Developing Android Apps with Liferay Screens

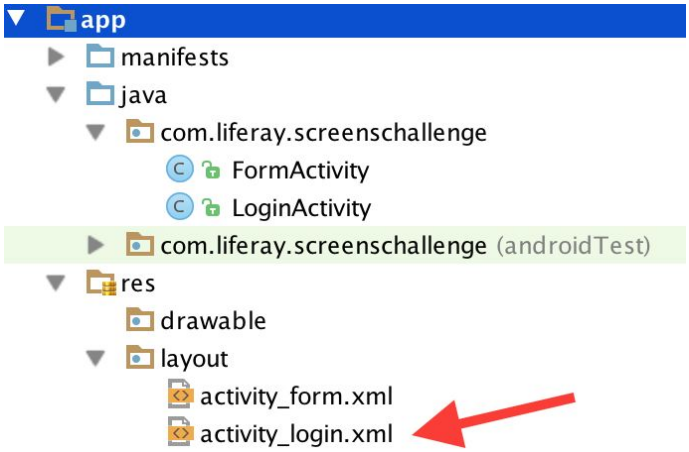
Introduction

Liferay Screens is a collection of visual components which integrates directly with the tools normally used for creating native mobile applications such as XCode for iOS and Android Studio for Android. The Liferay platform provides the backend for the mobile application handling user authentication, data persistence and serving content. The purpose of this exercise is to showcase using Liferay Screens with the Liferay platform as a mobile backend.

User Authentication with Liferay Screens

In this section, you will use a LoginScreenlet within Liferay Screens to handle user authentication in the mobile application. The user account will be created in the Liferay platform and will be in charge of handling the login on the backend.

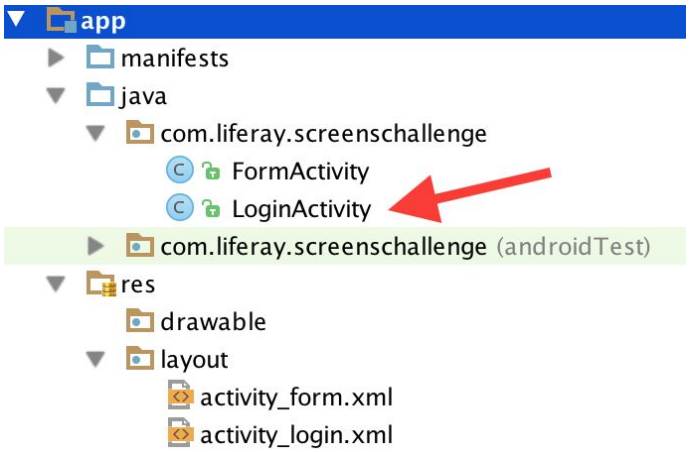
1. Create an account on the Screens Signup page if you haven't already done so.
2. In **Android Studio**, open res/layout/activity_login.xml file.



3. Locate the commented out **com.liferay.mobile.screens.auth.login.LoginScreenlet** and uncomment it by removing <!-- and -->.
4. The final result should look like the following:

```
<com.liferay.mobile.screens.auth.login.LoginScreenlet
   android:id="@+id/login_activity"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   liferay:layoutId="@layout/login_default" />
```

5. Open java/com.liferay.screenschallenge/LoginActivity



6. Locate the onCreate method and remove the comments by deleting // which creates a LoginListener. The method should now look like the following:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_login);

   LoginScreenlet loginScreenlet = (LoginScreenlet)
findViewById(R.id.activity_login);
   loginScreenlet.setListener(this);
}
```

7. Now in the onLoginSuccess and onLoginFailure methods uncomment the lines by removing // to enable logging events on successful and failed login attempts . The completed methods should look like the following:

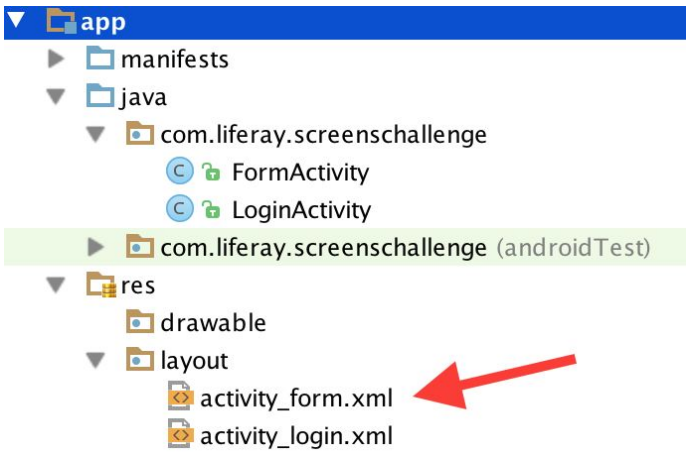
```
@Override
public void onLoginSuccess(User user) {
   System.out.println("Login is OK: " + user.getAttributes());
   startActivity(new Intent(this, FormActivity.class));
}

@Override
public void onLoginFailure(Exception e) {
   System.out.println("Login failed: " + e);
}
```

8. Run the project in Android Studio and switch to Genymotion for testing.
9. Try signing in with correct and incorrect user credentials. Notice the errors displayed by the Login screen in the application as well as the messages in the logs in Android Studio.

Displaying a Dynamic Form from Liferay

In this section, you will use a DDLFormScreenlet within Liferay Screens to display a dynamic form that is located on the backend Liferay platform. The form can be completed from within the mobile application and the captured data can be viewed from the backend Liferay platform.

1. In **Android Studio**, open res/layout/activity_form.xml file.



2. Locate the commented out **com.liferay.mobile.screens.ddl.form.DDLFormScreenlet** and uncomment it by removing <!-- and -->
3. The completed definition should look like:

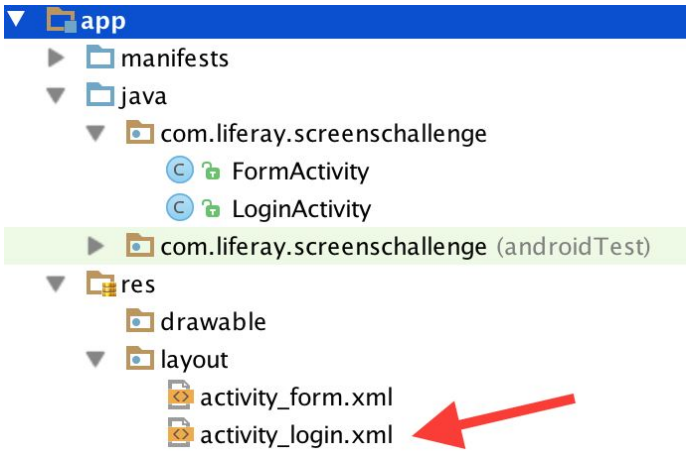
```
<com.liferay.mobile.screens.ddl.form.DDLFormScreenlet
   android:layout_height="match_parent"
   android:layout_width="match_parent"
   liferay:structureId="22354"
   liferay:recordSetId="22356"
   liferay:repositoryId="22339"
   liferay:folderId="22709"
   liferay:layoutId="@layout/ddl_form_default" />
```

4. That's it! Run the application, login using the same credentials used above and you'll see how the form will appear ready to be filled in.
    a. Leave some fields empty, and press Submit button. You'll see how validation fails because of required fields.
    b. Fill all values and press Submit button. You'll see a success message.
5. Go to the Screens List on the server. Notice the data that you used in the form appears near the top of the list.

Theming

In this section, you will learn how to modify the look & feel of your mobile application. For this, we use the concept of "viewset": a library of classes and layouts with a visual representation of the screenlets. You can change the screenlet layout and the look & feel will be changed, without changing the behavior.

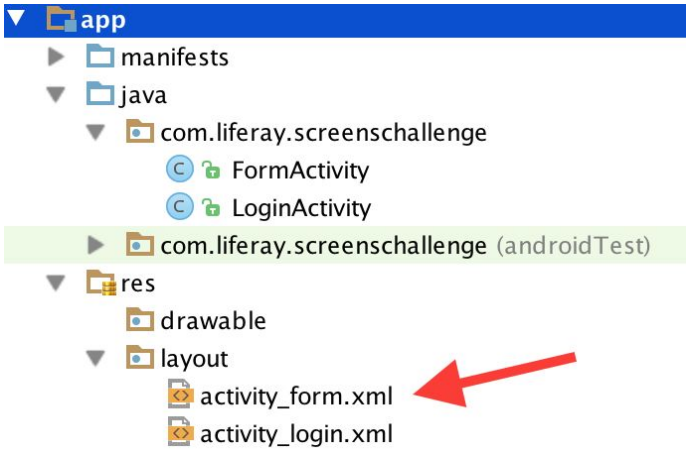
1. In Android Studio, re-open res/layout/activity_login.xml file.



2. In the **LoginScreenlet** XML tag, change the "layoutId" attribute to use the new viewset one (notice we use **@layout/login_material** instead of **@layout/login_default**)

```
<com.liferay.mobile.screens.auth.login.LoginScreenlet
   android:id="@+id/login_activity"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   liferay:layoutId="@layout/login_material" />
```

3. In Android Studio, re-open res/layout/activity_form.xml file.



4. In the **DDLFormScreenlet** XML tag, change the "layoutId" from **@layout/ddl_form_default** to **@layout/ddl_form_material**). The result should look like the following:

```
<com.liferay.mobile.screens.ddl.form.DDLFormScreenlet
   android:layout_height="match_parent"
   android:layout_width="match_parent"
```

```
   liferay:structureId="22354"
   liferay:recordSetId="22356"
   liferay:repositoryId="22339"
   liferay:folderId="22709"
   liferay:layoutId="@layout/ddl_form_material"
/>
```

5. Run the app again and notice how the Login Screen has a new look and feel. Login to the application and you'll also see how the form has a new look and feel applied.

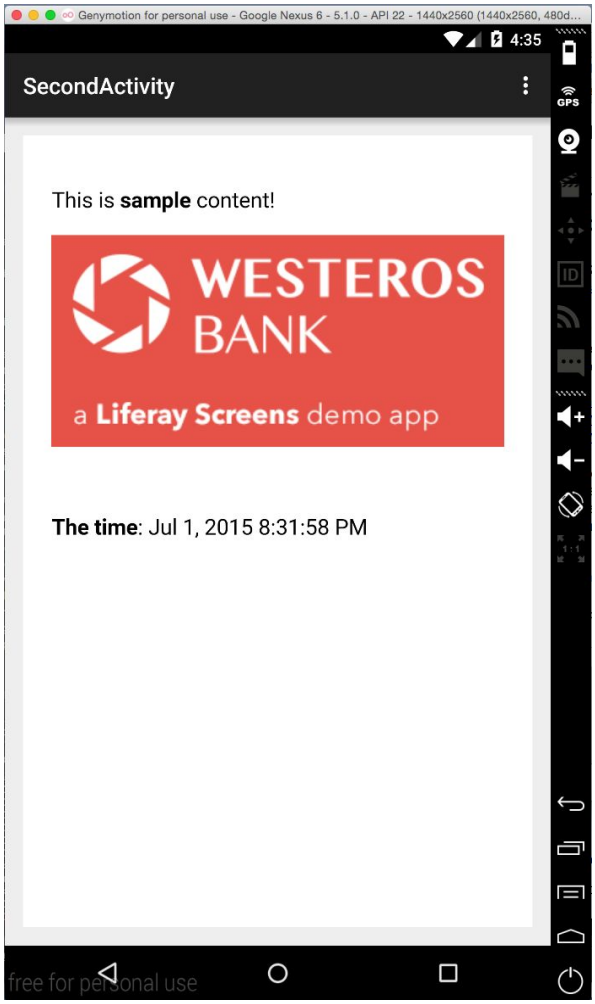
Post-Event Follow-up Activity

In this section, you will use the skills learned above to change the app to display web content from Liferay's web content management system. Don't worry, the content is already created, all you have to do is insert the screenlet and go!

1. In Android Studio, open FormActivity
2. Delete the <com.liferay.mobile.screens.ddl.form.DDLFormScreenlet> element.
3. Add a new <com.liferay.mobile.screens.ddl.form.WebContentArticleScreenlet> element to the activity_view_content.xml file.
4. Define the android:layout_height and android:layout_width attributes as match_parent
5. Define the liferay:groupId attribute as the repository id: **22339**
6. Define the liferay:articleId attribute as the repository id: **41713**
7. Define the liferay:layoutId attribute as @layout/webcontentdisplay_default which uses a predefined layout.
8. The completed definition should look like:

```
<com.liferay.mobile.screens.webcontentdisplay.WebContentDisplayScr
eenlet
   android:layout_height="match_parent"
   android:layout_width="match_parent"
   liferay:articleId="41713"
   liferay:groupId="22339"
   liferay:layoutId="@layout/webcontentdisplay_default" />
```

9. That's it! Launch the application, login using the same credentials used above and you'll see how the example content will appear:



10. Take a screenshot and post it to the Screens forum thread created for this event. Include your comments, questions, and other feedback! The first 10 people to do this will receive a small gift from Liferay! Note: in order to post, you will need to register for a free account at liferay.com if you do not yet have one.