08 March 2024

GooGLE LOGIN

version: 1.1.3

1. **OVERVIEW**

Add Google Login to your app in minutes-just drag and drop the component in your Volt MX Iris app, configure few properties and voila! Users can now login to your app with their Google credentials.

## **Use case:**

Google Sign-In is an authentication service that uses **OAUTH 2.0**. By using Google Sign-In, you can also integrate other Google APIs with your app. The Google Login component adds the Google Sign-In feature to your app. Under the hood, the component uses  [Identity Services](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Foundry/voltmx_foundry_user_guide/Content/Identity.html) on Volt MX Foundry.

## **Features:**

* Ready to use Google Login.
* Configure UI properties and service parameters directly from Volt MX Iris panel.

## **Percentage of re-use:**

Approximate 90% of reuse.

# **Getting Started**

## **Prerequisites**

Before you start using the Google Login component, ensure the following:

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

• Volt MX Iris

• A Google account with the following:

1. Authorized Domain of your Volt MX Foundry authentication cloud
2. OAuth Client ID
3. OAuth Client Secret

For more information, refer to the following links:

* [Obtaining the Client ID and Client Secret](#_Obtaining_the_Client)
* [Using OAuth 2.0 to Access Google APIs](https://developers.google.com/identity/protocols/oauth2)

## **Platforms Supported**

### Mobile

#### iOS

#### Android

### Tablet & iPad

### PWA & Responsive Web

## **Importing the app**

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

## **To import the Google Login component, do the following:**

## Open your app project in Volt MX Iris.

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

Graphical user interface, text, application

Description automatically generated

1. Right-click **Components**, and then select **Import Component**. The **Import Component** dialog box appears.

Graphical user interface, text, application, Teams

Description automatically generated

1. 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)

## **Obtaining the Client ID and Client Secret**

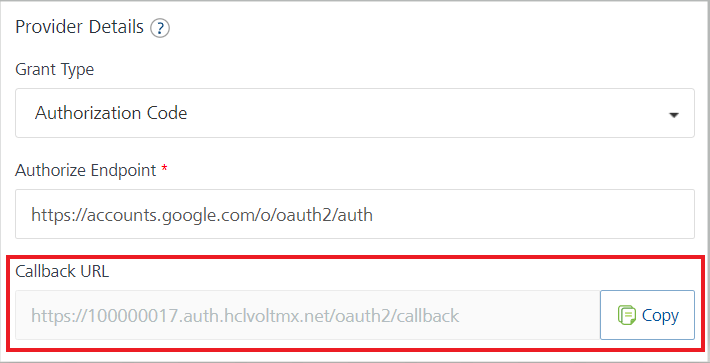
To obtain the client ID and client secret from Google, you need to perform the following steps:

1. **Authorizing the HCL Volt MX Foundry Cloud Domain**

To use the Google APIs in a domain, you need to authorize the domain on the Google Developer Console.

To authorize the HCL Volt MX Foundry Cloud domain, follow these steps:

1. Sign in to the [Google Developer Console](https://console.developers.google.com/).
2. From the left pane, select **OAuth consent screen**.
3. Under **Authorized Domains**, in the text box, type **hclvoltmx.net**, and then press Enter.
4. Click **Save**.
5. **Obtaining the Callback URL**
6. Sign in to the [HCL Foundry](https://manage.hclvoltmx.com/).
7. Open or edit the[Identity Service of the component.](#_Configuring_the_Identity)
8. Under **Provider Details**, for the **Callback URL**, click **Copy**.



1. **Adding a Redirect URL on Google:**
2. Sign in to the [Google Developer Console](https://console.developers.google.com/).
3. From the left pane, select **Credentials**.
4. On the **Credentials** page, click **Create credentials**, and then select **OAuth client ID**.
5. Under **Application** type, select **Web application**.
6. Under **Authorized redirect URIs**, in the text box, type the **Callback URL** that you [obtained from Volt MX Foundry.](#obtained_from_VolMX_Foundry)
7. Click **Create**.

After you create the credentials, the Google Developer Console displays your **Client ID** and **Client Secret**.

You need to add the Client ID and Client Secret to the [Identity Service of the component.](#Identity_service_of_the_component)

## **Service Parameters**

The Service Parameters of the component are the Client Details of the Identity Service that is linked to the component. For more information, refer to [Obtaining the Client ID and Client Secret.](#_Obtaining_the_Client)

## **Client ID**

|  |  |
| --- | --- |
| **Description:** | Specifies the Client ID that you received from the Google Developer Console. |
| **Remarks:** | * If you change the **Client ID**, make sure that you select **Publish Changes**. * You can only change this property at the design time. |

## **Client Secret**

|  |  |
| --- | --- |
| **Description:** | Specifies the Client Secret that you received from the Google Developer Console. |
| **Remarks:** | * If you change the **Client Secret**, make sure that you select **Publish Changes**.  You can only change this property at the design time. |

## 

## **Configuring Native Settings**

The Google Login component requires Deeplink Configuration for the Android and iOS platforms. Refer to the following sections for more information.

**Android**

1. From the left navigation menu, select **Project Settings**.
2. In the Project Settings window, go to **Native** → **Android**.
3. Under **Manifest Properties & Grade Build Entries**, select **Deeplink URL Scheme**.
4. Type the following names and values in the Deeplink URL Scheme table.

|  |  |
| --- | --- |
| **Name** | **Value** |
| Scheme | http (or) https |
| Host | Googlelogin |

## Graphical user interface, application Description automatically generated

## Click Finish.

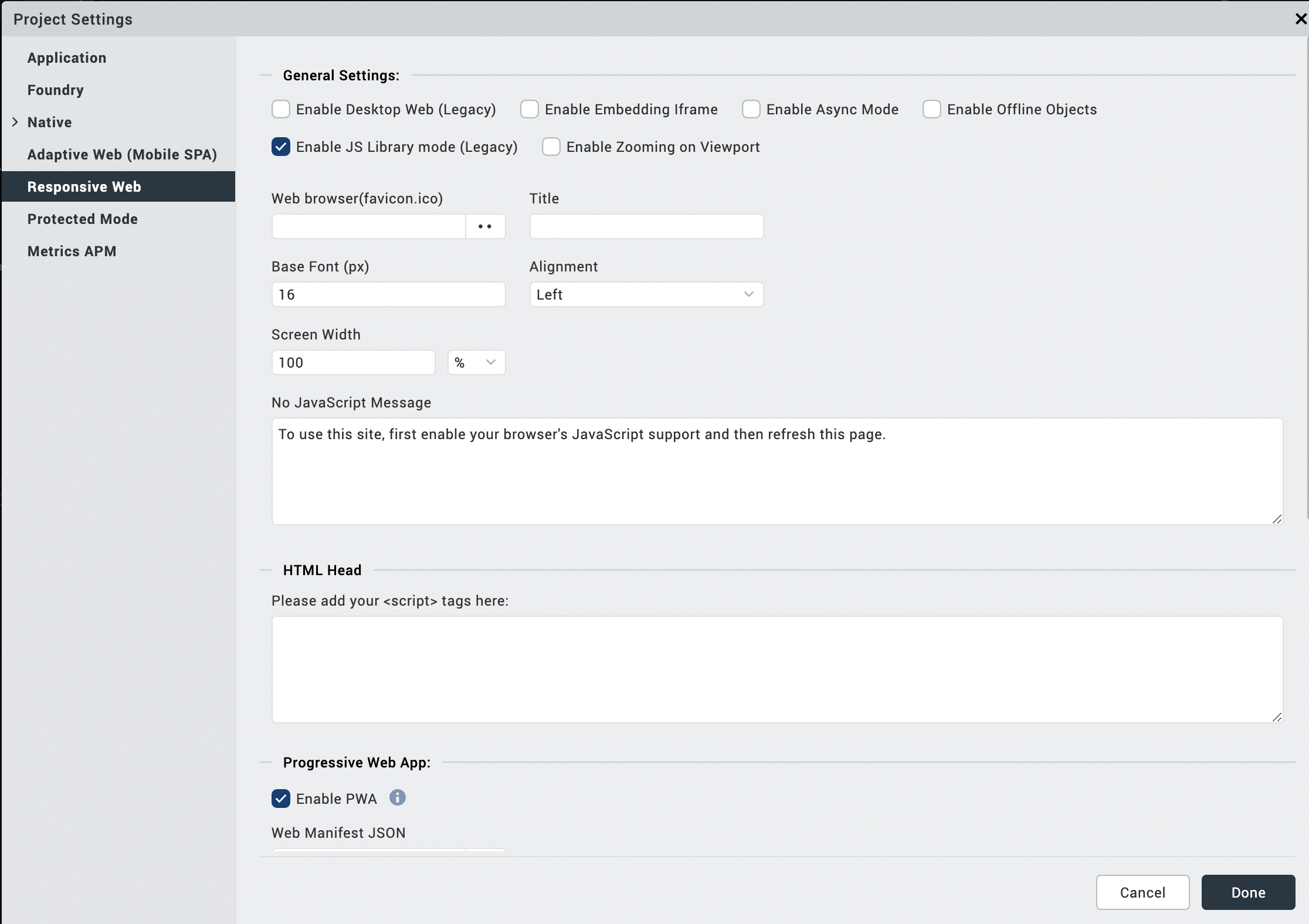
**iOS**

1. From the left navigation menu, select **Project Settings**.
2. In the Project Settings window, go to **Native** → **iPhone/iPad/Watch**.
3. In the **Deeplink URL Scheme** box, type **googlelogin**.

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

## **Responsive Web**

1. From the left navigation menu, select **Project Settings**.
2. In the Project Settings window, go to **Responsive Web** → **General Settings**.
3. Enable JS library mode (Legacy)



1. Click Finish.

## **Configuring the Identity Service on HCL Volt MX Foundry**

1. Sign in to the [HCL Foundry](https://manage.hclvoltmx.com/)
2. From the **Apps** page, select the app that contains the Google Login component services.
3. Under **Configure Services**, on the **Identity** tab, select the **GoogleLoginIdentity** Identity Service.
4. Under **Client Details**, type the [Client ID and Client Secret that you obtained from Google.](#_Obtaining_the_Client)

Graphical user interface, application, website

Description automatically generated

**Note:**  
If you want to verify the details, click **TEST LOGIN,** and sign in by using your Google credentials.

1. In the **Advanced** section, under **Redirect URL on successful authentication**, select **Allowed URL list**.

## In the URL boxes, type the [URLs that you configured in the Project Settings.](#_Configuring_Native_Settings)

1. For Responsive web, create a separate Identity service with the same client ID and Client Secret keys. To do that, repeat the above 4 steps for Responsive web.

In the **Advanced** section, under **Redirect URL on successful authentication**, select **Any URL**.



Identity Service for Responsive Web is **GoogleLoginIdentityWeb**.

## Click **SAVE.**

**Note:**  
Make sure that you select the **Redirect URL** for iOS as the **Default**.

## **Adding the Deeplink Code in the App Events**

1. From the **Project** explorer, select **Mobile** and **Tablet**.
2. From the **Properties** panel, under **App Events**, for **Deeplink**, click **Edit** for App Service. The **Action Editor** opens.
3. On the left menu of the **Action Editor**, from **Functions**, click **Add Snippet**.
4. In the code snippet, type the following code:

|  |
| --- |
| /\* handleDeeplinkCallback function checks whether the eventobject is a deeplink \*/  handleDeeplinkCallback(eventobject);  return voltmx.application.getCurrentForm(); |

## **Publishing an APP**

## After adding the **googlelogin** component to your app, you must publish the Volt MX Foundry app. For more information, refer to[Publish an app on Volt MX Foundry](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Foundry/voltmx_foundry_user_guide/Content/Publish.html)

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

## **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 Google Login work in real time.

# **References**

## **Dynamic Usage**

You can also add **Google Login** component dynamically. To do so,

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



|  |
| --- |
| var GoogleLogin = new com.voltmxmp.googlelogin(  {  "autogrowMode": voltmx.flex.AUTOGROW\_NONE,  "clipBounds": true,  "height": "50dp",  "id": "GoogleLogin",  "isVisible": true,  "left": "45%",  "top": "45%",  "width": "205dp"  }, {}, {});  /\* Setting the component's properties \*/  GoogleLogin.logoScaleMode = constants.IMAGE\_SCALE\_MODE\_MAINTAIN\_ASPECT\_RATIO;  GoogleLogin.iconSource = "voltmxmp\_gl\_btn\_google\_signin\_light.png";  GoogleLogin.redirectUrlAndroid = "http://googlelogin";  GoogleLogin.redirectUrlIos = "googlelogin://";  GoogleLogin.iconFocusSource = "voltmxmp\_gl\_btn\_google\_signin\_light\_pressed.png";  /\* Adding the component to a form \*/  this.view.add(GoogleLogin); |

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 allow you to customize the elements in the **Google Login** 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 JavaScript code.

**Note:** If you copy code from any example, make sure that you replace componentID with the name of your component.

**General Properties**

#### **logoScaleMode**

|  |  |
| --- | --- |
| **Description:** | Specifies the scale mode of the icon. |
| **Syntax:** | **logoScaleMode** |
| **Type:** | * List Selector * Constant |
| **Read/Write:** | Read + Write |
| **Values:** | * Fit to Dimensions * Maintain Aspect Ratio * Crop |
| **Default Values:** | [Maintain Aspect Ratio](javascript:void(0);) |
| **Remarks:** | For more information on scale modes, refer to[Image Scale Mode](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_user_guide/Content/Image2.html) [Volt MX Iris Widget guide](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_widget_prog_guide/Content/Image.html). |
| **Example:** | **this.view.componentID.logoScaleMode = constants.IMAGE\_SCALE\_MODE\_MAINTAIN\_ASPECT\_RATIO;** |

1. **iconSource**

|  |  |
| --- | --- |
| **Description:** | Specifies the image that you want to set as the icon. |
| **Syntax:** | **iconSource** |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Example:** | **this.view.componentID.iconSource = "voltmxmp\_gl\_btn\_google\_signin\_light.png";** |

1. **iconFocusSource**

|  |  |
| --- | --- |
| **Description:** | Specifies the image that you want to set as the icon when it is in focus. |
| **Syntax:** | **iconFocusSource** |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Example:** | **this.view.componentID.iconFocusSource = "voltmxmp\_gl\_btn\_google\_signin\_light\_pressed.png";** |

1. **redirectUrlAndroid**

|  |  |
| --- | --- |
| **Description:** | Specifies the Redirect URL for a successful sign-in from the Android platform. |
| **Syntax:** | **redirectUrlAndroid** |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Make sure that you type the value as http://hostname or https://hostname. You can replace **hostname** with the name of the host that you want to use. * Make sure that the Redirect URL in the property is the same as the Redirect URL on HCL Volt MX Foundry. Otherwise, the navigation from the browser to the app fails. * This property can only be changed at the design time. |
| **Example:** | **this.view.componentID.redirectUrlAndroid = "http://googlelogin";** |

1. **redirectUrlIos**

|  |  |
| --- | --- |
| **Description:** | Specifies the Redirect URL for a successful sign-in from the iOS platform. |
| **Syntax:** | **redirectUrlIos** |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Make sure that the Redirect URL in the property is the same as the Redirect URL on HCL Volt MX Foundry. Otherwise, the navigation from the browser to the app fails. * This property can only be changed at the design time. |
| **Example:** | **this.view.componentID.redirectUrlIos = "googlelogin://";** |

## **Events**

#### **onLoginSuccess**

|  |  |
| --- | --- |
| **Description:** | Invoked when the user sign-in to Google is successful. |
| **Syntax:** | onLoginSuccess(response) |
| **Parameters:** | *response [JSON]*: The response from the Identity Service on HCL Volt MX Foundry. |
| **Example:** | this.view.componentID.onLoginSuccess = function(response)  {  alert("Login Successful. Response: " + JSON.stringify(response));  }.bind(this); |

#### **onLoginFailure**

|  |  |
| --- | --- |
| **Description:** | Invoked when the user sign-in to Google fails. |
| **Syntax:** | onLoginFailure(response) |
| **Parameters:** | *response [JSON]*: The response from the Identity Service on HCL Volt MX Foundry. |
| **Example:** | this.view.componentID.onLoginFailure = function(response)  {  alert("Login Failed. Response: " + JSON.stringify(response));  }.bind(this); |

## **API’s**

### No API’s used.

# **Revision History**

App version 1.1.3:

## **Known Issues**

## No Known Issues

## **Limitations**

The component does not support landscape mode.