Date: 05-Apr-2023

LIST VIEW -DOCS

 Version: 1.0.2

1. **OVERVIEW**

Plug and play component to list documents in your app. Usually it is consumed in conjugation with a document rendering component like PDF viewer. The supported platforms (currently) are Android (Mobile) and iOS (Mobile) and Responsive Web.

1. **Use Case**:

Consider a case where you want to implement a feature in your app for listing a set of document files. Using the List view – docs component, you can achieve this feature in your app. The List view – docs component is a ready-to-use component. You can import the List view – docs component into your app and can achieve the features without developing it from scratch.

1. **Features**
* Fully customizable- in addition to document name and path, provide sub-text for additional information (author, creation/modification date etc.) and display image to distinguish different document types.
* Configure properties directly from Iris in case your document-list is static, otherwise use APIs to display the dynamic content.

 **C.** **Percentage of re-use:**

       90%

 **2. GETTING STARTED**

1. **Prerequisites**

             Before you start using the List view – docs component, ensure the following:

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

1. **Platforms Supported**
2. Mobile
3. *iOS*
4. *Android*
5. **Importing the Component**

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

**To import the List View Docs component, do the following:**

1. Open your app project in Volt MX Iris.

2. 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.



**3. REFERENCES**

**A. Dynamic Usage**

Create a function called createComponent() and write the code inside it to create and con figure the component .You can refer to the given sample code for more information

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 a component's Object \*/

var doclist = new com.voltmxmp.doclist(

{

 "clipBounds": true,

 "top" : "0%",

 "height": "100%",

 "id": "doclist",

 "isVisible": true,

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

 "left": "0dp",

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

 "skin": "slFbox",

 "width": "100%"

}, {}, {});

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

doclist.data =

[{

    "lblDocumentName": "Sample PDF1",

    "lblDocumentSubText" : "Sample Description 1",

    "lblDocumentPath" : "http://alert.com",

    "imgDocumentType" :"https://th.bing.com/th/id/OIP.outS4iVKtTsq8NLsZjE2-AAAAA?pid=ImgDet&rs=1"

},

{

    "lblDocumentName": "Sample PDF2",

    "lblDocumentSubText" : "Sample Description 2",

    "lblDocumentPath" : "http://sample.com",

    "imgDocumentType" :"https://th.bing.com/th/id/OIP.outS4iVKtTsq8NLsZjE2-AAAAA?pid=ImgDet&rs=1"

}];

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

this.view.add(doclist);

Now, you need to call this function using **Actions**. For more information, refer to the [Add Actions](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_user_guide/Content/working_with_Action_Editor.html) section of the Iris User Guide

1. **Save** the file.

**B. Setting Properties**

The properties provided on the **Component** tab allow you to customize the UI elements in the **list view docs** component. You can set the properties directly on the **Component** tab or by writing a JavaScript.

**A. General Properties**

1. **Documents List**

|  |  |
| --- | --- |
| **Description:** | Specifies the list of all the documents to be added. |
| Syntax: | data |
| **Type:** | Array of JSON Objects |
| **Read/Write:** | Read + Write |
| **Example:** | this.view.componentID.data =[{    "lblDocumentName": "Sample PDF1",    "lblDocumentSubText" : "Sample Description 1",    "lblDocumentPath" : "http://alert.com",    "imgDocumentType":"https://th.bing.com/th/id/OIP.outS4iVKtTsq8NLsZjE2-AAAAA?pid=ImgDet&rs=1"},{    "lblDocumentName": "Sample PDF2",    "lblDocumentSubText" : "Sample Description 2",    "lblDocumentPath" : "http://sample.com",    "imgDocumentType":"https://th.bing.com/th/id/OIP.outS4iVKtTsq8NLsZjE2-AAAAA?pid=ImgDet&rs=1"}];. |

**C. Events**

* **onDocumentClick**

|  |  |
| --- | --- |
| **Description:** | The event is invoked when user clicks on any row of the doclist. |
| **Syntax:** | onDocumentClick() |
| **Example:** | this.view.componentID.onDocumentClick = function(){ alert( “Document is clicked” );}; |

**D. APIs**

The following API pertains to the **doclist** component.

* **setDocumentData**

This API sets the data to the doclist.

**Syntax**

setDocumentData(data)

**Parameters**

Array of JSON Objects

**Return Value**

None

**Example**

var data=

[{

 "lblDocumentName": "Sample PDF1",

 "lblDocumentSubText" : "Sample Description 1",

 "lblDocumentPath" : "http://alert.com",

 "imgDocumentType" : "https://th.bing.com/th/id/OIP.outS4iVKtTsq8NLsZjE2-AAAAA?pid=ImgDet&rs=1"

},

{

 "lblDocumentName": "Sample PDF2",

 "lblDocumentSubText" : "Sample Description 2",

 "lblDocumentPath" : "http://sample.com",

 "imgDocumentType" : "https://th.bing.com/th/id/OIP.outS4iVKtTsq8NLsZjE2-AAAAA?pid=ImgDet&rs=1"

}];

this.view.componentID.setDocumentData(data);

**4. REVISION HISTORY**

 App version 1.0.2

1. **Limitations**

To change the height of the component, you first need to change the height of the **row Template** and then the height of the component. Otherwise, the UI of the component appears distorted.