07 Feb 2024

Tabs (Text with focus-button and shell)

version: 1.0.3

1. **OVERVIEW**

Tabs are used to organize content at app level. Using these, users can quickly switch between different sections of the app.

## **Use case:**

Consider a case that you want to develop an app for social networking that helps users to perform multiple transactions in separate sections. In this app, you want to develop a feature that helps users to make calls via internet, send messages, and view status. Using the Tabs component, you can achieve the feature in your app.

## **Features:**

* Ready-to-use and easy to implement Tab patterns
* Useful component for Navigation between different sections (functional modules)
* Easy to plug into your app
* Facilitates easy customization of UI

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

Approximate 80% of reuse.

# **Getting Started**

## **Prerequisites**

Before you start using the Tabs (Text with focus-Button and Shell) component, ensure the following:

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

• Volt MX Iris

## **Platforms Supported**

1. Mobile
2. iOS
3. Android
4. Tablet & iPad

## **Importing the app**

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

## **To import the Tabs (Text with focus-Button and Shell)** **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 Tabs (Text with focus-Button and Shell) work in real time.

# **References**

## **Dynamic Usage**

You can also add **Tabs (Text with focus-Button and Shell)** 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 the component's Object \*/  var TabsTextButtonShell = new com.voltmxmp.tabstextbuttonshell(  {  "clipBounds":true,  "id": "TabsTextButtonShell",  "isVisible": true,  "left": "0%",  "width": "100%",  "top": "0%",  "zIndex": 1  }, {}, {});  /\* Setting the component's properties \*/  TabsTextButtonShell.tabCount = 6;  TabsTextButtonShell.tabWidth = "33.3%";  TabsTextButtonShell.title0 = "News";  TabsTextButtonShell.title1 = "Business";  TabsTextButtonShell.title2 = "Technology";  TabsTextButtonShell.title3 = "Social";  TabsTextButtonShell.title4 = "Finance";  TabsTextButtonShell.title5 = "Product";  TabsTextButtonShell.sknBackground = "voltmxmpsknFlxInner";  TabsTextButtonShell.sknTextActive = "voltmxmpTabsTitletbsSkin";  TabsTextButtonShell.sknTextInactive = "voltmxmpttbsTabsTitleDullSkin";  TabsTextButtonShell.sknHighlight = "voltmxmpTabsIndicatorskin";  /\* Adding the component to the Form \*/  this.view.add(TabsTextButtonShell); |

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 **Tabs (Text with focus-Button and Shell)** 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.

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

**General Properties**

#### **No of Tabs**

|  |  |
| --- | --- |
| **Description:** | Specifies the number of tabs that you want to create at the run time. |
| **Syntax:** | tabCount |
| **Type:** | Integer |
| **Read/Write:** | Write |
| **Default Value:** | 3 |
| **Range of Values:** | * Minimum Value: 1 * Maximum Value: 6 |
| **Remarks:** | * Make sure that you specify a value that is within the range of values and is valid. Otherwise, the component throws an INVALID\_TABCOUNT exception. * You need to provide titles for all the tabs. Otherwise, the component throws an INVALID\_TAB\_TITLE exception. |
| **Example:** | this.view.componentID.tabCount = 3; |

#### **Tab Width**

|  |  |
| --- | --- |
| **Description:** | Specifies the width of the tab in percentage. |
| **Syntax:** | tabWidth |
| **Type:** | String |
| **Read/Write:** | Write |
| **Default Value:** | 33.3% |
| **Remarks:** | Make sure that you append the value with **%**. Otherwise, the component throws an INVALID\_TABWIDTH exception. |
| **Example:** | this.view.componentID.tabWidth = "33.3%"; |

#### **Tab 1**

|  |  |
| --- | --- |
| **Description:** | Specifies the title of the first tab (index: 0). |
| **Syntax:** | title0 |
| **Type:** | String |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.title0 = "Tab1"; |

#### **Tab 2**

|  |  |
| --- | --- |
| **Description:** | Specifies the title of the second tab (index: 1). |
| **Syntax:** | title1 |
| **Type:** | String |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.title1 = "Tab2"; |

#### **Tab 3**

|  |  |
| --- | --- |
| **Description:** | Specifies the title of the third tab (index: 2). |
| **Syntax:** | title2 |
| **Type:** | String |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.title2 = "Tab3"; |

#### **Tab 4**

|  |  |
| --- | --- |
| **Description:** | Specifies the title of the fourth tab (index: 3). |
| **Syntax:** | title3 |
| **Type:** | String |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.title3 = "Tab4"; |

#### **Tab 5**

|  |  |
| --- | --- |
| **Description:** | Specifies the title of the fifth tab (index: 4). |
| **Syntax:** | title4 |
| **Type:** | String |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.title4 = "Tab5"; |

#### **Tab 6**

|  |  |
| --- | --- |
| **Description:** | Specifies the title of the sixth tab (index: 5). |
| **Syntax:** | title5 |
| **Type:** | String |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.title5 = "Tab6"; |

## **Skins:**

#### **Tab Background**

|  |  |
| --- | --- |
| **Description:** | This skin links to the background of the tabs. |
| **Syntax:** | sknBackground |
| **Widget Type:** | FlexContainer |
| **Type:** | String |
| **Read/Write:** | Read+Write |
| **Example:** | this.view.componentID.sknBackground = "voltmxmpsknFlxInner"; |

#### **Tab Text Active**

|  |  |
| --- | --- |
| **Description:** | This skin links to the title of the tab when it is selected (active). |
| **Syntax:** | sknTextActive |
| **Widget Type:** | Label |
| **Type:** | String |
| **Read/Write:** | Read+Write |
| **Example:** | this.view.componentID.sknTextActive = "voltmxmpTabsTitletbsSkin"; |

#### **Tab Text Inactive**

|  |  |
| --- | --- |
| **Description:** | This skin links to the title of the tab when it is not selected (inactive). |
| **Syntax:** | sknTextInactive |
| **Widget Type:** | Label |
| **Type:** | String |
| **Read/Write:** | Read+Write |
| **Example:** | this.view.componentID.sknTextInactive = "voltmxmpttbsTabsTitleDullSkin"; |

#### **Tab Highlight**

|  |  |
| --- | --- |
| **Description:** | This skin links to the button-shaped highlight of the selected tab. |
| **Syntax:** | sknHighlight |
| **Widget Type:** | FlexContainer |
| **Type:** | String |
| **Read/Write:** | Read+Write |
| **Example:** | this.view.componentID.sknHighlight= "voltmxmpTabsIndicatorskin"; |

## **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, For more information, refer to [Add Actions](https://docs.kony.com/konylibrary/visualizer/visualizer_user_guide/Content/working_with_Action_Editor.htm) in the Iris User Guide.

**onTabChange**

|  |  |
| --- | --- |
| **Description:** | Invoked when the user clicks or taps a tab. |
| **Syntax:** | onTabChange |
| **Type:** | String |
| **Read/Write:** | Read+Write |
| **Parameters:** | *eventObject [JSON]*: Information about the selected tab.  Contains the following keys:   * *tabIndex [Integer]* : The index of the selected tab * *text [String]* : The title of the selected tab |
| **Example:** | this.view.componentID.onTabChange = function(eventObject)  {  alert("Tab Selected: " + JSON.stringify(eventObject));  } |
| **Sample JSON:** | {  "text" : "Tab1",  "tabIndex": 0  }; |

## **APIs**

The following APIs pertain to the Tabs (Text with focus-Button and Shell) component.

**goToTab**

|  |  |
| --- | --- |
| **Description:** | Switches to the specified tab. |
| **Syntax:** | goToTab(tabIndex) |
| **Parameters:** | *tabIndex [Integer]*: The index of the tab that you want to switch to. |
| **Return Value:** | None |
| **Example:** | var tabIndex = 2;  this.view.componentID.goToTab(tabIndex); |

**getSelectedTabIndex**

|  |  |
| --- | --- |
| **Description:** | Fetches the index of the tab that is currently selected. |
| **Syntax:** | getSelectedTabIndex() |
| **Parameters:** | None |
| **Return Value:** | *tabIndex [Number]*: The index of the tab that is currently selected. |
| **Example:** | var tabIndex = this.view.componentID.getSelectedTabIndex();  alert("Current tab: " + tabIndex); |

**3. moveLeft**

|  |  |
| --- | --- |
| **Description:** | Switches to the tab that is on the left of the tab that is currently selected. |
| **Syntax:** | moveLeft() |
| **Parameters:** | None |
| **Return Value:** | None |
| **Example:** | this.view.componentID.moveLeft(); |

**4. moveRight**

|  |  |
| --- | --- |
| **Description:** | Switches to the tab that is on the right of the tab that is currently selected. |
| **Syntax:** | moveRight() |
| **Parameters:** | None |
| **Return Value:** | None |
| **Example:** | this.view.componentID.moveRight(); |

**5. getSelectedTabDetails**

|  |  |
| --- | --- |
| **Description:** | Fetches the details of the tab that is currently selected. |
| **Syntax:** | getSelectedTabDetails() |
| **Parameters:** | None |
| **Return Value:** | *selectedTabDetails [JSON]*: The details of the tab that is currently selected.  Contains the following keys:   * *tabIndex [Integer]*: The index of the selected tab * *text [String]*: The title of the selected tab |
| **Example:** | var selectedTabDetails = this.view.componentID.getSelectedTabDetails();  alert("Current tab details: " + JSON.stringify(selectedTabDetails)); |
| **Sample JSON:** | {  "text" : "Tab1",  "tabIndex": 0  }; |

**6. createTabs**

|  |  |
| --- | --- |
| **Description:** | Creates tabs based on the specified data. |
| **Syntax:** | createTabs(tabData) |
| **Parameters:** | *tabData [Array of Strings]*:  The titles of the tabs that you want to create. |
| **Return Value:** | None |
| **Remarks:** | * Make sure that you add the Tabs component to a form before you use the createTabs API. * If the tabData parameter is empty, the API creates tabs by using the values of the component properties. |
| **Example:** | var tabData = ["Tab1", "Tab2", "Tab3", "Tab4", "Tab5", "Tab6"];  this.view.componentID.createTabs(tabData); |

# **Revision History**

App version 1.0.3:

## **Known Issues**

## No Known Issues.

## **Limitations**

* The component only supports up to six tabs.