Date : 28-Oct-21

secret FIELD

version: 2.0.1

**1. OVERVIEW**

Secret Field is a Volt Mx Iris component which is an input field that allows the user to toggle the value's mask on and off.

## **A. Use Case:**

This component is ideal for use cases where the user is expected to type in information that should be kept from other people’s eyes, but which can optionally be displayed at the user’s discretion, such as passwords, secret PIN's, and 2FA/MFA verification codes from coordinate cards, tokenizer devices or received via SMS.

## **B. Features:**

### Ready to use and customizable UI.

### Ready to use API.

### Input Field animations can be customized.

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

## 80%

**2. GETTING STARTED**

## **Prerequisites**

## Before you start using the Secret Field component, ensure you have the following:

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

## **Platforms Supported**

### Mobile

### iOS

### Android

### Tablets

### PWA

## **Importing the Component**

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

## **To import the Secret Field component, do the following:**

## Open your app project in Volt MX Iris.

## 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.

A picture containing text

Description automatically generated

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.

You can then run your app to see the Secret-Field work in real time.

**ii. Save** the file.

# **3. References**

## **A. Dynamic Usage**

You can also add a **Secret Field** 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 like 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.

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

createComponent : function(){

var secretFieldInstanse= new com.voltmx.SecretField(

{//look

"clipBounds": true,

"height": "100%",

"id": "secretFieldId",

"isVisible": true,

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

"left": "0dp",

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

"top": "0dp",

"width": "100%"

}, {}, {});

this.view.add(secretFieldInstanse);

/\* Setting component's properties \*/

//password - button skin

this.view.secretFieldId.placeholder = "Password";

this.view.secretFieldId.buttonSkin = "SecretFieldButtonSkin";

this.view.secretFieldId.buttonFocusSkin = "SecretFieldButtonFocusSkin";

this.view.secretFieldId.textSkin = "SecretFieldTextBoxSkin";

this.view.secretFieldId.iconSkin = "SecretFieldIconLabelSkin";

this.view.secretFieldId.flexSkin = "SecretFieldTopFlexSkin";

this.view.secretFieldId.icon = "\uF023";

this.view.secretFieldId.buttonIcon ="\uF06E";

this.view.secretFieldId.hideIcon = "\uF070";

//pin button skin

this.view.secretFieldId.skin1 = "SecretFieldButtonSkin";

//code button skin

this.view.secretFieldId.skin2 = "SecretFieldButtonSkin";

/\*Adding the component to the form\*/

}

});

## 2. **SAVE** the file

## **B. Properties**

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

### **TextBox (textAlign)**

### Specifies the alignment of the text.

### **Hide Button (btnpadding)**

### Specifies the space around an element’s content.

### **Button**

### ***Normal Skin***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | skinShowPin |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.skinShowPin = "SecretFieldButtonSkin"; |

### ***Normal Skin***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | skinShowKey |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.skinShowPin = "SecretFieldButtonSkin"; |

### ***Icon Skin***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | iconSkin |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.iconSkin = "SecretFieldIconLabelSkin"; |

### ***TextBox Skin***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | textSkin |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.textSkin = "SecretFieldTextBoxSkin"; |

### ***Button Skin***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | buttonSkin |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.buttonSkin = "SecretFieldButtonSkin"; |

### ***Container Skin***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | flexSkin |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.flexSkin = "SecretFieldTopFlexSkin"; |

### ***Button Focus Skin***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | buttonFocusSkin |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.buttonFocusSkin = "SecretFieldButtonFocusSkin"; |

### ***Decorating icon***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | icon |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.icon = "\uF023"; |

### ***Show icon***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | buttonicon |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.buttonIcon ="\uF06E"; |

### ***Hide icon***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | hideicon |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.hideIcon ="\uF070"; |

### ***Placeholder***

|  |  |
| --- | --- |
| **Description:** | Specifies the skin to be applied to the button. |
| **Syntax:** | placeholder |
| **Type:** | String |
| **Read/Write:** | Read + Write |
| **Remarks:** | * Before you set the property, ensure that the skin ID that you specify already exists in your app project. * The skin ID assigned to the property must be the skin of the Label. |
| **Example:** | this.view.secretFieldId.placeholder = "Password"; |

**Content Alignment** (textAlign)Specifies the alignment of the text.

## **C. Events**

-- None of the events are exposed.

## **D. APIs**

This API toggles the button in Secret Text Field. You need to click on the button to mask or unmask the entered data.

|  |  |
| --- | --- |
| **Example:** | this.view.*buttonID*.onTouchEnd = this.toggle; |

**Note:** The *buttonID* is the ID name of the specifc button. Eg: showButton, btnShowPin, btnShowKey.

**4. REVISION HISTORY**

App version 2.0.1

**A. Limitations**

* The initial version of the component does not support landscape mode.
* If browser height is minimized, then UI gets distorted.