**热插拔SDCard弹出重启对话框**

**一、需求说明**

手机开机后，插入或拔出sdcard，弹出对话框要求用户重启手机。在弹出对话框期间，不有允许用户进行其他操作，只能点击对话框确定按钮，重启手机。

**二、解决思路**

（1）在frameworks/base/services/core/java/com/android/server/policy/PhoneWindowManager.java中的init()方法重置开机完成属性。

（2）在frameworks/base/services/core/java/com/android/server/policy/PhoneWindowManager.java中的systemReady（）方法中发送开机完成消息。接收到开机完成消息后，存储sdcard是否插入信息。

（3）在aliyunos/packages/apps/AliSystemUI/src/com/android/systemui/usb/StorageNotification.java中的onVolumeStateChanged（）方法中发送存储器状态改变消息。

（4）在接收到存储器状态改变消息后，根据是否是存储卡插入或拔出消息和sdcard上次的插入信息判断sdcard是否是真正插入或拔出。

**三、修改文件**

****

**四、具体解决步骤**

**（1）在frameworks/base/services/core/java/com/android/server/policy/PhoneWindowManager.java中的init()方法重置开机完成属性：**

@Override

public void init(Context context, IWindowManager windowManager,

WindowManagerFuncs windowManagerFuncs) {

...

///Xunhu: Mount or remove the SD card ejector restart dialog at 2016-11-07 14:11:54 by qty{{&&

///Description: 加载或移除SD卡弹出重启对话框(XUNHU\_QTY\_SDCARD\_CHANGED\_REBOOT\_DIALOG)

if (XunhuOption.XUNHU\_QTY\_SDCARD\_CHANGED\_REBOOT\_DIALOG) {

SystemProperties.set("persist.sys.boot\_completed", "false");

}

///&&}}

...

}

**（2）在frameworks/base/services/core/java/com/android/server/policy/PhoneWindowManager.java中的systemReady（）方法中发送开机完成消息。接收到开机完成消息后，存储sdcard是否插入信息:**

**1. 发送开机完成消息：**

@Override

public void systemReady() {

...

///Xunhu: Mount or remove the SD card ejector restart dialog at 2016-11-07 14:11:54 by qty{{&&

///Description: 加载或移除SD卡弹出重启对话框(XUNHU\_QTY\_SDCARD\_CHANGED\_REBOOT\_DIALOG)

if (XunhuOption.XUNHU\_QTY\_SDCARD\_CHANGED\_REBOOT\_DIALOG) {

if (!SystemProperties.getBoolean("persist.sys.boot\_completed", false)) {

Intent service = new Intent();

ComponentName name = new ComponentName("com.bsm\_wqy.validationtools", "com.sdcard.dialog.SDCardStateService");

service.setComponent(name);

service.putExtra("boot\_completed", true);

mContext.startService(service);

}

}

///&&}}

}

**2. 保存sdcard当前状态（packages/apps/ValidationTools/src/com/sdcard/dialog/SDCardStateService.java）：**

@Override

public int onStartCommand(Intent intent, int flags, int startId) {

if (intent != null) {

...

} else {

boolean bootCompleted = intent.getBooleanExtra(KEY\_BOOT\_COMPLETED, false);

Log.d(TAG, "onStartCommand=>boot completed: " + bootCompleted);

if (bootCompleted) {

mSharedPreferences.edit().putBoolean(KEY\_SDCARD\_EXIST, isSdcardExist()).commit();

SystemProperties.set("persist.sys.boot\_completed", "true");

}

mHandler.sendEmptyMessageDelayed(MSG\_FINISH, FINISH\_DELAYED);

}

}

return Service.START\_STICKY;

}

**（3）在aliyunos/packages/apps/AliSystemUI/src/com/android/systemui/usb/StorageNotification.java中的onVolumeStateChanged（）方法中发送存储器状态改变消息：**

@Override

public void onVolumeStateChanged(VolumeInfo vol, int oldState, int newState) {

///Xunhu: Mount or remove the SD card ejector restart dialog at 2016-11-07 14:11:54 by qty{{&&

///Description: 加载或移除SD卡弹出重启对话框(XUNHU\_QTY\_SDCARD\_CHANGED\_REBOOT\_DIALOG)

if (com.mediatek.common.featureoption.XunhuOption.XUNHU\_QTY\_SDCARD\_CHANGED\_REBOOT\_DIALOG) {

if (vol.getType() ==VolumeInfo.TYPE\_PUBLIC) {

android.util.Log.d("qty", "onVolumeStateChanged=>vol: " + vol + " old: " + oldState + " new: " + newState);

Intent service = new Intent();

ComponentName name = new ComponentName("com.bsm\_wqy.validationtools", "com.sdcard.dialog.SDCardStateService");

service.setComponent(name);

service.putExtra("sdcard\_state", vol.getState());

mContext.startService(service);

}

}

///&&}}

onVolumeStateChangedInternal(vol);

}

**（4）在接收到存储器状态改变消息后，根据是否是存储卡插入或拔出消息和sdcard上次的插入信息判断sdcard是否是真正插入或拔出（packages/apps/ValidationTools/src/com/sdcard/dialog/SDCardStateService.java）：**

@Override

public int onStartCommand(Intent intent, int flags, int startId) {

if (intent != null) {

mHandler.removeMessages(MSG\_FINISH);

mSdcardState = intent.getIntExtra(KEY\_SDCARD\_STATE, -1);

mSdcardExist = isSdcardExist();

boolean lastExsit = mSharedPreferences.getBoolean(KEY\_SDCARD\_EXIST, false);

Log.d(TAG, "onStartCommand=>state: " + mSdcardState + " exist: " + mSdcardExist + " lastExist: " + lastExsit);

if (SystemProperties.getBoolean("persist.sys.boot\_completed", false)) {

if (lastExsit != mSdcardExist) {

if (VolumeInfo.STATE\_MOUNTED == mSdcardState

|| VolumeInfo.STATE\_CHECKING == mSdcardState) {

if (!lastExsit) {

showDialog(TYPE\_MOUNTED);

mSharedPreferences.edit().putBoolean(KEY\_SDCARD\_EXIST, mSdcardExist).commit();

}

} else if (VolumeInfo.STATE\_REMOVED == mSdcardState

|| VolumeInfo.STATE\_BAD\_REMOVAL == mSdcardState

|| VolumeInfo.STATE\_EJECTING == mSdcardState) {

if (lastExsit) {

showDialog(TYPE\_EJECT);

mSharedPreferences.edit().putBoolean(KEY\_SDCARD\_EXIST, mSdcardExist).commit();

}

}

if (!mShowing) {

mHandler.sendEmptyMessageDelayed(MSG\_FINISH, FINISH\_DELAYED);

} else {

startForeground();

}

}

} else {

boolean bootCompleted = intent.getBooleanExtra(KEY\_BOOT\_COMPLETED, false);

Log.d(TAG, "onStartCommand=>boot completed: " + bootCompleted);

if (bootCompleted) {

mSharedPreferences.edit().putBoolean(KEY\_SDCARD\_EXIST, isSdcardExist()).commit();

SystemProperties.set("persist.sys.boot\_completed", "true");

}

mHandler.sendEmptyMessageDelayed(MSG\_FINISH, FINISH\_DELAYED);

}

}

return Service.START\_STICKY;

}