将QT消息转发到logCat

#ifndef QDEBUG2LOGCAT\_H

#define QDEBUG2LOGCAT\_H

ifdef ANDROID

void installLogcatMessageHandler(const char \*TAG);//如果定义了ANDROID宏那么就会定义这个消息过滤函数，不定义则什么都不做

#else

#define installLogcatMessageHandler(TAG);

#endif

#endif // QDEBUG2LOGCAT\_H

#if defined(ANDROID)

#include"qDebug2Logcat.h"

#include<android/log.h>

#include<QDebug>

#include<QByteArray>

static const char \*g\_TAG=0;

static void messageOutput2Logcat(QtMsgType type, const QMessageLogContext&,

const QString &msg)

//QMessageHandler类型为void myMessageHandler(QtMsgType, const QMessageLogContext &, const QString &);也就是说这个函数就是我们自己定义的一个QMessageHandler类型的消息管理者函数

{

int prio=ANDROID\_LOG\_VERBOSE;

QByteArray localMsg=msg.toLocal8Bit();

switch (type) {

case QtDebugMsg:

prio=ANDROID\_LOG\_DEBUG;

break;

case QtWarningMsg:

prio=ANDROID\_LOG\_WARN;

break;

case QtCriticalMsg:

prio=ANDROID\_LOG\_INFO;

break;

case QtFatalMsg:

prio=ANDROID\_LOG\_FATAL;

abort();

}

\_\_android\_log\_write(prio,g\_TAG,localMsg.data());

//优先级，标签，以及内容

}

void installLogcatMessageHandler(const char \*TAG)

{

g\_TAG = (TAG == 0 ? "QDebug" : TAG);

qInstallMessageHandler(messageOutput2Logcat);

//QtMessageHandler qInstallMessageHandler(QtMessageHandler handler)这个函数就是QT的消息管理者装载函数

}

#endif

那么在之后添加

installLogcatMessageHandler("yourLogTag");

之后QT的消息就会发送到logCat里面，其标签为yourLogTag

QtDebugMsg 0 A message generated by the qDebug() function.

QtWarningMsg 1 A message generated by the qWarning() function.

QtCriticalMsg 2 A message generated by the qCritical() function.

QtFatalMsg 3 A message generated by the qFatal() function.

QtSystemMsg QtCriticalMsg