HCL Software Academy
for Digital Solutions

HCL

HCL Volt MX

コントラクトありのコンポーネント

HVMX-BC-500 Volt MX Iris Collaboration

HCL SOFTWARE

コントラクトを持つコンポーネント – 機能

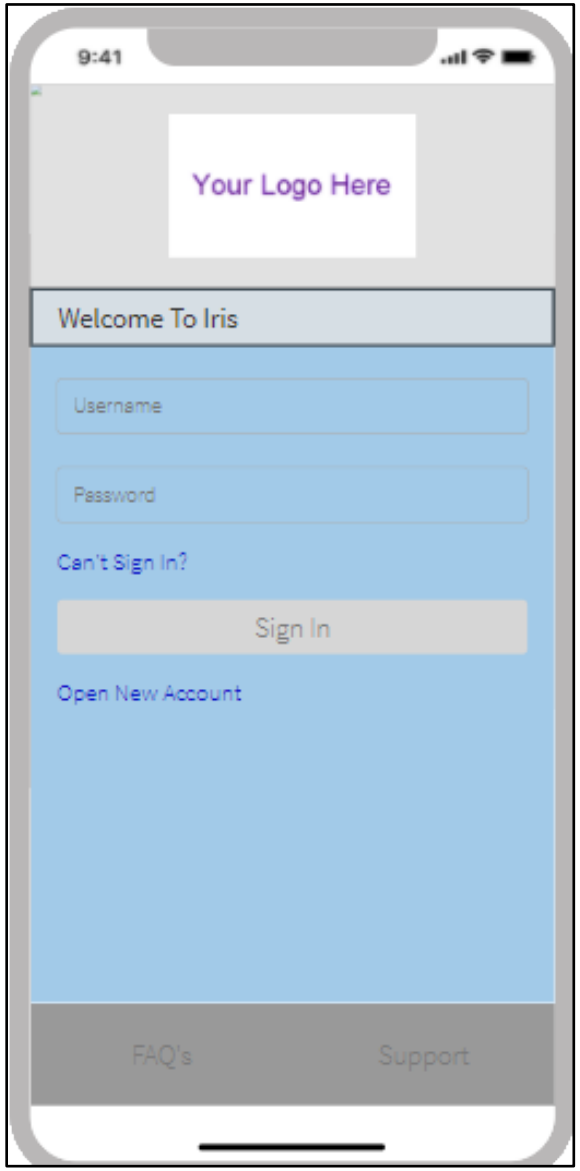
- コントラクトを持つコンポーネントは、以下を持ちます。
  - 独自の名前空間とクラス名
  - UIはビューファイルで、ビジネスロジックはコントローラファイルで、ビジネスロジックをコントローラファイルに記述
- UI階層全体とビジネスロジックは、親コンテナからアクセス可能です。
  - this.view.parent は、コントローラ内の親ビューへのアクセスを提供しませんが、親コントローラの widget.parent は、その親ビューを提供します。
- コントラクトを持つコンポーネントは、以下のことが可能です。
  - あるプロジェクトからエクスポートして、別のプロジェクトにインポートできます。

ユースケースの定義

- ログイン画面
  - 複数のアプリケーションで使用できるログイン画面を作成します。
  - 認証にはUser Directory Identity Serviceを使用します。
  - 画面を作成し、設定用のプロパティを公開します。

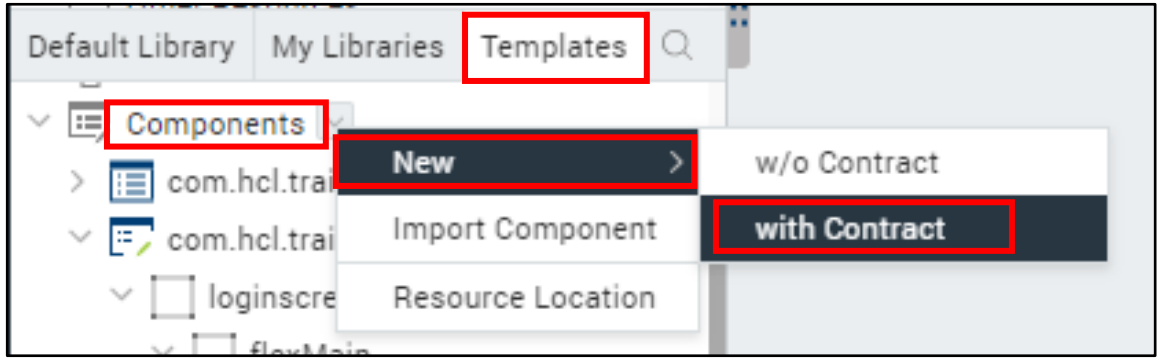
ユースケースの定義（つづき）

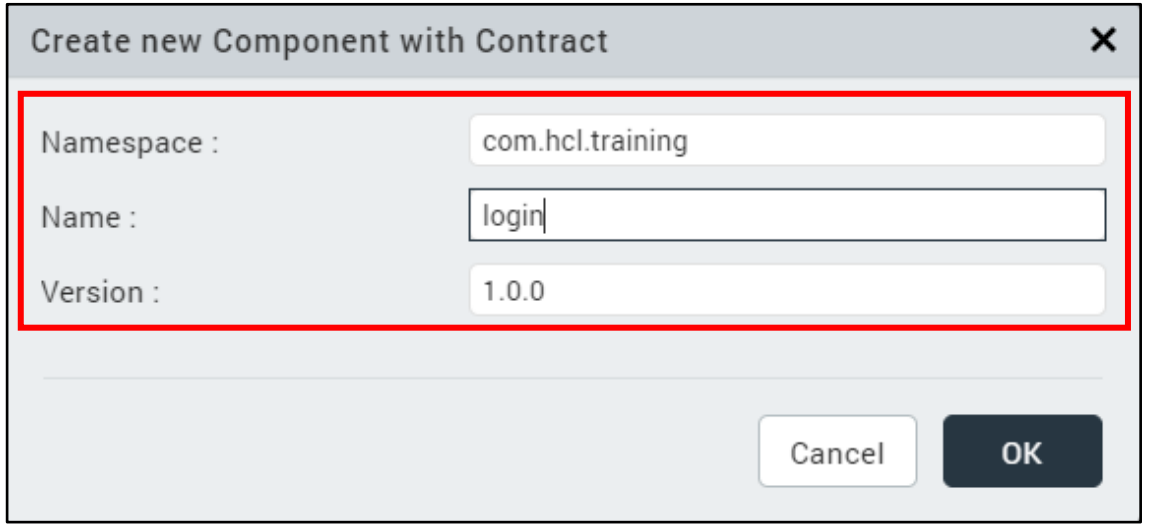
- 画面上の以下のリストの要素の可視性、寸法、位置、テキスト、色、フォントファミリ、フォントサイズを設定します。
    - ユーザー名テキストボックス
    - パスワードテキストボックス
    - サインインボタン
    - サインインできないオプション
    - アカウント作成オプション
    - フッターのオプション
- ログイン成功時のコールバックの定義
- ログイン失敗時のコールバックの定義
- ユーザー名とパスワードを検証するために使用する関数を定義する。



新しいプロジェクトの作成

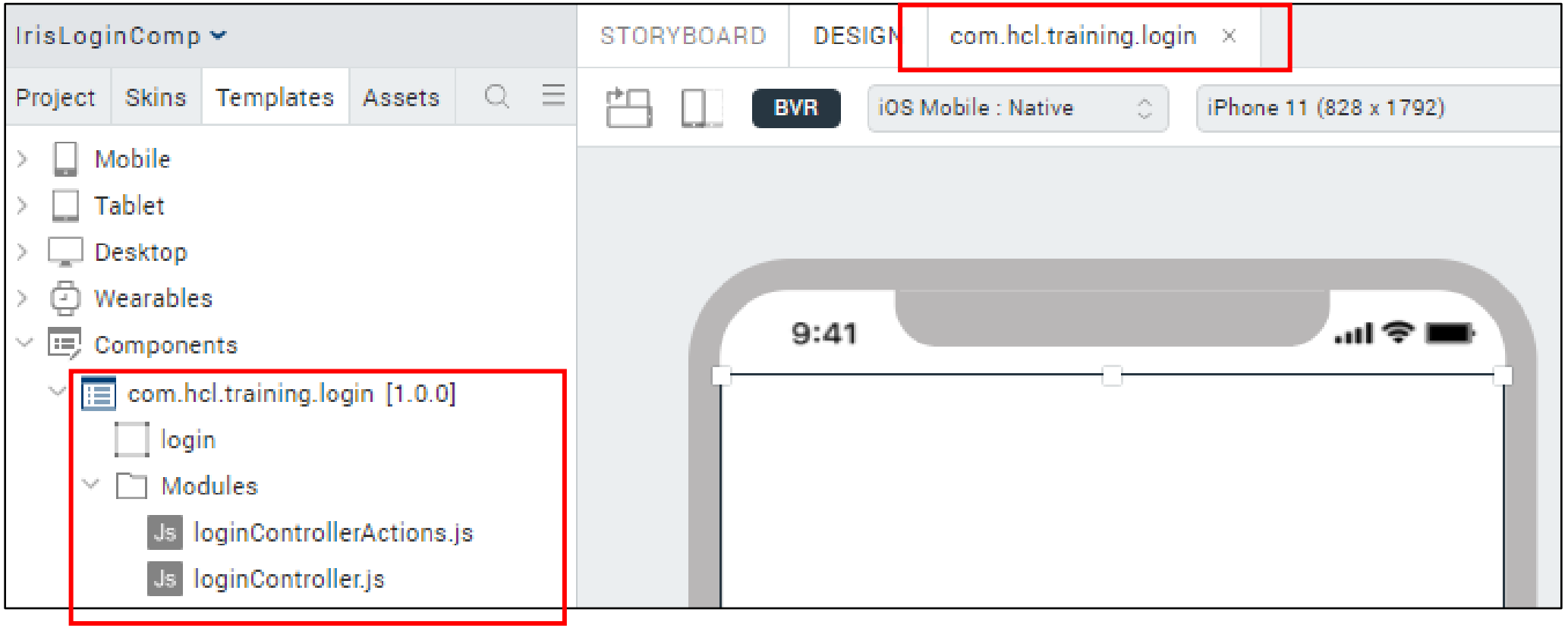
- New project を作成します。
- Templatesセクションに移動し、Componentsをクリックします。
- Namespace, Name に値を入れて OK をクリックします。





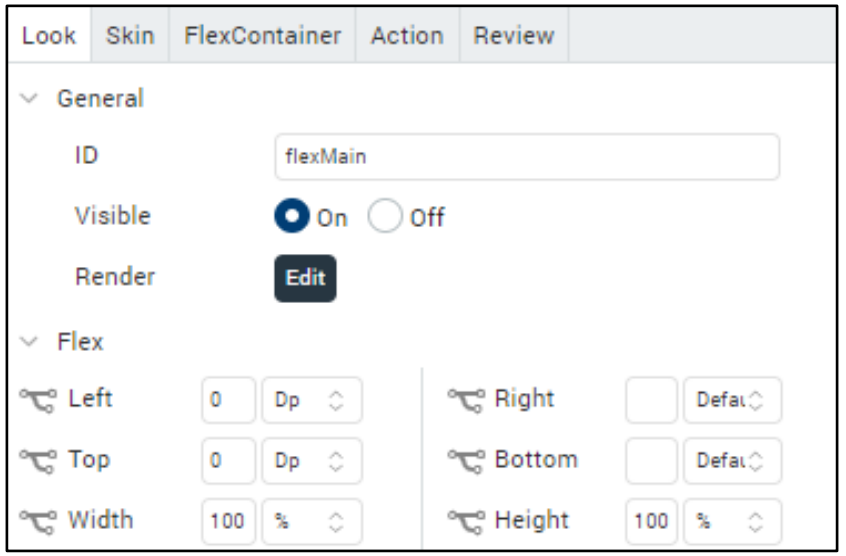
新しいプロジェクトの作成（つづき）

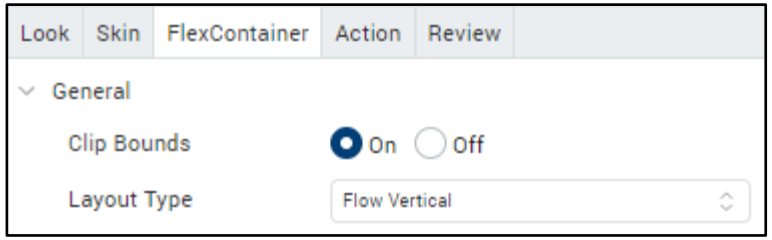
- 以下は、コンポーネントが作成される様子です。
  - コンポーネントを作成すると、フレックスコンテナ、コントローラファイル、コントローラアクションファイルが作成されます。

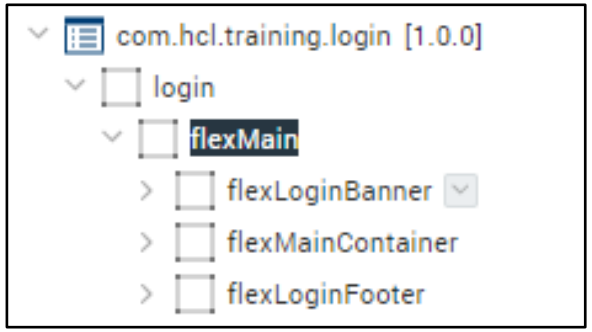


コンポーネントUIの作成

- フレックスコンテナをコンポーネント内に’flexMain’として追加します。
  - 位置は、左0dp、上0dpです。
  - 寸法は、高さ100%、幅100%で。
- このフレックスコンテナのレイアウトタイププロパティの値を Flow Vertical とします。
- flexMainに、3つのFlexContainerを追加します。
  - flexLoginBanner
  - flexMainContainer
  - flexLoginFooter
- プレビューは次のスライドを参照してください。

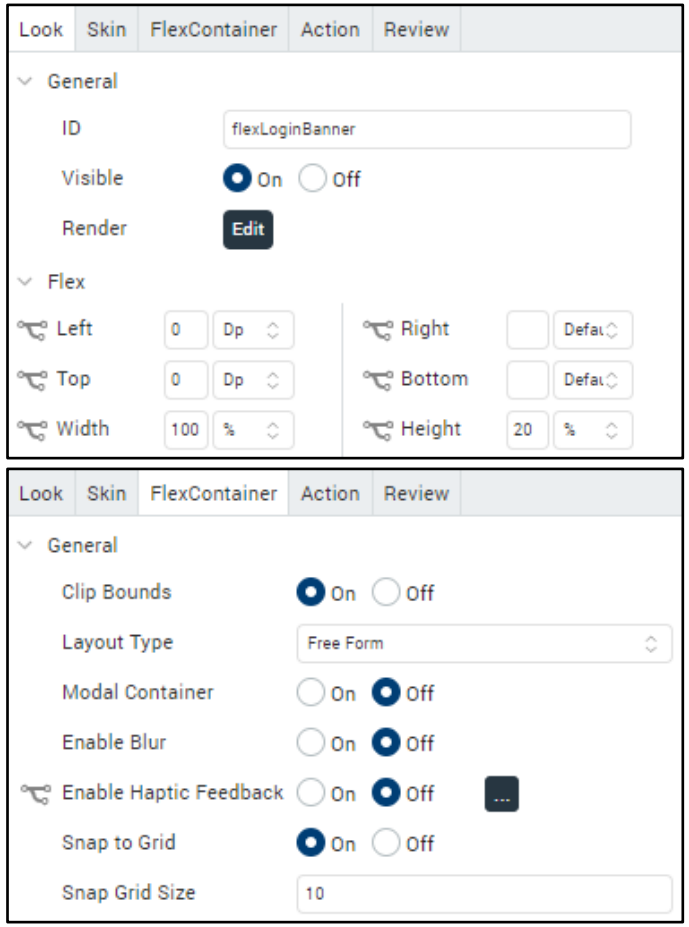




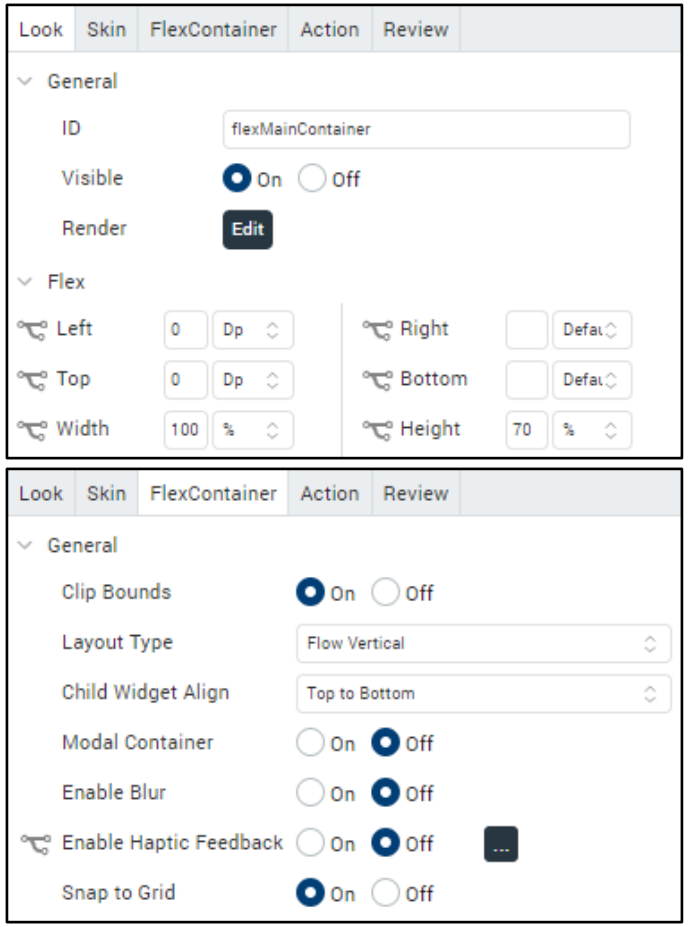


コンポーネントUIの作成（続き）

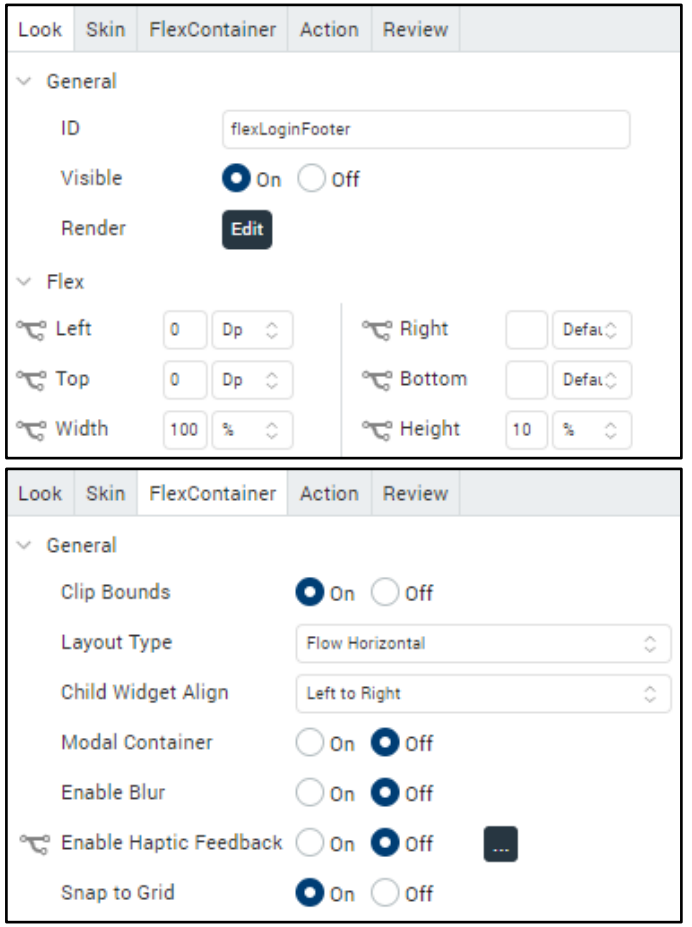
フレックスログインバナーを高さ20%、フリーフォームレイアウトタイプで作成します。



フレックスメインコンテナの高さを70%に、レイアウトタイプをフローバーティカルにします。

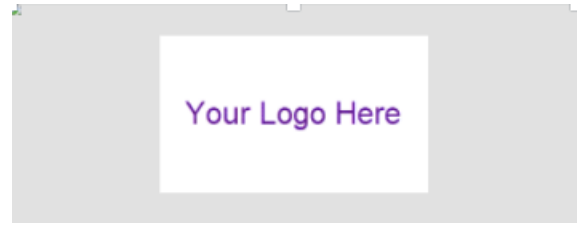


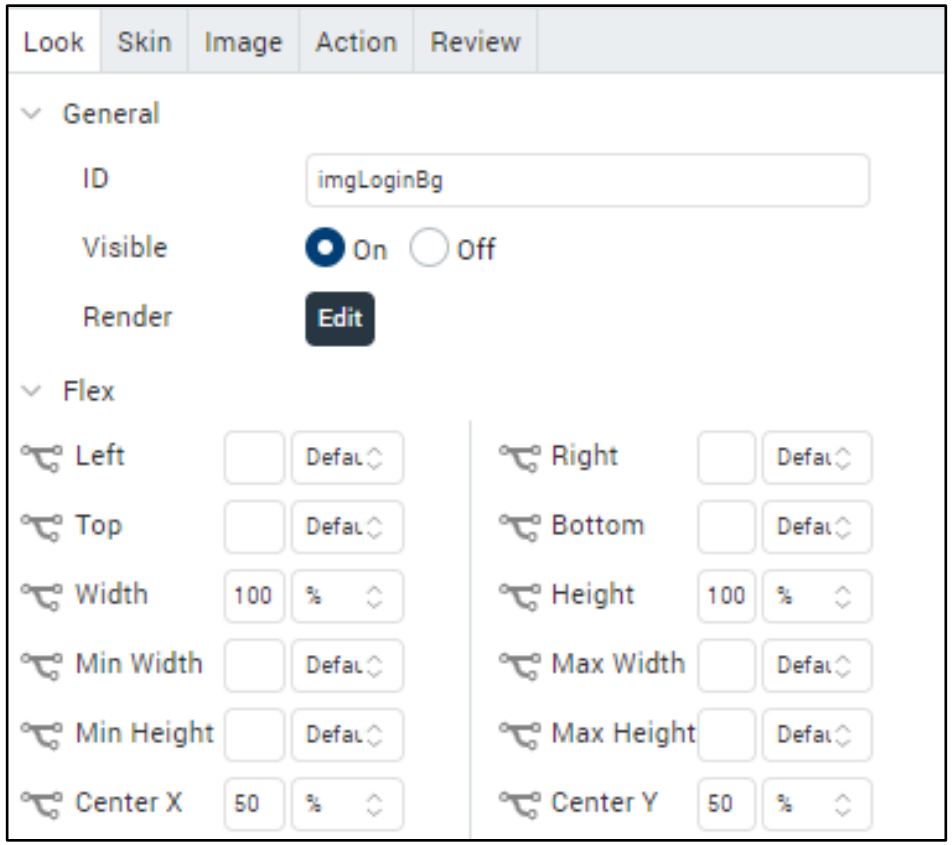
フレックスログインフッターの高さを10%に設定し、フロー横レイアウトタイプにします。

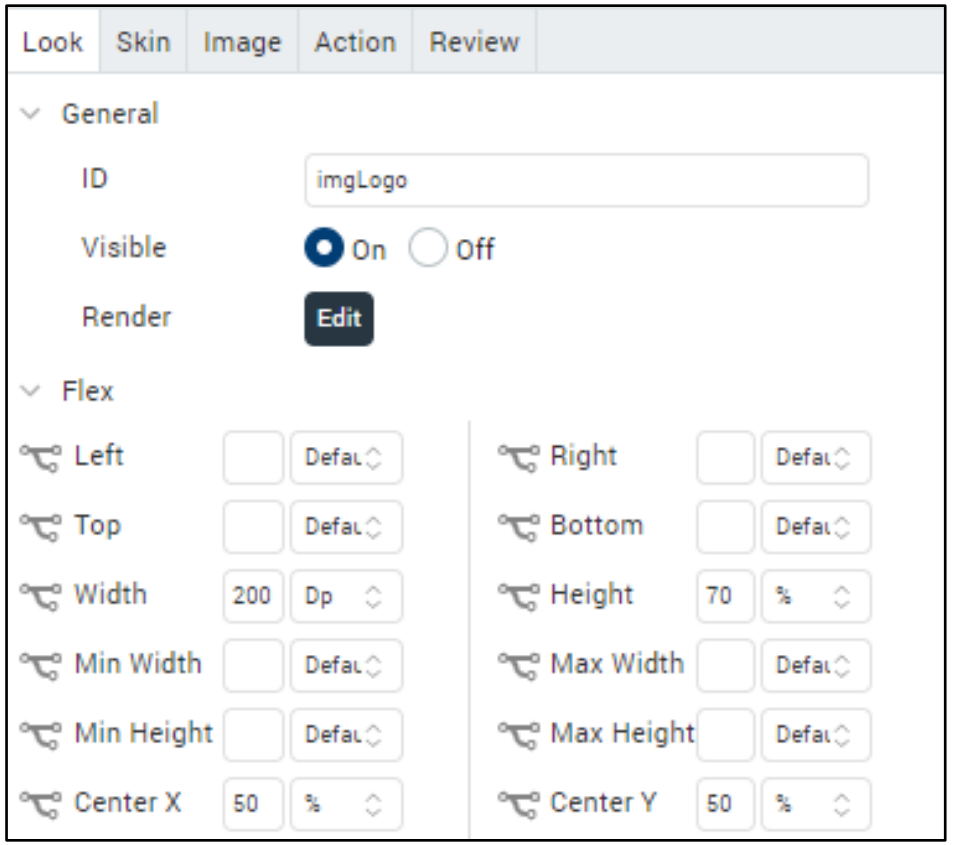


コンポーネントUIの作成（続き）

- flexLoginBannerを構築し、2つの画像を追加します。

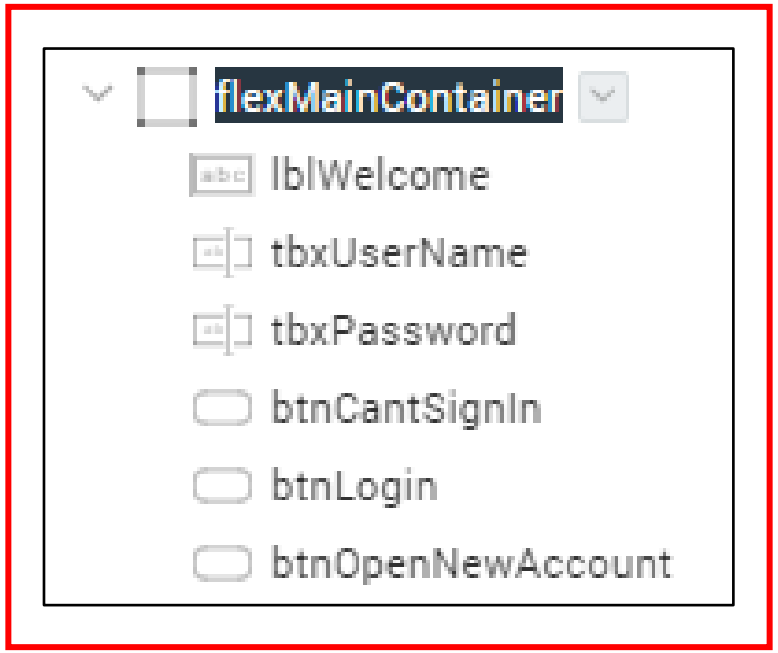


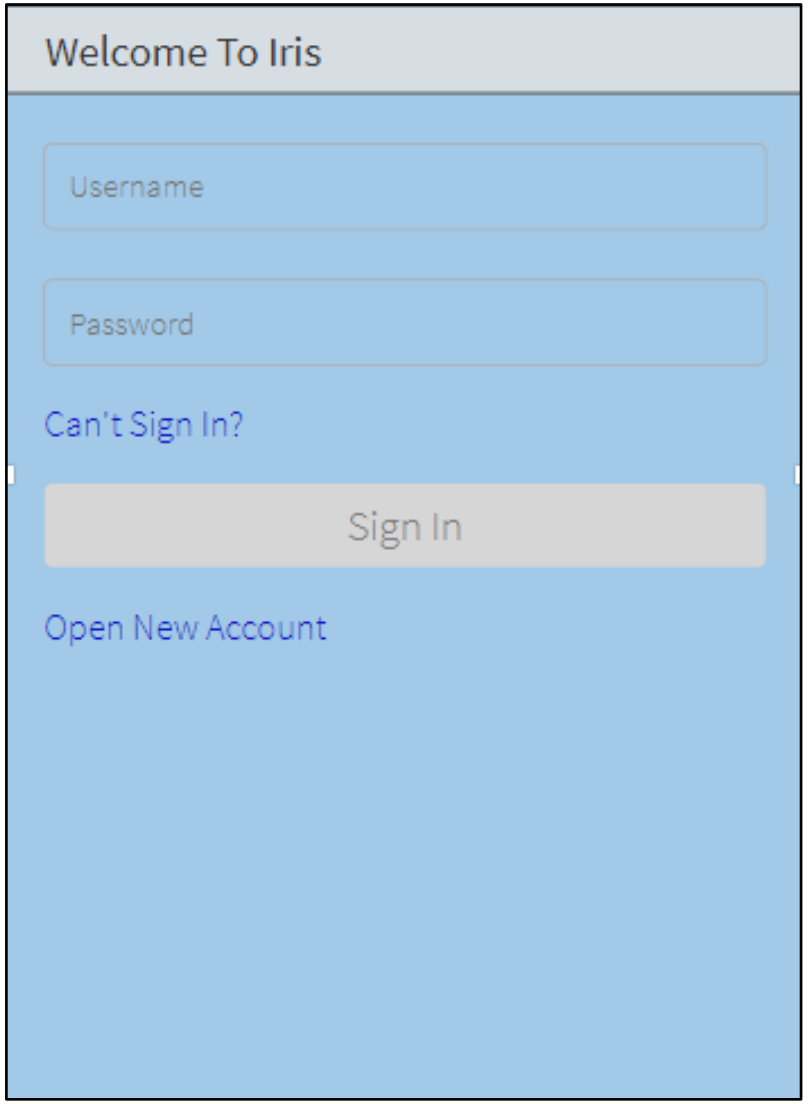




コンポーネントUIの作成（続き）

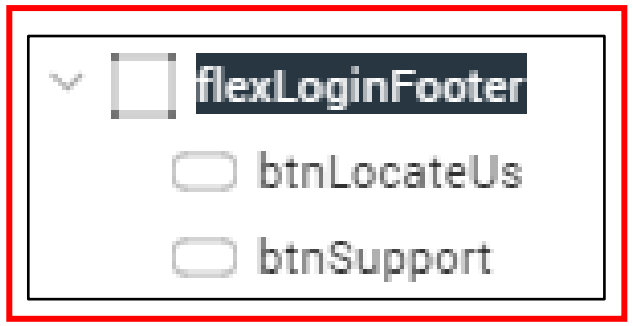
- flexMainContainerを構築して、図のウィジェットを追加します。

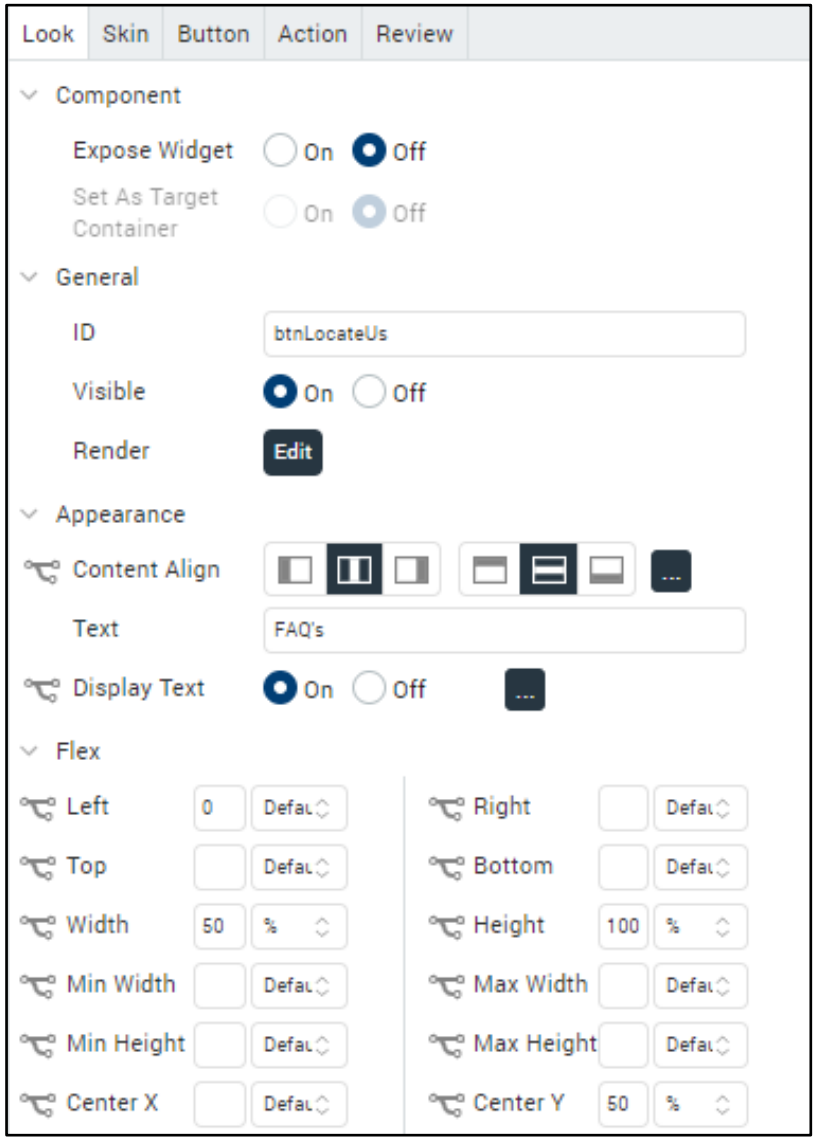




コンポーネントUIの作成（続き）

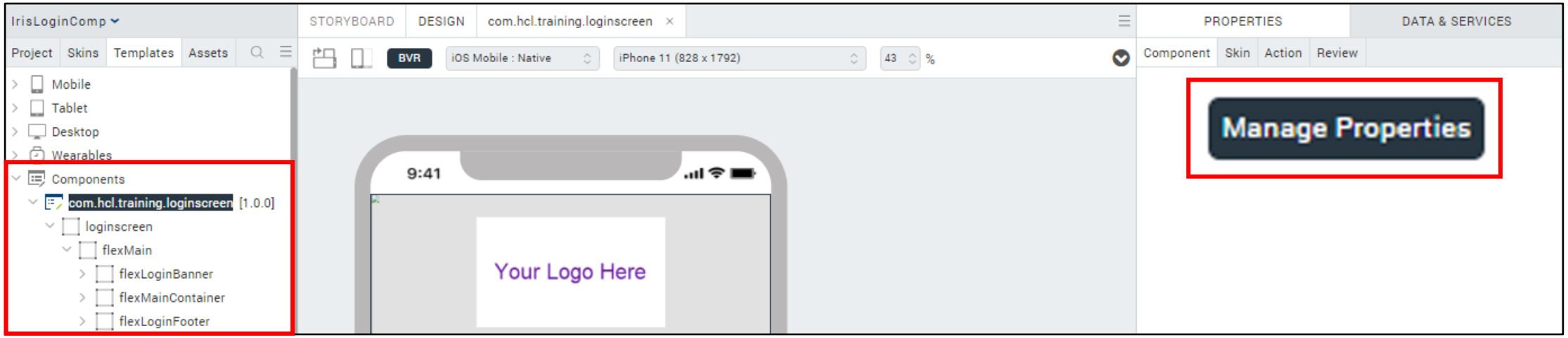
- flexLoginFooterを構築して、以下のウィジェットを追加します。
- このフレックスコンテナはFlow Horizontalに設定されているので、どちらのボタンも同じ外観のプロパティを持っています。





プロパティの公開

- コンポーネント(com.hcl.training.login)を選択し、コンポーネントタブを選択します。

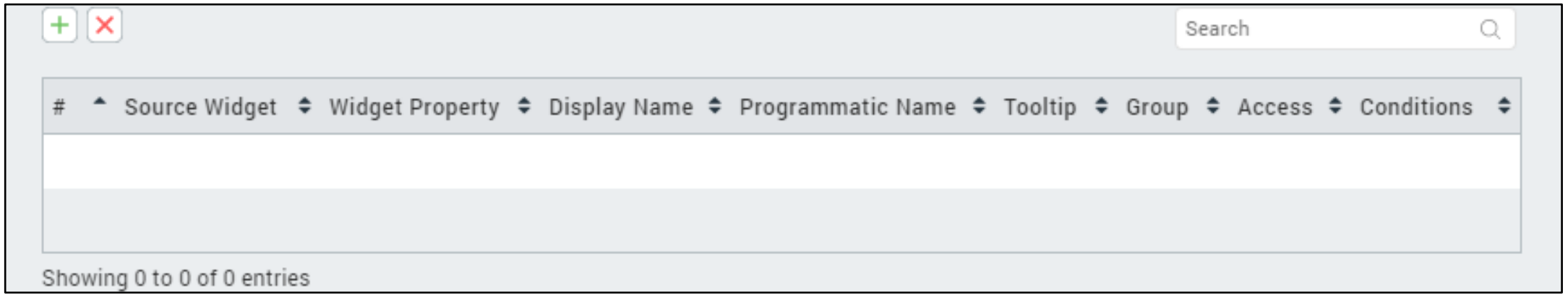


- 公開できるプロパティは2種類あります。

パススループロパティ	カスタムプロパティ
• パススループロパティは、プラットフォームの一部として利用可能なウィジェットプロパティです。 • パススループロパティの値が変更されたときに、プラットフォームがどのように反応するかが分かります。	• カスタムプロパティは、コンポーネント作成者によって定義されるプロパティです。 • コンポーネント作成者は、カスタムプロパティの値が変更されたときに何が起こるかについてのロジックを実装する必要があります。

パススループロパティの公開

- パススループロパティは、コンポーネントの使用者に公開したいプラットフォームの一部として利用可能なウィジェットプロパティです。



Source Widget	公開したいプロパティを持つウィジェットを選択します。
Widget Property	公開するウィジェットのプロパティを選択します。
Display Name	このプロパティを公開するための名前を定義します。
Programmatic Name	使用者がコードからこのプロパティの値にアクセスし、変更できるようにするための、プロパティのプログラム名を定義します。
Group	プロパティのグループを作成できます。Flex', 'General', 'Background', 'Border'などのグループと同じです。
Access	プロパティを有効または無効にできます。

パススループロパティの公開

- 以下のプロパティを公開します。
- グループ ドロップダウンを使用して、グループ化を作成および管理します。

1	flexMain	skin	mainFlexSkin	mainFlexSkin		General	Enable	Conditions
2	flexMainContainer	skin	loginCardSkin	loginCardSkin		General	Enable	Conditions
3	imgLoginBg	src	logoBackgroundImage	logoBackgroundImage		Logo Properties	Enable	Conditions
4	imgLogo	src	logoImageSource	logoImageSource		Logo Properties	Enable	Conditions
5	lblWelcome	skin	welcomeLabelSkin	welcomeLabelSkin		Welcome Message	Enable	Conditions
6	lblWelcome	text	welcomeLabelText	welcomeLabelText		Welcome Message	Enable	Conditions
7	tbxUserName	skin	userNameTextBoxSkin	userNameTextBoxSkin		Username Properties	Enable	Conditions
8	tbxUserName	placeholder	userNameTextBoxPlaceholde	userNameTextBoxPlaceholde		Username Properties	Enable	Conditions
9	tbxUserName	text	userNameTextBoxText	userNameTextBoxText		Username Properties	Enable	Conditions
10	tbxUserName	keyBoardStyle	userNameKeyboardStyle	userNameKeyBoardStyle		Username Properties	Enable	Conditions
11	tbxPassword	skin	passwordTextBoxSkin	passwordTextBoxSkin		Password Properties	Enable	Conditions
12	tbxPassword	text	passwordTextBoxText	passwordTextBoxText		Password Properties	Enable	Conditions
13	tbxPassword	placeholder	passwordTextBoxPlaceholder	passwordTextBoxPlaceholder		Password Properties	Enable	Conditions
14	tbxPassword	keyBoardStyle	passwordTextBoxKeyboardSt	passwordTextBoxKeyBoardSt		Password Properties	Enable	Conditions
15	tbxPassword	secureTextEntry	passwordTextBoxSecureText	passwordTextBoxSecureTextE		Password Properties	Enable	Conditions
16	btnCantSignIn	skin	cantSigninButtonSkin	cantSigninButtonSkin		Can't Signin Properties	Enable	Conditions
17	btnCantSignIn	focusSkin	cantSigninButtonFocusSkin	cantSigninButtonFocusSkin		Can't Signin Properties	Enable	Conditions
18	btnCantSignIn	text	cantSigninButtonText	cantSigninButtonText		Can't Signin Properties	Enable	Conditions

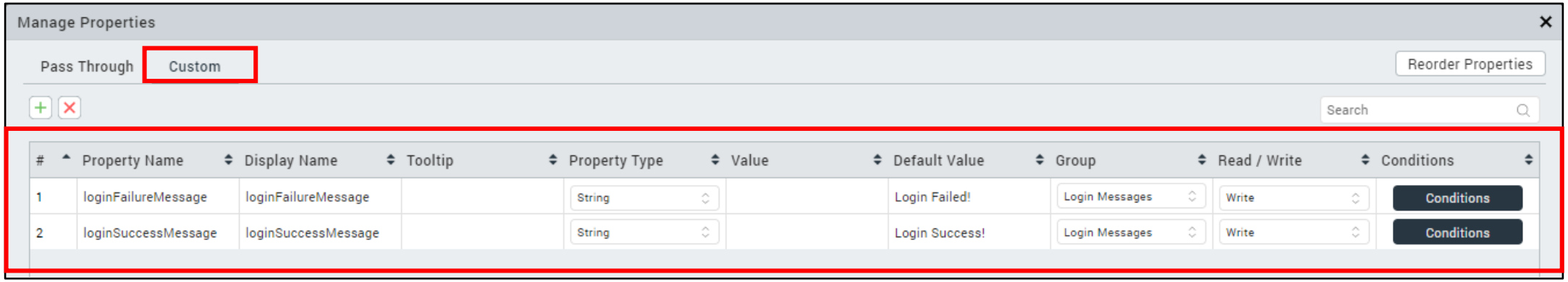
パススループロパティの公開

- 以下は、公開する残りのプロパティです。

19	btnLogin	skin	loginButtonSkin	Skin		Login Button Properties	Enable	Conditions
20	btnLogin	focusSkin	loginButtonFocusSkin	loginButtonFocusSkin		Login Button Properties	Enable	Conditions
21	btnLogin	text	loginButtonText	loginButtonText		Login Button Properties	Enable	Conditions
22	btnOpenNewAccount	skin	openNewButtonSkin	openNewButtonSkin		Open New Account Proper	Enable	Conditions
23	btnOpenNewAccount	focusSkin	openNewButtonFocusSkin	openNewButtonFocusSkin		Open New Account Proper	Enable	Conditions
24	btnOpenNewAccount	text	openNewButtonText	openNewButtontext		Open New Account Proper	Enable	Conditions
25	btnLocateUs	skin	footerLocateUsNormal	footerLocateUsNormal		Footer Options Properties	Enable	Conditions
26	btnLocateUs	focusSkin	footerLocateUsFocusSkin	footerLocateUsFocusSkin		Footer Options Properties	Enable	Conditions
27	btnLocateUs	text	footerLocateUsText	footerLocateUsText		Footer Options Properties	Enable	Conditions
28	btnSupport	skin	footerSupportSkin	footerSupportSkin		Footer Options Properties	Enable	Conditions
29	btnSupport	focusSkin	footerSupportFocusSkin	footerSupportFocusSkin		Footer Options Properties	Enable	Conditions
30	btnSupport	text	footerSupportText	footerSupportText		Footer Options Properties	Enable	Conditions
31	flexLoginFooter	skin	footerSkin	footerSkin		Footer Options Properties	Enable	Conditions

カスタムプロパティの公開

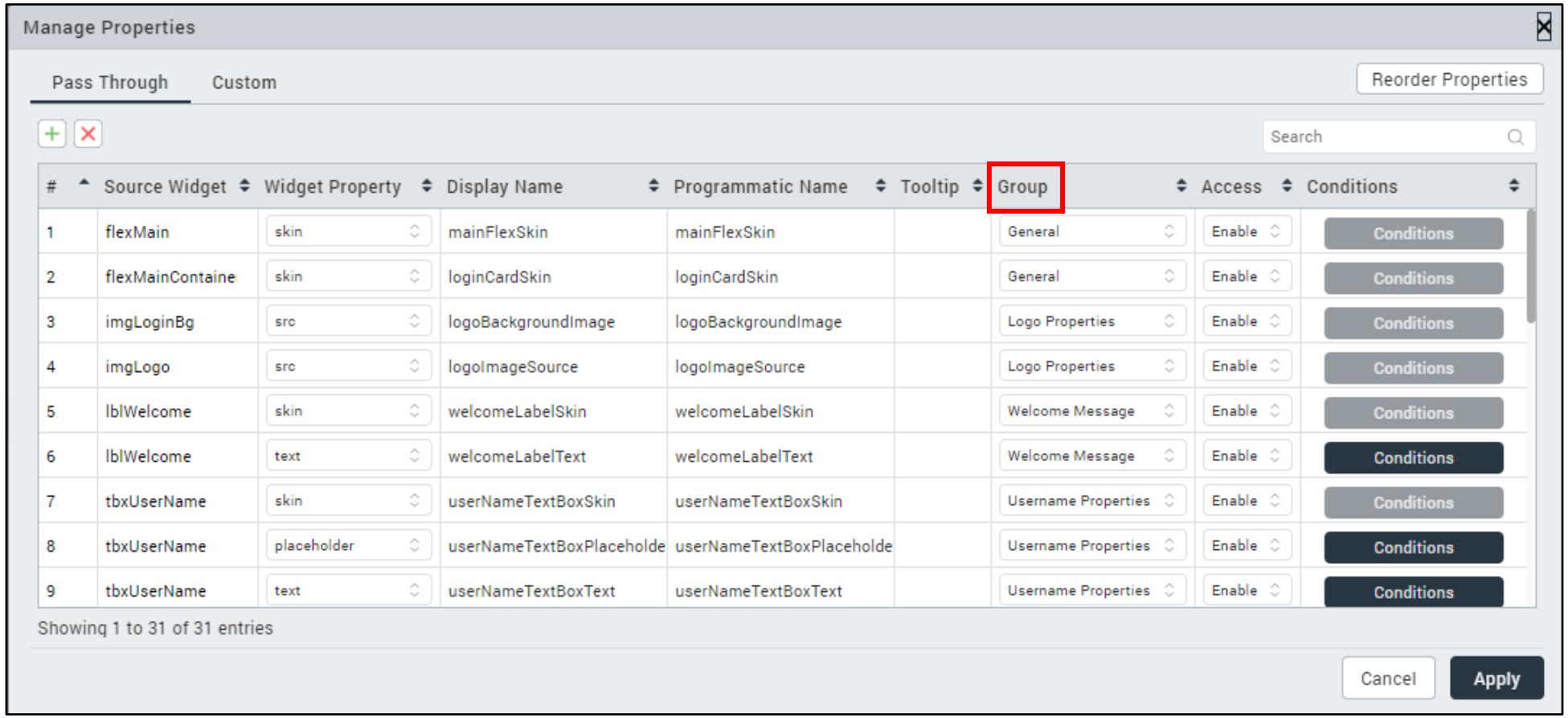
- カスタムプロパティは、どのウィジェットにも関連付けられていません。これらは、コンポーネントの作成者によって定義されます。
- このようなカスタムプロパティの値が変更されたときに、コンポーネントがどのように動作するかは、コンポーネント作成者が定義する必要があります。



Property Name	カスタムプロパティの名前を定義します。
Display Name	カスタムプロパティがIrisのプロパティパネルにどのように表示されるかを定義します。
Property Type	カスタムプロパティに設定できる値の型を定義します。例えば、String、Integer、Booleanなど。
Default Value	カスタムプロパティのデフォルト値を定義します。
Group	カスタムプロパティのグループを作成できます。
Read / Write	プロパティが読み取り専用か読み取り/書き込み可能かを指定します。

パススループロパティの公開（続き）

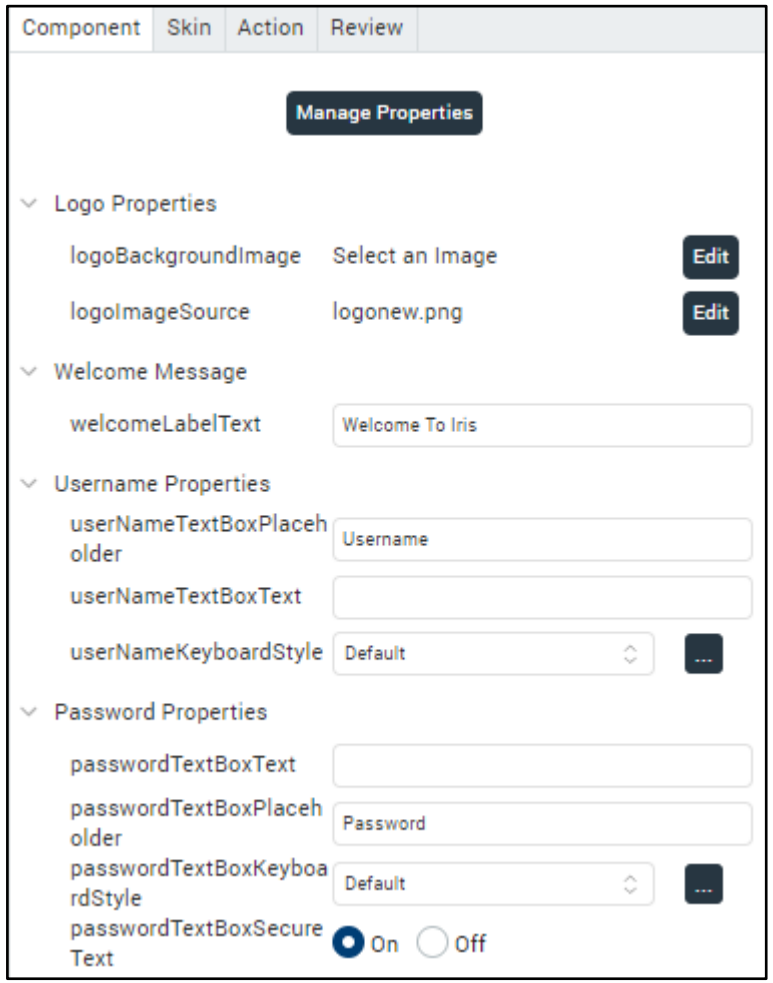
- 完成すると、前のスライドのステップに従って作成したプロパティになります。

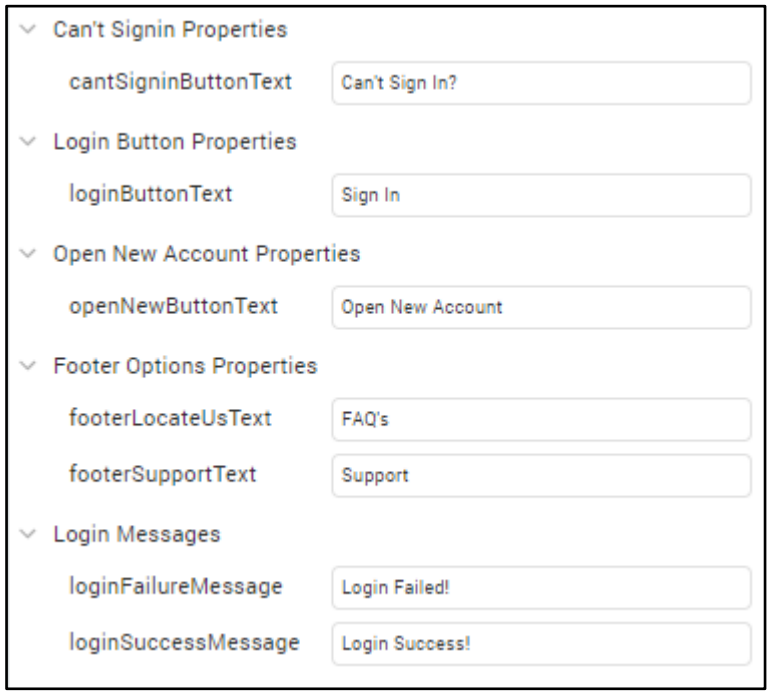


- ユーザーを支援するためのグループ分けに注目

パススループロパティの公開（続き）

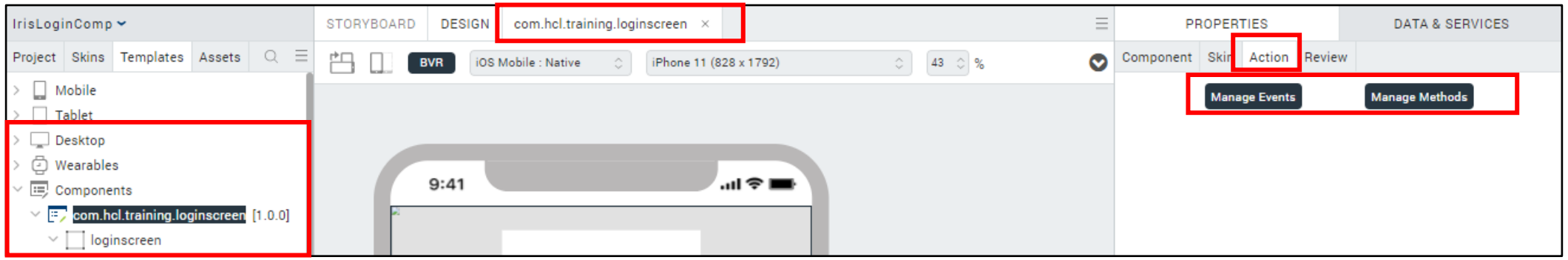
- 完成すると、前のスライドのステップに従って作成したプロパティになります。





イベントとメソッドの公開

- コンポーネントのイベントとメソッドを公開します。

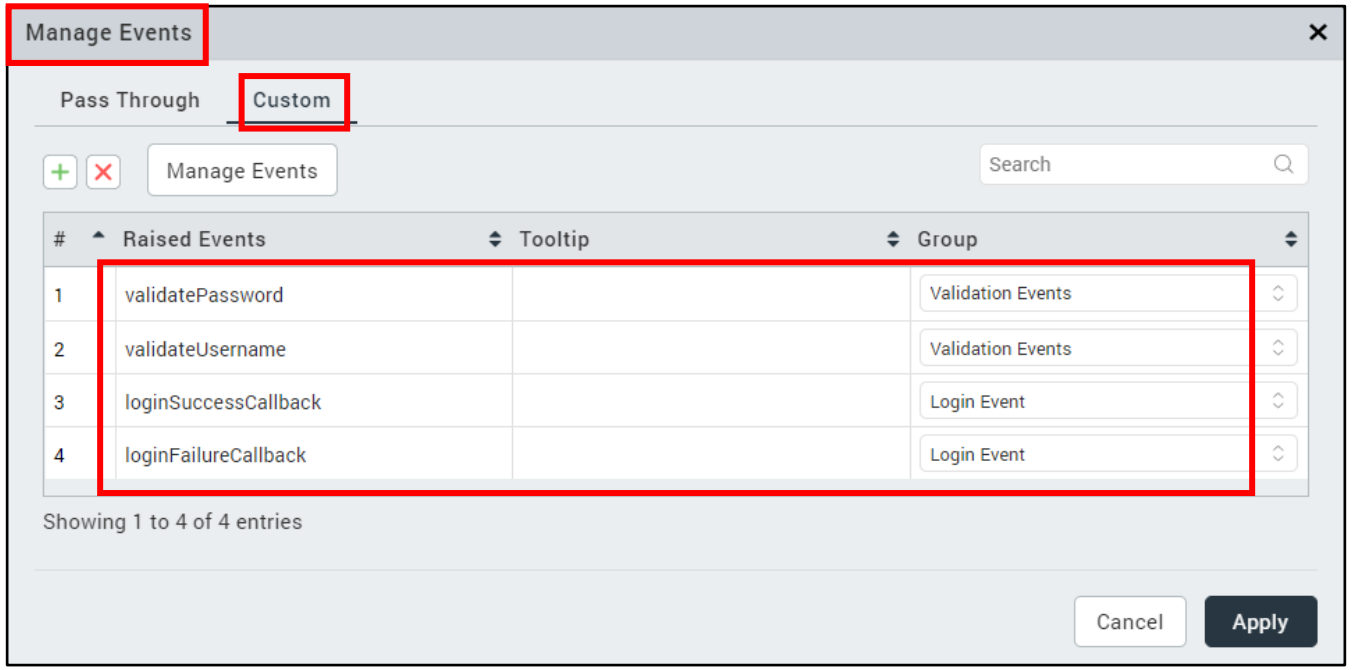


イベント	メソッド

• イベントは、ユーザーによってコンポーネント上でアクションが実行されたときにトリガーされます。
  • 例えば、ボタンでは 'onClick' イベントを、リストボックスでは 'onSelection' イベントを実行するなどです。
	• メソッドを使用すると、コンポーネントの使用者はコンポーネント内のデータにアクセス(読み取りまたは書き込み)できます。
	  • 例えば、'setVisibility' メソッドは、任意のウィジェットの可視性を変更できます、等々。

カスタムイベントの公開

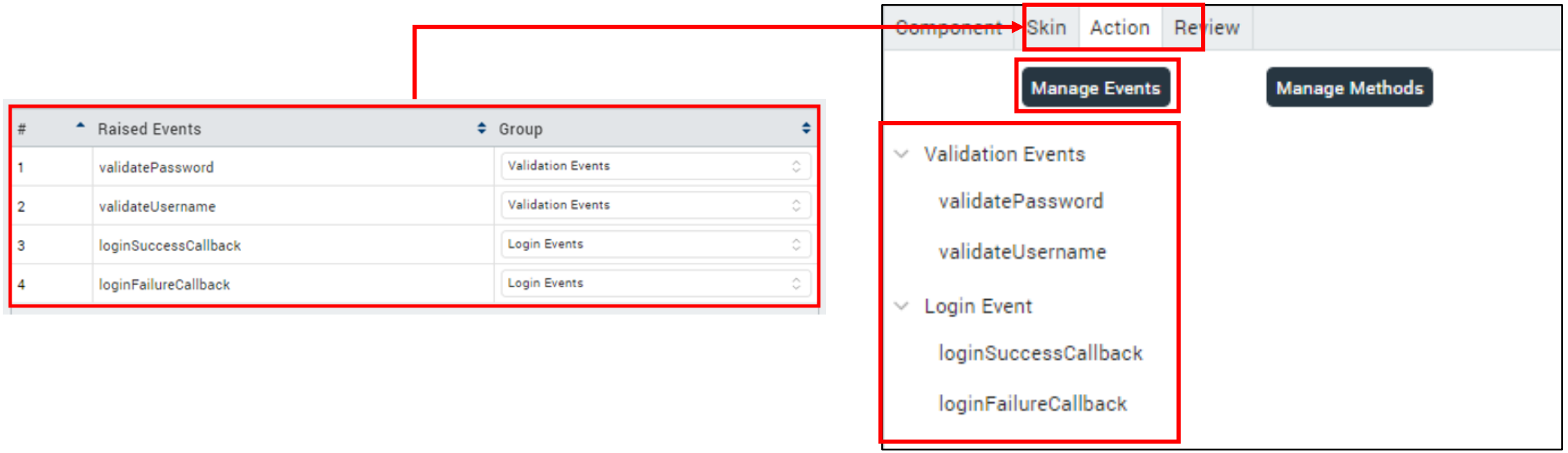
- カスタムイベントは、コンポーネント作成者によって定義されます。
- コンポーネント作成者は、これらのイベントをいつ発生させるかも定義する必要があります。
- カスタム・イベントを定義する際には、以下の属性を定義する必要があります。



Raised Events	カスタムイベントの名称を定義します。
Group	カスタムイベントのグループを作成できます。

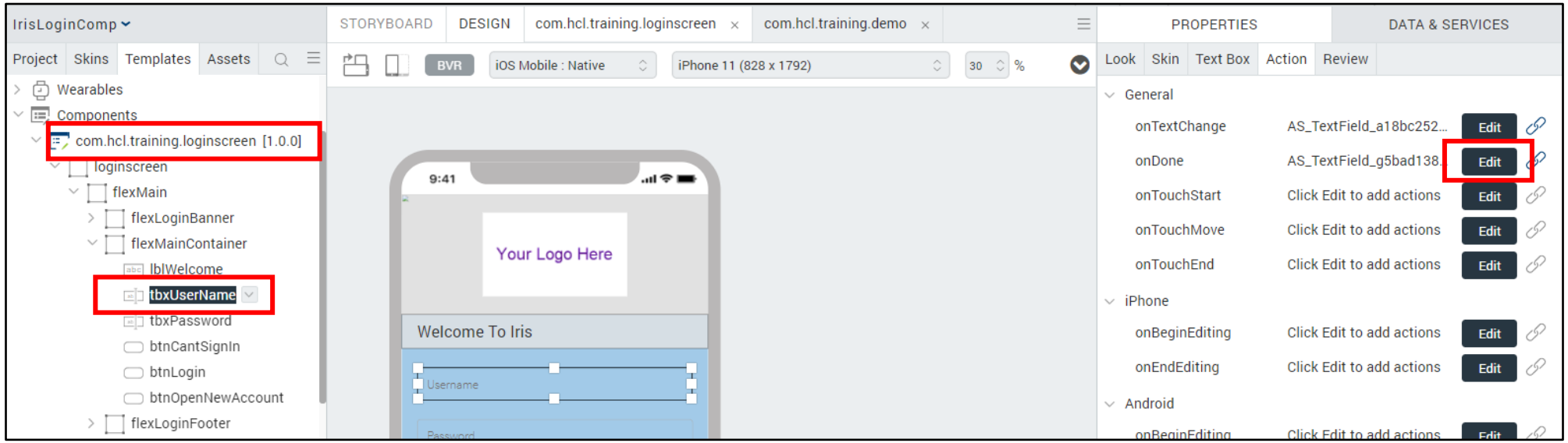
カスタムイベントの公開（つづき）

- これは、前のスライドで示された定義に対して、カスタムイベントが作成される方法です。



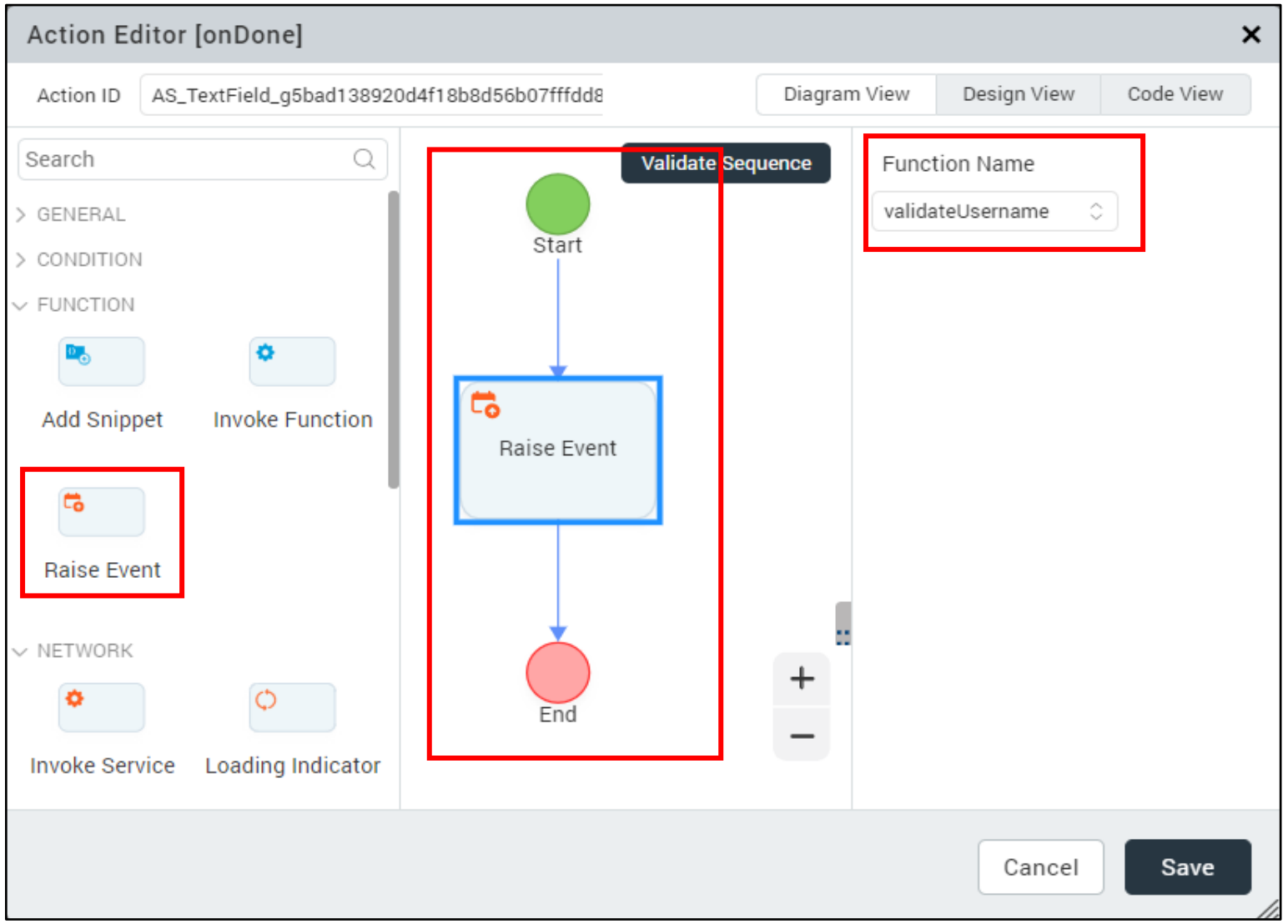
カスタムイベントの公開（つづき）

- カスタムイベント 'validatePassword' と 'validateUsername' は、ユーザー名テキストボックスとパスワードテキストボックスの 'onDone' イベントで発生します。
- 'validateUsername' と 'validatePassword' の実際のアクションシーケンスは、コンポーネントのコンシューマによって定義されます。



カスタムイベントの公開（つづき）

- カスタム・イベントの発生（Raising Event）
  - コンポーネントのコンシューマがカスタムイベントに対して定義したアクションシーケンスは、図のような形で発生します。



カスタムメソッドの公開

- カスタムメソッドは、コンポーネントの作成者によって定義されます。
- コンポーネント作成者は、これらのカスタムメソッドのロジックも定義する必要があります。
- カスタムメソッドを定義する際には、以下の属性を定義する必要があります。

Method	カスタムメソッドの名前を定義する
Group	カスタムメソッド用のグループを作成できます。

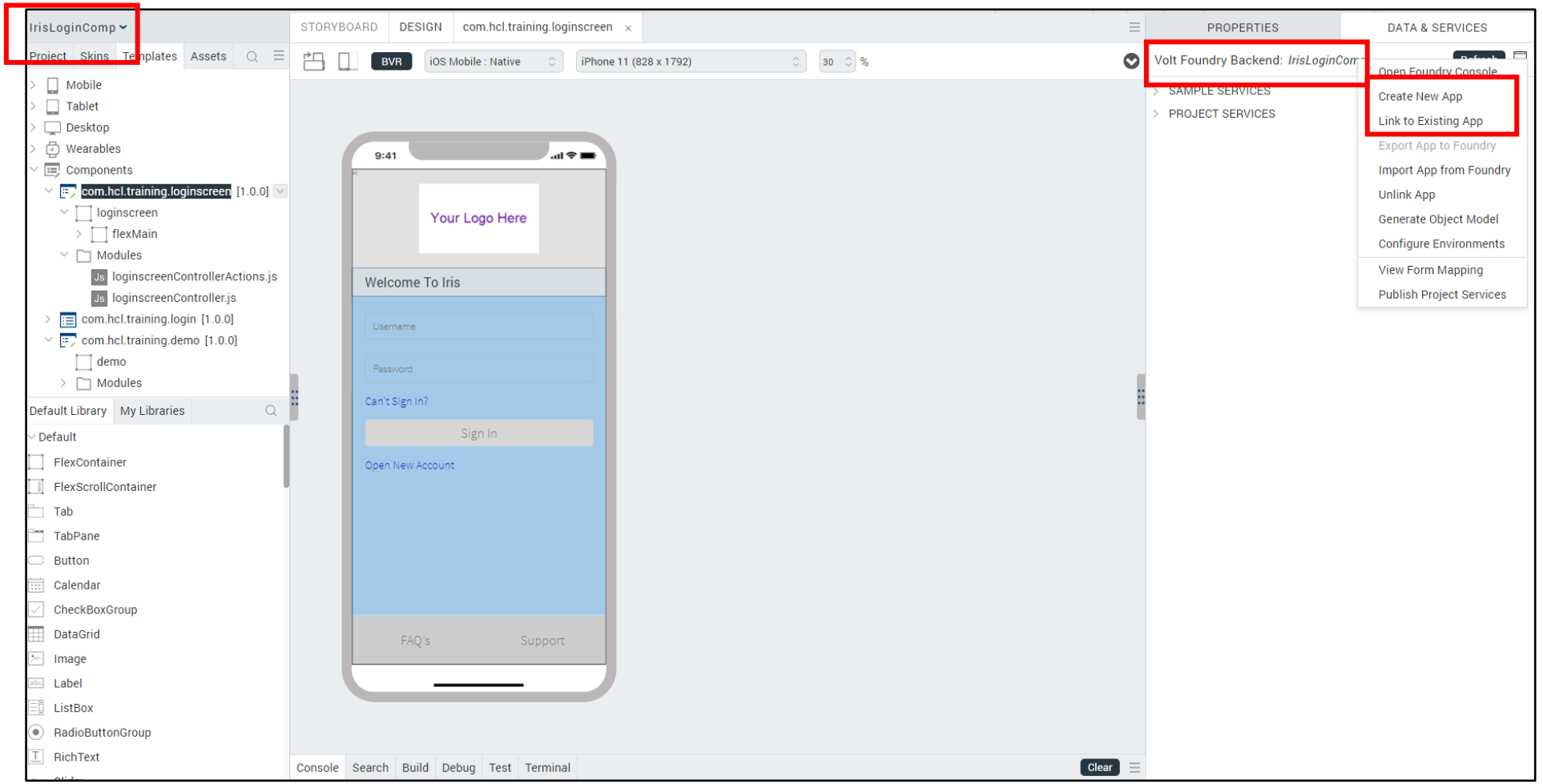
カスタムメソッドの公開

- 以下のコントローラロジックを確認します。
  - カスタムプロパティのinitGetterSettersを更新します。
  - カスタムメソッド用のロジックを追加します。

```
//Logic for getters/setters of custom properties
initGettersSetters: function() {
  defineGetter(this, 'loginFailureMessage', () => {
    return this._loginFailureMessage;
  });
  defineSetter(this, 'loginFailureMessage', value => {
    this._loginFailureMessage = value;
  });
  defineGetter(this, 'loginSuccessMessage', () => {
    return this._loginSuccessMessage;
  });
  defineSetter(this, 'loginSuccessMessage', value => {
    this._loginSuccessMessage = value;
  });
},
getUsername: function() {
  return this.view.tbxUserName.text;
}
```

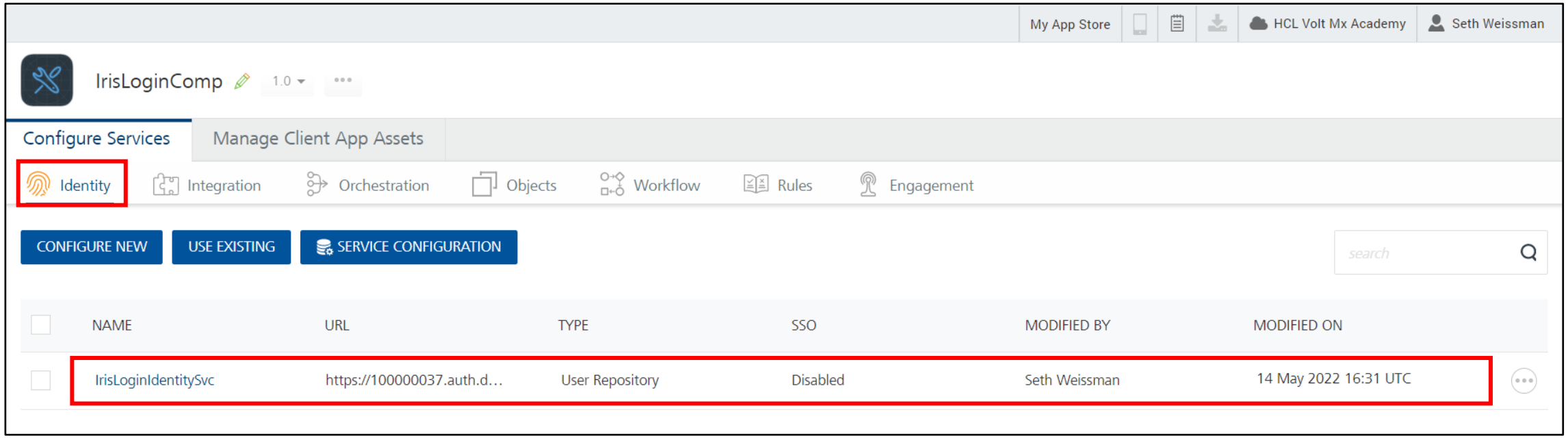
コンポーネントをFoundryにリンクする

- Iris プロジェクトを Foundry プロジェクトにリンクするか、Identity Service を持つ新しいプロジェクトを作成します。

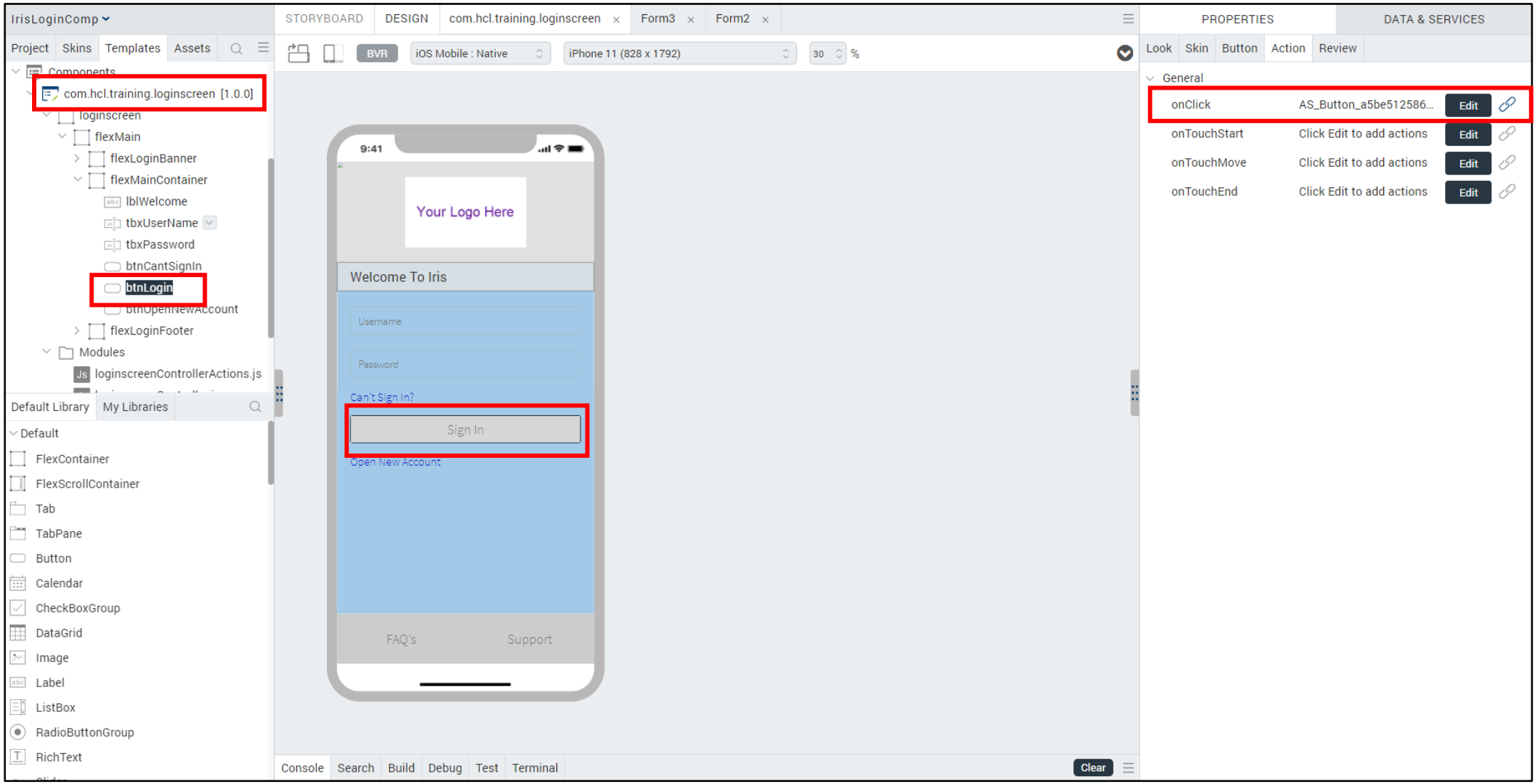


コンポーネントをFoundryにリンクする（続き）

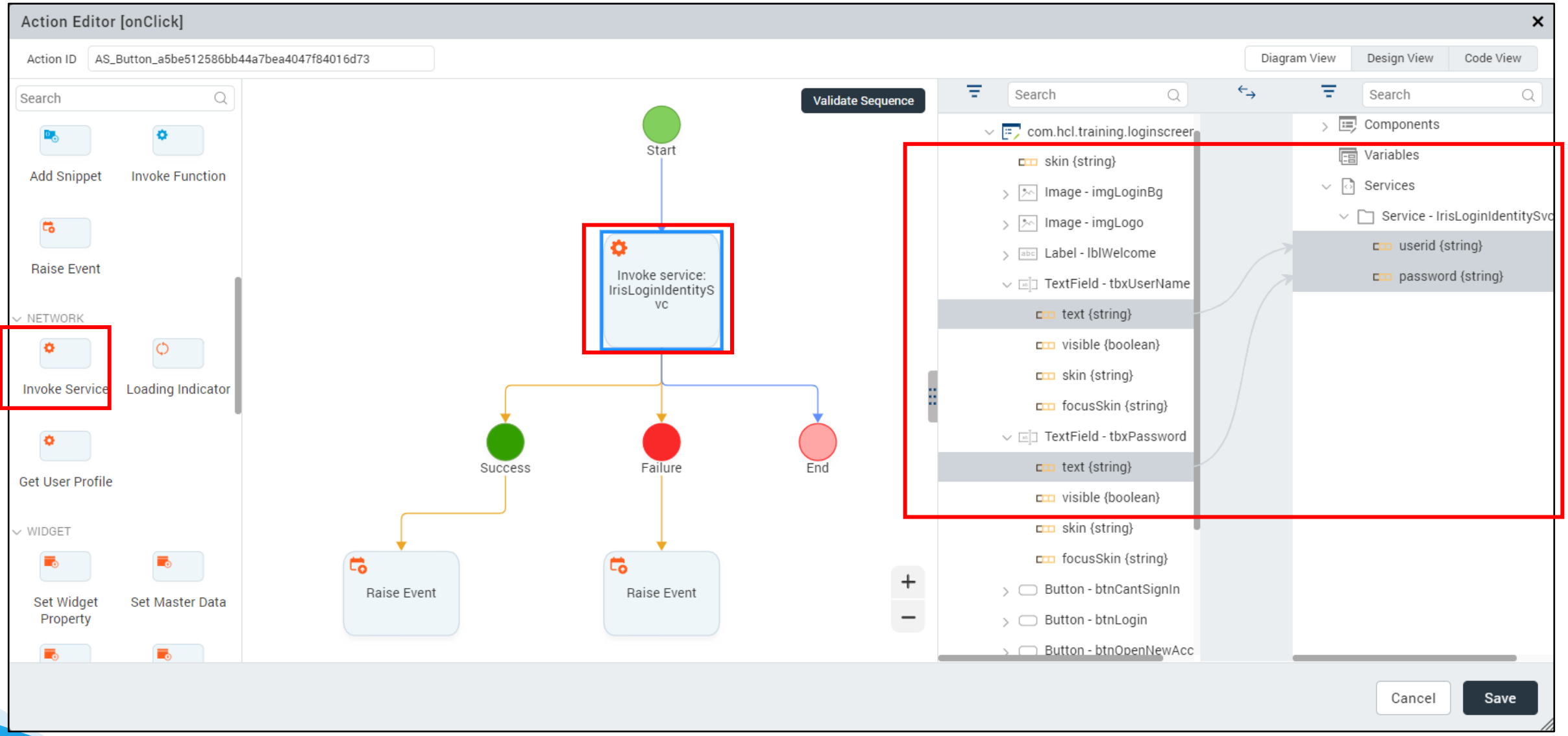
- Identity サービスのタイプは User Repository です。



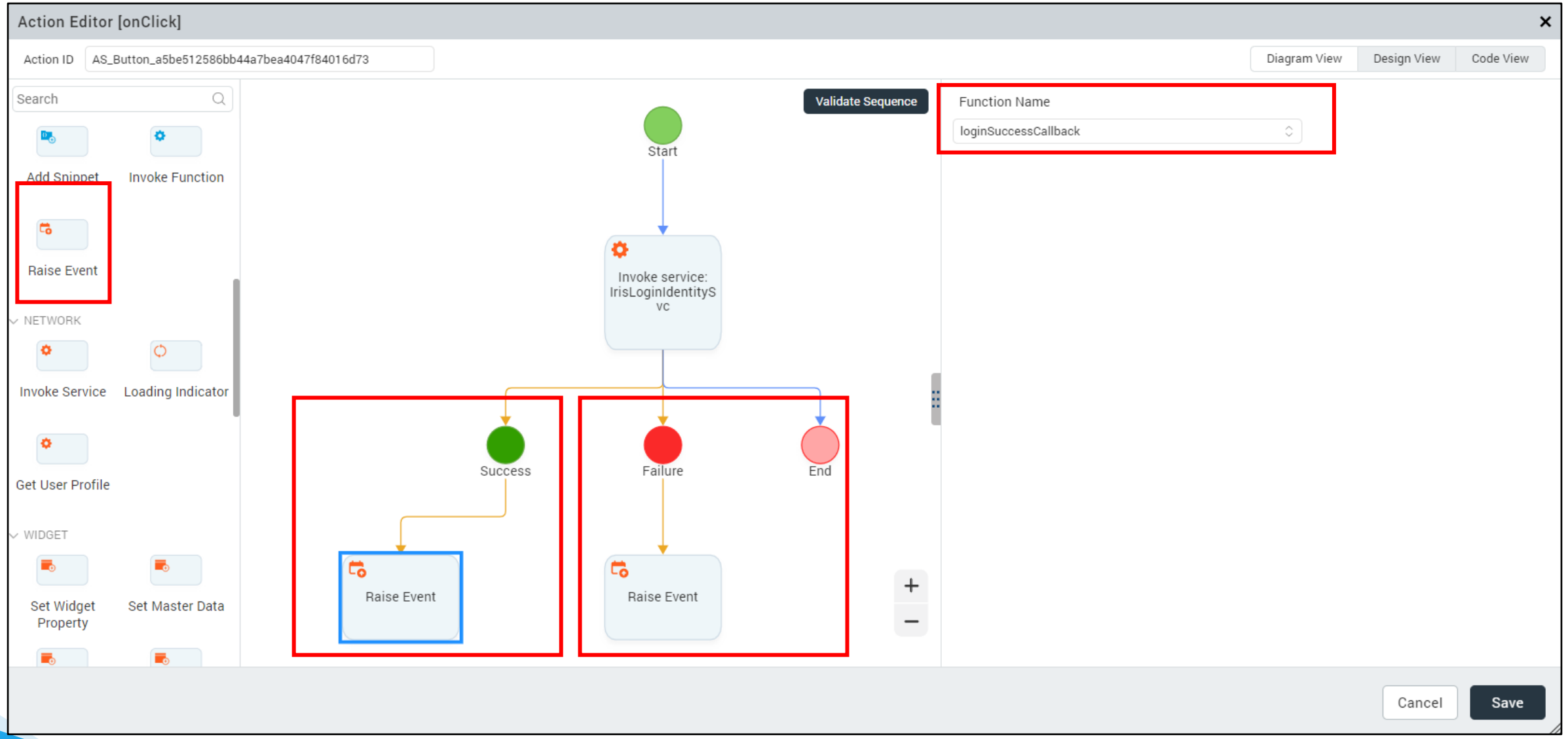
コンポーネントをFoundryにリンクする（続き）



コンポーネントをFoundryにリンクする（続き）



コンポーネントをFoundryにリンクする（続き）



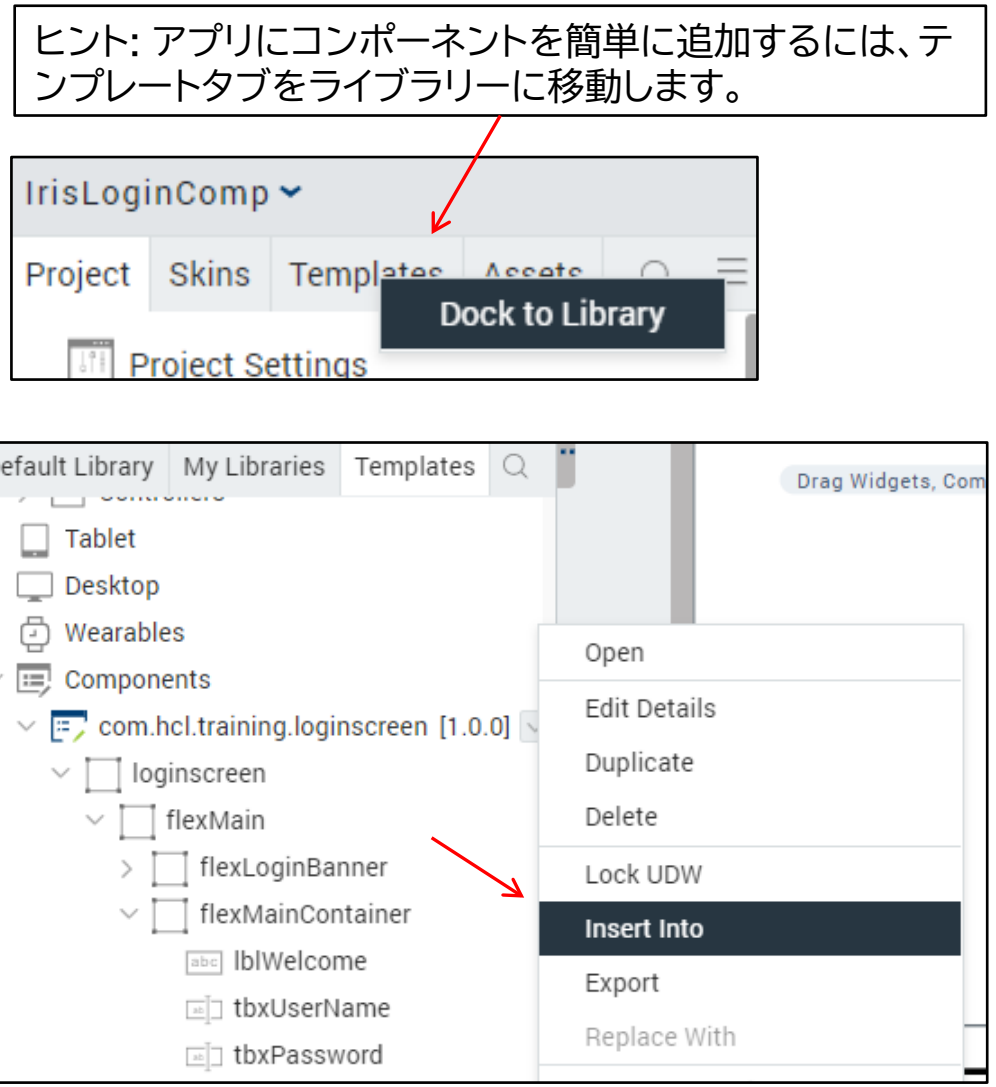
フォームでコンポーネントを使用する

- フォームを開く
- Template タブを開く
- ヒント:「テンプレート」タブを「ライブラリー」エリアに移動する

- コンポーネントを右クリック（またはコンテキストメニューから）-“insert into” オプションを選択。

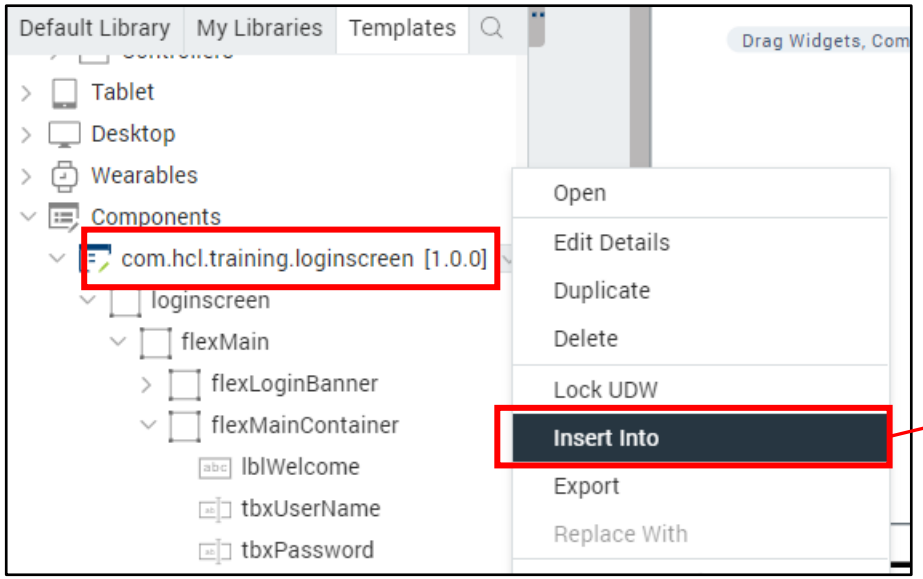
- コンポーネントは、フォームに追加されます。

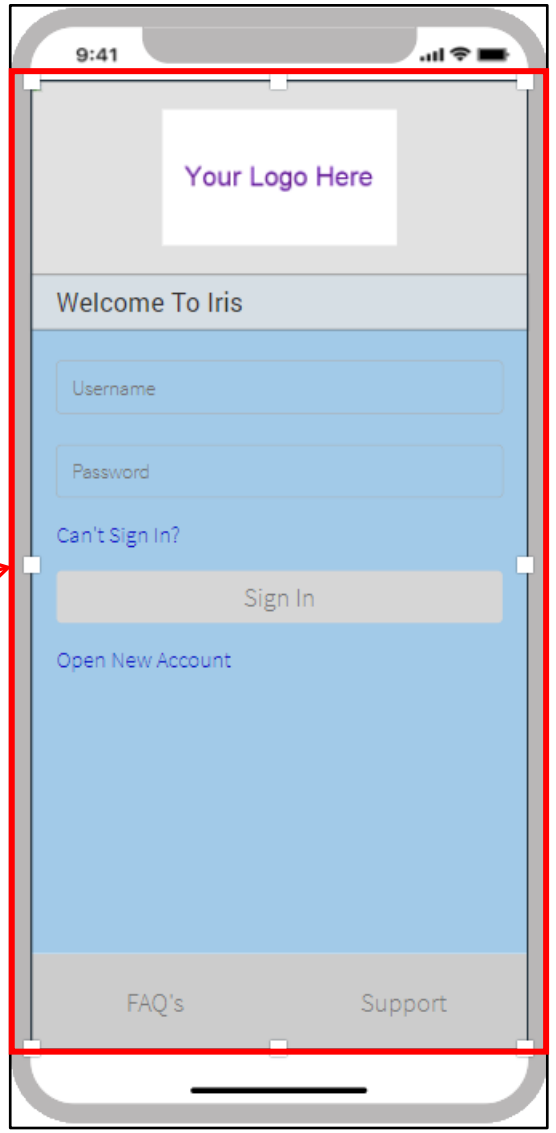
- または、フォームにドラッグすることもできます。



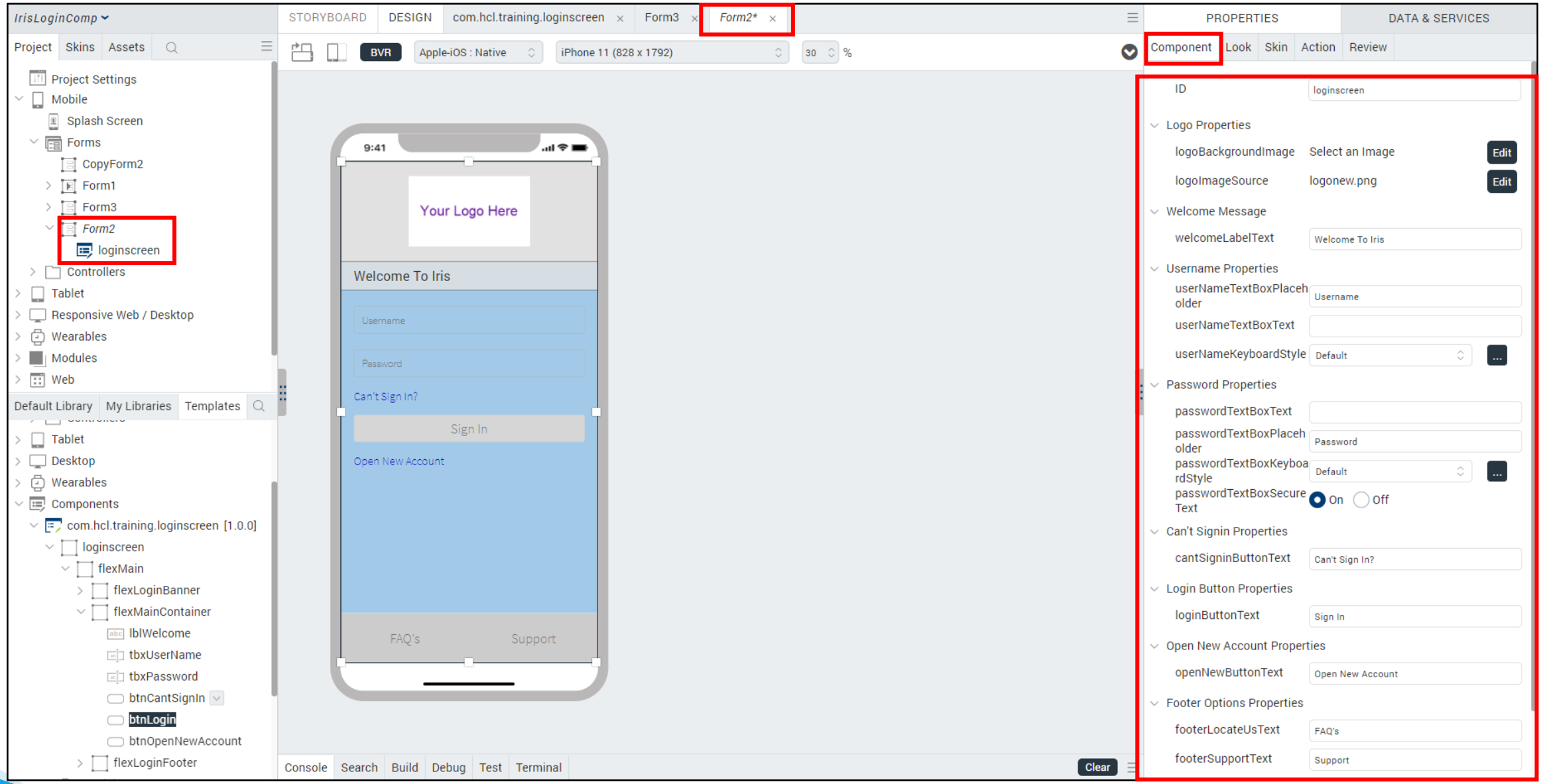
フォームでコンポーネントを使用する（続き）

- コンポーネントを追加した後のログイン画面は以下の通りです。

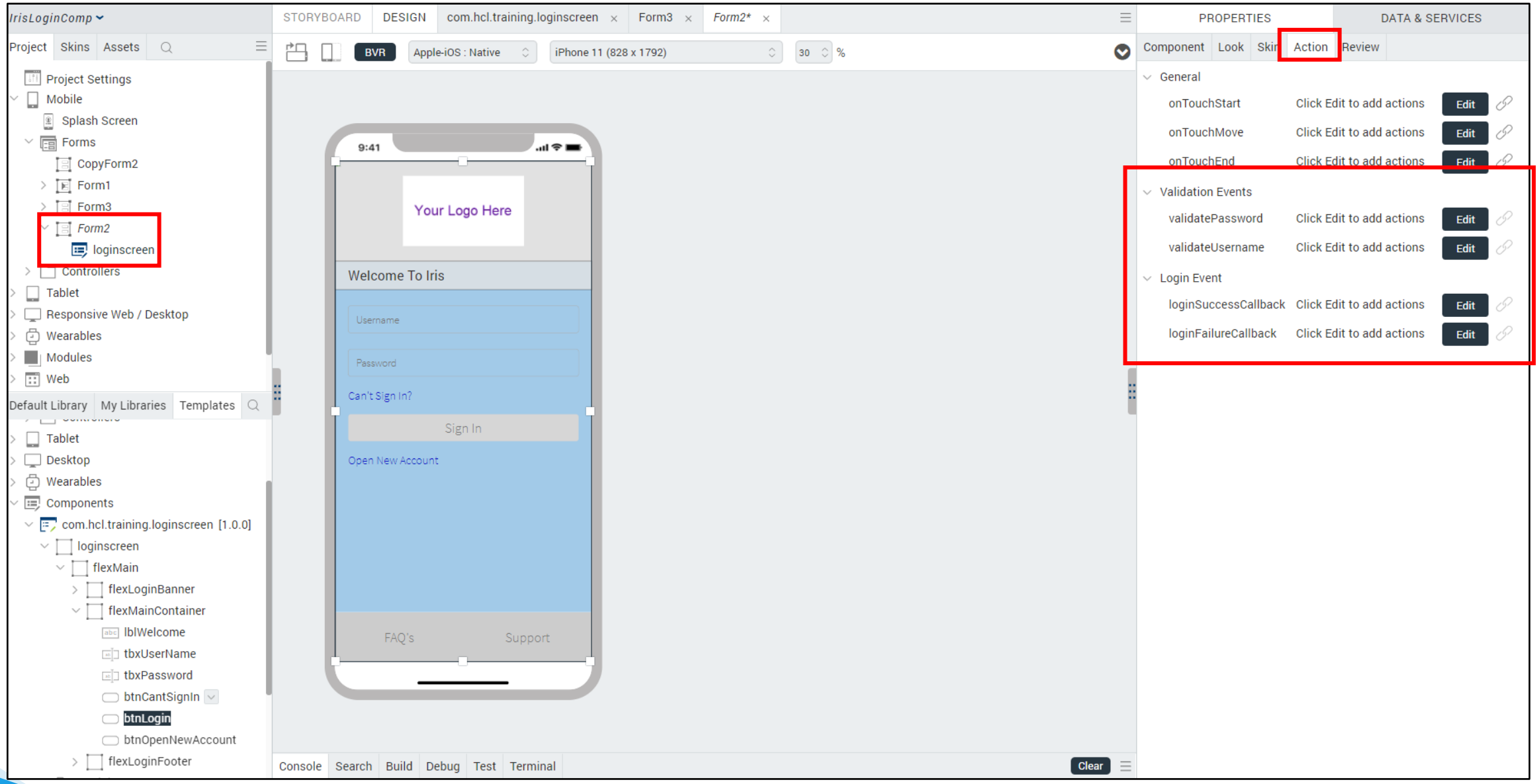




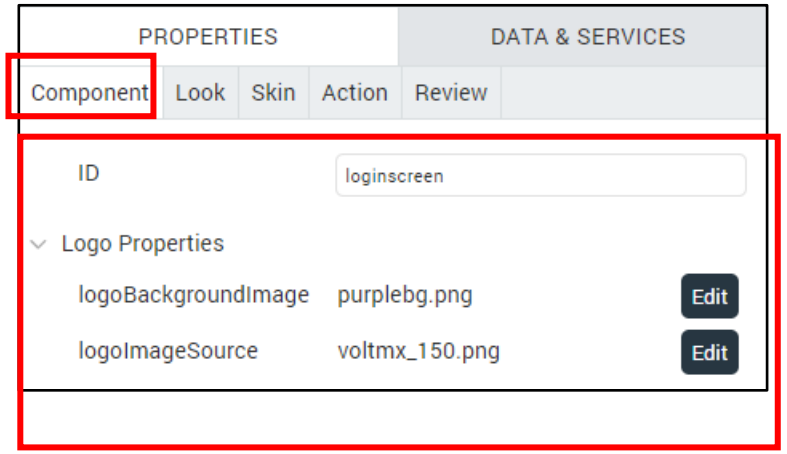
コンポーネントのカスタマイズ

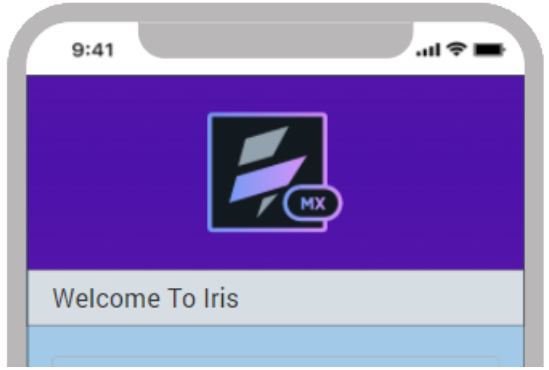


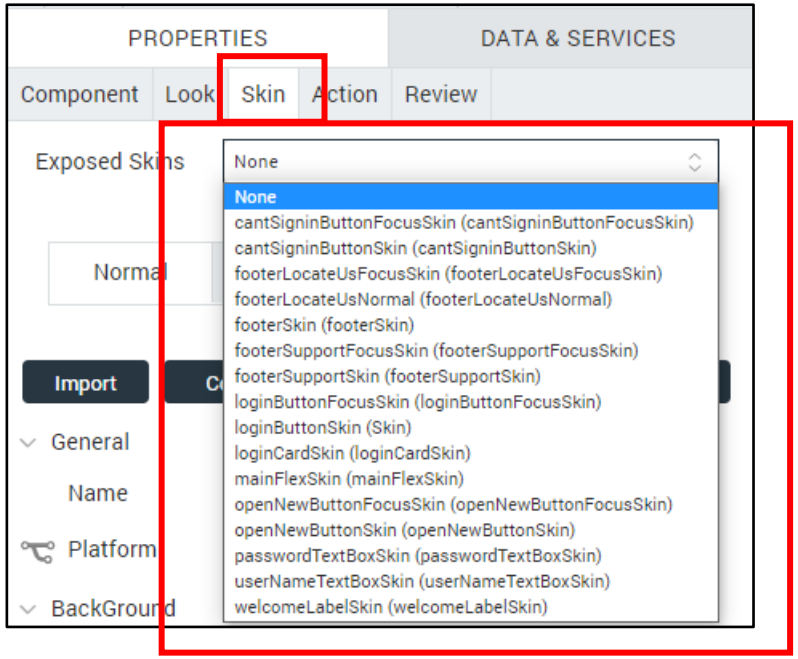
コンポーネントのカスタマイズ（続き）



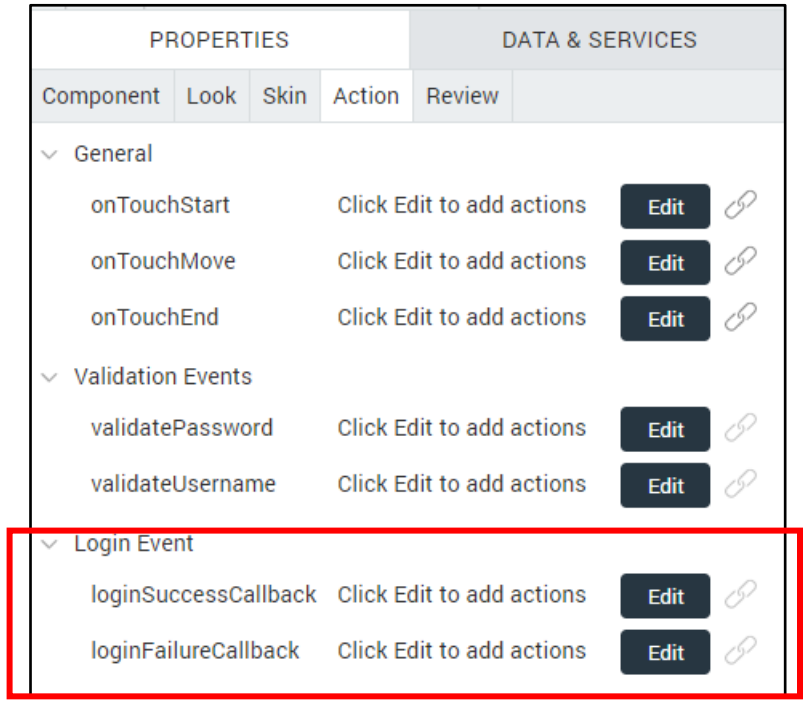
コンポーネントのカスタマイズ（続き）

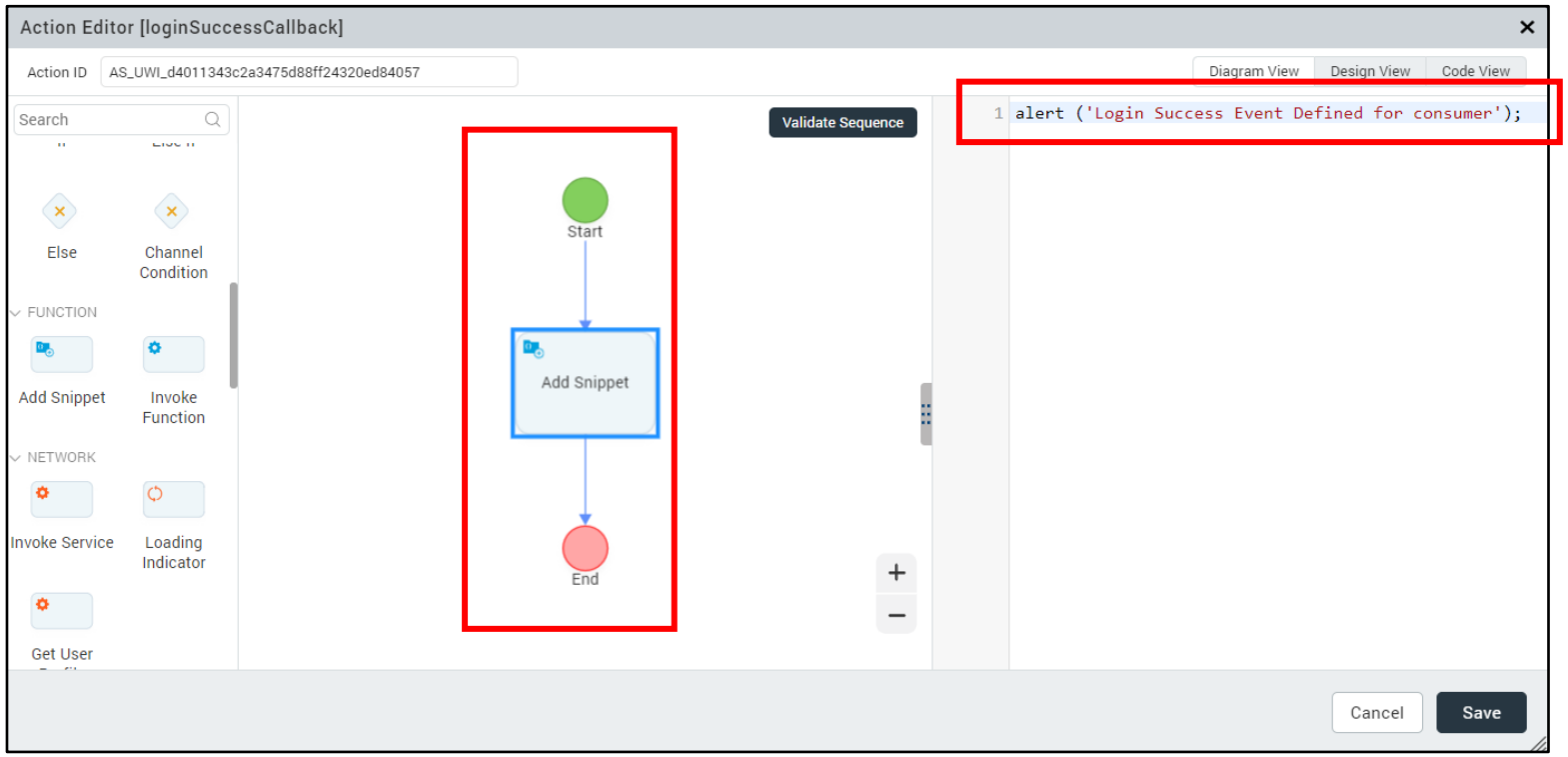






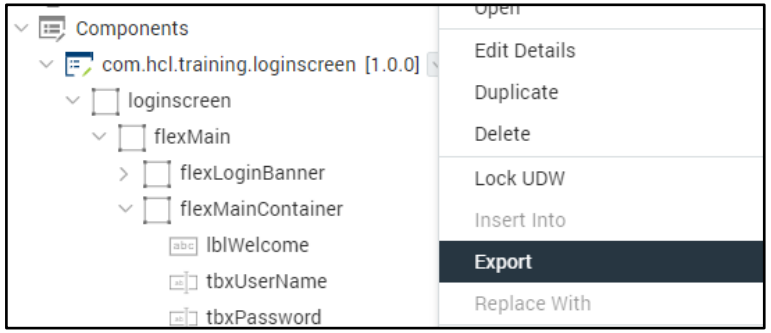
コンポーネントのカスタマイズ（続き）



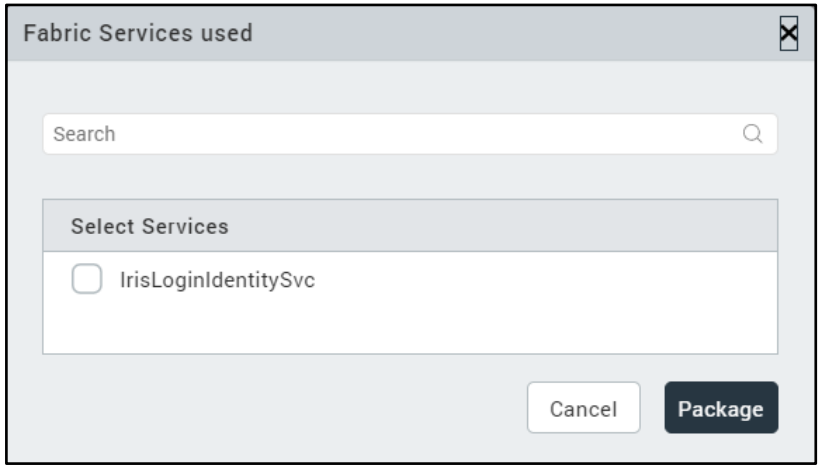


コンポーネントのエクスポートとインポート

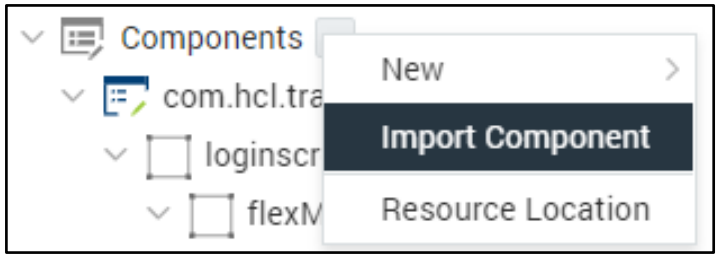
- エクスポートとインポートによるバックアップと共有できます。



- エクスポートする際に、どのサービスを含めるかを選択します。



- インポートオプションを選択して、コンポーネントを追加します。



練習

- このレッスンで説明した手順に従ってください。
- 設定するプロパティを追加する。
- 別のプロジェクトで使用するためにエクスポートとインポートを行う。

www.hcltech.com

$10 BILLION | 159,000+ IDEAPRENEURS | 50 COUNTRIES