Date

Crypto (1.0.3)

Overview

This code module wraps over the Volt MX Cryptography APIs and allows to easily utilise them wherever necessary. This component can be used to encrypt or decrypt any sensitive data, e.g. username, email-id, etc. Please refer https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris\_api\_dev\_guide/content/cryptography.html for more info on the Volt MX Cryptography APIs.

## Use case

1. To transmit sensitive or critical information across a network

## Features

1. This component can be used to encrypt or decrypt any sensitive data, e.g. username, email-id, etc.

## Percentage of re-use:

Approximate % of reuse. It sets an expectation of how much can be used out of the box, and how much needs to be customized for a specific app.

# Getting Started

## Prerequisites

Before you start using the Crypto 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 Crypto Component, do the following:**

## Open your app project in Volt MX Iris.

1. 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.

Graphical user interface, text, application

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" \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 Iris User Guide.

You can then run your app to see the Crypto work in real time.

# References

## Dynamic Usage

If you want to use the Crypto 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 [Temenos Forge](https://marketplace.kony.com/) 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 Volt MX 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.

1. Access the **Form Controller** of the form you want to add the component into.
2. Create a function called createComponent() and write the code inside it to create and configure the component.  
   You can refer to the given sample code for more information.

createComponent: function()

{

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

var crypto = new com.voltmx.cryptoComponent(

{

"clipBounds": true,

"height": "100%",

"id": "crypto",

"isVisible": false,

"left": "0dp",

"top": "0dp",

"width": "100%",

"zIndex": 1

}, {}, {});

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

crypto.encryptionMode = "ecb";

crypto.initializationVector = "1234567887654321";

crypto.paddingType = "pkcs7";

crypto.paddingTypeAndroid = "pkcs5";

crypto.paddingTypeIPhone = "pkcs7";

crypto.passphraseTextWeb = "randomtext";

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

this.view.add(crypto);

}

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

## 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](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_user_guide/Content/C_ComponentsOverview.html) section of the Volt MX Iris User Guide.

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.

* **General**

1. Algorithm Type(algorithmType)

|  |  |
| --- | --- |
| **Description:** | Specifies the type of algorithm that you want to use for the encryption. |
| **Syntax:** | algorithmType |
| **Type:** | * List Selector * String |
| **Read/Write:** | Write |
| **Values:** | * [aes](javascript:void(0);) * [tripledes](javascript:void(0);) * rsa |
| **Default Value:** | aes |
| **Remarks:** | For more information on the encryption algorithms, you can refer to the [Cryptography API](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_api_dev_guide/content/cryptography.html) guide. |
| **Example:** | this.view.componentID.algorithmType = "tripledes"; |

Note: For RSA Algorithm-

* iPhone and iPad-Maximun number of characters :214

Based on key size the maximum number of characters should be counted

For example : If Keysize  is : 2048

No of characters = Keysize/8-42 = 2048/8-42=214((OAEP padding)

* Android-Maximun number of characters : 245

Based on key size the maximum number of characters should be counted

For example : If Keysize  is : 2048

No of characters = Keysize/8-11 = 2048/8-11=245(PKCS1 padding).

2. Encryption Mode(encryptionMode)

|  |  |
| --- | --- |
| **Description:** | Specifies the mode that you want to use to control the encryption. |
| **Syntax:** | encryptionMode |
| **Type:** | * List Selector * String |
| **Read/Write:** | Write |
| **Values:** | * [cbc](javascript:void(0);) * [ecb](javascript:void(0);) |
| **Default Value:** | Ecb |
| **Remarks:** | For more information on the encryption modes, you can refer to the [Cryptography API](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_api_dev_guide/content/cryptography.html) guide. |
| **Example:** | this.view.componentID.encryptionMode = "ecb"; |

3. [Sixteen Digit Initialization Vector](javascript:void(0);)(initializationVector)

|  |  |
| --- | --- |
| **Description:** | Specifies the initialization vector that you want to set for the encryption. |
| **Syntax:** | initializationVector |
| **Type:** | String |
| **Read/Write:** | Write |
| **Default Value:** | "1234567887654321" |
| **Remarks:** | For more information on the initialization vector, you can refer to the [Cryptography API](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_api_dev_guide/content/cryptography.html) guide. |
| **Example:** | this.view.componentID.initializationVector = "1215094983478234"; |

4. [Padding Type](javascript:void(0);)(paddingType)

|  |  |
| --- | --- |
| **Description:** | Specifies the type of padding that you want to use for the encryption. |
| **Syntax:** | paddingType |
| **Type:** | * List Selector * String |
| **Read/Write:** | Write |
| **Values:** | * pkcs7 |
| **Default Value:** | pkcs7 |
| **Remarks:** | For more information on padding types, you can refer to the [Cryptography API](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_api_dev_guide/content/cryptography.html) guide. |
| **Example:** | this.view.componentID.paddingType = "pkcs7"; |

**5. Android**

[Padding Type](javascript:void(0);)(paddingTypeAndroid)

|  |  |
| --- | --- |
| **Description:** | Specifies the type of padding that you want to use for Android. |
| **Syntax:** | paddingTypeAndroid |
| **Type:** | * List Selector * String |
| **Read/Write:** | Write |
| **Values:** | * pkcs5 |
| **Default Value:** | pkcs5 |
| **Remarks:** | For more information on padding types, you can refer to the [Cryptography API](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_api_dev_guide/content/cryptography.html) guide. |
| **Example:** | this.view.componentID.paddingTypeAndroid = "pkcs5"; |

**6. iPhone**

[Padding Type](javascript:void(0);)(paddingTypeIPhone)

|  |  |
| --- | --- |
| **Description:** | Specifies the type of padding that you want to use for iOS. |
| **Syntax:** | paddingTypeIPhone |
| **Type:** | * List Selector * String |
| **Read/Write:** | Write |
| **Values:** | * pkcs7 |
| **Default Value:** | pkcs7 |
| **Remarks:** | For more information on padding types, you can refer to the [Cryptography API](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_api_dev_guide/content/cryptography.html) guide. |
| **Example:** | this.view.componentID.paddingTypeIPhone = "pkcs7"; |

**7. Web**

[Passphrase Text](javascript:void(0);)(passphraseTextWeb)

|  |  |
| --- | --- |
| **Description:** | Specifies the passphrase that you want to use to generate the key for Web channels. |
| **Syntax:** | passphraseTextWeb |
| **Type:** | String |
| **Read/Write:** | Write |
| **Default Value:** | "randomtext" |
| **Example:** | this.view.componentID.passphraseTextWeb = "randomtext"; |

## 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://opensource.hcltechsw.com/volt-mx-docs/docs/tutorials/ActionEditor.html) in the Volt MX Iris User Guide.

1. onErrorCallback

|  |  |
| --- | --- |
| **Description:** | Invoked when any error occurs in the component. |
| **Syntax:** | onErrorCallback |
| **Parameters:** | errObj [JSON]: Information about the error such as the error code and error message. |
| **Example:** | this.view.componentID.onErrorCallback = function(errObj)  {  alert("Error Occurred "+errObj);  }.bind(this); |

## D.API’s

The following APIs pertain to the Crypto component:

1. encryptData

|  |  |
| --- | --- |
| **Description:** | Encrypts the given data based on the component properties. |
| **Syntax:** | encryptData(data) |
| **Parameters:** | *data [String]*: The data that you want to encrypt. |
| **Return Value:** | *encryptedData [String]*: The data that the component encrypts. |
| **Example:** | var data = "This data will now be encrypted.";  var encryptedData = this.view.componentID.encryptData(data);  alert("Encrypted Data: "+encryptedData); |

2. decryptData

|  |  |
| --- | --- |
| **Description:** | Decrypts the given data based on the component properties. |
| **Syntax:** | decryptData() |
| **Parameters:** | data *[String]*: The data that you want to decrypt. |
| **Return Value:** | *decryptedData [String]*: The data that the component decrypts. |
| **Example:** | Var data=<Encrypted Data>;  var decryptedData =  this.view.componentID.decryptData(data);  alert("Decrypted Data: "+decryptedData); |

3. asymmetricEncrypt

|  |  |
| --- | --- |
| **Description:** | Encrypts the given data based on the component properties. |
| **Syntax:** | asymmetricEncrypt() |
| **Parameters:** | *data [String]*: The data that you want to encrypt. |
| **Return Value:** | asyEncryptedData *[String]*: The data that the component encrypts |
| **Example:** | var data = " This data will now be encrypted.";  var asyEncryptedData = this.view.componentID.asymmetricEncrypt (data);  alert("Encrypted Data: "+ asyEncryptedData); |

4. asymmetricDecrypt

|  |  |
| --- | --- |
| **Description:** | Decrypts the given data based on the component properties. |
| **Syntax:** | asymmetricDecrypt() |
| **Parameters:** | *None* |
| **Return Value:** | asyDecryptedData *[String]*: The data that the component decrypts. |
| **Example:** | var asyDecryptedData = this.view.componentID.asymmetricDecrypt ();  alert("Decrypted Data: "+ asyDecryptedData); |

# 4.Revision History

App version 1.0.3:

|  |  |
| --- | --- |
| **Asset Version** | **Modifications** |
| Version 1.0.3 | * Implemented RSA for Mobiles and Tablets. * Improved UI. |
| Version 1.0.2 | * Added support for **Web** channels. * Added the following properties.   + **General** → Padding Type   + **Web** → Passphrase Text * Added a default value for the Initialization Vector property. |
| Version 1.0.1 | * Bug fixes and improvements. |
| Version 1.0.0 | * Initial release of asset and documentation. |

## Limitations

1. For Mobile and tablets -- the encrypted data does not change every time when we encrypt the data.

2. For PWA – the encrypted data changes every time for same input value.

3. RSA the encrypted data changes every time for same input value.

4.RSA is supported only in Mobiles and Tablets not in PWA.