22 Oct 2021

SIGNATURE CAPTURE-FIELD SERVICES(1.0.1)

1. Overview

Signature Capture is a custom widget that can capture signatures and other written content and store it as images.

## Use case

### A delivery app where the user needs to sign into the app to authorize and accept the delivery.

## Features:

## The images can be stored either on the device, or on a Network File System.

## This component uses NFI and doesn’t work on Iris Preview.

## Signature canvas for capture.

## Signature saved as image (png or jpeg).

## Image can be saved on device or on network File System (NFS).

## Canvas and pen colour are customizable.

## Percentage of re-use:

Approximate 85% of reuse.

# Getting Started

## Prerequisites

Before you start using the Signature Capture-Field Services component, ensure the following:

• [HCL Foundry](https://manage.hclvoltmx.com/)

• Volt MX Iris

## Platforms Supported

### Mobile

#### iOS

#### Android

### Tablet, iPad

### Note: While building the app for Android in Debug and Release mode, in order to save signature to the device. You to need to add READ\_PHONE\_STATE and WRITE\_EXTERNAL\_STORAGE permissions in project settings.

### Graphical user interface, text, application, email Description automatically generated

## Importing the App

## You can import the Forge components only into the apps that are of the Reference Architecture type.

## **To import the Signature Capture-FS component, do the following:**

## Open your app project in Volt MX Iris.

2. In the Project Explorer, click the **Templates** tab.

## Graphical user interface, text, application Description automatically generated

3.Right-click **Components**, and then select **Import Component**. The **Import**

**Component** dialog box appears.

Graphical user interface, text, application, Teams

Description automatically generated

4.Click **Browse** to navigate to the location of the component, select the component, and then click **Import**. The component and its associated widgets and modules are added to your project.

## 

Once you have imported a component to your project, you can easily add the component to a form. For more information, refer [Add a Component to a Form](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_user_guide/Content/C_UsingComponents.html" \l "add-a-component-to-a-form)

## Building and previewing the app

After performing all the above steps, you can build your app and run it on your device. For more information, you can refer to the [Building and Viewing an Application](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_user_guide/Content/Cloud_Build_in_VoltMX_Iris.html#cloud) section of the Volt MX User Guide.

You can then run your app to see the Signature Capture-FS work in real time.

# References

## Dynamic Usage

You can also add **Signature Capture-Field Services** component dynamically. To do so,

1. In the **Project Explorer**, on the **Projects** tab, click **Controllers** section to access the respective **Form Controller**. Create a method and implement the code snippet similar to the sample code mentioned below.



/\* Creating signatureCapture component instance \*/

var signatureCapture = new com.voltmx.signaturecapture({

id: "signaturecapture",

isVisible: true,

top:"0dp",

left:"0dp",

width:"100%",

height:"60%",

clipBounds: true,

autogrowMode: voltmx.flex.AUTOGROW\_NONE,

layoutType: voltmx.flex.FLOW\_VERTICAL,

skin: "slFbox",

zIndex:1

},{},{});

/\* Setting component's properties \*/

signatureCapture.penColor = "000000";

signatureCapture.canvasBackground = "FFFFFF";

signatureCapture.saveSignature = true;

signatureCapture.saveAs = "png";

signatureCapture.saveTo = "Device";

/\*Adding the Signature Capture-FS component to a form\*/

this.view.add(signatureCapture);

In the code snippet, you can edit the properties of the component as per your requirement. For more information, see Setting Properties.

2. Save the file

## Properties

The properties provided on the **Component** tab allows you to customize the elements in the **Signature Capture-FS** component. These elements can be UI elements, service parameters, and so on. You can set the properties from the Volt MX Iris Properties panel on the right-hand side. You can also configure these properties using a JavaScript code.

General Properties

1. Pen Color Hexcode (penColor)

|  |  |
| --- | --- |
| **Description:** | Specifies the color of the pen that is used to draw on the canvas. |
| **Syntax**: | penColor |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Example:** | this.view.signatureCaptureFS.penColor= "000000"; |
| **Remarks:** | The default value for the property is “000000”. |

1. Background Color Hexcode (canvasBackground)

|  |  |
| --- | --- |
| **Description:** | Specifies the background color of the signature canvas. |
| **Syntax**: | canvasBackground |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Example:** | this.view.signatureCaptureFS.canvasBackground= "FFFFFF"; |
| **Remarks:** | The default value for the property is “FFFFFF”. |

3. Canvas Width(canvasWidth)

|  |  |
| --- | --- |
| **Description:** | Specifies the width of the signature canvas. |
| **Syntax**: | canvasWidth |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Example:** | this.view.signatureCaptureFS.canvasWidth= "94%"; |
| **Remarks:** | The default value for the property is "94%". |

4.Canvas Height(canvasHeight)

|  |  |
| --- | --- |
| **Description:** | Specifies the height of the signature canvas. |
| **Syntax**: | canvasHeight |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Example:** | this. view. signatureCaptureFS.canvasHeight= "75%"; |
| **Remarks:** | The default value for the property is "75%". |

**5. Save Signature as Image(saveSignature)**

|  |  |
| --- | --- |
| **Description:** | Specifies whether the component should save the signature. |
| **Syntax**: | saveSignature |
| **Type:** | Boolean |
| **Read/Write:** | Read + Write |
| **Example:** | this.view.signatureCaptureFS.saveSignature = true; |
| **Remarks:** | The default value for the property is true. |

**6. Save Button Visibility(isSaveVisibile)**

|  |  |
| --- | --- |
| **Description:** | Toggles the visibility of the Save Button. |
| **Syntax**: | isSaveVisibile |
| **Type:** | Boolean |
| **Read/Write:** | Read + Write |
| **Example:** | this. view.signatureCaptureFS.isSaveVisibile= true; |
| **Remarks:** | The default value for the property is true. |

**7. Clear Label Visibility (isCancelVisible)**

|  |  |
| --- | --- |
| **Description:** | Toggles the visibility of the Clear Label. |
| Syntax: | isCancelVisible |
| **Type:** | Boolean |
| **Read/Write:** | Read + Write |
| **Example:** | This.view. signatureCaptureFS.isCancelVisible= true; |
| **Remarks:** | The default value for the property is true. |

**8. Save Button Text(textSave)**

|  |  |
| --- | --- |
| **Description:** | Specifies the text that you want to display on the save button. |
| **Syntax**: | textSave |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Example:** | this. view. signatureCaptureFS.textSave= "SAVE"; |
| **Remarks:** | The default value for the property is “SAVE”. |

**9. Clear Label Text (textCancel)**

|  |  |
| --- | --- |
| **Description:** | Specifies the text that you want to display on the clear label. |
| **Syntax**: | textCancel |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Example:** | this. view. signatureCaptureFS.textCancel="Clear"; |
| **Remarks:** | The default value for the property is “Clear”. |

**Save Properties**

**10. Save To(saveTo)**

|  |  |
| --- | --- |
| **Description:** | Specifies where the component should save the image. |
| **Syntax**: | saveTo |
| **Type:** | * String * List Selector |
| **Read/Write:** | Read + Write |
| **Values:** | * Device * Network File System |
| **Example:** | this. view. signatureCaptureFS. saveTo= "Device"; |
| **Remarks:** | On Iris, you can select this value from a drop-down list. |

**11. Image Save Format(saveAs)**

|  |  |
| --- | --- |
| **Description:** | Specifies the file format of the saved image. |
| **Syntax**: | saveAs |
| **Type:** | * String * List Selector |
| **Read/Write:** | Read + Write |
| **Values:** | * Png * Jpeg |
| **Example:** | this. view. signatureCaptureFS. saveAs= "png"; |
| **Remarks:** | On Iris, you can select this value from a drop-down list. |

**Skins Section**

**12. Save Button Skin (skinSave)**

|  |  |
| --- | --- |
| **Description:** | Specifies the skin of the save button. |
| **Syntax**: | skinSave |

**13. Clear Label Skin (skinCancel)**

|  |  |
| --- | --- |
| **Description:** | Specifies the skin of the clear label. |
| **Syntax**: | skinCancel |

## C. **Events**

**1. On Save Image Success(onSaveImageSuccess)**

|  |  |
| --- | --- |
| **Description:** | Invoked when the component successfully saves image. |
| **Syntax**: | onSaveImageSuccess |
| **Parameters:** | response[string]:  Response from the platform on which the components save the image |
| **Example:** | this.view.componentID.onSaveImageSuccess = function(response)  {  alert(response);  }.bind(this); |

**2. On Save Image Failure(onSaveImageFailure)**

|  |  |
| --- | --- |
| **Description:** | Invoked when the component failed to save image. |
| **Syntax**: | onSaveImageFailure |
| **Parameters:** | response[string]:  Information about the failure of saving the image |
| **Example:** | this.view.componentID.onSaveImageFailure = function(response)  {  alert(response);  }.bind(this); |

**3. On Error Callback(onErrorCallback)**

|  |  |
| --- | --- |
| **Description:** | Invoked when an error occurs in the component. |
| **Syntax**: | onErrorCallback |
| **Parameters:** | response[string]:  Information about the exception raised by the component |
| **Example:** | this.view.componentID.onErrorCallback = function(response)  {  alert(response);  }.bind(this); |

**4. On Check Validity(onCheckValidity)**

|  |  |
| --- | --- |
| **Description:** | Invoked when save signature as image property is set as false. |
| **Syntax**: | onCheckValidity |
| **Parameters:** | bool[boolean]:  Checks whether the signature present or not. |
| **Example:** | this.view.componentID.onCheckValidity = function(bool)  {  alert(bool);  }.bind(this); |

## API’s

### getSignatureFromDevice

|  |  |
| --- | --- |
| **Description:** | Fetches the decrypted base64 of the signature from the device. |
| **Syntax**: | getSignatureFromDevice |
| **Parameters:** | None |
| **Return Value:** | signature[string]:  decrypted base64 value of the signature. |
| **Example:** | this. view. signatureCaptureFS.getSignatureFromDevice (); |

# Revision History

App version 1.0.1:

## Known Issues

## -

## Limitations - The maximum values for the width and height of the canvas are

* + Width: **338Dp** or **94%**
  + Height: **320Dp** or **75%**