MORE Screen Lock Controller SDK

Version 0.16.08.15

2016.08.15

Table of Contents

1. Introduction

1.1. Overview

MORE Screen Lock Controller SDK 主要提供 Smart Mobile Device 更快速與簡潔的開發模組，APP 開發人員只要將 MORE Screen Lock Controller SDK 加入到自己的 APP 專案裡,透過 API 的呼叫即可以裝置管理員身分鎖定螢幕使用權，目前提供 Android 版本 SDK Library。

2. Service

2.1. Screen Lock Controller SDK

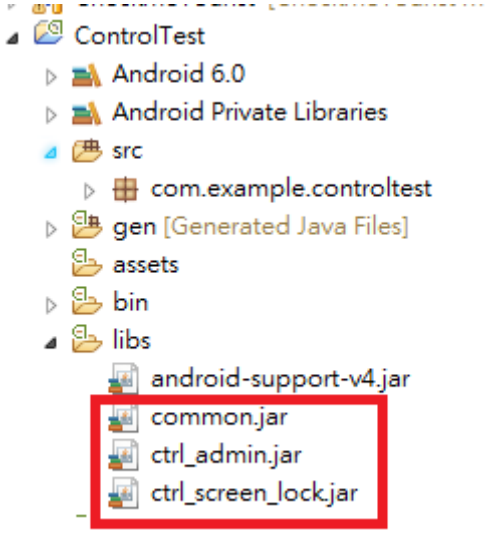
Screen Lock Controller SDK 主要提供給開發者以裝置管理員身分來鎖定智慧型裝置之鎖定螢幕控制權。

2.2. Use SDK Library

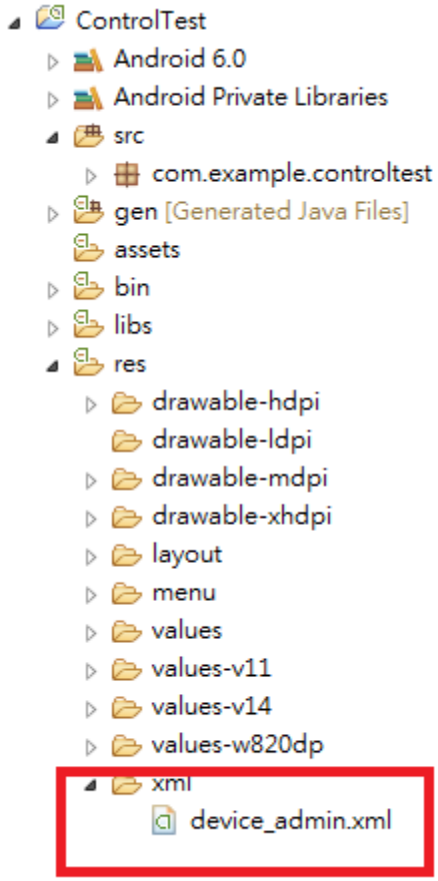
在使用 Screen Lock Controller SDK 前，需要先透過 Device Administrator 取得控制螢幕鎖定的權限，再加以 resetPassword() 跟 lockNow() 控制 。

使用步驟：

1. 先將以下 jar 檔加入到 Android 開發專案的 libs 資料夾



2. 在 res 資料夾中的 xml 資料夾裡新增一個 xml 檔，例如檔名為 device_admin.xml



3. 在device_admin.xml裡輸入以下內容，<force-lock />為鎖定螢幕，<reset-password /> 為變更螢幕解鎖密碼，<limit-password />為設定密碼規則等依據的參數。

```
<?xml version="1.0" encoding="utf-8"?>
<device-admin xmlns:android="http://schemas.android.com/apk/res/android" >

    <uses-policies>
        <force-lock />

        <reset-password />

        <limit-password />
    </uses-policies>

</device-admin>
```

4. 在 AndroidManifest.xml 的<application 裡新增以下幾行，注意一點的是 android:resource="@xml/*檔名名稱*" 中的檔名名稱須對應到第 2.步驟新增的檔

名名稱

```
<receiver android:name="sdk.ideas.ctrl.admin.DeviceAdministratorReceiver"
          android:permission="android.permission.BIND_DEVICE_ADMIN" >
          <meta-data
              android:name="android.app.device_admin"
              android:resource="@xml/device_admin" >
          </meta-data>

          <intent-filter>
              <action android:name="android.app.action.DEVICE_ADMIN_ENABLED" />
              <action android:name="android.app.action.DEVICE_ADMIN_DISABLED"/>
          </intent-filter>
      </receiver>
```

5. 匯入以下 library

```
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;
import java.util.HashMap;
import sdk.ideas.common.CtrlType;
import sdk.ideas.common.OnCallbackResult;
import sdk.ideas.common.ResponseCode;
import sdk.ideas.ctrl.admin.DeviceAdminHandler;
import sdk.ideas.ctrl.lock.LockHandler;
```

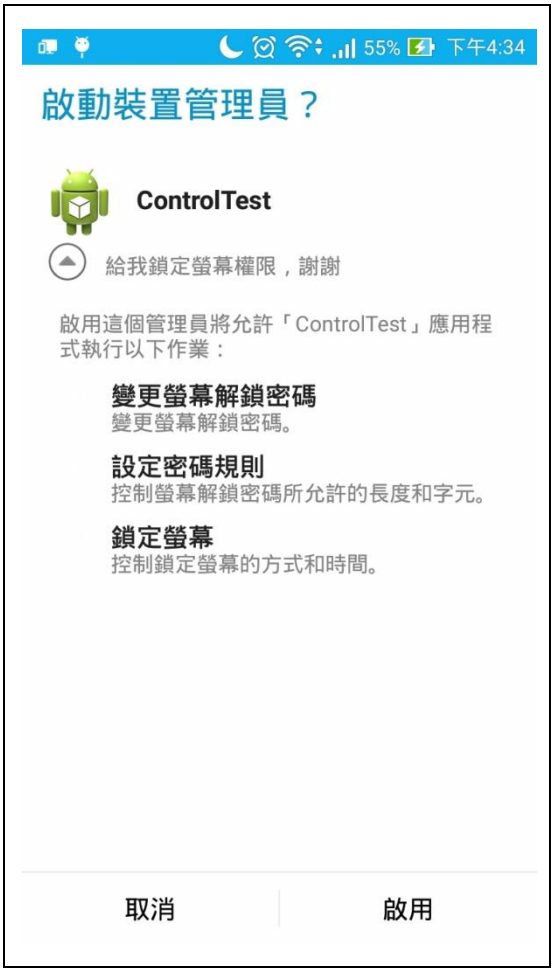
6. 設定 DeviceAdmin Handler 及 Callback 來取得權限

```
DeviceAdminHandler mDeviceAdminHandler = new DeviceAdminHandler(this);
mDeviceAdminHandler.setOnCallbackResultListener(new OnCallbackResult()
        {
            @Override
            public void onCallbackResult(int result, int what, int from,
HashMap<String, String> message)
            {
                switch(from)
                {
                case  ResponseCode.METHOD_ADMIN_CREATE_POLICY:

                    if (result == ResponseCode.ERR_SUCCESS )
                    {
                        Log.i("DeviceAdminSample" , "Administration
enabled!");
                    //取得到權限後，才可下一些命令
                    //ex.  mLockHandler.setSceenLockPassword("test");
                    }
                    else
                    {
                        Log.i("DeviceAdminSample" , "Administration
enable FAILED!");
                    }
                    break;
                case  ResponseCode.METHOD_ADMIN_REMOVE_POLICY:
                    if(result == ResponseCode.ERR_SUCCESS)
                    {
                        Log.i("DeviceAdminSample" ,"Administration
remove success!");
                    }
                    else
                    {
                        Log.i("DeviceAdminSample" , "Administration
remove FAILED!");
                    }
                    break;
                }
            }
        });

mDeviceAdminHandler.createPolicy("給我鎖定螢幕權限，謝謝");
```

7. 在Activity設定onActivityResult回傳function給DeviceAdmin Handler

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
    super.onActivityResult(requestCode, resultCode, data);
    switch (requestCode)
    {
    case CtrlType.REQUEST_CODE_ENABLE_ADMIN:
        mDeviceAdminHandler.onActivityResult(requestCode, resultCode, data);
    break;
    }
}
```

8. 如果以上步驟皆沒問題的話，在第一次執行程式時會跳出以下視窗。



9. 設定 Screen Lock Handler

```
LockHandler mLockHandler = new LockHandler(mDeviceAdminHandler.getPolicyData());
```

10. 本SDK 提供Handler 和Callback Listener方法來去知道程式是否有成功執行，開發者可以以自己喜好去做設定，以下是Handler 方法範例

1) 先宣告 Handler

```
    private Handler theHandler = new Handler()
    {
        @Override
        public void handleMessage(Message msg)
        {
            switch (msg.what)
            {
            case CtrlType.MSG_RESPONSE_LOCK_HANDLER:
                Log.i("handler response: ", "Result: "  +
String.valueOf(msg.arg1) + " From: "
                        + String.valueOf(msg.arg2) + " Message: " +
((HashMap<String,String>)msg.obj).get("message"));
                break;
            default:
                break;
            }
        }
    };
```

2) 設定此 handler

```
mLockHandler.setHandler(theHandler);
```

11. Callback Listener 方式

```
mLockHandler.setOnCallbackResultListener(new OnCallbackResult()
        {

            @Override
            public void onCallbackResult(int result, int what, int
from, HashMap<String, String> message)
            {
                Log.i("Callback Listener"," Result : " +
String.valueOf(result) +" What: "+String.valueOf(what)+ " From : "
                        + String.valueOf(from) + " Message : " +
message.get("message"));

            }});
```

12. 當 device admin callback 回來後，確定有拿到控制權限就能呼叫

```
mLockHandler.setSceenLockPassword("test");
mLockHandler.lockSceenNow();
```

2.3. Device Admin Function

Device Admin API	
Function	Description
DeviceAdminHandler(Context context)	建構元
void createPolicy(String addAdminExtraAppText)	新增控制權限，參數為跳出要求控制權限視窗時給使用者看得一些資訊及敘述，可 null

void onActivityResult(**int** requestCode, **int** resultCode, Intent data)	放置於 Activity 的 onActivityResult 位置，接收使用者在 2.2.8 圖的啟用或取消的資訊，requestCode 參考 3.2
PolicyData getPolicyData()	取得控制權限資訊，用於給需要權限的 contoller，注意，呼叫此 function 前需先執行 createPolicy (String addAdminExtraAppText)
boolean isActive()	判斷控制權限是否已啟用
void removePolicy()	移除控制權限
void setHandler(Handler handler)	設定 Handler，參數對應可參考 3.3
void setOnCallbackResultListener(OnCallbackResult listener)	設定 callback listener， 其參數參考 3.1

2.4. Screen Lock Function

Screen Lock Controller API	
Function	Description
LockHandler(PolicyData data)	建構元
void setSceenLockPassword(String password)	重新設定密碼，之前的密碼會直接被清除。
void lockSceenNow()	立即鎖定螢幕
void setHandler(Handler handler)	設定 Handler，參數對應可參考 3.3

void setOnCallbackResultListener(OnCallbackResult listener)	設定 callback listener，其參數參考 3.1

3. Parameter Definition

3.1. OnCallbackResult Function Parameter

result：

Value	Description
1	SUCCESS
0	UNKNOWN ERROR
-7	OS BUILD VERSION ERROR
-8	DEVICE ADMIN POLICY INACTIVE ERROR
-17	NO SPECIFY USE POLICY(沒有在xml裡加入 `<force-lock />` , `<limit-password />` `<reset-password />` 權限)
-34	PASSWORD FORMAT NOT SUPPORT ERROR

what：

Value	Description
1030	DEVICE ADMIN HANDLER
1040	SCREEN LOCK CONTROLLER HANDLER

from：

Value	Description
0	DEVICE ADMIN CREATE POLICY FUCTION
1	DEVICE ADMIN REMOVE POLICY FUCTION
0	RESET SCREEN LOCK PASSWORD FUCTION
1	LOCK SCREEN NOW FUCTION

message :

key	Description
“message”	其內容敘述執行成功或錯誤等詳細訊息

3.2. onActivityResult

requestCode：

Value	Description
3333	ENABLE ADMIN REQUEST CODE

3.3. Handler 與 Callback listener 參數對應

Handler	OnCallbackResult
`msg.what`	`int what`
`msg.arg1`	`int result`
`msg.arg2`	`int from`
`msg.obj`	`HashMap<String, String> message`