Date : 09-OCT-24

OFFICE365 LOGIN

VERSION: 1.0.2

1. **OVERVIEW**

Office365 login is a fast and convenient way for users to create an account and login to your app. Drag and drop the component in your app, configure a few properties and you are ready to go.

The following sections help you use the Office365 Login component in your app:

## **Use case:**

Add Office365 login to your app in minutes- just drag and drop the component in your Iris app, configure a few properties and voila! users can now login to your app with their Office365 credentials.

## **Features**

### Provides ​Ready to use Office365 login

### Configure UI properties and service parameters directly from Iris properties panel.

## **Percentage of re-use**

* 70-80%

# **Getting Started**

## **Prerequisites**

Before you start using the Privacy Preferences component, ensure you have the following:

* [HCL Foundry](https://manage.hclvoltmx.com/)
* Volt MX Iris

## **Platforms Supported**

### Mobile

#### iOS

#### Android

### PWA

## **Importing the Component**

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

**To import the Office365 Login component, do the following:**

1. Open your project in Volt MX Iris.
2. In the Project Explorer, click the **Templates** tab.

Graphical user interface, text, application

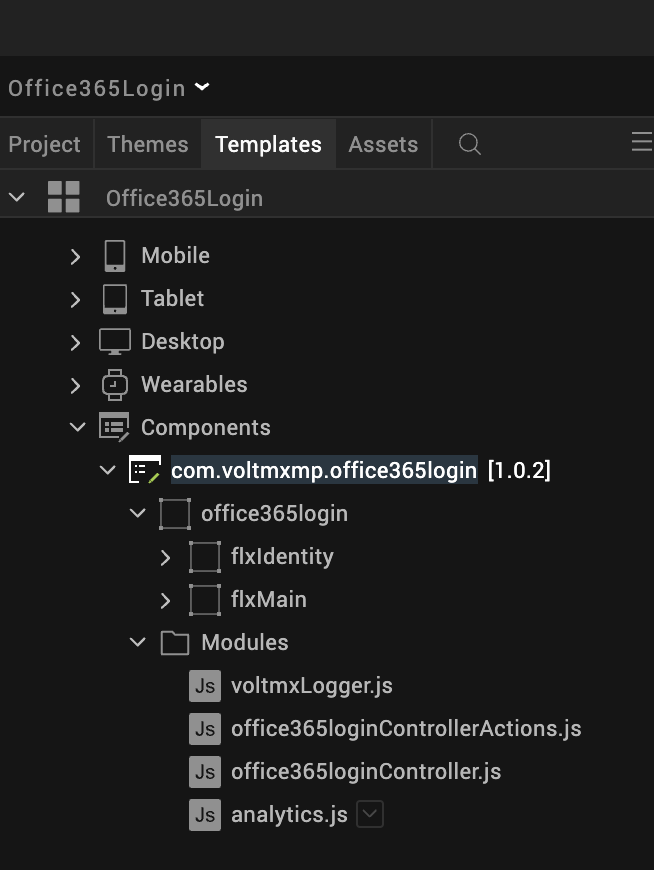
Description automatically generated

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#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 Iris User Guide.

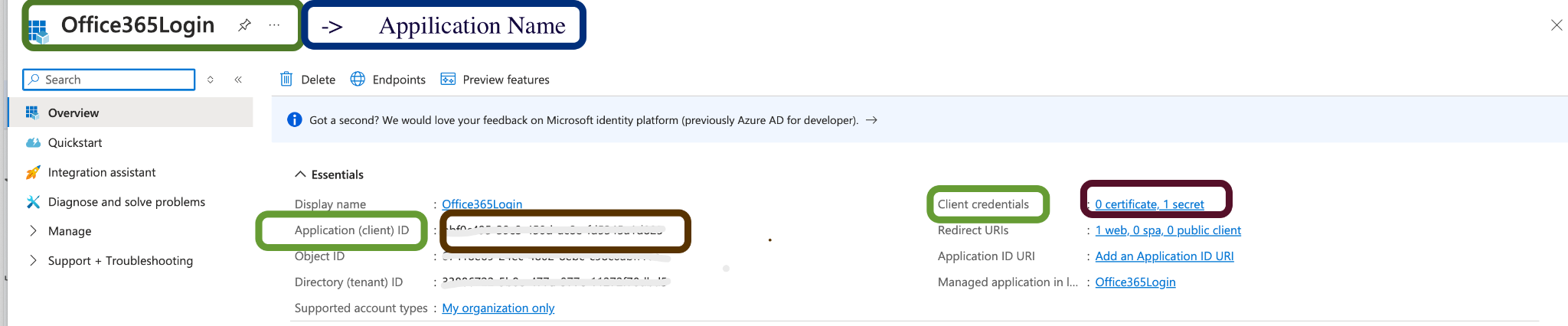
**E.** [**Creating an app in Office365 Developer Conso**](#_heading=h.e0hq5wn02kg0)**le**

i.Go to [Azure portal](https://portal.azure.com/#home)

ii.Search for “AppRegistrations” in the search bar.

iii.Click on “New registration”,Enter the required details and click on “Register”.

iv. You get the **Client Id** and **Client Secret .**



# **REFERENCES**

**A. Dynamic Usage**

Create a function call ,If you want to use the Privacy Preferences component dynamically, you will need to import the component into your project Templates. Follow the given steps to do so

* 1. Download the component from HCL VoltMX Marketplace as a zip file.
  2. Go to the Templates tab in your project explorer.
  3. Right click on Components and select Import Component.
  4. Navigate to where you downloaded your zip file and import it into Iris.

After you import the component into your project templates, you can add it to your app dynamically. To do so, follow the given steps.

You can also add an office365 login Component dynamically. To do so,

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

/\* creating a component's Object \*/

createComponent: function(){

var office365login = new com.voltmxmp.office365login(

{

"autogrowMode": voltmx.flex.AUTOGROW\_NONE,

"clipBounds": true,

"height": "50dp",

"id": "office365login",

"isVisible": true,

"layoutType": voltmx.flex.FREE\_FORM,

"left": "45%",

"masterType": constants.MASTER\_TYPE\_USERWIDGET,

"skin": "slFbox",

"top": "45%",

"width": "50dp"

}, {}, {});

/\* Setting the component's properties \*/

office365login.iconSource = "reusablelogin\_office365.png";

office365login.logoScaleMode= constants.IMAGE\_SCALE\_MODE\_FIT\_TO\_DIMENSIONS;

/\* Adding the component to a Form \*/

this.view.add(office365login);

},

});

1. **Save** the file.

| **Note:**  VoltMX Foundry services(Created app) must be published only after the app is imported into the form. The app must be published by an existing user only facilitating the authentication of credentials. |
| --- |

**B. Setting properties**

You can use a component’s Properties to customize and configure the elements. These elements can be UI elements, service parameters, and so on. For more information about properties, you can refer to the Components Overview section of the Iris User Guide.

You can set the properties from the Iris’s Properties panel on the right-hand side. You can also configure these properties using a JavaScript code.

#### **General**

1. **Icon Source**

| **Description:** | This property enables the consumer to change the icon source in the component. |
| --- | --- |
| **Syntax:** | iconSource |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | The default value for this property is "icon.png". |
| **Example:** | this.view.componentID.iconSource = "icon.png"; |

**Open2. Icon Scale Mode**

| **Description:** | This property enables the consumer to set the scale mode of the icon source. |
| --- | --- |
| **Syntax:** | logoScaleMode |
| **Type:** | Constant |
| **Read/Write:** | Read + Write |
| **Remarks:** | [Click here](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_api_dev_guide/content/constants_namespace.html#the-following-constants-set-the-skin-image-scaling-mode) for more information about the scaling properties of the image widget. |
| **Example:** | this.view.componentID.logoScaleMode = constants.IMAGE\_SCALE\_MODE\_FIT\_TO\_DIMENSIONS; |

### **Service Parameters**

Properties under the service parameters can be seen only as component perspective but not at project level. Copy the client ID and client secret and paste under the below properties. For more information on obtaining the Client ID and Client Secret, refer to the [Creating an app in Office365 Developer Console](#_heading=h.e0hq5wn02kg0) section.

**Open1. Client Id**

| **Description:** | This property enables the consumer to set the client id of the respective identity service in VoltMX Foundry from Iris. |
| --- | --- |
| **Remarks:** | If you change the client id, ensure to click on import changes.  This property can’t be changed at the run time.(Only at design time). |

**Open2. Client Secret**

| **Description:** | This property enables the consumer to set the client secret of the respective identity service in VoltMX Foundry from Iris.. |
| --- | --- |
| **Remarks:** | If you change the client secret, ensure to click on import changes.  This property can t be changed at the run time.(Only at design time). |

**Open3. Resource Value**

| **Description:** | This property enables the consumer to which resource in the Office 365 or API that this access token applies to. |
| --- | --- |
| **Example:** | https://graph.microsoft.com |

**C. Events**

You can define events to be executed when an action is performed. You can configure the events directly on the Actions tab or by writing a JavaScript. To configure the events on the Actions tab, click the Edit button against each event. For more information, refer to [Add Actions](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_user_guide/Content/ActionsNavigation_1.html).

This section provides details about each event that help you define the actions by writing a JavaScript.

**Open1. onLoginSuccess**

| **Description:** | An event callback is invoked when Office365 oauth login is a success. |
| --- | --- |
| **Syntax:** | onLoginSuccess(response) |
| **Example:** | this.view.componentID.onLoginSuccess = function(response)  {  alert("Login Successful");  }.bind(this); |

**Open2. onLoginFailure**

| **Description:** | An event callback is invoked when Office365 oauth login fails. |
| --- | --- |
| **Syntax:** | onLoginFailure(response) |
| **Example:** | this.view.componentID.onLoginFailure = function(response)  {  alert("Login Failed");  }.bind(this); |

# **Revision History**

App version 1.0.2

## **Known Issues**

NA

## **Limitations**

NA