Date : 9-Oct-21

FILE UPLOAD

VERSION: 1.0.1

1. **OVERVIEW**

The File Upload (also known as Browse File) component helps a user select files from their device. The user can either drag-and-drop files onto the component, or select files using a file explorer.

You can customize the text that you want to display to the user. You can also define your own logic to execute when the user drags and drops a file onto the component.

This document consists of the following sections, you can visit any one of them to understand the usage of the File upload component.

## Percentage of re-use:

70-80% (Browse text and button text can be customizable and skins are not customized but can be changed manually)

#  GETTING STARTED

##  Prerequisites

Before you start using the File Upload component, ensure you have the following:

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

##  Platforms Supported

### PWA

##  Importing the Component

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

## To import the File Upload component, do the following:

## Open your app project in Volt MX Iris.

## In the Project Explorer, click the Templates tab.



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



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).

1. **REFERENCES**

**A. Dynamic Usage**

If you want to use the File Upload component dynamically, you will need to import the component into your project **Templates**. Follow the given steps 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.

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

createComponent: function()

{

/\* Creating the component's object \*/

var BrowseFile = new com.voltmxmp.browsefile(

{

 "clipBounds": true,

 "height": "100%",

 "id": " BrowseFile",

 "isVisible": true,

 "centerx": "0dp",

 "centery": "0dp",

 "width": "100%",

 "zIndex": 1

}, {}, {});

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

BrowseFile.browseText = "<browseText>";

/\* Defining the component's events \*/

BrowseFile.onFileDragAndDrop = function(files)

{

 alert("Files added.");

};

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

this.view.add(BrowseFile);

}

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

1. **Save** the file.

# 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. You can set the properties directly on the **Component** tab or by writing a JavaScript.

**1. browseText**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies the text that you want to display as the browse message. |
| **Syntax:** | browseText |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Example:** | this.view.componentID.browseText = "Drag and Drop Files to add"; |

**2. buttonText**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies the text that you want to display as button text. |
| **Syntax:** | buttonText |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Example:** | this.view.componentID.buttonText = "Select some files"; |

# Events

The component invokes events when its corresponding action is performed. You can configure any logic you want the component to perform whenever an event occurs. You can configure the events directly on the Actions tab or by writing a JavaScript.

**1. onFileDragAndDrop**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Invoked when the user drags and drops a file onto the component. |
| **Syntax:** | onFileDragAndDrop |
| **Parameters:** | *files [File Object]* :Contains the files that are dragged and dropped on the component. |
| **Example:** | this.view.componentID.onFileDragAndDrop = function(files){ alert("Files added.");}.bind(this); |

# APIs

## None of the APIs are exposed.

1. **REVISION HISTORY**

App version 1.0.1

## E. Limitations

* The component supports a maximum file size of **100MB**.
* The initial version of the component only provides the file object after the user drags-and-drops the files. To upload the files to a server, you can use the [VoltMX Iris SDK](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Foundry/voltmx_foundry_user_guide/Content/VoltMXStudio/Binary_APIs.html).

## F. Known issues

 No known issues.