



# HCL Volt MX

## コードを使った Foundry Identity サービスの呼び出し

Student Guide



### HCL Software Academy for HCL Digital Solutions

Creating a new generation of experts

## もくじ

コードを使った Foundry Identity サービスの呼び出し .....	3
前提条件 .....	3
コードで Foundry Identity サービスを呼び出す .....	3

## コードを使った Foundry Identity サービスの呼び出し

このレッスンでは、Foundry Identity サービスを呼び出すためのコードを Iris で記述することを紹介します。Foundry Identity サービスを呼び出すために使用される API について学習します。

このドキュメントでは、このレッスンの実習部分の詳細な手順を説明します。

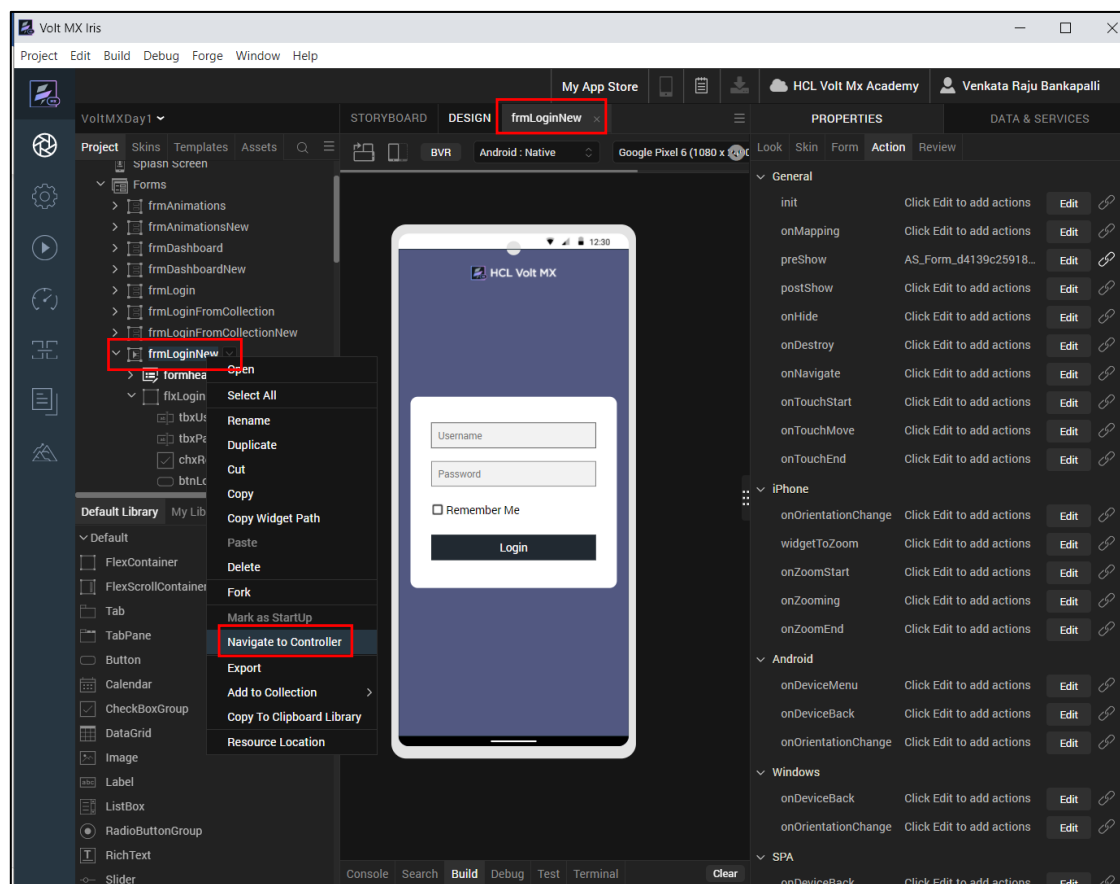
### 前提条件

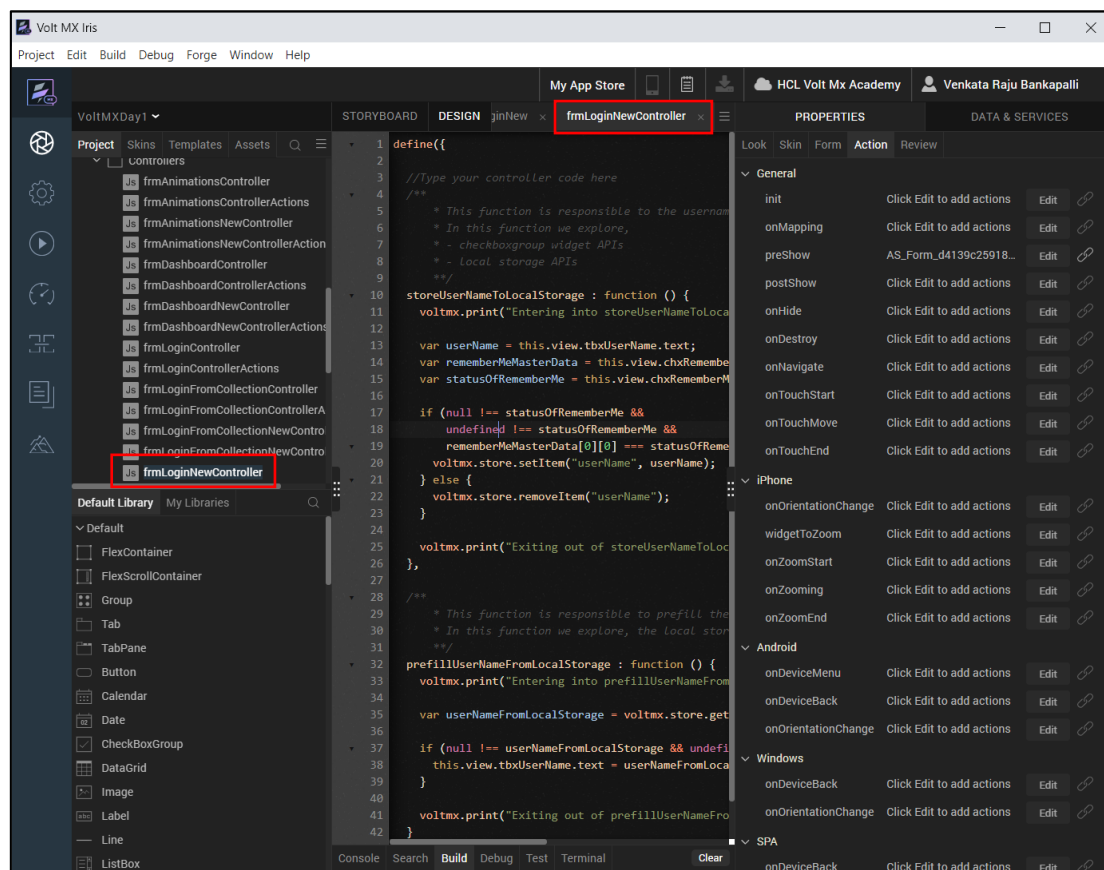
- 前のレッスンの実習手順を完了していること。

## コードで Foundry Identity サービスを呼び出す

### 注意事項

- 前のレッスンで作業していた Iris プロジェクトの上で、以下に説明する手順を引き続き実行します。
- フォーム **frmLoginNew** を開きます。
- フォーム **frmLoginNew** を右クリックし、**Navigate to Controller** をクリックします。すると、このフォームに関連するフォームコントローラが開きます。





- 3つの処理 (Identity サービスの起動、Identity サービスからの成功レスポンスの処理、Identity サービスからの失敗レスポンスの処理) を行うために、以下の3つの関数を追加します。

```
/**
 * この関数は、ID サービスを呼び出す役割を担っています。
 * この関数では、Foundry SDK の API を使用して、Foundry で定義された Identity サービスを Iris から呼び出す方法を見ていきます。
 */
invokelIdentityService : function () {
    voltmx.print("Entering into invokelIdentityService");

    var identityServiceName = "UserRepoldentityService";
    var requestForIdentityService =
{"userid":this.view.tbxCUserName.text,"password":this.view.tbxCPassword.text};

    var sdkCurrentInstance = voltmx.sdk.getCurrentInstance();
    var identityService = sdkCurrentInstance.getIdentityService(identityServiceName);
    identityService.login(requestForIdentityService, this.identitySuccessResponseHandler.bind(this),
this.identityFailureResponseHandler.bind(this));

    voltmx.print("Exiting out of invokelIdentityService");
},

/**
 * この関数は、Foundry Identity サービスからの成功レスポンスの処理を担当します。
 * この関数では、Identity サービスからの成功レスポンスを処理する方法について説明します。
 * この関数では、Identity サービスからの成功レスポンスを処理する方法について説明します。
 */
```

```

/**/
identitySuccessResponseHandler : function (successResponse) {
    voltmx.print("Entering into identitySuccessResponseHandler");

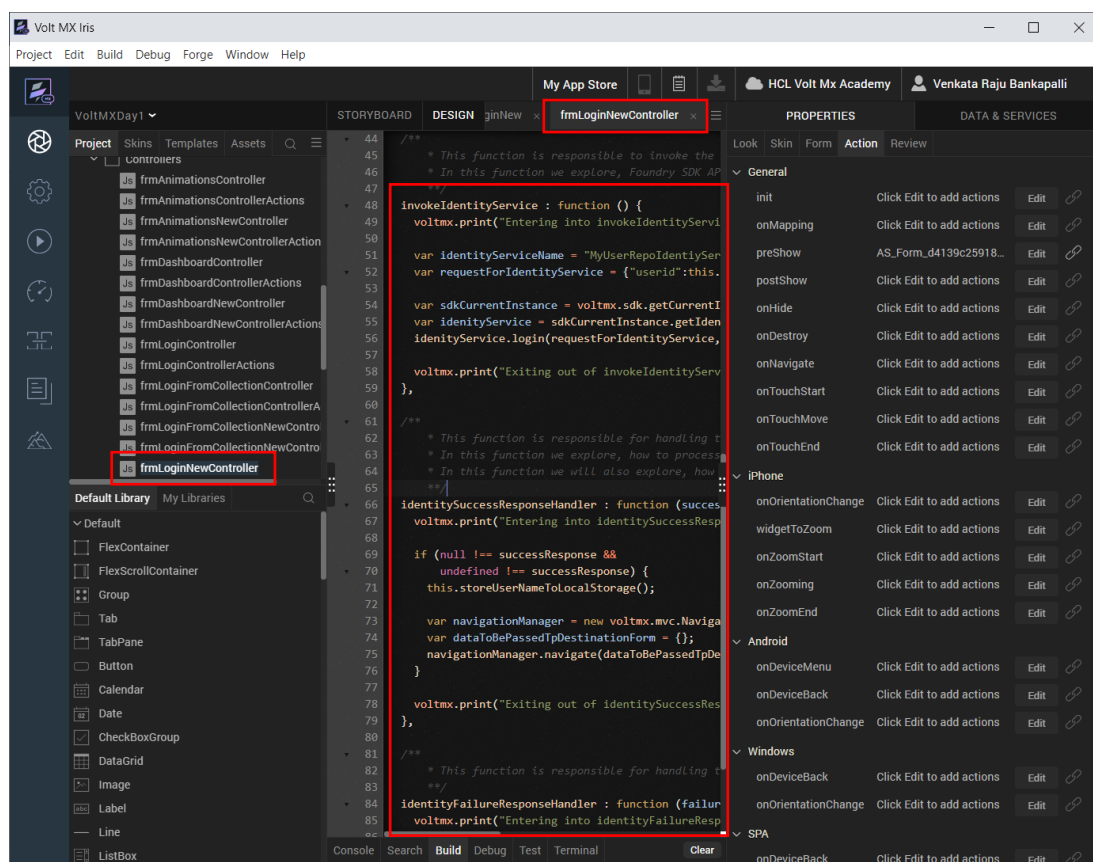
    if (null !== successResponse &&
        undefined !== successResponse) {
        this.storeUserNameToLocalStorage();

        var navigationManager = new voltmx.mvc.Navigation("frmDashboardNew");
        var dataToBePassedTpDestinationForm = {};
        navigationManager.navigate(dataToBePassedTpDestinationForm);
    }

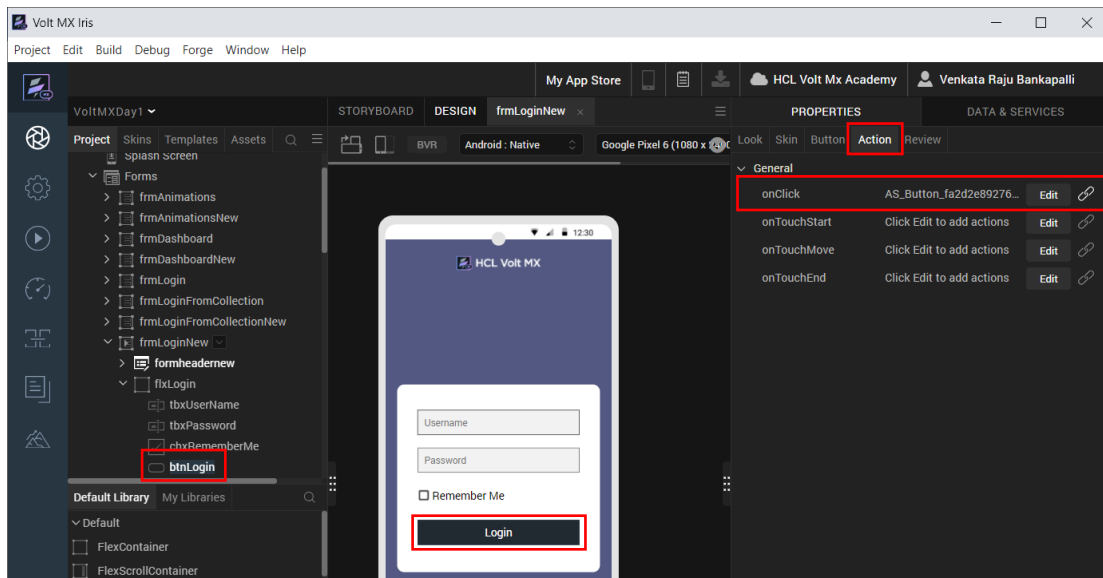
    voltmx.print("Exiting out of identitySuccessResponseHandler");
},

/**
 * この関数は、Foundry Identity サービスからの失敗レスポンスを処理する役割を担っています。
 */
identityFailureResponseHandler : function (failureResponse) {
    voltmx.print("Entering into identityFailureResponseHandler");
    alert ("LOGIN FAILED");
    voltmx.print("Exiting out of identityFailureResponseHandler");
},

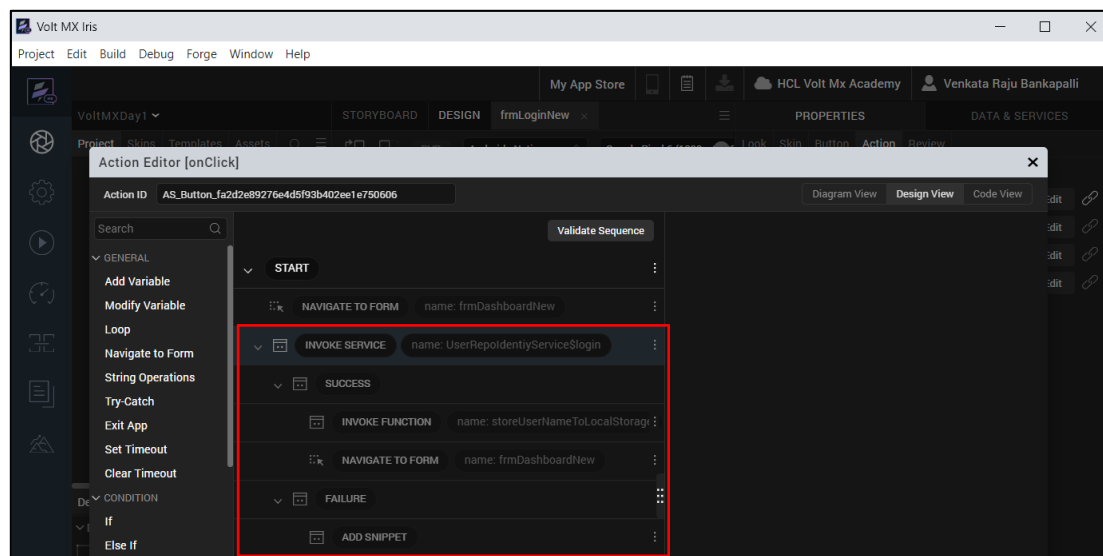
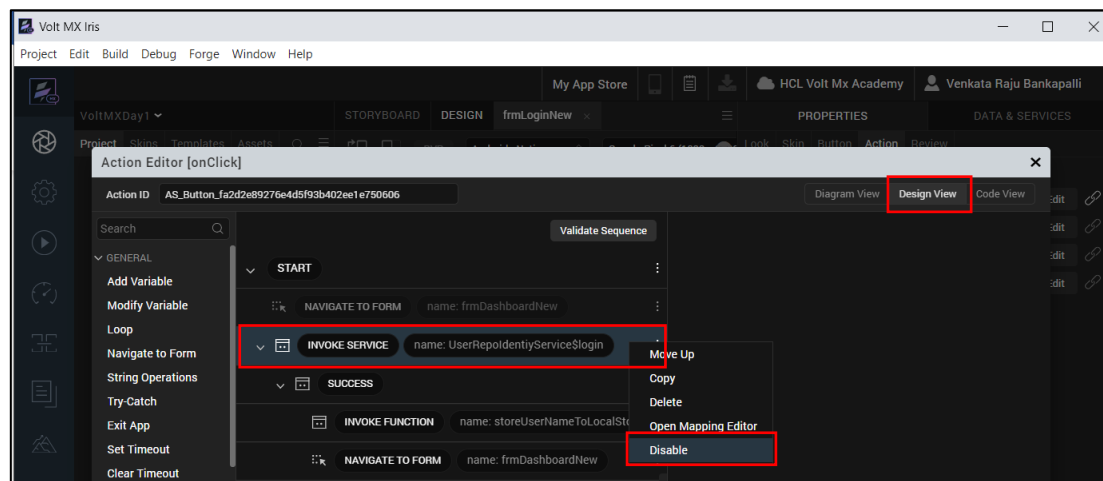
```



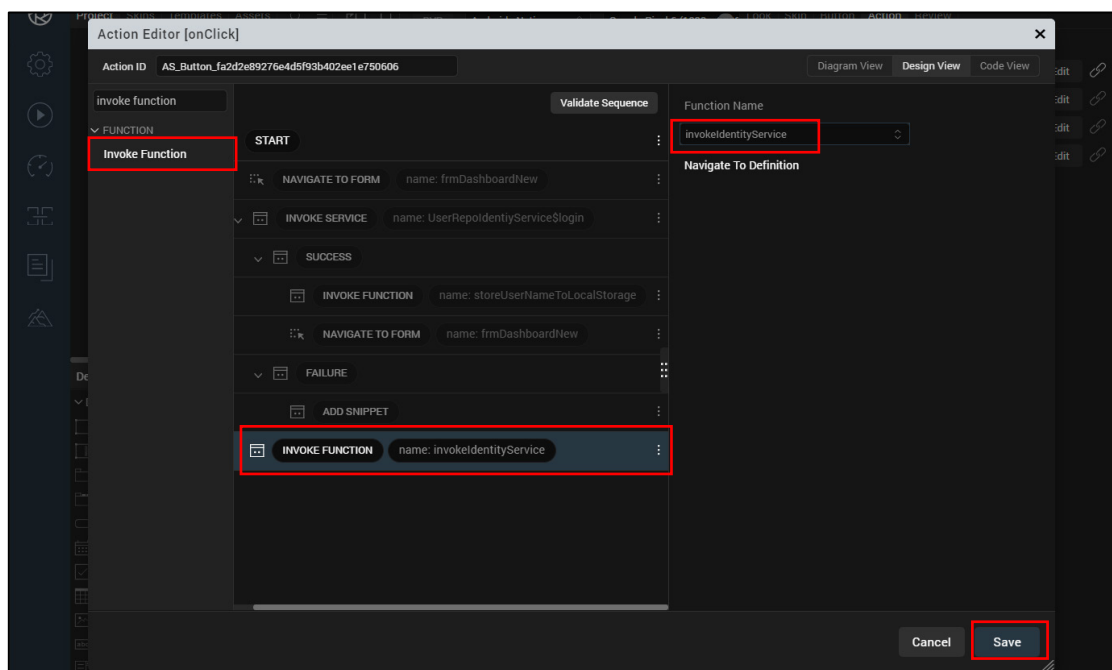
- アウトラインで **frmLoginNew > flxLogin > btnLogin** を選択し、プロパティの **Action** タブで **onClick > Edit** をクリックして、btnLogin ボタンウィジェットの onClick イベントを設定します。



- Action Editor で **Design View** に切り替え、既存のアクションシーケンスを無効化します。



- **Invoke Function** アクションを検索し、選択します。
- アクションシーケンスに追加されたアクションが表示されたら、右側のドロップダウンから **invokeIdentityService** という関数を選択します。
- **Save** をクリックし、アクションシーケンスを保存します。



- **Build > Run Live Preview** メニューでライブプレビュービルドを生成し、フォームコントローラのコードで Foundry Identity サービスの呼び出しをテストします。

注意:

- ライブプレビューの設定で以前に選択したチャネル/プラットフォーム/アプリケーションの種類はそのまま残ります。また、ライブプレビュービルドは、これらの選択されたチャネル/プラットフォーム/アプリケーションの種類に生成されます。
- アダプティブウェブのライブプレビュービルドは、お使いのマシンで開く **Volt MX** アプリのプレビューを使用してテストします。
- Wi-fi モードの **Volt MX** アプリプレビューアプリを使用して、IP アドレスを入力するか、バーコードをスキャンして、ネイティブライブプレビュービルドをテストします。
- スマホが接続されているネットワークが、**Volt MX Iris** が動作しているマシンのネットワークと通信できる必要があることに注意してください。

おめでとうございます。このレッスンのハンズオンは終了です。