HCL Volt MX

コントラクトありのコンポーネント

Student Guide

もくじ

コントラクトありのコンポーネント

このレッスンでは、Iris でコントラクトを使用したコンポーネントを作成し、使用する方法を紹介します。

このレッスンでは、以下について学習します。

- コントラクトを使用したコンポーネントの作成
- コントラクトを使用したコンポーネントの使用

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について説明します。

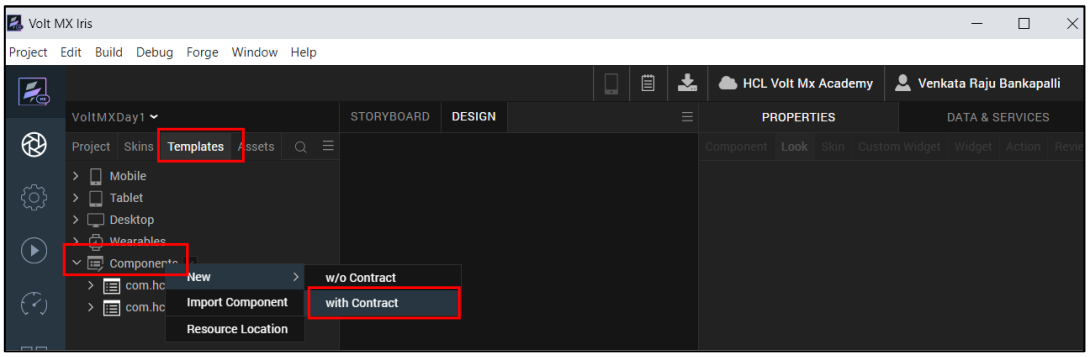
前提条件

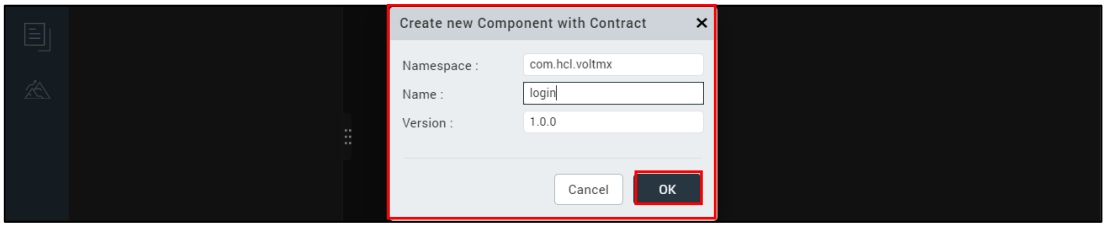
- 前のレッスンのハンズオンのステップを完了していること
- コントラクトを持つコンポーネント UI の作成

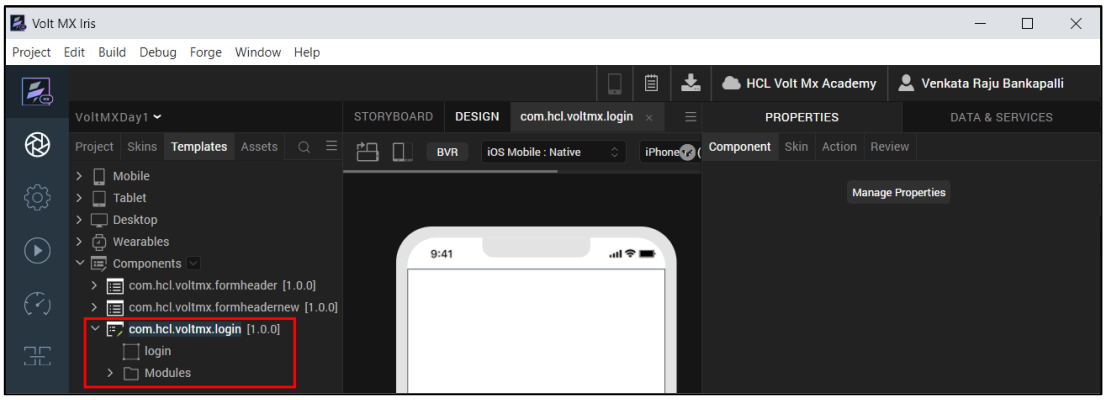
コントラクトを持つコンポーネント UI の作成

> 注意事項
>
> - 前のレッスンで作業していた Iris プロジェクトを使って、以下の手順を引き続き行ってください。

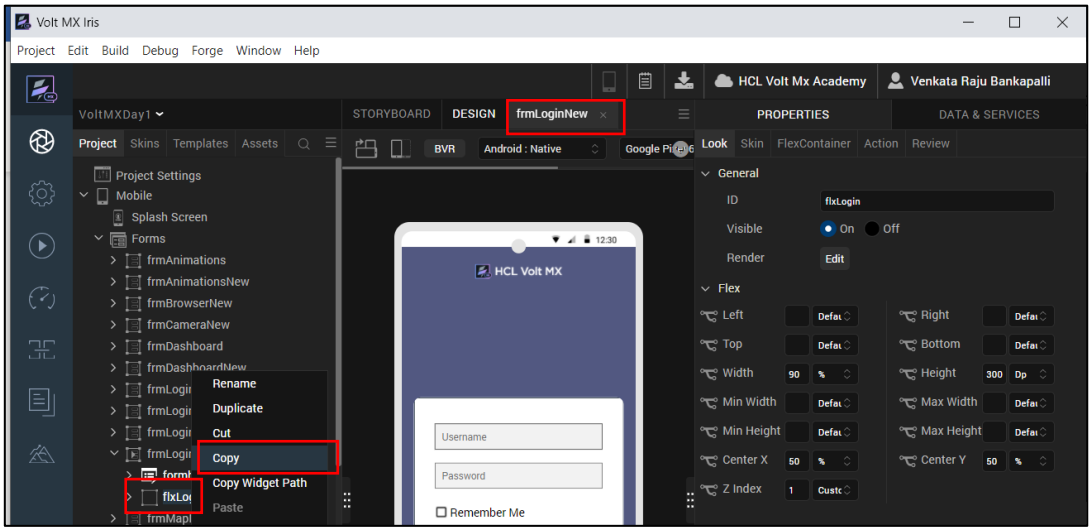
- **Templates** > **Components** に移動します。
- コントラクトを持つコンポーネントを作成します。
  - **namespace** の値を **com.hcl.voltmx** とします。
  - **name** の値を **login** にします。

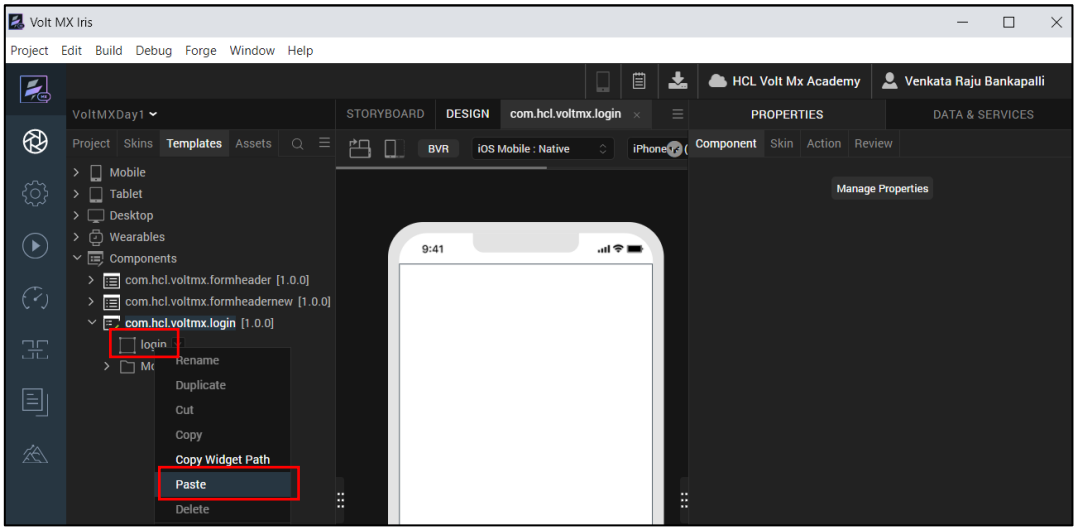


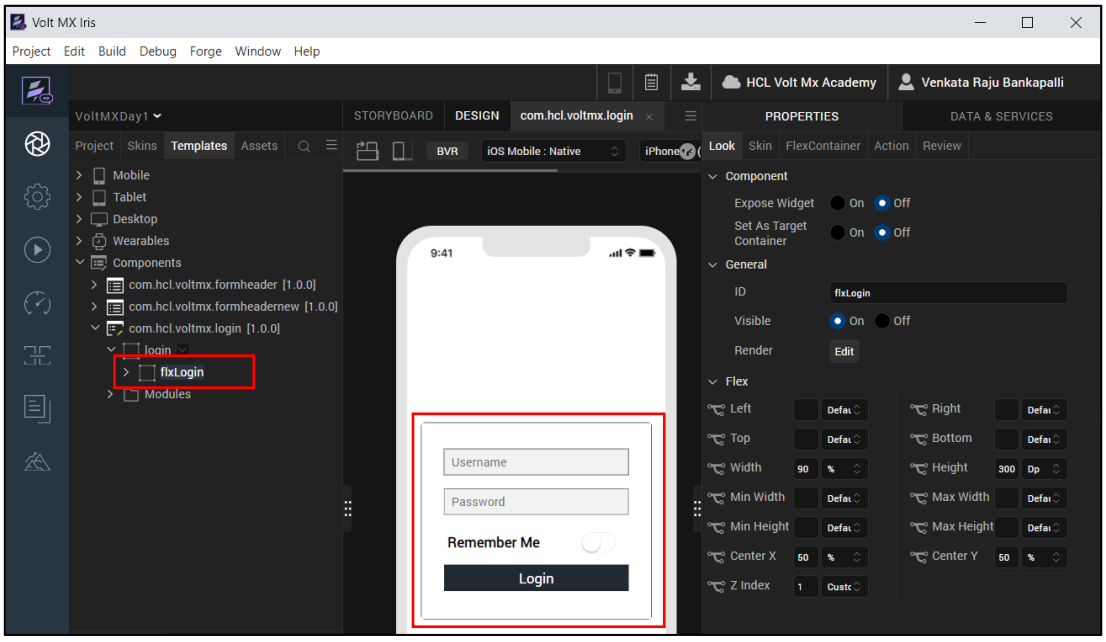




- **frmLoginNew** フォームを開きます。
- このフォームから **flxLogin** ウィジェットをコピーし、コンポーネント内の **login** ウィジェットに貼り付けます。

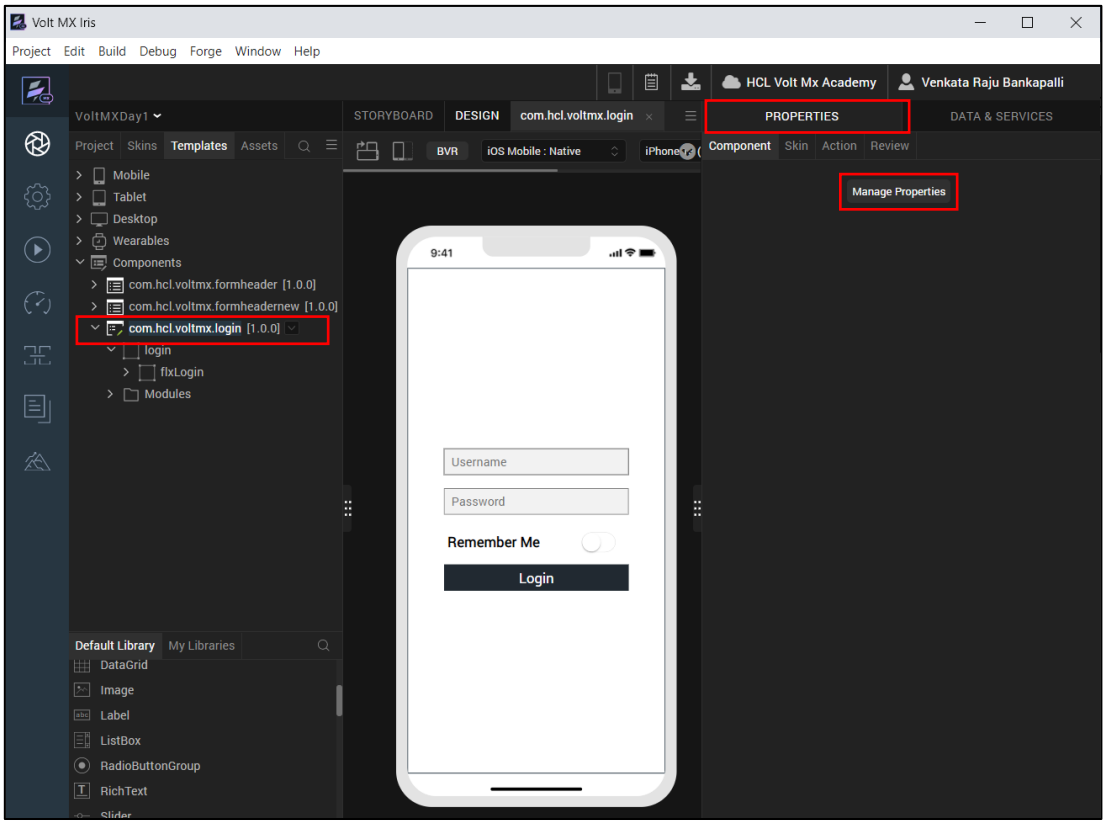




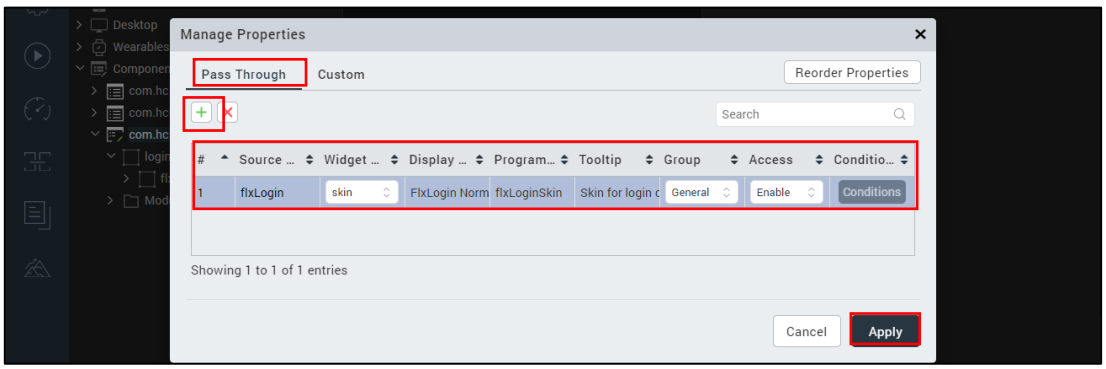


コンポーネント内のウィジェットのプロパティをコントラクトの一部として公開する

- **com.hcl.voltmx.login** コンポーネントを選択します。
- **Properties** > **Manage Properties** に移動します。

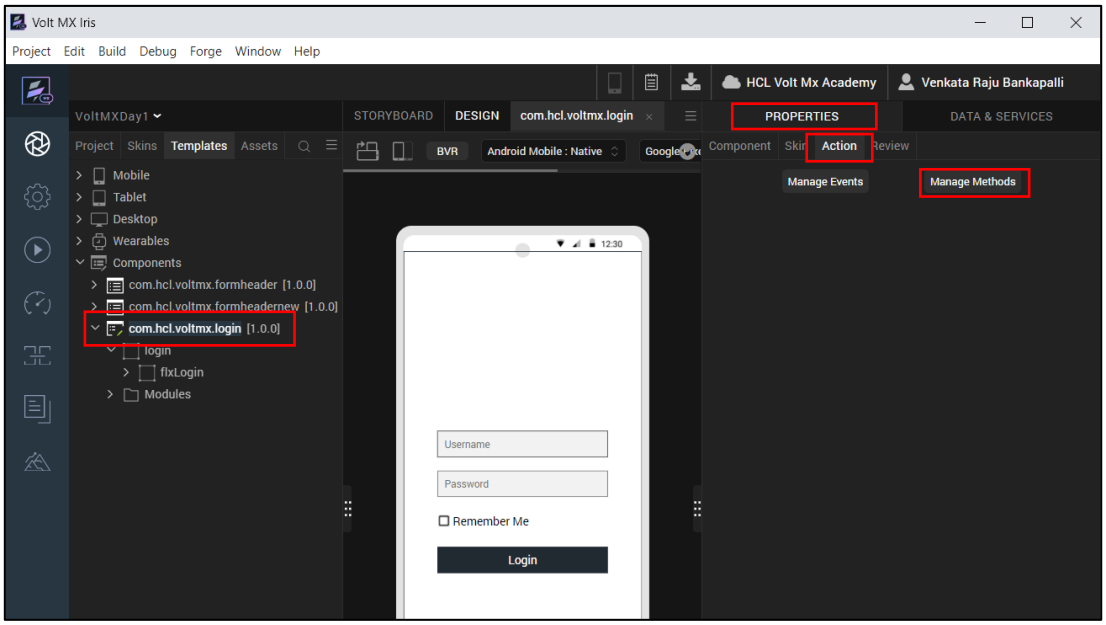


- **Pass Through** タブ で、＋ (**Add**) をクリックして、Pass Through プロパティを設定します。
  - **Source** として **flxLogin** を選択します。
  - **Widget Property** として **skin** を選択します。
  - **Display Name** の値を **FlxLogin Normal Skin** とします。
  - **Programmatic Name** に **flxLoginSkin** を指定します。
  - **Tool Tip** の値を **Skin for login container** に設定します。
- **Apply** をクリックします。

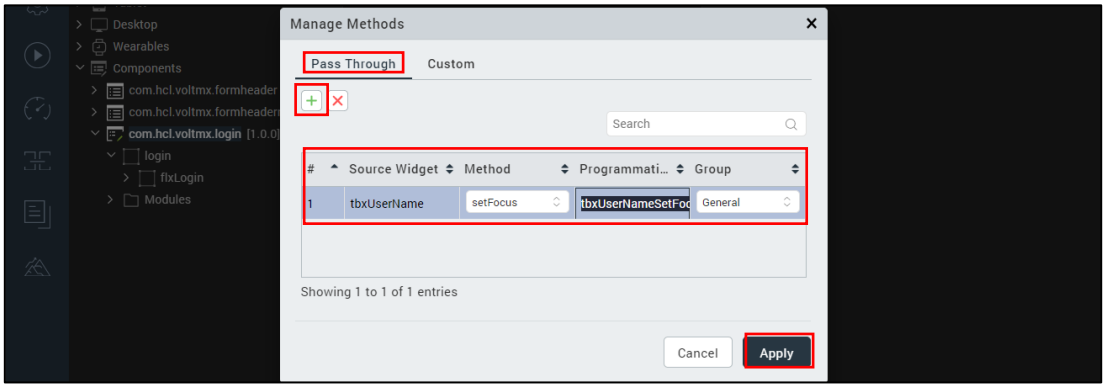


コントラクトの一部として、コンポーネント内のウィジェットのメソッドを公開する

- **com.hcl.voltmx.login** コンポーネントを選択します。
- **Properties** > **Actions** > **Manage Methods** をクリックします。

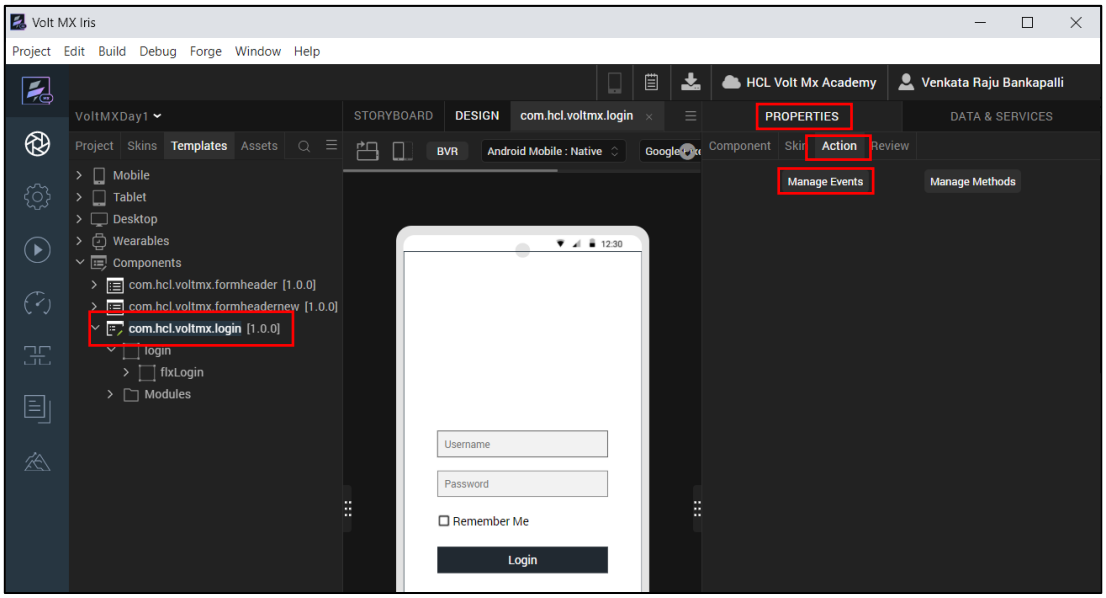


- **Pass Through** メソッドの下にある、＋ (**Add**) をクリックして、Pass Through メソッドを設定します。
  - **Source** を **tbxUserName** に設定します。
  - **Method** を **setFocus** に設定します。
  - **Programmatic Name** の値を **tbxUserNameSetFocus** に設定します。
- **Apply** をクリックします。

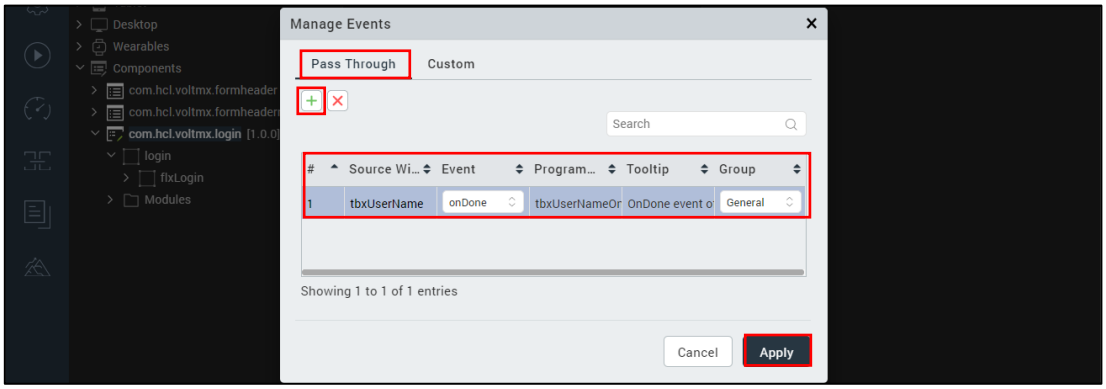


コントラクトの一部として、コンポーネント内のウィジェットのイベントを公開する

- **com.hcl.voltmx.login** コンポーネントを選択します。
- **Properties** > **Actions** > **Manage Events** を選択します。

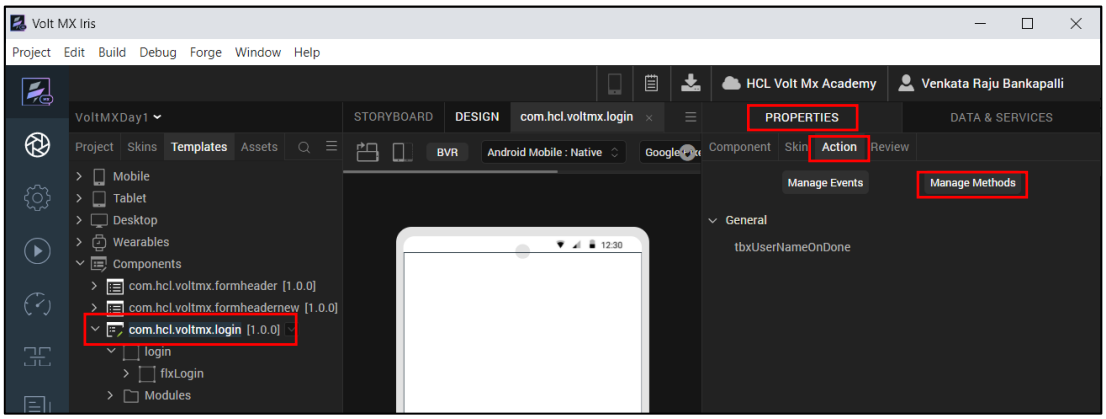


- **Pass Through events** で、＋(**Add**) をクリックして、Pass Through イベントを構成します。
  - **Source** を **tbxUserName** と選択します。
  - **Event** を **onDone** に設定します。
  - **Programatic Name** の値を **tbxUserNameOnDone** とします。
  - **Tool Tip** の値を **OnDone event of tbxUserName** とします。
- **Apply** をクリックします。

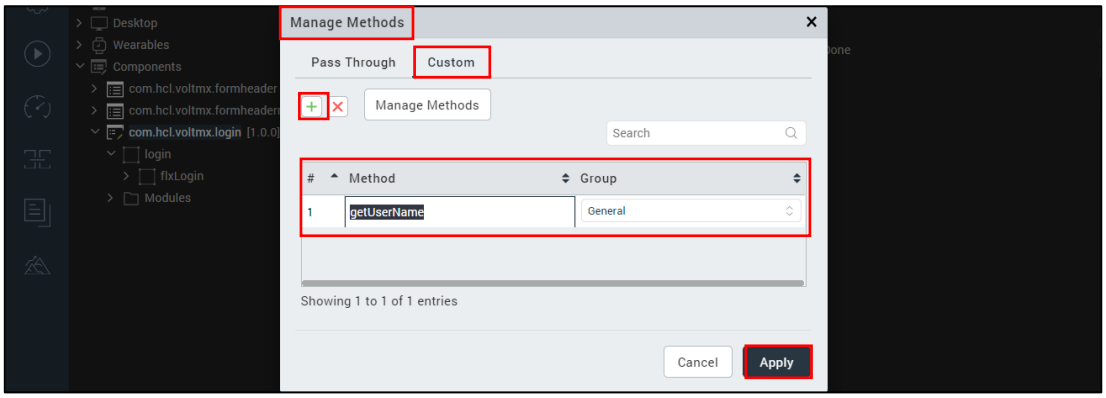


コントラクトの一部としてカスタムメソッドを公開する

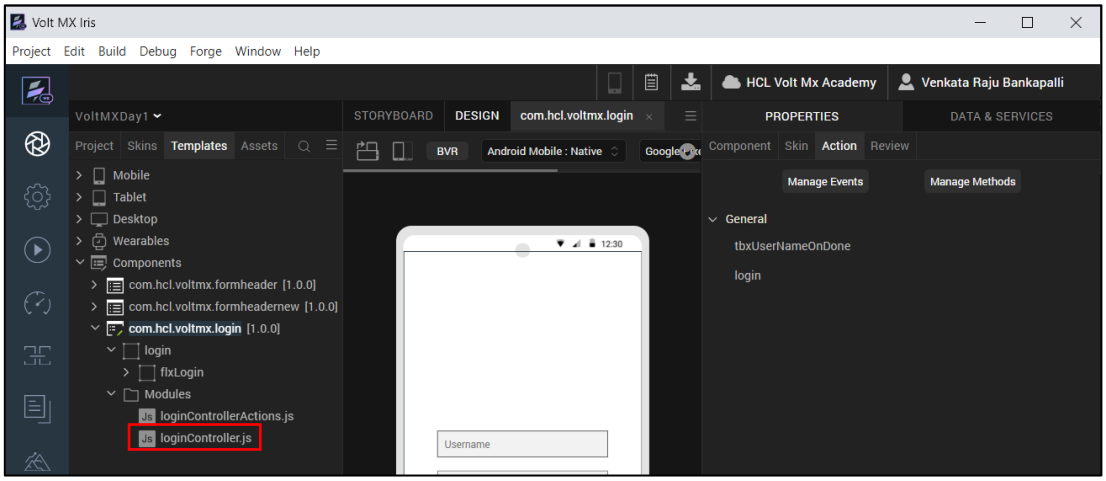
- コントラクトの一部として custom method を公開します。
- **com.hcl.voltmx.login** コンポーネントを選択します。
- **Properties** > **Actions** > **Manage Methods** に移動します。



- **custom method** で、＋ (**Add**) をクリックします。
- **custom method** を設定します。
- **Add** をクリックします。
- **Method** の値を **getUserName** とします。

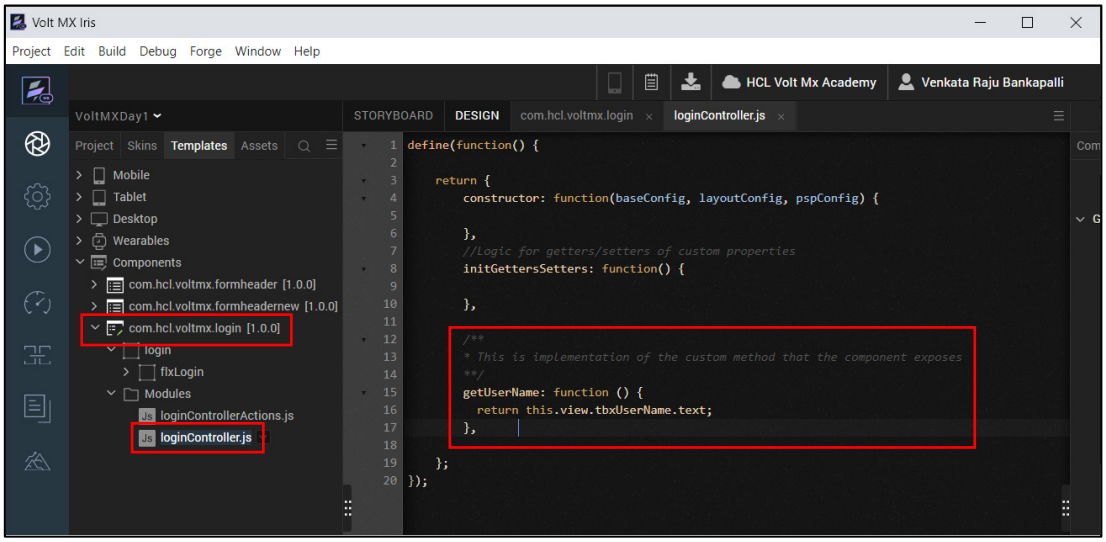


- **com.hcl.voltmx.login** コンポーネント > **Modules** > **loginController.js** を開きます。



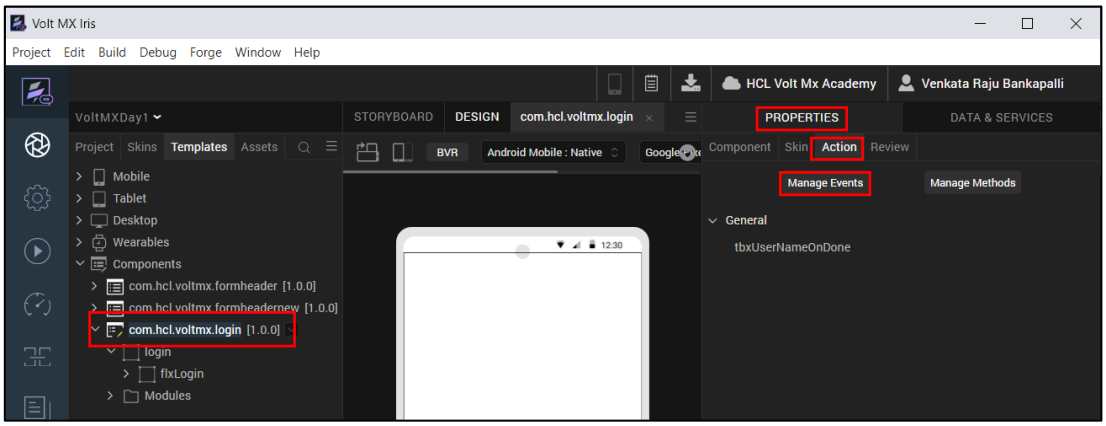
- 以下のコードを component controller に**追加**します。ペースト先に注意してください。

```
/**
* これは、コンポーネントが公開するカスタムメソッドの実装です。
**/
getUserName: function () {
  return this.view.tbxUserName.text;
},
```

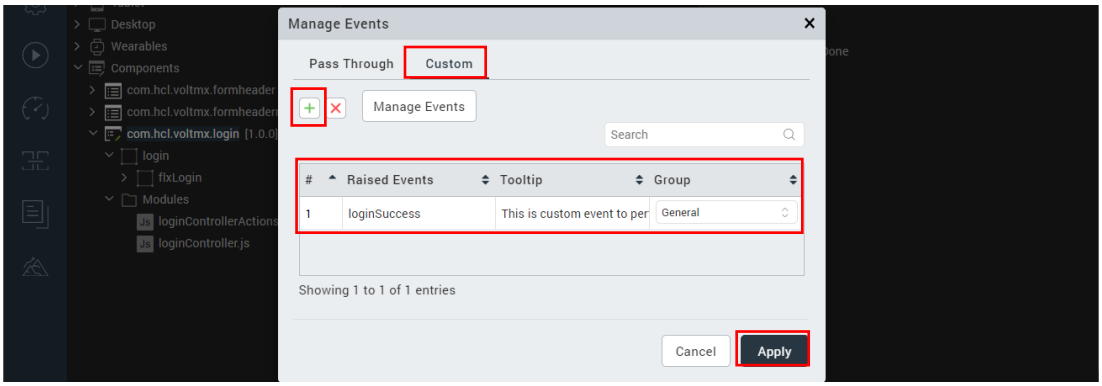


コントラクトの一部としてカスタムイベントを公開する

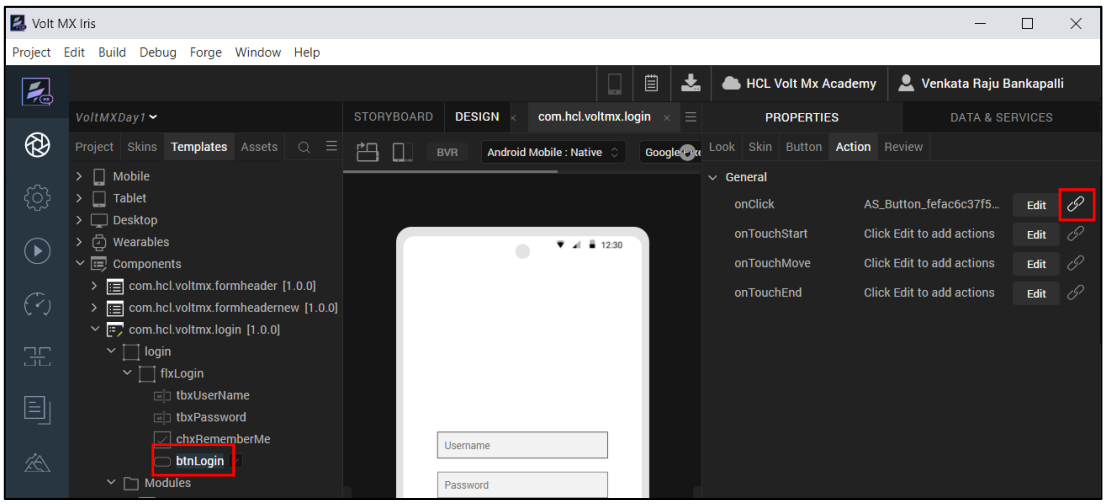
- コントラクトの一部として custom event を公開します。
- **com.hcl.voltmx.login** コンポーネントを選択します。
- **Properties** > **Actions** > **Manage Events** に移動します。



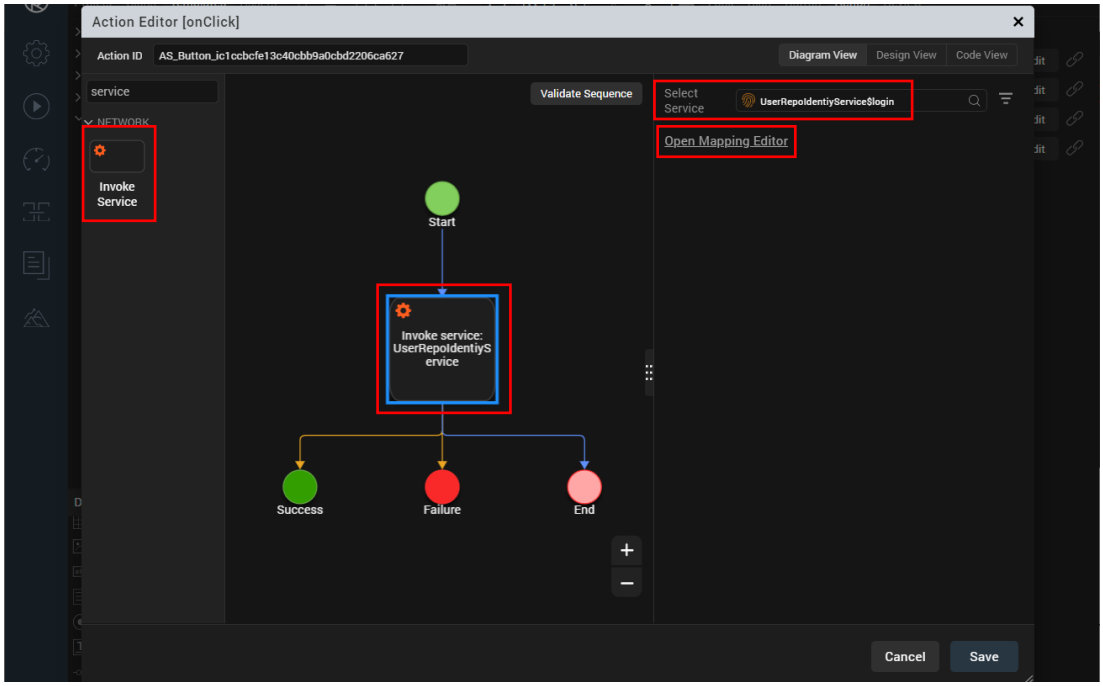
- **Custom** タブの下にある、**Add** をクリックして、custom event を設定します。
  - **Raised Event** の値を **loginSuccess** とします。
  - **Tool Tip** の値を **This is custom event to perform login** に設定します。
- **Apply** をクリックします。



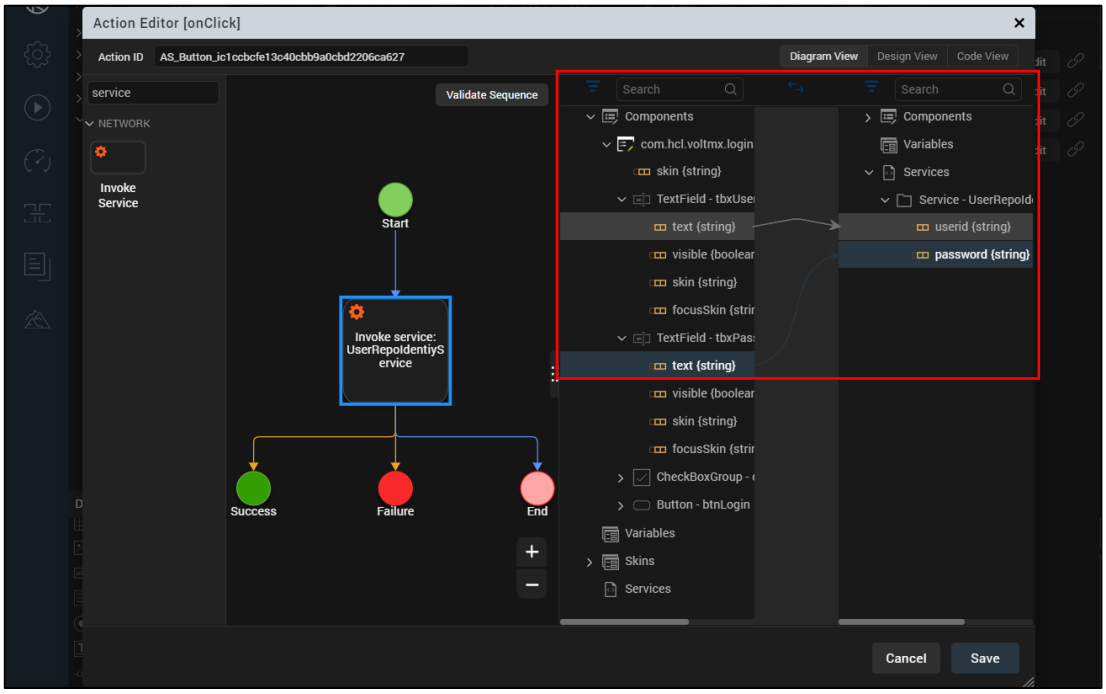
- ボタン **btnLogin** を選択し、ボタンの **onClick** イベントの **Action** のシーケンスを削除します。



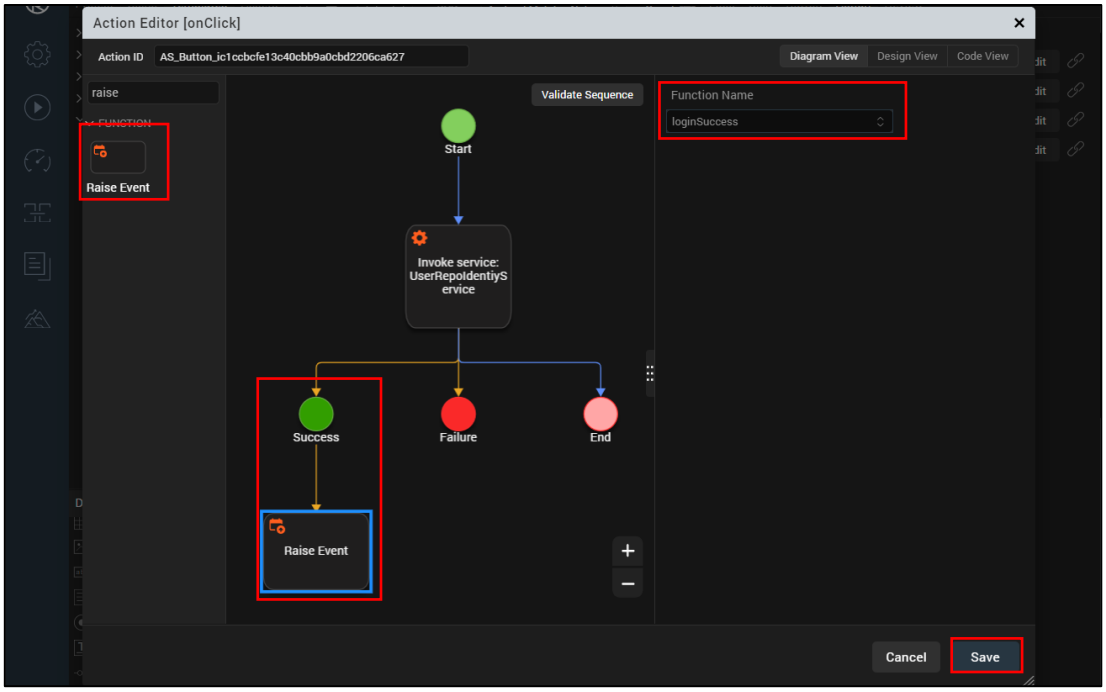
- Identity サービスを起動し、カスタムの loginSuccess イベントを発生させるようにアクションシーケンスを設定します。
- **btnLogin** の **onClick** イベントをクリックします。
- **Invoke Service** アクションを設定し、Identity サービスの **UserRepoIdentiyService$login** を選択します。



- **tbxUserName** と **tbxPassword** から、**username** と **password** の入力パラメータへのマッピングを設定します。



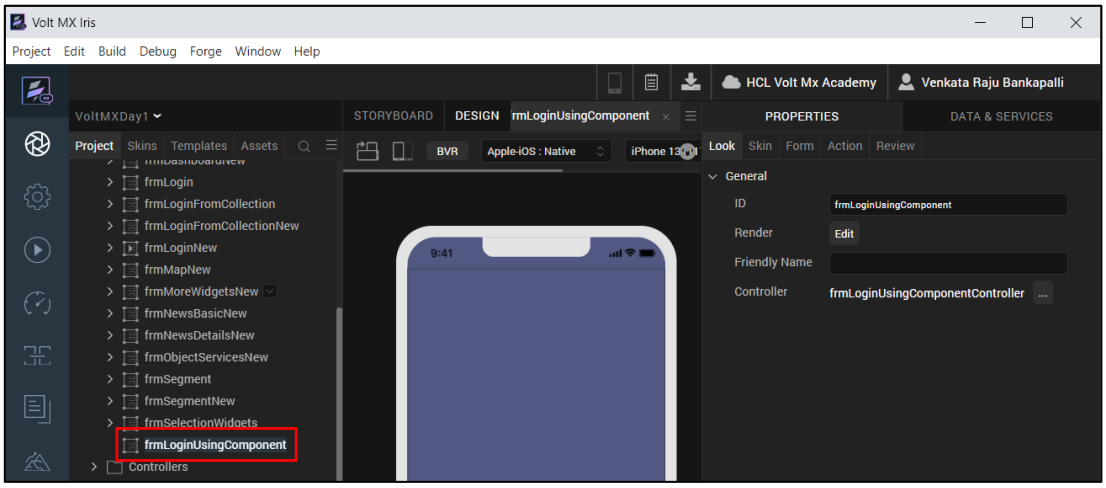
- **Success callback** から **loginSuccess** イベントを発生させます。これには **Raise Event** アクションを使用します。



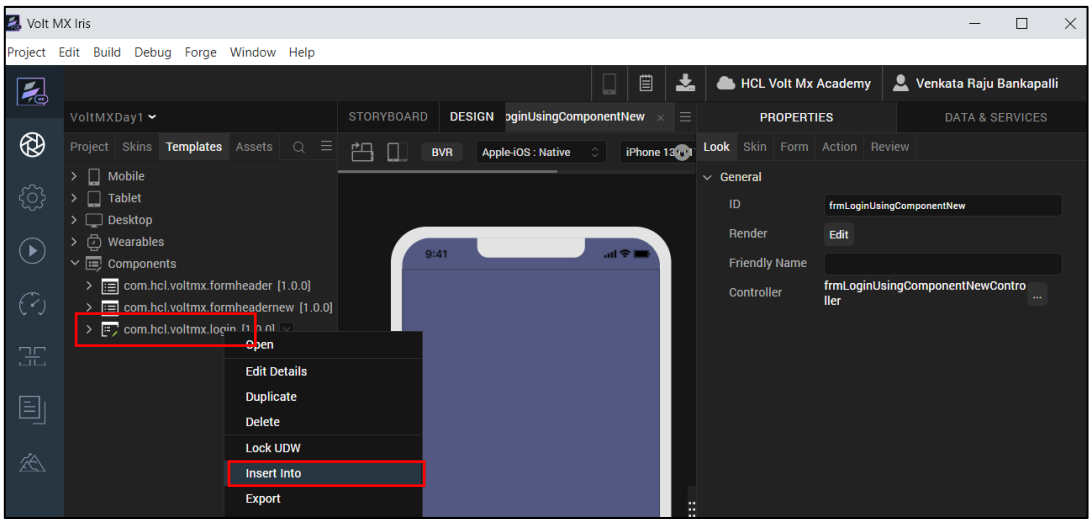
- **Save** をクリックします。

コントラクトを含むコンポーネントをプロジェクトで使用する

- 新しいフォーム **frmLoginUsingComponentNew** を作成します。



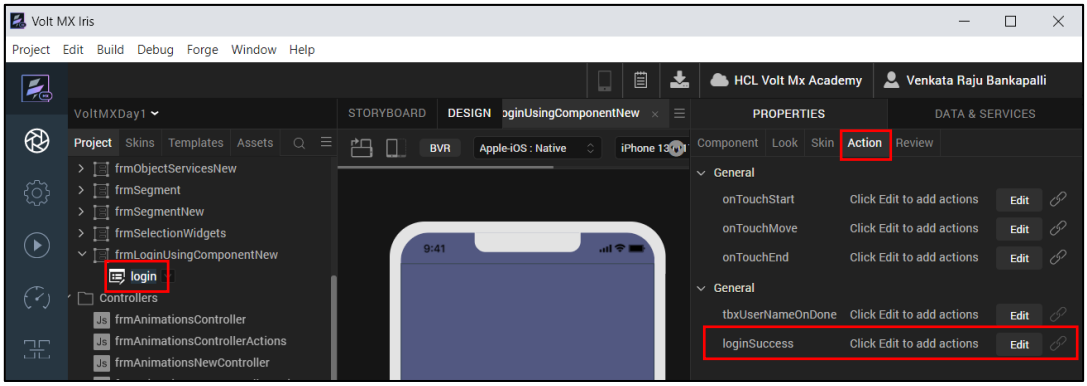
- このフォームにコンポーネント **com.hcl.voltmx.login** を挿入します。

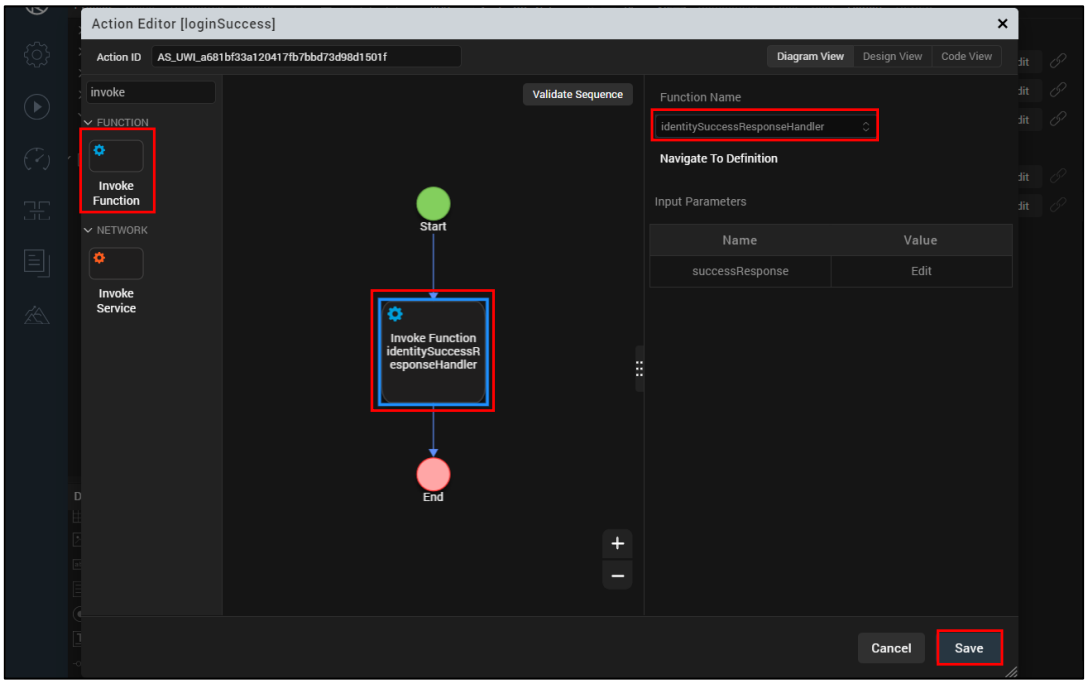


- 以下のコードスニペットを **frmLoginUsingComponentNew** フォームのコントローラに追加します。

```
/**
 * This function is responsible for handling the success response from Foundry identity service
 * In this function we explore, how to process the success response from identity service
 * In this function we will also explore, how to navigate to a form through code
 **/
identitySuccessResponseHandler : function (successResponse) {
        voltmx.print("Entering into identitySuccessResponseHandler");
        var navigationManager = new voltmx.mvc.Navigation("frmDashboardNew");
        var dataToBePassedTpDestinationForm = {};
        navigationManager.navigate(dataToBePassedTpDestinationForm);
}
```

- コンポーネントの **loginSuccess** イベントに **identitySuccessResponseHandler** 関数を割り当てます。





- **frmLoginUsingComponentNew** フォームをスタートアップフォームとしてマークします。
- **Build** > **Run Live Preview** メニューでライブプレビュービルドを生成し、**frmLoginUsingComponen** を使ってログインテストをします。

注意事項

- ライブプレビューの設定で以前に選択したチャンネル／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。そして、ライブプレビュービルドは、これらの選択されたチャンネル/プラットフォーム/アプリケーションの種類で生成されます。

おめでとうございます。あなたはこのレッスンのハンズオンを完了しました。