brightgen

BrightSIGN Implementation Guide

Version 3.0

BrightSIGN

1 Overview

Signature Capture is now BrightSIGN!

BrightSIGN is a Lightning Component that allows a user to draw their signature into an HTML5 canvas element via a browser or mobile device, convert the captured drawing to an image and store that image against a Salesforce record.

This guide is for version 2.0 of the SignatureCapture managed package.

2 Configuration

BrightSIGN is a public Aura Lightning Component that may be used programmatically or as part of a low/no-code Lightning Flow. It implements the interfaces required to be used in drag and drop tools such as the Lightning App Builder and is tested on the Salesforce Lightning desktop and mobile applications.

The behaviour of BrightSIGN is configurable via attributes which are available directly for programmatic applications, and as design attributes for low/no-code tools.

The component supports the following attributes:

Parameter	Description	Default
`startMsg`	Message to display to the user to start the capture process	Click the 'Capture Signature' button to begin
`enterMsg`	Text to display above the BrightSIGN canvas	Sign here please
`completeMsg`	Text to display above the image of the captured signature once it has been saved to the server	Here is the captured signature
`recordId`	The id of the Salesforce record to attach the signature image to	None - this will be supplied by the Salesforce platform when BrightSIGN is embedded in a record page.
`saveAttachment`	Should the image be saved as an attachment (true) or a file (false)	true
`width`	Width of the BrightSIGN canvas	100%
`height`	Height of the BrightSIGN canvas	300px
`captureButtonLabel`	Label for the capture signature button	Capture Signature

saveButtonLabel	Label for the save button	Save
clearButtonLabel	Label for the clear button	Clear
skipToCapture	Skip straight to capturing a signature without prompting the user or waiting for a button click	false
fileName	File name for BrightSIGN	Signature.png
backgroundColour	Background colour for image	white

3 Usage

3.1 Aura Lightning Component/Application

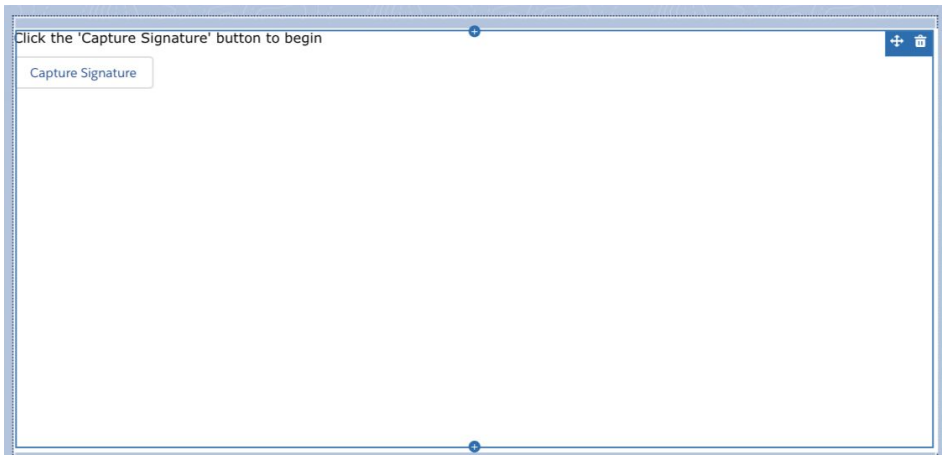
BrightSIGN can be embedded programmatically inside an Aura component or Application as follows:

```
<BGSIGCAP:SignatureCapture
                    recordId="{!v.recordId}"
                    startMsg="Click the button to get started"
                    enterMsg="Please sign below"
                    completeMsg="Here is the signature you entered"
                    width="50%"
                    height="200px"/>
```

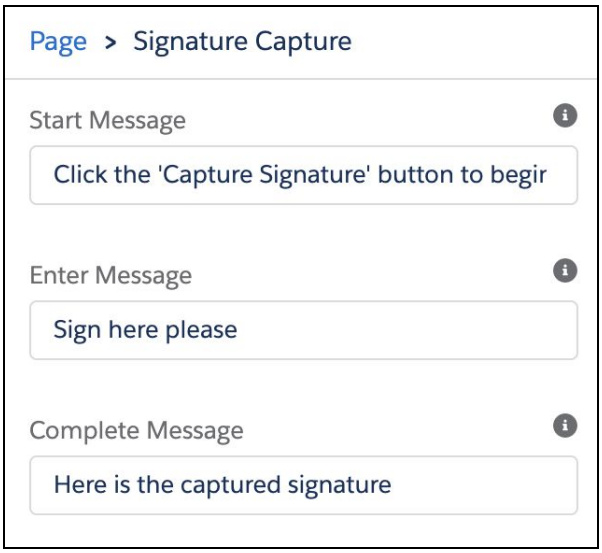
3.2 Lightning App Builder

BrightSIGN appears in the list of available components in the app builder, in the Custom - Managed section at the bottom of the Components list of the left hand side of the page.

Dragging the component onto the canvas displays the component with the default attribute values.



The values may be changed by clicking on the component, which opens the attribute editor on the right hand side of the page.

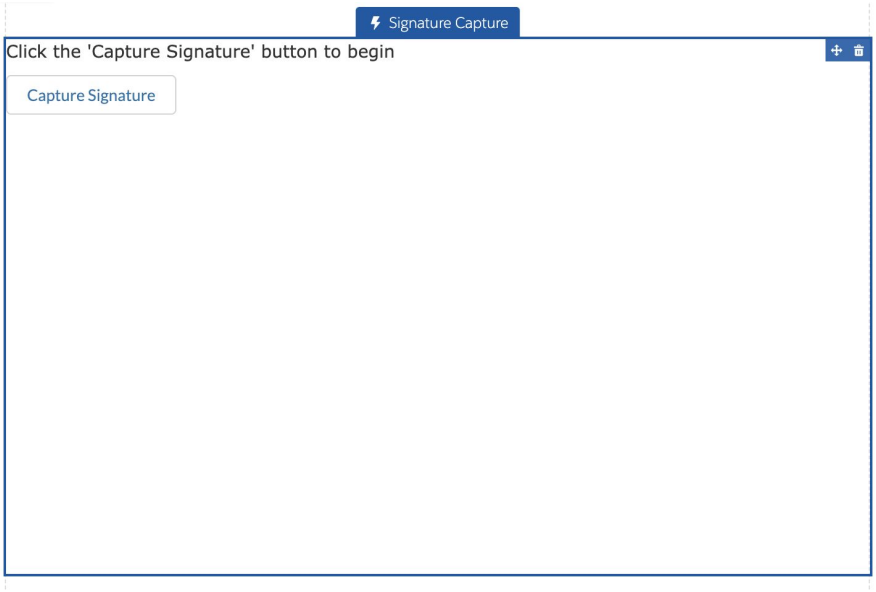


If the component is added to a Record page, the Salesforce Platform will automatically populate the `recordId` attribute with the ID of the record being viewed. For Home or Application pages, the record id will need to be manually entered.

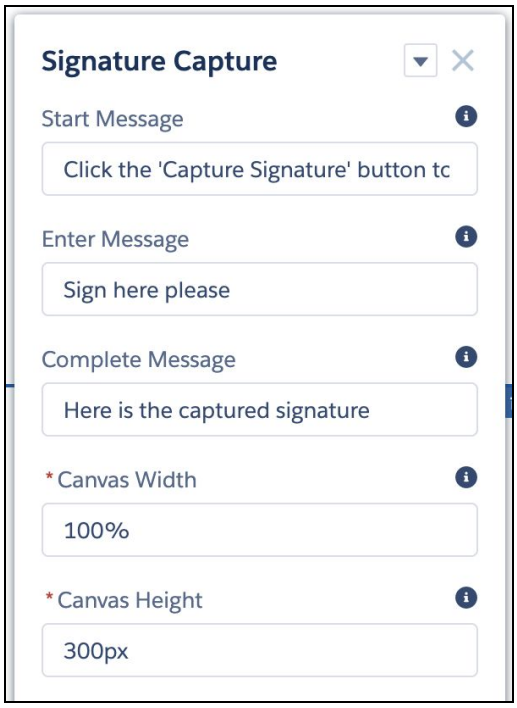
3.3 Community Builder

BrightSIGN appears in the list of available components in the app builder, in the Custom - Managed section at the bottom of the Components list of the left hand side of the page.

Dragging the component onto the builder canvas displays the component with the default attribute values:



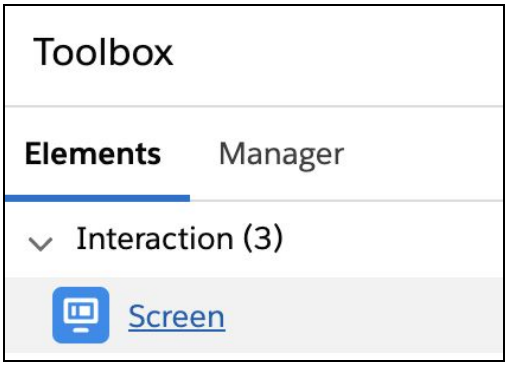
Clicking on the component opens the attribute editor on the right hand side of the page:



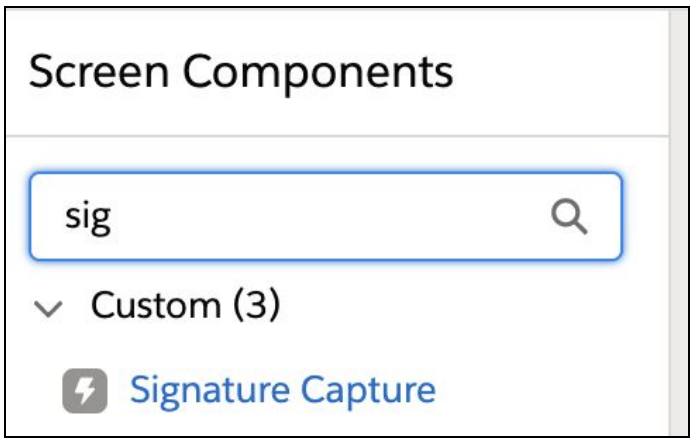
If the component is added as a direct descendant of a Record page, the Salesforce Platform will automatically populate the `recordId` attribute with the ID of the record being viewed. If nested inside another component in a Record page (a tab, for example) then you will need to specify the recordId attribute as : `{!recordId}`

3.4 Lightning Flow

To use BrightSIGN in a Lightning Flow, drag a screen element from the toolbox to the canvas:

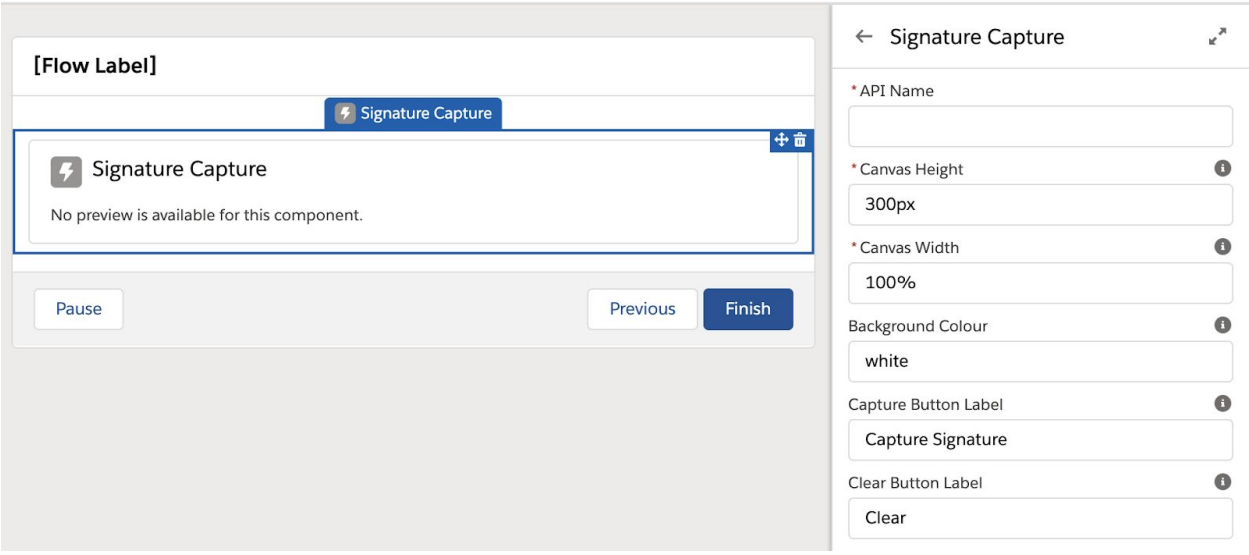


In the resulting dialog, type 'sig' into the search box on the left hand side:

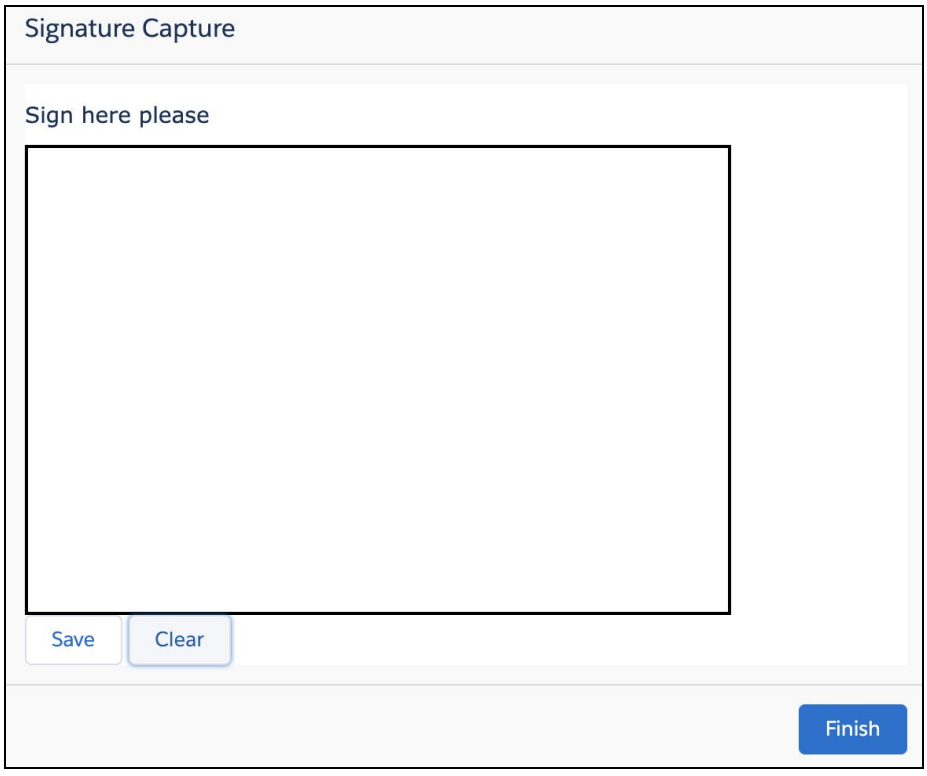


Choose BrightSIGN from the list of results and this adds the BrightSIGN to the screen with the default attributes.

Specify the API name and change any other attributes as desired.



Launching the flow shows the BrightSIGN component:



Note that by default the screen component displays the navigation footer - see the samples repo for a solution that handles the navigation automatically.

4 Extending

The functionality provided by BrightSIGN can be extended by wrapping it inside other Aura components. As the extended functionality tends to be specific to a customer scenario, these wrapper components are not added to the BrightSIGN package on the app exchange. Instead they are made available via a samples repository on Github at :

https://github.com/keirbowden/sigcapapp

These components are suitable for deployment to a subscriber org (the Salesforce instance that you have installed BrightSIGN) and are open source so may be copied and extended under the permissive terms of the MIT License.

Many samples are accompanied by a blog post that explains how it works in detail.

5 Signature Captured Event

When a signature is successfully captured against a record, the `SignatureCapturedEvt` Aura application event is fired. This contains the Id of the record in the `recordId` parameter. The `SigCapNotifier` component in the samples repository demonstrates how to consume this event to display a toast message.

6 Multiple BrightSIGN Components on a Single Page

New in version 2.0

Version 2.0 of BrightSIGN introduces the ability to embed multiple components to capture a signature on a single page. As this would have required significant changes to the existing component architecture, a new component has been produced to handle this scenario named `SignatureCaptureMulti`. This can be used in all of the scenarios shown in the Usage section and supports the same attributes as the SignatureCapture component. It also supports an additional attribute of `name` - a text string that uniquely identifies the component on the page. It is recommended that the name of the captured Attachment/File (the `fileName` attribute) is changed from the default to a unique value for each instance of `SignatureCaptureMulti` added to the page.

When SignatureCaptureMulti fires the `SignatureCapturedEvt` event, it includes a `name` parameter that identifies the instance that successfully captured the signature against the record.

6 Troubleshooting

Problem: When saving the signature, users receive the error message :
'You do not have update permission on the parent sobject type for record with id .'

Solution: Assign the user explicit edit permission to the parent sobject type you are adding the Attachment/File to via their Profile or a Permission Set.

Explanation: In order to pass the security review, any Apex code must demonstrate that it is respecting the security configuration of the Salesforce instance. This includes explicit CRUD and FLS checks before carrying out any DML. In order to save a signature image as an attachment or file associated with a parent record, the user must have **explicit** edit permission for the sobject type of the parent record via their profile or an assigned permission set.

0137 6520 578

info@brightgen.com