

# LG QCircle SDK Guide

---

Version 2.0 – September 14, 2014

LGD-QCTD-SDK-DG-V1.0EN

**LG Mobile Developer Network**

**Mobile Communication Company  
Mobile Handset R&D Center**

**Copyright © 2014 LG Electronics Inc. All Rights Reserved.**

Though every care has been taken to ensure the accuracy of this document, LG Electronics Inc. cannot accept responsibility for any errors or omissions or for any loss occurred to any person, whether legal or natural, from acting, or refraining from action, as a result of the information contained herein. Information in this document is subject to change at any time without obligation to notify any person of such changes.

LG Electronics Inc. may have patents or patent pending Applications, trademarks copyrights or other intellectual property rights covering subject matter in this document. The furnishing of this document does not give the recipient or reader any license to these patents, trademarks copyrights or other intellectual property rights.

No part of this document may be communicated, distributed, reproduced or transmitted in any form or by any means, electronic or mechanical or otherwise, for any purpose, without the prior written permission of LG Electronics Inc.

The document is subject to revision without further notice.

## Revision History

Document Version	Date	Comment
1.0	June 09, 2014	LG QCircle SDK Document Release
2.0	Sep 14, 2014	Added the UI Design & UX Scenario Guide

# Contents

<b>About This Document .....</b>	<b>8</b>
<b>1 Introduction .....</b>	<b>10</b>
1.1 LG QCircle Overview .....	11
1.1.1 QuickCircle case.....	11
1.1.2 QuickCircle UX Features.....	11
Direct Access to Screen for Applications and Notification.....	11
1.2 LG QCircle SDK Overview.....	13
1.2.1 Components of the LG QCircle SDK.....	13
1.2.2 Key Features of the LG QCircle SDK.....	13
1.2.3 Supported Devices.....	13
1.3 Terminology.....	14
<b>2 Getting Started.....</b>	<b>15</b>
2.1 Setting up Android Development Environment .....	16
2.1.1 Setting up Android Development Environment .....	16
2.2 Downloading & installing the LG QCircle SDK.....	18
2.3 LG QCircle SDK Release Notes .....	19
2.3.1 Android API Level 19, Revision 1 .....	19
New Features.....	19
Known Issues .....	19
<b>3 Developing .....</b>	<b>20</b>
3.1 Overview.....	21
3.1.1 Architectural Overview .....	21
3.1.2 QCircle Application Event Handling Flow.....	22
3.2 Creating Projects .....	23
3.2.1 Creating an Android Projects.....	23
3.2.2 Running Application using LG QCircle.....	24
3.3 Key Implementation.....	27
3.3.1 Managing a Broadcast Receiver .....	27
Creating an Intent Filter .....	27

Creating a Broadcast Receiver .....	27
Registering the Broadcast Receiver .....	27
Unregistering the Broadcast Receiver .....	27
3.3.2 Handling a Cover Event .....	28
Checking the Availability of a QuickCircle case.....	28
Getting the Current State of the Front Cover .....	28
3.3.3 Creating a Layout for the QuickCircle Window .....	29
Getting Information of the QuickCircle window .....	29
Taking Precedence over Lock Screen.....	30
3.3.4 Modifying the AndroidManifest.xml file.....	30
Presenting the Application on Settings.....	30
Removing the Title bar .....	31
Fixing the Orientation .....	31
Adding an Application Icon .....	31
3.4 API References .....	32
3.4.1 LG QCircle Intents .....	32
com.lge.android.intent.action.ACCESSORY_COVER_EVENT .....	32
com.lge.intent.extra.ACCESSORY_COVER_STATE .....	32
<b>4 Development Tips.....</b>	<b>33</b>
Emulating a Broadcast Intent via ADB Tool.....	34
Excluding QCircle Activity from Recent Used Apps .....	35
<b>5 UI Design &amp; UX Scenario Guide.....</b>	<b>36</b>
5.1 UI Design Guide.....	37
5.1.1 Overveiw.....	37
Lock Screen .....	37
Menu Screen .....	37
Common QCircle Application UI.....	38
5.1.2 Application UI.....	39
Circle background .....	39
Back button .....	40
Title 41	
5.1.3 Iconography.....	41
5.1.4 Supporting Multiple Screens.....	42
Supporting Different Screen Densities.....	42
Setting the Layout Dynamically in the Source.....	43

5.2	UX Scenario Guide.....	45
5.2.1	Overview.....	45
	Quick Circle UX Principles.....	45
5.2.2	Standard Interaction.....	46
5.2.3	Basic UX Flows.....	46
	Cover Closed.....	46
	Cover Open .....	47
	Launching Application.....	48
	Back to Lock Screen .....	49
	Settings.....	50
<b>6</b>	<b>Sample Tutorials.....</b>	<b>51</b>
6.1	Sample Application Overview.....	52
6.1.1	Introduction to the Sample Application.....	52
	QR Color Change .....	52
6.1.2	Importing the Sample Application.....	53
6.2	Sample Code Analysis.....	56
6.2.1	QR Color Change.....	56
	Getting Ready.....	56
	Managing a Broadcast Receiver.....	56
	Handling a Cover Event.....	57
	Adding an Window Flag .....	58
	Getting Cover Information.....	58
	Cropping the Application Layout .....	59
	Modifying the AndroidManifest.xml file .....	59

## Tables

<b>Table 3-1</b> QuickCircle case enabled/disabled information database field .....	28
<b>Table 3-2</b> Case type information database field .....	28
<b>Table 3-3</b> Descriptions of QuickCircle window parameters .....	30
<b>Table 5-1</b> QR Color Change files.....	56

## Figures

<b>Figure 1.1</b> QuickCircle case.....	11
<b>Figure 1.2</b> Application icons in the QuickCircle window .....	12
<b>Figure 1.3</b> Selecting applications from Settings.....	12
<b>Figure 3.1</b> LG QCircle framework.....	21
<b>Figure 3.2</b> Examples of QCircle application views in the QuickCircle window .....	29
<b>Figure 3.3</b> Properties of QuickCircle window size and position.....	29
Figure 5.1 Lock Screen .....	37
Figure 5.2 Menu Screen.....	38
Figure 5.3 Application UI .....	38
Figure 5.4 UI components.....	39
Figure 5.5 Circle Background.....	40
Figure 5.6 Weighted portion of the back button area .....	41
Figure 5.7 Weighted portion of the title area .....	41
Figure 5.8 UX flow of the cover closed .....	47
Figure 5.9 UX flow of the cover open.....	47
Figure 5.10 Navigating to menu screen .....	48
Figure 5.11 UX flow of launching application .....	48
Figure 5.12 Launching full screen activity .....	49
Figure 5.13 Back to lock screen.....	50
Figure 5.14 Setting QCircle applications .....	50
<b>Figure 6.1.</b> Sample application screen in the QuickCircle Window .....	52
<b>Figure 6.2.</b> Sample application screen in full screen.....	52

# About This Document

---

This document provides explanations for developing an application by using LG QCircle SDK. The document introduces LG QCircle SDK to help developers develop an application based on the guide. In this document, overall process of developing an Android application including the environmental setup, LG QCircle API reference,

## Who This Guide is for

This document is designed for:

- Developers who plan to develop Android applications using LG QCircle SDK
- Developers who plan to develop Android applications compatible with a QuickCircle case

## Conventions

### Notes & Cautions

Notes and Cautions are used to emphasize information. The following samples describe when each is used.

---

#### Note

Note contains information about something that is helpful to you.

---

---

#### Caution

Caution contains important information about something that you should know.

---



# Organization

This document is organized as:

- **Chapter 1** '[Introduction](#)' introduces LG QCircle features. The chapter also provides an overview of LG QCircle SDK.
- **Chapter 2** '[Getting Started](#)' explains setting up the environment for developing Android applications with LG QCircle SDK.
- **Chapter 3** '[Developing](#)' explains implementing an application by using LG QCircle SDK and also provides API reference.
- **Chapter 4** '[Development Tips](#)' provides useful tips for developing Android applications with LG QCircle SDK
- **Chapter 5** '[Sample Tutorials](#)' provides sample codes with code analysis you can refer to

# 1 Introduction

---

This chapter describes an overview of LG QuickCircle case and the LG QCircle SDK. The LG QuickCircle case is a hardware which is a folio phone case for LG smart phones. The LG QCircle SDK is a software development tool that allows the creation of applications using features of the case. Also, this document provides the related terminology.

## Contents

### [1.1 LG QuickCircle Overview](#)

This section provides an overview of LG QuickCircle Case and its UX features.

### [1.2 LG QCircle SDK Overview](#)

This section describes the components and the key features of the SDK, and provides a list of supported devices.

### [1.3 Terminology](#)

This section describes terms used in the LG QCircle SDK.

## 1.1 LG QCircle Overview

The QuickCircle case is a smart phone cover case with a circular window cut-out on the front. The case not only provides protection to a smart phone, but, once attached, also provides various UX features. For example, users can select applications directly through the cover's QuickCircle window. In addition, users can receive notifications and check new messages.

### 1.1.1 QuickCircle case

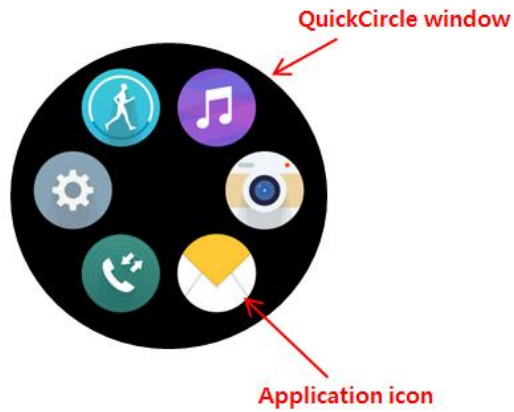


Figure 1.1 QuickCircle case

### 1.1.2 QuickCircle UX Features

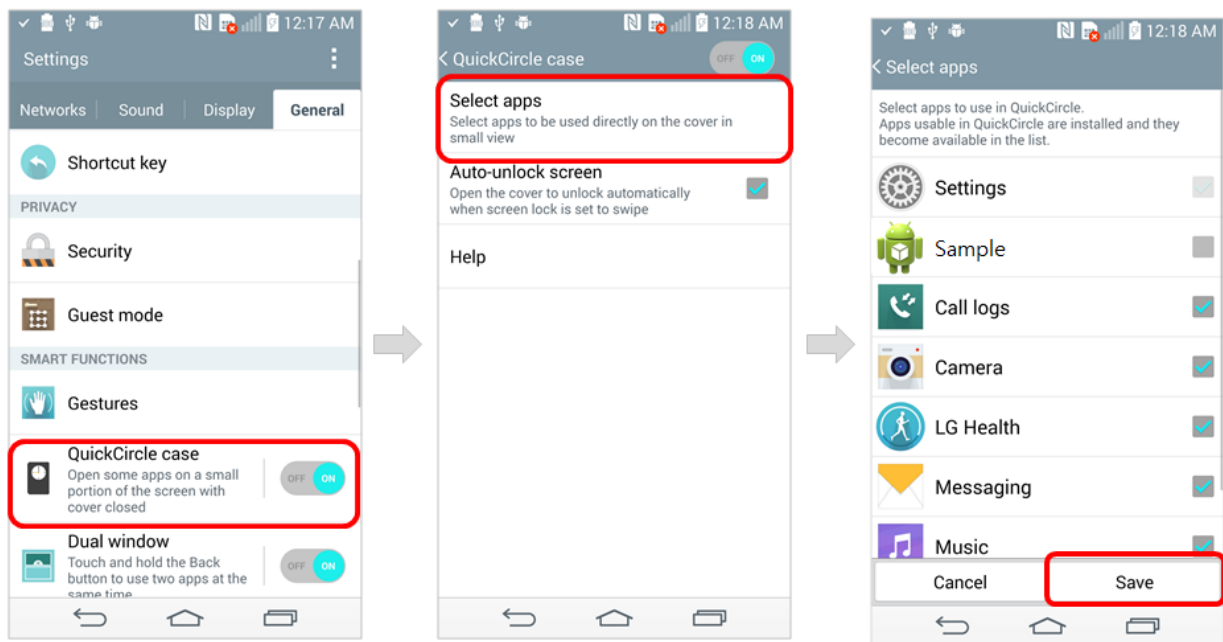
#### Direct Access to Screen for Applications and Notification

The QuickCircle displays application icons through Quick Circle window. Users can select and use applications directly without opening the cover. Icons are presented as in the layout shown below:



**Figure 1.2** Application icons in the QuickCircle window

To use QCircle Direct Access feature, a user should navigate through Settings, and click on the "QuickCircle Case" item. Applications usable through QuickCircle window are preinstalled and are available to be chosen in the Settings list. A user can select up to six applications to be displayed on QuickCircle window.



**Figure 1.3** Selecting applications from Settings

## 1.2 LG QCircle SDK Overview

LG QCircle SDK provides a set of open APIs that allows developers to make the best use out of LG QCircle features. Developers can make their applications directly accessible from and act accordingly with the QuickCircle cover.

### 1.2.1 Components of the LG QCircle SDK

The LG QCircle SDK consists of the following components:

- **Documentation:** LG QCircle SDK developer guide.
- **Samples:** A sample application and its source code using LG QCircle APIs.
- **Emulator:** Plug-in Android emulator for testing QCircle applications. (To be added)

### 1.2.2 Key Features of the LG QCircle SDK

LG QCircle SDK provides following key features:

- Recognizing the current state of the cover, open or closed
- Running a QCircle application in the QuickCircle window
- Providing the QuickCircle window information for customizing an application layout

### 1.2.3 Supported Devices

LG QCircle SDK supports the following device:

- LG G3

## 1.3 Terminology

Terms used in the LG QCircle SDK are explained below.

Term	Description
QuickCircle case	The name of the case for LG smart phones.
QuickCircle window	The cut-out window on the front cover of the QuickCircle case.
QCircle Application	An application which provides QuickCircle features. It generally indicates applications developed using LG QCircle SDK.
QCircle Intent	Broadcast intents included in QCircle APIs.

## 2 Getting Started

---

This document describes how to install and use the LG QCircle SDK.

### [2.1 Setting up Android Development Environment](#)

This section describes how to set up the Android Development Environment.

### [2.2 Downloading & Installing the LG QCircle SDK](#)

This section describes where to download and how to install the LG QCircle SDK.

### [2.3 LG QCircle SDK Release Notes](#)

This section describes the changes and enhancements made to the LG QCircle SDK as it has evolved between releases. It identifies new features, changes, and known issues for each release.

## 2.1 Setting up Android Development Environment

This section describes how to set up the environment for developing applications with the LG QCircle SDK. The following tools are required.

- Android Development Environment
  - JDK
  - Eclipse
  - ADT Plug-in
  - Android SDK

### 2.1.1 Setting up Android Development Environment

The LG QCircle SDK works on the Android platform. Therefore, before starting to develop LG QCircle applications, make sure that Android development environment is available on your PC. If the following tools are not installed on your PC, download the latest version and install it.

- **JDK (Java Development Kit)**

It is a software which provides development environment for Java applications. Android application is based on Java, so make sure that JDK is installed on your PC. If not, you should install it. For more installation guide, refer to the following link:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

- **Eclipse**

It is a software which provides IDE (Integrated Development Environment).

- **ADT (Android Developer Tools) Plug-in**

It is a software which integrates Android development environment into Eclipse.

- **Android SDK (Software Development Kit)**

It is a software which enables developers to create applications for Android platform.

Eclipse, ADT and Android SDK are provided under the name of 'ADT Bundle' at Android Developer Site. ADT Bundle can be downloaded at the following link:

<http://developer.android.com/sdk/index.html>

---

**Note**

These tools have platform dependency, so you should choose the appropriate version depending on your PC environment such as window 32-bit, window 64-bit, Linux 32-bit, etc.

If you want to use an existing IDE, you don't need to download a full ADT Bundle. Choose the option,



DOWNLOAD FOR OTHER PLATFORMS > **SDK Tools Only**.

If you don't have any IDE, you should download a full ADT Bundle including Eclipse. Choose the option, DOWNLOAD FOR OTHER PLATFORMS > **ADT Bundle**.

---

## 2.2 Downloading & installing the LG QCircle SDK

The LG QCircle SDK can be downloaded manually with Zip file.

1. On the LG Developer Site, download the zip file from the following path.

Resource Center > Mobile > Android > SDK & Tools > SDK > [LG QCircle SDK](#)

2. Simply unpack it to the 'extras' directory, which is the sub-directory of the root directory that you installed Android SDK on your PC.

Ex) < root directory of your android-sdk installation>/<extras>/<LG>/<QCircle>

This directory path is not compulsory but we recommend you to use it for effective file management for your Android projects.

## 2.3 LG QCircle SDK Release Notes

This section describes the changes and enhancements made to the LG QCircle SDK as it has evolved between releases. It identifies new features, changes, and known issues for each release.

### 2.3.1 Android API Level 19, Revision 1

Release Date: 2014/ 06/2

#### New Features

The LG QCircle SDK provides APIs to develop an application using LG QCircle features. LG QCircle enables features interactive with the LG phone case, “QuickCircle case”.

LG QCircle APIs provide the following key features:

- Recognizing the current state of the cover, open or closed
- Running a QCircle application in the QuickCircle window.
- Providing QuickCircle window information for customizing application layout

#### Known Issues

This section summarizes the issues that developers are most likely to encounter when using the LG QCircle SDK. It is not an exhaustive list of known bugs.

None

# 3 Developing

---

This chapter is intended to help developers who want to develop applications using the LG QCircle SDK. It shows how to set an Android Project and implement LG QCircle features into your application. API references are also provided.

## Contents

### [3.1 Overview](#)

This section provides the architectural overview of LG QCircle framework and explains a typical flow of handling an event of the front cover.

### [3.2 Creating Projects](#)

This section describes how to create and run a project using the LG QCircle APIs.

### [3.3 Key Implementation](#)

This section describes how to implement applications using LG QCircle, and provides sample codes for using LG QCircle features.

### [3.4 API References](#)

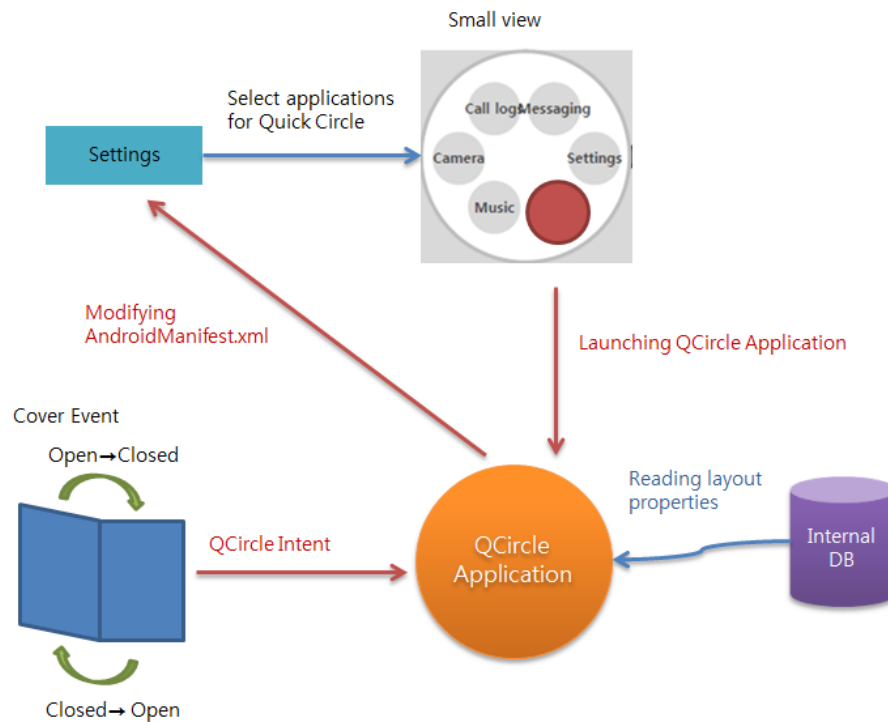
This section provides the LG QCircle API references. Descriptions of all available APIs are given in this section.

## 3.1 Overview

This section gives some background information to understand the LG QCircle SDK.

### 3.1.1 Architectural Overview

The diagram below shows the general workflow of the LG QCircle framework architecture.



**Figure 3.1** LG QCircle framework

Here's explanation for each component:

- **QCircle Application:** When an application icon is selected in the QuickCircle window, the activity of the application is called by the lower framework.
- **QCircle Intent:** LG QCircle framework sends the QCircle intent which is the broadcast intent to notify about cover events. QCircle application receives a QCircle intent to detect the cover events and completes the related actions. QCircle intent includes the extra data of the current state of the cover.
- **Settings:** You can list your application in Settings by modifying AndroidManifest.xml file. When the application is listed in Settings, the user can choose to have the application icon displayed in the QuickCircle window. The activity of the selected application is registered to system.
- **Quick Circle window:** Some applications can be used through the Quick Circle window. When an icon is selected in the window, corresponding application is launched by startActivity.
- **Internal DB:** You can customize the application layout shown in the QuickCircle window when the front cover is closed. Internal DB stores the Quick Circle window information including its height and width, and the position on the cover. You can also access the Internal DB to find out whether the user installed the Quick Circle case on a device.

### 3.1.2 QCircle Application Event Handling Flow

LG QCircle API provides QCircle intents to get information about the event and the current state of the cover. By using LG QCircle API, you can develop your application to operate differently according to the current state of the cover. Information on the LG QCircle API is provided in API reference. The below is the event flow for handling a broadcast intent in the QCircle application.

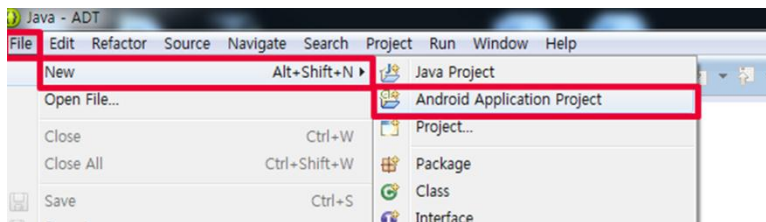
1. **Registering a broadcast receiver:** The broadcast receiver that can receive QCircle intents is registered in the system. The intent filter that allows the broadcast receiver to only receive LG QCircle related broadcast intent can be registered with the broadcast receiver.
2. **Receiving a QCircle intent:** The broadcast intent will be sent according to the event on the front cover.
3. **Checking LG QCircle availability:** The QCircle application checks if the user installed a Quick Circle case. The application determines whether to provide LG QCircle UX.
4. **Completing a related action according to the front cover state:** The application can complete a related action according to the current state of the cover.

## 3.2 Creating Projects

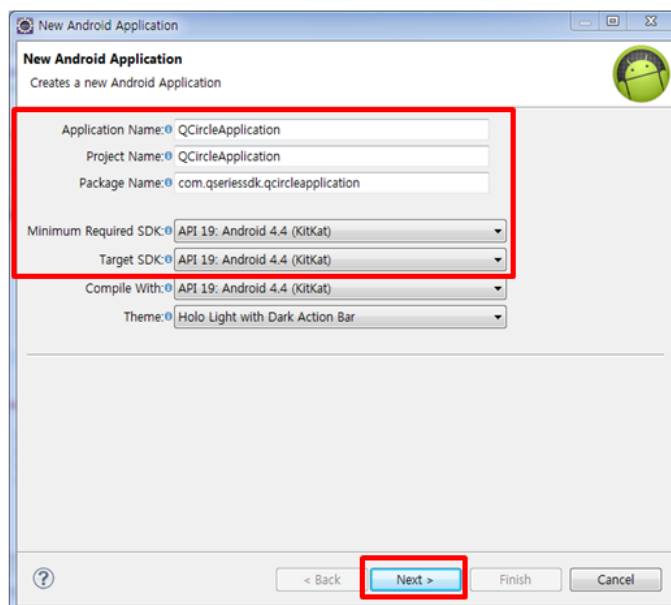
This section describes how to create and run a project using the LG QCircle APIs.

### 3.2.1 Creating an Android Projects

1. Start Eclipse.
2. Select **File > New > Android Application Project**.



3. Fill in the form as follows.

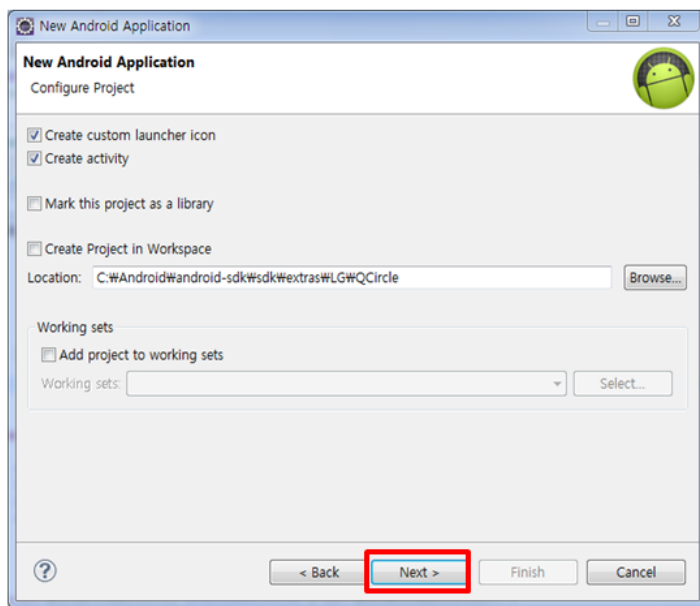


- **Application Name:** Enter the appropriate name for your application.
- **Project Name:** Enter the appropriate name for your application.
- **Package Name:** Enter the appropriate name for your application
- **Minimum Required SDK:** Select the 'API 19: Android 4.4 (Kit Kat)'.

#### Note

LG QCircle SDK requires the minimum level of Android API 19.

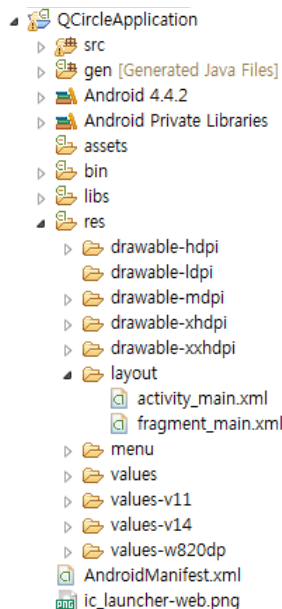
4. Check the boxes if you need to and click **Finish**.



#### Note

Creating a launcher icon or activity is not mandatory for applications using QCircle. Those settings depend on the purpose of your application.

5. With the options above, the Android Project will be created as follows.

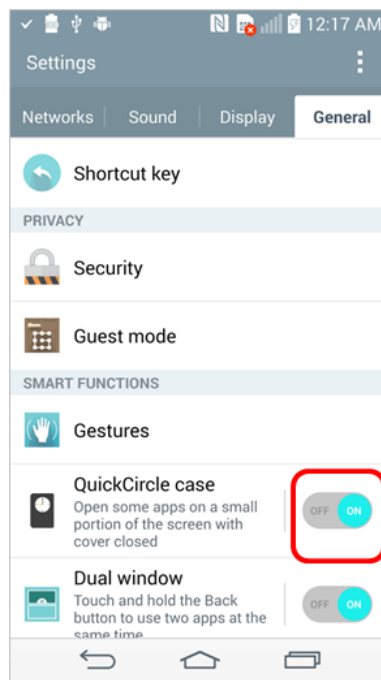


### 3.2.2 Running Application using LG QCircle

You can run an application developed using the LG QCircle SDK on real devices in the same way as other Android applications.

1. Connect the QuickCircle case with your LG smart phone.
2. Go to Settings and turn on the smart function for QuickCircle Case.





---

**Note**

If you want to simply test your application without directly using the QuickCircle case, you can emulate the running environment. Please, refer to [Development Tips](#).

---

3. Connect your smart phone to PC.

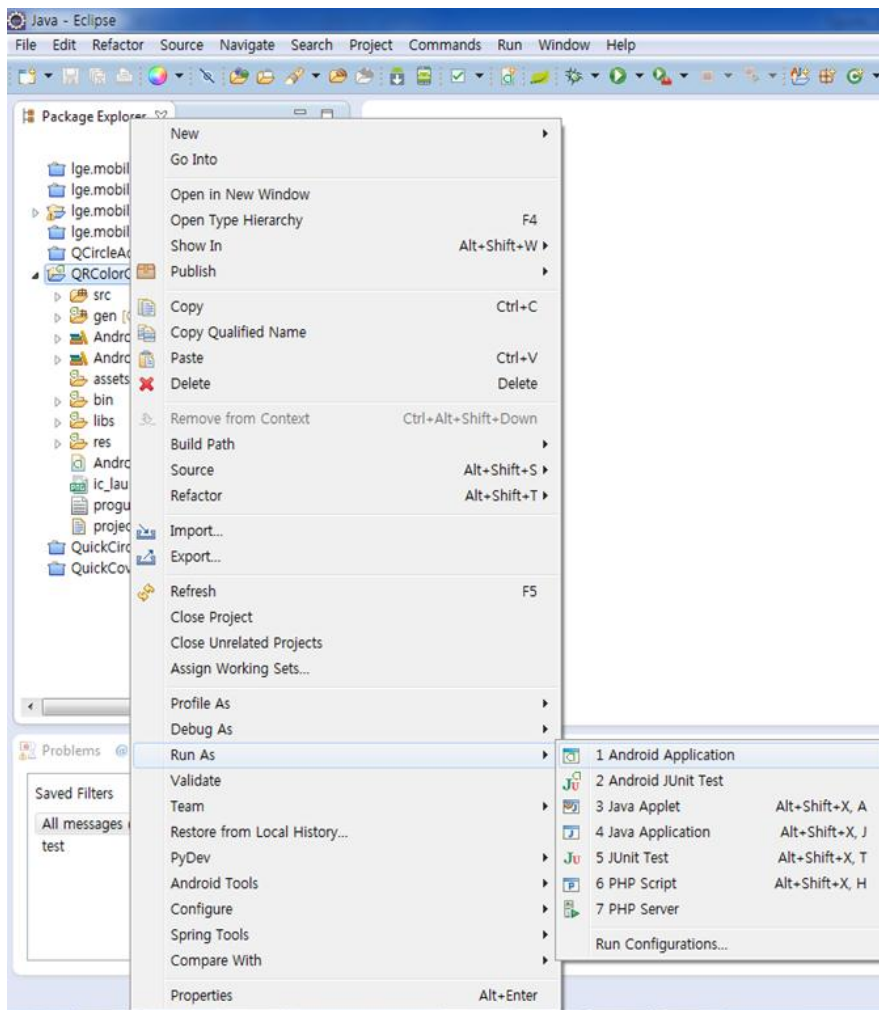
---

**Note**

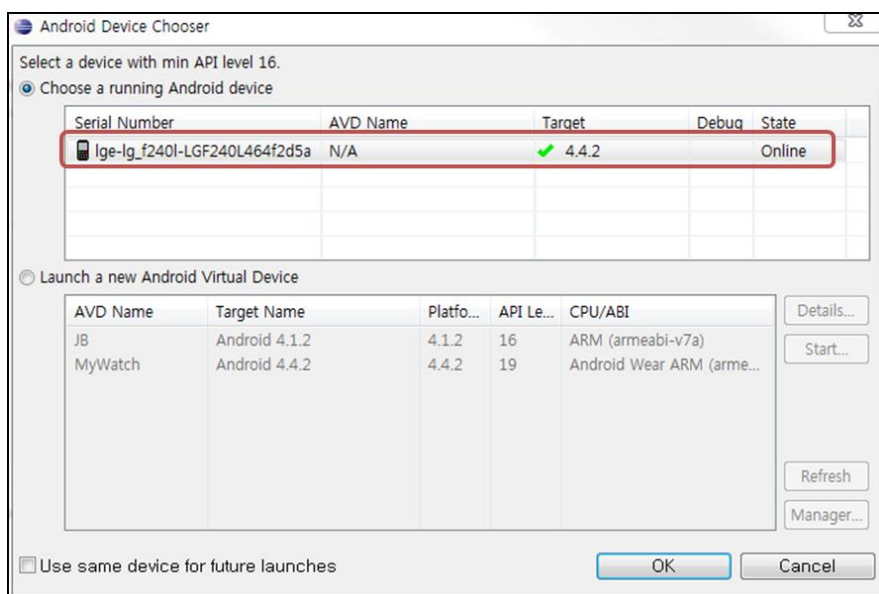
If your PC does not recognize the device, install the USB driver for LG smartphones from the LG Electronics website: <http://www.lg.com>

---

4. Run the application by selecting from the menus as follows: project name > **Run as** > **Android Application**.



5. When the pop-up window appears, select the device and then click **OK**.



## 3.3 Key Implementation

This section provides detailed explanation for implementing QCircle application including managing a broadcast receiver and getting the layout information. You can refer to [API References](#) for LG QCircle intents.

### 3.3.1 Managing a Broadcast Receiver

To receive QCircle intents, you need to create a broadcast receiver. In the broadcast receiver, you can implement codes for each cover event.

#### Creating an Intent Filter

The QCircle application needs to receive specific broadcast intents that are related to LG QCircle only. To filter out LG QCircle intents, create an intent filter and add the QCircle intent for the cover event.

```
public class QuickCircleViewActivity extends Activity {
    public static final String ACTION_ACCESSORY_COVER_EVENT =
"com.lge.android.intent.action.ACCESSORY_COVER_EVENT";
    ..
    private void registerIntentReceiver() {
        IntentFilter filter = new IntentFilter();
        filter.addAction(ACTION_ACCESSORY_COVER_EVENT);
        ..
    }
}
```

#### Creating a Broadcast Receiver

A broadcast receiver is created to receive the QCircle intents. You can use Broadcast Receiver class.

```
public class QuickCircleViewActivity extends Activity{
    private BroadcastReceiver mIntentReceiver = new BroadcastReceiver() {
        ..
    };
}
```

#### Registering the Broadcast Receiver

You can register the broadcast receiver as shown below.

```
mContext = getApplicationContext();
mContext.registerReceiver(mIntentReceiver, filter);
```

#### Unregistering the Broadcast Receiver

You must unregister the broadcast receiver when the application is either stopped or destroyed.

```
mContext.unregisterReceiver(mIntentReceiver);
```

### 3.3.2 Handling a Cover Event

The broadcast receiver can complete actions according to the current cover state. You can refer to the event handling flow to implement the cover-related actions in a broadcast receiver

#### Checking the Availability of a QuickCircle case

You first have to check whether any cover case has been attached to the user's smart phone. Then, you need to check if the user has decided to use the cover's functionality. You can check both by accessing *quick\_view\_enable* database field shown in Table 3-1.

**Table 3-1** QuickCircle case enabled/disabled information database field

Field Name	Integer value	Meaning	Default
quick_view_enable	0	Disabled	Disabled
	1	Enabled	

This is the example code for reading the database field to check whether the case is both attached and enabled.

```
contentResolver = getContentResolver();
quickCoverEnabled = Settings.Global.getInt(contentResolver,
quick_view_enable", 0) == 1 ? true : false;
```

Once the case is available, you can read the case type from the database field to see if the case is a QuickCircle case. The value 3 indicates that the user has a QuickCircle case installed.

**Table 3-2** Case type information database field

Field Name	Integer value	Default
cover_type	3	0

This is the example code for reading the database field to check a case type.

```
quickCaseType = Settings.Global.getInt(contentResolver, "cover_type", 0);
if(quickCaseType != 3)
{
    ..
}
```

#### Getting the Current State of the Front Cover

Once the application has received a broadcast intent from the front cover event, you can read the extra data from the broadcast intent to retrieve the current front cover state. The description of the QCircle broadcast intent is provided in [API reference](#). The code below shows how to receive the cover event and retrieve its current state.

```
String action = intent.getAction();
if (ACTION_ACCESSORY_COVER_EVENT.equals(action)) {
    mQuickCoverState =
intent.getIntExtra( EXTRA_ACCESSORY_COVER_STATE, EXTRA_ACCESSORY_COVER_OPENED)
;
```

```

if(mQuickCoverState ==1) {
    ..
}
else if (mQuickCoverState == 0) {
    ..
}
}

```

### 3.3.3 Creating a Layout for the QuickCircle Window

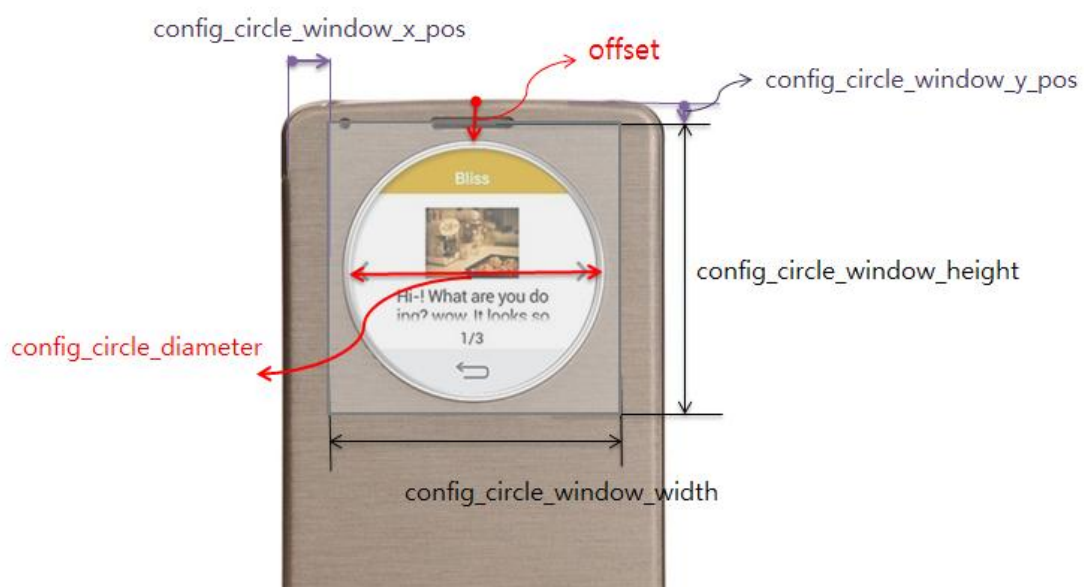
#### Getting Information of the QuickCircle window

When the cover is closed and the QCircle application is launched, the application will be shown through the QuickCircle window. You can customize the layout of your application in this window. LG QuickCircle SDK provides information about the QuickCircle window properties.



**Figure 3.2** Examples of QCircle application views in the QuickCircle window

The layout of the QuickCircle window is defined by five properties. Each description is provided in Table 3-3.



**Figure 3.3** Properties of QuickCircle window size and position

**Table 3-3** Descriptions of QuickCircle window parameters

Name	Description
<code>config_circle_window_x_pos</code>	The position of the QuickCircle window on the horizontal axis of the case from the left
<code>config_circle_window_y_pos</code>	The position of the QuickCircle window on the vertical axis of the case from the top
<code>config_circle_window_width</code>	The width of square area which the QuickCircle window fits in
<code>config_circle_window_height</code>	The height of square area which the QuickCircle window fits in
<code>config_circle_diameter</code>	The diameter of the QuickCircle window

The code below shows how to get the parameter value from the internal database. You can set the name of each parameter listed in Table 3-3. All the units are in pixels. Any negative value for either `config_circle_window_x_pos` or `config_circle_window_y_pos` indicates its position is at the center of the QuickCircle cover.

```
int id = getResources().getIdentifier("Parameter Name", "dimen",
"com.lge.internal");
int value = getResources().getDimensionPixelSize(id);
```

**NOTE**

The diameter of the window is smaller than the widow width. In order to fit the whole layout width of your application to the QuickCircle window, you should set your layout width to the value of `config_circle_diameter`. The difference between `config_circle_window_width` and `config_circle_diameter` indicates the gap between the device and the case, which is not normally seen from the top, but can be seen when viewed at an angle.

The offset value shown in Figure 3-3 gives you the exact location of your app to be positioned. Its value can be calculated as below:

$$\text{offset} = \text{config\_circle\_window\_y\_pos} + (\text{config\_circle\_window\_height} - \text{config\_circle\_window\_width})/2$$

**Taking Precedence over Lock Screen**

Your application needs to have control over the Lock screen; otherwise, the lock screen will hide your application view. To avoid this, you can raise `FLAG_SHOW_WHEN_LOCKED` flag.

`FLAG_SHOW_WHEN_LOCKED` is a window flag of `WindowManager.LayoutParams` in Android framework. Setting this flag in your application lets your application view to take control over the Lock screen. You can set this flag when it is launched and clear the flag when it exits.

**3.3.4 Modifying the AndroidManifest.xml file****Presenting the Application on Settings**

For the QuickCircle application to be listed in settings, you need to modify `AndroidManifest.xml` file as shown in below.

```
<activity
>
```

```

        <intent-filter>
            <action android:name="com.lge.quickcover" />
        ..
    </intent-filter>
</activity>

```

### Removing the Title bar

The title bar needs to be removed in order to fit your application to QuickCircle window.

```

<activity
    android:theme="@android:style/Theme.NoTitleBar.Fullscreen"
>
</activity>

```

### Fixing the Orientation

The orientation has to be fixed to the portrait direction.

```

<activity
    android:screenOrientation="portrait"
    ..
</activity>

```

### Adding an Application Icon

For the QuickCircle window, an application icon needs to be added to the activity.

```

<activity
    android:icon="@drawable/icon"
    ..
</activity>

```

## 3.4 API References

LG QCircle SDK does not provide a specific library or methods. Instead, it provides LG QCircle Intents which are broadcasted intents. Information for LG QCircle Intents in LG QCircle APIs is provided here.

### 3.4.1 LG QCircle Intents

Constants	
<a href="#"><code>com.lge.android.intent.action.ACCESSORY_COVER_EVENT</code></a>	A broadcast intent for informing state changes of the front cover
<a href="#"><code>com.lge.intent.extra.ACCESSORY_COVER_STATE</code></a>	Extra data for the current state of the front cover

#### **`com.lge.android.intent.action.ACCESSORY_COVER_EVENT`**

Broadcast Action: A sticky broadcast intent to inform the front cover state changes: from open to close; closed to open.

The broadcast intent contains extra data:

- `ACCESSORY_COVER_STATE`: Contains the current state of the front cover

#### **`com.lge.intent.extra.ACCESSORY_COVER_STATE`**

Contains an integer value indicating the current state of the front cover.

- 0: The front cover is open.
- 1: The front cover is closed.



## 4 Development Tips

---

This chapter provides useful tips for reference while developing the QCircle application using LG QCircle SDK.

.

## Emulating a Broadcast Intent via ADB Tool

You can conveniently test your application by emulating a Broadcast Intent via ADB tool.

1. Type 'adb devices' and see your device is connected.
2. If your device is connected, type 'adb shell'
3. Sending a broadcast intent for the cover action as follow:

To emulate cover-closed state:

```
Shell> am broadcast -a com.lge.android.intent.action.ACCESSORY_COVER_EVENT --ei  
com.lge.intent.extra.ACCESSORY_COVER_STATE 0
```

To emulate cover-open state:

```
Shell> am broadcast -a com.lge.android.intent.action.ACCESSORY_COVER_EVENT --ei  
com.lge.intent.extra.ACCESSORY_COVER_STATE 1
```

---

### Note

You can find ADB tool from **< root directory of your android-sdk installation>/<platform-tools>**. To find the more explanation, refer to [Android developer site](#).

---

## Excluding QCircle Activity from Recent Used Apps

If you created a separate activity for QuickCircle window, the activity should be excluded from the "recent used app" list when the cover is open. You can exclude the activity from the list by adding an attribute to AndroidManifest.xml file. The value attributed should be `excludeFromRecents`. You can set it to `true` and add the attribute to corresponding activity as shown below.

```
<activity
...
    android:excludeFromRecents="true"
...
</activity>
```

---

### Note

For more details *about excludeFromRecents*, refer to [Android developer site](#).

---

# 5 UI Design & UX Scenario Guide

---

This chapter provides explanations of UI Design and UX Scenario for LG Quick Circle feature. The chapter consists of two parts: the UI design guide and the UX scenario guide.

## Contents

### [5.1 UI Design Guide](#)

This section provides a guide for UI design of QCircle applications.

### [5.2 UX Scenario Guide](#)

This section provides explanations of basic UX flows of LG QCircle feature.

## 5.1 UI Design Guide

This section provides the design guide for QCircle applications. The guide provides the overview for Quick Circle UIs and the specification for QCircle application UIs.

### 5.1.1 Overview

The Quick Circle framework consists of UI components: menu screen; lock screen; settings; applications.

#### Lock Screen

The lock screen is a starting point of UI flow. When users press a power key or knock on the Quick Circle window of the closed cover, the lock screen is shown. Quick Circle UI provides more than 7 lock screens as default. Users can select a preferred lock screen.



Figure 5.1 Lock Screen

#### Menu Screen

The menu screen provides access to QCircle applications. The menu screen displays up to six QCircle applications. App icons for QCircle applications are in a circular shape. Users can select apps from Settings and selected apps will be displayed on the Menu screen. Users can tap an app icon to launch the corresponding app without opening the cover.

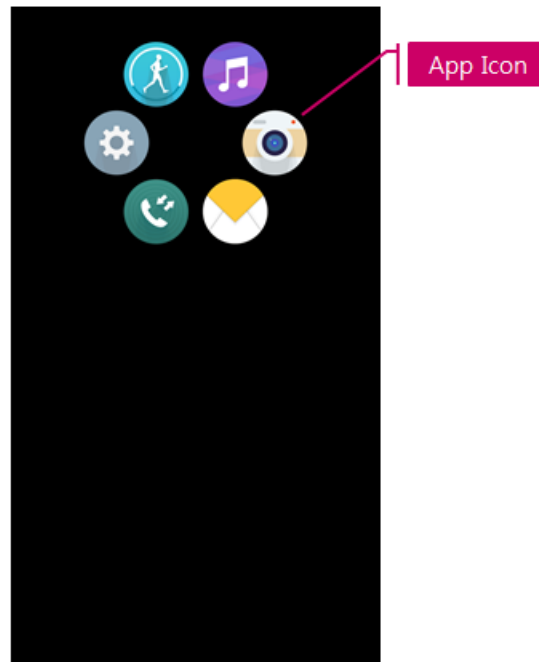


Figure 5.2 Menu Screen

### Common QCircle Application UI

QCircle applications have layouts to be shown in the circular window. QCircle applications typically use a back button, title area and functional buttons in circular shapes.

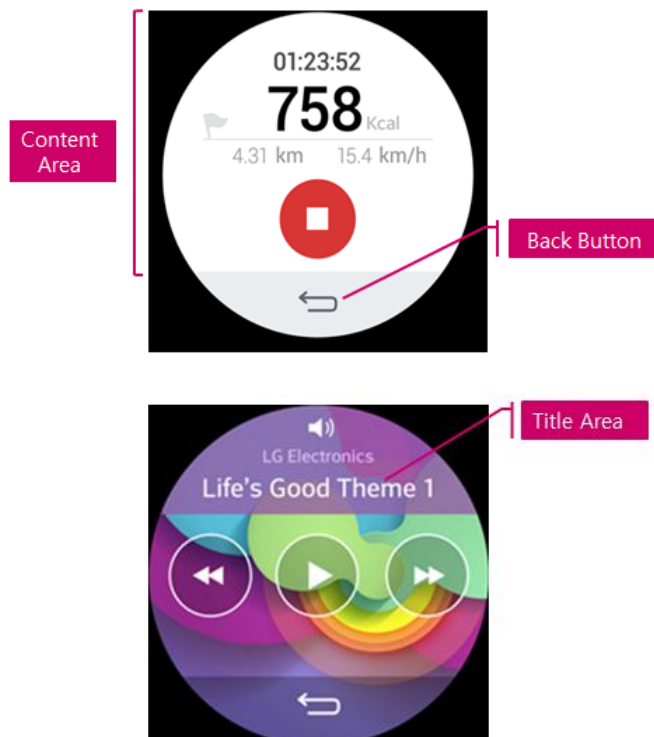


Figure 5.3 Application UI

### 5.1.2 Application UI

QCircle apps have a circular view. The circular view is a background for other UI components so that called circle background. To be shown in the QCircle window, all UI components are placed within the circular background. Figure 5.4 shows the description for each basic component of QCircle application.



Figure 5.4 UI components

#### Circle background

Circle background is an outline for the cover view. The diameter of the circle background is different in device models. The typical diameter value is based on LG G3 model.

Device Model	Circle Diameter (px)
LG G3	1046

#### NOTE

The diameter value is given in pixel. Since Quick Circle feature is used in some models with different screen densities physical display sizes, the pixel value of the diameter is stored natively in device software.

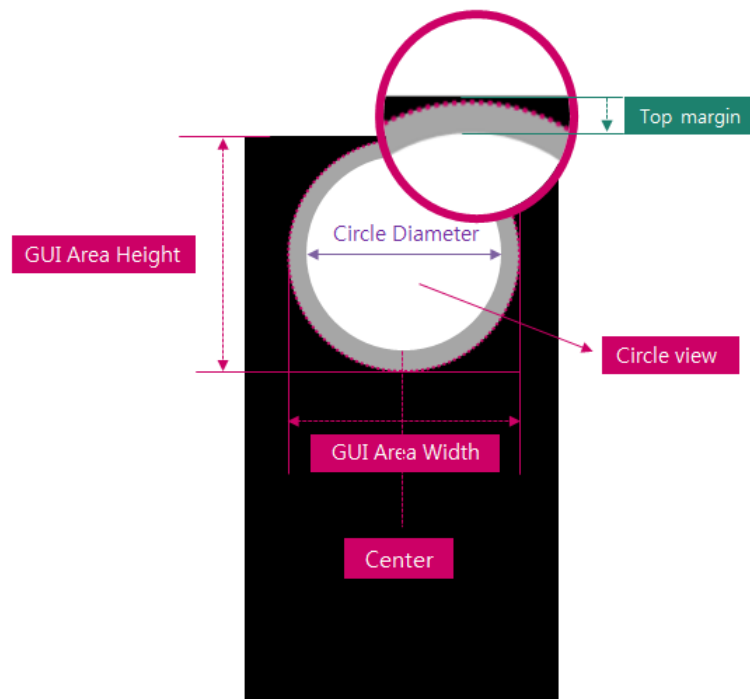


Figure 5.5 Circle Background

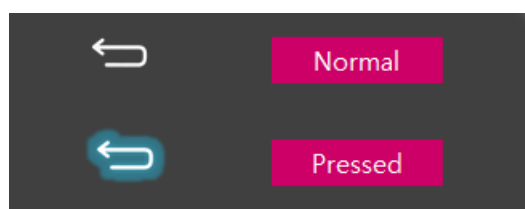
### Back button

The back button exits the app and takes users back to the menu screen. QCircle applications need to provide a software back button allows users to exit the app without opening the cover. The back button is placed on the cover view. Back button can be designed as either basic or borderless styles. Please, give users proper touch feedback to indicate the button is pressed.

#### Basic Style



#### Borderless Style



#### NOTE

Button color should be distinctive from the background.





Figure 5.6 Weighted portion of the back button area

### Title

The title of the application or the page can be placed in the top of the cover view. Fig 4 shows weighted portion for the title area. The portion of title area is a twenty percent of the circle diameter. Text is placed 5dp up from the border.

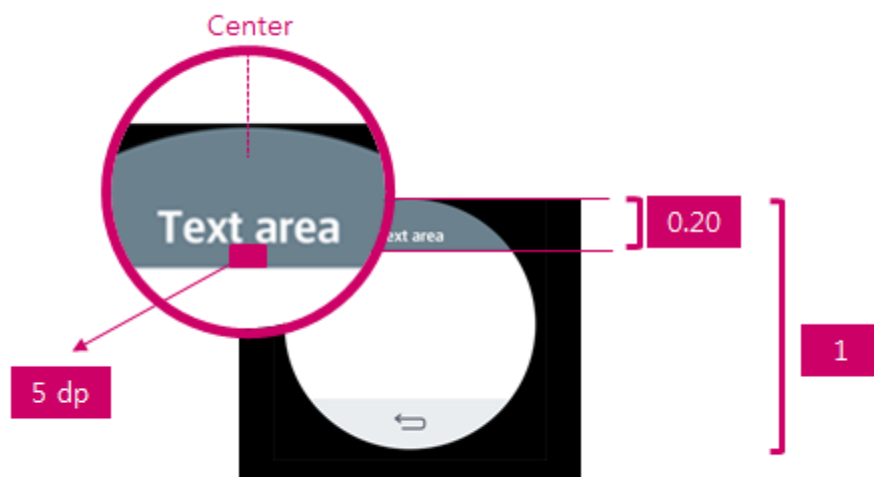
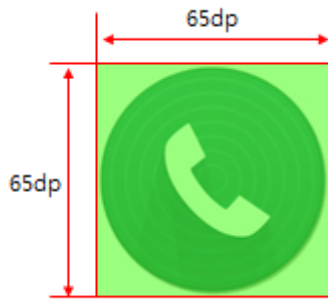


Figure 5.7 Weighted portion of the title area

### 5.1.3 Iconography

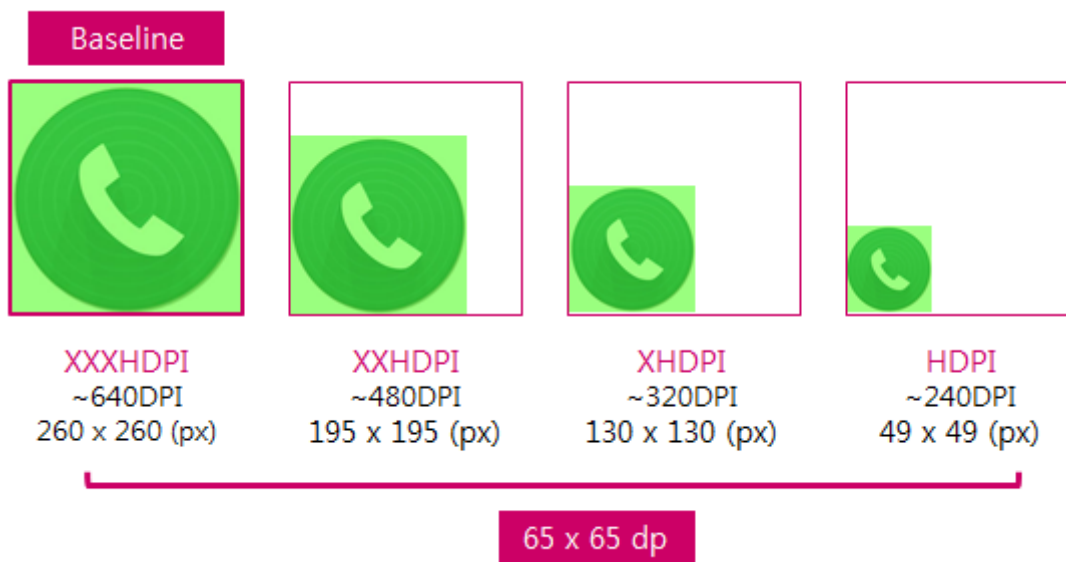
QCIRCLE application needs an app icon to be shown in the menu screen. Having a circular app icon is strongly recommended for achieving unified look and feel with other QCIRCLE applications. The format and the size for the icon are described below.



- **Format:** PNG
- The size includes margin and stroke of the icon.

**NOTE**

For multiple screens, provide icon images for different densities.



For more details about using dp units, please, refer to [Android developer site](#).

### 5.1.4 Supporting Multiple Screens

LG Quick Circle feature will be available in the more range of devices. Devices supporting the Quick Circle feature offer different sizes and densities. By giving flexibility in your design, your app can be clearly shown from lower density to high density devices.

Device Model	Screen size in inch	Pixel	Resolution
LG G3, LG G3 Cat.6	5.5"	1440 X 2560	QHD
LG G3 Beat	5.0"	720 X 1280	HD
Upcoming models	-	-	FHD

### Supporting Different Screen Densities

To support wide range of screen densities, please provide image resources for each density.

Images can be generated for each density using the following size scale:

- **xxxhdpi:** 1.0x (baseline)
- **xxhdpi:** 0.75x
- **xhdpi:** 0.5x
- **hdpi:** 0.375x
- **mdpi:** 0.25x

For example, if you generate 1200 x 1200 image for xxxhdpi devices, you should generate the same resource in 900 x 900 for xxhdpi.

Images are placed in the appropriate subdirectory via res/:

```
MyQCircleProject/
  res/
    drawable-xxxhdpi/
      circlebackground.png
    drawable-xxhdpi/
      circlebackground.png
    drawable-xhdpi/
      circlebackground.png
    drawable-hdpi/
      circlebackground.png
    drawable-mdpi/
      circlebackground.png
```

## Setting the Layout Dynamically in the Source

### Circle Background

Even though scaled images for different densities provides, for some models, the size of the circle view is not fully generalized. The circle view is properly set by setting its size and the margin dynamically in the code.

For example:

```
//Get the circle diameter dynamically
int id = getResources().getIdentifier("config_circle_diameter", "dimen",
"com.lge.internal");
int circleDiameter = getResources().getDimensionPixelSize(id);

//Set layout size same as a circle window size
layoutParam.width = circleDiameter;
layoutParam.height = circleDiameter;
```

### Back button & Title

The portion of back button and the title area are also set by codes. The height of a back button is 0.23 of the circle diameter and the portion of title area is 0.20 of the circle diameter.

For example:

```
//Set Back button height
```

```
int buttonAreaHeight = (int)(circleDiameter * 0.23);  
  
// add a button into the bottom of the circle layout  
RelativeLayout.LayoutParams params = new  
RelativeLayout.LayoutParams(circleDiameter, buttonAreaHeight);
```

## 5.2 UX Scenario Guide

This section provides an UX scenario guide for developers to understand basic UX flows provided from Quick Circle framework.

### 5.2.1 Overview

The QR Color Change consists of the following files:

Quick Circle framework provides unique UX through a decent circular window on the front. Users can access application directly through the window and select up to five applications to be displayed.

Basically, Quick Circle UX flow is a sequence of screens; lock screen, menu screen and application.

The basic UX flows are listed here:

- Cover closed or opened
- Application Launching flow
- Back to the menu screen
- Quick Circle Settings

### Quick Circle UX Principles

#### One Tap UX

Use touch interactions that are simple such as tapping once and flicking.

Minimize the number of complex interactions.

#### Accessible & Simple


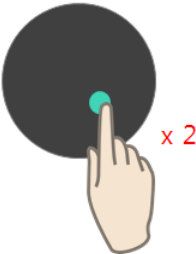

The gem of Quick Circle UX is using functions without opening the cover.

Design UX that allows accessing functions directly without opening the cover.

Make functions that are completed in few steps.

### 5.2.2 Standard Interaction

Quick Circle UX mainly uses touch interaction such as tapping, knock on and flicking. Table 1 explains about frequent touch interactions used in Quick Circle UX.

Touch Interaction	Action	Typical use case
Tap 	Tap once on a button or an icon.	<ul style="list-style-type: none"> <li>Opens, selects, launches an item you tap on.</li> </ul>
Knock on 	Double tap on the screen.	<ul style="list-style-type: none"> <li>Wakes or shutting off the screen instead of using Power button.</li> </ul>
flick 	Slide on the screen.	<ul style="list-style-type: none"> <li>Moves backward or forward between screens.</li> </ul>

### 5.2.3 Basic UX Flows

#### Cover Closed

Once the cover is closed, the lock screen is shown. From the lock screen, the user can move to the menu to browse applications or leave the phone to sleep after 6 seconds. When the screen detects no action for 6 seconds after the QCircle application is launched, the phone will sleep as well.



Figure 5.8 UX flow of the cover closed

### Cover Open

When the user opens the cover, the lock screen or the home screen will be loaded on full screen.



Figure 5.9 UX flow of the cover open

## Launching Application

For Quick Circle UX, the lock screen is a starting point and also a closing point. When the phone is in the sleep mode with LCD off and the user awakes the phone, the lock screen is always shown first. From the lock screen, users can access to the menu screen.

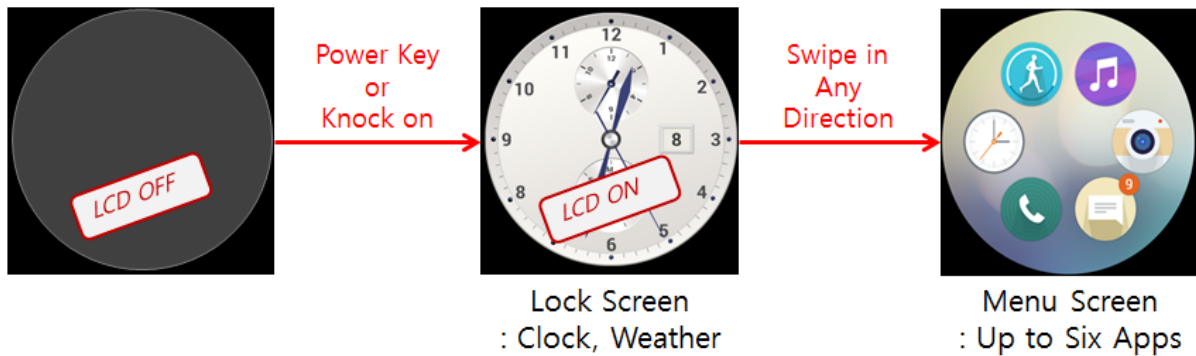


Figure 5.10 Navigating to menu screen

### NOTE

The lock screen is a starting point of the entire UX flow. For example, the user launches a QCircle application and opens the cover. When the user closes the cover again, the lock screen is shown. This scenario keeps the system safe from any incorrect operations.

Next, the user can access to the application by tapping an app icon. Also, the user can back to the menu by tapping a back button.



Figure 5.11 UX flow of launching application

When the user launches the QCircle application and opens the cover, the corresponding application is launched if available.





Figure 5.12 Launching full screen activity

### Back to Lock Screen

Users can simple back to the Lock screen from the application or the menu screen anytime. The flow to back to the lock screen is described below.

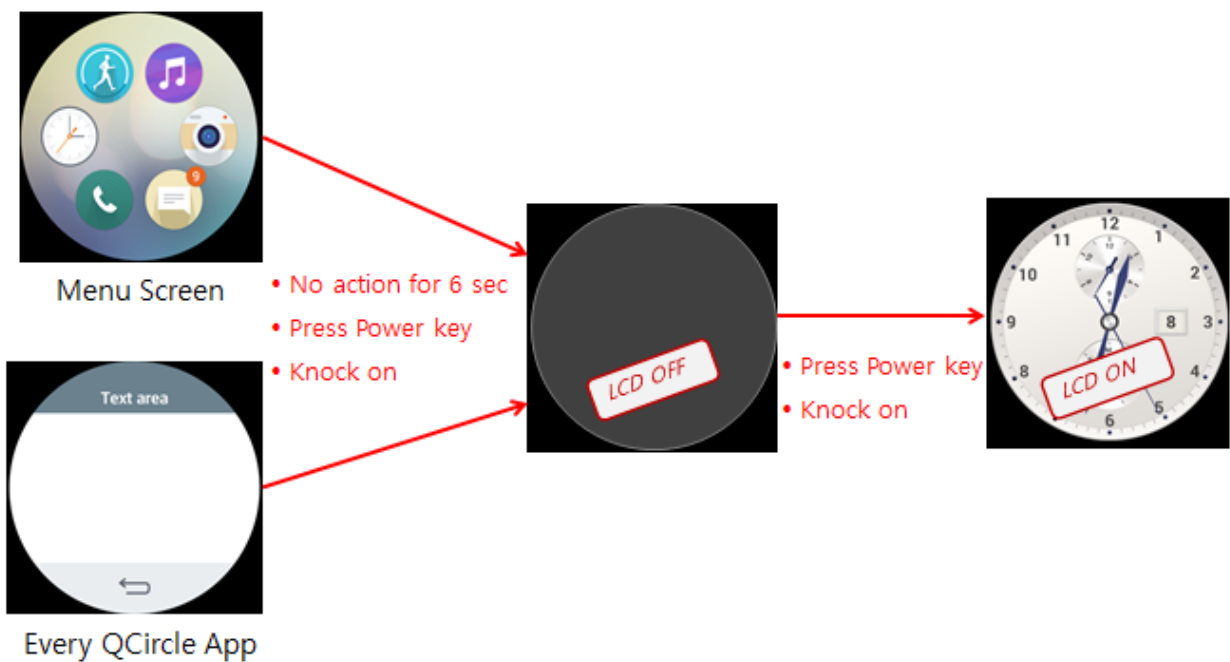




Figure 5.13 Back to lock screen

## Settings

When a Quick Circle application is installed, users can find the application via Quick Circle case -> select apps. All Quick Circle applications are listed in select apps so that the user can select an application to be displayed in the menu screen.

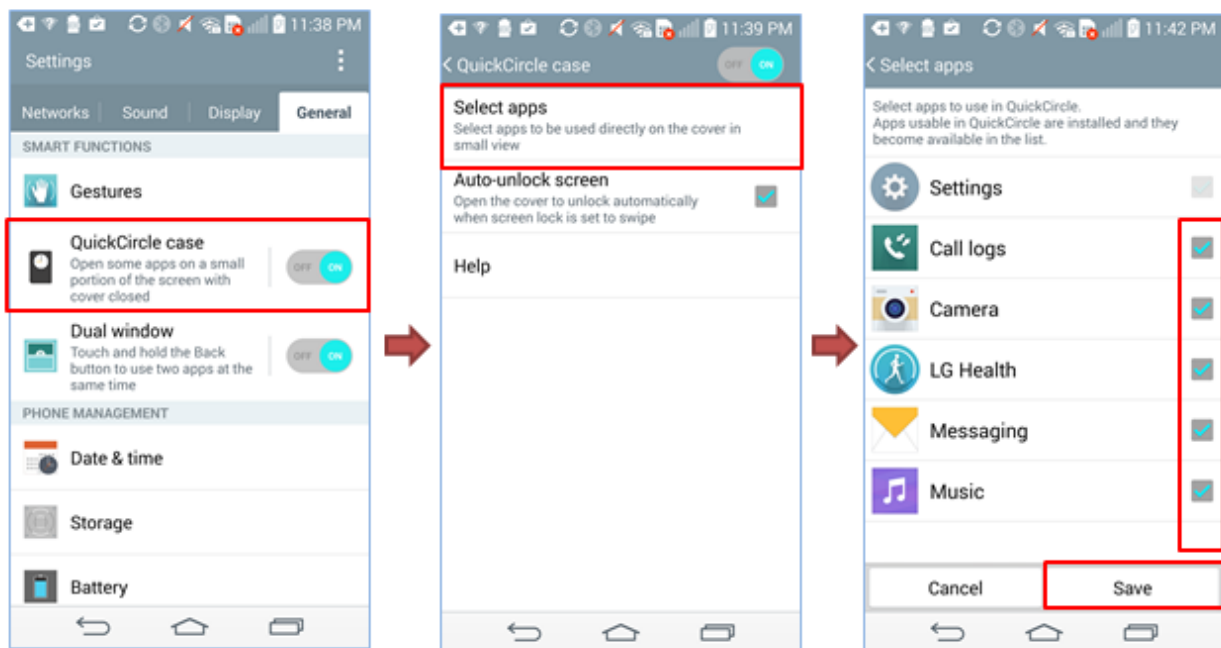


Figure 5.14 Setting QCircle applications

# 6 Sample Tutorials

---

This chapter describes the given sample application in the LG QCircle SDK and provides its code analysis.

## Contents

### [5.1 Sample Application Overview](#)

This chapter introduces the sample application included in the LG QCircle SDK

### [5.2 Sample Code Analysis](#)

This section provides the analysis of the source code of the sample application. You can learn how to develop a QCircle application by using LG QCircle SDK.

## 6.1 Sample Application Overview

The LG QCircle SDK includes a sample application with its source codes. The sample application uses LG QCircle APIs and runs in the QuickCircle window once the cover is closed.

This section describes the functions of the sample application and the LG QCircle APIs used in the sample application. Also, this section tells you how to import the sample application into your workspace.

### 6.1.1 Introduction to the Sample Application

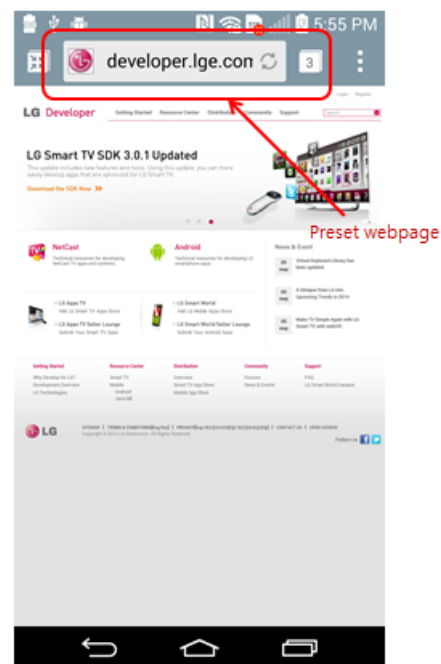
The LG QCircle SDK includes a sample application with its source codes. The sample application uses LG QCircle APIs and runs in the QuickCircle window when the cover is closed.

#### QR Color Change

The sample application below shows how the application can change depending on the cover's state. When the cover is closed, the sample application will display a QR code through the QuickCircle window as shown in Fig. 5-1, and users can change its color by pressing either of the arrow buttons. Once the cover is opened, the application will direct users to the website stored in the QR code, in this case it is LG developer's website.



**Figure 6.1.** Sample application screen in the QuickCircle Window



**Figure 6.2.** Sample application screen in full screen

The sample application has two activities and three buttons. Here is a description of the sample application layout:

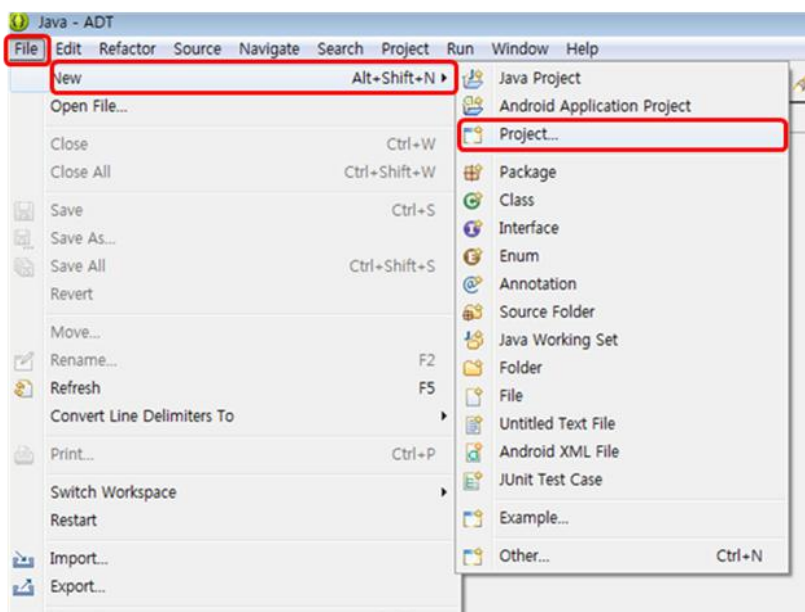
- **circlemainView:** This is an application view shown in the QuickCircle when the QR Color Change is launched with the front cover closed.
- **iv:** This is an image view includes a QR code.

- **rightSideBtn**: This is an arrow button on the right side. You can switch the color of the QR code by pressing this button.
- **leftSideBtn**: This is an arrow button on the left side. You can switch the color of the QR code by pressing this button.
- **backBtn** : This is a button to finish the application.

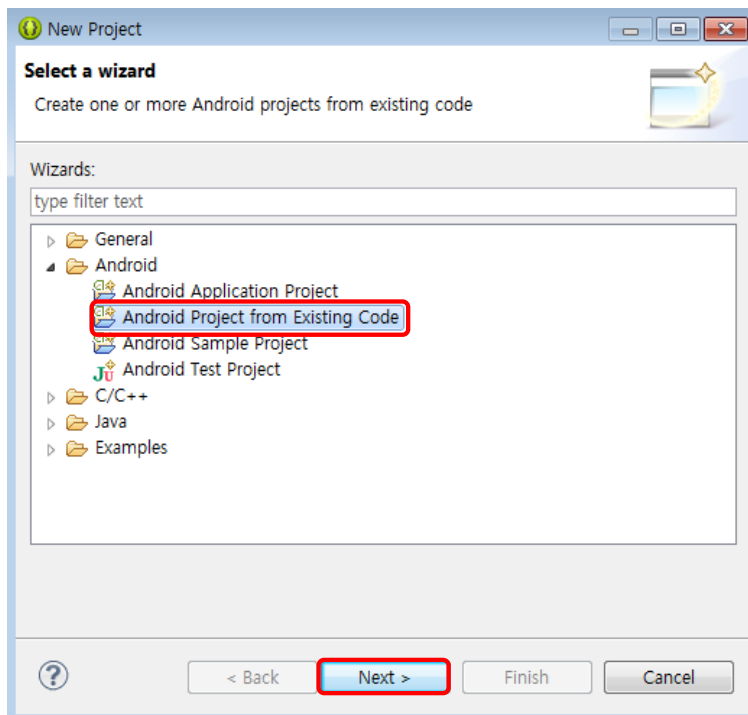
### 6.1.2 Importing the Sample Application

In order to execute the sample application, you should follow the steps below.

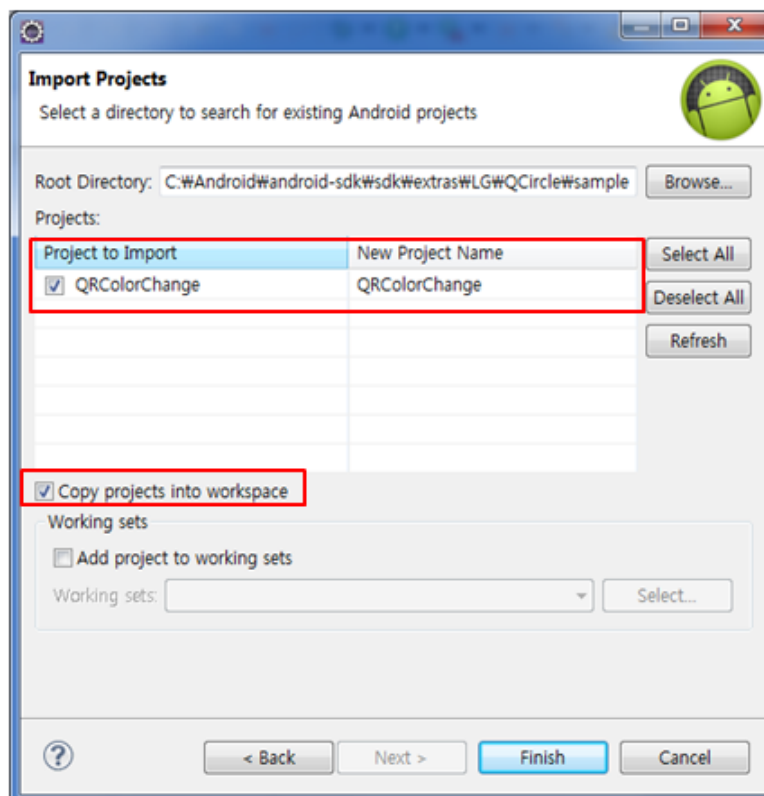
1. Start Eclipse
2. Select **FILE > New > Project**.



3. Select **Android > Android Project from Existing Code**, and click **Next**.



4. Click **Browse...** and select the directory you installed the LG QCircle SDK.
5. Check the samples to import and '**Copy projects into workspace**' box, click **Finish**



6. The project 'QRColorChange' will appear.



7. The sample project is now ready to run.

## 6.2 Sample Code Analysis

This section provides analysis of the sample application codes to show how to develop QCircle applications.

### 6.2.1 QR Color Change

The QR Color Change consists of the following files:

**Table 6-1** QR Color Change files

Files	Description
src/com/sample/qcircle/qccc/QCircleActivity.java	Main activity file
src/com/sample/qcircle/qccc/FullScreenActivity.java	A file for the full screen activity
res/layout/activity_qcircle.xml	Layout for the main activity
res/layout/activity_fullscreen.xml	Layout for the full screen activity

#### Note

In this section, only source files written in Java are explained. A detailed description of the layout files can be found on the [Android Developer site](#).

### Getting Ready

The codes explained here is in QCircleActivity.java.

#### Retrieving Views

The sample application retrieves a view to be shown in the QuickCircle window.

```
public class QCircleActivity extends Activity{
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_qcircle);
        final View circlemainView = findViewById(R.id.cover_main_view);
        ..
    }
}
```

### Managing a Broadcast Receiver

#### Adding an Intent Filter

A filter adds the `ACTION_ACCESSORY_COVER_EVENT` to receive QCircle intents.

```
private void registerIntentReceiver() {
    IntentFilter filter = new IntentFilter();
    filter.addAction(ACTION_ACCESSORY_COVER_EVENT);
    ..
}
```



### Creating a Broadcast Receiver

A broadcast receiver is created to receive the QCircle intents and handle the cover events.

```
private BroadcastReceiver mIntentReceiver = new BroadcastReceiver() {
    ..
};
```

### Registering the Broadcast Receiver

The broadcast receiver is registered with the system.

```
private void registerIntentReceiver() {
    mContext.registerReceiver(mIntentReceiver, filter);
    ..
}
```

### Unregistering the Broadcast Receiver

The broadcast receiver is unregistered with the system.

```
protected void onDestroy() {
    super.onDestroy();
    mContext.unregisterReceiver(mIntentReceiver);
}
```

## Handling a Cover Event

When *QCircleActivity* receives QCircle intent, codes for the related action are implemented in broadcast receiver. The *QCircleActivity* adjusts the *circlemainView* to view in the QuickCircle window when the cover is closed. *QCircleActivity* calls *FullScreenActivity* for the front cover open.

```
private BroadcastReceiver mIntentReceiver = new BroadcastReceiver() {

    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction(); ①
        if (ACTION_ACCESSORY_COVER_EVENT.equals(action)) { ②
            mQuickCoverState = intent.getIntExtra(EXTRA_ACCESSORY_COVER_STATE,
                EXTRA_ACCESSORY_COVER_OPENED); ③

            if (mQuickCoverState == EXTRA_ACCESSORY_COVER_CLOSED) { // closed
                setQuickCircleWindowParam(); ④
            }
            else if (mQuickCoverState == EXTRA_ACCESSORY_COVER_OPENED) { // opened
                Intent callFullscreen = new Intent(mContext, FullScreenActivity.class);
                startActivity(callFullscreen); ⑤
                QCircleActivity.this.finish(); ⑥
            }
        }
    }
};
```

- ① Receives a QCircle intent.
- ② Detects the event of the front cover.

- ③ If the front cover is closed, the window parameters are set for the QuickCircle window.
- ④ When the front cover is open, the activity calls the full screen activity.
- ⑤ The activity for the QuickCircle window view is finished.

## Adding an Window Flag

When the sample application is launched with the cover closed, setting FLAG\_SHOW\_WHEN\_LOCKED flag is necessary to take precedence over the lock screen.

For the flags, [FLAG\\_FULLSCREEN](#) and [FLAG\\_KEEP\\_SCREEN\\_ON](#), please refer to the link directing to Android Developer's site.

```
private void setQuickCircleWindowParam() {
    win = getWindow();
    if (win != null) {
        win.addFlags(WindowManager.LayoutParams.FLAG_SHOW_WHEN_LOCKED
            | WindowManager.LayoutParams.FLAG_FULLSCREEN
            | WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
    }
}
```

## Getting Cover Information

To fit the application layout into the QuickCircle window, the related parameters are loaded.

```
void initializeCoverInformationFromDB() {

    quickCircleEnabled = Settings.Global.getInt(contentResolver,
        CASESETTINGS_QUICKCIRCLE_ENABLE, 0) == 1 ? true : false;

    quickCaseType = Settings.Global.getInt(contentResolver,
        "cover_type", 0/*default value*/); ①

    int id = getResources().getIdentifier("config_circle_window_width",
        "dimen", "com.lge.internal"); ②
    circleWidth = getResources().getDimensionPixelSize(id);

    id = getResources().getIdentifier("config_cover_window_height", "dimen",
        "com.lge.internal"); ③
    circleHeight = getResources().getDimensionPixelSize(id);

    id = getResources().getIdentifier("config_circle_window_x_pos", "dimen",
        "com.lge.internal"); ④
    circleXpos = getResources().getDimensionPixelSize(id);

    id = getResources().getIdentifier("config_circle_window_y_pos", "dimen",
        "com.lge.internal"); ⑤
    circleYpos = getResources().getDimensionPixelSize(id);

    id = getResources().getIdentifier("config_circle_diameter", "dimen",
        "com.lge.internal"); ⑥
    circleDiameter = getResources().getDimensionPixelSize(id);
}
```

- ① Gets a type of the installed case.
- ② Gets the width of the QuickCircle window.

- ③ Gets the height of the QuickCircle window.
- ④ Gets the horizontal position of the QuickCircle window.
- ⑤ Gets the vertical position of the QuickCircle window.
- ⑥ Gets the diameter of the QuickCircle window.

## Cropping the Application Layout

Sample application adjusts the size of layout for showing on the QuickCircle window when the cover is closed.

```
void setCircleLayoutParam(View view) {
    RelativeLayout layout = (RelativeLayout) view;
    RelativeLayout.LayoutParams layoutParam = (RelativeLayout.LayoutParams)
    layout.getLayoutParams();
    layoutParam.width = circleDiameter; ①
    layoutParam.height = circleDiameter; ②

    if (circleXpos < 0) {
        layoutParam.addRule(RelativeLayout.CENTER_HORIZONTAL,
        RelativeLayout.TRUE); ③
    }
    else {
        layoutParam.leftMargin = circleXpos;
    }
    layoutParam.topMargin = circleYpos + (circleHeight -
    circleDiameter)/2; ④
    layout.setLayoutParams(layoutParam);
}
```

- ① Sets the width of the view layout.
- ② Sets the height of the view layout.
- ③ Sets the horizontal position of the view layout. The position will be set to the center when the x position value is negative.
- ④ Sets the vertical position of the view layout. The position is set to the offset.

## Modifying the AndroidManifest.xml file

Android Manifest file is modified to present the sample application in Settings and to properly set the layout for the QuickCircle window.

```
<activity
    android:name="com.sample.qcircle.qrcc.QCircleActivity"
    android:excludeFromRecents="true" ①
    android:screenOrientation="portrait" ②
    android:theme="@android:style/Theme.NoTitleBar.Fullscreen" ③
>
<intent-filter>
    <action android:name="com.lge.quickcover" /> ④
    ..
</intent-filter>
</activity>
```

- ① Excludes the *QCircleActivity* from recent used apps.
- ② Fixes the orientation to portrait for the *QCircleActivity*.
- ③ For *circlemainView*, the title bar is removed to set to the proper position.
- ④ Presents QR Color Change in Settings

