [Configuring Require Drawn Signature Setting](#)

Configuring Conga Sign

The Conga Sign configuration process involves running the Automatic Configuration from the Conga Sign Setup tab and modifying the page layout of the object in which you're utilizing Conga Sign.


1. Select a **Conga Sign Processing Region**.

- Navigate to the **Conga Sign Setup** tab.
- In **Conga Sign Processing Region** section, select a specific region to connect to for data processing and storage related to Conga Sign. Generally, Conga suggests selecting the same region as your Salesforce org or your primary business location.

 This selection is permanent and is only available in orgs without existing region connections. Existing connections cannot be modified.



2. **Connect with OAuth.**

- After selecting a Conga Sign Processing Region, click **Connect** in the **Connect with OAuth** section.

 We recommend using an Integration User account to complete this step (an account that no one uses normally). The person completing this step is the person under whose authority data from Conga Sign syncs with your Salesforce org.

- Click **Allow**.

3. Run the **Automatic Configuration** tool.

- Modifying the object's page layout to include the Send with Conga Sign button and add the Conga Sign Transactions related list.
- Click **Start Automatic Configuration** under **Organization Configuration**.
- Select the object(s) you want to utilize with Conga Sign (limit 10 per run. If you need to add more than 10 objects, you can run Automatic Configuration again).
- Click  icon to add an object(s).
- Click **Next**.
- Select the profiles permitted to use Conga Sign (limit 5 per run. If you need to add more than 5 profiles, you can run Automatic Configuration again).
- Click  icon to add profile(s).
- Click **Save**.

You see a green checkbox in the Organization Configuration box when


Automatic Configuration is complete. See [Adding a Sign button to the page layout for next steps](#).

Connecting to Conga Sign with OAuth

Conga Sign integrates with your Salesforce org using the login credentials of a user of your choice. This integration is done through an [OAuth connection](#). The specific user connecting with OAuth should have the [necessary permissions for Conga Sign](#) and sufficient permissions to modify records in Salesforce.

To connect to Conga Sign with OAuth

1. Log in to Salesforce as the designated user.
2. Click the **Conga Sign Setup** tab.
3. Click **Connect** in the **Connect with OAuth** section.

 You must select a Conga Sign Processing Region before clicking Connect to connect with OAuth.

4. Click **Allow**.

After successfully connecting, the Status field will display as "Connected" and show the specific Conga Sign Processing region your Salesforce org is connected to.

Permissions Needed to Use Conga Sign


Connected Service User

- Package License granted
- Read access to objects used with (for example, Opportunity)
- Read/Write access to Conga Sign Transaction [APXT_CongaSign__Transaction__c] and all fields
 - Including any setup generated lookup fields (e.g Opportunity [Parent_006__c])
- Read/Write access to Conga Sign Document [APXT_CongaSign_Document__c] and all fields
- Read/Write access to Conga Sign Recipient [APXT_CongaSign__Recipient__c] and all fields
- Permission to Apex Classes: APXT_CongaSign__apxt_setup
- Permission to VF pages: APXT_CongaSign__apxt_setup

- Permission on the User Record: Salesforce CRM Content User is enabled
- Profile Permission → Administrative Permissions → Manage Salesforce CRM Content is enabled

Regular Users

- Package License granted
- Read access to objects used with (for example, Opportunity)
- Read access to Conga Sign Transaction [APXT_CongaSign__Transaction__c] and all fields
 - Including any setup generated lookup fields (e.g Opportunity [Parent_006__c])

 To allow non-admin users to cancel Conga Sign Transactions that they are not the record owner of, you must provide them both Read and Edit Access to the Conga Sign Transaction object

- Read/Write access to Conga Sign Document [APXT_CongaSign_Document__c] and all fields
- Read access to Conga Sign Recipient [APXT_CongaSign__Recipient__c] and all fields

Permission to Apex Classes

- APXT_CongaSign__apxt_cancelTransaction
- APXT_CongaSign__apxt_contentInfo
- APXT_CongaSign__apxt_coretransactionservice
- APXT_CongaSign__apxt_resendTransaction
- APXT_CongaSign__apxt_sendForSignature
- APXT_CongaSign__apxt_serviceRedirect
- APXT_CongaSign__apxt_setup
- APXT_CongaSign__apxt_conga_sign.apxt_transactionlinkcontentdoc
- APXT_CongaSign__apxt_conga_sign.apxt_transactionlinksrcobject

Permission to VF pages


- APXT_CongaSign__apxt_auditTransaction
- APXT_CongaSign__apxt_cancelTransaction
- APXT_CongaSign__apxt_contentInfo
- APXT_CongaSign__apxt_editTransaction
- APXT_CongaSign__apxt_resendTransaction

- APXT_CongaSign__apxt_sendForSignature
- APXT_CongaSign__apxt_serviceRedirect
- APXT_CongaSign__apxt_setup

Permission on the User Record

- Salesforce CRM Content User is enabled
- Profile Permission → Administrative Permissions
 - Manage Salesforce CRM Content is enabled


Settings for Conga Sign Connected App

 It is highly recommended to configure the IP Relaxation and Refresh Token Policy settings for the Conga Sign Connected App, as mentioned in this article.


Relax IP Restrictions in the Conga Sign Connected App by March 31, 2020, to avoid delayed or failed delivery of Conga Sign documents and transactions back to your Salesforce org. Adjusting the Relax IP Restriction setting for Conga Sign ensures that seamless updates to Conga Sign are delivered in the future.

To configure Conga Sign Connected App

1. Navigate to **Setup** in Salesforce.
2. Type Connected Apps in the Quick Find search box.
3. Click **Manage Connected Apps**.
4. Click **Conga Sign**.
5. Click **Edit Policies**.
6. Set the IP Relaxation field's value from **Enforce IP restrictions** to **Relax IP restrictions**.

 If the IP Relaxation is set to Enforce IP restrictions, Conga Sign may not be able to connect to the org as the Conga Sign IP address used to connect is not always the same.

7. Set the Refresh Token Policy field's value to **Refresh token is valid until revoked**.

 If the Refresh Token Policy is set to another policy, Conga Sign may be disconnected when the refresh token expires.

8. Click **Save**.

Click [here](#) to watch a video on how to update Conga Sign IP address settings.

Conga Sign Email Change

The Conga Team has migrated Conga Sign services to a new infrastructure on July 8, 2020. As a result of this change, IP addresses used or referenced to whitelist Conga Sign emails are no longer supported.

Conga has taken the necessary steps to ensure that all emails originating from the Conga Sign service are valid. Additionally, Conga has protections in place to prevent spam and spoofing attempts, which makes whitelisting Conga Sign emails mostly unnecessary.

Instead of whitelisting specific Conga IP addresses, it is highly recommended to use DKIM and SPF validation in your inbound email server software, if it is not already enabled. Also consider adding the domain "conga-sign.com" to an allow list in your email server configuration, instead of using a dedicated IP address.

What's in the Conga Sign package?

Buttons

- Conga Sign Setup
- View Audit
- Download Original Version
- View File
- Edit Draft
- Cancel Transaction
- Download Final Version

Custom Objects

- Conga Sign Transaction
- Conga Sign Document
- Conga Sign Settings
- Conga Sign Recipient

Page Layout

- Transaction Layout
- Recipient Layout

Custom Tabs

- Conga Sign Transaction
- Conga Sign Setup
- Conga Sign Recipients

Adding Sign Button and Transactions Related List to Page Layout

To add button and related list to the object's page layout

1. Navigate to the **Object Manager** in Salesforce **Setup**.
2. Click on the object where you configured Conga Sign.
3. Click **Page Layouts**.
4. Click **[Object] Layout**.
5. Click **Buttons > Send with Conga Sign** and drag onto the page's Custom Buttons section.
6. Click **Mobile & Lightning Actions > Send with Conga Sign** and drag onto the page's Salesforce Mobile and Lightning Experience Actions section.
7. Click **Related Lists > Conga Sign Transactions** and drag to the page's Related Lists section.
8. Click **Save**.
9. OPTIONAL: The Conga Sign Transactions Related Listed defaults to recommended columns. If you want to customize the columns:
 - Click the wrench icon to access the related list properties.
 - Add desired fields to the Related List and arrange them in the preferred order.
 - OPTIONAL: Sort by descending for newest on top.
 - Click Ok.
10. Save the page layout.

The Conga Sign configuration process involves running the Automatic Configuration from the Conga Sign Setup tab and modifying the page layout of the object in which you're utilizing the Conga Sign.

Adding Conga Sign Button to a Page Layout Manually

To add a Conga Sign button or link to a page layout in the Lighting UI

1. Navigate to **Setup > Objects and Fields > Object Manager > [Object] > Page Layouts** and select the page layout. The Page Layout Properties screen appears.
2. Click **Buttons** or **Custom Links**.
3. Drag and drop your new button to the Custom Button area on the page layout.
4. Click **Save**.

To add a Conga Sign button or link to a page layout in the Classic UI

1. Navigate to **Setup > Customize > [Object] > Page Layouts** and select **Edit** next to the desired page layout. The Page Layout Properties screen appears.
2. Click **Buttons** or **Custom Links**.
3. Drag and drop your new button to the **Custom Button** area on the page layout.
4. Click **Save**.

Configuring an Object for Conga Sign Manually

Use the Automatic Configuration in Conga Sign Setup to configure objects for Conga Sign.

Prerequisites:

- [Install Conga Sign from the Salesforce AppExchange](#)
- [Connect to OAuth in Conga Sign Setup](#).

As an alternative to using the [Start Automatic Configuration feature in Conga Sign Setup](#), you can also manually set up a Salesforce object for Conga Sign.

To manually configure a Salesforce object for Conga Sign

1. Find the 3-digit prefix of the object you are configuring for Conga Sign. For a list of standard object 3-digit prefixes, see [Standard Field Record ID Prefix Decoder](#). You can also find the 3-digit object prefix by navigating to an object record in Classic and copying the first 3 digits of the full record ID.
2. Navigate to **Setup** in Salesforce.
3. Toggle open Create and select Custom Objects.
4. Click **Conga Sign Transaction**.
5. Under Custom **Fields and Relationships**, click **New**.
6. Choose **Lookup Relationship** and click the **Next** button.
7. Select the object that you are configuring for Conga Sign and click Next.
8. Use the Field Label that Salesforce automatically generates. Change the **Field Name** to Parent_[3-digit object prefix] and click **Next**. Example Field Name for custom object: Parent_a5G
9. Choose the desired assignments for Field Level Security, Page Layouts, and the Related List.
10. Click **Save**.
11. Create a button. For more information see: [Manually create a Conga Sign Button](#)

To manually configure an object for Conga Sign in lightning

1. Find the 3-digit prefix of the object you are configuring for Conga Sign. For a list of standard object 3-digit prefixes, see [Standard Field Record ID Prefix Decoder](#). You can also find the 3-digit object prefix by navigating to an object record in Classic and copying the first 3 digits of the full record ID.
2. Navigate to **Setup** in Salesforce.
3. Under the Platform Tools section, toggle open Objects and Fields and select **Object Manager**.
4. Select and access the object that you are configuring for Conga Sign.
5. Select **Fields and Relationships** and click the **New** button.
6. Choose **Lookup Relationship** and click the **Next** button.
7. Select the object that you are configuring for Conga Sign and click Next.
8. Use the Field Label that Salesforce automatically generates. Change the **Field Name** to Parent_[3-digit object prefix] and click **Next**. Example Field Name: Parent_a5G

9. Choose the desired assignments for Field Level Security, Page Layouts, and the Related List.
10. Click **Save**.
11. Create a button. For more information see: [Manually create a Conga Sign Button](#).

Creating a Conga Sign button Manually

Several buttons are necessary when using Contracts with Salesforce CPQ. This page describes how to create and place them.

To create a Conga Sign button

1. Navigate to **Setup > Object Manager** tab.
2. Select the appropriate Object.
3. Click **Buttons, Links, and Actions**.
4. Click **New Button or Link**.
5. Enter the following information:
 - **Label:** Send for Conga Sign
 - **Display Type:** Detail Page Button
 - **Behavior:** Display in a new window
 - **Button or Link URL:** /apex/APXT_CongaSign__apxt_sendForSignature?id={!Object.Id}

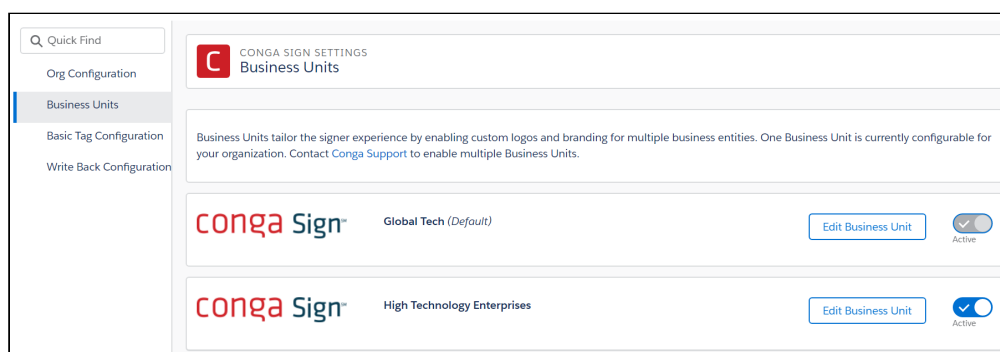
To add Buttons to the selected Object

1. In Salesforce Lightning, navigate to the Object layout page where you selected the Object on the **Object Manager** tab.
2. Click the **Page Layout** name. The edit screen opens.
3. Click **Buttons**.
4. Drag your Conga Sign button to the Custom buttons area.
5. Click **Mobile & Lightning Actions**.
6. Drag your Conga Sign button to the Salesforce Mobile and Lightning Experience Actions area.
7. Click **Save**.

Business Units

Business Units allow administrators to configure branding in Conga Sign Transactions and emails for specific organizations, departments, and entities within a Salesforce org. Conga Sign users can use the default Business Unit, or select a different Business Unit, for each Conga Sign Transaction to include specific branding for the signing experience.

Business Units are located and configured in the Business Units section of Conga Sign Setup.



Business Units allow administrators to configure the following fields:

- **Business Unit Name** - defines the Business Unit name
 - The Business Unit Name is displayed in the Business Unit picklist on the Create Transaction user interface and is also displayed in the Audit Trail.
- **Business Unit API Name** - used to specify a Business Unit with the [Conga Sign businessUnit parameter](#) and the *Composer CSBusinessUnit parameter* topic in *Composer Parameter Guide*.
 - The Business Unit API name can only contain underscores and alphanumeric characters. It must be unique for the org, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.
- **Business Unit Status** - determines if a Business Unit is activated and available for use in Conga Sign Transactions and emails.
- **Company Logo** - displays the selected logo in Conga Sign emails and documents.
- **Email From Display Name** - defines the company name displayed on Conga Sign emails.
- **Email Subject** - defines the subject field in emails sent from Conga Sign.
- **Email Message** - defines the content displayed in the body of emails sent from Conga Sign.
- **Contact Information fields** - defines the contact information displayed at the bottom of emails sent from Conga Sign.
 - Company
 - Address, City, State, Zip/Postal Code, Country

- Phone Number
- Website

With the multiple Business Units feature enabled, Conga Sign users can easily select a specific Business Unit to use for each Conga Sign Transaction under the Advanced Options section in the Create Transaction user interface.

Advanced Options

MESSAGE

Please update the Subject and Message below for a custom message.

Subject

Message

Choose Business Unit

Business Unit

Global Tech Inc

The default Business Unit auto-populates as the selected Business Unit in each Conga Sign Transaction. For more information about choosing a default Business Unit, see [Choosing a Default Business Unit](#).

Conga Sign Transaction

CS-000000


Status	Write Back Status	Business Unit	Created On	Completed On
SENT	NONE	Conga America	7/10/2020, 8:17 AM	

Related Details

Information


Transaction Name	CS-000000	Status	SENT
Sender Name	User User	Write Back Status	NONE
Sender Email	sally.sales@conga.com	Created On	7/10/2020, 8:17 AM
Business Unit	Conga America	Completed On	


Each Salesforce org is entitled to one Business Unit. After configuring the initial Business Unit, it is then selected as the default Business Unit and used for all Conga Sign Transactions. Users can change the default Business Unit to a different Business Unit if the multiple Business Units feature is enabled.

-  Information previously stored in the Email Configuration section of Conga Sign Setup is automatically carried over and applied to an org's default Business Unit in orgs that had a version of Conga Sign preceding version 1.49.0. The Company Logo is also moved from the Org Configuration section and carried over to the default Business Unit.

A default Business Unit is automatically created and enabled if any of the following criteria were met in Salesforce orgs that had a version preceding Conga Sign version 1.49.0:

- A Company Logo was previously uploaded in the Org Configuration section of Conga Sign Setup
- Any field in the Email Configuration section of Conga Sign setup was previously filled out
- The Require Drawn Signature setting in the Org Configuration section of Conga Sign Setup is enabled
- The Hide Signature Logo setting in the Org Configuration section of Conga Sign Setup enabled

 Orgs with new installations of Conga Sign version 1.49.0 will not have a default Business Unit automatically created.

 The Company Logo, Email Configuration fields, Require Drawn Signature setting and Hide Signature Logo setting are moved to the Business Units section of Conga Sign Setup.


If none of the criteria above are met, you must configure and enable a Business Unit in a Conga Sign Setup. If no Business Unit is configured, Conga Sign will apply its default Conga Sign branding for all Conga Sign Transactions. For information about configuring Business Units, see [Configuring Business Units](#).

Configuring Business Units


Configure Business Units in Conga Sign Setup to enable custom branding for specific organizations, departments, and entities within your org.

To configure Business Units


1. Navigate to **Conga Sign Setup**.
2. Click **Business Units**.
3. Click **Edit Business Unit** next to a specific Business Unit.

 Administrators only have access to the default Business Unit unless the multiple Business Units feature is enabled.

4. Modify the existing values in the **Business Unit Name** and **Business Unit API Name** fields or enter new field values.

 The Business Unit Name is displayed in the Business Unit picklist on the Create Transaction user interface and is also displayed in the Audit Trail. The Business Unit API Name is used to specify a Business Unit with the [businessUnit Conga Sign parameter](#) and the *CSBusinessUnit Composer parameter* topic in *Composer Parameter Guide*. The Business Unit API Name can only contain underscores and alphanumeric characters. It must be unique for the org, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.

5. Click Upload Files to upload a new Company Logo for the Business Unit.
6. Configure email branding for the Business Unit by modifying the values of the following fields.
 - a. Email Display Name
 - b. Subject
 - c. Message

 Information previously stored in the Email Configuration section of Conga Sign Setup is automatically carried over and applied to an org's default Business Unit in orgs that had a version of Conga Sign preceding version 1.49.0. The Company Logo is also moved from the Org Configuration section and carried over to the default Business Unit.


7. Configure the contact information for the Business Unit by entering values in the following fields.
 - a. Company
 - b. Address
 - c. City, State, and Zip/Postal Code
 - d. Country
 - e. Phone Number
 - f. Website
8. Click Save.
9. Enable the Business Unit.
 - If you are configuring the default Business Unit, it is enabled by default and no further action is required.

- If you are configuring a Business Unit that is not the default Business Unit, click the Set Status toggle to change the value from Inactive to Active.

 Default Business Units are set to active by default. You must activate non-default Business Units to use them in Conga Sign Transactions and emails.

Orgs with the multiple Business Units feature enabled can also choose a new default Business Unit. For information on choosing a default Business Units, see [Choosing a Default Business Unit](#).


Choosing a Default Business Unit

 Choosing a default Business Unit requires the multiple Business Units feature. To enable the multiple Business Units feature, [contact Conga Support](#). Without the multiple Business Units features enabled, users only have access to modify the singular default Business Unit.

The default Business Unit auto-populates as the selected Business Units for all Conga Sign Transactions and emails. Users also have the option to select a different Business Unit in the Create Transaction user interface if the multiple Business Units feature is enabled. Additionally, administrators can use the [businessUnit parameter](#) in a native Conga Sign button, or the *CSBusinessUnit* topic in Composer parameter in a Conga Sign-integrated Composer solution, to automatically populate a specific Business Unit in new Conga Sign Transactions.

A default Business Unit is automatically created and enabled if any of the following criteria were met in Salesforce orgs that had a version preceding Conga Sign version 1.49.0:

- A Company Logo was previously uploaded in the Org Configuration section of Conga Sign Setup
- Any field in the Email Configuration section of Conga Sign setup was previously filled out
- The Require Drawn Signature setting in the Org Configuration section of Conga Sign Setup is enabled
- The Hide Signature Logo setting in the Org Configuration section of Conga Sign Setup enabled

 Orgs with new installations of Conga Sign version 1.49.0 will not have a default Business Unit automatically created.

To set a default Business Unit

1. Navigate to **Conga Sign Setup**.
2. Click **Business Units**.
3. Click **Edit Business Unit** next to a Business Unit record.
4. Click the **Set as Default** toggle to change the value from Disabled to Enabled.
5. Click **Change the Default**.

The Business Unit is now selected as the default Business Unit and auto-populates as the selected Business Unit in the Create Transaction user interface.

Sender Email Notification Frequency

Configure Email Notification frequency to control how frequently your notifications are sent.

To configure notification frequency

1. Navigate to **Conga Sign Setup**.
2. Click on **Business Unit**.
3. Select **Email Notification Frequency** and check the boxes for each value you want to enable/disable. The following are the options available.
 - **Transaction is Complete** - When enabled, sends email notification when Transaction has been completed by all Signers.
 - **Transaction is Cancelled** - When enabled, sends email notification when Transaction has been canceled by the Signer or the Sender.
 - **Transaction is Expired** - When enabled, sends email notification when Transaction has reached its marked expiration date.
 - **Transaction is Sent** - When enabled, sends email notification when Transaction has been sent out for signature.
 - **Transaction is Reassigned** - When enabled, sends email notification when the recipient has chosen to reassign the signer.
 - **Transaction is completed by Signer** - When enabled, sends email notification when each Signer has completed signing.
 - **Transaction is viewed by Signer** - Two sub-options for this feature:
 - **Document view first time only** - When enabled, sends email notification only when the received document is viewed for the first time.
 - **Document view every time** - When enabled, sends email notification every time the received document is viewed by a recipient.

Transaction Expirations and Reminders

Transaction Expirations and Reminders can be configured for each Business Unit or at the Business Unit level.

To configure expirations and reminders

1. Navigate to **Conga Sign Setup**
2. Click on **Business Unit**
3. Modify any of the following sections:

Transaction Expiration - determines the number of days and the time the transaction will expire.

Expiration Reminder - determines the number of days before an expiration email is sent to recipients and the time at which it is sent.

Time Zone Conversion - determines if the times set for Expiration and Reminder will be set to the time zone of the user sending the transaction or if it will match the **Time Zone** designated for the business unit.

Time Zone - determines the time zone in which Expirations and Reminders will take place. This will only be available when the "Always Use Business Unit time zone" option is enabled.

Days to First Reminder - determines the number of days before Conga Sign sends the first reminder email to recipients.

Reminder Frequency - determines the frequency with which Conga Sign sends reminder emails to recipients. Reminders are limited to sending for 120 days or up to the expiration date (whichever comes first).

Transaction Expiration <small>Transaction Expiration determines the number of days before a transaction expires and the time at which the expiration will take effect.</small>	Time: 11:45 PM ⓘ Days: 14
Expiration Reminder <small>Expiration Reminder determines the number of days before a reminder email is sent to recipients and the time at which the reminder will be sent.</small>	Time: 8:00 AM ⓘ Days: 12
Time Zone Conversion <small>Determines if Transaction Expiration Reminder and Transaction Expiration times will be converted to the Sending User's timezone upon sending, or if the time will match the timezone set for the Business Unit.</small>	<input checked="" type="radio"/> Convert to sender's time zone <input type="radio"/> Always use Business Unit time zone
Time Zone <small>Determines the timezone in which reminders will be sent and transactions will expire.</small>	(GMT -05:00) EST ▼
Days to First Reminder <small>Days to First Reminder determines the number of days before Conga Sign sends a reminder email to recipients. Use the Reminder Frequency option to configure recurring reminders.</small>	Days: <input type="text"/>
Reminder Frequency <small>Reminder Frequency determines the frequency with which Conga Sign sends reminder emails to recipients. <small>Note: If you want reminders sent with the same frequency, you can use this field without needing to populate the Days to First Reminder field.</small></small>	Days: <input type="text"/> 2

Recipient Email Notifications

Configure Recipient Email Notification frequency to control how frequently recipients receive notifications during the transaction process.

To configure notification frequency

1. Navigate to **Conga Sign Setup**.
2. Click on **Business Unit**.
3. Select **Recipient Email Notification Frequency** and check the boxes for each value you want to enable/disable. The following are the options available.
 - **Signer:**
 - **Transaction is Complete** - When enabled, sends an email notification when the Transaction has been completed by all Signers.
 - **Transaction is Reassigned** - When enabled, sends an email notification when the recipient has chosen to reassign the signer.
 - **Transaction is Expired** - When enabled, send an email notification when the Transaction has reached its marked expiration date.
 - **Signing Reminder** - When enabled, sends email notifications to remind the recipient to sign the Transaction.
 - **Transaction Expiration Reminder** - When enabled, sends an email notification reminder that the Transaction will expire.
 - **CC:**
 - **Transaction is Complete** - When enabled, sends an email notification when the Transaction has been completed by all Signers.
 - **Transaction is Expired** - When enabled, sends an email notification when the Transaction has reached its marked expiration date.
 - **Transaction is canceled by Signer or Sender manually** - When enabled, sends an email notification when the Transaction has been canceled by either the Signers or Sender.
 - **Transaction is canceled by Signer multi-factor authentication failure** - When enabled, sends an email notification when the Transaction has been canceled by the Signer's SMS verification code is entered incorrectly, resulting in a multi-factor authentication failure.

Setting Final Document Delivery Options


Conga Sign allows you to choose how completed documents are delivered to recipients. Navigate to **Conga Sign Setup** and select one of the options described below. The selected method applies to all Conga Sign Transactions created moving forward.

Final Document Delivery Options

Administrators can choose from the following three final document delivery methods in the Final Document Delivery section of Conga Sign Setup:

- **PDF Attachment**

- The final document delivered as a PDF attachment in an email.


 This is the default final document delivery method for Conga Sign Transactions.

- **Download Link**

- The final document is downloadable through a download link delivered in an email.
- Upon clicking the **View Document** button in the initial email, users then click the **Download Document** button to download the final document as a PDF file.
- The download link delivery option supports final documents with a file size of up to 50MB.
- The download link expires five days from when it is delivered in an email. Recipients can simply click the Send new link button in the expired download link window to request a new download link.

- **Attachment and Link**

- The final document is attached as a PDF file in an email and is also downloadable as a PDF file through a download link.
- All information from the options above applies to the Attachment and Link option.

 Final documents greater than 10MB are only accessible through a downloadable link, regardless of the selected delivery option.

Setting Guided Signing

This feature allows signers to navigate to the locations of each assigned Conga Sign tag in the document while signing. By default, the Guided Signing setting is disabled. You can enable the Guided Signing setting from Conga Sign Setup.

To enable guided signing

1. Go to **Conga Sign Setup**.
2. Navigate to the **Guided Signing** section.
3. By default, the Guided Signing setting is disabled. Click the toggle for **Guided Signing** to enable it.

The Guided Signing is enabled and when signing, signers can use the navigation button to auto-navigate to the locations of each assigned tag in a Conga Sign document. When the signers are ready to sign, they can click Begin button that navigates them to the first tag needing their attention. Once completing the first tag, they can click to Next button to navigate to the next tag in the document. For more information on how to use Guided Signing, refer to [Signing a Document](#).

Adding Reassign Signer Button

Conga Sign offers users in Salesforce the option to reassign a signer's designated signature responsibilities to a different recipient after the Conga Transaction is sent. Reassigning signers is useful in scenarios where a Conga Sign user wants to designate signature responsibilities to a different person within their Salesforce org.

To add reassign signer button

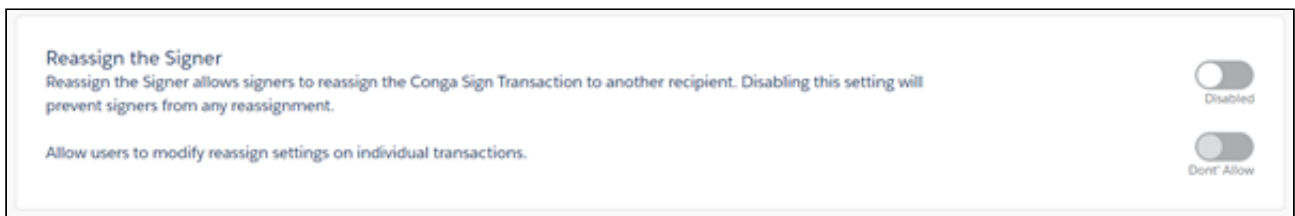
1. Navigate to **Conga Sign Setup** and click **Reconnect** to ensure the Salesforce org is connected to OAuth.
2. Click **Start Automatic Configuration** in Conga Sign Setup and complete the Automatic Configuration process.
3. Add the **Reassign** button to the **Recipient** object page layout.
 - Navigate to **Salesforce Setup**.
 - Use the Quick Find search box to navigate to Object Manager in Lightning, or Objects under the Create section in Classic.
 - Select the **Conga Sign Recipient** object.
 - Click **Page Layouts** and select the page layout in use.
 - Click **Buttons** in the page layout menu.
 - Drag and drop the **Reassign Signer** button into the **Custom Buttons** section of the page layout and click **Save**.

Remove Reassign Signer

Remove Reassign Signer is an optional setting that allows Administrators to enable or disable the Reassign Signer option. Reassign Signer allows a recipient to reassign the Conga Sign Transaction to a different recipient. Disabling this setting prevents signers from reassigning.

To setup reassign signer


1. Navigate to **Conga Sign Setup** and go to **Org Configuration**.
2. Under **Reassign the Signer**, modify the toggles to enable/disable Reassign Signer from either the organization level or transaction level respectively.



Enabled per Organization	Enabled per Transaction	Result
Enabled	Disabled	Reassign is not available on the Create Transaction screen but is available on the received email.
Enabled	Enabled	Reassign is available on the Create Transaction screen and is also available on the received email.
Disabled	Disabled	Reassign is disabled for the organization and not available on the received email.

⚠ If Reassign Signers is disabled for the entire Organization, it prevents recipients from reassigning from individual transactions.

Removing Conga Logo from Signature and Initial


 This feature is currently available by request only. To enable **Remove Conga Logo From Signatures and Initials** setting in Conga Sign Setup, contact [Conga Support](#).

Conga Sign allows you to remove the Conga logo from signature and initial tags in Conga Sign documents. You can enable this setting while configuring Business Units in Conga Sign Setup.

To remove Conga logo from signature and initial

1. Navigate to **Conga Sign Setup**.
2. Click the **Business Units** tab.
3. Click **Edit Business Unit** next to a specific Business Unit.
4. Goto **Remove Conga Logo From Signature and Initial** section.
5. By default, the setting is disabled. Click the toggle for **Remove Conga Logo From Signature and Initial** to enable it.

Configuring Require Drawn Signature Setting

 This feature is currently available by request only. To enable the **Require Drawn Signature** setting in Conga Sign Setup, contact [Conga Support](#).

You can enable the Require Drawn Signature setting to require Signers to draw a signature, instead of selecting one of the default signature styles. This is a global setting that applies to all Conga Sign Transactions. You can enable this setting while configuring Business Units in Conga Sign Setup.

To configure Require Drawn Signature setting

1. Navigate to **Conga Sign Setup**.
2. Click the **Business Units** tab.
3. Click **Edit Business Unit** next to a specific Business Unit.
4. Go to **Require Drawn Signature** section.
5. By default, the setting is disabled. Click the toggle for **Require Drawn Signature** to enable it.

License Provisioning Tips

My sandbox license has expired, how do I extend it?

- You can submit a case to have your sandbox co-termed with your subscription. If you are a Conga Communities User, then please submit the case through [Conga Communities](#).
- It is a best practice to submit cases when you refresh your sandboxes, as all refreshes come with 30-day trials.

Why does my Sandbox have an expiration countdown, but my production org doesn't?

- All Sandbox Orgs are provisioned as trial licenses. Your production org is provisioned per your subscription term and only displays the expiration date 14 days prior to expiration.
- You can always submit a case to have your sandbox co-termed with your subscription.

I don't see "manage licenses" in my sandbox as I do in production, why?

- We do not enable License Management in sandboxes unless you ask specifically. Therefore, you have an unlimited Site License in all sandboxes. This means that you do not have to assign a Salesforce user a license for Composer.
- If you would like this enabled, please keep in mind you need to enable all the users you would like to have access. You can submit a case on the support page and request LMA be turned on.

How do I assign users licenses?

- Please click here for instructions: [Assigning User Licenses for Conga Sign](#).

I just renewed, but I am still seeing an expiration warning.

- If you have signed your renewal order form, and your expiration warning is less than 2 weeks away, please submit a case.

Does the Conga use License Keys?

- Conga does not use license keys. All new installs for Conga Sign, Contracts, Conga Grid will be assigned with 5 Trial licenses for 15 days.
- All products can be downloaded via the Salesforce AppExchange. Exceptions to this are Conga Collaborate and Conga Contracts (off the platform), and Contracts for Salesforce.
- Here is a link to the [Conga Installation Links](#).
- Conga Collaborate, Conga Contracts, and Contracts for Salesforce will be installed by your Conga Professional Services team member as part of your Professional Services engagement. (If this does not apply to you, then please contact your Conga account representative.

Assigning User Licenses for Conga Sign

When installing Conga Sign from the AppExchange, the default license count upon installation is 10, one of which is assigned to the administrator who installed the Conga Sign. Conga Provisioning then updates your license count to match the amount that you have purchased within 1-3 business days.

 It is recommended to use Salesforce Classic to assign licenses.

To assign the remaining licenses for Conga Sign


1. Navigate to the **Installed Packages** in Salesforce **Setup**.
2. Click **Manage Licenses** next to **Conga Composer**.
3. Click **Add Users**.
4. Select the box to the left of the names of the users who need access to the Conga Sign.

5. Click **Add** to add the selected users.
6. Click **Add Users** to save changes.

Remember to email onboarding@conga.com when you have installed (if you have not done so already) to ensure the fastest provisioning of any additional licenses.

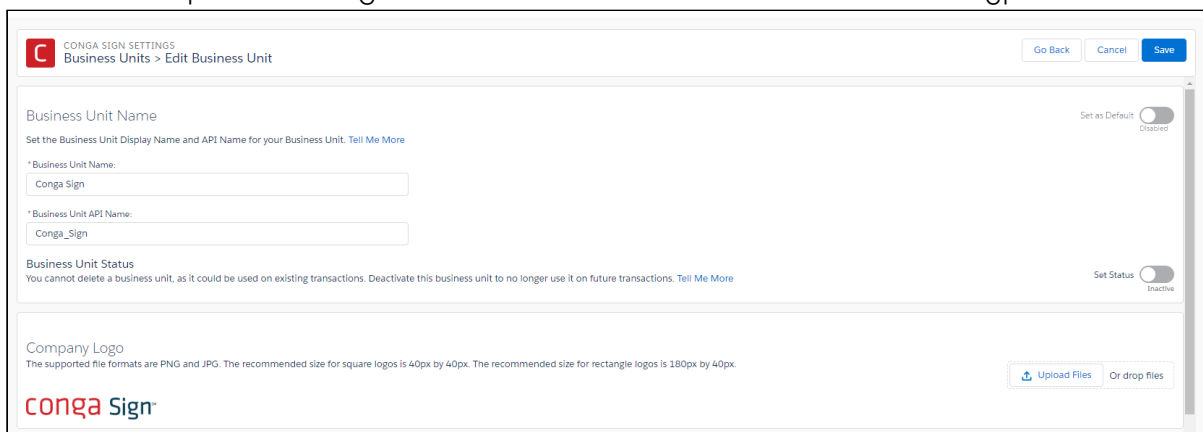
Customizing Your Logo

- Ensure the image is a supported file type: PNG or JPG
- The image must be 40 x 40 pixels for a square image and 180 x 40 pixels for a rectangle image.

 Logos are uploaded to individual Business Units. If multiple Business Units are enabled for an org, users can select from multiple Business Units in the Create Transaction user interface. For more information on Business Units, see [Business Units](#).

To add a logo

1. Navigate to **Conga Sign Setup** and click **Business Units**.
2. Click **Edit Business Unit** next to a specific Business Unit.
3. Click **Upload Files**.
4. Locate and upload the logo file. Ensure that the file is the correct file type and size.



CONGA SIGN SETTINGS
Business Units > Edit Business Unit

Go Back Cancel Save

Business Unit Name
Set the Business Unit Display Name and API Name for your Business Unit. [Tell Me More](#)

* Business Unit Name:
Conga Sign

* Business Unit API Name:
Conga_Sign

Business Unit Status
You cannot delete a business unit, as it could be used on existing transactions. Deactivate this business unit to no longer use it on future transactions. [Tell Me More](#)

Set as Default ☐ Disabled

Set Status ☐ Inactive


Company Logo
The supported file formats are PNG and JPG. The recommended size for square logos is 40px by 40px. The recommended size for rectangle logos is 180px by 40px.

Upload Files Or drop files

conga Sign

Customizing Emails


Customize and brand your company email templates to provide a professional business experience to the recipients. Additionally, you can customize the body text of emails sent to the recipient when you send a transaction.

 Email customization is done for individual Business Units. If multiple Business Units are enabled for an org, users can select from multiple Business Units in the Create Transaction user interface. For more information on Business Units, see [Business Units](#).


If you need a document to be signed, the email the signer receives can have a specified text. Additional details are added to the Sent Email body so your users can more easily identify and confirm information associated with the transaction. The email body is defined. The document name is bolded and includes a list of recipients to whom the transaction email is sent.

To customize emails

1. Navigate to **Conga Sign Setup** and click **Business Units**.
2. Click **Edit Business Unit** next to a specific Business Unit.
3. Enter values in the following fields to customize your Conga Sign emails:
 - a. Email Display Name
 - b. Subject
 - c. Message
 - d. Company
 - e. Address
 - f. City, State, and Zip/Postal Code
 - g. Country
 - h. Phone Number
 - i. Website

 It is not required to enter values for all email configuration and contact fields in a Business Unit. For more information on customizing email fields for a Business Unit, see [Configuring Business Units](#).

4. Click **Save**.

 Both the original Signer and Requester receive Re-assigned Requester email notifications. Error messages were updated to explain to users who attempt to re-assign Signer after previous successful re-assignment. The error message states: This document has already been re-assigned to a new Signer. Each Signer can only re-assign to a new Signer once.

Conga Sign Parameters

The native Send with Conga Sign button's behavior and functionality is customizable with Conga Sign parameters.

There is a separate set of integration parameters when using Conga Sign with Composer. For more information, see [Conga Sign Integration Parameters](#). Also, note that parameters are case-sensitive.

Parameters

. authRequired

- Allows Administrators to control whether their Users can disable SMS Authentication in the Create Transaction UI.
- When enabled, Senders are unable to change the Authentication Type for a signer to "Standard", the "Standard" option will not be available in the Auth dropdown. Senders can still modify a signer's phone number by clicking the authentication dropdown and selecting the SMS option.
- The following prerequisite parameters are required for this parameter:
&recipient[1-10]
&recipientrole[1-10]
&recipientlanguage[1-10]
&recipientauth[1-10]=SMS
&recipientmobile[1-10]

Example

```
authRequired=true
```

. businessUnit

- A string that defines a specific Business Unit to select for a Conga Sign Transaction.
- The parameter value must be the API name of one active Business Unit. You can use this parameter once in a transaction to designate a singular Business Unit.

Example

```
businessUnit=Global_Tech_Inc
```

. documentId

- A ContentDocument ID that automatically selects a document on the screen.
- The Id referenced here can be a document that is attached to the parent object, or attached to a different object.
- Intended for individual document Ids only, not to be used in conjunction with documentIds. Buttons that contain both documentId and documentIds will result in error.

Example

```
&documentId=0696A000005bXNn
```

. documentIds

- A selection of document Ids that automatically selects multiple documents on the screen.
- The Ids referenced here can be documents that are attached to the parent object, or attached to different objects.
- This parameter **must** be used independently of the documentId parameter. Buttons that contain both documentId and documentIds will result in error.
- Defined Ids must be comma-delimited.
- Documents will appear in the Create Transaction screen in the order of the Ids applied to the parameter from left to right.

Example

```
&documentIds=0696A000005bXNn,0531A001005bDcS
```

. emailMessage

- A string with a limit of 50000 characters.

- Add parameter email message (string, max length 50000) that defaults the message field in the email section of the Conga Sign user interface.

Example

```
&emailMessage=Please+sign+this+document.
```

. expireDays

- A number that determines the number of days before the Transaction expires.
- The number must be a positive integer.

Example

```
&expireDays=7
```

. expireOn

- A string for a datetime in UTC format.
- It can be used with a merge field if wrapped in a TEXT() function.
- Add parameter expires (ISO8601 date) that default the transaction expires field in the notification section.

Example

```
&expireOn={!TEXT(Object.DateTime)}
```

Example

```
&expireOn=2019-07-10T14:00:00.000Z
```

. expireReminder

- A string for a datetime in UTC format.
- Add parameter expirationReminder (ISO8601 date) that defaults the expiration reminder in the notification section.

Example

```
&expireReminder=2019-07-10T14:30:00.000Z
```

. expireReminderDays

- A number to determine the number of days before the first reminder email is sent.
- The number must be a positive integer.

Example

```
&expireReminderDays=2
```

. emailSubject

- A string with a limit of 1000 characters.
- Add parameter emailSubject (string, max length 1000) that defaults the subject field in the email section.

Example

```
&emailSubject=Document+for+Signature
```

. reminderDays

- A number which is a positive integer.
- Add parameter reminder (integer) that defaults the request reminder field in the notification section.

Example

```
&reminderDays=3
```


. id

- A string of a 15 digit or 18 digit Salesforce record id.
- Determines the master object id for a Conga Sign transaction.

Example

```
&id=006f400000RTyAN
```

. reassignAllowed

- Allows the recipient of a Conga Sign transaction to reassign the signer of the transaction.

Example

```
&reassignAllowed=true
```

. recipient[1-10]

- A string for a Contact, Lead, or User Salesforce Id.
- Recipients 1 through 10 to assign to the transaction. If any recipients are skipped the resulting list is squashed - i.e. if only recipients 3 and 5 are set they show up as 1 and 2 in the UI.

Example

```
&recipient1={!Opportunity.ContactId__c}&recipient2=0036A00000033p4
```

. recipient[1-10]auth

- A string for a Contact, Lead or User Salesforce Id; SMS is the only available value
- Assigns SMS authorization to a recipient.
- If the recipient[1-10]mobile parameter is not designated for a recipient, Conga Sign automatically uses the mobile number on the recipient's Salesforce record (Contact, User, or Lead) to send the SMS authorization to. The Mailing Country

field on the record must be configured with 2 character ISO 3166-1 alpha-2 country code (example: US Mobile number: 3038675309, country code: US).

Example

```
&recipient1={!Opportunity.ContactId__c}&recipient1Auth=SMSNote: SMS
Authentication is currently available for Early Adopters only. To become
an Early Adopter, contact your designated Customer Success Manager
or Conga Support.
```

. recipient[1-10]curl

- A string to determine a website that recipients can chose to be taken to after successfully completing signing.
- With this parameter, recipients can click Continue after completing signing and are taken to the URL defined with the recipient[1-10]curl parameter.
- If the recipient is a standard SIGNER and clicks Continue after completing signing, they are taken to the URL in the same browser tab. If the recipient is an IN_PERSON_SIGNER and clicks Continue after completing signing, they are taken to the URL in a new browser tab.
- The format of the URL must start with either http:// or https://. The www. portion of a URL is optional.

Example

```
recipient[1-10]curl parameter values:
http://www.conga.com
http://conga.com
https://www.conga.com
https://conga.com
```

. recipient[1-10]email

- A string to determine the email address necessary to create and add a new Conga Sign Recipient to a transaction.

- This parameter also requires the **recipient[1-10]first** and **recipient[1-10]last** parameters, or the **recipient[1-10]name** parameter, to successfully create and add new Conga Sign Recipients.

✓ Use the **recipient[1-10]name** parameter to define a new Conga Sign Recipient's full name, as opposed to using **recipient[1-10]first** and **recipient[1-10]last** to define first and last name separately. You cannot use **recipient[1-10]name** with **recipient[1-10]first** and **recipient[1-10]last** to define a new Conga Sign Recipient's name.

- The parameter value must have the following email address format:
<username>@<domain>.com
- The **recipient[1-10]email**, **recipient[1-10]first**, **recipient[1-10]last**, and **recipient[1-10]name** parameters are used to specifically create a new Conga Sign Recipient that does not already exist as a Contact, User, or Lead record in Salesforce.
- Creating and adding a new Conga Sign Recipient to a transaction does not create a Contact, Lead, or User record in Salesforce. It creates a Conga Sign Recipient (custom object) record specific to that transaction only. New Conga Sign Recipients created with Conga Sign parameters cannot be added or re-used in other transactions.

✓ Use the **recipient[1-10]** parameter to add existing Contact, Leads, and Users as Conga Sign Recipients, instead of creating new Conga Sign Recipients.

• recipient[1-10]first

- A string to determine the first name necessary to create and add a new Conga Sign Recipient to a transaction.
- This parameter also requires the **recipient[1-10]last** and **recipient[1-10]email** parameters to successfully create and add a new Conga Sign Recipient.
- The **recipient[1-10]email**, **recipient[1-10]first**, **recipient[1-10]last**, and **recipient[1-10]name** parameters are used to specifically create a new Conga Sign Recipient that does not already exist as a Contact, User, or Lead record in Salesforce.

✓ Use the **recipient[1-10]** parameter to add existing Contact, Leads, and Users as Conga Sign Recipients, instead of creating new Conga Sign Recipients.

- Creating and adding a new Conga Sign Recipient to a transaction does not create a Contact, Lead, or User record in Salesforce. It creates a Conga Sign Recipient (custom object) record specific to that transaction only. New Conga Sign Recipients created with Conga Sign parameters cannot be added or re-used in other transactions.

. recipient[1-10]language

- A string for a Contact, Lead or User Salesforce Id; either de, fr, en-US, es, pt-PT, pt-BR, it, el, nl-NL, hr, pl, uk, ro, ka, ru or sh
- Assigns a language to a specific recipient.
- The following languages are supported:
 - German: de
 - French: fr
 - English (US): en-US
 - Portuguese (Portugal): pt-PT
 - Portuguese (Brazil): pt-BR
 - Italian: it
 - Greek: el
 - Dutch: nl-NL
 - Croatian - hr
 - Polish - pl
 - Ukrainian - uk
 - Romanian - ro
 - Georgian - ka
 - Russian - ru
 - Serbian - sh

Example

```
&recipient1={!Opportunity.ContactId__c}&recipient1language=fr
```

. recipient[1-10]last

- A string to determine the last name necessary to create and add a new Conga Sign Recipient to a transaction.

- This parameter also requires the **recipient[1-10]first** parameter and **recipient[1-10]email** parameters to successfully create and add a new Conga Sign Recipient.
- The **recipient[1-10]email**, **recipient[1-10]first**, **recipient[1-10]last**, and **recipient[1-10]name** parameters are used to specifically create a new Conga Sign Recipient that does not already exist as a Contact, User, or Lead record in Salesforce.

✔ Use the **recipient[1-10]** parameter to add existing Contact, Leads, and Users as Conga Sign Recipients, instead of creating new Conga Sign Recipients.

- Creating and adding a new Conga Sign Recipient to a transaction does not create a Contact, Lead, or User record in Salesforce. It creates a Conga Sign Recipient (custom object) record specific to that transaction only. New Conga Sign Recipients created with Conga Sign parameters cannot be added or re-used in other transactions.

• **recipient[1-10]name**

- A string to determine the full name (both first and last name) necessary to create and add a new Conga Sign Recipient to a transaction.

⚠ Use a space or + symbol to separate the first and last name for this parameter value.

- This parameter requires the **recipient[1-10]email** parameter to successfully create and add a new Conga Sign Recipient.

✔ Use the **recipient[1-10]name** parameter to define a new Conga Sign Recipient's full name, as opposed to using **recipient[1-10]first** and **recipient[1-10]last** to define first and last name separately. You cannot use **recipient[1-10]name** with **recipient[1-10]first** and **recipient[1-10]last** to define a new Conga Sign Recipient's name.

- The **recipient[1-10]email**, **recipient[1-10]name**, **recipient[1-10]first**, and **recipient[1-10]last** parameters are used to specifically create a new Conga Sign Recipient that does not already exist as a Contact, User, or Lead record in Salesforce.

- ✓ Use the recipient[1-10] parameter to add existing Contact, Leads, and Users as Conga Sign Recipients, instead of creating new Conga Sign Recipients.

- Creating and adding a new Conga Sign Recipient to a transaction does not create a Contact, Lead, or User record in Salesforce. It creates a Conga Sign Recipient (custom object) record specific to that transaction only. New Conga Sign Recipients created with Conga Sign parameters cannot be added or re-used in other transactions.

. recipient[1-10]role

- A string for a Contact, Lead or User Salesforce Id; either IN_PERSON_SIGNER, SIGNER, or CC
- Assigns a role to a specific recipient.

Example

```
&recipient1={!Opportunity.ContactId__c}&recipient1role=SIGNER
```

. routingType

- A string for routing type; either PARALLEL or SERIAL
- If set to PARALLEL, all recipients are routed at the same level (default behavior). If set to SERIAL, recipients are routed based upon their defined routing order. Example: CSRecipient1, CSRecipient2, CSRecipient3.
- If you want CC role(s) to receive emails at the beginning and end of the transaction, place them at the beginning of the recipients list. If you want CC role(s) to receive an email only after a document has been signed, place them at the end of the recipients list.

Example

```
&routingType=SERIAL
```

• recipient[1-10]mobile

- A string for a Contact, Lead, or User Salesforce Id that assigns a mobile number to a specific recipient for SMS Authentication.
- The string must be all numeric and include the country code (example: US mobile number: 13038675309).
- If the recipient[1-10]mobile parameter is not designated for a recipient, Conga Sign automatically uses the mobile number on the recipient's Salesforce record (Contact, User, or Lead) to send the SMS authorization to. There are two acceptable format options for phone numbers in the Mobile field.
 - i. Full E.164 format, which includes the '+' and the country code (ex: US mobile number: +13036669999)
 - ii. The mobile number does not include the '+' and country code, but the Mailing Country in the Address Information section for the recipient is configured with the 2 character ISO 3166-1 alpha-2 country code. (ex: US mobile number: 3035171753, country code: US).
- Requires recipient to have the recipient[1-10]auth=SMS parameter.

Example

```
&recipient1={!Opportunity.ContactId__c}
&recipient1Auth=SMS&recipient1Mobile=13038675309
```

• reminderFrequencyDays

- A number to determine the frequency with which Conga Sign sends reminder emails to recipients.
- The value must be numeric and can only be between the values of 1-120.

Example

```
&reminderFrequencyDays=4
```

• senderReturnUrl

- A string to determine a page that the sender is returned to after successfully sending a transaction.

- Add senderReturnUrl = http/https site or relative url.
- By using the below example, you are redirected to the Conga Support website after successfully sending a transaction.

Example

```
{!URLFOR($Site.Prefix)}/apex/APXT_CongaSign__apxt_sendForSignature?id={!Opportunity.Id}&senderReturnUrl=https://support.conga.com
```

- By using the below example, you are redirected to the Account record associated with the Opportunity record after successfully sending a transaction.

Example

```
{!URLFOR($Site.Prefix)}/apex/APXT_CongaSign__apxt_sendForSignature?id={!Opportunity.Id}&senderReturnUrl={!Opportunity.AccountId}
```

• sendonbehalfname

- A string to identify the Name of the intended recipient you are sending a transaction on behalf of.

Example

```
&sendonbehalfname=First+Last
```

• sendonbehalfemail

- A string to identify the Email of the intended recipient you are sending a transaction on behalf of.

Example

```
&sendonbehalfemail=fLast@conga.com
```


· syncDocsToParent

- A string to integrate with Conga CLM, allows you to determine if you would like the documents in a transaction to be synced to the record the button was launched from.
- When syncDocsToParent is set to false, the original and final documents will only be synced to the transaction record in Salesforce.
- Signer attachments will only sync back to the transaction record in Salesforce.

Example

&syncDocsToParent=**false**

· syncFinalDocAsNewContentDoc

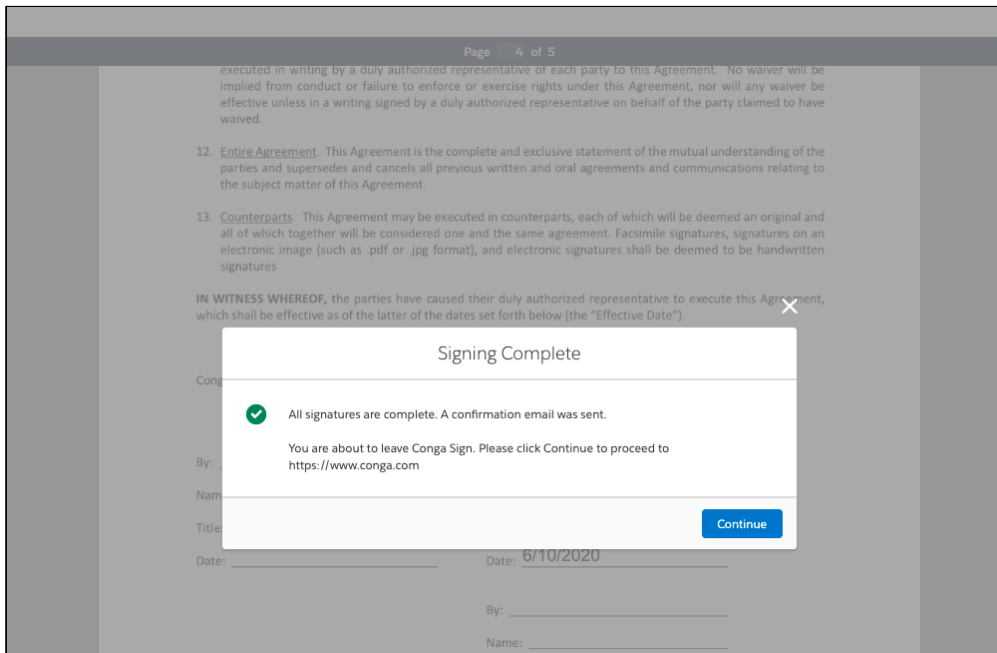
- A string to integrate with Conga CLM, allows you to determine if you would like the final document to come as a separate content document.
- The Final Documents Sync Status field will be populated with a note of SUCCESS once the final documents are written back into Salesforce if the parameter is set to true.

Example

&syncFinalDocAsNewContentDoc=**true**

Conga Sign Redirect URL

The Conga Sign Redirect URL allows Conga Sign users to define a URL that signers can click and access after successfully completing their portion of a Conga Sign document. Upon successfully signing a Conga Sign document, signers can simply click Continue and Conga Sign redirects the signer to the defined URL.



Defining a Conga Sign Redirect URL

A Conga Sign Redirect URL is defined with a specific parameter and parameter value for each signer.

In a native Conga Sign button, the Conga Sign Redirect URL is defined with the `recipient[1-10]curl` Conga Sign parameter. In a Conga Sign-integrated Composer solution, the Conga Sign Redirect URL is defined with the `CScurl[1-10]` Composer parameter.

i A Conga Sign Redirect URL must be defined with the aforementioned parameters. There is no option in the Conga Sign user interface to add or designate a Conga Sign Redirect URL.

Both a Conga Sign button and Conga Sign-integrated Composer solution can have up to 10 Conga Sign Redirect URLs specified for signers. Each signer assigned a Conga Sign Redirect URL must have the role of `SIGNER` or `IN_PERSON_SIGNER`.

The format of a URL specified as a Conga Sign Redirect URL must start with either `http://` or `https://`. The `www.` portion of a URL is optional.

Example of acceptable Conga Sign Redirect URL parameter values:


- <http://www.conga.com>
- <http://conga.com>
- <https://www.conga.com>
- <https://conga.com>

Automatically Upload a Conga Sign Document to a Third Party Repository


Automatically upload a completed Conga Sign document to a third-party repository, such as Box and SharePoint, using a folder-specific upload email address. Users can also simply add the upload email address Contact as the final signer with a CC role manually when creating and sending a Conga Sign Transaction.

To automatically upload a completed Conga Sign document to an online repository

1. Create a **Conga Sign-integrated Composer** button.
2. Create a **Contact** record in Salesforce and use a repository folder's upload email address as the standard Email field's value. Example:

Contact Detail		Edit	Delete	Clone
Contact Owner	 Walker Llewellyn [Change]	Phone		
Name	Upload Email Address	Home Phone		
Talent Name		Mobile		
Title		Other Phone		
Department		Fax		
Birthdate		Email <u>completw814jep7zhqc4i0s@u.box.com</u>		

3. Add the **&CSRoutingType=Serial** parameter to the **Composer** button. Serial routing is required for this solution.
4. Add a new recipient to the **Composer** button using the **CSRecipient** parameter. Use the record ID of the new upload email address Contact record as the parameter value.

 The upload email address Contact must be designated as the last Conga Sign recipient.

5. Add the **CSRole** parameter to designate a CC signer role to the upload email address Contact. Example:

```

Button or Link URL    /apex/APXTConga4__Conga_Composer
                     ?serverUrl={!API.Partner_Server_URL_370}
                     &id={!Opportunity.Id}
                     &TemplateID=a0Cf400000HF27S
                     &Qvar0Id=a04f4000009gDSJ
                     &CSVisible=1
                     &CSRoutingType=SERIAL
                     &CSEExpiration={!Opportunity.csexpiration__c}
                     &CSEExpirationReminder={!Opportunity.csexpirationreminder__c}
                     &CSRoutingType=Serial
                     &CSRecipient1={!Opportunity.OwnerId}
                     &CSRecipient2={Qvar0}
                     &CSRecipient3=003f400000tJ26Y
                     &CSRole3=CC

```

Following the steps above, the upload email address automatically populates as CC signer role when launching a Conga Sign-integrated Composer button.

Add Recipients
You must select an existing Contact, Lead, or User. Email addresses are not allowed.

Search by Name or Account

☒ Signing Order * Drag and drop recipients to define signing order.

NAME	EMAIL	COMPANY	TITLE	ROLE
Walker Llewellyn	walkerllewellyn@gmail.com	Innitech Music Solutions		Signer
John Doe	walker.llewellyn@getconga.com	Burlington Textiles Corp of America		Signer
Upload Email Address	complet.w814jep7zhgc4j0s@u.box.com			CC

The document is then uploaded to the third-party repository once the final signer completes their assigned portion of the Conga document.

Using Conga Sign in Salesforce Communities


Use a native Conga Sign button in a Salesforce Community to allow Community users to send documents for signature. Additionally, administrators can setup a Conga Sign-integrated Composer button or a Conga Sign-integrated Trigger solution in a Salesforce Community to generate and send documents for e-signature.

i Conga Trigger calls Conga Sign through a Salesforce Workflow Rule and bypasses the Conga Sign user interface. Conga Trigger is a paid add-on and is not included with the Composer.

Prerequisites

- [Setup a Salesforce Community.](#)
- [Install Conga Sign.](#)

- If using Conga Composer, [Install Conga Composer](#).
- If using Conga Trigger, [Install Conga Composer](#) and [Install Conga Trigger](#).

 Conga Sign is not compatible with the Guest User profile in Salesforce Communities.

To create and add a Conga Sign Button to a Salesforce Community

1. Create a Conga Sign button or use the native **Send with Conga Sign** button.
2. Test that the Conga Sign-integrated Composer button successfully sends a Conga Sign Transaction.
3. Place the Conga Sign button on the desired object's Lightning Page or Page Layout so it appears in the Salesforce Community.


To create and add a Conga Sign-integrated Composer Button to a Salesforce Community

1. Create a **Conga Sign-integrated Composer** button.
 - Use Conga Sign Integration Parameters to integrate Conga Sign with a **Composer** button. The button must have the **CSVisible**, **CSRecipient**, and **TemplateID** parameters at a minimum.
 - (Optional) Pre-tag the Conga Template document with [Conga Sign tagging syntax](#).
2. Test that the **Conga Sign-integrated Composer** button successfully sends a Conga Sign Transaction.
3. Place the **Conga Sign** button on the desired object's Lightning Page or Page Layout so it appears in the Salesforce Community.

To create and add a Conga Sign-integrated Trigger solution to a Salesforce Community


1. Create a **Conga Sign-integrated Composer** button.

- Use Conga Sign Integration Parameters to integrate Conga Sign with a **Composer** button. The button must have the **CSVisible**, **CSRecipient**, and **TemplateID** parameters at a minimum.
 - Pre-tag the Conga Template document with [Conga Sign tagging syntax](#).
 - Test that the **Conga Sign-integrated Composer** button successfully sends a Conga Sign Transaction.
2. Convert the **Composer** button into a Conga Trigger formula field using the Conga Trigger Formula Builder.
 - The Conga Trigger formula field must have **Qmode=CongaSign**.
 3. Create a Salesforce Workflow Rule for the Conga Trigger Solution. The Workflow Rule determines the criteria for when the document is merged and sent with Conga Sign.
 4. Test the Conga Trigger solution to confirm it successfully creates a Conga Sign Transaction.

 Conga Trigger bypasses the user interface and automatically sends the transaction.

Using Conga Sign with Conga CLM

Conga Sign integrates with Conga CLM, allowing you to generate and send documents for eSignature via Conga CLM. Agreements will now have an option for *Send with Conga Sign* under the Agreement Lifecycle Actions.

 For more information regarding setting up Conga CLM, refer to [Setting up Conga CLM](#) documentation.

CLM-Conga Sign Custom Settings

The Conga Sign/Conga CLM integration requires the configuration of Custom Settings, found in the [Post-Installation](#) steps of Conga CLM.

Customizing the CLM-Conga Sign Settings

System Properties describes the configuration of the working environment and allows you to customize your system settings. Use custom settings to create and manage custom data at the organization, profile, and user levels. Custom settings data is stored in the application cache that you can access efficiently, without the cost of repeated queries.

To customize the CLM-Conga Sign Settings

1. Go to **Setup > App Setup > Develop > Custom Settings**.
2. Click **Manage** for **CLM-Conga Sign Settings**.
3. Click **New** in the Default Organization Level Value section.
4. Select Profile or User from the Location dropdown.
5. According to your selection in the previous step, enter a profile name or user name and click the lookup icon (🔍) to open the Lookup window.
6. Select a relevant profile name or user name.
7. Enable or disable the following properties for the selected profile or user and click **Save**.

Custom Setting	Help Text	Default Value
Show Related Documents	Allow selection of documents from related, sibling, parent and child agreements	FALSE
Filter By Primary Agreement Status	Filter documents from related, sibling, parent and child agreements based on primary agreement status. This setting applies when Show Related Documents is enabled.	FALSE

Using Conga Sign with Conga Composer

Conga Sign integrates with Conga Composer, allowing you to generate and send documents for eSignature using all the features of Conga Composer, the leading document generation app for Salesforce. After purchasing Conga Composer, use the [Conga Sign Integration Parameters](#) to set up the integration.

Additionally, Conga Sign and Conga Composer can be automated using Conga Batch and Conga Trigger. For more information, see the [QMode](#) parameters.

⚠ If you want to send multiple documents for eSignature in a single transaction using Conga Composer, you must provide the [CSMultiDocs=1](#) and [zipFiles=0](#) parameters. For more information on sending multiple documents for eSignature, refer to [Sending Document for eSignature](#) in the *Conga Sign User Guide*.

Using Conga Sign with Conga Contracts for Salesforce

Conga Sign integrates with Contracts for Salesforce, allowing you to use Conga Sign for your eSignature transactions. The integration simplifies the process flow for customers who use both products. The Conga Sign button is available on the Send for Negotiation user interface in Contracts for Salesforce.

Setting up a Process Builder for Conga Sign Transaction

You can set up a process builder to show a Conga Sign Transaction record on the parent account. The below steps are explained for an Opportunity account. You can build this type of process for any parent object that is configured for Conga Sign.

To set up a Process Builder for Conga Sign Transaction

1. Go to Salesforce **Setup**.
2. Search for Process Builder.
3. On the My Processes screen, click **New**.
4. Under New Process, enter the following:
 - a. **Process Name**. For example, "CS Transaction on Account".
 - b. **API Name**
 - c. For **The process starts when**, select **A record changes** from the dropdown menu.
5. Click **Save**.
6. Click **+Add Object**.
 - a. For **Object**, select **Conga Sign Transaction**.
 - b. For **Start the process**, select **when a record is created or edited**.
7. Click **Save**.
8. Click **+Add Criteria**.
 - a. Enter **Criteria Name**. For example, "Update CS txn from Oppty".
 - b. For **Criteria for Executing Actions**, select **Conditions are met**.
 - c. Under **Set Conditions**, set the following fields:

- Click on Find a Field under **Field**, and in the dropdown list select **[APXT_CongaSign__Transaction__c].APXT_CongaSign__Status__c**.
 - Select **Operator** as **Equals**.
 - Select **Type** as **Picklist**.
 - Select **Value** as **Complete**.
- d. Click **Add Row** and set the following fields:
- Click on Find a Field under **Field**, and in the dropdown list select **[APXT_CongaSign__Transaction__c].Parent_006__c**.
 - Select **Operator** as **Is null**.
 - Select **Type** as **Boolean**.
 - Select **Value** as **False**.
- e. For **Conditions**, select **All of the conditions are met (AND)**.
9. Click **Save**.
10. Click **+Add Action** in Immediate Actions.
- a. For **Action Type**, select **Update Records** from the dropdown list.
 - b. Enter **Action Name**. For example, "Update CS txn".
 - c. For **Record Type**, select **Select the record that started your process**.
 - d. For **Criteria for Updating Records**, select **No criteria - just update the records!**.
 - e. Under **Set new field values for the records you update**, set the following fields:
 - For **Fields**, enter **Account**.
 - For **Type**, select **Field Reference**.
 - Click on Find a Field under **Value**, and in the dropdown list select **Opportunity > AccountID**.
11. Click **Save**.

Conga Sign for Users

With the Conga Sign User Guide, you can find out how Conga Sign works and how to manage your organization's and your customers' eSignature workflows.

Topic	Description
What's Covered	This guide walks the Conga Sign user through a set-up of the entire e-Signature process. It covers step-by-step instructions, and use cases for the features provided by the Conga Sign.
Primary Audience	Individuals who are working on any aspect of contracts: contract managers, contract creators, contract negotiators, and contract reviewers
IT Environment	Refer to the latest Release Notes for information on System Requirements and Supported Platforms.
Updates	For a comprehensive list of updates to this guide for each release, refer to What's New in Conga Sign Documentation .
Other Resources	<ul style="list-style-type: none"> • <i>Conga Sign for Administrators</i>: Refer to this guide to set-up the entire Conga Sign process.

Select one of the following topics for more information:

- [Conga Sign Product Guide](#)
- [Conga Sign Product Limits](#)
- [Available Features for Conga Sign Integrations](#)
- [List of Conga Sign Signature Tags](#)
- [Using Signature and Custom Tags](#)
- [Creating a Document to Sign](#)
- [Creating New Recipients in Create Transaction UI](#)
- [Sending Documents for eSignature](#)
- [Signing a Document](#)
- [In-Person Signing](#)
- [Canceling a Conga Sign Transaction](#)
- [Conga Sign Link Expiration](#)
- [Conga Sign Recipient Object](#)
- [Reassigning Signers as a Recipient](#)
- [Reassigning Signers from Salesforce](#)

- [Audit Trail](#)
- [SMS Authentication](#)
- [Supported Languages in Conga Sign](#)
- [Troubleshooting Topics](#)

Conga Sign Product Guide

Learn Conga Sign

- [About Conga Sign for Users](#)

Install Conga Sign

- [Install Conga Sign](#)
- [What's in the Package?](#)

Configure Conga Sign

- [Configure Conga Sign](#)
- [Add Sign Button and Transactions Related List to the Page Layout](#)
- [Permissions Needed to Use Conga Sign](#)

Use Conga Sign

- [Sending Documents for eSignature](#)
- [Creating a Document to Sign](#)

Use Signature Tags with PDF Forms

- [How to Sign a Document](#)
- [Audit Trail](#)

Conga Sign Product Limits

The table below summarizes current product limits for Conga Sign.

Conga Sign Feature	Documented Limit
Conga Sign Tags in a Conga Sign document	1,000 tags
Amount of Recipients per Conga Sign Transaction	10 recipients
File size of a Conga Sign document	50 MB
Page size of a Conga Sign document	500 pages
File size of a Conga Sign via Conga Contracts integration	10 MB
Conga Sign Email Subject	1,000 characters
Conga Sign Email Body	50,000 characters
Salesforce objects a user can configure in Automatic Configuration at one time	10 objects
Salesforce profiles a user can configure in Automatic Configuration at one time	10 profiles
Amount of signers assignable to a Write back tag	1 signer
Conga Sign document files	Word and PDF
Signing user interface timeout	1 hour
Conga Sign invitation email expiration	5 days

Available Features for Conga Sign Integrations

Conga Sign integrates with Conga Collaborate and Conga Contracts in addition to the standard Salesforce integration. The feature comparison matrix below depicts available Conga Sign features for each product integration.

FEATURE	Conga Sign for Salesforce	Conga Sign for Conga Collaborate	Conga Sign for Conga Contracts
Send for Signature functionality	Yes	Yes	Yes
Basic Tags	Yes	Yes	Yes
Custom Tags	Yes	Yes	Yes
Setting tags as optional	Yes	No	No
Parallel Routing Type	Yes	No	Yes
Serial Routing Type	Yes	Yes	Yes
Custom Email Subject	Yes	Yes	No
Custom Email Message	Yes	Yes	Yes
Audit Trail	Yes	Yes	Yes
Reassign signers	Yes	No	Yes
Tagging user interface	Yes	No	Yes
Write back tags	Yes	No	No
Branding in the Signing user interface	Yes	No	No
Transaction expirations	Yes	No	No
Transaction expiration reminders	Yes	No	No
SMS Authentication	Yes	No	No
Transaction Reminders	Yes	No	Yes
Attachment Tags	Yes	No	No

FEATURE	Conga Sign for Salesforce	Conga Sign for Conga Collaborate	Conga Sign for Conga Contracts
Business Units	Yes	No	No
Multiple Documents	Yes	No	Yes

The Conga Sign Audit Trail is accessible in Conga Contracts by clicking a completed signature or initial tag in the final Conga Sign document. For more information, see *Conga Sign Integration with Conga Contracts* topic in the *Conga Contracts Administrator Guide*.

List of Conga Sign Signature Tags

The following eSignature tags are available to use in a document with Conga Sign. You can add tags by dragging and dropping a tag in the Preview and Tag screen of Conga Sign. Conga Sign tags can be specified as required or optional for signers to sign. If a tag is set to required, a signer must complete the tag in a Conga Sign transaction. In the Basic Tag Configuration section of the Conga Sign Setup tab, you can set tags as Required or Optional.

When using pre-tag syntax in a Conga Sign document, Conga Sign places the tag at the bottom left of the first line of syntax text. If tagging syntax wraps, Conga Sign aligns the tag to the left of the first line. It is suggested to use small font size in the Conga Sign document when a tag is lengthy (write back tags may be lengthy when using a default value and setting a text size).

You can add custom tags to address business needs in addition to the standard tags below. To learn more about custom tags see: [Using signature and custom tags](#).

Purpose	Conga Sign Tag	Required or Optional
Signature of Recipient 1	\signature\	Required
Initial of Recipient 1	\initial\	Required
Stamp of Recipient 1	\stamp\	Required
Name of Recipient 1	\fullname\	Required
Title of Recipient 1	\title\	Required or Optional

Purpose	Conga Sign Tag	Required or Optional
Email of Recipient 1	<code>\email\</code>	Required
Date of Recipient 1	<code>\date\</code>	Required
Company of Recipient 1	<code>\company\</code>	Required or Optional
Checkbox of Recipient 1	<code>\checkbox\</code>	Required or Optional
Attachment of Recipient 1	<code>\attachment1 {"label": "Driver's License"}\</code>	Required or Optional
Customtext	<code>\customtext1 {"label": "Address"}\</code>	Required or Optional

Tags can be incremented with the number 1-10. Each number represents the corresponding signer. For example, `\signature2\` is the tag for the second recipient. You can give custom tags attributes and properties. For example, you can add a label to the custom text tag: `\customtext1 {"label": "Add a label here"}\`

The checkbox tag features a transparent section to the right of the checkbox that can be placed over text without blocking out content in a Conga Sign document. This allows users to closely place the actionable checkbox tag next to the text in a Conga Sign document. For more information on custom tags, see [Using signature and custom tags](#).

- ✓ If you do not want Conga Sign tags to appear in your document, change the font color to white.

The table below defines the available tag syntax properties.

Purpose	Property	Values	Example
Add a label to a custom text tag or write back tag	"label"	Any string	<code>\customtext1 {"label": "Address"}\</code>
Add a default value for custom text or write back tag	"defaultValue"	Any string	<code>\customtext1 {"label": "Association", "defaultValue": "Conga Sign"}\</code>

Purpose	Property	Values	Example
Set a maximum length of a value for custom text or write back tags	"maxLength"	Any number from 1 to 255	\customtext1 {"label": "Phone Number", "maxLength": "10"}\
Associate a write back tag with the correct Salesforce field	"apiName"	Any string	\wb1 {"label": "Billing City", "apiName": "Account__r.BillingCity"}\
Make a tag required	"required"	"true"	\customtext1 {"label": "Phone Number", "required": "true"}\
Custom format for a Sign Date tag	"format"	"MM/DD/YYYY", "DD/MM/YYYY", "DD-MMM-YYYY", or "MMM DD, YYYY"	\date1 {"format": "DD-MMM-YYYY"}\
Specify the text size of a tag	"textSize"	"x-small", "small", "medium", "large", or "x-large"	\customtext1 {"textsize": "x-large"}\
Specify the size of a Signature or Initials tag	"size"	"small", "medium", "large", or "x-large"	\signature1 {"size": "large"}\

List of recognized Adobe Sign formats for signature tags

The table shows the supported syntax, where 1 is the signer index [1..10]:

Purpose	Adobe sign Tag
Signer Signature Tag	{{<UniqueFieldName>_es_:signer1:signature}}
Signer Initials Tag	{{<UniqueFieldName>_es_:signer1:initials}}
Signer Title Tag	{{<UniqueFieldName>_es_:signer1:title}}
Signer Company Tag	{{<UniqueFieldName>_es_:signer1:company}}

Purpose	Adobe sign Tag
Signer Full Name Tag	{{<UniqueFieldName>_es_:signer1:fullname}}
Signer E-mail Address Tag	{{<UniqueFieldName>_es_:signer1:email}}
Signer Signed Date Tag	{{<UniqueFieldName>_es_:signer1:date}}

Standard Adobe Sign Tag syntax:

```
{{<Prefix><FieldName>_es_:<Role>:<FieldType>:<Rule1>:<Rule2>}}
```

Ignored Adobe Features

Adobe Sign has some optional flags that can be applied to a tag, such as required and read-only. These flags can be added using prefixes, as follows:

```
{{*!<UniqueFieldName>_es_:signer1:title}}
```

 where * == required, ! == read-only

The flags can be trailing the suffix, as follows:

```
{{<UniqueFieldName>_es_:signer1:title:readonly:required}}
```

Unsupported Adobe Features

Adobe Sign has some other tag types, such as tags that allow custom tag input, and tags that dynamically execute expressions or logic to determine actions. Note that none of these types are supported by Conga Sign. These tags are treated as blocks of text.

- Custom input tag without a trailing field type: `{{<UniqueFieldName>_es_:signer1}}`
- Dynamic return value through a calculator, dynamic hide or show through show if rule: `{{<UniqueFieldName>_es_:signer1:calc(1+4):showif(1+4>6)}}`

For more information on Adobe Sign text tags, see [Adobe Sign text tags](#).

List of recognized DocuSign formats for signature tags

Conga Sign recognizes DocuSign formats for signature tags. The table below shows the supported syntax, where 1 is the signer index [1..10].

Purpose	DocuSign Tag
Signer Signature Tag	\s\
Signer Initials Tag	\i\
Signer Title Tag	\t\
Signer Company Tag	\c\
Signer Full Name Tag	\n\
Signer E-mail Address Tag	\e\
Signer Signed Date Tag	\d\

For more information on DocuSign signature tags, see [Automatic anchor text and tags](#).

Using Signature and Custom Tags

You can gather some information from another user in the documents being sent for the signature that is not addressed with Conga Sign seven standard tags. Custom tags or tag syntax can be added to collect additional information in the document. You can drag and drop the custom tag onto your document and add a label in the tag itself.

Custom tags

You can also add custom tags, like the customtext tag. When pre-tagging documents through tagging syntax, use the following format:

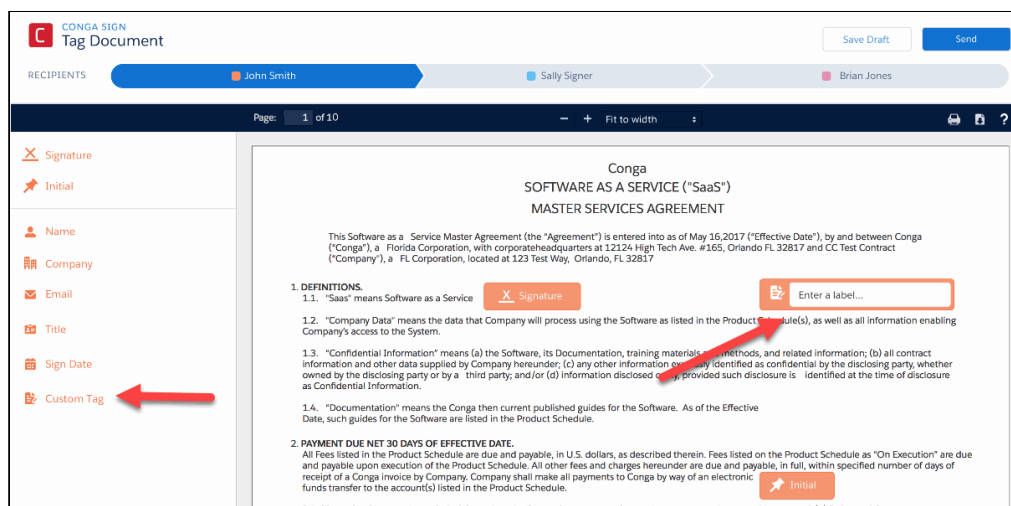
`\customtext1{"label": "value"}\` where value is the label name the signers see. For example, If you want the label to be a PO Number: `\customtext1{"label": "PO Number"}\`

You can give customtext tags attributes and properties. For example, you can add a label to the custom text tag: `\customtext1{"label": "Add a label here"}\`

Similar to other tag properties, the 1 represents the signer ID. The attributes are encoded in JSON format and attribute keys are case sensitive. The key and values must be within quotation marks.

i Labels are the only properties supported.

Custom tags can be found in the signature panel. Drag and drop the tag anywhere in the document and label it however best meets your business needs. You can also use custom tag syntax to add a tag to the document.



Conga Sign also supports tag syntax for other products. For example, Conga Sign recognizes the default automatic anchor text used for different DocuSign fields (formerly tags), based on the Role assigned to a signer in Salesforce. This is the text typed in documents as a placeholder when creating and saving the documents. Conga Sign also recognizes text tag syntax that is used in Adobe Sign.

Use Conga Sign Signature Tags with PDF Files

This article describes how to add Conga Sign signature tags to Conga Sign PDF file. Use this method to send a PDF file for signature with Conga Sign. These instructions are specific to Adobe Acrobat X Standard.

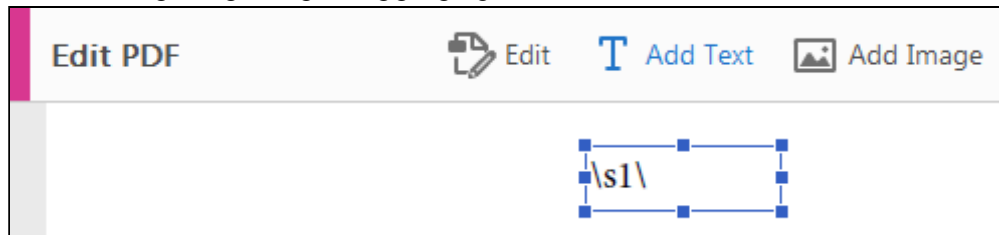
Add Conga Sign tags to a PDF file for use with Conga Sign and Conga Sign-integrated Composer solutions. This method allows users to utilize pre-tagged PDF files as Conga Sign documents or Conga Composer Templates.

i Both Conga Sign automatic anchor tags and Conga Composer merge fields populate when the template is sent for signature using this method. These instructions are specific to Adobe Acrobat X Standard.

To add Conga Sign signature tags to a PDF template (using Adobe Acrobat X Standard):

1. Create a PDF file or open an existing PDF file.


- (Optional) If using in conjunction with Conga Composers, add Composer merge fields to the PDF document.
 - For more information on creating a Composer PDF template, see [PDF forms reference guide](#).
- 2. Open the PDF file in Adobe Acrobat, click **Tools**, and then click **Edit**.
- 3. Click **Add Text** and select an area in the PDF template to place a Conga Sign tag.
- 4. Add a Conga Sign tag's tagging syntax to the text field.



- For a list of all Conga Sign tags and their respective tagging syntax, see [List of Conga Sign signature tags](#).
5. (Optional) Repeat Steps 3 and 4 to additional Conga Sign tags.
 6. Save the PDF template.

Formatting Conga Sign Date of Recipient Tag

Use tagging syntax in a Conga Sign document to format the standard Date of Recipient Date tag to your preferred date format.

 If no date format is specified in the tagging syntax, the default date format of mm/dd/yyyy is used.

The following date formats are supported:

- MM/dd/yyyy
 - Example: 01/23/2020
- dd/MM/yyyy
 - Example: 23/01/2020
- dd-MMM-yyyy
 - Example: 23-Jan-2020
- MMM dd, yyyy
 - Example: Jan 23, 2020

To format the Conga Sign Date tag, add one of the supported date formats to the Date of Tag tag syntax as depicted in the example below.

Example syntax: `\date1 {"format":"dd-MMM-yyyy"}\`

Example output: 14-Feb-2019

Write Back tags


What is Write Back?

Conga Sign Write Back functionality uses Conga Sign tags to configure and map standard and custom Salesforce fields in documents. The information added by signers within documents is sent directly to Salesforce and updates the appropriate records. The Sign Write Back functionality eliminates redundant data entry and possible errors associated with those manual processes.

How it works

The Write Back process consists of users selecting tags and Salesforce objects related to those tags, signers adding data to those tags, and then the data is directly updated to the appropriate Salesforce record.

The Write Back Configuration tab is located within Conga Sign Setup. On the Configuration tab, users can search for both standard and custom objects in Salesforce, choose fields associated with those objects that they would like to make available for write back, and define basic properties for how those fields should work when used as tags in a Conga Sign document.

 The object where the Salesforce field used for a Write Back tag is located must be configured in the Org Configuration section of Conga Sign Setup.

When fields are configured to use with Conga Sign Write Back, signers have two options to get the appropriate tags into documents.

- In the Tagging screen
 - A new section of tags is made available under the Write Back heading. The list of write back tags made available in the Tagging screen is based on the fields that were configured for the parent object from which the Conga Sign transaction originated.
 - Drag and drop a write back tag onto the document for the signer that is responsible for providing the requested information.
- Add write back tags to a document is through the Conga Sign tagging syntax

- Like standard Conga Sign tags can be added through tagging syntax, any field configured to function with write back can be added to your document templates.
- In order to simplify the process, the write back field properties screen for each write back field has an example of the syntax. RECOMMENDED: Copy and paste the example syntax into your templates. Update the signer index and you're ready to go.

i Only one signer can be assigned each write back tag to avoid any potential data conflicts during the write back process. Ensure only one signer is used per write back tag while updating your templates because any conflicts result in an error and a new transaction must be created.

Conga Sign Write Back enforces a limit of one signer per write back tag to eliminate any potential data conflicts if multiple signers were asked to provide the same information. This same limit is not in place for write back tags specific to the signer's contact, lead, or role record in Salesforce as those are specific to an individual signer.

Field Types available for Write Back

The following field types are available to use with write back:

- Lookup - Lookup fields can be used to write back to other object records, including parent and grandparent records.
- Text - Standard Text fields can be used for write back. Conga Sign is unable to write back to Text Area, Text Area (Long), Text Area (Rich), or Text (Encrypted) at this time.
- Number
- Checkbox
- Email
- Phone
- URL
- Picklist - Standard Picklist fields can be used for write back. Conga Sign is unable to write back using Picklist (Multi-select) and is unable to be used with a Dependent Picklist.

Write Back Tagging Syntax

The best way to get your newly configured write back tags into your documents is to simply copy and paste the tagging syntax from the write back field properties page. Update the signing index to reflect the appropriate signer.

Define Write Back Field Properties

*Field Label
PO Number

*API Name
congasignalpha__PO_Number__c

*Tagging Syntax
\\wb1 {"apiName":"congasignalpha__PO_Number__c"}\\

Note: When adding this syntax to templates, only one signer per write back tag

Field Options
☐ Required

Field Size
255 Character Maximum

Cancel Save

Using write back tags


The information added by signers within documents is sent directly to Salesforce and updates the appropriate records. The Sign write back functionality eliminates redundant data entry and possible errors associated with those manual processes.

A Write Back Configuration tab is located within the Conga Sign setup. On the Configuration tab, signers can search for both standard and custom objects in Salesforce, choose fields associated with those objects that they would like to make available for write back, and define basic properties for how those fields should work when used as tags in a Conga Sign document.

When fields are configured to use with Conga Sign Write Back, signers have two options to add appropriate tags into documents: Use the tagging user interface or tagging syntax.

To configure write back tags in the Tagging user interface

1. In Salesforce, open the App Launcher and select **Conga Sign**.
2. Go to **Conga Sign Setup > Write Back Configuration**.
3. Select an **Object**. You can do a quick search to locate an object.
4. Select the object fields.
5. Define the properties in the Define the Write Back Field Properties screen.
6. Click **Save**.


 Conga Sign Write Back enforces a limit of one signer per write back tag to eliminate any potential data conflicts if multiple signers were asked to provide the same information. This same limit is not in place for write back tags specific to the signer's contact, lead, or role record in Salesforce as those are specific to an individual signer.

Setting Default Values for Write Back Tags

Conga Sign has the capability to set a pre-populated default value for write back tags. Signers can keep or change the default value in the Conga Sign document. The value entered in the completed document is then updated on the corresponding field in Salesforce.

There are two ways to pre-populate default values in Conga Sign write back tags.

- Using Conga Composer merge fields to dynamically populate a record specific default value
- Populate a static value that is not record specific default value

 Dynamically populating a default value for write back tags requires Conga Composer, as it requires merge fields. This is the only method available for referencing Salesforce field values as default values in write back tags.

Using Conga Composer Merge Fields to Set a Dynamic Default Value

After obtaining the text-based merge field from Conga Composer's Template Builder, adjust the write back tag's syntax to include the code for default values as demonstrated below.

```
\wb1 {"apiName": "Account_Field__c", "defaultValue": "{{ACCOUNT_FIELD}}"}\
```

Using Static Data to Set a Default Value

Use the syntax below to reflect a static value in a write back tag.

```
\wbl {"apiName": "Account_Field__c", "defaultValue": "Blue"}\
```

Setting a Write Back Tag as Optional or Required

Required versus Optional Write Back Tags

Conga Sign Write Back tags can be set as required or optional for signers.

If a Write Back tag is set as Required, Signers cannot complete the Conga Sign Transaction until the required Write Back tag is filled out. Conversely, configuring a Write Back tag as optional allows signers to complete the Conga Sign Transaction without filling out the Write Back tag.

- ✔ Write Back functionality allows one signer per Write Back tag in a Conga Sign Transaction to prevent multiple field updates on the same record in Salesforce.

Setting a Write Back Tag as Required or Optional

Setting a Write Back tag as required or optional is done in the Write Back Configuration section of Conga Sign Setup object.

To set a tag as Required or Optional

1. Navigate to **Write Back Configuration** section of the **Conga Sign Setup** object.
2. Click the drop-down arrow next to the related field and click **Edit**.
3. Check the **Required** field under **Field Options** to make the tag required for signers.

Define Write Back Field Properties

***Field Label**

Favorite Movie

***API Name**

Favorite_Movie__c

***Tagging Syntax**

\wb1 {"apiName":"Favorite_Movie__c"}\

Note: When adding this syntax to templates, only one signer per write back tag

Field Options

☒ Required

Field Size

155

Character Maximum

Cancel

Save

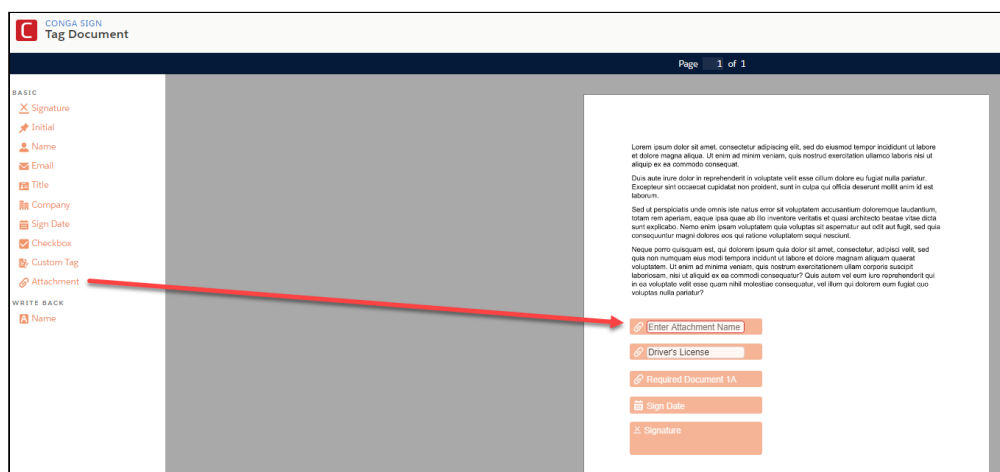
The required Write Back field appears with a red outline and the Signer cannot complete the signing until the field is filled out.

Attachment Tags

The Attachment tag allows recipients to upload files directly from the signing user interface into Salesforce. Files uploaded with the Attachment tag automatically upload to the related Salesforce record as a Salesforce File. This provides Conga Sign users the ability to collect additional documents and images in a Conga Sign Transaction, such as identification, purchase orders, and proof of insurance.

Adding an Attachment Tag to a Conga Sign Document

Simply drag and drop the Attachment tag onto your Conga Sign document in the Conga Sign tagging user interface. Once the tag is added to the document, enter a unique label for the specific Attachment tag to prompt users to upload the correct file.



Conga Sign users can also use [tagging syntax](#) to pre-tag the document with Attachment tags.

Example tagging syntax:

- \attachment1 {"label":"Driver's License"}\
- \a1 {"label":"Driver's License"}\

⚠ It is recommended to use a label in an Attachment tag's tagging syntax. This ensures that the tag is appropriately named in the Conga Sign document. A label is required for an Attachment tag when adding it to the Preview and Tag user interface. If no label is provided in an Attachment tag's tagging syntax and the document is sent for signature, the tag will appear with no text.

The Attachment tag is set to Optional by default. To make Attachment tags required in Conga Sign documents by default, change Attachment's value from Optional to Required in the Basic Tag Configuration section of Conga Sign Setup. Users can also set an Attachment tag to Required in the tagging user interface by modifying the tag properties or by designating it as Required with tagging syntax.

i Supported file types for the Attachment tag include PDF, .Doc, .Docx, JPEG, PNG, and GIF. The maximum filesize for an attachment is 10 MB.

Uploading an Attachment as a Recipient

To upload a file with the Attachment tag, click an Attachment tag in a Conga Sign document.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.


Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.


Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt.

Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?



 Proof of Insurance

 Driver's License

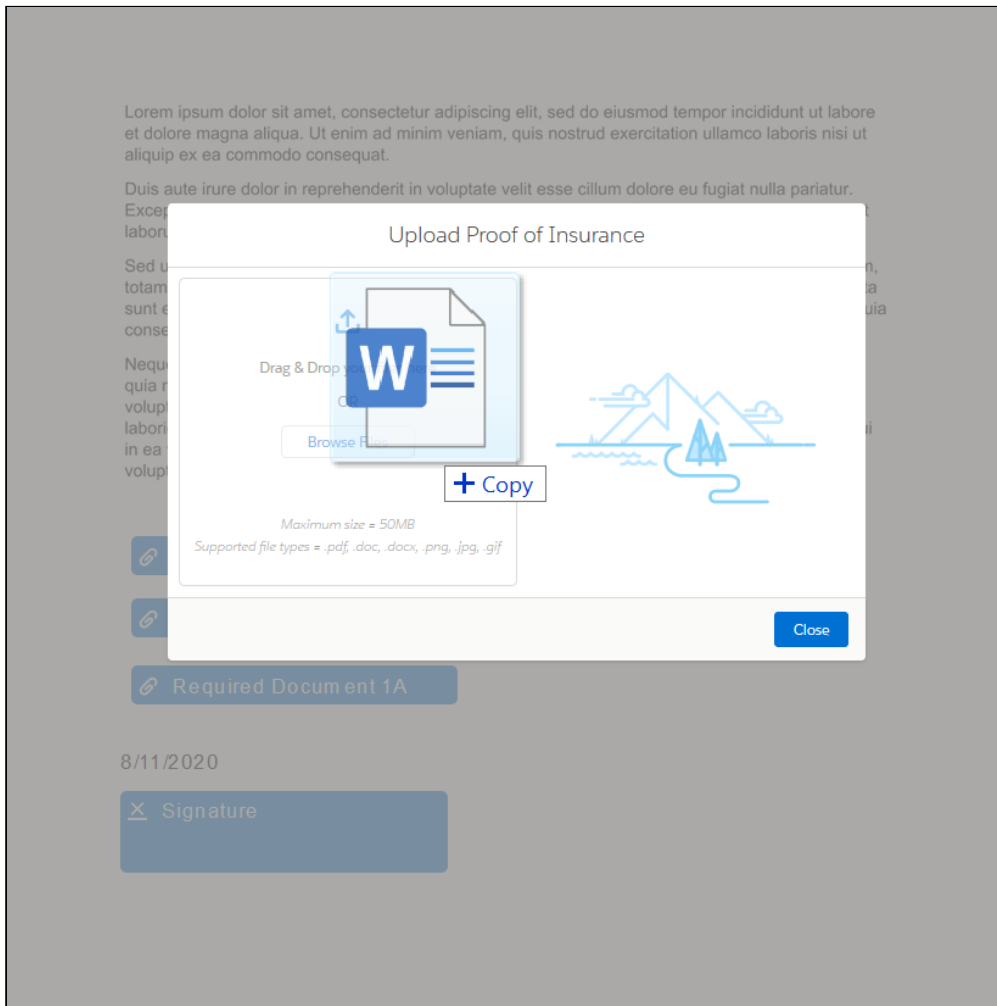
 Required Document 1A

8/11/2020

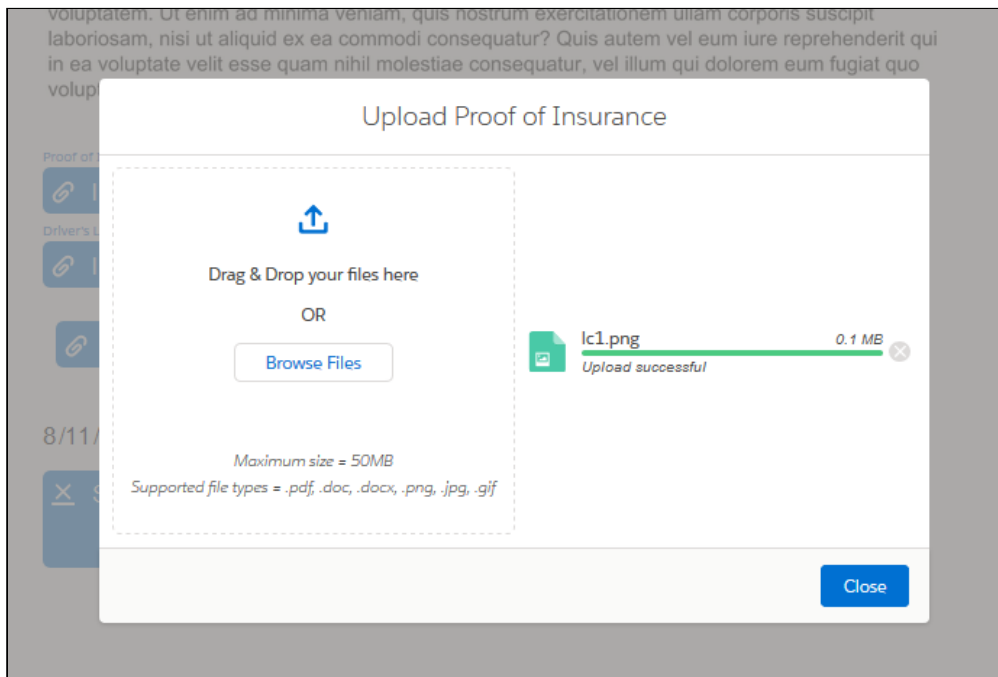
 Signature

Recipients can then click Browse Files to search and select a file on their device, or drag and drop a file into the file drop zone.

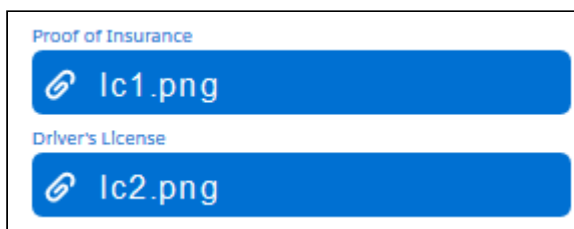
 Users can upload a maximum of one file per Attachment tag.



Once a file is successfully uploaded to the Attachment tag, the progress bar displays as completed and a message confirms the successful file upload.



Upon clicking Close, the Attachment tag's designated label is displayed above and the tag itself displays the uploaded file name.



Recipients can remove a file uploaded to an Attachment tag (before the Conga Sign document is completed) by clicking the Attachment tag and then clicking the x icon next to the progress bar.

i The Attachment tag's file is uploaded to Salesforce once all recipients complete their assigned tags and the Conga Sign Transaction is complete.

Resizing Conga Sign Tags

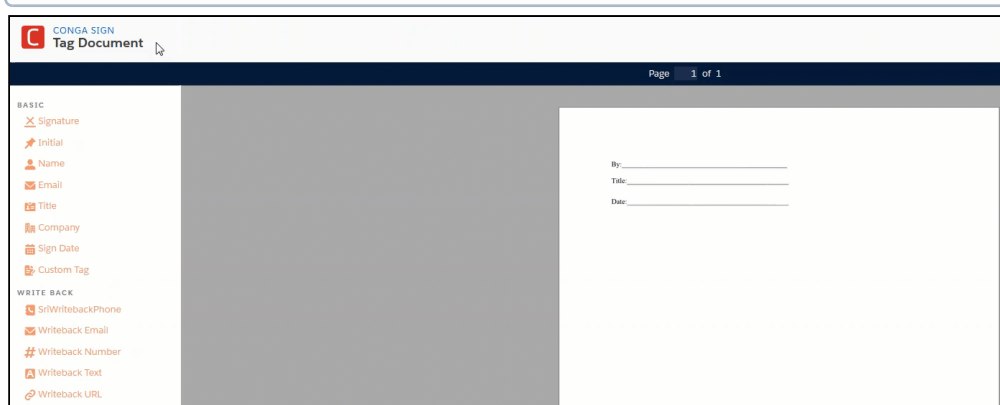
Conga Sign has the capability to increase and decrease the size of [Conga Sign tags](#) within the Preview and Tag user interface.

Different use cases and the corresponding templates have varying requirements, many of which need smaller and larger tag sizes. Conga Sign offers flexibility to change tag size on

the fly. You can resize both standard and custom Conga Sign tags by simply hovering above the tag and dragging the corners.

In the Preview and Tag user interface, all tags (Custom tags, Basic tags, and Write Back tags) initially have the same default size. When resizing tags, the text size and tag properties are automatically proportioned to the resized tag. Conga Sign offers four tag sizes for text fields, each with matching text size.

i It is recommended to resize and test Conga Sign tags in your template prior to sending for signature. This allows you to determine the appropriate tag size to best fit the template.



Setting the Text Size of Conga Sign Tags

Conga Sign features the ability to set the text size of tags in a pre-tagged document. Setting the text size of tags in a pre-tagged document saves users time, as they do not need to re-size tags manually in the Preview and Tag user interface.


i When using a Conga Sign integrated Composer button, set the text size of your tags in the pre-tagged document and leverage the DS7 parameter to send the document with resized tags in background mode.

Setting the text size in a pre-tagged document is available for standard tags, custom tags, and write back tags. The available sizes are small, medium, large, and extra-large. All tags other than the standard Initial and Signature tags retain the default medium tag size if a tag size is not defined. The standard Signature and Initial tags retain a default size of extra-large.

Setting the Text Size for Different Tag Types

The syntax for tag size differs depending on the type of tag. The Signature and Initial tags use the size property in the tagging syntax. All other tags use the textsize property in the tagging syntax to determine tag size. Add the following syntax to the end of your tag:

- Signature and Initial tags - {"size":"add tag size here"}
- Write back tags - {"textsize":"add tag size here"}
- All other Conga Sign tags - {"textsize":"add tag size here"}

 The textsize property will not work with the Signature and Initial tags. Ensure that you use the correct size or textsize property syntax when setting a tag's size.

Available Tag Sizes

Use the following sizes in the tag syntax to modify the size of the tags in the pre-tagged document.

- x-small
- small
- medium
- large
- x-large

Tag Size Syntax Examples for Each Tag Type

The examples below demonstrate tag sizing syntax for each tag types.

- Signature tag: \signature1 {"size":"medium"}\
- Initial tag: \initial1 {"size":"small"}\
- Standard tag: \title1 {"textsize":"small"}\
- Custom tag: \customtext1 {"label":"Address", "textsize":"medium"}\
- Write back tag: \wb1 {"apiName":"Customer_Email__c", "textsize":"x-large"}\

Font Sizes for Each textSize

The font sizes below correspond to each textSize for Conga Sign tags.

⚠ The Signature and Initial tags do not having corresponding font sizes.

- x-small - 8 pt
- small - 10 pt
- medium - 13 pt
- large - 16 pt
- x-large - 18 pt

Example of Different Tag Sizes in a Conga Sign Document

Here is an example of tag sizing for different tag types in a pre-tagged document.

Please fill out the information below.

`\signature1 {"size": "large"}\`

`\initial1 {"size": "medium"}\`

`\wb1 {"apiName": "PO Number _c", "textsize": "small"}\`


`\company1 {"textsize": "medium"}\`


`\title1 {"textsize": "large"}\`


`\date1 {"textsize": "x-large"}\`


Test out the sizing of each tag in the Preview and Tag user interface to ensure it best fits your document.


Please fill out the information below.


 Signature

 Initial

 PO Number

 Company

 Title

 Sign Date

Once the Conga Sign document is received, the input text automatically resizes based upon the defined size for each tag.

Please fill out the information below.





Setting the Width of a Conga Sign Tag

Conga Sign allows users to set the width of tags in a pre-tagged document. Setting the width of tags in a pre-tagged document determines the amount of space (in pixels) that a signer has to enter data in the Signing user interface.

Signers can enter more characters than the specified tag width displays. Tag width affects the display of text input in a tag within the Signing user interface. In the final document, all entered text will render despite whether or not it is over the designated width's amount of pixels.

Setting the tag width in a pre-tagged document is available for all Conga Sign tags except for the Signature, Initial, and Checkbox tags.

 The default tag width is 200 pixels.

Tag Width Syntax

Tag width syntax dictates a tag's width that a signer sees during the signing portion of a Conga Sign Transaction. Users can reference both [text size](#) and width syntax to determine both the text size and width of a Conga Sign tag.

For example, a Conga Sign tag with a width of 150 allows signers to input 150 pixels worth of text before it is cut off in the Signing user interface. If the signer enters more text than specified in the tag width syntax, all entered text displays in the completed document, but text that surpasses the designated width does not display in the Signing users interface.

Use the syntax below to a tag to determine the tag width:

- {"width":"number of pixels"}
- {"w":"number of pixels"}

Tag Width Syntax Examples

The examples below demonstrate tag width syntax:

- Company tag with tag width syntax: \company1 {"width":"100"}\
- Custom tag with tag size and width syntax: \customtext1 {"label":"Address","textsize":"large","w":"150"}\
- Write back tag with tag size and width syntax: \wb1 {"apiName":"Customer_Email__c", "textsize":"x-large","width":"275"}\

Examples of Different Tag Sizes in a Conga Sign Document

Below is an example of tag width and sizing syntax for different tag types in a pre-tagged document.

Company Name:	\company1 {"width":"150"}\
State Code:	\customtext1 {"label":"State Code","textsize":"large","w":"50"}\
Email:	\wb1 {"apiName":"congasignalpha_Writeback_Email__c", "textsize":"small", "width":"250"}\

Test out the width and sizing of each tag in the Tagging user interface to ensure it best fits the Conga Sign document.

Company Name:

State Code:

Email Address:

Once the Conga Sign document is received, the tag width adjusts to the pixel size defined in the tagging syntax.

Company Name:

State Code:

Email:

Conditionally Display a Conga Sign Tag in Word

Using Conga Composer and Conga Sign, you can display or hide a Conga Sign Tag based upon a merge field's value.

For example, you can leverage an IF statement in Word to show the standard \date\ tag if the {{Date_Requested_Checkbox_Field}} merge field is equal True. If the {{Date_Requested_Checkbox_Field}} merge field is equal to False, then the standard \data\ tag is hidden.

Syntax for Referencing Conga Sign Tags in an IF Statement

Below is example syntax for referencing Conga Sign tags in a Word IF Statement.

Standard Tag: `\\date\\`

Custom Tag: `\\customtext1 {\\label\\:"Address\\"}\\`

Write back Tag: `\\wb1 {"apiName\\:"account__text_field__c"}\\`

IF Statement Syntax

See the syntax below for an IF statement corresponding to the example use case.


Example IF statement: `{ IF "<<Date_Requested_Checkbox_Field>>" = "True" "\\date\\ " "No Date Requested" }`

For more information on using a Word IF Statement with Conga Composer, see [How to Compare Two Values in Word Templates using IF Statements](#) topic in *Composer Guide*.

Text Wrapping Conga Sign Tags

Use Text wrapping to wrap a Conga Sign tag's input text in the final document. Text wrapping prevents a completed tag's input text from cutting off or overflowing off the page horizontally.

Completed tags with text wrapping that exceeds the [defined tag width](#) or the 255 character default limit automatically wrap and display text on new lines. The point at which text wraps is determined by a tag's width. Without text wrapping, lengthy tag entries can often be cut off and will not fully display in the final document.

 Text wrapping is reflected in the completed Conga Sign documents. It does not appear until the Conga Sign document is complete.

Text wrapping is only supported in pre-tagged documents. Conga Sign users cannot add text wrapping in the Preview & Tag user interface.

Words are never broken up in tags with text wrapping. If a word cannot fit on the same line, it is pushed to the next line automatically. Conga Sign wraps text by shifting the first lines of the text upwards, as opposed to shifting new lines down.

 Signature, Initial, Date, and Email tags do not support text wrapping.

Text Wrapping Syntax

Add the "wraptext":"true" syntax or "wt":"true" (short form) to a tag's pre-tag syntax to enable text wrapping.

Example Syntax:

```
\customtext1 {"label":"Description","width":"300","textsize":"small","wraptext":"true"}\
```

```
\title1 {"width":200,"textsize":"medium","wt":"true"}\
```

Best Practices for Text Wrapping

- Use smaller font size for pre-tag syntax so it does not take up as much space in the document.
- Use the [text size](#) and [tag width](#) properties with text wrapping.
- Test different variations of text size and tag width to ensure the completed tag fits appropriately in the document.

Examples Text Wrapping with Text Size and Tag Width Variations

The examples below depict using the different tag sizes and tag widths in conjunction with text wrapping.

Pre-Tag Syntax:

```
\customtext1 {"label":"Description 1","width":"500","textsize":"large","wraptext":"true"}\
```

```
\customtext1 {"label":"Description 2","width":"400","textsize":"medium","wraptext":"true"}\
```

```
\customtext1 {"label":"Description 3","width":"200","textsize":"small","wraptext":"true"}\
```

```
\customtext1 {"label":"Description 4","width":"150","textsize":"x-small","wraptext":"true"}\
```


Signing Experience User Interface:

Page 1 of 1

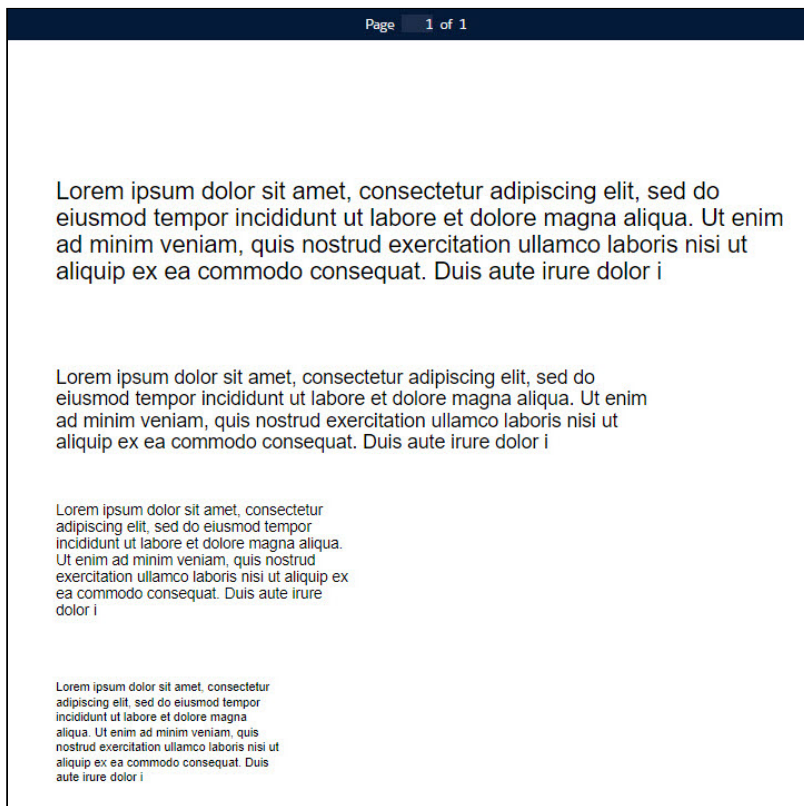
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod

Lorem ipsum dolor sit amet, consectetur adipiscing

Lorem ipsum dolor sit amet, consectetur

Completed Tags with Text Wrapping:



Creating a Document to Sign

Another option is to use Conga Composer to generate the document with signature tags in it and then send for signature with Conga Sign.


To upload the document and add tags

1. Click **Preview and Tag**.
2. Tag the document with signature fields, initial fields, or any other fields you require. Drag and Drop the blocks from the left-hand panel into the document.

Documents requiring more than one signer have an extra bar at the top of the document with color-coded names to distinguish field blocks. You can manually add signature tags as well. For a list of signature tags, see [List of Conga Sign Signature Tags](#).

Creating New Recipients in Create Transaction UI

Conga Sign provides users the capability to easily create and add new Conga Sign Recipients in the Create Transaction user interface. In addition to selecting existing Contacts, Leads, and Users records, Conga Sign allows users to create new Conga Sign Recipients for a specific transaction without creating new Contact, Lead, and User records.

 Creating and adding a new Conga Sign Recipient to a transaction does not create a Contact, Lead, or User record in Salesforce. It creates a Conga Sign Recipient (custom object) record specific to that transaction only. New Conga Sign Recipients created with the parameters or methods below cannot be added or re-used in other transactions.

Administrators can also use the following [Conga Sign parameters](#) and [Conga Composer integration parameters](#) to dynamically create and add new Conga Sign Recipients to a transaction.

- Conga Sign Parameters to Add New Recipients
 - recipient[1-10]email
 - recipient[1-10]first
 - recipient[1-10]last
 - recipient[1-10]name
- Composer Integration Parameters to Add New Recipients
 - CSEmail[1-10]
 - CSFirst[1-10]
 - CSLast[1-10]
 - CSName[1-10]

To add a New Recipient in the Create Transaction User Interface

1. Navigate to the **Create Transaction** user interface by launching a Conga Sign solution.
2. Click **Create New Recipient**.
3. Complete the required fields.

- a. First Name
 - b. Last Name
 - c. Email Address
4. Click **Save**.

Upon creating a new Recipient, Conga Sign adds that Recipient to the transaction and creates a new Conga Sign Recipient record.

Sending Documents for eSignature

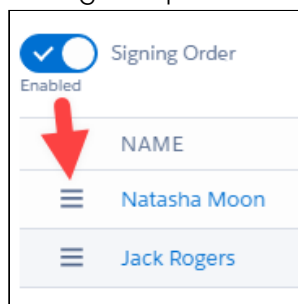
You can initiate an eSignature process from any object in Salesforce using pre-defined templates or by adding signature tags in real-time. In order to send documents for eSignature, you can use documents already attached to the record and select the documents from the list of the Available Documents, or you can upload the documents to be sent for eSignature using the Upload button, or you can use the Search bar to find documents stored in Salesforce. You can select multiple documents for signing within a single transaction. Then, you can select up to 10 recipients, set the signing order, customize the subject and message of the email, and set the expiration & reminders.

To send documents for eSignature


1. Navigate to a record on an object where [Conga Sign is configured](#). (For example, navigate to an Opportunity record.)
2. Click **Send with Conga Sign**.
3. **Add Documents**
 - a. You can search for a file in Salesforce. Search look at all the Files in your Org.
 - b. Click **Upload File** to upload any Word or PDF file. You can also drag and drop the file.
 - c. **Available Documents** list shows all documents attached to that record. Select the documents that you want to send for eSignature. You can select multiple documents.
 - d. To sequence, drag documents into the correct order use the icon (≡) on the left.
 - e. To display only selected documents in the Available Documents list, click the toggle for **Only Show Selected** to enable it.
4. **Add Recipients**.
 - a. Search for Salesforce Contacts, Leads, or Users within Salesforce.
 - b. If the Recipient is not a Contact, Lead, or User in Salesforce, click **Create New Recipient** and complete the required fields to add a non-Salesforce Recipient.

For more information about creating a new Recipient in a Conga Sign Transaction, see [Creating New Recipients in Create Transaction UI](#).


- c. Configure optional features:



- Customize **Role**, **Language**, and **Authentication**, for each Recipient.

 For more information on setting up authentication, refer to [SMS Authentication](#).

- Enable **Signing Order** if you want the document to be signed by recipients in a specific order. To sequence, drag signers into the correct order using the icon (≡) on the left.

 If you want CC role(s) to receive emails at the beginning and end of the transaction, place them at the beginning of the recipients list. If you want CC role(s) to receive an email only after a document has been signed, place them at the end of the recipients list.

5. Advanced Options

- Compose a customized Subject and Message. If none is provided, a default is used.
 - Set the date and time for transaction expiration and reminders if desired. You can configure the following fields:
 - **Transaction Expiration**- Select the date and time when you want the transaction to expire. After the transaction is expired, the signer receives an email notification.
 - **Expiration Reminder** - Select the date and time when you want to send a reminder to the signer about the expiration of the transaction.
 - **Request Reminder** - Enter the number of days in which you want to send a reminder to the signer after you send the document. For example, you send the document today and you enter the value as 5. A reminder will be sent to the signer after 5 days.
- If tags exist, click **Send Now**.
 - If you need to add tags or preview, click **Preview and Tag**.

- a. Tag the document with signature fields, initial fields, or any other fields you require. Documents requiring more than one signer have an extra bar at the top of the document with color-coded names to distinguish field blocks.
- b. Drag and Drop the blocks from the left-hand panel into the document.
- c. In case of multiple documents, select a file name from the dropdown list at the top to navigate between the documents. The dropdown displays the file name that you are viewing as you scroll between documents.
- d. Click **Send**.

This concludes the send for eSignature process using Conga Sign. You receive an email confirming the documents are sent and a link to the new audit trail. Your recipients are sent emails inviting them to sign the documents. Visit [Signing a Document](#) to learn about a recipient's signing experience.

Sending Transactions with Multiple Documents

You now have the ability to use multiple documents in a single Conga Sign transaction. There are multiple changes to the way that documents are attached to the Transaction record.

Fields on Transaction page layout

When multiple documents are used in a Conga Sign Transaction, the **Original Document Id**, **Content Document Id** and **Final Document Id** are not populated. They are populated, however, when only using a single document per transaction.

Conga Sign Documents related list

There is a new related list added to the Transaction page layout, **Conga Sign Documents**. All documents on the Transaction have a new record created for this new related list, which has its own **Original Document Id**, **Content Document Id** and **Final Document Id** fields, which are populated when using multiple documents for a single Conga Sign Transaction. Also included is a **Type** field, to indicate the type of documents.


Conga Sign Documents has its own custom links for **View File** and **Download File**, where your users can view or download transaction files.

 **Note**

The custom links **View File**, **Download Original Version**, and **Download Final Version** on the Transaction record do not pull attached documents if multiple documents have been sent for a single transaction. They will still function if a single document is used. It is recommended that System Administrators remove these links from the Transaction page layout and use the **Conga Sign Document** links.

New Users to Conga Sign have the related list already on the Transactions record. Users that have previously installed Conga Sign need to manually add the related list to the page layout.

To add the related list page layout

1. On Transaction, select Edit Layout
2. In the page layout editor, select **Related Lists**, then drag the **Conga Sign Documents** option to the related list segment.
3. Configure the Conga Sign Related List properties.
 - a. From the Conga Sign Documents related list, click the  icon and select the fields you wish to display on the related list.
 - b. We suggest including **Document Name**, **Document Order**, **Type**, and **Last Modified Date**.


Sending Transactions using Send On Behalf Of


Send on Behalf allows you to send another User's information to someone other than the user who sends the Conga Sign transaction out. The transaction represents a user who does not create the initial transaction.

To use Send On Behalf Of features, you must set up either the necessary parameter on the Conga Sign **Send with Conga Sign** button or configure the Conga Sign [Business Units](#).

As you create the initial transaction using Send on Behalf Of, you will receive the success or failure email notification, informing you of the transaction status. The individual you have designated as the Send on Behalf Of recipient will receive all further transaction email notifications.


Events in the [Audit Trail](#) are updated to reflect any transactions that are sent using the Send on Behalf Of feature.

Transaction ID	
36v6am8ak3mepssytp80ftwu73buj706jmpz9vr0jkv7hbp78l	
Transaction Status	
COMPLETE - 10/26/2021	
File Name	File Hash
LoremIpsum.pdf	58203576f3dc4b8ff6fb2c3d50a01cb2970b3ae970abee34e734fa60ba77866c
Full Audit Trail	
ACTIONS	
 Transaction was created by User User <senderexample@>.	
 Transaction was sent by User User on behalf of Blue Cube Team <soboexample@>.	

 Replying to a received Send on Behalf of email may cause responses to be caught by specific spam filters that are looking for emails that have different "reply-to" and "from" email addresses.

Conga Sign Transaction records display the Send of Behalf of User's information in two fields.


- **Send on Behalf Of Name:** the name of the User the transaction is sent on behalf of.
- **Send on Behalf Of Email:** the email of the User the transaction is sent on behalf of.

 If you have directly installed Conga Sign v 1.74.0, these fields will already exist on the page layout. If you have upgraded from a previous version to 1.74.0, you will need to add the fields to your page layout manually.

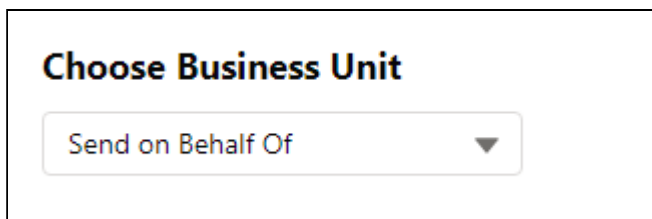
Configuring Send on Behalf of using Business Units

1. Navigate to Conga Sign Setup.
2. Select the Business Units option and click **Add New Business Unit**.
3. Set the name for your Business Unit as Send on Behalf Of, or any name of your choice, and the API name will autofill as you type. Click **Save**.

4. Under the input section for Send on Behalf Of, add the Name and Email of the User's name you want to send information on behalf of.

 Only one user can have transactions sent on their behalf.

When you create a transaction, you will be able to select your new option from the **Choose Business Unit** dropdown.



The image shows a rectangular box with the title "Choose Business Unit" in bold. Below the title is a dropdown menu with the text "Send on Behalf Of" and a downward-pointing arrow.

Configuring Send on Behalf of using Send with Conga Sign

1. Navigate to Salesforce Setup.
2. In Setup, navigate to an object where [Conga Sign is configured](#). (For example, navigate to Opportunity.)
3. Click **Edit** on the **Send with Conga Sign** button.
4. Add either the &businessUnit parameter or the &sendonbehalfname and &sendonbehalfemail parameters. See [Conga Sign Parameters](#) for more details.

Configuring Send on Behalf of using Conga Composer Integration

Conga Composer utilizes two parameters to integrate with Conga Sign's Send on Behalf Of feature.

- **&cssendonbehalfname:** the name of the User the transaction is sent on behalf of.
- **&cssendonbehalfemail:** the email of the User the transaction is sent on behalf of.

For details on how to add parameters to Conga Composer buttons, refer to [Conga Sign Composer Integration](#).

Sending Transactions from Conga CLM

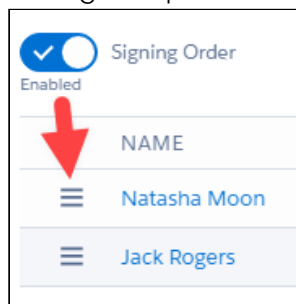
Conga Sign can be used from a Conga CLM Agreement. Once an Agreement Status is In Signature or Ready for Signature, the **Send for Conga Sign** action is available to use in the

Agreement Lifecycle Actions panel.

For more information, see [Enabling and Customizing the Action Panel](#) for CLM.

To send documents for eSignature

1. Click **Send for Conga Sign** in the Agreement Lifecycle Action panel.
2. Select agreement documents and related agreement documents and click **Next**.
3. (Optional) **Add Documents**
 - a. You can search for a file in Salesforce. Search look at all the Files in your Org.
 - b. Click **Upload File** to upload any Word or PDF file. You can also drag and drop the file.
 - c. **Available Documents** list shows all documents attached to that record. Select the documents that you want to send for eSignature. You can select multiple documents.
 - d. To sequence, drag documents into the correct order use the icon (≡) on the left.
 - e. To display only selected documents in the Available Documents list, click the toggle for **Only Show Selected** to enable it.
4. **Add Recipients.**
 - a. Search for Salesforce Contacts, Leads, or Users within Salesforce.
 - b. If the Recipient is not a Contact, Lead, or User in Salesforce, click **Create New Recipient** and complete the required fields to add a non-Salesforce Recipient. For more information about creating a new Recipient in a Conga Sign Transaction, see [Creating New Recipients in Create Transaction UI](#).
 - c. Configure optional features:



- Customize **Role**, **Language**, and **Authentication**, for each Recipient. For more information on setting up authentication, refer to [SMS Authentication](#).
- Enable **Signing Order** if you want the document to be signed by recipients in a specific order. To sequence, drag signers into the correct order using the icon (≡) on the left.

5. Advanced Options

- a. Compose a customized Subject and Message. If none is provided, a default is used.
- b. Set the date and time for transaction expiration and reminders if desired. You can configure the following fields:
 - **Transaction Expiration**- Select the date and time when you want the transaction to expire. After the transaction is expired, the signer receives an email notification.
 - **Expiration Reminder** - Select the date and time when you want to send a reminder to the signer about the expiration of the transaction.
 - **Request Reminder** - Enter the number of days in which you want to send a reminder to the signer after you send the document. For example, you send the document today and you enter the value as 5. A reminder will be sent to the signer after 5 days.
6. If tags exist, click **Send Now**.
7. If you need to add tags or preview, click **Preview and Tag**.
 - a. Tag the document with signature fields, initial fields, or any other fields you require. Documents requiring more than one signer have an extra bar at the top of the document with color-coded names to distinguish field blocks.
 - b. Drag and Drop the blocks from the left-hand panel into the document.
 - c. In case of multiple documents, select a file name from the dropdown list at the top to navigate between the documents. The dropdown displays the file name that you are viewing as you scroll between documents.
 - d. Click **Send**.

This concludes the send for eSignature process using Conga Sign. You receive an email confirming the documents are sent and a link to the new audit trail. Your recipients are sent emails inviting them to sign the documents. Visit [Signing a Document](#) to learn about a recipient's signing experience.

Once the document has been signed and Document Versioning is enabled, it will be saved back to CLM Agreement record under the Document Version in addition to the Transaction. The Document Type of the signed document is updated as the *Executed Document*. For more information about Document Versioning, see [Contract Document Versioning](#). If the Document Versioning is disabled, it will be saved back to the CLM Agreement record to Files or Notes & Attachments in addition to the Transaction. All smart clauses are marked as Final in the Agreement Clauses section.


Signing a Document

The signing process begins when you receive an email notification from Conga Sign informing you that you have a document to sign. Conga Sign sends the email notification on


behalf of the sender and contains a message from the sender and a link to open the document. If the sender has sent multiple documents for signing, the email subject contains the first document name of the document package and the email body contains a list of document names. To begin the signing process, you must open the document and review the electronic record and signature disclosure. You must agree to use electronic records and signatures to sign the document. You must complete each Conga Sign tag and add your electronic signature where required to sign or initial. Optionally, you can use the Guided Signing feature to auto-navigate to the locations of each assigned Conga Sign tag in the document. This feature is only available when the Guided Signing setting is enabled in your sender's org. After you add the required information in each Conga Sign tag and add your electronic signature, you can complete the signing process.

To sign a document

1. Open the email you received from Conga Sign.
2. Click **View Document**. If the sender has sent multiple documents for signing, the **View Document** link opens all documents in a single transaction.
3. View the electronic record disclosure and click **I Agree**.


 Clicking **I agree** confirms that the Signer accepts the electronic record disclosure. The full electronic record disclosure is available in the eSignature Adoption and Terms portion of the signing user experience.

4. In the case of multiple documents, select a file name from the dropdown list at the top of the signing user interface to navigate between the documents. The dropdown displays the file name that you are viewing as you scroll between documents.
5. Click **Begin** to navigate to the first tag in the document.
6. After you click the **Begin** button or complete the first tag, it changes to **Next**.
or
After you complete any tag except the first tag, it changes to **Back** and **Next**.
7. Click **Next** to navigate to the next tag.
8. Click **Back** to navigate to the previous tag.


 Steps 5 to 8 are optional. The **Begin** button is only available when the Guided Signing setting is enabled in your sender's org. For more information on how to configure the Guided Signing setting, refer to [Setting Guided Signing](#) in *Conga Sign Administrator Guide*. If the Guided Signing setting is disabled, you are

automatically navigated to the next tag after completing the first tag. In the case of multiple document signing, you are automatically navigated to the next document after all tags of the previous document are navigated.

9. Customize or draw your Signature and Initials if Signature or Initials tag is present. Alternatively, you can select Upload your Own to upload an image of your signature or initials.
10. Click **Adopt Signature and Sign**.
11. Click on each Conga Sign tag to complete the tags. Date fields are automatically populated. Additional fields, such as title, email, or company, are updated by typing text values into the fields directly.

 If a Stamp tag has been added, click the Stamp tag to give it a name and upload your signature Stamp. Optionally, you may mark a stamp as your default Stamp. Click **Adopt and Sign**.

12. Click **Complete Signing** once all tags are complete.
13. The signing process is complete. If all recipients have finished signing, you can click **Download Document** to save a copy if you choose.

 If multiple documents are signed in a single transaction, **Download Document** downloads a .zip file containing all of the signed documents.


You receive another email when all requested signatures are collected. This email contains a link to the audit trail as well as a PDF attachment of the fully executed document. For multiple document signing, all PDF attachments signed for a transaction are included in the email.

Using Signature Image Upload

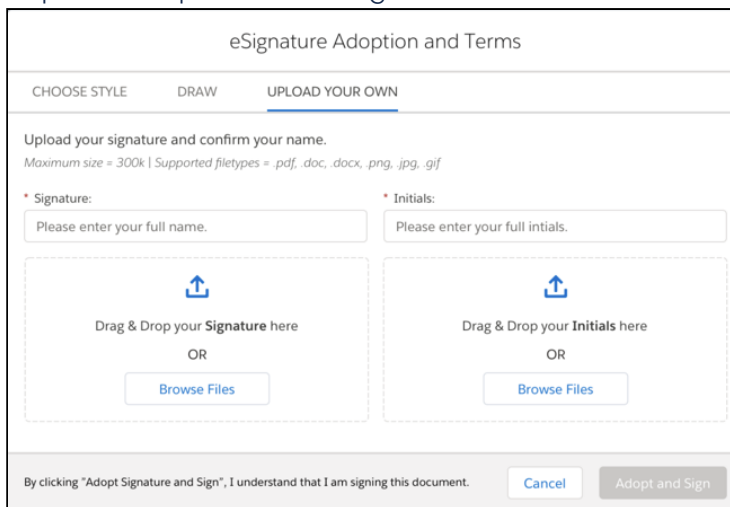
With Conga Sign Signature Image Upload, you are able to upload your own customized signatures and initials to be used as a signature for a transaction. This topic instructs you on how to upload your own custom Signature and Initial images.

To sign a document with Signature Image Upload


1. Open the email you received from Conga Sign.
2. Click **View Document**. If the sender has sent multiple documents for signing, the **View Document** link opens all documents in a single transaction.
3. View the electronic record disclosure and click **I Agree**.

 Clicking **I Agree** confirms that the Signer accepts the electronic record disclosure. The full electronic record disclosure is available in the eSignature Adoption and Terms portion of the signing user experience.

4. In the case of multiple documents, select a file name from the dropdown list at the top of the signing user interface to navigate between the documents. The dropdown displays the file name that you are viewing as you scroll between documents.
5. Select **Upload your Own**, then drag and drop the image of your Signature and Initials from your saved location, or click the Browse Files buttons to select them from your stored file location. If there is no Initials tag in the Sign Document, you are not required to upload an image for it.



The screenshot shows the 'eSignature Adoption and Terms' interface. At the top, there are three tabs: 'CHOOSE STYLE', 'DRAW', and 'UPLOAD YOUR OWN' (which is selected). Below the tabs, the text reads 'Upload your signature and confirm your name.' followed by 'Maximum size = 300k | Supported filetypes = .pdf, .doc, .docx, .png, .jpg, .gif'. There are two main sections: one for 'Signature' and one for 'Initials'. Each section has a text input field with a placeholder 'Please enter your full name.' or 'Please enter your full initials.', a dashed box with an upload icon and the text 'Drag & Drop your Signature here' or 'Drag & Drop your Initials here', and a 'Browse Files' button. At the bottom, there is a disclaimer: 'By clicking "Adopt Signature and Sign", I understand that I am signing this document.' and two buttons: 'Cancel' and 'Adopt and Sign'.

 A Signature/Initials image uploaded this way will remain saved and will be available to use for subsequent transactions. As long as you are within the same region in Conga Sign Setup, it will be available to use with future transactions.

6. Click **Adopt Signature and Sign**, then click on each Conga Sign tag to complete the tags. Date fields are automatically populated. Additional fields, such as title, email, or company, are updated by typing text values into the fields directly.
7. Click **Complete Signing** once all tags are complete.
8. The signing process is complete. If all recipients have finished signing, you can click **Download Document** to save a copy if you choose.

Techniques to make your images their best

To get your Signature image to look how you would like and to fit them to the image sizes, you can try the following techniques in your photo application of choice:

1. Change the dimensions of the images to fit within the recommended sizing - *400 x 150* pixels for Signature images and *150 x 150* pixels for Initials images.
 - a. Typically images can be adjusted either by pixels or by percentage.
 - b. When adjusting your image, it is recommended that you resize proportionally to keep the integrity of the image.
2. Rotate the image so it applies to the document in a way that represents your natural signature. You can use the Windows Photos application or the Paint tool, or the Apple Preview tool to resize and modify an image.
3. Images must be .png, .jpg, .gif, .ipx or .bmp file format. Your maximum file size cannot be greater than 300kb.

In-Person Signing

In-Person Signing Functionality

Use Conga Sign to capture signatures while In-Person with your prospects, customers, and partners at any time without the need to send an email invitation. In-Person Signing Transactions bypass the standard email functionality, allowing users to expedite the signing experience without needing to access their email. In-Person Signing functionality is useful for scenarios that require a signed document on the spot, such as trade shows, door to door sales, and on-site visits. In-Person Signing feature is compatible with the Salesforce Mobile App, Salesforce Classic, and Salesforce Lightning, allowing users to easily create In-Person Signing Transactions on smartphones, tablets, and laptops.

In-Person Signing Roles

In-Person Signing feature requires two different roles for each In-Person Signing Transaction: the In-Person Signer and the Facilitator. The Signer role is also available for assignment in an In-Person transaction but is not required.

In-Person Signer

The In-Person Signer is the recipient signing a Conga Sign document In-Person. The In-Person Signer Role designates the Conga Sign transaction as a In-Person Signing Transaction, rather than the default email-based Transaction. This role is required to create the In-Person Signing transaction. An In-Person Signer must be an existing Contact, Lead, or User record with a valid email address.

The In-Person Signer is always the first signer required to complete their assigned Conga Sign tags, and as such Conga Sign requires Serial Routing order for this feature. Any additional Signers defined in an In-Person Signing Transaction must complete their assigned tags after the In-Person Signer completes their portion of the Transaction.

Facilitator

The Facilitator is the Salesforce user that created the In-Person Signing Transaction and is responsible for conducting the In-Person signing. The Facilitator role is automatically populated once the In-Person Signer role is assigned to a recipient. This role is assigned to the running Salesforce user that designated the In-Person Signer role. Users must assign the In-Person Signer role before the Facilitator role is assigned. Only one Facilitator is allowed for each Sign In-Person Transaction.

A Facilitator is responsible for physically handing the device to the In-Person Signer and guiding them through the In-Person Signing transaction. The Facilitator is not considered a signer in the Transaction and therefore cannot have Conga Sign tags added or assigned for that role. However, the running Salesforce User is available to add as a Signer to the Transaction in addition to their Facilitator role. Facilitators will not receive a "Transaction Sent" email notification, as there is no need for the role to receive one.

Signer (optional)

The Signer is a recipient responsible for signing the Conga Sign document over email. This role is not required for an In-Person transaction. A Signer is emailed the Conga Sign Transaction after the In-Person Signer completes their assigned tags. Users can easily define the signer order for additional Signers when creating the In-Person Signing Transaction. In-Person Signing functionality is not yet available for Conga Sign-integrated Composer solutions.

To create an In-Person signing transaction

1. Navigate to an object record in Salesforce and click the **Send with Conga Sign** button to launch Conga Sign.
2. Select a Conga Sign document.
3. Add a **Recipient** to the Conga Sign transaction.
4. Click the **Role** dropdown field and change the Recipient's role from Signer to In-Person Signer.

Add Recipients
You must select an existing Contact, Lead, or User. Email addresses are not allowed.

Search by Name or Account

☒ Signing Order * In Person transactions require a signing order.

NAME	EMAIL	COMPANY	TITLE	ROLE
John Bond	bond_john@grandhotels.com	Grand Hotels & Resorts Ltd	VP, Facilities	In Person Signer
Amanda Knapp	leancat.vdusja65@mailosaur.io	Conga	Sales Representative	Signer

Changing a Recipient to an In-Person Signer role automatically adds the running Salesforce user as the Facilitator role.

Add Recipients
You must select an existing Contact, Lead, or User. Email addresses are not allowed.

Search by Name or Account

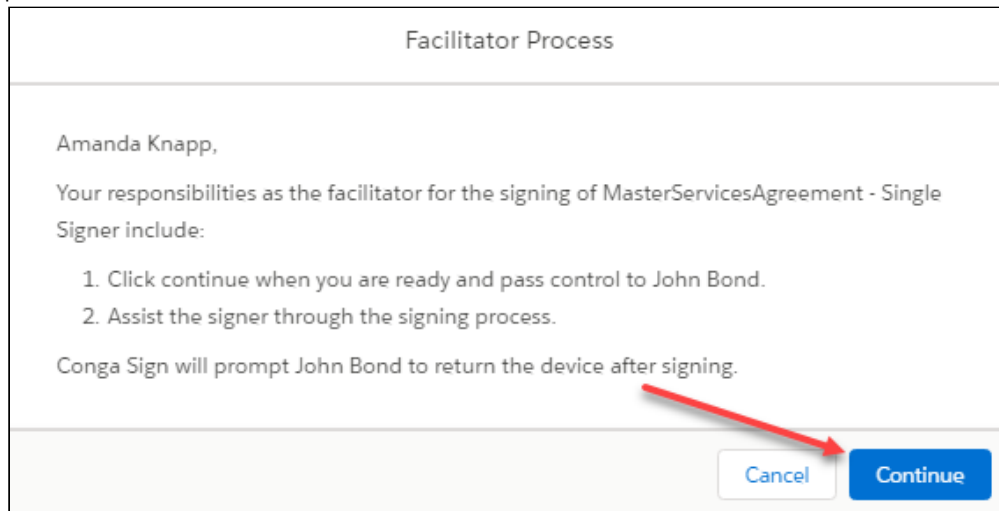
☒ Signing Order * In Person transactions require a signing order.

NAME	EMAIL	COMPANY	TITLE	ROLE
John Bond	bond_john@grandhotels.com	Grand Hotels & Resorts Ltd	VP, Facilities	In Person Signer
Amanda Knapp	leancat.vdusja65@mailosaur.io	Conga	Sales Representative	Facilitator

5. Click **Begin In-Person Signing**.
 - a. If the Conga Sign document is pre-tagged, click the **Begin In-Person Signing** button to begin the transaction.
 - b. If a user needs to add tags to the Conga Sign document, click **Preview and Tag**. After adding tags to the document, click **Begin In-Person Signing** button from the tagging user interface to begin the transaction.

To facilitate and complete an In-Person Signing transaction

1. The Facilitator is prompted to review the role's responsibilities. Click **Continue** to proceed with the transaction.



Facilitator Process

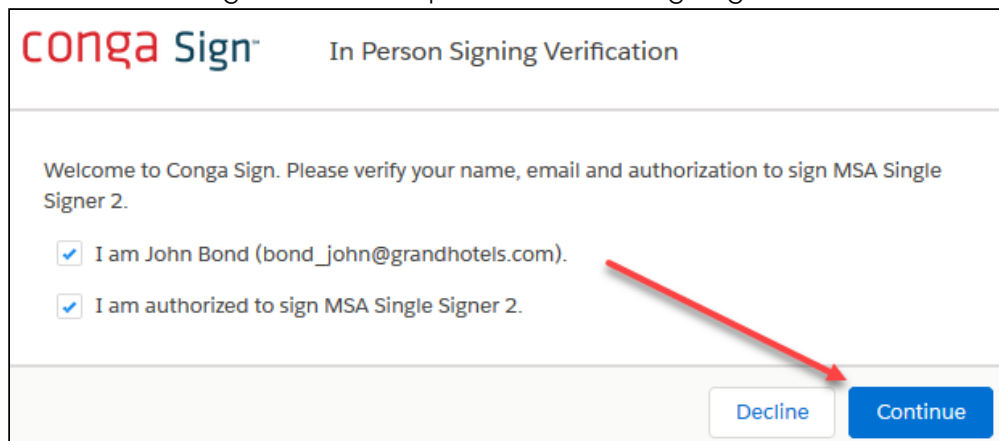
Amanda Knapp,

Your responsibilities as the facilitator for the signing of MasterServicesAgreement - Single Signer include:

1. Click continue when you are ready and pass control to John Bond.
2. Assist the signer through the signing process.

Conga Sign will prompt John Bond to return the device after signing.

2. Pass the device to the In-Person Signer.
3. The In-Person Signer must complete In-Person Signing Verification and click **Continue**.



conga Sign In Person Signing Verification


Welcome to Conga Sign. Please verify your name, email and authorization to sign MSA Single Signer 2.

☒ I am John Bond (bond_john@grandhotels.com).

☒ I am authorized to sign MSA Single Signer 2.

4. Click **I Agree** to agree to the electronic disclosure and begin In-Person Signing. Clicking **I Agree** confirms that the In-Person Signer accepts the electronic record disclosure. The full electronic record disclosure is available in the eSignature Adoption and Terms portion of the signing user experience.
5. Click **Begin** to navigate to the first tag in the document.
6. After you click the **Begin** button or complete the first tag, it changes to **Next**.
or
After you complete any tag except the first tag, it changes to **Back** and **Next**.

7. Click **Next** to navigate to the next tag.
8. Click **Back** to navigate to the previous tag.

 Steps 5 to 8 are optional. The **Begin** button is only available when the **Guided Signing** setting is enabled in your facilitator's org. If the Guided Signing setting is disabled, you are automatically navigated to the next tag after completing the first tag.

9. Customize or draw your Signature and Initials if Signature or Initials tag is present.
10. The In-Person Signer must complete all assigned and required Conga Sign tags, then click **Complete Signing**.
11. The In-Person Signer hands the device back to the Facilitator.
12. (Optional) If another In-Person Signer is present, the Facilitator clicks Continue and hands the device to the next signer. The next In-Person Signer then completes steps 8 - 12.
13. The In-Person Signer hands the device back to the Facilitator.
 - a. Click **Complete Signing** and Salesforce will redirect back to the related object record.

The transaction is complete if there are no other Signers for the transaction. Confirmation emails with the attached signed document are sent to both the Facilitator and In-Person Signer once the transaction is complete.

Canceling a Conga Sign Transaction

Conga Sign allows users to easily cancel a Conga Sign Transaction. Follow the steps below to cancel a Conga Sign Transaction.

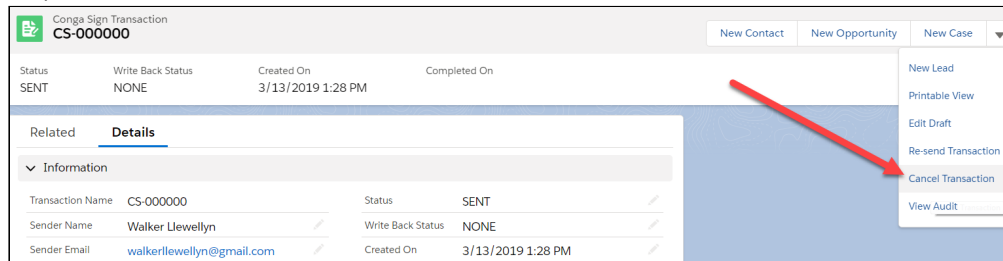
To cancel a Conga Sign Transaction as a Salesforce User

1. After sending the Conga Sign Transaction, navigate to the related Conga Sign Transaction record and click the Transaction name.

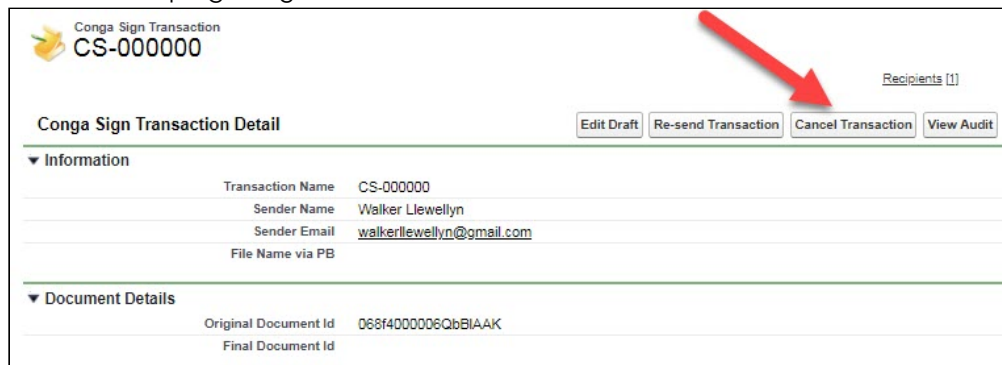
Conga Sign Transactions (6+)		
TRANSACTION NAME	SENDER NAME	STATUS
CS-000241	Walker Llewellyn	SENT

2. Click **Cancel Transaction**.

- In Lightning, the Cancel Transaction button is located in the Lightning buttons dropdown list.

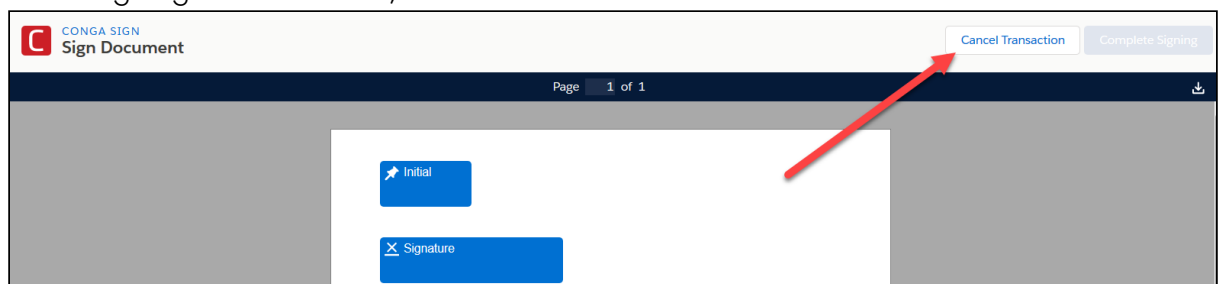


- In Classic, the Cancel Transaction button is located on the Conga Sign Transaction page layout as a button



To cancel a Conga Sign Transaction as a Signer

1. Open the initial Conga Sign Transaction email and click the **Begin Signing** button.
2. Complete the Full Name and Initials fields and click the **Agree and Start Signing** button.
3. In the Signing user interface, click the **Cancel Transaction** button.



4. Enter a short description for cancellation in the text box and click the **Confirm** button to execute the cancellation.


⚠ After canceling as either a Salesforce user or Signer, an email is sent to all parties in the related Transaction notifying them of the cancellation. The Conga Sign Transaction record's Status field's value also updates to Canceled.

Frequently Asked Questions

- At what point can a user no longer cancel a Conga Sign Transaction?
A Conga Sign Transaction cannot be canceled once it is Completed or Canceled. Users may cancel a Conga Sign Transaction at any point before the Transaction is Completed.
- As a Salesforce User sending a Conga Sign Transaction, where can I cancel the Transaction?
The sender can only cancel the transaction from the "Cancel Transaction" button on the Conga Sign Transaction record.
- As a Signer, where can I cancel the Transaction?
Signers can cancel the transaction after clicking the Begin Signing button in the invitation email and then clicking the Cancel button in the Signing user interface.
- Does deleting a Conga Sign Transaction record cancel the Transaction?
No, deleting a Conga Sign Transaction record does not cancel the Transaction.
- Can a Recipient with a Signer Role of 'CC' cancel a Conga Sign Transaction?
No, a Recipient must have a Signer Role of 'Signer' to cancel a Conga Sign Transaction.

Cancel Conga Sign Transactions in Bulk

Conga Sign provides users the capability to cancel Conga Sign Transaction in bulk by leveraging Apex code. Specifically, Salesforce administrators and developers can use the `aptx_cancelTransactionsInvocable` Apex Class as a sub Flow within a Process Builder or Flow solution to trigger bulk transaction cancellations.


 The `aptx_cancelTransactionsInvocable` Apex Class appears as Cancel Conga Sign Transactions when selecting it as in Apex Action in Flow Builder.

Additionally, administrators and developers can leverage the `apxt_bulkTransactionService` Apex Class to trigger bulk cancellations through the REST API.

Located below are tips, recommendations, and examples on using Apex to cancel Conga Sign Transactions in bulk.

Bulk Cancellation Scenarios Using Process Builder to Manage Conga Sign Transactions

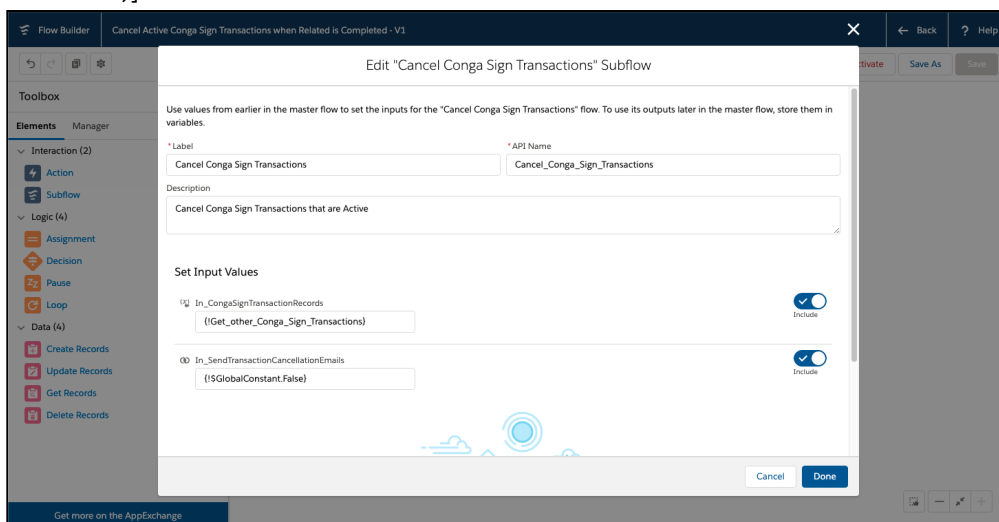
1. Cancel any incomplete Conga Sign Transactions when a Campaign status changes to Complete.
2. Cancel any incomplete Conga Sign Transactions when an Opportunity is Won.
3. Cancel any incomplete Conga Sign Transactions when a Campaign Offering Price change occurs.
4. Cancel related Conga Sign Transactions when one Conga Sign Transaction is completed.

 Salesforce solutions, such as Process Builder and Flows, leveraging Apex code are outside of the scope of Conga Support. It is highly suggested to only create custom solutions leveraging Apex Classes if you have prior experience with using Apex in Salesforce.

Common Steps to Manage and Bulk Cancel Conga Sign Transaction with Process Builder

1. Define a Process Builder or Flow to trigger off of data changes to the standard or custom Salesforce object data or workflow state.
2. Connect the Process Builder trigger action to the appropriate Flow for that use case.
3. Follow Salesforce [Best Practices for building Flows](#).
4. For each of the different scenarios, the input, criteria, and rules change, but the steps from a Flow perspective are the same:
 - a. A Get Records Data element that collects the Salesforce or Custom Objects and has the Conga Sign Transactions associated to it
 - b. A Loop Logic element that iterates over the gathered Salesforce or Custom Object(s)
 - c. A second Get Records Data element that finds the Conga Sign Transactions associated with the current iteration object
 - d. A second Loop Logic element that collects the Conga Sign Transaction records that should be cancelled
 - e. An Assignment Logic element that adds the record of the current iteration Conga Sign Transaction record to a Collection Variable Resource

- f. An ApexAction Interaction element that calls the Sub Flow aptx_cancelTransactionsInvocable Apex Class (Cancel Conga Sign Transactions) and sets the Input Variables:
- In_CongaSignTransactionRecords: [The Collection Variable containing Conga Sign Transaction Records]
 - In_SendCancellationEmails: [GlobalConstant or some other variable containing a boolean; True (send recipients cancellation notice), False (do not send)]



- g. An optional notification

Common Sub Flow - Send List of Transaction Ids to the Conga Sign Cancel Transaction Invocable

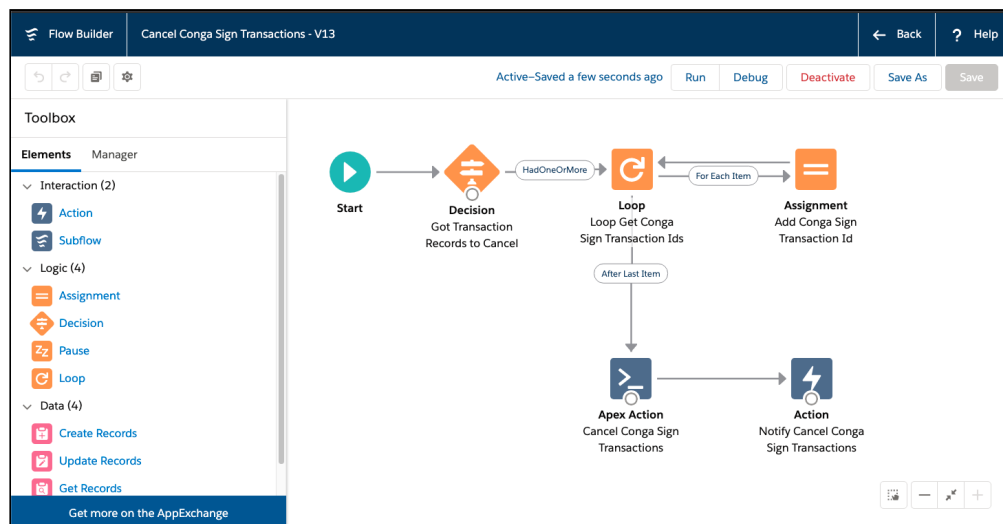
i This flow is used as a sub flow in each of the example use cases below. Follow Salesforce [Best Practices for building Flows](#).

1. Input a list of Conga Sign Transaction records.

i This list is gathered by other use case Flows as an input into this flow.

2. Add a Decision Logic element to check that the input contains records.
 - a. Add an Outcome that there are one or more records from the input and it is not null.
3. Connect Start to the Decision.
4. Add a Loop logic element to iterate over the Conga Sign Transactions.
5. Connect the Decision to the Loop using the appropriate Outcome.
6. Add an Assignment Logic element and add each Id to a Text collection variable.

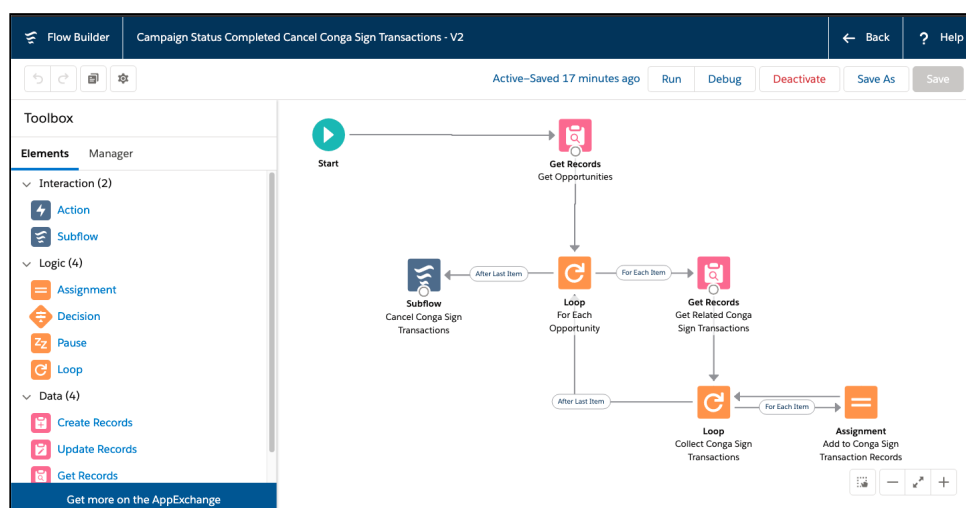
7. Connect the Loop to the Assignment with For Each Item and return from the Assignment to the Loop.
8. Add an Apex Action Interaction element that calls the Conga Sign Cancel Transactions Invocable method.
 - a. Configure the collected Conga Sign Transaction Id's as the Input Variable "Conga Sign Transaction Record Ids".
 - b. Configure either "True" or "False" for the as the Input Variable "Send Cancellation Emails". This will send cancellation emails to each recipient when set to true.
 - c. (Optional) Configure the output variables to process return value and messages. (Shown As "Notify Cancel Conga Sign Transactions").
9. Connect the Loop to the Apex Action with After Last Item.
10. Save and Activate the Flow.



Example: Cancel Any Incomplete Conga Sign Transactions When a Campaign Status Changes to Complete

1. Create a new Process Builder.
2. Select Campaign as the object.
3. Select "when a record is created or edited" for Start the process.
4. Add Criteria:
 - a. Criteria Name: Campaign Price Changes
 - b. Criteria for Executing Actions:
 - i. Field: [Campaign].[Status]

- ii. Operator: Equals
 - iii. Type: Picklist
 - iv. Value: Completed
- c. Conditions: All of the conditions are met (AND)
- 5. Add Immediate Action:
 - a. Action Type: Flows
 - b. Action Name: Campaign Status Completed Cancel Conga Sign Transactions
 - c. Flow: Campaign Status Completed Cancel Conga Sign Transactions
 - i. Create the Flow using Common Steps, if not already created.
 1. Get all Opportunities related to the Campaign.
 2. Get all Conga Sign Transactions from the Opportunities related to the Campaign.
 3. Collect the Conga Sign Transaction Ids.
 4. Call the Cancel Conga Sign Transaction Invocable with the list of Conga Sign Transaction Ids.

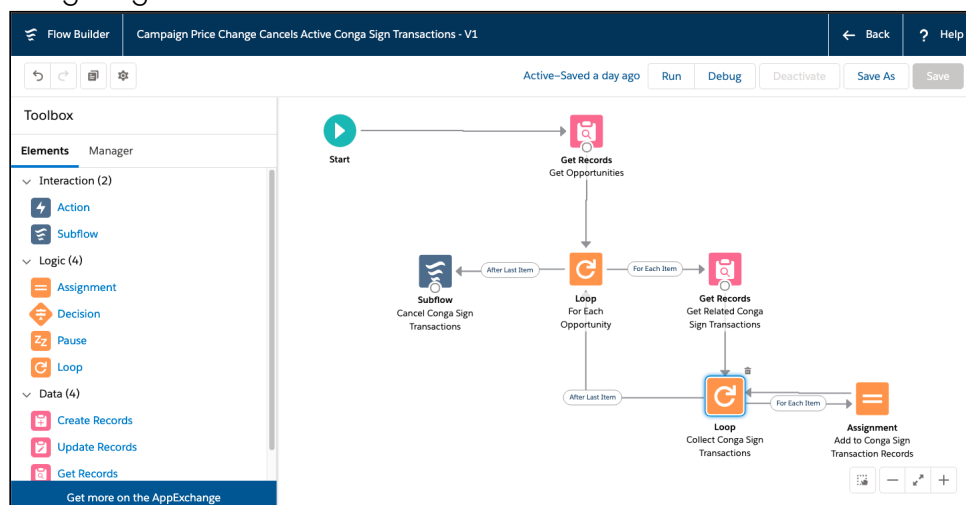


- d. Set Flow Variables.
 - i. Flow Variable: In_CampaignRecord (the input variable you used in your flow)
 - ii. Type: Field Reference
 - iii. Value: [Campaign] (select the campaign record that started your process)
- e. Save and activate the Process Builder.

Example: Cancel Any Incomplete Conga Sign Transactions When a Campaign Offering Price Change Occurs

1. Create a new Process Builder.
2. Select Campaign as the object.
3. Select when a record is created or edited for Start the process.
4. Add Criteria:
 - a. Criteria Name: Campaign Price Changes
 - b. Criteria for Executing Actions:
 - i. Field: [Campaign].[OfferingPrice] (Use the field you monitor for price changes)
 - ii. Operator: Is Changed
 - iii. Type: Boolean
 - iv. Value: True
 - c. Conditions: All of the conditions are met (AND)
5. Add Immediate Action:
 - a. Action Type: Flows
 - b. Action Name: Campaign Price Change Cancel Conga Sign Transactions
 - c. Flow: Campaign Price Change Cancel Conga Sign Transactions
 - i. Create the Flow using Common Steps, if not already created.
 1. Get all Opportunities related to the Campaign.
 2. Get all active Conga Sign Transactions from Opportunities related to the Campaign.
 3. Collect the Conga Sign Transaction Ids.

4. Call the Cancel Conga Sign Transaction Invocable with the of Conga Sign Transaction Ids.



- d. Set Flow Variables:

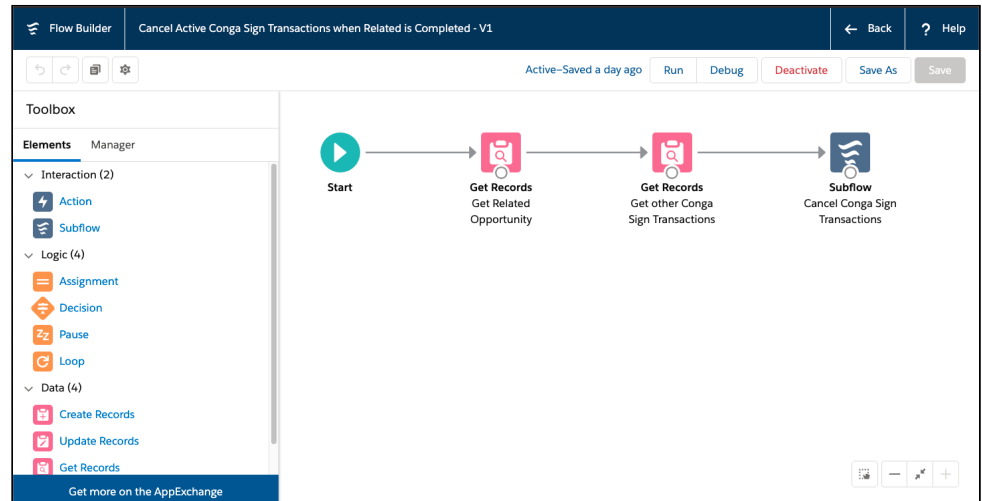
- i. Flow Variable: In_CampaignRecord (the input variable you used in your flow)
- ii. Type: Field Reference
- iii. Value: [Campaign] (select the campaign record that started your process)

6. Save and activate the Process Builder.

Example: Cancel Related Conga Sign Transactions When One Conga Sign Transaction Is Completed

1. Create a new Process Builder.
2. Select Conga Sign Transaction as the object.
3. Select when a record is created or edited for Start the process.
4. Add Criteria:
 - a. Criteria Name: Conga Sign Transaction Completed
 - b. Criteria for Executing Actions:
 - i. Field: [Transaction__c].[Status__c]
 - ii. Operator: Equals
 - iii. Type: Picklist
 - iv. Value: COMPLETE
 - c. Conditions: All of the conditions are met (AND)
5. Add Immediate Action:
 - a. Action Type: Flows
 - b. Action Name: Cancel Related Conga Sign Transactions
 - c. Flow: Cancel Related Conga Sign Transactions
 - i. Create the flow using Common Steps if not already created.

1. Get all Active Conga Sign Transactions related to the Opportunity.
2. Collect the Active Conga Sign Transaction Ids.
3. Call the Cancel Conga Sign Transaction Invocable with the list of Active Conga Sign Transaction Ids.



d. Set Flow Variables:

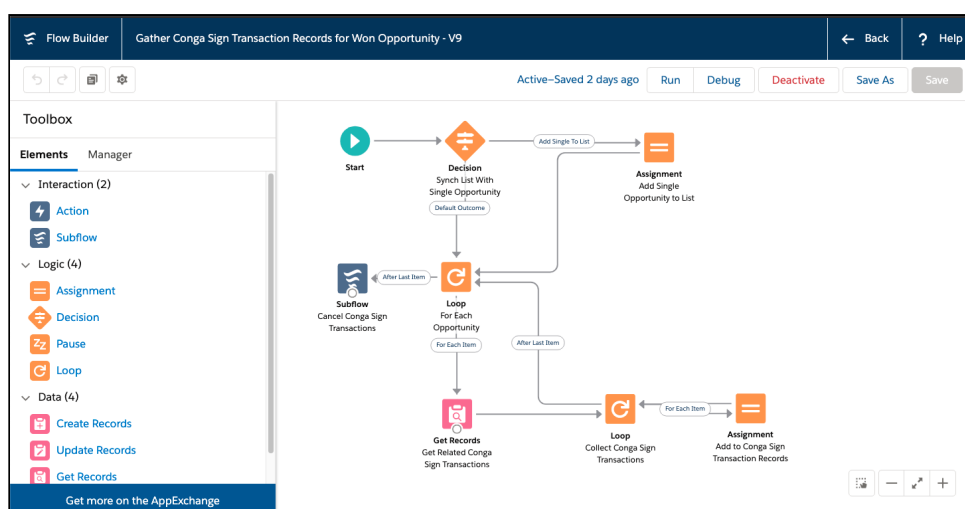
- i. Flow Variable: In_CongaSignTransactionRecord (the input variable you used in your flow)
- ii. Type: Field Reference
- iii. Value: [Transaction__c] (select the Conga Sign Transaction record that started your process)

6. Save and activate the Process Builder.

Example: Cancel Any Incomplete Conga Sign Transactions When an Opportunity Is Won

1. Create a new Process Builder.
2. Select Conga Sign Transaction as the object.
3. Select when a record is created or edited for Start the process.
4. Add Criteria:
 - a. Criteria Name: Is Opp Won
 - b. Criteria for Executing Actions
 - i. Field: [Opportunity].[Won]
 - ii. Operator: Equals
 - iii. Type: Boolean
 - iv. Value: True

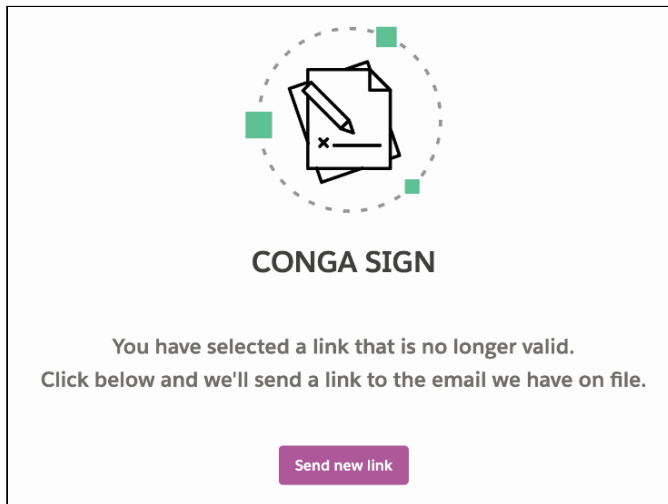
- c. Conditions: All of the conditions are met (AND)
- 5. Add Immediate Action:
 - a. Action Type: Flows
 - b. Action Name: Gather Active Conga Sign Transactions
 - c. Flow: Gather Conga Sign Transaction Records for Won Opportunity
 - i. Create the flow using Common Steps if not already created
 - 1. Get all Active Conga Sign Transactions related to the Opportunity.
 - 2. Collect the Active Conga Sign Transaction Ids.
 - 3. Call the Cancel Conga Sign Transaction Invocable with the list of Active Conga Sign Transaction Ids.




- d. Set Flow Variables
 - i. Flow Variable: In_AnOpportunity (the input variable you used in your flow)
 - ii. Type: Field Reference
 - iii. Value: [Opportunity] (select the Opportunity record that started your process)
- 6. Save and activate the Process Builder.

Conga Sign Link Expiration

To better secure documents, Conga Sign automatically expires signing links after 5 days. Recipients that access an expired link can click **Send new link** to send a new signing link to their email address.



 The Signing user interface times out after 1 hour. After 1 hour, Recipients must refresh the browser to complete their signing portion.

Conga Sign Recipient Object

The Conga Sign Recipient object is accessible from the [Audit Trail](#) and provides key details for Recipients related to a specific Conga Sign Transaction.

Salesforce users can access Recipients related to a Conga Sign Transaction by using the Recipients Related List on a Conga Sign Transaction record. Recipient records are also accessible from the Recipients object tab.

Recipient Fields

Recipient object records contain the following fields:

- Recipient Name
 - An auto-assigned record name
 - This is the name of the Recipient record in Salesforce. It is not the name of the actual Recipient
- Signer Index
 - The assigned signer order for a Recipient
- UUID
- Recipient Status
- Conga Sign Transaction
 - Lookup field to the related Conga Sign Transaction

- Name
- Initials
- Email
- Title
- Company
- Signed On
 - The date the recipient signed the Conga Sign document
- Decline Reason
- Created By
 - Date and time when the Recipient is added to a Transaction as a Recipient
- Last Modified By
 - Date and time the Recipient record is last updated


Recipient Status

The Recipient Status field displays the current status of a Recipient related to a Transaction by displaying the field values below:

- COMPLETE
 - The Recipient completed their portion of the Conga Sign Transaction.
- PENDING
 - An invitation to sign has not yet been sent to this Recipient.
- SENT
 - The Conga Sign Transaction email is sent to the Recipient although they have not viewed the Conga Sign document.
- VIEWED
 - The Recipient viewed the Conga Sign document but has not completed their portion of the Conga Sign Transaction.
- DECLINED
 - The Recipient declined the Conga Sign Transaction.
- CANCELLED
 - The Salesforce user (who sent the Conga Sign Transaction) canceled the Conga Sign Transaction.
- EXPIRED
 - The Recipient did not complete the Conga Sign Transaction before the defined expiration date.
- AUTHENTICATION FAILED
 - The Recipient did not successfully complete SMS Authentication.


Reassigning Signers as a Recipient

Conga Sign offers recipients the flexibility to reassign a signer's designated signature tags to a different recipient after the Conga Transaction is sent. Reassigning signers is useful in scenarios where the designated signer would like to pass signature responsibilities to a different contact within their organization.

 The process for reassigning signers as a recipient is different than reassigning signers as a Conga Sign user. For more information on reassigning signers as a Conga Sign user in Salesforce, see [Reassigning Signers from Salesforce](#).

To reassign a designated signer's Conga Sign tags to a different signer

1. Open the Conga Sign Transaction email as the initial designated signer.
2. Click the Click Here hyperlink next **To reassign the signer** prompt.
3. Fill out the **First Name**, **Last Name**, and **Email Address** fields to identify the new reassigned signer.
 - a. (Optional) Fill out the Optional Message field with an explanation of the document or signing responsibilities.
4. Click the **Send Now** button. A prompt appears letting the user know that the Signer Reassignment is complete.

 A signer can only reassign their signature responsibilities once in a Conga Sign Transaction. Once reassigned, that specific set of signature responsibilities cannot be reassigned again.

After an initial designated signer reassigns signature responsibilities to a new signer, the new signer receives a new Conga Sign Transaction email prompting them to complete the signature tags.


The Conga Sign user who sent the transaction and the initial signer both receive confirmation emails detailing the successful reassignment of the signature responsibilities. The new reassigned signer also receives all standard email notifications related to the Transaction moving forward.

New reassigned signers are added and accounted for in the Conga Sign Transaction record and the related Audit Trail. If a signer reassigns their signature responsibilities to a new signer, their status is marked as DECLINED on the Conga Sign Transaction record. The new reassigned signer has a status of Recipient status of SENT until they complete the assigned Conga Sign tags.

Reassigning Signers from Salesforce


Conga Sign offers users in Salesforce the option to reassign a signer's designated signature responsibilities to a different recipient after the Conga Transaction is sent. Reassigning signers is useful in scenarios where a Conga Sign user wants to designate signature responsibilities to a different person within their Salesforce org.

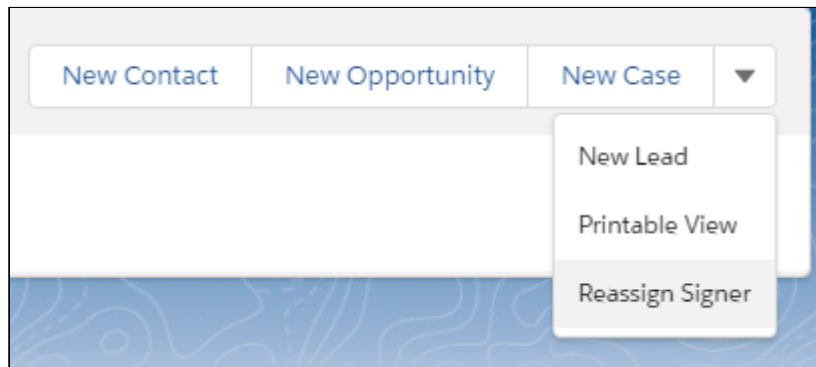
 If you are not able to see the Reassign Signer button, contact your administrator. The administrator must add the Reassign Signer button. For more information, refer to [Adding Reassign Signer Button](#).

 The process for reassigning signers from Salesforce is different than reassigning a signer as a recipient. For more information on reassigning signers as a recipient, see [Reassigning Signers as a Recipient](#).


To reassign a signer from Salesforce

1. Navigate to the specific **Conga Sign Transaction** record in Salesforce.
2. Locate the related **Recipient** record on the Conga Sign Transaction record and click the **Recipient Name** hyperlink.
3. Click **Reassign Signer**.

 In Salesforce Lightning, you must click the drop-down option in the top-right corner to select the Reassign Signer action.




4. Complete the reassignment fields pertaining to the new signer:
 - a. First Name
 - b. Last Name
 - c. Email Address
 - d. (Optional) Language
 - e. (Optional) Message
5. Click **Send Now**.

 Conga Sign users can only reassign a recipient's signing responsibilities one time in a Conga Sign Transaction. Once reassigned, that specific set of signing responsibilities cannot be reassigned again.

After a Conga Sign user in Salesforce reassigns signature responsibilities to a new signer, the new reassigned signer receives a Conga Sign Transaction email prompting them to complete the signature tags.

The Conga Sign user who sent the transaction, and the initial signer, both receive confirmation emails detailing the successful reassignment of the signature responsibilities. The new reassigned signer also receives all standard email notifications related to the transaction moving forward.


New reassigned signers are accounted for in the Conga Sign Transaction record and the related Audit Trail. If a signer reassigns their designated signature responsibilities to a new signer, their Recipient status is marked as DECLINED on the Conga Sign Transaction record. The new reassigned signer has a status of Recipient status of SENT until they complete the assigned Conga Sign tags.

 Users cannot reassign signers from Salesforce in Conga Sign Transactions created before the Conga Sign 1.43.0 release on May 6th, 2020.

Audit Trail

Every Conga Sign transaction contains a unique audit trail providing detailed information about events related to that specific transaction. The audit trail history contains information including:

- The document's filename
- The transaction status (Sent, Complete, Canceled)
- A unique transaction ID number
- The chronological record of events, including:
 - When a transaction was created
 - When the document was sent for signatures and who sent it
 - (If SMS Authentication is enabled) When an SMS code is requested and sent
 - Who viewed the document

 Signers appear in the Audit Trail as "Invitation email sent to..."
CC recipients appear in the Audit Trail as "Invitation to view sent to..."

- Who signed the document and when did they sign it
- If an attachment was uploaded and who uploaded it
- If the transaction was canceled and when it was canceled
- If the transaction was Sent on Behalf of another User
- The unique IP Address for each event
- When a recipient accesses the download link for final document delivery
- When a download link expires and a recipient requests a new download link

Access the Audit Trail for a Transaction

When a document is sent or signed using Conga Sign, an email confirmation containing a link to the audit trail is automatically sent to all recipients and the sender. Clicking this link will take users to a unique link containing that specific transaction's audit history.

You may also access a document's audit trail by clicking on any of the signature blocks in the PDF output.

Signers do not need to be Salesforce users to see audit history.

CONGA SIGN Transaction Audit		
Transaction ID 4lq8b1f2hpcau7urn9rpee4he4g87wftelo2unijm3kuydw01p		
File name Basic Nondisclosure Agreement.docx.pdf		
Transaction status COMPLETE		
File hash d123ace741dfe89def486f47d17bbeb44a8ec36b6731102801279e564cee4946742512c9ab4eeabc64bd6a2c98d447eae12d865c2a712fe079fd359dc6d7c93		
Completed on 1/30/2018		
	IP ADDRESS	DATE
+ Transaction was created by Marianne Sheldon <marianne.sheldon@getconga.com>.	13.108.254.8, 10.152.95.149	01/30/2018 18:59:55 UTC
✉ Transaction was sent by Marianne Sheldon <marianne.sheldon@getconga.com>.	204.132.216.210, 10.152.95.149	01/30/2018 19:02:42 UTC
👁 Transaction was viewed by Jane Doe.	204.132.216.210, 10.152.94.6	01/30/2018 19:22:52 UTC
✓ Transaction was signed by Jane Doe.	204.132.216.210, 10.152.95.149	01/30/2018 19:39:28 UTC
📄 Final document was downloaded by Jane Doe.	204.132.216.210, 10.152.94.6	01/30/2018 19:41:11 UTC

⚠ Every Conga Sign transaction contains a unique audit trail providing detailed information about events related to that specific transaction.

SMS Authentication

SMS Authentication is an optional recipient setting that adds an additional layer of identity assurance to Conga Sign by requiring signers to enter a one-time passcode delivered through SMS, prior to viewing a Conga Sign document. SMS Authentication is currently available for Salesforce integration only and must be purchased from Conga as a separate feature.

SMS Authentication Setup

Administrators must setup SMS Authentication prior to using the feature in Conga Sign Transactions.

To setup SMS Authentication

1. Click the **Conga Sign Setup** tab.
2. Under the **Org Configuration** section of **Conga Sign Setup**, change the **SMS Authentication** toggle from **Disabled** to **Enabled**.

- i** To use SMS Authentication, Conga Sign users must have a minimum of read access to the fields below on the Contact, Lead, and User objects.
- Contact: MobilePhone and MailingCountry
 - Lead: MobilePhone and Country
 - User: MobilePhone and Country

Enable SMS Authentication for Recipients

SMS Authentication is enabled for individual recipients on the Create Transaction screen. Conga Sign users must first add a Recipient to the transaction before enabling SMS Authentication. SMS Authentication is available for Recipients with a role of Signer, In-Person Signer, and CC.

To enable SMS Authentication for a Recipient

1. Navigate to the **Create Transaction** screen.
2. Add a recipient to the transaction.
3. Change the Authentication field's value from **Standard** to **SMS**.
4. Confirm or enter the mobile phone number of the recipient.

Add Recipients
You must select an existing Contact, Lead, or User. Email addresses are not allowed.

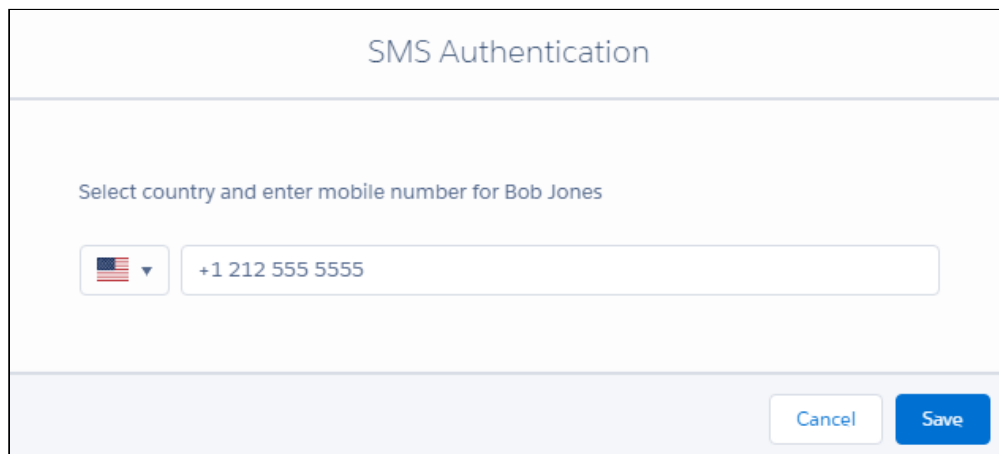
Search by Name or Account

☐ Signing Order * No signing order specified. All recipients can sign simultaneously.

NAME	EMAIL	COMPANY	TITLE	ROLE	LANGUAGE	AUTHENTICATION
Bob Jones	jones.bob@gmail.com			Signer	English	Standard


Standard (checked)
SMS

- i** The Recipient's phone number is automatically pulled from the Contact, Lead, or User record's standard Mobile field. If no value is entered in the Mobile field, users must enter a phone number with the appropriate country code when setting the Authentication field to SMS. A country code is required for SMS Authentication. Use the dropdown field with the flag icon to select the corresponding country code if the mobile phone number does not yet include a country code.



SMS Authentication

Select country and enter mobile number for Bob Jones

 +1 212 555 5555

Cancel Save


5. Click **Save**.
6. Click **Send Now** or **Preview and Tag** to send the transaction.

Completing SMS Authentication as a Recipient

Once the transaction is sent to a Recipient with SMS Authentication enabled, Recipients must follow the steps below to complete SMS Authentication.

To complete SMS authentication

1. Open the initial Conga Sign email and click **Begin Signing**.
 - Clicking the button redirects the recipient to the **SMS Authentication** page.
2. On the **SMS Authentication** page, verify that the phone number is correct and then click **Send Code**.
 - Clicking the button sends the SMS authentication code to the user's mobile phone.
 - The code is sent typically sent from a 5-digit, 6-digit, or a 10-digit phone number.

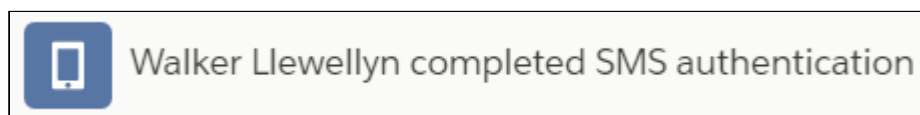
 The Recipient cannot modify the phone number that the code is sent to. If the phone number is incorrect, the Recipient must contact the Sender.

3. Enter the 6-digit code in the Authentication Code field and click **Submit Code**. Upon entering the correct code, Conga Sign redirects the recipient to begin signing the Conga Sign document.

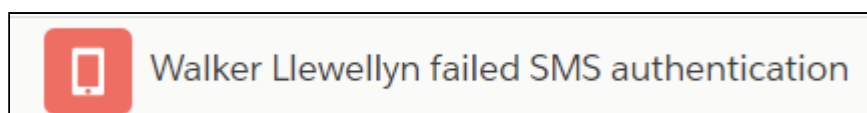
⚠ Signers have three attempts to complete SMS Authentication. The transaction is canceled with a status of AUTHENTICATION FAILED after the third incorrect attempt. If the Recipient completes SMS Authentication successfully but does not complete signing, the Recipient must complete SMS Authentication again once they revisit the Conga Sign document. If the Recipient completes their portion of the Conga Sign transaction and revisits the document, they do not have to complete SMS Authentication again.

Tracking SMS Authentication in the Audit Trail

SMS Authentication is displayed in a Conga Sign Transaction's Audit Trail to show if a Recipient completes or fails SMS Authentication. Successful SMS Authentication results in a transaction Status of COMPLETE. An event is also logged on the Audit Trail displaying [Recipient] completed SMS Authentication.



Failed SMS Authentication results in a transaction Status of AUTHENTICATION_FAILED. An event is also logged on the Audit Trail displaying [User] failed SMS Authentication.



Reassigning Signers with SMS Authentication


A Recipient with SMS Authentication specified can reassign signature responsibilities to another Recipient. In this scenario, the initial Recipient must complete SMS Authentication prior to reassigning to a new Recipient. Additionally, the initial signer must provide the new Recipient's phone number. The new Recipient must also complete SMS Authentication.

Frequently Asked Questions

Where does Conga Sign pull the recipient's phone number used for SMS Authentication? Conga Sign automatically uses the phone number listed in the standard Mobile field on Contact, Lead, and User records.

What is the best practice for storing phone numbers used for SMS Authentication? It is best practice to add phone numbers with the country code included in the standard Mobile field on Contact, Lead, and User records. For example, a phone number from the US is stored as +17325671098 and a phone number from the UK is stored as +447921123456.

How does Conga Sign choose a country code if no country code is present in the Mobile field? If a country code is not present in the Mobile field, Conga Sign automatically uses the country code corresponding to the Recipient's country. If the Recipient's country is not present, Conga Sign uses the country code corresponding to the Sender's (the running User) country. If the Sender's country is not present, Conga Sign uses the country code corresponding to the Org's country.

 If a country is not entered correctly, Conga Sign requires the user to enter a country code on the SMS Authentication page.

Supported Languages in Conga Sign

Conga Sign supports English (default language), French, German, Spanish (Spain), Portuguese (Brazil), Portuguese (Portugal), Dutch (Netherlands), Greek, Italian, Croatian, Polish, Ukrainian, Romanian, Georgian, Russian, Serbian (Latin), Japanese and Chinese(Simplified), Korean, Spanish - Latin American, Armenian, Bulgarian, Bosnian, British English, Norwegian, Slovenian, Turkish in the Signing user interface and [Audit Trail](#).

 Custom text, such as a Custom Email Message or Custom Tag labels, is not translated to selected languages.

Selecting a language for a Signer sets the text in the Conga Sign emails and Signing interface to the selected language. Choosing a different language on the Audit Trail displays all Audit Trail fields and events in the selected language.

For more information on selecting a language, see the documentation linked below.

- [Selecting a Language for a Recipient](#)
- [Selecting a Language as a Recipient](#)
- [Selecting a Language for the Audit Trail](#)

Selecting a Language for a Recipient

To select a language for a Recipient

1. Add a new Recipient or navigate to an existing Recipient on the **Create Transaction** page.
2. Use the **Language** dropdown field to select a language for the Recipient.
 - a. Repeat this step for all desired Recipients.
3. Click **Send Now** or **Preview and Tag** to proceed and send the Transaction.

Selecting a Language as a Recipient

Recipients also have the option to change the language upon receiving the Conga Sign Transaction.

To select a different language as a Recipient

1. Click **Begin Signing** in the Conga Sign Transaction email.
2. Confirm the **Full Name** and **Initials**, then click **Agree and Start Signing**.
3. Use the **Language** dropdown field on the top right-hand corner of the Signing user interface to select a new language. Upon selecting a different language, the Signing user interface and tags appear in the selected language.

Selecting a Language for the Audit Trail

To select a language for the Audit Trail

1. Navigate to a **Conga Sign Transaction** record in Salesforce.
2. Click **View Audit**.
3. Use the **Language** dropdown field on the top right on the top right-hand corner of the Audit Trail to select a language. The Audit Trail translates all fields and events to the selected language.

Troubleshooting Topics

See the troubleshooting topics for more information.

- [Browser tabbed navigation issues](#)
- [Transactions cannot be linked to Opportunity Product](#)
- [Why aren't My Documents displayed on the Send with Conga Sign Screen?](#)
- [Write Back tags FAQs and Tips](#)

Browser tabbed navigation issues

Question or Problem

When you send a document to be signed through the Send with Conga Sign button, you are redirected to a different tab instead of the Conga Sign tab.

Causes or Possible Causes

Tab behavior is browser-specific. The Google Chrome browser uses the default settings to navigate through tabs.

Workarounds

How Salesforce and Conga Sign open tabs are browser-specific behavior. Ensure you have the proper settings in your browser to ensure your tabbed navigation setting is set.

Transactions cannot be linked to Opportunity Product

Issue

Salesforce does not allow the creation of a lookup related to the Opportunity Product record. Therefore, Conga Sign cannot link transactions to Opportunity Products records.

Workaround

Attach the transactions to the parent opportunity. For example, you might see 5 transaction and 5 products in the opportunity related lists.

Why aren't My Documents displayed on the Send with Conga Sign Screen?

Issue

Documents don't display when trying to send with Conga Sign.

Workaround

Follow the steps to use Attachments instead of Files.

To configure Salesforce Files

1. In the Salesforce File Configuration box, click Configure Files Settings. The Salesforce Files Settings screen appears.
2. Select the Enable Files Sync and Files uploaded to the Attachments related list... checkboxes.

Salesforce Files Settings

Use this page to modify Salesforce Files settings for your organization.

- ☒ Enable Files Sync
- ☒ Files uploaded to the Attachments related list on records are uploaded as Salesforce Files, not as attachments [i](#)

[Edit](#)

You have successfully configured Conga Sign. Now when you visit an object record on which you utilized Conga Sign, you should see the Send with Conga Sign button and a new Conga Sign Transactions-related list.

Write Back tags FAQs and Tips

Frequently Asked Questions and tips

Conga Sign Write Back functionality uses Conga Sign tags to configure and map standard and custom Salesforce fields in documents. The information added by signers is sent directly to Salesforce and records are updated.

Can I make a write back tag optional?

Yes. From the Write Back Configuration screen, each write back field has a set of properties and can be made optional or required for the signer.

Does write back support picklists?

Yes, write back supports picklist fields. The supported field types are text, number, email, phone, checkbox, picklists, and URL.

Does write back support checkboxes?

Yes, write back supports checkbox fields. The supported field types are text, number, email, phone, checkbox, picklists, and URL.

Can I assign a single write back tag to multiple signers?

No. To eliminate potential conflicts with multiple signers responding to the same request for information, each write back tag can only be assigned to one signer. Determine who the best person is to provide the necessary information, and assign the tag to them.

Does Conga Sign Write Back support any languages other than English?

No. Conga Sign only supports English in its current state.

Conga Sign for REST API Developers

This section describes the REST APIs provided by Conga Sign.

Topic	Description
What's Covered	This section walks an API user through the list of REST APIs provided by Conga.
Primary Audience	REST API developers.
IT Environment	Refer to the latest Conga Sign Release Notes for information on System Requirements and Supported Platforms.
Updates	For a comprehensive list of updates to this guide for each release, see the What's New in Conga Sign Documentation topic.

This guide describes the following APIs:

- [Conga Sign REST APIs](#)
- [Obtain the JWT access token](#)
- [Authentication Tokens](#)
- [Using the Signer Experience](#)
- [Storing Signing Session Data](#)
- [Supported Languages](#)
 - [Configuring Languages](#)
- [Creating and Sending a Transaction](#)
 - [Checking Transaction Status and Downloading Documents](#)
 - [Changing a Recipient](#)
 - [Adding Signatures](#)
 - [Managing Signers' Attachments](#)
 - [Delivering Signed Documents](#)
 - [Creating a Signer Workflow](#)
 - [Downloading Signed Documents and Audit Trails](#)
 - [Retrieving Audit Trails](#)
 - [Adding Fields](#)
 - [Configuring Optional Signatures](#)
- [Adding, Updating, Removing Signers](#)

- [Document Extraction](#)
- [Uploading & Deleting Documents](#)
- [Customizing Invitation Emails](#)
- [Injecting Field Values](#)
 - [Form Building Fields](#)
- [Supported Fonts](#)
- [Managing Groups](#)
- [Authenticating Signers](#)
- [Enforcing Signature Capture](#)
- [Custom Transaction Data](#)
- [Custom Document Data](#)
- [Mobile Signing Ceremony](#)
- [Position Extraction](#)
- [Signer Experience Settings](#)
- [Text Anchors](#)

Before using Conga Sign API, you must be familiar with the following:

- Basic knowledge of Salesforce
- Basic knowledge of REST APIs
- Salesforce and Conga terms and definitions

Minimum Software Requirements

What follows is the list of environments supported by Conga Sign. Its dependencies are generally based on the browser, not on lower-level components.

 All of the following requirements are subject to change.

Operating Systems

- Microsoft Windows 8, 8.1, 10 (except external touch-screen devices)
- Mac OS X

Browsers

- Edge (latest version, plus the two previous versions)

 Microsoft Edge in IE mode and Microsoft Edge in IE Compatibility mode are no longer supported by the New User Experience, including the Signer Experience.

- Chrome (latest version, plus the two previous versions)
- Firefox (latest version, plus the two previous versions)

- Safari (latest version, plus the two previous versions. Requires enabling)

❗ The Signer Experience is not supported in Safari as iFraming is not a supported feature in Safari browsers.

❗ Safari macOS browsers block the Preview and Tag features unless third-party cookies are enabled. To enable third-party cookies, navigate to browser Preferences > Privacy and uncheck the "Prevent Cross-Site Tracking" and "Block All Cookies" options.

ℹ For all browsers, cookies must be enabled.

Mobile Devices

- Android
- Windows
- iOS

❗ Safari iOS browsers block the Preview and Tag features unless third-party cookies are enabled. To enable third-party cookies, navigate to Settings > Safari and disable the "Prevent Cross-Site Tracking" and "Block All Cookies" options.
If the issue persists, navigate to Settings > Safari > Advanced > Experimental Features and enable the "Disable Full 3rd-Party Cookie Blocking" option.
There is no current workaround for Chrome on iOS.

PDF Viewers

Acrobat or similar software may be required to view and print PDF files. Occasionally, a PDF viewer may give different results than those of Adobe Reader.

Conga Sign REST APIs

(Open API documentation is only available to view online)

Obtain the JWT access token

To request an access token, send a Post request containing your Client Id and Client Secret to the Conga Sign authentication service.

- To get the access token, the request URL is <https://login.congacloud.com/prod/api/v1/auth/connect/token>

POST /api/v1/auth/connect/token

Host: <https://login.congacloud.com>

Content-Type: application/x-www-form-urlencoded

Body:

client_id={{client_id}}

client_secret={{client_secret}}

scope=sign

grant_type=client_credentials

If successful, an access token will be returned in the response body.

Example Response

Sample successful response body

```
1  {
2    "access_token":
3    "eyJraWQiOiJmMTJlaE02ZEpmMVhja20wR2hXMUVJMGxPNTQ2dEppXC9NXC9T...",
4    "expires_in": 3600
5    "token_type": "Bearer",
6    "scope": "sign"
7  }
```

The access token contains the following fields:

Name	Description
access_token	The value of the access token. This value will be added to the Authorization header of all Conga Sign API calls.
expires_in	The expiration date of your access token. When your access token expires, your application will have to request a new access token.
scope	The resources you are requesting access to. This should be sign .

Name	Description
<code>grant_type</code>	The method of authentication that is used. Using Client Id and Client Secret, this grant type should be <code>client_credentials</code> .

 The access token granted by JWT Grant expires at the given time, and no refresh token is provided. After the token expires, you must reauthenticate for a new access token.


Otherwise, the operation may return one of the following errors:

Response code and body	Description
400 { <code>"error": "ERROR_AUTHENTICATION_HEADER"</code> }	Invalid headers. Example header: Authorization: Basic <ENCODED_CREDENTIALS>
400 { <code>"error": "ERROR_AUTHENTICATION_NOT_VALID"</code> }	The request could not be authorized. Make sure you provided the correct credential.
403 { <code>"error": "ERROR_AUTHENTICATION_CLIENT_DISABLED"</code> }	The request has been forbidden because access is disabled. Contact Conga's customer service for further support.

Authentication Tokens

An authentication token is used to obtain a valid session for a particular user of the system. This topic introduces the following types of authentication tokens:

- `userAuthToken`
- `senderAuthToken`
- `signerAuthToken`
- `singleUseSignerAuthToken`

 With the exception of `signerAuthToken` these tokens are all single-use. The default expiry time for all these tokens is 30 minutes.

User Authentication Tokens

A user authentication token is a token that can be used to obtain a session for a user with complete access to the account. The following code will create a user authentication token:

HTTP Request

```
POST /api/cs-authenticationTokens/user
```

HTTP Headers

```
Accept: application/json
Authorization: Bearer access_token
```

For a complete description of each field, see the **Request Payload** table below.

Property	Type	Editable	Required	Default	Sample Values
packagelid	string	No	No	n/a	5vjLRY5MWrDJ6MzRAEyCK0y5IH0=
signature	string	No	No	n/a	8b734331-bc5b-4843-9784-d4ece4b7dc22
value	string	No	No	n/a	ZDNmMDNiNGUtNGYxOC00YWZiLTkwMmUtNWE5YmIwZTRjZDg1

Response Payload

```
{  "value": "MjY0MjQ4MzgtMTJlOS00MzhjLTgzODMtMzJmMGNiZTg3ODBl" }
```

Sender Authentication Tokens

A sender authentication token can be used to obtain a session for a sender with access only to a specific package.

HTTP Request

```
POST /api/cs-authenticationTokens/sender
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{  "packageId": "5vjLRY5MWrDJ6MzRAEyCK0y5IH0="  }
```

Response Payload

```
{  "value": "MjY0MjQ4MzgtMTJlOS00MzhjLTgzODMtMzJmMGNiZTg3ODBl"  }
```

Signer Authentication Tokens

A signer authentication token can be used to obtain a session for a signer with access to the Signer Experience.

HTTP Request

```
POST /api/cs-authenticationTokens/signer/multiUse
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{  "packageId": "5vjLRY5MWrDJ6MzRAEyCK0y5IH0=",  "signerId": "8b734331-bc5b-4843-9784-d4ece4b7dc22"  }
```

Response Payload

```
{  "packageId":"5vjLRY5MWrDJ6MzRAEyCK0y5IH0=",  "signerId":"8b734331-bc5b-4843-9784-d4ece4b7dc22",  "value":"ABCdEFghIJKLMNOpQR00STUvWXYZNoWiKN05MyabCsNtWySWm"  }
```

The signer token above can be used multiple times. You can also create a single-use signer token:

HTTP Request

```
POST /api/cs-authenticationTokens/signer/singleUse
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{  "packageId":"s-wy6PFASBlAKfnLJjcbzoaMyTg=",  "signerId":"signer1@example.com"  }
```

Response Payload

```
{  "packageId":"s-wy6PFASBlAKfnLJjcbzoaMyTg=",  "sessionFields": null,  "signerId":"signer1@example.com",  "value":"ABCdEFghIJKLMNOpQR00STUvWXYZNoWiKN05MyabCsNtWySWm"  }
```

Using the Signer Experience

The ability to sign documents using electronic signature platforms, like Conga Sign, the fast and secure way to get your documents signed anytime, anywhere, and on any device. E-Signature software is a core component of creating that superior customer experience. Signers will receive an email inviting them to the Signer Experience to sign documents in electronic form. The signer can simply open the documents, read the agreement, and click-to-sign from their desktop computer, tablet, or mobile phone. Once your signers complete their signatures, each party can access a copy of the documents as well as a detailed

audit trail.

This section discusses the following topics:

- [Signing Documents as a Transaction Owner](#)
- [Signing Documents as a Signer](#)
- [Uploading Attachments](#)
- [Accessible Signing](#)
- [Changing Signers](#)
- [Declining to Sign](#)
- [Completing Signing](#)
- [Downloading Signed Documents](#)
- [Reviewing Signed Documents](#)

Signing Documents as a Transaction Owner

The Transaction Owner is the first person prompted to sign a transaction's documents unless a Signer Workflow has been created that specifies a different signing order.

To sign documents as a Transaction Owner:

1. Where indicated, click **Sign** or **Initial**.
2. Click **OK** to confirm your signature.
3. Repeat for each of the document's Signature Boxes that you must sign. A progress bar at the top of the document indicates the number of signatures you have completed, as well as the total number of signatures required by this document.
4. If additional documents in the transaction require your signature, continue to sign each document where indicated.

Signing Documents as a Signer

You will be notified that you have documents to sign via an email message, which states that you have been added as a signer to a Sign transaction. That message will include a link to the transaction. The Transaction Owner may have included additional text, which will appear below the link.

To sign documents as a signer:

1. In the email message you receive, click **Go to Documents**.

2. Depending on the settings chosen by the Transaction Owner, you will need to verify yourself as a signer for the transaction. One of the following authentication methods may be used:
 - a. Authentication by Email is the simplest form of authentication. It simply requires you to click the link in your invitation email.
 - b. Authentication by Q&A requires you to answer one or more questions. Correctly answer all questions, and then click Login.
 - c. Authentication by SMS requires you to enter a seven-digit SMS code that you received on your mobile phone. Enter the code, and click Login. This SMS code is sent when you click the link in your invitation email. If you have not received a code, use the **Click Here** link.
3. Read and **Accept** the Electronic Disclosure and Signatures Consent document. Sign automatically includes the Electronic Disclosure and Signatures Consent agreement in all its transactions. All signers must accept the terms of this agreement before they can access transaction documents.

Currently, Sign provides Electronic Disclosure and Signatures Consent agreements in English, French, German, Spanish (Latin American), Greek, Italian, Dutch, Portuguese, Russian, Chinese (Simplified), Japanese and Korean. If you would like to provide an agreement in a different language or do not want this agreement in your transactions, please contact our Support team.
4. Sign all your Signature Boxes within the document. If any other signer has already signed the document, their signatures are visible in the document.
5. Once you have completed signing the document, a confirmation request will appear. Click **Confirm** to confirm your signatures on the current document. If there are more documents for you to sign, they will appear.
6. To verify your signed documents, click **Review Documents**.
7. If you would like to keep copies of the signed documents, click **Download Documents**.
8. Once you have finished, close your browser to end the Signer Experience.

Uploading Attachments

During the Signer Experience, you may be asked to upload additional supporting documents.

1. When prompted, click **Go to Uploads**.
2. In the **Uploads** tab, select the document that you are being asked to include with your signatures.
3. Drag and drop the required documents to the Signer Experience. You can also click **Browse** to search for the documents.
4. Repeat for each document you are required to attach.



- By default, the following file types are supported: PDF, DOC, DOCX, RTF, ODT, JPG, JPEG, PNG, BMP, TXT, TIFF, TIF, GIF, XLS, XLSX.
 - Account owners can define a whitelist of file types that may be uploaded as transaction attachments by signers on the account. This list cannot include file types on the global blacklist.
 - To prevent signers from uploading malicious code, the following file types cannot be uploaded as attachments to a transaction: EXE, DLL, MSI, DMG, SO. This blacklist supersedes all account-level whitelists.
 - Attachments added by signers are immediately scanned for malware. If malware is detected, the system will not upload the infected document or attachment.
- Click **Finish**.

Accessible Signing

To sign accessible PDFs within an accessible transaction, signers must first click the **Enable accessibility mode** button within the Signer Experience.



This button is available only if the transaction is accessible.

Accessibility can be enabled in the Signer Experience in one of the following ways:

- By using the **More Actions** menu.
- By pressing the Tab key. Once pressed, an additional menu appears which will allow you to tab through the following options:
 - **Enable/Disable Accessibility:** Enables or disables accessibility.
 - **Skip to document:** Brings up the document to be signed.

Once accessibility mode is enabled, the page refreshes and presents the transaction's documents in a format that is consumable by screen-reader technology – including navigating all documents and completing the entire signing process.

Changing Signers

The Transaction Owner may permit a signer to delegate their signing responsibilities to another person. If an original signer does this, their delegate can sign on their behalf.

Delegating a New Signer

- ✓ An original signer can sign any Signature Boxes they want before they delegate their remaining Signature Boxes to a new signer.

1. In the pages, top-right corner, click **More Actions**.
2. Click **Reassign Recipient**.
3. In the dialog box that appears, enter your delegate's **Email, First Name, Last Name, Title, and Company**.
4. *Optional*: Type a **Message** for your delegate. Though not required, a message can be useful to tell your delegate what you would like them to do.
5. Click **Next** and select an Authentication Method for the new signer.
6. Click **Reassign**.
7. Click **OK**.

Signing Documents as a New Signer

Sign sends a new signer an email that: (1) identifies the person who has delegated their signing responsibilities to them; (2) states that documents are available for signing. The email also includes a link to those documents.

Perform the procedure above, Signing Documents as a Signer. After a new signer has signed, their signature will appear in all the Signatures Boxes they signed.

Declining to Sign

After a signer has been authenticated, they can decline to sign a transaction.

1. From the **More Actions** drop-down list, click **Decline To Sign**.
2. Specify your reason for declining.
3. Click **Decline**. The transaction sender will be notified of your action, and you will no longer be able to view any document in the transaction.

Completing Signing


When you have signed all your Signature Boxes in a transaction's document, a **Confirm** button appears. Click this button to continue.

Downloading Signed Documents

After all signers have signed a transaction's documents, Sign flags the transaction as Complete. A completed transaction is one that has been securely signed and digitally sealed with a tamper-evident seal. Any attempt to tamper with the documents in the

transaction or their signatures will break the seal. If that happens, the signatures will stop being legally binding.

Sign sends an email to all signers to notify them that the signing is complete. That email includes a link that signers can click to download the transaction with its securely signed documents. The email may also include the documents as attachments, depending on the options selected by the Transaction Owner when they created the transaction.

 If Document Visibility was configured for the transaction, each person can download only those documents that the configuration permits them to view.

Signers are not required to download documents. They remain available for download until the Transaction Owner archives or deletes the transaction. It's nonetheless a good idea to download, so signers will have a copy of the documents for their records.

Signed documents are provided as PDFs. Each PDF includes visual indicators and messages that signers can use when they are Reviewing Signed Documents.


To download signed documents:

1. In the email message you receive, click **Download Completed Documents**.
2. On the page that appears click **Download Documents**.

Reviewing Signed Documents

After you download a transaction, you can review its documents. Each document is a PDF that contains your signatures and the signatures of all other signers. If you view a downloaded PDF using Adobe Reader, you can verify if its signatures are valid. One way of doing so is to click each signature – a popup states if the signature is valid, and provides further details about the signature's status.

You can also review signed documents by clicking the **Download Completed Documents** link in the email message that you received, and then clicking **Review Documents**.

 Signature validation is largely a standard process across PDF viewers. However, occasionally a PDF viewer may give results that differ from those of Adobe Reader. When that occurs, Conga Sign considers the Adobe Reader results to be more definitive.

Storing Signing Session Data

When creating a signing session, extra data can be stored as part of the evidence summary. This is done by passing additional session fields per signer when creating the signing session. These fields are saved in the evidence summary and are stored in the signature details of your signed documents.

If you need a comparison to the basic object creation procedure, or if this is the first time creating a transaction, see [Creating and Sending a Transaction](#).

After you have created your package, create a signer authentication token with the key/value pair session fields:

HTTP Request

```
POST /cs-authenticationTokens/signer/singleUse
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{  "packageId": "x31anVFrwliCsGQXLJoqv0GRZ1w=",  "sessionFields": {  "Transaction ID": "1234567890",  "Login Token": "88888888888888"  },  "signerId": "signer1"  }
```

For the complete description of each field, see the Request Payload Table below.

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
packageId	string	Yes	No	n/a	x31anVFrwliCsGQXLJoqv0GRZ1w=
signerId	string	Yes	No	n/a	signer1
sessionFields	string	Yes	No	n/a	Transaction ID / 1234567890

Response Payload

```
{  "packageId":"x31anVFrwliCsGQXLJoqv0GRZ1w=",  "signerId":"signer1",  "value":"ZDNmMDNiNGUtNGYxOC00YWZiLTkwMmUtNWE5YmIwZTRjZDg1",  "sessionFields": {    "Transaction ID": "1234567890",    "Login Token": "8888888888888888"  },  }
```

Supported Languages

Conga Sign supports the following languages:

Language	Code Value
English	[en]
Français (French)	[fr]
Dansk (Danish)	[da]
Deutsch (German)	[de]
Español (Spanish)	[es]
Ελληνικά (Greek)	[el]
Italiano (Italian)	[it]
Nederlands (Dutch)	[nl]
Português (Portuguese)	[pt]
Русский (Russian)	[ru]
中文简体 (Chinese - Simplified)	[zh-ch]
中文繁體 (Chinese - Traditional)	[zh-tw]
日本語 (Japanese)	[ja]

Language	Code Value
한국어(Korean)	[ko]
اللغة العربية (Arabic)	[ar]

Configuring Languages

The languages that are available during the creation of a transaction or during the Signer Experience can be customized. The languages available for Sign are listed in [Supported Languages](#).

Changing a Language at the Transaction Level

The following request shows you how to build your JSON payload in order to change the language settings at both the transaction level, as well as at the signer level:

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: multipart/form-data
Authorization: Bearer access_token
```

Request Payload

```
-----WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data;
name="file"; filename="testDocumentExtraction.pdf" Content-Type: application/pdf
%PDF-1.5 %µµµµ 1 0 obj <>> endobj.... -----
WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data; name="payload"
{ "status": "DRAFT", "language": "fr", "roles": [ { "id": "Signer1",
"type": "SIGNER", "signers": [ { "id": "Signer1", "firstName": "John",
"lastName": "Smith", "email": "signer1@example.com" } ], "name": "Signer1"
}, { "id": "Signer2", "type": "SIGNER", "signers": [ { "id": "Signer2",
"firstName": "Mary", "lastName": "Doe", "email": "signer2@example.com",
```

```
"language": "en"  } ],  "name": "Signer1"  } ],  "type": "PACKAGE",  "name":
"Example Package"  }  -----WebKitFormBoundary1bN060n7FqP5W04t--
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{  "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI="  }
```

Note that the language is set on a transaction level. The language will be changed for all recipients in the transaction.

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
status	string	Yes	No	DRAFT	DRAFT/SENT/ COMPLETED/ARCHIVED/ DECLINED/OPTED_OUT/ EXPIRED
language	string	Yes	No	en	en/fr/es ...
type	string	Yes	No	PACKAGE	PACKAGE/TEMPLATE/ LAYOUT
name	string	Yes	No	n/a	Example Package
roles					
id	string	Yes	No	n/a	Signer1
type	string	Yes	No	SIGNER	SIGNER/SENDER
name	string	Yes	No	n/a	Signer1
signers					
email	string	Yes	Yes	n/a	signer1@example.com
firstName	string	Yes	Yes	n/a	John

Property	Type	Editable	Required	Default	Sample Values
lastName	string	Yes	Yes	n/a	Smith
id	string	Yes	Yes	n/a	Signer1
language	string	Yes	Yes	en	en/fr/es ...

Creating and Sending a Transaction

This topic will walk you through the process to create and send a package using the Conga Sign API.

In this example a minimum payload is used, where two signers, one document, and one signature per signer are added to the package.

For a complete description of each field and other optional attributes, see the **Request Payload Table** below.

HTTP Request

```
POST https://rls.congacloud.com/v1/sign/cs-packages
```

HTTP Headers

```
Authorization: Bearer access_token
Accept: application/json
Content-Type: multipart/form-data
```

Request Payload

```
-----WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data; name="file";
filename="testDocumentExtraction.pdf" Content-Type: application/pdf %PDF-1.5
%ÂµÂµÂµÂµ 1 0 obj <>> endobj... -----WebKitFormBoundary1bN060n7FqP5W04t Content-
Disposition: form-data; name="payload" { "roles":[ { "id":"Role1", "signers":[ {
"email":"signer1@example.com", "firstName":"1.firstname", "lastName":"1.lastname",
"company":"Conga Sign" } ] }, { "id":"Role2", "signers":[ { "email":"signer2@example.
com", "firstName":"2.firstname", "lastName":"2.lastname", "company":"Conga Sign" } ]
} ], "documents":[ { "approvals":[ { "role":"Role1", "fields":[ { "page":0, "top":100
, "subtype":"FULLNAME", "height":50, "left":100, "width":200, "type":"SIGNATURE" } ]
}, { "role":"Role2", "fields":[ { "page":0, "top":300, "subtype":"FULLNAME", "height":
```

```
50, "left":100, "width":200, "type":"SIGNATURE" } ] } ], "name":"Test Document" } ],
"name":"Example Package", "type":"PACKAGE", "language":"en", "emailMessage":"",
"description":"New Package", "autocomplete":true, "status":"SENT" } -----
WebKitFormBoundary1bN060n7FqP5W04t--
```

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values	Reference
roles						Adding, Updating, Removing Signers
id	string	Yes	No	n/a	Signer1	
name	string	Yes	No	n/a	Signer1	
emailMessage	array	Yes	No	n/a	check reference	Customizing Invitation Emails
reassign	boolean	Yes	No	0	true/false	Changing a Recipient
attachement Requirements	array	Yes	No	n/a	check reference	Managing Signers' Attachments
index	string	Yes	No	0	0/1/2/3	Creating a Signer Workflow
type	string	Yes	No	SIGNER	SIGNER/SENDER	
signers						
id	string	Yes	No	n/a	Signer1	

Property	Type	Editable	Required	Default	Sample Values	Reference
email	string	Yes	Yes	n/a	signer1@example.com	
firstName	string	Yes	Yes	n/a	Mary	
lastName	string	Yes	Yes	n/a	Doe	
company	string	Yes	No	n/a	Acme Inc.	
title	string	Yes	No	null	Managing Director	
group	array	Yes	No	n/a	check reference	Managing Groups
language	string	Yes	No	n/a	en/fr/es	Configure Languages
signature						
textual	string	No	No	null		
handdrawn	string	Yes	No	n/a	AQAAAAAMkGiVM7tmRJzS2c ANoDcyT0ASAABWA=	
delivery	array	Yes	No	n/a	check reference	Deliver Signed Documents
knowledgeBasedAuthentication	array	Yes	No	n/a	check reference	Authenticating Signers
auth	array	Yes	No	n/a	check reference	Authenticating Signers

Property	Type	Editable	Required	Default	Sample Values	Reference
documents						Uploading & Deleting Documents
description	string	Yes	No	n/a	Test document1 description	
id	string	Yes	No	n/a	document1	
data	array	Yes	No	n/a	check reference	Uploading & Deleting Documents
approvals						Adding Signatures
role	string	Yes	Yes	n/a	Signer1	
id	string	Yes	No	n/a	approval1	
optional	boolean	Yes	No	0	true/false	Optional Signature
enforceCaptureSignature	boolean	Yes	No	0	true/false	Enforce Capture Signature
fields	array	Yes	No	n/a	check reference	Adding Fields
name	string	Yes	No	n/a	document 1	
extract	boolean	Yes	No	0	true/false	Document Extraction /Position Extraction

Property	Type	Editable	Required	Default	Sample Values	Reference
extractionTypes	array	Yes	No	n/a	["TEXT_TAGS","ACROFIELDS"]	Document Extraction /Position Extraction
fields	array	Yes	No	n/a	check reference	Field Injection
name	string	Yes	No	n/a	document1	
settings	array	Yes	No	n/a	check reference	Signer Experience Customization
sender						
lastName	string	Yes	No	n/a	Smith	
firstName	string	Yes	No	n/a	John	
email	string	Yes	No	n/a	bobsmith@email.com (who is a sender under your main account)	
status	string	Yes	No	DRAFT	DRAFT/SENT/COMPLETED/ARCHIVED/DECLINED/OPTED_OUT/EXPIRED	
name	string	Yes	No	n/a	Package created from template through REST API	
type	string	Yes	No	PACKAGE	PACKAGE/TEMPLATE/LAYOUT	
description	string	Yes	No	n/a	Package created with the Conga Sign REST API	

Property	Type	Editable	Required	Default	Sample Values	Reference
language	string	Yes	No	en	en/fr/es	Configure Languages
visibility	string	Yes	No	ACCOUNT	ACCOUNT/SENDER	
autoComplete	boolean	Yes	No	1	true/false	
data	array	Yes	No	n/a	check reference	Package Attribute
due	string	Yes	No	null	08-26-17	
emailMessage	string	Yes	No	n/a	This message should be delivered to all signers	

Response Payload

```
{
  "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI="
}
```

How to Create a Package

The first part of the package JSON string is the roles object. Inside this, you can set items like the id, company, first name, last name, email, and name to customize your signer roles. Some of the other notable settings would be email, message, title, and delivery.

```
{
  "roles": [
    {
      "id": "Role1",
```

```

        "signers":[
            {
                "email":"signer1@example.com",
                "firstName":"1.firstname",
                "lastName":"1.lastname",
                "company":"Conga Sign"
            }
        ]
    },

```

The next section of the package JSON string is the documents object. Inside this, you will set items like the name and the approvals (signature blocks).

```

    "documents":[
        {
            "approvals":[
                {
                    "role":"Role1",
                    "fields":[
                        {
                            "page":0,
                            "top":100,
                            "subtype":"FULLNAME",
                            "height":50,
                            "left":100,
                            "width":200,
                            "type":"SIGNATURE"
                        }
                    ]
                }
            ]
        }
    ],

```

In the approvals, the main items to set would be the type, subtype, role, page, and location settings. These will define the signatures required in each document.

```

    jsonString += "\"documents\": [{\"approvals\": [{\"role\": \"Role1\", \"fields\": [\"page\": 0, \"top\": 100, \"subtype\": \"FULLNAME\", \"height\": 50, \"left\": 100, \"width\": 200, \"type\": \"SIGNATURE\"]}], {\"role\": \"Role2\", \"fields\": [{\"page\": 0, \"top\": 300, \"subtype\": \"FULLNAME\", \"height\": 50, \"left\": 100, \"width\": 200, \"type\": \"SIGNATURE\"]}], \"name\": \"Test Document\"]}],\"";

```

Finally, the last few settings of the package JSON string that you will want to note are the name, type, status, emailMessage, and autoComplete. In this example, status is set to

SENT. This will send the package and notify the signers. If you want to save this package as a draft, make the value DRAFT, instead.

```
jsonString += "\"name\": \"Example Package\", \"type\": \"PACKAGE\", \"language\": \"en\",  
  \"emailMessage\": \"\", \"description\": \"New Package\", \"autocomplete\": true,  
  \"status\": \"SENT\"";  
jsonString += "}";
```

The next line of the code will take your JSON string and make the StringContent object that you will pass as the payload in your REST API command.

```
StringContent jsonContent = new StringContent(jsonString, Encoding.UTF8,  
  "application/json");
```

These next lines read your PDF file and use that to create the file ByteArrayContent object for your REST call. Be sure to change the placeholder to your filepath.

```
byte[] fileByteArray = File.ReadAllBytes("PATH_TO_YOUR_PDF.pdf");  
ByteArrayContent content = new ByteArrayContent(fileByteArray);
```

The next section defines the content-disposition header of the PDF file content object.

```
content.Headers.ContentDisposition = new ContentDispositionHeaderValue("form-data");  
content.Headers.ContentDisposition.Name = "\"file\"";  
content.Headers.ContentDisposition.FileName = "\"NAME_OF_YOUR_FILE.pdf\"";  
content.Headers.ContentType = new MediaTypeHeaderValue("application/pdf");
```

The next block is where you create your multipart form and add your content objects created above to pass with your REST call. Be sure to set the File name.

```
MultipartFormDataContent form = new MultipartFormDataContent();  
form.Add(content, "\"file\"", "\"NAME_OF_YOUR_FILE.pdf\"");  
form.Add(jsonContent, "\"payload\"");
```

Next, create the HttpClient that you will use to make your POST request and set the appropriate header authorization and accept values.

```
HttpClient myClient = new HttpClient();
```

```
myClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Basic",  
apiKey);  
myClient.DefaultRequestHeaders.Add("Accept", "application/json");
```

Finally, you will make the PST call and pass your form to Conga Sign, creating your document package. The Debug line will display the result to the debug output console.

```
var response = myClient.PostAsync(new Uri(url) + "/packages/", form).Result;  
Debug.WriteLine(response.Content.ReadAsStringAsync().Result);
```

Checking Transaction Status and Downloading Documents

A continuation of the steps in [Creating and Sending a Transaction](#).

Retrieve Package JSON

HTTP Request

```
POST /api/cs-packages/{packageId}
```

HTTP Headers

```
Accept: application/json; esl-api-version=11.21  
Authorization: Bearer access_token
```

Retrieving Signing Status

HTTP Request

```
GET /api/cs-packages/{packageId}/signingStatus?signer={signerId}  
&document={documentId}
```

HTTP Headers

```
Accept: application/json
Authorization: Bearer access_token
```

Response Payload

```
{
  "status": "COMPLETED"
}
```

Download Zipped Documents

HTTP Request

```
GET /api/cs-packages/{packageId}/documents/zip
```

HTTP Headers

```
Accept: application/zip
Authorization: Bearer access_token
```

Download Evidence Summary

HTTP Request

```
GET /api/cs-packages/{packageId}/evidence/summary
```

HTTP Headers

```
Accept: application/pdf
Authorization: Bearer access_token
```

The first few lines define the connection info for Conga Sign. Make sure to replace the placeholder text with your API key

```
string apiKey = "YOUR_API_KEY";
```



```
string url = "https://sandbox.congasign.com/api";
```

Next, create the HttpClient that you will use for your GET requests and set the appropriate header authorization and accept values.

```
HttpClient myClient = new HttpClient();
myClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer",
    access_token);
myClient.DefaultRequestHeaders.Add("Accept", "application/json,application/zip,text/
html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8");
```

Check the Status

Using the packageId previously mentioned, you can set up your GET request to verify the package status. Possible values are ARCHIVED, COMPLETED, DECLINED, DRAFT, EXPIRED, OPTED_OUT, and SENT. The code below performs a GET request to the URL created from the base URL defined previously, With the remainder of the URL from the appropriate REST call defined in the packageId above. The content of the response is then parsed into a JObject and the status property of the JObject is obtained and written to the Debug output.

```
var packageStatusResponse = myClient.GetAsync(new Uri(url + "/packages/
YOUR_PACKAGE_ID")).Result;
JObject packageStatusResponseJSON =
JObject.Parse(packageStatusResponse.Content.ReadAsStringAsync().Result);
var packageStatus = packageStatusResponseJSON["status"];
Debug.WriteLine("Package Status: " + packageStatus);
```

Next, check the signing status of the package. The possible values are ARCHIVED, CANCELLED, COMPLETE, DECLINED, EXPIRED, INACTIVE, OPTED_OUT, COMPLETED, and SIGNING_PENDING.

```
var signingStatusResponse = myClient.GetAsync(new Uri(url + "/packages/
0487df67-6417-48f8-9575-b520c0f977ff/signingStatus")).Result;
JObject signingStatusResponseJSON =
JObject.Parse(signingStatusResponse.Content.ReadAsStringAsync().Result);
var signingStatus = signingStatusResponseJSON["status"].ToString();
Debug.WriteLine("Signing Status: " + signingStatus);
```

Download the Files

Finally, check if the package is complete. If it is, the zip file of all documents will be downloaded as well as the audit trail.

```
if (signingStatus.Equals("COMPLETED"))
{
    var downloadZipResponse = myClient.GetAsync(new Uri(url + "/packages/0487df67-6417-48f8-9575-b520c0f977ff/documents/zip")).Result;
    ByteArrayContent content = new
    ByteArrayContent(downloadZipResponse.Content.ReadAsByteArrayAsync().Result);
    File.WriteAllBytes("C:/Eclipse/myzip.zip", content.ReadAsByteArrayAsync().Result);
    Debug.WriteLine("Zip File Downloaded");

    var evidenceSummaryResponse = myClient.GetAsync(new Uri(url + "/packages/0487df67-6417-48f8-9575-b520c0f977ff/evidence/summary")).Result;
    ByteArrayContent evidenceSummaryContent = new
    ByteArrayContent(evidenceSummaryResponse.Content.ReadAsByteArrayAsync().Result);
    File.WriteAllBytes("C:/Eclipse/myevidencesummary.pdf",
    evidenceSummaryContent.ReadAsByteArrayAsync().Result);
    Debug.WriteLine("Evidence Summary Downloaded");
}
else
{
    Debug.WriteLine("Signing not complete");
}
```

Changing a Recipient

The Change Recipient feature enables a recipient to delegate their signature to another recipient. The delegating recipient must provide the email address and full name of the delegate. Optionally, the delegating recipient can also provide an email message for the delegate. Both recipients are notified of the recipient change and are copied on the email message. If a recipient has delegated their signature, the transaction creator is also informed.

The following code will do this:

HTTP Request

```
POST /api/cs-packages/{packageId}/roles
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{ "id": "Signer5", "reassign": true, "type": "SIGNER", "signers": [ { "email":
"signer5@example.com", "firstName": "John", "lastName": "Smith", "id": "Signer5" } ],
"name": "Signer5" }
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{ "id": "Signer5", "data": null, "specialTypes": [], "emailMessage": null,
"attachmentRequirements": [], "locked": false, "reassign": true, "index": 0,
"signers": [ { "group": null, "language": "en", "signature": null, "id": "Signer5",
"delivery": { "provider": false, "email": false, "download": false }, "auth": {
"scheme": "NONE", "challenges": [] }, "knowledgeBasedAuthentication": null, "data":
null, "title": "", "company": "", "email": "signer5@example.com", "firstName": "John",
"lastName": "Smith", "external": null, "updated": "2017-11-16T16:53:01Z", "phone":
"", "professionalIdentityFields": [], "userCustomFields": [], "address": null,
"created": "2017-11-16T16:53:01Z", "name": "", "specialTypes": [] } ], "name":
"Signer5", "type": "SIGNER" }
```

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
id	string	Yes	No	n/a	Signer5
reassign	boolean	Yes	No	false	true/false
name	string	Yes	No	n/a	Signer5
id	string	Yes	No	n/a	Signer5

Property	Type	Editable	Required	Default	Sample Values
type	string	Yes	No	SIGNER	SIGNER/SENDER
signers					
email	string	Yes	No	n/a	signer5@example.com
firstName	string	Yes	No	n/a	John
lastName	string	Yes	No	n/a	Smith
id	string	Yes	No	n/a	Signer5

Adding Signatures

To add a signature to a document, you will first need to edit the fields object. The following code will do this:

HTTP Request

```
POST /api/cs-packages/{packageId}/documents/{documentId}/approvals
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{  "role": "Signer1",  "id": "signature1",  "fields": [  {  "top": 510,  "left": 215,  "width": 200,  "height": 50,  "page": 0,  "type": "SIGNATURE",  "subtype": "FULLNAME"  }  ]  }
```

```
{  "role": "Signer1",  "id": "signature1",  "fields": [  {  "top": 510,  "left": 215,  "width": 200,  "height": 50,  "page": 0,  "type": "SIGNATURE",  "subtype": "CAPTURE"  }  ]  }
```

```
{  "role": "Signer1",  "id": "signature1",  "fields": [  {  "top": 510,  "left": 215,  "width": 200,  "height": 50,  "page": 0,  "type": "SIGNATURE",  "subtype": "INITIALS"  }  ]  }
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{  "signed": null,  "role": "Signer1",  "accepted": null,  "id": "signature1",  "data": null,  "fields": [  {  "top": 50,  "left": 300,  "height": 50,  "page": 0,  "type": "SIGNATURE",  "validation": null,  "binding": null,  "width": 200,  "subtype": "FULLNAME",  "extract": false,  "extractAnchor": null,  "id": "tx1qz7485780",  "data": null,  "value": "",  "name": ""  }  ],  "name": ""  }
```

Updating a Signature On a Document

You can update a signature that has already been associated with a document. To update a signature, you will need to make a PUT request to:

```
https://services.congamerger.com/api/cs-packages/{packageId}/documents/{documentId}/approvals/{approvalId}
```

With the updated signature parameters:

```
{  "role": "Signer1",  "fields": [  {  "top": 510,  "left": 215,  "width": 300,  "height": 50,  "id": "signature1",  "page": 0,  "type": "SIGNATURE",  "subtype": "CAPTURE"  }  ]  }
```

i When updating a signature, your new Signature object must have the same id as the signature you want to update.

Deleting a Signature From a Document

Finally, deleting a signature is done by making a DELETE request to:

```
https://sandbox.congamerge.com/api/cs-packages/{packageId}/documents/{documentId}/
approvals/{signatureId}/
```

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
role	string	Yes	No	n/a	Signer1
id	string	Yes	No	n/a	signature1
fields					
top	integer	Yes	No	0	510
left	integer	Yes	No	0	215
width	integer	Yes	No	200	200
height	integer	Yes	No	50	50
page	integer	Yes	No	0	0
type	string	Yes	No	n/a	SIGNATURE/ INPUT
subtype	string	Yes	No	n/a	FULLNAME/ INITIALS/ CAPTURE/ MOBILE_CAPTURE/ LABEL/ TEXTFIELD/ TEXTAREA/ CHECKBOX/DATE/ RADIO/LIST

Managing Signers' Attachments

Sign offers the ability for signers to upload attachments during the signing workflow. Senders can review the provided attachment and mark the transaction as complete. This topic describes how a sender can require the signer to upload an attachment to the transaction.

Creating an Attachment Request

The following code describes how to add a recipient with an attachment requirement.

HTTP Request

```
POST /api/cs-packages/{packageId}/roles
```

HTTP Headers

```
Accept: application/json  
Content-Type: application/json  
Authorization: Bearer access_token
```

Request Payload

```
{  
  "id": "client",  
  "type": "SIGNER",  
  "index": 1,  
  "attachmentRequirements": [  
    {  
      "description": "Please upload a scanned copy of your driver's license.",  
      "required": true,  
      "id": "lD6p5QnWk905",  
      "status": "INCOMPLETE",  
      "comment": "",  
      "name": "Driver's license"  
    }  
  ],  
  "signers": [  
    {  
      "firstName": "John",
```

```

        "lastName": "Smith",
        "email": "john.smith@example.com"
    }
],
    "name": "client"
}

```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```

{
    "id": "client",
    "data": null,
    "emailMessage": null,
    "attachmentRequirements": [
        {
            "status": "INCOMPLETE",
            "description": "Please upload a scanned copy of your driver's license.",
            "required": true,
            "id": "1JMGfs9xRDoD",
            "comment": "",
            "name": "Driver's license",
            "data": null
        }
    ],
    "locked": false,
    "reassign": false,
    "specialTypes": [],
    "index": 1,
    "type": "SIGNER",
    "signers": [
        {
            "group": null,
            "language": "en",
            "signature": null,
            "id": "fe666c24-c18d-4d93-bbb7-2b1a6ce8332e",
            "auth": {
                "scheme": "NONE",
                "challenges": []
            },
            "data": null,
            "title": "",

```



```

        "external": null,
        "updated": "2017-10-19T18:18:37Z",
        "company": "",
        "email": "john.smith@example.com",
        "firstName": "John",
        "lastName": "Smith",
        "phone": "",
        "professionalIdentityFields": [],
        "userCustomFields": [],
        "knowledgeBasedAuthentication": null,
        "delivery": {
            "provider": false,
            "email": false,
            "download": false
        },
        "address": null,
        "created": "2017-10-19T18:18:37Z",
        "name": "",
        "specialTypes": []
    }
],
    "name": "client"
}

```

Downloading Attachments

Attachments can be downloaded in the following ways:

- As a single attachment. This requires the attachment id.
- In a package that includes all attachments.
- In a package that includes all attachments from a specified signer.

To download an individual attachment, you will need the package and attachment ID. The following code will do this:

HTTP Request

```
GET /api/cs-packages/{packageId}/attachment/{attachmentId}
```

HTTP Headers

```

Accept: application/octet-stream
Content-Type: application/octet-stream

```

```
Authorization: Bearer access_token
```

Response Payload

```
[document.pdf]
```

You can also download all attachments as a zip file in a package. The following code will do this:

HTTP Request

```
GET /api/cs-packages/{packageId}/attachment/zip
```

HTTP Headers

```
Accept: application/zip  
Content-Type: application/zip  
Authorization: Bearer access_token
```

Response Payload

```
[document.zip]
```

To download all attachments as a zip file for a particular signer, you will make your request to:

HTTP Request

```
GET /api/cs-packages/{packageId}/attachment/zip/{roleId}
```

HTTP Headers

```
Accept: application/zip  
Content-Type: application/zip  
Authorization: Bearer access_token
```

Request Payload

```
[document.zip]
```

Reviewing Attachments

After reviewing the attachment, you can then either refuse or accept it. If you refuse the attachment, you can also include a small feedback message explaining why the attachment was refused. The following code will do this:

HTTP Request


```
PUT /api/cs-packages/{packageId}/roles/{roleId}
```

HTTP Headers

```
Accept: application/json  
Content-Type: application/json  
Authorization: Bearer access_token
```

Request Payload

```
{  
  "attachmentRequirements": [  
    {  
      "id": "q66CYiDrxTU1",  
      "status": "REJECTED",  
      "comment": "Invalid copy."  
    }  
  ]  
}
```

 It is important to note that transactions that require attachments will not auto-complete. This gives the sender the opportunity to review the attachment, and either accept or reject it.

If all the required attachments have been uploaded and approved, you can complete the package by updating the status of your package to COMPLETED. The following code will do this:

HTTP Request

```
PUT /api/cs-packages/{packageId}/
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{
  "status" : "COMPLETED"
}
```

JSON Property Table

Property	Type	Editable	Required	Default	Sample Values
id	string	Yes	No	n/a	client
type	string	Yes	No	SIGNER	SIGNER / SENDER
index	integer	Yes	No	0	0/1/2
name	string	Yes	No	n/a	client
attachmentRequirements					
description	string	Yes	No	n/a	Please upload a scanned copy of your driver's license
required	boolean	Yes	No	false	false/true
id	string	Yes	No	n/a	1D6p5QnWk905
status	string	Yes	No	INCOMPLETE	INCOMPLETE/COMPLETE/REJECTED
comment	string	Yes	No	n/a	wrong drivers license

Property	Type	Editable	Required	Default	Sample Values
name	string	Yes	No	n/a	Driver's license
signers					
firstName	string	Yes	No	n/a	John
lastName	string	Yes	No	n/a	Smith
email	string	Yes	No	n/a	john.smith@example.com

Delivering Signed Documents

Once a transaction has been completed, you can automatically deliver the signed documents to your signers. The following code will do this:

HTTP Request

```
POST /api/cs-packages/{packageId}/roles
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{  "id": "Signer5",  "reassign": true,  "type": "SIGNER",  "signers": [  {
    "email": "signer5@example.com",  "firstName": "John",  "lastName": "Smith",  "id":
    "Signer5",  "delivery": {  "email": true,  "download": true,  "provider": true
  }  },  "name": "Signer5"  }
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{
  "id": "Signer5",
  "data": null,
  "specialTypes": [],
  "emailMessage": null,
  "attachmentRequirements": [],
  "locked": false,
  "reassign": true,
  "index": 0,
  "signers": [
    {
      "group": null,
      "language": "en",
      "signature": null,
      "id": "Signer5",
      "delivery": {
        "provider": true,
        "email": true,
        "download": true
      },
      "auth": {
        "scheme": "NONE",
        "challenges": []
      },
      "knowledgeBasedAuthentication": null,
      "data": null,
      "title": "",
      "company": "",
      "email": "signer5@example.com",
      "firstName": "John",
      "lastName": "Smith",
      "external": null,
      "updated": "2017-11-16T16:53:01Z",
      "phone": "",
      "professionalIdentityFields": [],
      "userCustomFields": [],
      "address": null,
      "created": "2017-11-16T16:53:01Z",
      "name": "",
      "specialTypes": []
    }
  ],
  "name": "Signer5",
  "type": "SIGNER"
}
```

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
id	string	Yes	No	n/a	Signer5
reassign	boolean	Yes	No	false	true/false
name	string	Yes	No	n/a	Signer5
type	string	Yes	No	SIGNER	SIGNER/SENDER
signers					
email	string	Yes	No	n/a	signer5@example.com
firstName	string	Yes	No	n/a	John
lastName	string	Yes	No	n/a	Smith
id	string	Yes	No	n/a	Signer5
delivery					
email	boolean	Yes	No	false	true/false

Property	Type	Editable	Required	Default	Sample Values
download	boolean	Yes	No	false	true/false
provider	boolean	Yes	No	false	true/false

Creating a Signer Workflow

When creating a transaction, you may have multiple signers required to sign several documents. Accordingly, you may wish to control the flow of the Signer Experience. With Sign, you can set the order in which multiple signers participate or the order in which multiple documents are signed by a signer. The signer workflow determines the order in which multiple signers can participate in the Signer Experience. For example, a signer with a signing order 1 means that they will sign first.

Creating a Signer Workflow

In this example, there are four signers where each signer is required to sign in order. For example, the second signer will receive a notification to sign the document only after the first signer has completed signing. The following code will do this:

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{ "roles": [ { "id": "role1", "index": 1, "type": "SIGNER", "signers": [ { "email": "signer1@example.com", "firstName": "FirstNameSigner1", "lastName": "LastNameSigner1" } ], "name": "signer1" }, { "id": "role2", "index": 2, "type": "SIGNER", "signers": [ { "email": "signer2@example.com", "firstName": "FirstNameSigner2", "lastName": "LastNameSigner2" } ], "name": "signer2" }, { "id": "role3", "index": 3, "type": "SIGNER", "signers": [ { "email": "signer3@example.com", "firstName":
```

```
"FirstNameSigner3", "lastName": "LastNameSigner3" } ], "name": "signer3" }, { "id": "role4", "index": 4, "type": "SIGNER", "signers": [ { "email": "signer4@example.com", "firstName": "FirstNameSigner4", "lastName": "LastNameSigner4" } ], "name": "signer4" } ] }
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{ "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI=" }
```

Editing the Role Workflow

To update the role workflow, you will need to make a PUT request to:

HTTP Request

```
PUT /api/cs-packages/{packageId}/roles
```

HTTP Headers


```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
[{ "id":"OGCBuBLGa7gR", "index":0 }, { "id":"CnvwToUWLPgW", "index":1 }]
```

Editing your Signer Workflow

After you have created your transaction, you may wish to edit the signing order. In this example, the signing order of the first two signers is changed.

 If you've already sent your transaction, you will need to change its status to DRAFT before you can update the signer workflow.

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
roles					
id	string	Yes	No	n/a	role1
index	integer	Yes	No	0	1/2/3
type	string	Yes	No	SIGNER	SIGNER/SENDER
signers					
name	string	Yes	No	n/a	signer1
email	string	Yes	No	n/a	signer1@example.com
firstName	string	Yes	No	n/a	FirstNameSigner1
lastName	string	Yes	No	n/a	LastNameSigner1

Downloading Signed Documents and Audit Trails

Any documents uploaded during the creation of a document package can be downloaded at any given time.

Downloading Documents

The `DownloadDocument()` call can be called both before and after the package's completion. If called before, the documents will be flattened, removing all pending signatures and fields.

HTTP Request

```
GET https://sandbox.congamerge.com/api/cs-packages/{packageId}/documents/
{documentId}/pdf
```

HTTP Headers

```
Accept: application/pdf  
Content-Type: application/json  
Authorization: Bearer access_token
```

Downloading the Original Documents

The `DownloadOriginalDocument()` call can be called both before and after the package's completion. It retrieves the original document that was uploaded by the sender, without any signatures and fields.

HTTP Request

```
GET https://sandbox.congamerge.com/api/cs-packages/{packageId}/documents/  
{documentId}/original
```

HTTP Headers

```
Accept: application/pdf  
Content-Type: application/json  
Authorization: Bearer access_token
```

Downloading Signed Documents

The `DownloadZippedDocuments()` call can only be called after all signing has been completed. Once called, the method will deliver an archive containing all signed documents.

HTTP Request

```
GET https://sandbox.congamerge.com/api/cs-packages/{packageId}/documents/zip
```

HTTP Headers

```
Accept: application/zip  
Content-Type: application/json  
Authorization: Bearer access_token
```

Downloading an Audit Trail

The `DownloadEvidenceSummary()` call can be called both before and after the package's completion. It retrieves the current audit trail activity for the package.

HTTP Request

```
GET https://sandbox.congamerge.com/api/cs-packages/{packageId}/evidence/summary
```

HTTP Headers

```
Accept: application/pdf  
Content-Type: application/json  
Authorization: Bearer access_token
```

Retrieving Audit Trails

A document's Audit Trail contains the digital certificate used to sign, as well as the signature block image, time stamp, and unique signer identification information. The document and the e-signatures it contains are tamper-sealed with a digital signature to guarantee the integrity and authenticity of the e-signed record. Especially relevant in a multi-signer process, Sign independently time stamps and locks down each signature and any data entered by each signer independently of the others. So when you review the audit trail even years later, it is clear who signed what, in what order, at what time, etc.

To retrieve the audit trail of a package, you will need the `PackageId`, which is returned to you during package creation.

HTTP Request

```
GET /api/cs-packages/{packageId}/audit
```

HTTP Headers

```
Accept: application/json  
Content-Type: application/json  
Authorization: Bearer access_token
```

Response Payload

```

{
  "package-id": "c2e57376-7391-4559-8da9-5d0ed4a24c56",
  "audit-events": [
    {
      "type": "Signing Session For Signer",
      "date-time": "2016-04-20 14:14:51",
      "target": "signing url group",
      "user": "somegroup",
      "user-email": "9fe0d028-0572-4731-8520-99191340d78f@groups.congamerger.com",
      "user-ip": "11.11.111.111",
      "data": "Signing session to signer :
9fe0d028-0572-4731-8520-99191340d78f@groups.congamerger.com(somegroup"
    },
    {
      "type": "View",
      "date-time": "2016-04-20 14:14:54",
      "target": "doc1",
      "user": "somegroup",
      "user-email": "9fe0d028-0572-4731-8520-99191340d78f@groups.congamerger.com",
      "user-ip": "11.11.111.111",
      "data": null
    },
    {
      "type": "Click To Sign",
      "date-time": "2016-04-20 14:23:31",
      "target": "doc1",
      "user": "somegroup",
      "user-email": "9fe0d028-0572-4731-8520-99191340d78f@groups.congamerger.com",
      "user-ip": "11.11.111.111",
      "data": "Approval: FAP0ezPt0H8E"
    },
    {
      "type": "View",
      "date-time": "2016-04-20 15:20:47",
      "target": "doc1",
      "user": "John Smith",
      "user-email": "csa.signer1@example.com",
      "user-ip": "11.11.111.111",
      "data": null
    },
    {
      "type": "View",
      "date-time": "2016-04-20 15:23:52",

```


```

    "target": "doc1",
    "user": "John Smith",
    "user-email": "csa.signer1@example.com",
    "user-ip": "11.11.111.111",
    "data": null
  },
  {
    "type": "View",
    "date-time": "2016-04-20 15:44:41",
    "target": "doc1",
    "user": "John Smith",
    "user-email": "csa.signer1@example.com",
    "user-ip": "11.11.111.111",
    "data": null
  },
  {
    "type": "View",
    "date-time": "2016-04-20 15:44:47",
    "target": "doc1",
    "user": "John Smith",
    "user-email": "csa.signer1@example.com",
    "user-ip": "11.11.111.111",
    "data": null
  }
]
}

```

Adding Fields

Fields enable the placement of additional data in a document at the time of signing. Like signature fields, non-signature fields can be placed anywhere inside a document. Nonetheless, each field is linked to a particular signature. Only the signer of that signature can assign or change the field's value. Once the signature is signed, however, the value of the field cannot be changed by anyone.

 You cannot add fields to (1) documents that are Accept Only; (2) the Electronic Consent form; (3) documents in an accessible transaction.

Adding a Field to a Document

The sample request below shows you how to add a field to an existing document:

HTTP Request

```
POST /api/cs-packages/{packageId}/documents/{documentId}/approvals/{approvalId}/fields
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{  "name": "Signer Name",  "top": 500,  "left": 300,  "width": 100,  "height": 30,  "page": 0,  "type": "INPUT",  "value": null,  "binding": "{signer.name}",  "subtype": "LABEL" }
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{  "binding": "{signer.name}",  "validation": null,  "id": "BsZv3XvmpsMH",  "page": 0,  "data": null,  "subtype": "LABEL",  "top": 500,  "height": 30,  "extractAnchor": null,  "width": 100,  "extract": false,  "left": 300,  "type": "INPUT",  "value": "{signer.name}",  "name": "Signer Name" }
```

Updating a Field in a Document

It is important to note that when updating a field, your new Field object must have the same id as the field you want to update.

To update a field, you will need to make a PUT request to:

```
https://sandbox.congamerger.com/api/packages/{packageId}/documents/{documentId}/approvals/{signatureId}/fields/{fieldId}
```

With the updated field parameters:

```
{  "id": "myLabelField",  "top": 200,  "left": 100,  "width": 250,  "height": 50,  "page": 0,  "type": "INPUT",  "binding": "{approval.signed}",  "subtype": "LABEL"  }
```

Deleting a Field in a Document

Deleting a field is done by making a DELETE request to:

```
https://services.congamerger.com/api/cs-packages/{packageId}/documents/{documentId}/
approvals/{signatureId}/fields/{fieldId}
```


Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
name	string	Yes	No	n/a	Singer Name
top	integer	Yes	No	0	500
left	integer	Yes	No	0	300
width	integer	Yes	No	200	100
height	integer	Yes	No	50	30
page	integer	Yes	No	0	0/1/2 ...
type	string	Yes	No	n/a	SIGNATURE/INPUT
value	string	Yes	No	null	STU BANK
subtype	string	Yes	No	n/a	FULLNAME/INITIALS/ CAPTURE/LABEL/ TEXTFIELD/ CHECKBOX/RADIO/ LIST

Property	Type	Editable	Required	Default	Sample Values
binding	string	Yes	No	null	<code>{ approval.signed } /</code> <code>{ signer.title } /</code> <code>{ signer.name } /</code> <code>{ signer.company }</code>

Configuring Optional Signatures

Optional Signatures give your signers the flexibility to complete a transaction without having them sign all the signature fields in their documents. This feature is useful for documents that require the signer to either accept or decline sections of a form.

 At least one signer must have either a required signature or a separate **Accept Only** document for a transaction to be completed. This does not include the default consent form.

Creating Optional Signatures

The following code will make the Signature object optional:

If you need a comparison to the basic object creation procedure, or if this is the first time creating a transaction, see [Creating and Sending a Transaction](#).

HTTP Request

```
POST /api/packages
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload


```
-----WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data;
name="file"; filename="optional signature basic usage.pdf" Content-Type:
application/pdf %PDF-1.5 %µµµµ 1 0 obj <>> endobj... -----
WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data; name="payload"
{  "documents": [ {  "approvals": [ {  "fields": [ {  "height": 50,
"left": 100,  "page": 0,  "subtype": "FULLNAME",  "top": 100,  "type":
"SIGNATURE",  "width": 200  }  ],  "id": "signature1",  "role": "Role1"  }, {
"fields": [ {  "height": 50,  "left": 100,  "page": 0,  "subtype": "FULLNAME",
"top": 300,  "type": "SIGNATURE",  "width": 200  }  ],  "id": "signature2",
"role": "Role1",  "optional": true  }  ],  "name": "Test Document"  }  ],
"name": "Optional Signature Basic Usage REST",  "roles": [ {  "id": "Role1",
"signers": [ {  "email": "signer1@example.com",  "firstName": "1.firstname",
"lastName": "1.lastname",  "company": "Conga Sign"  }  ]  }  ],  "type":
"PACKAGE",  "status": "SENT"  } -----WebKitFormBoundary1bN060n7FqP5W04t--
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{  "id": "7g-LhL0zEzN0jZi7Yccell7y6jA="  }
```

Creating Optional Signatures With Document Extraction

To make a signature field optional, apply the following format to your PDF form:

```
[Signer.SigStyle#.Optional]
```

The last `.Optional` part is extended in order to set the signature as optional. Here are a few examples:

<code>[Signer1.Fullname1]</code>	<code>[Signer1.Fullname2.Optional]</code>
<code>[Signer1.Initials1]</code>	<code>[Signer1.Initials2.Optional]</code>
<code>[Signer1.Capture1]</code>	<code>[Signer1.Capture2.Optional]</code>



- Signatures are by default required, so only add .Optional when necessary.
- PDF form name syntax is case insensitive, including the Role Name and Custom ID of the signer. `[Agent1.Fullname1.Optional]` is treated the same as `[AGENT1.FULLNAME1.OPTIONAL]`.
- This syntax extension is only available for signatures and will not be recognized when creating any other fields. Names like `[Agent1.Fullname1.Textfield1.Optional]` will not work.

Adding, Updating, Removing Signers

The following segment will allow you to invite, update or remove signers from the created transaction.

Adding Signers to a Transaction

To add a signer to an existing transaction:

HTTP Request

```
POST /api/cs-packages/{packageId}/roles
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
```

```
Authorization: Bearer access_token
```

Request Payload

```
{  "id": "Signer5",  "type": "SIGNER",  "signers": [  {  "email": "signer5@example.com",  "firstName": "John",  "lastName": "Smith",  "id": "Signer5"  }  ],  "name": "Signer5"  }
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{  "id": "Signer5",  "data": null,  "specialTypes": [],  "emailMessage": null,  "attachmentRequirements": [],  "locked": false,  "reassign": false,  "index": 0,  "signers": [  {  "group": null,  "language": "en",  "signature": null,  "id": "Signer5",  "delivery": {  "provider": false,  "email": false,  "download": false  },  "auth": {  "scheme": "NONE",  "challenges": []  },  "knowledgeBasedAuthentication": null,  "data": null,  "title": "",  "company": "",  "email": "signer5@example.com",  "firstName": "John",  "lastName": "Smith",  "external": null,  "updated": "2017-11-16T16:53:01Z",  "phone": "",  "professionalIdentityFields": [],  "userCustomFields": [],  "address": null,  "created": "2017-11-16T16:53:01Z",  "name": "",  "specialTypes": []  }  ],  "name": "Signer5",  "type": "SIGNER"  }
```

Updating an Existing Signer

To update an existing signer role, create a payload with the updates you want to make, and then use the request examples below to make your updates. Use the roleId you assigned in the previous request. If you did not assign a roleId, a randomly generated ID will be assigned.

HTTP Request

```
PUT /api/cs-packages/{packageId}/roles/{roleId}
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{  "type": "SIGNER",  "signers": [  {    "email": "mary.doe@example.com",    "firstName": "Mary",    "lastName": "Doe"  }  ] }
```

Response Payload

```
{  "id": "Signer5",  "data": null,  "specialTypes": [],  "emailMessage": null,  "attachmentRequirements": [],  "locked": false,  "reassign": false,  "index": 0,  "signers": [  {    "group": null,    "language": "en",    "signature": null,    "id": "Signer5",    "delivery": {      "provider": false,      "email": false,      "download": false    },    "auth": {      "scheme": "NONE",      "challenges": []    },    "knowledgeBasedAuthentication": null,    "data": null,    "title": "",    "company": "",    "email": "mary.doe@example.com",    "firstName": "Mary",    "lastName": "Doe",    "external": null,    "updated": "2017-11-16T16:53:01Z",    "phone": "",    "professionalIdentityFields": [],    "userCustomFields": [],    "address": null,    "created": "2017-11-16T16:53:01Z",    "name": "",    "specialTypes": []  }  ],  "name": "Signer5",  "type": "SIGNER" }
```

For a complete description of each field, see the Request Payload Table below.

Removing Signers From a Transaction

To remove a signer, you can make a DELETE request:

HTTP Request

```
DELETE /api/cs-packages/{packageId}/roles/{signerId}
```

HTTP Headers


```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
id	string	Yes	No	n/a	Signer5
reassign	boolean	Yes	No	false	true/false
name	string	Yes	No	n/a	Signer5
id	string	Yes	No	n/a	Signer5
type	string	Yes	No	SIGNER	SIGNER/SENDER
signers					
email	string	Yes	No	n/a	signer5@example.com
firstName	string	Yes	No	n/a	John
lastName	string	Yes	No	n/a	Smith
id	string	Yes	No	n/a	Signer5

Document Extraction

The document extraction feature automatically creates all signatures and fields that exist in an uploaded PDF file. The positions and sizes of the signatures and fields in the PDF file are automatically retained in Conga Sign.

-  The following limitations affect Document Extraction:
- Document Extraction is available only in the API call that uploads the document. It is not available in subsequent calls.
 - Radio button fields are not supported when using Document Extraction.

Configuring PDF Form Fields

First, you will need a PDF with form fields, named in a way that Conga Sign can recognize them. For more information on the proper format of the form field names, see the feature overview. The form field names on the fields are shown in the image below:

Sample Agreement June 10, 2021

Signer Info:

Name:	<input type="text" value="[Signer1.Capture1.Label1.Name]"/>	Age:	<input type="text" value="[Signer1.Capture1.Textfield1]"/>
Address:	<input type="text" value="[Signer1.Capture1.Textfield2]"/>	City:	<input type="text" value="[Signer1.Capture1.Textfield3]"/>
City:	<input type="text" value="[Signer1.Capture1.Textfield4]"/>	Zip:	<input type="text" value="[Signer1.Capture1.Textfield5]"/>

Preparer area:

Agreement Number:	<input type="text" value="[Owner.Capture1.Textfield1]"/>	Comments:	<input type="text" value="[Owner.Capture1.Textfield2]"/>
-------------------	--	-----------	--

Signer's Signature:	<input type="text" value="[Signer1.Capture1]"/>	Preparer's Signature	<input type="text" value="[Owner.Capture1]"/>
---------------------	---	----------------------	---

As you can see, the two signers on the document will be Signer1 and Owner. These will be the custom IDs used in the code section below to let Conga Sign know what fields to associate with each signer. By default, the values used for the role of the extraction tag/fieldName are `Signer1`, `Signer2`, and so on. If you want to include the sender of the transaction as a signer, use the tag `Owner`.

Configuring Document Extraction

Typically, you will build your JSON string dynamically, versus having a giant static string, like this. This is to give a good representation of the structure of the JSON you will need to create your transaction properly.

The JSON below is formatted for readability. In each roles object, you will see that the custom IDs coincide with the ones in the image of the PDF form shown above. The documents object also has **extract to true**.

Because this is done, you do not have to define the signature locations and who needs to sign the document. This is already taken care of with the IDs and the associated form field names from the PDF.

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: multipart/form-data
Authorization: Bearer access_token
```

Request Payload

```
-----WebKitFormBoundary1bN060n7FqP5W04t
Content-Disposition: form-data; name="file"; filename="testDocumentExtraction.pdf"
Content-Type: application/pdf
%PDF-1.5
%ÂµÂµÂµÂµ
1 0 obj
<>>
endobj....
-----WebKitFormBoundary1bN060n7FqP5W04t
Content-Disposition: form-data; name="payload"
{
  "roles": [
    {
      "id": "Signer1",
      "type": "SIGNER",
      "signers": [
        {
          "firstName": "John",
          "lastName": "Smith",
          "email": "signer1@example.com",
          "id": "Signer1"
        }
      ],
      "name": "Signer1"
    }
  ],
  "documents": [
    {
      "name": "testDocumentExtraction",
      "extract": true
    }
  ],
  "name": "Test Document Extraction",
```

```

    "type": "PACKAGE",
    "autoComplete": true,
    "status": "SENT"
  }
  -----WebKitFormBoundary1bN060n7FqP5W04t--

```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{  "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI=" }
```


Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
status	string	Yes	No	DRAFT	DRAFT/SENT/COMPLETED/ARCHIVED/DECLINED/OPTED_OUT/EXPIRED
autoComplete	boolean	Yes	No	true	true/false
type	string	Yes	No	PACKAGE	PACKAGE/TEMPLATE/LAYOUT
name	string	Yes	Yes	n/a	Test Document Extraction
Documents					
name	string	Yes	No	n/a	testDocumentExtraction
extract	boolean	Yes	No	false	true/false
Roles					

Property	Type	Editable	Required	Default	Sample Values
id	string	Yes	No	n/a	Preparer1
name	string	Yes	No	n/a	Sender
type	string	Yes	No	SIGNER	SIGNER/SENDER
Signers					
email	string	Yes	Yes	n/a	preparer1@example.com
firstName	string	Yes	Yes	n/a	Michael
lastName	string	Yes	Yes	n/a	Williams
phone	string	Yes	No	n/a	514-555-8888
id	string	Yes	No	n/a	Preparer1
Delivery					
email	boolean	Yes	No	false	true/false
provider	boolean	Yes	No	false	true/false
download	boolean	Yes	No	false	true/false

Feature Overview

Document Extraction looks for appropriately named signatures and fields in an uploaded PDF document, and for each one, it creates a Sign signature or field. The positions and sizes of the signatures and fields from the PDF are automatically retained in Sign.


-  The following limitations affect Document Extraction:
- Document Extraction is available only in the API call that uploads the document. It is not available in subsequent calls.
 - Radio button fields are not supported when using Document Extraction.

The information needed to create each Sign signature or field is taken from the name of the PDF signature or field.

Naming Conventions

To create a Conga Sign signature field, the PDF field must have a name of the form `[Signer.SigStyle#]`, where:

- **Signer:** By default, the values used for the role of the extraction tag/fieldName are Signer1, Signer2, and so on. If you want to include the sender of the transaction as a signer, use the tag Owner. If you specify a custom role ID for your recipients, you'd use that value in your tags.
- **SigStyle#:** A signature style (Capture, Initials, or Fullname) combined with an integer for uniqueness. Example: Capture1 or Fullname09.

-  If you want the signature field to be optional, simply use the following format:
`[Signer.SigStyle#.Optional]`


To create an unbound Conga Sign field, the PDF field must have a name of the form `[Signer.SigStyle#.FieldStyle#]`, where:

- **Signer.SigStyle#** identifies the signature associated with this field.
- **FieldStyle#** is a field style (Textfield or Checkbox) combined with an integer for uniqueness. For example: Textfield2 or Checkbox6.

To create a bound Conga Sign field, the PDF field must have a name of the form `[Signer.SigStyle#.label#.Binding]`, where:

- **Signer.SigStyle#** identifies the signature associated with this field.
- **label#** is an identifier of the field, consisting of the word label combined with an integer for uniqueness.

- **Binding** is a field style. The possible values are Date or {approval.signed}, Name or {signer.name}, Title or {signer.title}, and Company or {signer.company}. Both members of each pair signify the same field style.

 All parts of a PDF field name are matched using case-insensitive matches. For example, a field named [Signer1.Fullname1.label1.Date] is treated the same as a field named [Signer1.FULLNAME1.LABEL1.DATE]. Field names are alphanumeric. They cannot contain special characters other than the underscore (_).

Example

Here is an example. Suppose a package has two signers whose custom IDs are Signer1 and Owner. Further, suppose that extraction is enabled and that a PDF is uploaded which has fields with the following names:

```
[Owner.Fullname1] [Owner.Fullname1.label1.Date] [Owner.Fullname1.Textfield1]
[Owner.Fullname1.Checkbox1] [Owner.Fullname2.Optional]
```

```
[Signer1.Capture1] [Signer1.Capture1.label1.Name] [Signer1.Capture1.label2.Date]
[Signer1.Capture1.label3.Title] [Signer1.Initials1] [Signer1.Initials2]
```

Before signing, **Owner** must complete two fields:

```
[Owner.Fullname1.Textfield1] [Owner.Fullname1.Checkbox1]
```

Once those fields are complete, **Owner** can sign **[Owner.Fullname1]**, and **[Owner.Fullname1.label1.Date]** will automatically be filled with the date of signing.

Signer1 needs to sign in three places:

```
[Signer1.Initials1] [Signer1.Initials2] [Signer1.Capture1]
```

Once these are all signed, the remaining fields will be filled:

If you have specified custom IDs for the sender and the signer (for example, **Agent1** and **Client1**) user form field names similar to the ones below, and the signing flow will remain the same:

[Agent1.Fullname1] [Client1.Initials1]

Uploading & Deleting Documents

This topic includes the following:

- How to upload a document
- How to replace an existing document
- How to update the metadata in a document
- How to delete a document after you have created a package

Uploading Documents

Uploading a Single Document

To upload a document after creating a package, you will need to make a multipart-form POST request.

HTTP Request

```
POST /api/cs-packages/{packageId}/documents
```

HTTP Headers

```
Accept: text/html
Content-Type: multipart/form-data
Authorization: Bearer access_token
```

Request Payload

```
-----WebKitFormBoundary1bN060n7FqP5W04t
      Content-Disposition: form-data; name="file"; filename="testDocumentEx
traction.pdf"
      Content-Type: application/pdf
      %PDF-1.5
      %µµµµµµ
      1 0 obj
      <>>
      endobj....
```

```

-----WebKitFormBoundary1bN060n7FqP5W04t
Content-Disposition: form-data; name="payload"
{
  "name": "Example Document",
  "description": "Example Description",
  "id": "Document1"
}
-----WebKitFormBoundary1bN060n7FqP5W04t--

```

For a complete description of each field, see the Request Payload Table section below.

Response Payload

```

{
  "status": "",
  "description": "Example Description",
  "id": "Document1",
  "data": {
    "ese_document_texttag_extract_needed": "false"
  },
  "approvals": [
    {
      "id": "k1r2qMRtCsI5",
      "role": "48d0c024-0609-4255-9087-941a66f80738",
      "data": null,
      "signed": null,
      "accepted": null,
      "fields": [],
      "name": ""
    }
  ],
  "pages": [
    {
      "id": "",
      "top": 0,
      "height": 1030,
      "width": 796,
      "left": 0,
      "index": 0,
      "version": 0
    }
  ],
  "external": null,
  "extract": false,
  "signedHash": null,

```

```

    "signerVerificationToken": null,
    "index": 1,
    "fields": [],
    "name": "Example Document",
    "size": 185808
  }

```

Uploading Multiple Documents

To upload multiple documents in one call, you would use the same API URL but modify the request body in the following way:

HTTP Request

```
POST /api/cs-packages/{packageId}/documents
```

HTTP Headers

```

Accept: text/html
Content-Type: multipart/form-data
Authorization: Bearer access_token

```

Request Payload

```

-----WebKitFormBoundary1bN060n7FqP5W04t
  Content-Disposition: form-data; name="file"; filename="doc1.pdf"
  Content-Type: application/pdf
  %PDF-1.5
  %µµµµµµ
  1 0 obj
  <>>
  endobj....
-----WebKitFormBoundary1bN060n7FqP5W04t
  Content-Disposition: form-data; name="file"; filename="doc2.pdf"
  Content-Type: application/pdf
  %PDF-1.5
  %µµµµµµ
  1 0 obj
  <>>
  endobj....

```

```

-----WebKitFormBoundary1bN060n7FqP5W04t
Content-Disposition: form-data; name="payload"
[
{
"name": "Document1",
"description": "Example Description",
"id": "Document1"
},
{
"name": "Document2",
"description": "Example Description",
"id": "Document2"
}
]
-----WebKitFormBoundary1bN060n7FqP5W04t--

```

Replacing an Existing Document

Replacing an existing document in a transaction can also be done using the upload document API. This is useful, for example, if you would like to replace the document, but still keep the layout of the fields and signatures.

To do so, make a multipart-form POST request where the document JSON payload carries the same document ID as the original document:

HTTP Request

```
POST /api/cs-packages/{packageId}/documents
```

HTTP Headers

```

Accept: text/html
Content-Type: multipart/form-data
Authorization: Bearer access_token

```

Request Payload

```

-----WebKitFormBoundary1bN060n7FqP5W04t
Content-Disposition: form-data; name="file"; filename="doc1.pdf"
Content-Type: application/pdf
%PDF-1.5
%ÂµÂµÂµÂµ

```

```

1 0 obj
<>>
endobj....
-----WebKitFormBoundary1bN060n7FqP5W04t
Content-Disposition: form-data; name="payload"
{
"name": "Replaced Document",
"id": "keep_the_document_id_same"
}
-----WebKitFormBoundary1bN060n7FqP5W04t--

```

Similarly, you can replace multiple documents using the following request:

Request Payload

```

-----WebKitFormBoundary1bN060n7FqP5W04t
Content-Disposition: form-data; name="file"; filename="doc1.pdf"
Content-Type: application/pdf

%PDF-1.5
%ÂµÂµÂµÂµ
1 0 obj
<>>
endobj....
-----WebKitFormBoundary1bN060n7FqP5W04t
Content-Disposition: form-data; name="file"; filename="doc2.pdf"
Content-Type: application/pdf

%PDF-1.5
%ÂµÂµÂµÂµ
1 0 obj
<>>
endobj....
-----WebKitFormBoundary1bN060n7FqP5W04t

Content-Disposition: form-data; name="payload"
[
{
"name": "Replaced Document1",
"id": "keep_the_document1_id_same"
},
{
"name": "Replaced Document2",

```



```
"id": "keep_the_document2_id_same"
}
]-----WebKitFormBoundary1bN060n7FqP5W04t--
```

Updating Document Metadata

To update the metadata for a document, you will need to create your JSON using the updated document metadata:

```
{
  "name": "Example Document",
  "description": "Example Description"
}
```

Then use the following command:

```
PUT https://services.congamerge.com/api/cs-packages/{packageId}/documents/
{documentId}
```

The call above will only update the document level attributes you defined in your new JSON payload and not any other embedded attributes that can be updated.

Deleting a Document

To delete a document, make a DELETE request to the following URI:

```
https://services.congamerge.com/api/cs-packages/{packageId}/documents/{documentId}
```

Results

After running your code, if you log in to Sign and navigate to your transaction, you will be able to log into your Sign account and view the documents you just added.

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
name	string	Yes	No	n/a	Example Document

Property	Type	Editable	Required	Default	Sample Values
description	string	Yes	No	n/a	Example Document
id	string	Yes	No	n/a	Document1

Customizing Invitation Emails

When sending transactions for signing you can customize the email message your signers receive. Customization can be applied on both the Transaction level and by the Signer.

Customizing emails by transaction

You can add a transaction level message for all signers in the transaction. The following code will do this.

HTTP Request

```
PUT /api/cs-packages/{packageId}
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{  "emailMessage": "Changed package level email message." }
```

Customizing emails by Signer

You can add a transaction level message for each signer in the transaction. Signer level messages override the transaction level message. The following code will do this.

HTTP Request

```
POST /api/cs-packages/{packageId}/roles
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{
  "emailMessage": {
    "content": "Please sign the documents ASAP."
  },
  "id": "Signer5",
  "reassign": true,
  "type": "SIGNER",
  "signers": [ { "email": "signer5@example.com", "firstName": "John",
    "lastName": "Smith", "id": "Signer5"  }  ],
  "name": "Signer5"
}
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{
  "id": "Signer5",
  "data": null,
  "specialTypes": [],
  "emailMessage": {
    "content": "Please sign the documents ASAP."
  },
  "attachmentRequirements": [],
  "locked": false,
  "reassign": true,
  "index": 0,
  "signers": [ { "group": null, "language": "en", "signature": null,
    "id": "Signer5", "delivery": { "provider": false, "email":
```

```

false, "download": false      },      "auth": {      "scheme": "NONE",
"challenges": []
    }
    ,
    "knowledgeBasedAuthentication": null,
    "data": null,
    "title": "",
    "company": "",
    "email": "signer5@example.com",
    "firstName": "John",
    "lastName": "Smith",
    "external": null,
    "updated": "2017-11-16T16:53:01Z",
    "phone": "",
    "professionalIdentityFields": [],
    "userCustomFields": [],
    "address": null,
    "created": "2017-11-16T16:53:01Z",
    "name": "",
    "specialTypes": []
  }
],
"name": "Signer5",
"type": "SIGNER"
}

```

Results Payload Table

Property	Type	Editable	Required	Default	Sample Values
id	string	Yes	No	n/a	Signer5
emailMessage					
content	string	Yes	No	n/a	Please sign the documents ASAP
name	string	Yes	No	n/a	Signer5
id	string	Yes	No	n/a	Signer5

Property	Type	Editable	Required	Default	Sample Values
type	string	Yes	No	SIGNER	SIGNER/SENDER
signers					
email	string	Yes	No	n/a	signer5@example.com
firstName	string	Yes	No	n/a	John
lastName	string	Yes	No	n/a	Smith
id	string	Yes	No	n/a	Signer5

Injecting Field Values

- The URL for the Push Notification Service is:
http://{SERVER_NAME:PORT}/PostEvent.svc/PostESLEvent
 Here { [SERVER_NAME:PORT](#) } is a placeholder for the server name and port number.
- The service can be hosted on any port you want.
- The server name and port must be accessible from the Internet.
- You may want to open your firewall exclusively to Sign's IP address.
- This URL should be used in Sign Account > Callback URL in the connector configuration.
- [PostESLEvent](#) is the method in the service that receives [HTTP POST](#) requests with a JSON payload as event data.

While this topic describes how to inject text fields into a document, the same procedure can be used to inject any type of unbound field. For more information on the types of unbound fields that are available, see [Form Building Fields](#).

If you need a comparison to the basic object creation procedure, or if this is the first time creating a transaction, see [Creating and Sending a Transaction](#).

The sample code below shows you how to edit the document block for injecting text fields. The [withName\(\)](#) method is used to input the name of the PDF form field you want to inject text into, as it is named in your PDF form. The [withValue\(\)](#) method is used as the text you want to stamp on your document.

The sample JSON below shows you how to edit the document object for injecting text fields. In the fields object, the name attribute is the name of the PDF form field you want to inject text into. The value attribute is where you input the text you want to inject into your document.

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: multipart/form-data
Authorization: Bearer access_token
```

Request Payload

```
-----WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data;
name="file"; filename="testDocumentExtraction.pdf" Content-Type: application/pdf
%PDF-1.5 %ÃµÃµÃµÃµ 1 0 obj <>> endobj.... -----
WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data; name="payload"
{  "documents": [  {  "fields": [  {  "value": "200 E MAIN ST, PHOENIX AZ,
85123 USA",  "name": "Text1"  },  {  "value": "Lawn mower",  "name": "Text2"  }
,  {  "value": "Fertilizer",  "name": "Text3"  },  {  "value": "100",  "name":
"Text4"  },  {  "value": "12 12 12",  "name": "Text5"  }  ],  "extract": true,
"name": "Sample Contract"  }  ],  "type": "PACKAGE",  "status": "DRAFT",
"roles": [  {  "id": "client",  "index": 0,  "type": "SIGNER",  "signers": [  {
"email": "john.smith@example.com",  "firstName": "John",  "lastName": "Smith",
"id": "client"  }  ],  "name": "client"  },  {  "id": "contractor",  "index":
1,  "type": "SENDER",  "signers": [  {  "email": "mary.doe@example.com",
"firstName": "Mary",  "lastName": "Doe",  "id": "contractor"  }  ],  "name":
"contractor"  }  ],  "name": "Field Injection Example"  }  -----
WebKitFormBoundary1bN060n7FqP5W04t--
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{  "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI="  }
```

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
status	string	Yes	No	DRAFT	DRAFT/SENT/COMPLETED/ARCHIVED/ DECLINED/OPTED_OUT/EXPIRED
type	string	Yes	No	PACKAGE	PACKAGE/TEMPLATE/LAYOUT
name	string	Yes	Yes	n/a	Field Injection Example
documents					
name	string	Yes	No	n/a	Sample Contract
extract	boolean	Yes	No	false	false/true
fields					
value	string	Yes	No	n/a	200 E MAIN ST, PHOENIX AZ 85123 USA
name	string	Yes	No	n/a	Text1
roles					
id	string	Yes	No	n/a	Client1
name	string	Yes	No	n/a	Client1
type	string	Yes	No	SIGNER	SIGNER/SENDER
signers					
email	string	Yes	Yes	n/a	john.smith@example.com
firstName	string	Yes	Yes	n/a	John

Property	Type	Editable	Required	Default	Sample Values
lastName	string	Yes	Yes	n/a	Smith
phone	string	Yes	No	n/a	514-555-8888
id	string	Yes	No	n/a	Client1
company	string	Yes	No	n/a	Acme Inc.
title	string	Yes	No	n/a	Managing Director

Form Building Fields

When you drop these form-building fields into the document, the Field Settings panel displays to the right of the document.

Field	Usage	Field Settings
Text Field	This field accepts any text entered by the signer prior to signing.	<p>Field Name</p> <p>Recipient: Specify which signer(s) must complete the field.</p> <p>Default Value</p> <p>Is Required: Make the field mandatory at signing.</p>

Field	Usage	Field Settings
Text Area	This is a multi-line area that accepts any text entered by the signer prior to signing.	<p>Field Name</p> <p>Recipient: Specify which signer(s) must complete the field.</p> <p>Default Value</p> <p>Is Required: Make the field mandatory at signing.</p>
Checkbox	<p>This is a simple check box that the signer can either select or clear. Use check boxes when you want to enable signers to choose more than one option in a list.</p> <p>Use Checkbox Group to place selected checkboxes in a group of associated boxes. If the sender sets any box in a group as required, then during signing the signer must select at least one member of the group.</p>	<p>Field Name</p> <p>Recipient: Specify which signer(s) must complete the field.</p> <p>Checkbox Group: Use this to place selected checkboxes in a group of associated boxes.</p> <p>Default Value: Toggle On to check the box by default.</p> <p>Is Required: Make the field mandatory at signing.</p>


Field	Usage	Field Settings
Radio	<p>Radio buttons enable signers to select one of many predefined options. Use radio buttons when you want signers to choose only one option.</p> <p>Use Radio Group to place selected radio buttons in a group of associated buttons. You can identify one button in the group as the Default, thus ensuring that it will be pre-selected.</p>	<p>Field Name</p> <p>Recipient: Specify which signer(s) must complete the field.</p> <p>Radio Group: Use this to place selected radio buttons in a group of associated buttons.</p> <p>Default Value: Toggle On to select the radio button by default.</p> <p>Is Required: Make the field mandatory at signing.</p>

Field	Usage	Field Settings
List	This is a drop-down list that offers multiple predefined options. The signer can choose one option from the list. Once that option is selected, only that option is displayed, thus minimizing the amount of space this element occupies on the screen.	<ul style="list-style-type: none"> • Field Name • Recipient: Specify which signer(s) must complete the field. • List: Enter the items that should be available in the drop-down list, one item per line. • Default Value: Pre-populate the drop-down list with one of the options. • Is Required: Ensure that the signer selects one item from the list.
Label	This is a READ-ONLY label that will be stamped on the PDF.	<p>Field Name</p> <p>Recipient: Specify which signer(s) must complete the field.</p> <p>Default Value</p>

Field	Usage	Field Settings
Custom Fields	These fields are populated at the time of signing with data. Custom Fields are created for an entire account, and an account can have any number of Custom Fields.	If the sender has defined a value for their custom field (as described in the Custom Fields section of the <i>My Account</i> page), then that value will be used in the Signer Experience. If the sender has not defined a value for their custom field, a default value can be defined by your administrator, and that value will be used in the transaction.


Supported Fonts

Conga Sign supports a standard set of fonts that can be used when documents are created or uploaded for use in a transaction.

 Using a font that is not supported by Conga Sign could result in unintended consequences, including document text, becoming unreadable. While some unsupported fonts may display properly, Conga Sign cannot guarantee this on all systems or devices where the PDF is rendered if the unsupported fonts are not embedded in the document.

Sign thus recommends that you use the supported fonts listed below and that you embed any unsupported fonts in your PDF when you create it.


The table below lists only the fonts that Sign supports. It does not list any other fonts (e.g., those installed with your Operating System, or fonts installed later – such as Microsoft Office’s core fonts).

 One way to embed fonts is to use a PDF/A document. For more information, see Using PDF/A Documents below.

Font Name (displayed in the User Interface)	Comments
Times New Roman	Part of PDF base14 fonts
Times New Roman Bold	Part of PDF base14 fonts
Times New Roman Italic	Part of PDF base14 fonts
Times New Roman Bold Italic	Part of PDF base14 fonts
Helvetica	Part of PDF base14 fonts, Similar to Arial
Helvetica Bold	Part of PDF base14 fonts
Helvetica Oblique	Part of PDF base14 fonts Oblique, not Italic
Helvetica Bold Oblique	Part of PDF base14 fonts Oblique, not Italic
Courier	Part of PDF base14 fonts
Courier Oblique	Oblique, not Italic
Courier Bold	Part of PDF base14 fonts
Courier Bold Oblique	Part of PDF base14 fonts Oblique, not Italic
Symbol	
ZapfDingbats	

Using PDF/A Documents

A *PDF/A* document embeds the fonts that the document uses into the PDF file itself. This ensures that the document's readers don't need to install the document's fonts on their local machine.

 The steps you need may differ slightly from those below, depending on your version of Microsoft Word, and whether you are using *Adobe Acrobat Pro*.

To create a PDF/A document from a Word document:

1. Open the Word document.
2. Click **Save As**, and select **PDF**.
3. Click **Options**.
4. Select the **PDF/A compliant** check box.
5. Click **OK**.
6. Enter a name for your PDF, and click **Save**.

Managing Groups

Sometimes it's convenient to share requests for signatures among the members of a group. For example, it might be convenient to treat the pharmacists in a particular pharmacy as a group, so that any available member of the group can sign the paperwork for a patient's prescriptions.

In Conga Sign, a Signer Group is a set of Conga Sign users who can act as a single signer from the point of view of the package creator. Users who can become group members must already be members of the associated Conga Sign account.

Conga Sign group members receive an email invitation to sign a related document package. Among those members, signing is on a first-come, first-serve basis. When one member is signing, all other members are locked out.

Any member who signs does so on behalf of the group, but their name will be stamped on the documents they sign. Anyone verifying the document through the Audit Trail will see the individual member's signature information and identity.

All group members can monitor the progress of the group's transactions, which helps ensure that those transactions are completed on time.

Creating Groups

The sample code below will create a group with two members. It is important to note that these members have to be Senders in your account.

HTTP Request

```
POST /api/cs-groups
```

HTTP Headers

```
Accept: application/json  
Content-Type: application/json  
Authorization: Bearer access_token
```

Request Payload

```
{  
  "email": "your_group_email@example.com",  
  "name": "your_group_name",  
  "members": [  
    {  
      "pending": true,  
      "email": "signer1@example.com",  
      "memberType": "REGULAR",  
      "firstName": "John",  
      "lastName": "Smith"  
    },  
    {  
      "pending": true,  
      "email": "signer2@example.com",  
      "memberType": "REGULAR",  
      "firstName": "Mary",  
      "lastName": "Doe"  
    }  
  ]  
}
```

Response Payload

```
{
```

```

"id": "bc65203e-99df-47b4-a51c-33e8082780c5",
"members": [
  {
    "userId": "2q37oSloj5AD",
    "pending": false,
    "lastName": "Drake",
    "email": "signer3@example.com",
    "firstName": "Bill",
    "memberType": "REGULAR"
  },
  {
    "userId": "FxktNzFzmkiY",
    "pending": false,
    "lastName": "Mann",
    "email": "signer4@example.com",
    "firstName": "Audrey",
    "memberType": "REGULAR"
  }
],
"emailMembers": false,
"reciprocalDelegation": false,
"data": null,
"account": {
  "id": "3vD0Dc9Fh7wQ",
  "data": null,
  "updated": "2017-11-20T19:03:28Z",
  "company": {
    "id": "",
    "data": null,
    "address": null,
    "name": ""
  },
  "licenses": [],
  "logoUrl": "",
  "providers": null,
  "customFields": [],
  "created": "2017-11-20T19:03:28Z",
  "owner": "",
  "name": ""
},
"updated": "2017-11-20T19:03:28Z",
"email": "your_group_email@example.com",
"created": "2017-11-20T19:03:28Z",

```



```
{
  "name": "your_group_name"
}
```

You can also create an empty group, and invite members to join your group at a later date.

HTTP Request

```
POST /api/cs-groups
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{
  "email": "invitee@email.com",
  "name": "REST Developers"
}
```

Response Payload

```
{
  "id": "540b86f9-2d93-4498-bdb4-b7b320540bb6",
  "members": [],
  "emailMembers": false,
  "reciprocalDelegation": false,
  "data": null,
  "account": {
    "id": "3vD0Dc9Fh7wQ",
    "data": null,
    "updated": "2017-11-20T19:04:38Z",
    "company": {
      "id": "",
      "data": null,
      "address": null,
      "name": ""
    },
    "licenses": [],
    "logoUrl": "",
    "providers": null,
  },
}
```

```
    "customFields": [],
    "created": "2017-11-20T19:04:38Z",
    "owner": "",
    "name": ""
  },
  "updated": "2017-11-20T19:04:38Z",
  "email": "invitee@email.com",
  "created": "2017-11-20T19:04:38Z",
  "name": "REST Developers"
}
```

Retrieving Groups

Retrieving your groups can come in handy when you wish to invite members to your group. It is important to note that the group id is required when sending invitations.

HTTP Request

```
GET /api/groups
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Response Payload

```
{
  "count": 1,
  "results": [
    {
      "account": {
        "providers": null,
        "updated": "2016-01-07T18:49:19Z",
        "company": {
          "id": "",
          "address": null,
          "data": null,
          "name": ""
        }
      }
    }
  ]
}
```

```

    },
    "licenses": [],
    "logoUrl": "",
    "customFields": [],
    "created": "2016-01-07T18:49:19Z",
    "owner": "",
    "id": "zRcJCHV3ztIB",
    "data": null,
    "name": ""
  },
  "updated": "2016-01-07T13:58:33Z",
  "email": "group@email.com",
  "members": [
    {
      "userId": "1XyqlkXM9uIX",
      "email": "john.smith@email.com",
      "firstName": "John",
      "lastName": "Smith",
      "memberType": "REGULAR",
      "pending": false
    },
    {
      "userId": "kVooKpofKuUK",
      "email": "mary.doe@email.com",
      "firstName": "Mary",
      "lastName": "Doe",
      "memberType": "REGULAR",
      "pending": false
    }
  ],
  "emailMembers": false,
  "reciprocalDelegation": false,
  "created": "2016-01-07T13:58:33Z",
  "id": "4a8868d7-1968-4172-aedb-0a9dc8edb683",
  "data": null,
  "name": "REST Developers"
}
]
}

```

Adding a Group signer

Once your group has been created, you can now then add a group signer to your package. The code below shows you how to edit the signer block in order to add a group signer.

If you need a comparison to the basic document object creation or if this is the first time creating a package with the REST API, see this [guide](#).

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: multipart/form-data
Authorization: Bearer access_token
```

Request Payload

```
-----WebKitFormBoundary1bN060n7FqP5W04t
Content-Disposition: form-data; name="file"; filename="testDocumentExtraction.pdf"
Content-Type: application/pdf
%PDF-1.5
%ÂµÂµÂµÂµ
1 0 obj
<>>
endobj....
-----WebKitFormBoundary1bN060n7FqP5W04t
Content-Disposition: form-data; name="payload"
{
  "roles": [
    {
      "id": "1945f2e1-3390-4297-bc3b-89af3c92e567",
      "type": "SIGNER",
      "index": 0,
      "signers": [
        {
          "group": {
            "id": "540b86f9-2d93-4498-bdb4-b7b320540bb6",
            "email": "invitee@email.com",
```

```

        "name": "REST Developers"
      },
      "id": "98b9db64-39ca-4b5b-a1eb-629e49c46dec",
      "email": "540b86f9-2d93-4498-bdb4-b7b320540bb6@groups.e-signlive.com",
      "firstName": "REST Developers",
      "lastName": ""
    }
  ],
  "name": "Signer1"
}
],
"status": "DRAFT",
"language": "en",
"documents": [
  {
    "id": "90277a614bf73b783fe2a5e04b68a99d4badf449b33ecfbf",
    "approvals": [
      {
        "id": "cVvJzBDX7lwP",
        "role": "1945f2e1-3390-4297-bc3b-89af3c92e567",
        "fields": [
          {
            "subtype": "FULLNAME",
            "height": 52,
            "extract": false,
            "width": 235,
            "left": 217,
            "top": 512,
            "type": "SIGNATURE"
          }
        ]
      }
    ]
  }
],
"name": "sample_contract"
}
],
"visibility": "ACCOUNT",
"type": "PACKAGE",
"name": "Group Signature Example"
}
-----WebKitFormBoundary1bN060n7FqP5W04t--

```

Response Payload

```
{
  "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI="
}
```

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
email	string	Yes	No	n/a	<code>your_group_email@example.com</code>
name	string	Yes	No	n/a	<code>your_group_name</code>
members					
pending	boolean	No	No	n/a	<code>true / false</code>
email	string	Yes	No	n/a	<code>john.smith@example.com</code>
memberType	string	Yes	No	REGULAR	<code>REGULAR / MANAGER</code>
firstName	string	Yes	No	n/a	<code>John</code>
lastName	string	Yes	No	n/a	<code>Smith</code>

Authenticating Signers

To add an extra layer of security to your online transactions, Sign offers robust and flexible recipient-authentication options. Specifically, you can select various ways of validating the identity of the recipient of an invitation to a transaction before they are permitted to access the transaction's documents.

General Authentication refers to tools built into Conga Sign that enable you to verify the identity of the recipient through SMS, Email, or a custom Question and Answer format.

Knowledge-Based Authentication (KBA) relies on a third-party KBA provider to perform the authentication. That provider is either Equifax US or Equifax Canada.

KBA questions are generated dynamically, based on information in a signer's personal credit report.

i KBA authentication can be used in conjunction with any one of the General authentication methods above.

Using General Authentication

The code below illustrates how to edit the *auth* object for each authentication method. If you need a comparison with basic document-object creation, or if this is your first time creating a package with the Java SDK, see this [Creating a Transaction](#).

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
{  "roles": [ {  "type": "SIGNER",  "index": 0,  "signers": [ {  "auth": {
    "scheme": "CHALLENGE",  "challenges": [ {  "answer": "golf",  "question":
    "What's your favorite sport?",  "maskInput": false  }  ]  },  "email":
    "john.smith@example.com",  "firstName": "John",  "lastName": "Smith"  }  ],
    "name": "Signer1"  }, {  "type": "SIGNER",  "index": 0,  "signers": [ {
    "auth": {  "scheme": "SMS",  "challenges": [ {  "answer": null,  "question":
    "+15515584587",  "maskInput": false  }  ]  },  "email": "mary.doe@example.com",
    "firstName": "Mary",  "lastName": "Doe"  }  ],  "name": "Signer2"  }  ],
    "status": "DRAFT",  "type": "PACKAGE",  "name": "Signer Authentication Example"  }
```

For a complete description of each field, see the Request Payload Table section below.

Response Payload

```
{  "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI="  }
```

Manually sending an SMS code

A new SMS code is generated and sent each time a signer clicks the email link. If for any reason you need to manually send a new SMS code, the following code will do this:

HTTP Request

```
POST /api/cs-packages/{packageId}/roles/{roleId}/sms_notification
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Using KBA

You can also authenticate a signer with KBA. The sample JSON string below shows you how to edit the signer's object with KBA. The `withTimeAtAddress` field can be left empty. The following code will do this:

```
{  "signers":[    {      "delivery":{"email":false},      "email":"signer1@example.com",      "firstName":"John",      "lastName":"Doe",      "auth":{"scheme":"NONE",      "challenges":[ ]},      "knowledgeBasedAuthentication":{"signerInformationForEquifaxUSA":{"firstName":"John",      "lastName":"Doe",      "streetAddress":"2020 Broadway Street",      "city":"New York",      "zip":"12345",      "state":"NY",      "timeAtAddress":5,      "driversLicenseNumber":"1234567890",      "dateOfBirth":"1969-12-09T00:00:00Z",      "socialSecurityNumber":"123456789",      "homePhoneNumber":"1234567890"}    }    },    {      "reassign":false,      "emailMessage":{        "content":""      },      "attachmentRequirements":[ ]    }  ]}
```

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
status	string	Yes	No	DRAFT	DRAFT/SENT/ COMPLETEDARCHIVED/ DECLINED/OPTED_OUT/ EXPIRED
type	string	Yes	No	PACKAGE	PACKAGE/TEMPLATE/ LAYOUT
name	string	Yes	No	n/a	Signer Authentication Example
roles					
type	string	Yes	No	SIGNER	SIGNER/SENDER
index	index	Yes	No	0	0/1/2 ...
name	string	Yes	No	n/a	Signer1
signers					
email	string	Yes	No	n/a	signer1@example.com
firstName	string	Yes	No	n/a	Patty
lastName	string	Yes	No	n/a	Galant
auth					
scheme	string	Yes	No	n/a	CHALLENGE/SMS
challenges					
answer	string	Yes	No	n/a	golf

Property	Type	Editable	Required	Default	Sample Values
question	string	Yes	No	n/a	What's your favorite sport?
maskInput	boolean	Yes	No	false	/ +15515584587

Enforcing Signature Capture

In Conga Sign, signers will only be asked to draw their signature once by default. If the transaction requires multiple signatures on the document, Conga Sign will automatically replicate the first drawn signature for any subsequent signatures. However, you have the option to require your signer to draw their signature on any given capture signature. This topic will focus on enforcing signature capture at the signature level.

Enforcing Capture Signature

The sample request below shows you how to set `enforceCaptureSignature: true` at the approval level.

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

Request Payload

```
-----WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data;
name="file"; filename="enforce capture signature.pdf" Content-Type: application/pdf
%PDF-1.5 %ÂµÂµÂµÂµ 1 0 obj <>> endobj.... -----
WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data; name="payload"
{  "documents": [  {  "approvals": [  {  "fields": [  {  "height": 50,  "id":
"Signature1",  "left": 100,  "page": 0,  "subtype": "CAPTURE",  "top": 100,
```

```

"type": "SIGNATURE", "width": 200 } ], "id": "Signature1", "role":
"signer1" }, { "fields": [ { "height": 50, "id": "Signature2", "left":
100, "page": 0, "subtype": "CAPTURE", "top": 200, "type": "SIGNATURE",
"width": 200 } ], "id": "Signature2", "role": "signer1",
"enforceCaptureSignature": true }, { "fields": [ { "height": 50, "id":
"Signature3", "left": 100, "page": 0, "subtype": "CAPTURE", "top": 300,
"type": "SIGNATURE", "width": 200 } ], "id": "Signature3", "role":
"signer1" } ], "name": "Document1", "id": "Document1" } ], "name":
"test Enforce Capture at Signature Level", "roles": [ { "id": "signer1",
"signers": [ { "email": "signer1@example.com", "firstName": "John", "id":
"signer1", "lastName": "Smith" } ] } ], "type": "PACKAGE", "status":
"SENT" } -----WebKitFormBoundary1bN060n7FqP5W04t--

```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{ "id": "Q6v0Tn_v62G5tfRY43VBE-TEbU4=" }
```



- “By default, “enforceCaptureSignature” is set to false at both the transaction and signature levels. This means you do not need to do anything if you do not want to use this feature.
- Unlike the enforceCaptureSignature true setting in the transaction level, which requires the signer to capture the signature every capture approval, at the signature level enforceCaptureSignature only forces the signer to capture their signature when the approval contains the “true” setting.

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
status	string	Yes	No	DRAFT	DRAFT/SENT/COMPLETED/ARCHIVED/DECLINED/OPTED_OUT/EXPIRED
type	string	Yes	No	PACKAGE	PACKAGE/TEMPLATE/LAYOUT
name	string	Yes	No	n/a	Enforce Capture Signature Example

Property	Type	Editable	Required	Default	Sample Values
documents					
name	string	Yes	No	n/a	sample doc
approvals					
fields					
subtype	string	Yes	No	n/a	FULLNAME/INITIALS/CAPTURE/ MOBILE_CAPTURE/LABEL/ TEXTFIELD/ TEXTAREA/CHECKBOX/DATE/RADIO/ LIST
type	string	Yes	No	n/a	SIGNATURE/INPUT
extract	boolean	Yes	No	false	true/false
height	integer	Yes	No	50	50/100/150...
left	integer	Yes	No	0	50/100/150...
page	integer	Yes	No	0	0/1/2...
top	integer	Yes	No	0	50/100/150...
width	integer	Yes	No	200	50/100/150...
role	string	Yes	No	n/a	Client1

Property	Type	Editable	Required	Default	Sample Values
enforceCaptureSignature	boolean	Yes	No	false	true/false
roles					
id	string	Yes	No	n/a	Client1
name	string	Yes	No	n/a	Client1
type	string	Yes	No	SIGNER	SIGNER/SENDER
signers					
email	string	Yes	Yes	n/a	john.smith@example.com
firstName	string	Yes	Yes	n/a	John
lastName	string	Yes	Yes	n/a	Smith
phone	string	Yes	No	n/a	514-555-8888
id	string	Yes	No	n/a	Client1
company	string	Yes	No	n/a	Acme Inc
title	string	Yes	No	n/a	Managing Director

Custom Transaction Data

Document attributes are customized data related to a document in a transaction, or to all data in a transaction. This data is not interpreted by Conga Sign. As such, the users of this data are free to store and interpret whatever data they want.

Customized data can also be applied at the transaction level. For more information, see [Custom Document Data](#).

The following code shows you how to edit your DocumentPackage object to add package attributes.

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: multipart/form-data
Authorization: Bearer access_token
```

Request Payload

```
-- -- --WebKitFormBoundary1bN060n7FqP5W04t Content - Disposition: form - data;
name = "file";
filename = "testDocumentExtraction.pdf"
Content - Type: application / pdf % PDF - 1.5 % ÂµÂµÂµÂµ 1 0 obj < >> endobj....--
-- --WebKitFormBoundary1bN060n7FqP5W04t Content - Disposition: form - data;
name = "payload" {
  "documents": [{
    "approvals": [{
      "id": "ExampleSignatureId",
      "role": "Signer1",
      "fields": [{
        "page": 0,
        "top": 200,
        "subtype": "LABEL",
        "height": 50,
        "left": 100,
        "width": 200,
        "id": "myLabelField",
        "type": "INPUT",
        "value": "Example label field value"
      }, {
        "page": 0,
        "top": 100,
        "subtype": "FULLNAME",
        "height": 50,
        "left": 100,
        "width": 200,
```

```

        "type": "SIGNATURE",
        "name": "ExampleSignatureId"
    }],
    "name": ""
  }],
  "id": "sample-contract",
  "name": "Test Document"
}],
"status": "DRAFT",
"type": "PACKAGE",
"roles": [{
  "id": "Signer1",
  "type": "SIGNER",
  "signers": [{
    "email": "signer1@example.com",
    "firstName": "John",
    "lastName": "Smith",
    "id": "Signer1"
  }],
  "name": "Signer1"
}],
"name": "Example Package",
"data": {
  "First Name": "Bill",
  "Last Name": "Drake",
  "Signing Order": "1"
}
}
}-- -- --WebKitFormBoundary1bN060n7FqP5W04t--

```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{ "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI=" }
```

Attributes are set on a package level. You will need to get your package JSON in order to retrieve your package attributes.

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
status	string	Yes	No	DRAFT	DRAFT / SENT / COMPLETED / ARCHIVED / DECLINED / OPTED_OUT / EXPIRED
type	string	Yes	No	PACKAGE	PACKAGE / TEMPLATE / LAYOUT
name	string	Yes	Yes	n/a	Document Attributes Example
documents					
name	string	Yes	No	n/a	sample doc
id	string	Yes	No	n/a	sample-contract
approvals					
fields					
subtype	string	Yes	No	n/a	FULLNAME / INITIALS / CAPTURE / MOBILE_CAPTURE / LABEL / TEXTFIELD / TEXTAREA / CHECKBOX / DATE / RADIO / LIST
type	string	Yes	No	n/a	SIGNATURE / INPUT
extract	boolean	Yes	No	false	true / false
height	integer	Yes	No	50	50 / 100 / 150 ...
left	integer	Yes	No	0	50 / 100 / 150 ...

Property	Type	Editable	Required	Default	Sample Values
page	integer	Yes	No	0	0 / 1 / 2 ...
top	integer	Yes	No	0	50 / 100 / 150 ...
width	integer	Yes	No	200	50 / 100 / 150 ...
role	string	Yes	No	n/a	Signer1
id	string	Yes	No	n/a	ExampleSignatureId
roles					
id	string	Yes	No	n/a	Client1
name	string	Yes	No	n/a	Client1
type	string	Yes	No	SIGNER	SIGNER / SENDER
signers					
email	string	Yes	Yes	n/a	john.smith@example.com
firstName	string	Yes	Yes	n/a	John
lastName	string	Yes	Yes	n/a	Smith
phone	string	Yes	No	n/a	514-555-8888
id	string	Yes	No	n/a	Client1
company	string	Yes	No	n/a	Acme Inc.
title	string	Yes	No	n/a	Managing Director
data					

Property	Type	Editable	Required	Default	Sample Values
First Name	string	Yes	No	n/a	Bill
Last Name	string	Yes	No	n/a	Johnson
Signing Order	string	Yes	No	n/a	1

Custom Document Data

Document attributes are customized data related to a document in a transaction, or to all data in a transaction. This data is not interpreted by Conga Sign. As such, the users of this data are free to store and interpret whatever data they want.

Customized data can also be applied at the transaction level. For more information, see [Custom Transaction Data](#).

The following code sample illustrates the simplest way to create a package with customized document attributes. This example illustrates how to create customized attributes with the keys Department, and Employee. The customized attribute data is constructed as a map.

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: multipart/form-data
Authorization: Bearer access_token
```

Request Payload

```
-----WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data;
name="file"; filename="testDocumentExtraction.pdf" Content-Type: application/pdf
%PDF-1.5 %ÃµÃµÃµÃµ 1 0 obj <>> endobj.... -----
WebKitFormBoundary1bN060n7FqP5W04t Content-Disposition: form-data; name="payload"
{  "autocomplete": true,  "documents": [  {  "approvals": [  {  "fields":
[  {  "extract": false,  "height": 50,  "left": 100,  "page": 0,  "subtype":
```

```
"FULLNAME", "top": 100, "type": "SIGNATURE", "width": 200 } ], "role":
"Client1" } ], "data": { "Department": "1806", "Employee": "135526" },
"name": "sample doc" } ], "name": "Document Attributes Example", "roles":
[ { "id": "Client1", "name": "Client1", "signers": [ { "company": "Acme
Inc.", "email": "signer1@example.com", "firstName": "John", "id": "Client1",
"lastName": "Smith", "title": "Managing Director" } ] } ], "trashed":
false, "type": "PACKAGE", "visibility": "ACCOUNT" } -----
WebKitFormBoundary1bN060n7FqP5W04t--
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{ "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI=" }
```

If you want to retrieve your attributes at a later stage, you can do so by simply retrieving your document JSON:

```
GET https://services.congamerge.com/api/cs-packages/{packageId}/documents/
{documentId}
```

Then loop through the data filed property.

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
status	string	Yes	No	DRAFT	DRAFT/SENT/COMPLETED/ARCHIVED/DECLINED/OPTED_OUT/EXPIRED
autoComplete	boolean	Yes	No	true	true/false
type	string	Yes	No	PACKAGE	PACKAGE/TEMPLATE/LAYOUT
name	string	Yes	Yes	n/a	Document Attributes Example
trashed	boolean	Yes	No	false	true/false

Property	Type	Editable	Required	Default	Sample Values
visibility	string	Yes	No	ACCOUNT	ACCOUNT/SENDER
documents					
name	string	Yes	No	n/a	sample doc
approvals					
fields					
subtype	string	Yes	No	n/a	FULLNAME/INITIALS/CAPTURE/ MOBILE_CAPTURE/LABEL/ TEXTFIELD/ TEXTAREA/CHECKBOX/DATE/RADIO/ LIST
type	string	Yes	No	n/a	SIGNATURE/INPUT
extract	boolean	Yes	No	false	true/false
height	integer	Yes	No	50	50/100/150...
left	integer	Yes	No	0	50/100/150...
page	integer	Yes	No	0	0/1/2...
top	integer	Yes	No	0	50/100/150...
width	integer	Yes	No	200	50/100/150...
role	string	Yes	No	n/a	Client1
data					

Property	Type	Editable	Required	Default	Sample Values
Department	string	Yes	No	n/a	1806
Employee	string	Yes	No	n/a	135526
roles					
id	string	Yes	No	n/a	Client1
name	string	Yes	No	n/a	Client1
type	string	Yes	No	SIGNER	SIGNER/SENDER
signers					
email	string	Yes	Yes	n/a	john.smith@example.com
firstName	string	Yes	Yes	n/a	John
lastName	string	Yes	Yes	n/a	Smith
phone	string	Yes	No	n/a	514-555-8888
id	string	Yes	No	n/a	Client1
company	string	Yes	No	n/a	Acme Inc
title	string	Yes	No	n/a	Managing Director


Mobile Signing Ceremony

The ClassicUser Experience supports the following:

- Full Site Signing Experience – This is the signing experience provided to users on desktops, laptops, and tablets. It is also provided to users on mobile phones if they try to use a feature that is not supported by the Mobile Signing Ceremony.

- Mobile Signing Ceremony – This is the signing experience provided to users of Mobile Web Applications after the feature has been activated via our Support Team. For a list of the supported mobile phones, see the Prerequisites below.

As just noted, the Mobile Signing Ceremony does not support all the features of the Full Site Signing Experience.

 An Administrator can enable the redirection of signers to a customized external page when their session expires. To arrange this, please contact our Support Team.

 Please be aware of the following:

- If a user in a Mobile Signing Ceremony tries to use an unsupported feature, they will be directed to the Full Site Signing Experience, where the feature is supported.
- Because of the smaller size of mobile phone screens, some terms that appear in the Mobile Signing Ceremony may differ from those that appear in the Full Site Signing Experience. We do our best to ensure that the signing experience is optimal for all users.
- Although senders can specify any language listed [here](#), the Mobile Signing Ceremony can display information only in English, Français, Deutsch, Português, and Español. If any other language is specified, the ceremony will default to English.

Prerequisites

Before using the Mobile Signing Ceremony, you must meet the following conditions:


- You have activated the Mobile Signing Ceremony via our Support Team.
- Your mobile device uses one of the following Operating Systems:
 - iOS 9+
 - Android KitKat 4.4+
 - BlackBerry OS 10+

Signing Packages

To sign a package on your mobile phone via the Mobile Signing Ceremony:

1. From your mobile phone, click the link to the package in the email message you received.

2. All packages contain an Electronic Disclosure and Signature Consent. Read it, and click Accept.
3. A preview appears of the documents to be signed.
4. At any time, you can click the mobile menu button to display other package information, such as:
 - A list of all documents in the package, and their completion status
 - The Package Description provided by the package creator
 - Conga Sign contact information
 - An Opt-out button: If you tap this button, you are prompted to provide a reason for opting out of signing online. After providing a reason, click OK.
 - A Decline button: If you tap this button, you are prompted to provide a reason for declining to sign the document. After providing a reason, click OK.
5. Tap a Signature Block to zoom in on it.

 If your phone is in Portrait Mode (held upright), you are asked to rotate it to Landscape Mode before capturing your signature.

6. Draw your signature with your fingertip or a stylus. Repeat this step for all Signature Blocks in all documents.
7. After signing is complete, you will receive an email that will enable you to review and/or download the package's documents.

Signer Options

Change Signer

Change Signer enables a signer to delegate the signing of their documents to someone else. The signer will be asked to enter the email address and full name of their delegate, along with an optional email message to the delegate.

In-Person Signing

It is sometimes convenient for a package's documents to be signed by the package owner and all other signers on the same device. For example, a life insurance agent may visit clients in their homes. Using a mobile device such as a tablet, the agent can prepare the insurance package while meeting with the clients, and both agent and clients can sign all documents during their meeting.

Such scenarios are called in-person signing. When that Sign feature is enabled for a package, all the package's documents must be signed on the same device by all signers.

By default, after each signer has completed their Signer Experience, the next signer must: (1) accept an affidavit that confirms their identity; (2) read and accept the Electronic Disclosure and Signature Consent form.

 In-person signing can be disabled by contacting our Support Team.

Position Extraction

Position Extraction enables integrators to automatically extract signatures and fields by placing Text Tags in a document. Conga Sign will analyze the uploaded document and replace every text that matches the Text Tag pattern with the appropriate signature or field.

To add text tags you need an approved document type that already has text tags in it. These text tags must be named in a way that Conga Sign can recognize. For more information on text tag parameters and formats, see the feature overview below.

The text tags used in these topics are shown in the image below.

SAMPLE CONTRACT

Please fill-in the following:

First Name: {{esi:Signer1:textfield:size(200,50)}}

Last Name: {{esi:Signer1:textfield:size(200,50)}}

Address: {{esi:Signer1:textfield:size(200,50)}}

Email: {{esi:Signer1:textfield:size(200,50)}}

Client Signature: {{esi:Signer1:capture:size(200,50)}} **Date:** {{esi:Signer1:SigningDate:size(200,50)}}

The signer for this document is named Signer1. This will be the custom ID used in the code to let Conga Sign know what fields to associate with each signer.

Adding Text Tags

Typically, a JSON string is built dynamically, and not with a giant static string as demonstrated here. However, this example is shown as a giant static string so as to give a good representation of the structure of the JSON you will need to create your transaction.

The JSON below is formatted for readability. In each roles object, you will see that the custom ID is the same as shown in the image of the PDF form above. The documents object also has extract set to true.

You do not have to define signature locations, nor do you have to define who needs to sign the document. This is already taken care of with the ID and the associated text tags from the PDF.

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: multipart/form-data
Authorization: Bearer access_token
```

Request Payload

```
-----WebKitFormBoundary1bN060n7FqP5W04t
      Content-Disposition: form-data; name="file"; filename="sample-
contract.pdf"

      Content-Type: application/pdf
      %PDF-1.5
      %ÃµÃµÃµÃµ
      1 0 obj
      <>>
      endobj....
-----WebKitFormBoundary1bN060n7FqP5W04t
      Content-Disposition: form-data; name="payload"
      {
        "documents": [
          {
            "id": "sample-contract",
            "name": "Test Document",
            "extract": true,
            "extractionTypes": [
              "TEXT_TAGS"
            ]
          }
        ],
        "status": "DRAFT",
```

```

    "type": "PACKAGE",
    "roles": [
      {
        "id": "Signer1",
        "type": "SIGNER",
        "signers": [
          {
            "email": "signer1@example.com",
            "firstName": "John",
            "lastName": "Smith",
            "id": "Signer1"
          }
        ],
        "name": "Signer1"
      },
      {
        "name": "Text Tags Example Package"
      }
    ]
  }
  -----WebKitFormBoundary1bN060n7FqP5W04t--

```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{  "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI=" }
```

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values(s)
status	string	Yes	No	DRAFT	DRAFT / SENT / COMPLETED / ARCHIVED / DECLINED / OPTED_OUT / EXPIRED
type	string	Yes	No	PACKAGE	PACKAGE / TEMPLATE / LAYOUT
name	string	Yes	Yes	n/a	Text Tags Example Package
documents					

Property	Type	Editable	Required	Default	Sample Values(s)
id	string	Yes	No	n/a	sample-contract
name	string	Yes	No	n/a	Test Document
extract	boolean	Yes	No	false	true / false
extractionTypes	string array	Yes	No	[]	["TEXT_TAGS","ACROFIELDS"]
roles					
id	string	Yes	No	n/a	Signer1
name	string	Yes	No	n/a	Signer1
type	string	Yes	No	SIGNER	SIGNER / SENDER
signers					
email	string	Yes	Yes	n/a	signer1@example.com
firstName	string	Yes	Yes	n/a	John
lastName	string	Yes	Yes	n/a	Smith
id	string	Yes	No	n/a	signer1

Feature Overview

Position Extraction enables integrators to automatically extract signatures and fields by placing Text Tags in a document. Conga Sign will analyze the uploaded document, and

replace every text that matches the Text Tag pattern with the appropriate signature or field.

Best Practices for Using Text Tags

Here are some best practices that you should keep in mind when using text tags:

- When you have multiple attributes specified in text tags, try using commas to separate them, like below:
 - `{{esl:Signer1:textfield:size(40,15),Maxlen(3)}}`
 A more complicated example could look like:
 - `{{esl_checkbox1:Signer1:checkbox:offset(0,-15),size(275,25),Maxlen(40)}}`
- Try using a smaller font for your text tags. This will save space and help you avoid affecting your content layout.
- Make sure your text tag is within one line.

Here is some sample code that incorporates these suggestions:

```
{{esl_optionA:Signer1:Radio:Group("Frequency"),Value("X"),size(10,10),offset(40,-8)}}
{{esl_optionB:Signer1:Radio:Group("Frequency"),Value(""),size(10,10),offset(40,-8)}}
{{esl_optionC:Signer1:Radio:Group("Frequency"),Value(""),size(10,10),offset(40,-8)}}
```

- ✓ If you are having problems putting your text tags in their proper location, try using the offset attribute. This is especially helpful when using a Microsoft Word document.

Text Tag Syntax

The following line illustrates the syntax of a Text Tag:

```
{{Xesl[_fieldName]:roleName:fieldType[:parameter1,parameter2,...]}}
```

- ✓ When you have multiple attributes specified in text tags, try using commas to separate them, like below:
 - `{{esl:Signer1:textfield:size(40,15),Maxlen(3)}}`
 A more complicated example could look like:
 - `{{esl_checkbox1:Signer1:checkbox:offset(0,-15),size(275,25),Maxlen(40)}}`

- ⚠ Field names are alphanumeric. They cannot contain special characters other than the underscore (_).

Note from this line that:

- The beginning and end of a Text Tag have double curly brackets. Curly brackets should not appear anywhere else in a Text Tag.
- The third character – the `x` – is a placeholder for an optional character. That character can be either a question mark (`?`) or an asterisk (`*`). An asterisk indicates that an Input Field is required. A question mark indicates that an Input Field is optional. The question mark is the default value – i.e., if neither character is present, an Input Field will be optional. Example: `{{*esl:Signer1:textfield}}`
- `fieldName` is the name of the field that will be created by the Text Tag (a Signature, an Autofield, or an Input Field) – see Text Tag Types below. If specified, this parameter will follow the “`esl`” prefix. Field names are alphanumeric. They cannot contain special characters other than the underscore (`_`). If you plan to retrieve the value entered during the Signer Experience, enter a unique name per field. Example: `{{esl_SignerAutograph:Signer1:Signature}}`
- `roleName` is the name of the role associated with the intended signer. To find the role name you need to use, see Role Names in Text Tags below.
- `fieldType` is one of the field types described in Text Tag Types below.
- `parameter1` and `parameter2` are parameters described in Text Tag Parameters below.



Please be aware of the following:

- A Text Tag cannot appear on multiple lines.
- If a list of parameters is present, it must be a comma-separated list.

The following line illustrates the shortest valid syntax for a Text Tag:

```
{{esl:roleName:fieldType}}
```

Role Names in Text Tags

If you create a Recipient signer or a Group signer using Conga Sign, the role name of the signer is `Signer#`, where `#` starts at 1 and increases by one for each signer. The lowest available `#` is used for each new signer. Thus, if you delete the signer with role name `Signer1` and then add a new signer, the new signer's role name will be `Signer1`.

If you create a placeholder signer, and name it `Client`, then `Client` will be the role name. `Client` will remain as the role name when that placeholder is later replaced with a specific recipient or group.

If you create a signer using the SDK, and assign a Custom ID to that signer, that ID will be the signer's role name. If you do not assign a Custom ID to the signer, one is generated for the signer using the same rules as when using the Conga Sign web UI.

Text Tag Types

Signatures

A particular Signature Text Tag can be any of the following field types:

- **Signature** – This Signature Block is Click-to-Sign. The signer's name will be stamped on the clicked block.
- **Initials** – This Signature Block is Click-to-Initial. The signer's initials will be stamped on the clicked block.
- **Capture** – The signer clicks this Signature Block, and draws their signature using a mouse or another input device. The signer can also choose to sign on a mobile device such as a smartphone if the sender has mobile capture enabled on their account. The drawing is then stamped on the Signature Block.
- **Mobile_Capture** – To e-sign this Signature Block, the signer will receive a link via email that redirects them to open the document on their mobile phone. They are then required to draw their signature using their finger or a stylus. The drawing of the signature is then stamped on the block.

Autofields

A particular Autofield Text Tag can be any of the following field types:

- **SignerName** – At the time of signing, the system automatically populates this field with the full name of the signer.
- **Signature** – If the signer is required to sign multiple signature fields, this field allows you to bind different text tags for each required signature.
- **SignerTitle** – If the signer's title is available, the system automatically populates this field with that title at the time of signing. Otherwise, the field is left blank.
- **SignerCompany** – If the name of the signer's company is available, the system automatically populates this field with that name at the time of signing. Otherwise, the field is left blank.
- **SigningDate** – At the time of signing, the system automatically populates this field with the current date.

Input Fields

A particular Input Field Text Tag can be any of the following field types:

- **TextField** – This box accepts any text entered by the signer prior to signing.
- **TextArea** – This is similar to the TextField type, in that it provides an area where free-form text can be entered by signers. However, unlike Text Fields, it provides an automatic wraparound. Each Text Area can accept up to 4000 characters.

- List – This drop-down menu provides the ability to select one of many options.
- Radio – Radio buttons also provide the ability to select one of many options.
- Checkbox – Prior to signing, the signer can either select or clear this simple check box.
- Label – The package sender can add a Label Field to embed text in a document. This is a read-only field with a value that will be simply stamped on the PDF. During the Signer Experience, the Label Field is displayed as non-editable text.

Text Tag Parameters

When building a Text Tag, the system uses the parameters described in the following table.

- Offset and Size are used for all three Text Tag types.
- The table's other parameters are used only for Input Field types.
- All parameters in the following table are optional.

i If quotes are required in any parameter in the following table, you must use straight quotes. You cannot use curly quotes.

Parameter	Description	Examples
Offset	<p>Offset (in points x 1.3) to be applied in positioning the field relative to the top left corner of the Text Tag</p> <p>If unspecified, the field will be inserted at the exact position of the Text Tag.</p> <p>Values can be positive or negative. To move the extracted field more to the right and lower, use positive offset values.</p>	<pre>{{esl:Signer3:initials:offset(20,40)}}</pre> <pre>{{esl:Signer3:initials:offset(-20,-40)}}</pre>

Parameter	Description	Examples
Size	Size of the field (in points x 1.3). If unspecified, the system's default values will be used. Valid values must be positive.	<code>{{esl:Signer1:capture:size(200,50)}}</code>
Group	The radio group to which the field will belong If unspecified, the system will use its default radio group.	<code>{{esl:Signer1:Radio:Group("MyGroup"),Value("X")}}</code>
Options	The list of values available for a List field	<code>{{esl_colour:signer1:list:options("Red", "Blue", "Green")}}</code>

Parameter	Description	Examples
Value	The field's default value For Radio and Checkbox field types: (1) if no value is specified, the option will by default be deselected/unchecked; (2) to have the option selected/checked by default, the value " X " must be specified; (3) if any other value or the empty string is specified, the option will by default be deselected/unchecked.	<pre>{{esl:Signer1:label:value("This is a test label")}}</pre> <pre>{{esl_paymentMethod:signer1:textfield:value("Please enter your preferred payment method")}}</pre> <pre>{{esl:Signer1:checkbox:value("X")}}</pre>
Maxlen	The maximum permitted value for the field	<pre>{{esl_paymentMethod:signer1:textfield:Maxlen(200)}}</pre>

 You can't add Custom Fields or Notary Fields via Text Tags.

Signer with Multiple Signatures

Suppose a document extraction uses Text Tags, and the document has more than one Signature Field for a given signer. This requires the specification of appropriate metadata.

Signatures

The following metadata enable a document to have two signers, each with two Signature Fields:

```
Signer1: {{esl_signer1Sig1:Signer1:Signature}}
```

```
Signer1: {{esl_signer1Sig2:Signer1:Signature}}
```

```
Signer2: {{esl_signer2Sig1:Signer2:Signature}}
```

```
Signer2: {{esl_signer2Sig2:Signer2:Signature}}
```

Autofields

If any of the autofields `SignerName`, `SignerTitle`, `SignerCompany`, or `SigningDate` is bound to any of a signer's multiple Signature Fields, the autofield's metadata must be updated for the specific Signature Field to which it is bound (not necessarily bound to the signer's first Signature Field).

The following metadata enable a signer's multiple signatures to have different date stamps in the same document:

```
{{esl:Signer1:SigningDate:signature("signer1Sig2")}}
```

```
{{esl:Signer1:SigningDate:signature("signer1Sig1")}}
```

```
{{esl:Signer2:SigningDate:signature("signer2Sig1")}}
```

```
{{esl:Signer2:SigningDate:signature("signer2Sig2")}}
```

```
{{esl:Signer1:SignerName:signature("signer1Sig2")}}
```

```
{{esl:Signer1:SignerName:signature("signer1Sig1")}}
```

```
{{esl:Signer2:SignerName:signature("signer2Sig1")}}
```

```
{{esl:Signer2:SignerName:signature("signer2Sig2")}}
```

Input Fields

If an Input Field is bound to any of a signer's multiple Signature Fields, that Input Field's metadata must be updated for the specific Signature Field to which it is bound (not necessarily bound to the signer's first Signature Field).

The following metadata enable a signer's multiple signatures to have different Text Field and Text Area bindings in the same document:

```
{{esl:Signer1:TextField:signature("signer1Sig2")}}
```

```
{{esl:Signer1:TextField:signature("signer1Sig1")}}
```

```
{{esl:Signer2:TextField:signature("signer2Sig1")}}
```

```
{{esl:Signer2:TextField:signature("signer2Sig2")}}
```

```
{{esl:Signer1:TextArea:signature("signer1Sig2")}}
```

```
{{esl:Signer1:TextArea:signature("signer1Sig1")}}
```

```
{{esl:Signer2:TextArea:signature("signer2Sig1")}}
```

```
{{esl:Signer2:TextArea:signature("signer2Sig2")}}
```

Signer Experience Settings

The Signer Experience can be customized. By customizing the Signer Experience you can seamlessly integrate the signing process within your own site.

Document Package Settings

The following table provides a brief description of the Signer Experience settings that can be customized. Note that it is not required to specify any of these settings. If a setting is not defined, the default values shown below will be used.

Setting Name	Document Package Settings Method	Description
In-person	withInPerson	Defines whether all documents in a package have to be signed on the same device.
	withoutInPerson (default)	
Owner in-person dropdown	hideOwnerInPersonDropdown	When in-person is enabled, defines whether the package owner is present in the signer dropdown menu.
	showOwnerInPersonDropdown (default)	
First Affidavit	disableFirstAffidavit	When in-person is enabled, defines whether the first affidavit page is shown when switching signers from the signer dropdown menu.
	enableFirstAffidavit (default)	

Setting Name	Document Package Settings Method	Description
Second Affidavit	disableSecondAffidavit	When in-person is enabled, defines whether the second affidavit page is shown when switching signers from the dropdown menu.
	enableSecondAffidavit (default)	
Decline	withDecline	Defines whether a signer can decline the document package.
	withoutDecline (default)	
Decline Reason	withDeclineReason	Defines a predefined decline reason for when a signer refuses to sign a package.
Decline Other	withDeclineOther (default)	Allows a signer to enter a different reason for refusing to sign a package.
	withoutDeclineOther	Prevents a signer from entering a different reason for refusing to sign a package.
Language Dropdown	withLanguageDropdown (default)	Defines whether the language dropdown menu from the Signer Experience page is displayed.
	withoutLanguageDropdown	
Display Help Page	withShowNseHelp	Defines whether a help page will be displayed, where you can provide descriptive instructions, external site links, or contact information that the signer can use for additional information or instructions.
	withoutShowNseHelp (default)	

Setting Name	Document Package Settings Method	Description
Hand-over link	withHandOverLinkHref	Replaces the continue button in the Signer Experience. If replaced, the signer will be redirected to a URL you specify.
Hand-over link Tooltip	withHandOverLinkHrefTooltip	Defines the text that will appear when hovering over the handover button.
HandOver Link Parameters	withHandOverLinkParameters(Sets.newHashSet("PACKAGE", "SIGNER", "STATUS"))	Defines which parameters will be appended to the Handover URL. The available options are ["PACKAGE", "SIGNER", "STATUS"]. If left empty, no parameters will be appended.
HandOver Link Auto-Redirect	withHandOverLinkAutoRedirect	Automatically redirects the signer to the handover URL page once the signer has completed signing.
	withoutHandOverLinkAutoRedirect (default)	
Capture text	withCaptureText (default)	Defines whether the signed documents are stamped with the date, time, and signer's name at each location they were signed.
	withoutCaptureText	
Navigator	withNavigator (default)	Shows the Navigator. The Navigator directs the signer to the next available signature.
	withoutNavigator	
Title	withTitle (default)	Shows the title.
	withoutTitle	
Progress Bar	withProgressBar (default)	Shows the progress bar.

Setting Name	Document Package Settings Method	Description
	withoutProgressBar	
Download Button	withDocumentToolBarDownloadButton (default)	Defines whether the download button is available.
Show Logo In iFrame	withShowNseLogoInIframe	Defines whether to display a logo when the New Signer Experience is presented in an iFrame,
	withoutShowNseLogoInIframe (default)	
Show Overview Page	withShowNseOverview (default)	Defines whether to display an overview page before a signer can begin to sign the package.
	withoutShowNseOverview	
Left Menu Expand	withLeftMenuExpand	Defines whether the left menu will be expanded by default.
	withoutLeftMenuExpand (default)	
	withoutDocumentToolBarDownloadButton	
Dialog on complete	withoutDialogOnComplete (default)	
	withDialogOnComplete	Defines whether the completion dialog, which asks the signers if they want to review the documents or leave the system, is shown after signing.

Customizing the Signer Experience

The sample JSON below shows you how to edit the settings object. Each method has been described above, in the Document Package Level Settings section.

If you need a comparison to the basic object creation procedure, or if this is the first time creating a transaction, see [Creating and Sending a Transaction](#).

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer access_token
```

```
{  "name": "Customizing Signer Experience",  "settings": {    "ceremony": {      "events": {        "complete": {          "dialog": true,          "redirect": ""        },        "inPerson": true,        "declineButton": true,        "declineReasons": [],        "disableDeclineOther": false,        "disableDownloadForUncompletedPackage": false,        "disableFirstInPersonAffidavit": true,        "disableInPersonAffidavit": false,        "disableOptOutOther": false,        "disableSecondInPersonAffidavit": true,        "documentToolBarOptions": {          "downloadButton": true        },        "handOver": {          "title": "You will be redirected to Google homepage",          "href": "http://www.google.com",          "text": "Exit to site"        },        "hideCaptureText": false,        "hideLanguageDropdown": true,        "hidePackageOwnerInPerson": true,        "maxAuthFailsAllowed": 3,        "optOutButton": true,        "optOutReasons": [          "Decline terms."        ],        "style": null,        "layout": {          "footer": {},          "navigator": true,          "brandingBar": {            "logo": {              "src": "http://www.logomaker.net/images/common/company-logo8.gif",              "link": ""            },            "header": {              "feedback": true,              "globalActions": {                "confirm": true,                "download": false,                "hideEvidenceSummary": false,                "saveAsLayout": false              },              "titleBar": {                "title": true,                "progressBar": true              },              "breadcrumbs": false,              "globalNavigation": false,              "sessionBar": true            }          }          },          "type": "PACKAGE",          "status": "DRAFT"        }      }    }
```

Response Payload

```
{  "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI=" }
```

Text Anchors

Text anchors allow you to position a field or signature based on the text in your document.

Adding Text Anchors

In this example, text anchors are used to add a signature, signing date, and signer name for each signer.

If you need a comparison to the basic object creation procedure, or if this is the first time creating a transaction, see [Creating and Sending a Transaction](#).

Below is the JSON that will create your document package with text anchors. Please note that Sign's API convention allows only for only one signature field (for example, type: SIGNATURE) per approval in the approvals object. A second signature for a given role would need to be added as a separate approval.

Text Anchor Parameters

The following table provides a brief description of every parameter that can be passed when building your text anchors. Note the following:

- Text anchors are case-sensitive, and thus when using text anchors, the search is performed in a case-sensitive fashion.
- If a parameter is not specified, the default value below will be used.

Parameter	Description	Required	Default Value
Anchor Text	The exact string that will be searched for in your document.	Yes	Not applicable
Anchor Point	The corner of the specified character to use as the base for calculating the position. Available values: TOPLEFT, TOPRIGHT, BOTTOMLEFT, and BOTTOMRIGHT.	No	TOPLEFT

Parameter	Description	Required	Default Value
Index	The occurrence of the string. For example, a value of 1 will skip the first occurrence and use the second instance to calculate position.	Yes	Not applicable
Character Index	The index of the character within the anchor text that will be used to calculate the position.	Yes	Starting from 0
Left Offset	The offset applied to the final x-coordinate.	Yes	Refer to the Offset Axis
Top Offset	The offset applied to the final y-coordinate.	Yes	Refer to the Offset Axis
Height	The height of the field or signature.	Yes	Not applicable
Width	The width of the field or signature.	Yes	Not applicable

Best Practices for Using Text Anchors

If you are having problems putting your text anchors in their proper location, try using the offset attribute.

HTTP Request

```
POST /api/cs-packages
```

HTTP Headers

```
Accept: application/json
Content-Type: multipart/form-data
Authorization: Bearer access_token
```

Request Payload

```
-----WebKitFormBoundary1bN060n7FqP5W04t    Content-Disposition: form-data;
name="file"; filename="Sample Contract.pdf"    Content-Type: application/pdf
```

```
%PDF-1.5 %ÃµÃµÃµÃµ 1 0 obj <>> endobj.... -----
WebKitFormBoundary1bN060n7FqP5W04t {  "roles": [  {  "id": "client",  "type":
"SIGNER",  "index": 1,  "signers": [  {  "firstName": "John",  "lastName":
"Smith",  "email": "john.smith@example.com"  }  ],  "name": "client"  },  {
"id": "contractor",  "type": "SIGNER",  "index": 2,  "signers": [  {
"firstName": "Mary",  "lastName": "Doe",  "email": "mary.doe@example.com"  }  ],
"name": "contractor"  }  ],  "documents": [  {  "approvals": [  {  "fields":
[  {  "type": "SIGNATURE",  "extract": false,  "extractAnchor": {  "text":
"Signature of the Client",  "index": 0,  "width": 150,  "height": 40,
"anchorPoint": "TOPLEFT",  "characterIndex": 0,  "leftOffset": 0,  "topOffset":
-50  },  "left": 0,  "subtype": "FULLNAME",  "top": 0  },  {  "type": "INPUT",
"binding": "{signer.name}",  "extract": false,  "extractAnchor": {  "text":
"(hereafter referred to as",  "index": 0,  "width": 150,  "height": 20,
"anchorPoint": "TOPRIGHT",  "characterIndex": 0,  "leftOffset": -175,  "topOffset":
-5  },  "left": 0,  "subtype": "LABEL",  "top": 0  },  {  "type": "INPUT",
"binding": "{approval.signed}",  "extract": false,  "extractAnchor": {  "text":
"Date",  "index": 0,  "width": 75,  "height": 40,  "anchorPoint": "TOPRIGHT",
"characterIndex": 4,  "leftOffset": 10,  "topOffset": -30  },  "left": 0,
"subtype": "LABEL",  "top": 0  }  ],  "role": "client"  },  {  "fields": [  {
"type": "SIGNATURE",  "extract": false,  "extractAnchor": {  "text": "Signature of
the Contractor",  "index": 0,  "width": 150,  "height": 40,  "anchorPoint":
"TOPLEFT",  "characterIndex": 0,  "leftOffset": 0,  "topOffset": -50  },  "left":
0,  "subtype": "FULLNAME",  "top": 0  },  {  "type": "INPUT",  "binding":
"{signer.name}",  "extract": false,  "extractAnchor": {  "text": "(hereafter
referred to as",  "index": 1,  "width": 150,  "height": 20,  "anchorPoint":
"TOPRIGHT",  "characterIndex": 0,  "leftOffset": -175,  "topOffset": -5  },
"left": 0,  "subtype": "LABEL",  "top": 0  },  {  "type": "INPUT",  "binding":
"{approval.signed}",  "extract": false,  "extractAnchor": {  "text": "Date",
"index": 1,  "width": 75,  "height": 40,  "anchorPoint": "TOPRIGHT",
"characterIndex": 4,  "leftOffset": 10,  "topOffset": -30  },  "left": 0,
"subtype": "LABEL",  "top": 0  }  ],  "role": "contractor"  }  ],  "name":
"Sample Contract"  }  ],  "name": "Text Anchor Extraction Example REST API",
"type": "PACKAGE",  "language": "en",  "autoComplete": true,  "status": "DRAFT"
}  -----WebKitFormBoundary1bN060n7FqP5W04t--
```

For a complete description of each field, see the Request Payload Table below.

Response Payload

```
{ "id": "9sKhW-h-qS9m6Ho3zRv3n2a-rkI=" }
```

Request Payload Table

Property	Type	Editable	Required	Default	Sample Values
status	string	Yes	No	DRAFT	DRAFT / SENT / COMPLETED / ARCHIVED / DECLINED / OPTED_OUT / EXPIRED
autoComplete	boolean	Yes	No	true	true / false
type	string	Yes	No	PACKAGE	PACKAGE / TEMPLATE / LAYOUT
name	string	Yes	Yes	n/a	Text Anchor Extraction Example REST API
language	string	Yes	Yes	en	en / fr / de ...
documents					
name	string	Yes	No	n/a	Sample Contract
approvals					
role	string	Yes	No	n/a	client
fields					
type	string	Yes	Yes	n/a	SIGNATURE / INPUT
extract	boolean	Yes	No	false	true / false

Property	Type	Editable	Required	Default	Sample Values
subtype	string	Yes	Yes	n/a	FULLNAME / INITIALS / CAPTURE / MOBILE_CAPTURE / LABEL / TEXTFIELD / TEXTAREA / CHECKBOX / DATE / RADIO / LIST
binding	string	Yes	No	null	{null} / {approval.signed}/ {signer.title} / {signer.name} / {signer.company}
left	integer	Yes	No	0	0 / 10 / 20 ...
top	integer	Yes	No	0	0 / 10 / 20 ...
extractAnchor					
text	string	Yes	Yes	n/a	Signature of the Client
anchorPoint	string	Yes	Yes	n/a	TOPLEFT / TOPRIGHT / BOTTOMLEFT / BOTTOMRIGHT
index	integer	Yes	No	0	0 / 1 / 2 ...
width	integer	Yes	No	200	150
characterIndex	integer	Yes	No	0	0
height	integer	Yes	No	50	40

Property	Type	Editable	Required	Default	Sample Values
leftOffset	integer	Yes	No	0	40
rightOffset	integer	Yes	No	0	-10
roles					
id	string	Yes	No	n/a	client
index	integer	Yes	No	0	1 / 2 / 3 ...
name	string	Yes	No	n/a	client
type	string	Yes	No	SIGNER	SIGNER / SENDER
signers					
email	string	Yes	Yes	n/a	john.smith@example.com
firstName	string	Yes	Yes	n/a	John
lastName	string	Yes	Yes	n/a	Smith
id	string	Yes	No	n/a	client

Conga Sign Features by Release

Review the latest Conga Sign Features by Release document.

- [Features by Release](#)

Features by Release

This document contains an overview of features introduced in each major release of Conga Sign. For more information, see [Conga Sign Features by Release](#).

Conga Customer Community & Learning Center Resources

The Conga Customer Community is your one-stop shop for success!

After registering as a new member, you'll gain access to a wide variety of resources, from a [personalized onboarding checklist](#) to [free expert-led webinars](#), to our [thought-leadership blog](#)! It's also your portal to manage your Conga account, access the Install Center, and submit support tickets.

You can also access the [Conga Learning Center](#). All customers get access to a limited catalog of getting started courses. Consider upgrading to the Conga Learning Pass to unlock the premium training subscription.

Ready to get started?

Log into the [Conga Customer Community](#) with your credentials.

Not yet registered? No problem. Set up an account to receive your login credentials via our [registration](#) page.

After you log in, there are 2 ways to access the Conga Learning Center:

- On the main page, click the Learning Center tile.
- Navigate to the "Resources" dropdown menu at the top, then click the "Learning Center" option.

Conga Copyright Disclaimer

Copyright © 2023 Apttus Corporation ("Conga") and/or its affiliates. All rights reserved.

No part of this document, or any information linked to or referenced herein, may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written consent of Conga. All information contained herein is subject to change without notice and is not warranted to be error free.

This document may describe certain features and functionality of software that Conga makes available for use under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not, in any form, or by any means, use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part of the software. Reverse engineering, disassembly, decompilation of, or the creation of derivative work(s) from, the software is strictly prohibited. Additionally, this document may contain descriptions of software modules that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document.

U.S. GOVERNMENT END USERS: Conga software, including any operating system(s), integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

Neither the software nor the documentation were developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Conga and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Apttus, AI Analyze, Conga, Conga AI, Conga AI Discover, Conga Batch, Conga Collaborate, Conga Composer, Conga Conductor, Conga Connect, Conga Courier, Conga Grid, Conga Mail Merge, Conga Merge, Conga Orchestrate, Conga Sign, Conga Trigger, Digital Document Transformation, True-Up, and X-Author are registered trademarks of Conga and/or its affiliates.

The documentation and/or software may provide links to web sites and access to content, products, and services from third parties. Conga is not responsible for the availability of, or any content provided by third parties. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Conga is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Conga is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

For additional resources and support, please visit <https://community.conga.com>.