QT에서 동영상 play를 위해서 기본으로 제공하는 QMovie의 경우 GIF혹은 MNG 파일 포맷만 가능하다. 이외의 포맷을 QT내에서 플레이 하기 위해서 아래와 같은 방법이 있다.

1. video4linux를 이용해서 QT내부 widget에서 보여주기

아래 프로그램 코드 중 CVideoDevice는 http://www.smcc.demon.nl/camstream/에서 얻을 수 있습니다.

```
void zxCam::init()
{
    device = new CVideoDevice("/dev/video0");
    device->Open(1);
    if (!device->SetSize(320, 240)) {
        qDebug("Couldn't set the frame size to 320x240!");
    }
    device->EnableYUV(true);
    device->EnableRGB(true);
    if (!device->SetFramerate(30)) {
        qDebug("Couldn't set the frame rate to 30fps!");
    }
    connect(device, SIGNAL(Notify()), this, SLOT(camFrameReady()));
}
void zxCam::destroy()
{
    device->Close();
    delete(device);
}
void zxCam::camFrameReady()
{
    image = *device->GetRGB();
    repaint(false);
}
void zxCam::paintEvent(QPaintEvent *e)
{
    QPainter p(this);

    p.drawImage(0, 0, image);
}
```

2. xine-lib (http://xinehq.de) 를 이용. 이미 QT 로 포팅된 프로젝트가 있음.
(http://xinehq.de/index.php/releases)

※QT 로 포팅은 되었지만 xine 를 사용하기 위해서는 xine 라이브러를 arm 용으로 포팅해야 하는 번거러움이 있습니다. 이 부분이 아무래도 까다다로운 듯 합니다.

3. openML 을 QT Widget 에서 보여주기

4. mplayer 소스 참고하기

사실 플레이어의 완벽한 제어를 위해서는 MPlayer 의 소스분석을 통해서 제어구문을 QT 내에 넣어주는 방법이 제일 완벽한 방법이라고 합니다. 많이 어렵겠지요.

여기서는 MPlayer 를 arm 용으로 간단하게 포팅해서 플레이만 가능하게(system 함수 사용) QT 로 작성한 예제를 소개해 드리겠습니다.

Mplayer Arm 용 포팅 가이드

Termcap 포팅

termcap 은 GNU와 리눅스 두가지가 있다. 이중 우리는 리눅스 버전을 구하여야 한다. 다음으로 접속하면 리눅스용 termcap 관련 화일을 얻을수 있다.

http://www-ftp.lip6.fr/pub/linux/GCC/termcap-2.0.8.tar.gz

다음은 이 화일 적당한 디렉토리에 놓고 다음과 같은 과정을 취하여 푼다.

```
tar zxvf termcap-2.0.8.tar.gz
```

이 과정이 끝나면 termcap-2.0.8 란 디렉토리가 생긴다.

termcap-2.0.8 디렉토리로 이동한다.

```
cd termcap-2.0.8
```

Makefile 을 수정한다.

```
vi Makefile
```

Makefile 의 내용중 다음을 고친다.

```
CC=gcc              를 CC=armv5l-linux-gcc          로 수정
CFLAGS=-O -I. -g 를 #CFLAGS=-O -I. -g 로 주석처리한다.
AR=ar               를 AR=armv5l-linux-ar           로 수정

prefix = /  를 prefix = /usr/armv5l-linux 로 수정
```

tparam.c 의 28 번째 라인에서

```
#define bcopy(s, d, n) memcpy ((d), (s), (n))
```

을 주석 처리한다.

termcap.texi 화일의 2615 라인에서

```
    but in that regard it is obsolete (@xref{Cursor Motion}).
```

문장에서 "(" ")" 를 없애준다.

```
    but in that regard it is obsolete @xref{Cursor Motion}.
```

컴파일 한다.

```
make
```

인스톨 한다. make install 다음과 같은 명령으로 정확히 설치되었는지를 확인한다.

```
[root@jdt termcap-2.0.8]# ls -al /usr/armv5l-linux/lib/libtermcap.*
```

```
    -rw-r--r--       1 bin         bin             12680 Nov 26 16:35 /usr/armv5l-linux/lib/libtermcap.a
    lrwxrwxrwx       1 bin         bin                38 Nov 26 16:35 /usr/armv5l-linux/lib/libtermcap.so
-> /usr/arm-linux/lib/libtermcap.so.2.0.8
    -rwxr-xr-x       1 bin         bin             15097 Nov 26 16:35 /usr/armv5l-
linux/lib/libtermcap.so.2.0.8
```

libmad 포팅

libmad 는 interger 연산을 위해 필요한 라이브러리로 Arm 용으로 Mplayer 를 포팅하면 floating 연산때문에 동영상재생이 원할치 않다.
libmad 를 enable 시켜서 컴파일 하면 현저하게 좋아짐을 느낄수 있다.

libmad 를 구해오자.
#wget ftp://ftp.mars.org/pub/mpeg/libmad-0.15.1b.tar.gz

설치과정
#tar zxvf libmad-0.15.1b.tar.gz
#cd libmad-0.15.1b
#./configure --prefix=/usr/local/arm/libmad --enable-fpm=arm --host=armv5l-linux

--prefix 로 libmad 를 설치할 디렉토리를 지정해주고 --host 옵션에 크로스컴파일러의 선행문자를 적어준다.
#make
#make install

MPlayer 포팅

http://www.mplayerhq.hu
Mplayer 홈페이지에서 소스(MPlayer-1.0pre7try2.tar.bz2)랑 코덱(all-20050412.tar.bz2)을 받아온다.

코덱은 /usr/local/lib/codecs 에 설치한다.
#tar jxvf all-20050412.tar.bz2
#mv all-20050412 /usr/local/lib/codecs

이제 Mplayer 설치로 들어가 보자.

#tar jxvf Mplayer-1.0pre7try2.tar.bz2
#cd Mplayer-1.0pre7try2
#./configure -cc=armv5l-linux-gxx -target=arm-linux -enable-static -ccprefix=/usr/local/arm/mplayer -disable-win32 -disable-dvdread -disable-noparanoia -enable-fbdev -disable-mencoder --disable-mpdvdkit -disable-x11 -disable-s 이 --enable-mad --with-madlibdir=/usr/local/arm-linux/lib

※여기서 에러 발생을 대비해서 m.sh 이란 쉘 파일을 만들어서 configure 의 내용을 적어놓았다. 매번 적을 필요없이 쉘 파일만 실행함으로써 make 파일이 만들어 지도록 하였다.

make 하기전 소스 파일들을 조금 손 보아야 한다.

Arm 용 codec-cfg 에 문제가 있어서 x86 용 gcc 로 변경한다.
#vi Makefile
280 번 라인 수정
$(HOST_CC) $(HOST_CFLAGS) -I. -g codec-cfg.c .....를
gcc $(HOST_CFLAGS) -I. -g codec-cfg.c .....으로 고친다.

Arm 용으로 컴파일한 mad 나 termcap 라이브러리를 참조할 수 있게한다.
#vi config.mak
111 번 라인 수정
TERMCAP_LIB=-ltermcap -L/usr/local/arm/termcap/lib 으로 수정

155 번 라인 수정
MAD_LIB=-lmad -L/usr/local/arm/libmad/lib 으로 수정

역시 mad 관련 헤드파일의 경로를 올바르게 지정해준다.
#vi libmpcodecs/ad_libmad.c
23 번 라인 수정
#include<mad.h> 을
#include "/usr/local/arm/libmad/include/mad.h"로 수정한다.

이제 컴파일하고 시스템 사양에 따라 기다리는 일만 남았다.
#make
.....

#ls -l mplayer
-rwxr-xr-x 1 root root 10425412 Feb 23 16:37 mplayer
mplayer 실행 파일이 생긴 것을 알수있다. ^_^
#file mplayer
mplayer: ELF 32-bit LSB executable, ARM, version 1 (ARM), for GNU/Linux 2.4.19, statically linked, not stripped

make install 을 하면 strip 관련 에러가 난다. make 까지만 하고 직접 strip 한다.
#armv5l-linux-strip mplayer
#ls -l mplayer
-rwxr-xr-x 1 root root 5362716 Feb 23 16:43 mplayer
파일 용량을 반정도 줄일 수 있다.

에러 메시지

다음은 make 관련 에러 메시지를 정리해 보았다.
위 내용대로 하였다면 아래 에러는 나오지 않아야 정상이다.

에러 - codec-cfg.c:500: for each function it appears in.)
codec-cfg.c:501: `builtin_audio_codecs' undeclared (first use in this function)
make: *** [codec-cfg.o] Error 1
해결 - Makefile 에서 codec-cfg 관련 부분을 수정해야한다.

에러 - /usr/lib/libtermcap.a(termcap.o)(.text+0x25): R_ARM_PC24 relocation against SEC_MERGE section
collect2: ld returned 1 exit status
make: *** [mplayer] 오류 1
해결 - Arm 용 termcap 관련 라이브러리를 설치한다.

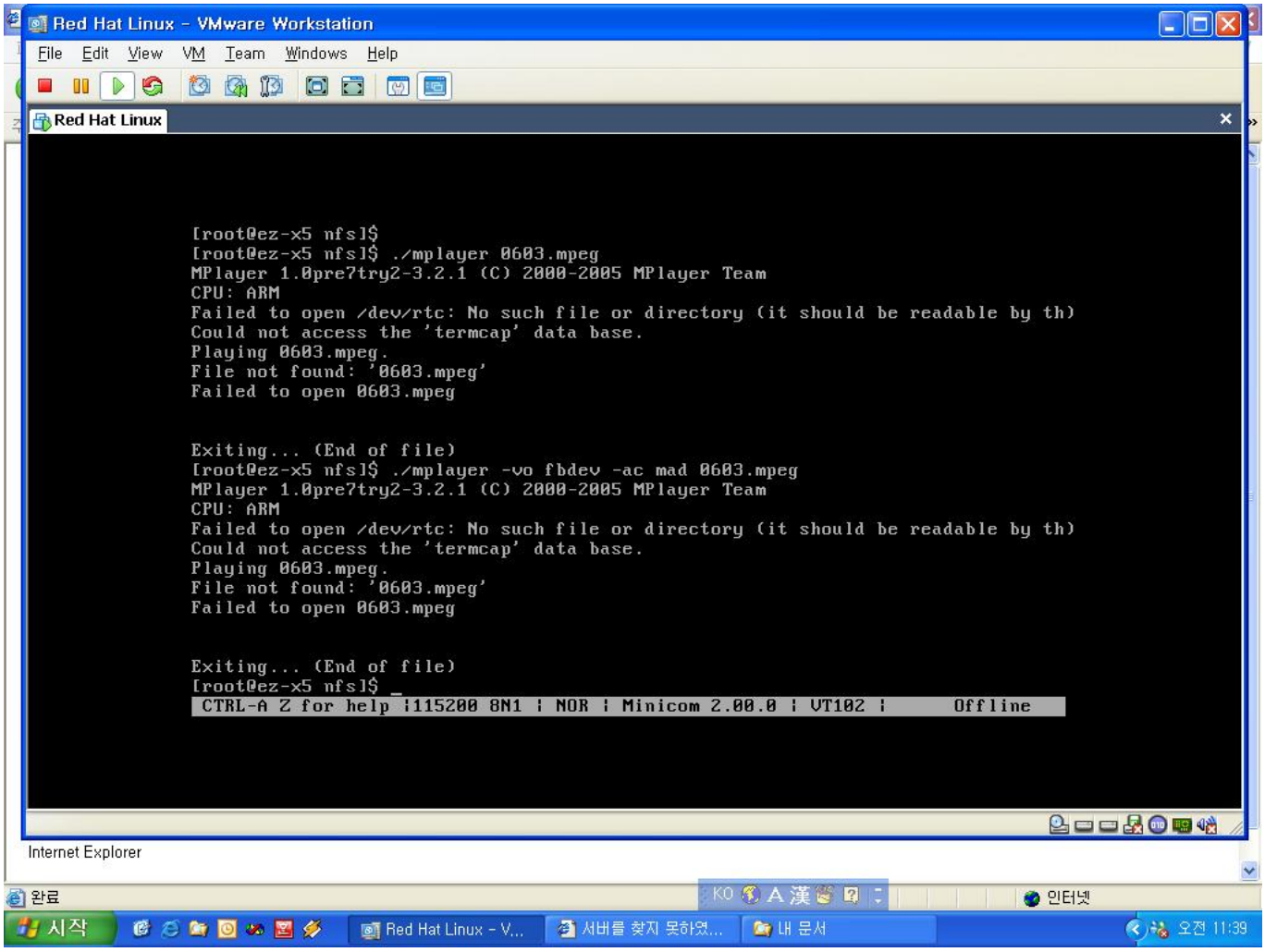
에러 - ad_libmad.c:41: confused by earlier errors, bailing out
make[1]: *** [ad_libmad.o] Error 1
make[1]: Leaving directory `/root/download/MPlayer-1.0pre7/libmpcodecs'
make: *** [libmpcodecs/libmpcodecs.a] Error 2
해결 - libmpcodecs/ad_libmad.c 파일에서 헤드파일 경로을 수정해야 한다.

에러 - /usr/local/arm-linux/lib/gcc-lib/arm-linux/3.2.1/../../../../arm-linux/bin/ld: cannot find -lmad
collect2: ld returned 1 exit status
make: *** [mplayer] Error 1
해결 - config.mak 파일에서 mad 라이브러리 경로를 수정해야 한다.

실행하기

./mplayer -vo fbdev -ac mad test.mpg(재생할 파일명)
-ac 옵션을 주고 mad 사용해서 돌리면 확실히 낫다.

./mplayer -ac mad test.mp3
성공적으로 컴파일을 하고 nfs 를 이용해서 돌려보려 했으나 예상치 못한 에러가 발생했다.



/dev/rtc 에러와 could not access the 'termcap'data base
두 개의 에러…

첨엔 rtc가 올라가지 않아서 그런 줄 알고 커널을 다시 컴파일해서 올렸으나 여전히 같은에러… 그러다가 보드상에서 동영상 파일명만 입력(#test.mpeg 엔터)하고 실행했는데 퍼미션
에러가 발생하였다. 그래서 혹시나 하고 nfs상의 동영상 파일의 퍼미션을 777로 변경.
다시실행 재생에 성공하였다. Mp3역시 재생이 가능. 둘다 처음엔 소리가 재생이 되지 않았으나 커널을 다른걸로 바꿔 올려서 해결함.

QT로 작성한 간단한 예제(Qt/embedded 2.3.7)

mywidget.h

```
#ifndef MYWIDGET_H
#define MYWIDGET_H

#include <iostream.h>
#include <qwidget.h>
#include <qpushbutton.h>
#include <qfiledialog.h>

Class Mywidget : public QWidget
{
        Q_OBJECT;
public :
        Mywidget();

Private :
        QPushButton* bt1;
        QPushButton* bt2;
        QPushButton* bt3;
        QFileDialog* fd;

Protected slots;
        void playClicked();
        void stopClicked();
        void openClicked();
};
#endif          //MYWIDGET_H
```

mwidget.cpp

```
#iinclude <string.h>
#include “mywidget.h>

MyWidget :: MyWidget()
{
        setCaption(“test player”);

        bt1 = new QPushButton(“Play”, this);
        bt2 = new QPushButton(“Stop”, this);
        bt3 = new QPushButton(“Open”, this);
        bt1->setGeometry(10,20,40,20);
        bt2->setGeometry(60,20,40,20);
        bt3->setGeometry(110,20,40,20);

        connect(bt1, SIGNAL(clicked()), this, SLOT(playClicked()));
        connect(bt2, SIGNAL(clicked()), this, SLOT(stopClicked()));
        connect(bt3, SIGNAL(clicked()), this, SLOT(openClicked()));
}

void MyWidget :: playClicked()
{
        //system(str);
}

void MyWidget :: stopClicked()
{
        //system(“p”);
}

void MyWidget :: openClicked()
{
        QString file;
        char str[100] = “mplayer “;
        strcat(str, file);
        file = fd->getOpenFileName(“/”, “*.*”);
        system(str);
```

```
}
```

main.cpp

```
#include <qapplication.h>
#include “mywidget.h”

Int main(int argc, char** argv)
{
        QApplication app(argc, argv);
        MyWidget mw;

        app.setMainWidget(&mw);
        mw.resize(640,480);
        mw.show();

        return app.exec();
}
```

실행화면

Q & A

현재 QT 로 인터페이스는 만들었습니다. 재생, 정지, 기타등등 그런데 한가지 문제점이
동영상 플레이서(mplayer)를 system("mplayer 파일이름");을 해주면 QT 의 상단왼쪽에 화면이 뜨는데 물론 Widget 의 키는 동작이 됩니다.
mplayer 의 화면 위치를 정 중앙에 뛰우고자 하는데 어떤 옵션을 줘야 하나요
-x 위치 -y 위치를 했는데 위치 변화가 없더라구요 ^^; 이건 아닌것 같구
-fs 를 주면 플스크린에 화면 중앙에 320x240 으로 뜨고.... (이건 바탕이 검정색으로
변해 버려서 ㅡ.ㅡ 뒤에 Widget 이 가려져 버려서 클릭이 안되고.. help~~)

fbdev 를 이용하시면 mplayer 폴더안에 libvo/ 보시면 vo_fbdev.c 파일이 있습니다. 여기서 offset 값을 수정해 주셔야 합니다. 위치는 직접 찾아보시길

안녕하세요. 지금 한창 임베디드 리눅스를 공부하고 있는 학생입니다.
저는 지금 arm 보드에 제가 제작한 qt 용 프로그램을 올려서 플레이 버튼을 누르면
mplayer 가 뜨고 제어 할수 있도록 하려고 합니다.
지금 상태는 플레이 버튼을 누르면 플레이어가 실행은 됩니다.(system()함수사용)
그런데 플레이가 되고 있는 상태에서는 제가 만든 qt 프로그램으로 제어권이 넘어가지 않습니다.(정지버튼,,,등등 버튼이 먹지 않고 정지상태)
스레드를 이용한다면 어떤식으로 사용해야 할까요??
어떤식으로 해결해야 할런지요..
지금 고민한지 한참이 지났는데 해결방법이 없네요..

만약 mplayer 를 사용하는 것이 힘들다면 제가 만든 qt 프로그램으로 동영상 재생 음악재생하게 하려면 어떤식으로 접근하는 것이 좋을까요? 고수님들 답변 부탁드립니다..(__)

우선 system 함수를 이용해서 제어하는것은 제어의 많은 한계점이 있어서 원하시는 기능을 제대로 구현할 수 없습니다. Thread역시 마찬가지겠죠...

이걸 해결하는 가장 좋은 방법은, mplayer의 해당 동영상 소스부분을 카피해서 직접 프로그램내에 embedding 시키는 것입니다.
mplayer 소스코드를 보고 직접 구현해 넣으셔야 합니다.

안녕하세요~ 제가 초보라서 system 함수를 어떻게 사용하는지 정확히 모르겠습니다.
예를 들어서 qt 로 버튼을 하나 만들고 그 버튼을 눌렀을 때 /mnt/nfs 디렉토리에 있는
a.mp3 를 mplayer 로 동작시킨다고 가정 했을 때

```
Button :: Button()
{
      resize(480,320);
      p = new QPushButton("play",this);
      p -> setGeometry(20,20,80,80);
      connect(      ); // 이부분과
      system();         // 이부분의 사용법을 잘 모르겠습니다.
}
```

물론 위 소스코드가 잘못되었겠지만
잘 아시는 분께서는 쉽게 아시리라 생각됩니다. ^^;;

```
conncet(p,SIGNAL(clicked)),this,SLOT(player()));
void Main::player()
{
   system("mplayer /mnt/nfs/a.mp3");
}
```

이렇게 함 될듯 한데요..
약간의 보충설명입니다.
system 함수는 콘솔에서 실행하는것을 프로그램에서 실행하는 프로그램입니다
가능한 쓰지 않는 편이 좋구여..
system("command");
이런식으로 호출하면되구여...
예를 들어 system("cp /mnt/nfs/a.mp3 /mnt/nfs/b.mp3");
라고 실행하면 a.mp3 를 b.mp3로 복사하는것도 가능합니다..
콘솔에서 실행하는 것으로 생각하시면 될듯...

창에 뿌리기 위해서는 첫째 mplayer를 system 함수가 아닌 thread로 만들어 제어를 해야 합니다.
mplayer를 하나의 object로 만들어서 ui에서 thread로 생성하여 input.c의 command 에 ui에서 넣어 주는 방식으로 구동 되어야 합니다.
그리고 창에 뿌린다는 말은 mplayer 에서 fb0 에 뿌릴 때 창의 좌표를 넘겨 줘서 뿌리 도록 수정 해야 합니다.

□ Qt-embedded-2.3.2에서 Qprocess사용하기

Qt/Embedded-3 이상 버전에서는 기본 적으로 제공하는 기능인 Qprocess

를 Qt/Embedded-2.3.2 에서는 제공하지 않는다. 이 때문에 우리는

Qprocess 를 Qt-embedded-2 버전에서도 사용하기 위해서 다음과 같은

방법을 사용해서 Qprocess 를 사용하였다.

- $QTDIR/tools/designer/util 에 있는 util.pro 파일을 다음과 같이 수정한다.

CONFIG += qt warn_on debug

→ CONFIG += qt warn_on debug staticlib

일반 DLL 로 할경우 일반 Application 에서 제대로 링크가 걸리지 않아서

위와같이 static library 형태로 만들어서 성공하였다.

- Makefile 생성

  tmake -o Makefile util.pro

  vi Makefile 로 -lqutil 을 추가해준다.

- make

  make 가 성공적으로 수행되면 $QTDIR/lib 밑에 libqutil.a 파일이 생성됨

- header 파일 copy

  실제 프로그램에서 사용하기 위해서 $QTDIR/tools/designer/util/qprocess.h

파일을 $QTDIR/include 로 복사함

2.x 는 3.x 의 QProcess 와는 다른방법으로 사용해야한다.

```
QProcess* proc =new QProcess("fdisk");
connect(proc, SIGNAL(dataStdout(const QString&)), this,
SLOT(dataStdout(const QString&)));
proc->addArgument("-l");
proc->start();
protected slots:
void dataStdout(const QString& buf)
{
        qDebug("buf:[%s]", buf.ascii());
}
```

또는

```
QProcess* proc =new QProcess;
connect(proc, SIGNAL(dataStdout(const QString&)), this,
SLOT(dataStdout(const QString&)));
proc->setCommand("fdisk");
proc->addArgument("-l");
proc->start();
```

QProcess 사용예.

1. mplayer 실행

```
QStringList commandAndParameters;
QProcess myProcess ;
   commandAndParameters<<"./mplayer"  <<"-vo"  <<  "fbdev"  <<  "-ac"  <<  "mad"<<
"sample.mpeg";
myProcess.setArguments(commandAndParameters);
myProcess.start();
```

2. 볼륨제어

```
myProcess.setCommunication(QProcess::Stdin) ;
myProcess.writeToStdin("9") ; // 볼륨을 작게
```

MyWidget.h 헤더 파일

```
#ifndef [u]MYWIDGET_H[/u]
#define [u]MYWIDGET_H[/u]

#include <qwidget.h>

class MyWidget : public QWidget
{
    Q_OBJECT
public:
        MyWidget();

public slots:
    void btnClicked();

};

#endif
```

MyWidget.cc 구현 파일

```
#include "MyWidget.h"
#include <qpushbutton.h>

void MyWidget::btnClicked()
{
	QStringList commandAndParameters;
	commandAndParameters<<"firefox";
	QProcess myprocess(commandAndParameters);
	myprocess.start();
}

MyWidget::MyWidget()
{
	QPushButton* mybutton1 = new QPushButton("Firefox", this);
	mybutton1->setGeometry(270,190,100,100);
	QObject::connect(mybutton1, SIGNAL(clicked()), SLOT(btnClicked()));
}
```

main.cc 파일

```
#include <qapplication.h>
#include "MyWidget.h"

int main(int argc, char* argv[])
{
	QApplication myapp(argc, argv);

	MyWidget* mywidget = new MyWidget();

	mywidget->setGeometry(0,0,640,480);
	mywidget->setCaption("Test Window!!");
	myapp.setMainWidget(mywidget);
	mywidget->show();

	return myapp.exec();
}
```

```
#include <qapplication.h>
#include <qprocess.h>
#include <qstringlist.h>

int main(int argc,char* argv[])
{
    QApplication myApplication(argc,argv);

        // 실행 커맨드와 인수가 저장될 QStringList
    QStringList commandAndParameters;

        // 실행할 파라미터를 설정합니다.
        // 여기서는 자기 local 에 있는 /home/korone 폴더를 컹커를를 통해 열게됩니다.
    commandAndParameters<<"konqueror" <<"file:/home/korone";

        // QProcess 인스턴스를 생성합니다.
    QProcess myProcess(commandAndParameters);

        // 생성된 myProcess 를 시작합니다.
    myProcess.start();

        // 외부 프로그램이 계속 수행되고, 현재 프로그램은 종료됩니다.
    return 0;
}
```