Samsung Apps

Interworking Guide for Android Samsung Apps

Media Solution Center
Samsung Electronics

Table of Contents

1. Overview

This document describes how to launch the Samsung Apps client with special requests.

Using information of this document, you can directly invoke some Samsung Apps pages like the Main page, a Details page of an app, an app list of a category, an application list linked to a banner image and a Seller page.

2. Interworking Cases and Methods

Samsung Apps supports five interworking cases as below.

1) Invoking the Main page of the Samsung Apps client
2) Invoking a Details page of an app
3) Invoking a Seller page of a Seller
4) Invoking an app list of a category
5) Invoking an app list linked to a banner image

And Samsung Apps supports two methods for invoking the Samsung Apps client.

1) Linking an URI defined for the Samsung Apps on a Web page
2) Making and sending an Intent (Android)

3. Descriptions

3.1. Invoking the Main page of the Samsung Apps client

3.1.1. Linking an URI on a Web page

You can make an URI for invoking the Main page of the Samsung Apps client and link it on your Web page.

Below is the format and an example of the URI.

```
samsungapps://MainPage/

ex) <a href="samsungapps://MainPage/">Samsung Apps Main Page</a>
```

Below is the format and an example of the URI.

```
http://www.samsungapps.com/main/main.as

ex)
<a href="http://www.samsungapps.com/main/main.as"> A Main Page of
Samsung Apps Client</a>
```

3.1.2. Making and sending an Intent

If you develop an app and make a link for invoking the Main page of the Samsung Apps client in it, you have to make an Intent and send it to the Samsung Apps client.

Below is a sample code.

```
Intent intent = new Intent();

// set data
intent.setData(Uri.parse("samsungapps://MainPage/"));

// add flags
intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK                          |
Intent.FLAG_ACTIVITY_CLEAR_TOP                                         |
Intent.FLAG_INCLUDE_STOPPED_PACKAGES);
// The Intent.FLAG_INCLUDE_STOPPED_PACKAGES flag must be added only
if the API level of your Android SDK is over 11 like Honeycomb.

startActivity(intent);
```

3.2. Invoking a Details page of an application

3.2.1. Linking an URI on a Web page

You can make an URI for invoking a Details page of an app and link it on your Web page.

Below is the format and an example of the URI.

```
samsungapps://ProductDetail/{the package name of the
AndroidManifest.xml file in your application}

ex) In case of the package name of the ChatON APK
<a href="samsungapps://ProductDetail/com.sec.chaton">ChatON</a>
```

Below is the format and an example of the URI.

```
http://www.samsungapps.com/appquery/appDetail.as?appId={the package
name of the AndroidManifest.xml file in your application}

ex) In case of the package name of the ChatON APK
<a href="http://www.samsungapps.com/appquery/appDetail.as?appId=
com.sec.chaton">ChatON</a>
```

3.2.2. Making and sending an Intent

If you develop an app and make a link for invoking a Details page of an app in it, you have to make an Intent and send it to the Samsung Apps client.

Below is a sample code.

```
Intent intent = new Intent();

// set data
intent.setData(Uri.parse(string_of_uri)); // The string_of_uri is an
String object including a URI like "samsungapps://ProductDetail/{the
package name of the AndroidManifest.xml file in your application}".

// add flags
intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK                          |
Intent.FLAG_ACTIVITY_CLEAR_TOP                                         |
Intent.FLAG_INCLUDE_STOPPED_PACKAGES);
// The Intent.FLAG_INCLUDE_STOPPED_PACKAGES flag must be added only
if the API level of your Android SDK is over 11 like Honeycomb.

startActivity(intent);
```

3.2.3. Try and Purchase

You can develop a 'Trial version' which has a link, banner, button and etc to a paid app using this method.

When user clicks the link, banner, button and etc, as phrase 3.2.2, it will invoke the Details page of the paid app.

3.3. Invoking a Seller page

You can get the globally unique ID of Seller(=Seller DeepLink ID) at the Seller Office (My Profile > Profile > Information for Seller Page). It is 10-digit alphanumeric unique ID which is automatically generated when a Seller signs up at Seller Office.

3.3.1. Linking an URI on a Web page

You can make an URI for invoking a Seller page and link it on your Web page.

Below is the format and an example of the URI.

```
samsungapps://SellerDetail/{Seller DeepLink id from Seller Office >
My Profile > Profile > Information for Seller Page}

  ex) In case of the Gameloft(1a2b3c4d5e is fake data for sample)
    <a href="samsungapps://SellerDetail/1a2b3c4d5e">Gameloft</a>
```

Below is the format and an example of the URI.

```
http://www.samsungapps.com/appquery/sellerProductList.as?SellerID=
{Seller DeepLink id from Seller Office > My Profile > Profile >
Information for Seller Page}

  ex) In case of the Gameloft(1a2b3c4d5e is fake data for sample)
<a href="http://www.samsungapps.com/appquery/sellerProductList.as?
SellerID=1a2b3c4d5e">Gameloft</a>
```

3.3.2. Making and sending an Intent

If you develop an app and make a link for invoking a Seller page in it, you have to make an Intent and send it to the Samsung Apps client.

Below is a sample code.

```
Intent intent = new Intent();

// set data
intent.setData(Uri.parse(string_of_uri)); // The string_of_uri is an
String object including a URI like
"samsungapps://SellerDetail/{Seller DeepLink id}".

// add flags
intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TOP |
Intent.FLAG_INCLUDE_STOPPED_PACKAGES);
```

```
// The Intent.FLAG_INCLUDE_STOPPED_PACKAGES flag must be added only
if the API level of your Android SDK is over 11 like Honeycomb.

startActivity(intent);
```