RealEvo-Qt Product Manual

graphic interface middleware

SPC003001 V1.00 Date: 2018/05/16 Product Manual

类别 Type	内容 Contents
Key Word	Qt QtCreator debugging
Abstract	Use QtCreator integrated development environment for development and commission the SpaceChain OS Qt program under the Windows environment

Revision History

Version	Date	Reason
V1.00	2018/05/16	Create file

Contents

Chapter 1 Introduction

1.1 Qt

Qt is a cross-platform C++ graphical user interface application development framework developed by Qiqu Technology Company in 1991. Since the release of its first version in 1995, Qt has experienced the history of over 20 years. It could be used either for the development of GUI (Graphical User Interface) program, or for the development of non-GUI program like console tools and servers.

In 2008, Qiqu Technology Company was acquired by Nokia, and Qt became the programming language tool belonging to Nokia. In 2009, Nokia added the business friendly LGPL v2.1 protocol in the Qt authorization agreement. In 2012, Qt was purchased by Digia company and the Qt5.0 version was introduced in the year.

Qt has been extensively applied into the desktop and embedded and mobile calculating system in different industrial sectors; it is known that over 7000 companies and 800,000 developers are using Qt all over the world currently.

Qt official website: http://www.qt.io/

1.2 Qt Creator

Qt Creator is the cross-platform integrated development environment of the lightweight class introduced by Nokia after the procurement. Qt Creator is to support the new user of Qt to get familiar with and run project faster, as well as help experienced Qt developers improve working efficiency.

Qt Creator provides the functions of project generation guide, advanced C++ code editor, file and class browsing tools, graphic GDB debugging front end, integrating the Qt Designer (interface designer), Qt Assistant (helpfile system), Qt Linguist (translation tool), and qmake build tool.

1.3 Qwt

The full name of Qwt is Qt Widgets for Technical Applications. It is an open source project based on LGPL protocol, being able to generate various statistical charts and meter diagrams.

Qwt official website is :https://osdn.jp/projects/sfnet_qwt/

1.4 RealEvo-QtSylixOS

In March 2013, Qt was transferred to SylixOS embedded real-time operating system. SylixOS system became one of the very few embedded real-time operating systems supporting Qt in the world.

RealEvo-QtSylixOS is the software introduced by ACONIFO to develop the Qt applications in SylixOS system specially.

RealEvo-QtSylixOS integrates the SylixOS plugins of Qt Creator, Qt shared library based on LGPL protocol in all kinds of processors, gdb debugging tool supporting Python, and Qwt.

SylixOS plugin enables Qt Creator to develop the Qt application in SlixOS with almost zero configuration. One-button deployment of Qt shared library to SylixOS device makes the configuration of Qt application environment extremely simple. One-button deployment, operation, debugging and analysis of Qt application improves the efficiency of developers greatly.

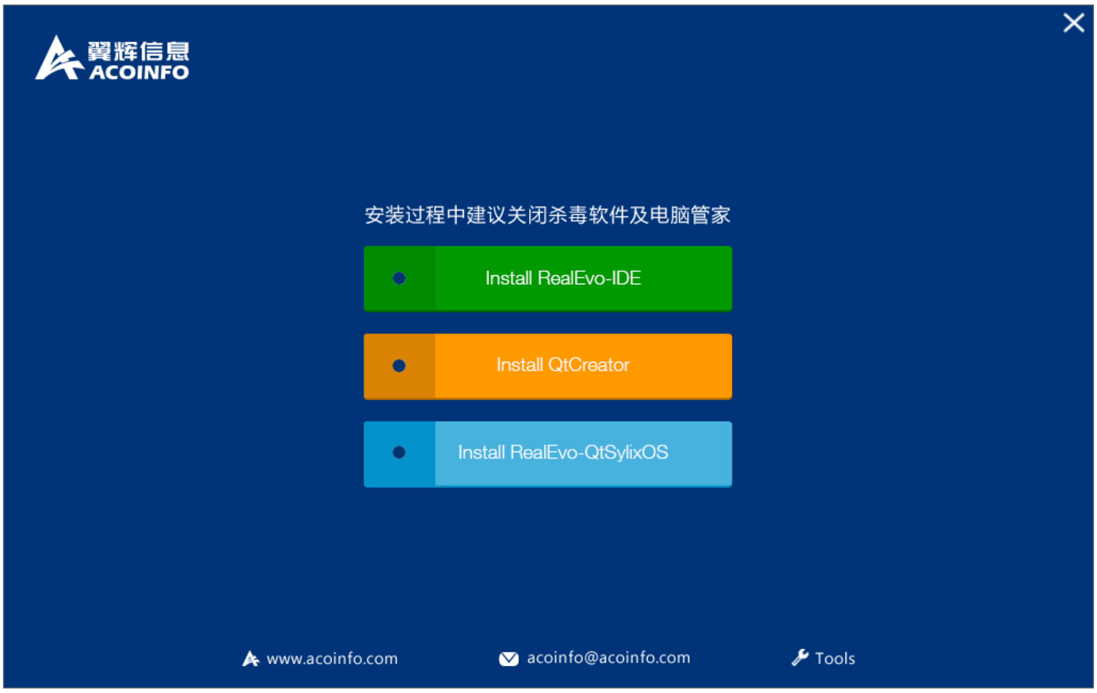
Meanwhile, RealEvo-QtSylixOS integrates the Qt shared libraries based on LGPL protocol in all kinds of processors, which enables developers to be exempted from Qt compiling but concentrate on the development of Qt application programs.

Note: LGPL allows the business software to use LGPL class library through class library link (link) without needing the code of open-source commercial software. Such enables the open-source code adopting LGPL protocol to be linked by the commercial software as class library, released and sold.

Chapter 2 Install RealEvo-QtSylixOS

2.1 Install Qt Creator

Open the SylixOS IDE installation program as shown by figure 2.1, select “Install QtCreator”, install the Qt Creator development environment.



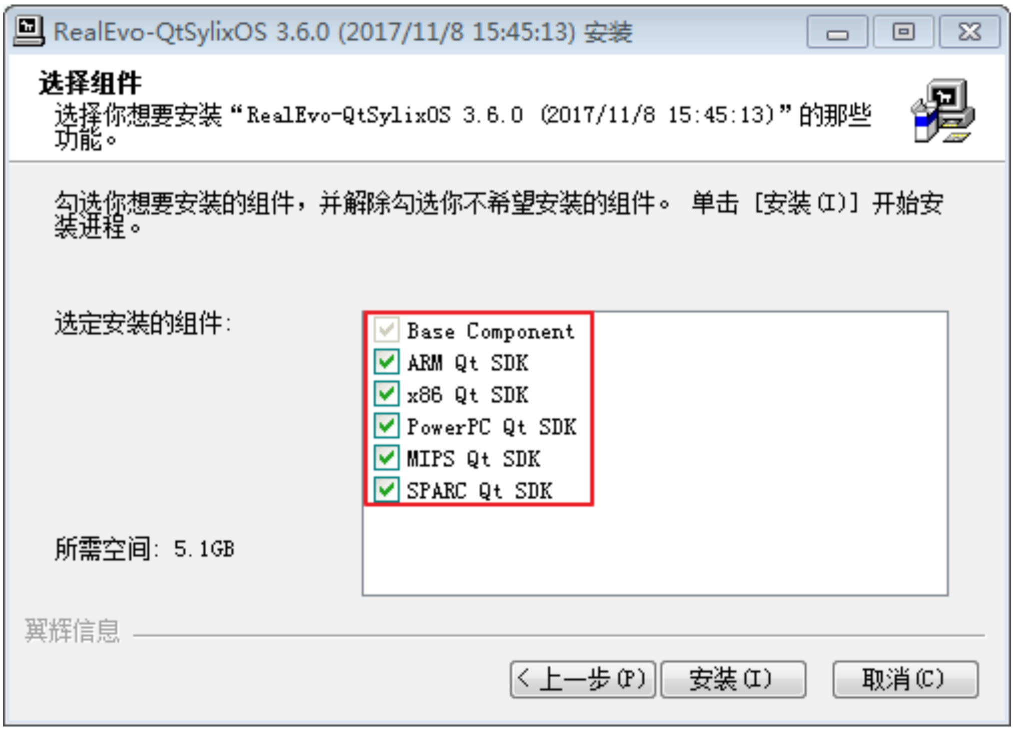
安装过程中建议关闭杀毒软件及电脑管家

Figure 2.1 SylixOS IDE installation program

Note: First, install the Qt Creator, then install RealEvo-QtSylixOS, and do not install Qt Creator of the other version.

2.2 Install RealEvo-QtSylixOS

Click the “Install RealEvo-QtSylixOS” installation option in the SylixOS IDE installation program to install QtSylixOS in the designated installation directory of QtCreator; select the Qt SDK for the corresponding CPU architecture in accordance with requirements, select all in default (it is suggested that the user install all Qt SDK) as shown in figure 2.2.



RealEvo-QtSylixOS 3.6.0 (2017/11/8 15:45:13) 安装

选择组件

选择你想要安装“RealEvo-QtSylixOS 3.6.0 (2017/11/8 15:45:13)”的那些功能。

勾选你想要安装的组件，并解除勾选你不希望安装的组件。 单击 [安装(I)] 开始安装进程。

选定安装的组件:

☑ Base Component
☑ ARM Qt SDK
☑ x86 Qt SDK
☑ PowerPC Qt SDK
☑ MIPS Qt SDK
☑ SPARC Qt SDK

所需空间: 5.1GB

Figure 2.2 RealEvo-QtSylixOS CPU system module

Note: Exit from Qt Creator before installing the RealEvo-QtSylixOS.

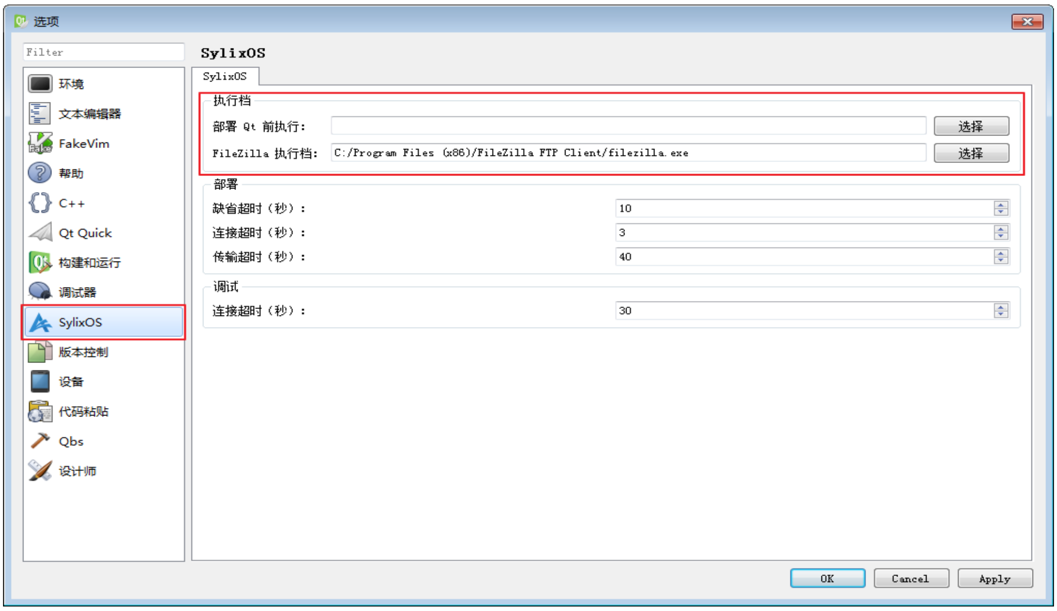
Chapter 3 Configure RealEvo-QtSylixOS

3.1 Open RealEvo-QtSylixOS

Double click the icon of RealEvo-QtSylixOS in the desktop, then the RealEvo-QtSylixOS could be opened.

3.2 Configure RealEvo-QtSylixOS

Click “tools→ option...” in the menu, and the option dialog box would pop out; click the “SylixOS” in the left column and then it would switch to the subpage of “SylixOS” as shown in figure 3.1.



执行档

部署 Qt 前执行:

FileZilla 执行档: C:/Program Files (x86)/FileZilla FTP Client/filezilla.exe

部署

缺省超时（秒）: 10

连接超时（秒）: 3

传输超时（秒）: 40

调试

连接超时（秒）: 30

Figure 3.1 Option dialog box

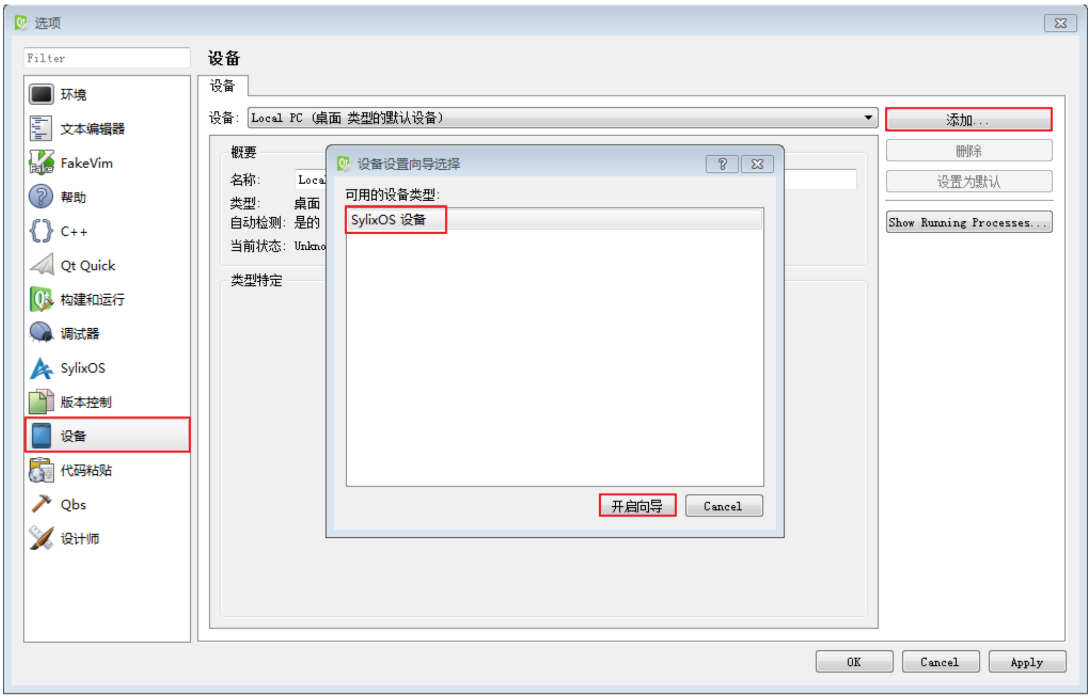
In the group of “execute file”, “execute before deploy Qt” input box could designate an executable program of Windows which would be executed every time before deploying the Qt shared library to the SylixOS device. Such an executable program realizes the revision on the deployed file system, in order to reach the target of customizing different SylixOS device.

In the group of “execute file”, “FileZilla execute file” input box could designate an executable program filezilla.exe. If the user installs the FileZilla in the default directory, then it will be automatically identified.

For the time parameters in the group of “Deployment” and “debugging”, it is not suggested for the user to make revision. Default is ok.

3.3 Configure SylixOS device

Click the “device” in the left column of figure 3.1, switch to the “device” subpage, click “add...” button, the device installation guide selection dialog box would pop out as shown in figure 3.2.



设备: Local PC (桌面 类型的默认设备)

名称: Loca

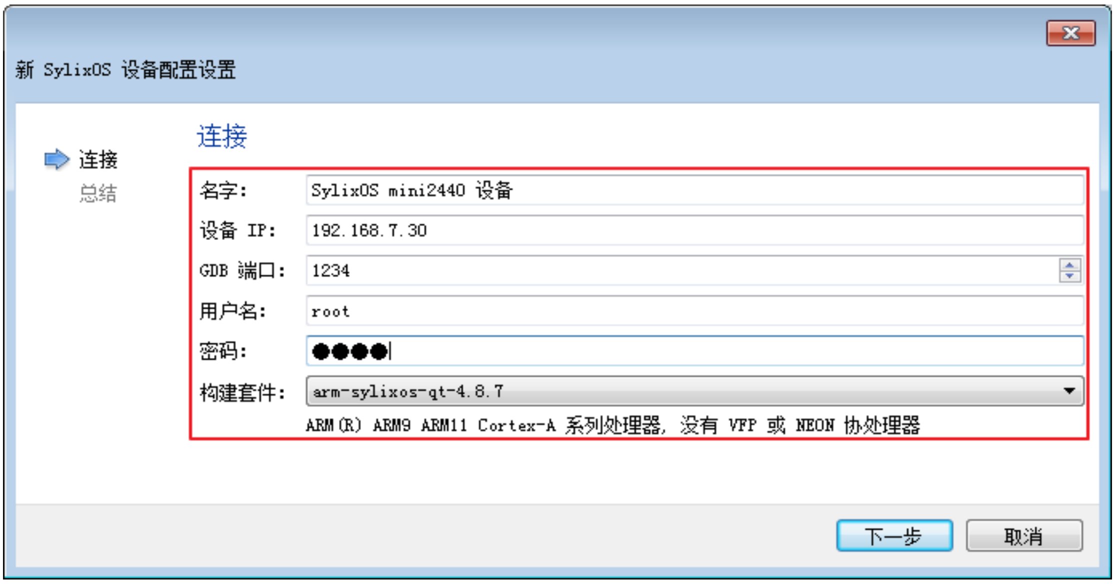
类型: 桌面

自动检测: 是的

当前状态: Unkno

Figure 3.2 Device configuration guide selection dialog box

Select the “SylixOS device”, then click the “open guide” button to start the SylixOS device configuration setup guide as shown in figure 3.3.



名字: SylixOS mini2440 设备

设备 IP: 192.168.7.30

GDB 端口: 1234

用户名: root

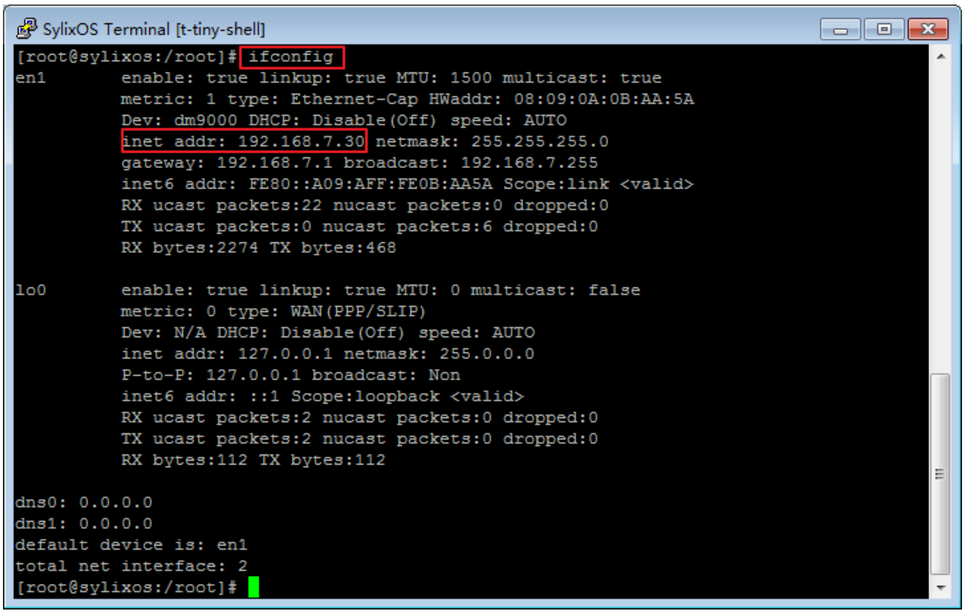
密码: ●●●●

构建套件: arm-sylixos-qt-4.8.7

ARM(R) ARM9 ARM11 Cortex-A 系列处理器, 没有 VFP 或 NEON 协处理器

Figure 3.3 SylixOS device configuration setup guide

Input the name of SylixOS device in the “name” input box, such as “SylixOS mini2440 device”; input the IP address in the “device IP” input box, such as “192.168.7.30”; the IP address of SylixOS device could be checked by inputting “ifconfig” order in the terminal of SylixOS device as shown by figure 3.4. The value of “inet addr” is the IP address of SylixOS device.



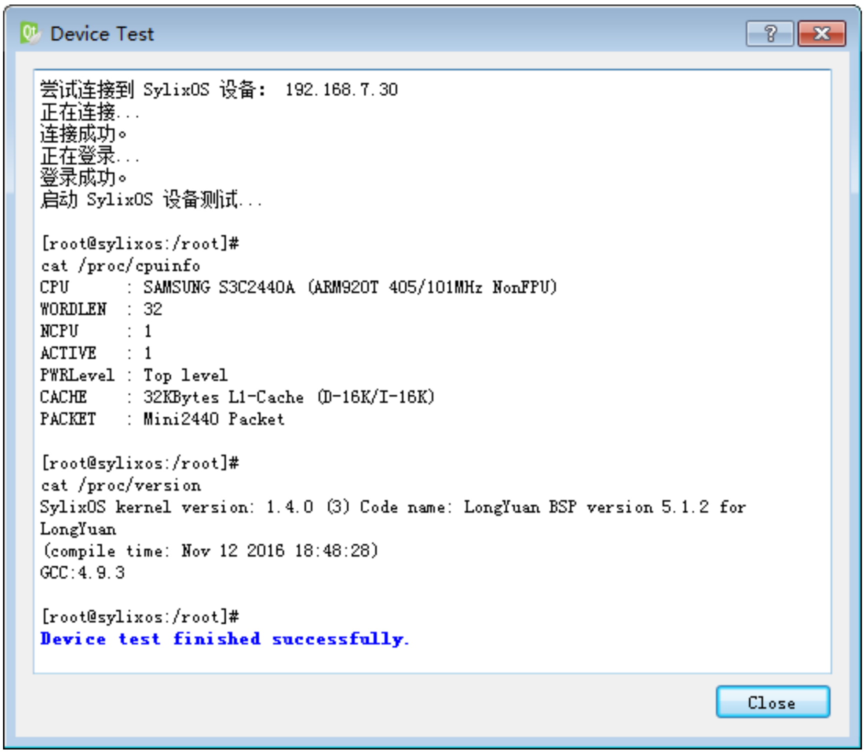
```
[root@sylixos:/root]# ifconfig
en1       enable: true linkup: true MTU: 1500 multicast: true
          metric: 1 type: Ethernet-Cap HWaddr: 08:09:0A:0B:AA:5A
          Dev: dm9000 DHCP: Disable(Off) speed: AUTO
          inet addr: 192.168.7.30 netmask: 255.255.255.0
          gateway: 192.168.7.1 broadcast: 192.168.7.255
          inet6 addr: FE80::A09:AFF:FE0B:AA5A Scope:link <valid>
          RX ucast packets:22 nucast packets:0 dropped:0
          TX ucast packets:0 nucast packets:6 dropped:0
          RX bytes:2274 TX bytes:468

lo0       enable: true linkup: true MTU: 0 multicast: false
          metric: 0 type: WAN(PPP/SLIP)
          Dev: N/A DHCP: Disable(Off) speed: AUTO
          inet addr: 127.0.0.1 netmask: 255.0.0.0
          P-to-P: 127.0.0.1 broadcast: Non
          inet6 addr: ::1 Scope:loopback <valid>
          RX ucast packets:2 nucast packets:0 dropped:0
          TX ucast packets:2 nucast packets:0 dropped:0
          RX bytes:112 TX bytes:112

dns0: 0.0.0.0
dns1: 0.0.0.0
default device is: en1
total net interface: 2
[root@sylixos:/root]#
```

Figure 3.4 ifconfig order

Input the port number monitored when the device starts Qt application debugging in the input box of “GDB port”, such as “1234”; input the user name “root” in the “user name” input box; input the password “root” in the “password” input box.

Select the applicable building kit (Kit) in the drop-down box of the building kit (when the Kit configures the device, it could only select the applicable device here).

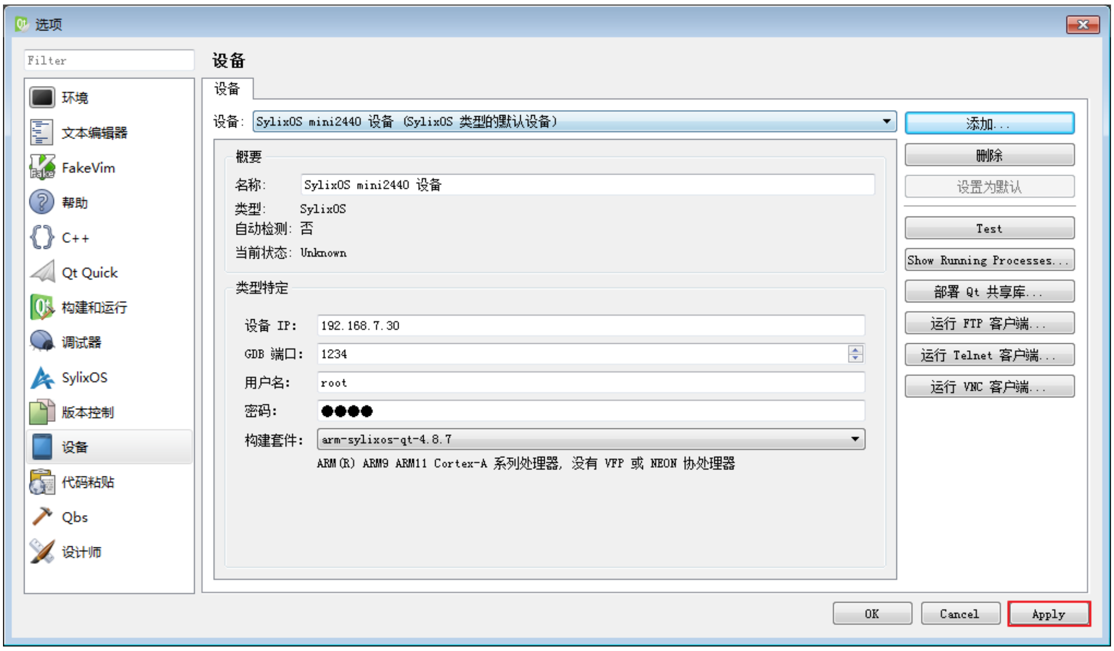
Click the “Next” button to enter the device summary dialog box; click “complete” button in the device summary dialog box to enter the device text dialog box (make sure the SylixOS device could connect to the PC through network) as shown by the figure 3.5.



```
尝试连接到 SylixOS 设备: 192.168.7.30
正在连接...
连接成功。
正在登录...
登录成功。
启动 SylixOS 设备测试...

[root@sylixos:/root]#
cat /proc/cpuinfo
CPU       : SAMSUNG S3C2440A (ARM920T 405/101MHz NonFPU)
WORDLEN   : 32
NCPU      : 1
ACTIVE    : 1
PWRLevel  : Top level
CACHE     : 32KBytes L1-Cache (D-16K/I-16K)
PACKET    : Mini2440 Packet

[root@sylixos:/root]#
cat /proc/version
SylixOS kernel version: 1.4.0 (3) Code name: LongYuan BSP version 5.1.2 for
LongYuan
(compile time: Nov 12 2016 18:48:28)
GCC:4.9.3

[root@sylixos:/root]#
Device test finished successfully.
```

Figure 3.5 Device test dialog box

After the successful test of the device, click “Close” button to finish setup, return to the subpage of “device”, and click “Apply” button for the setup application as shown by figure 3.6.



设备

设备: SylixOS mini2440 设备 (SylixOS 类型的默认设备)

概要

名称: SylixOS mini2440 设备
类型: SylixOS
自动检测: 否
当前状态: Unknown

类型特定

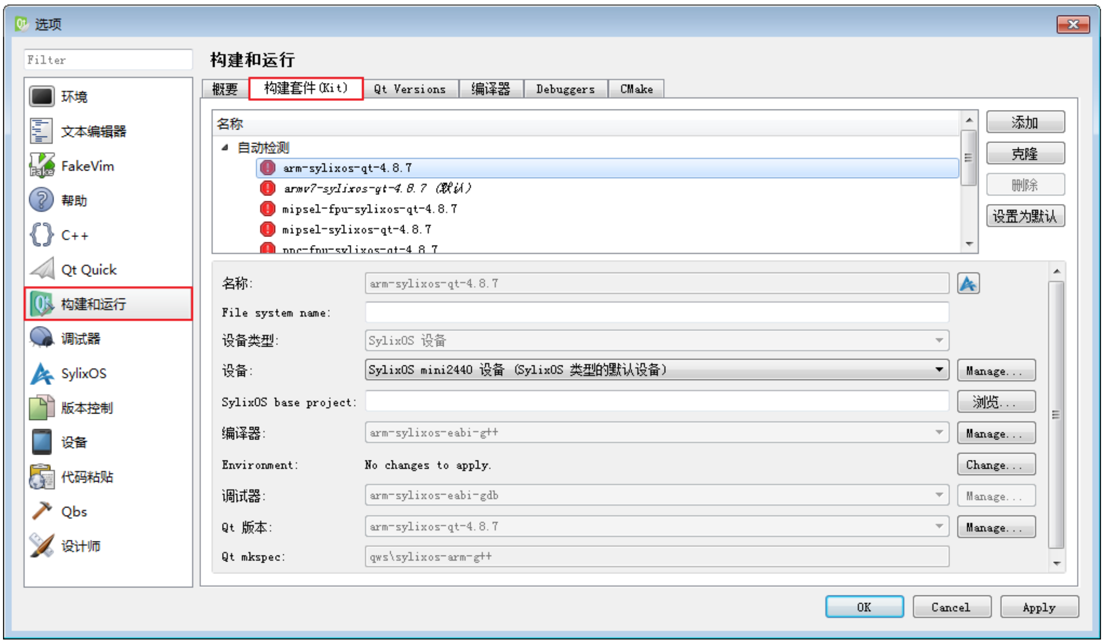
设备 IP: 192.168.7.30
GDB 端口: 1234
用户名: root
密码: ●●●●
构建套件: arm-sylixos-qt-4.8.7
ARM(R) ARM9 ARM11 Cortex-A 系列处理器, 没有 VFP 或 NEON 协处理器

Figure 3.6 “Device” subpage

In the figure 3.6, click the three buttons of “run FTP client...”, “run Telnet client...”, “run VNC client...” respectively to make fast run of FTP client, Telnet client and VNC client and connect to the SylixOS device. The detailed information of VNC can be found in section 8.2.

3.4 Configure Kit

Click the “build and run” at the left column of the option dialog box, and switch to the “Kit” option card, as shown by figure 3.7.



自动检测
- arm-sylixos-qt-4.8.7
- armv7-sylixos-qt-4.8.7 (默认)
- mipsel-fpu-sylixos-qt-4.8.7
- mipsel-sylixos-qt-4.8.7
- ppc-fpu-sylixos-qt-4.8.7

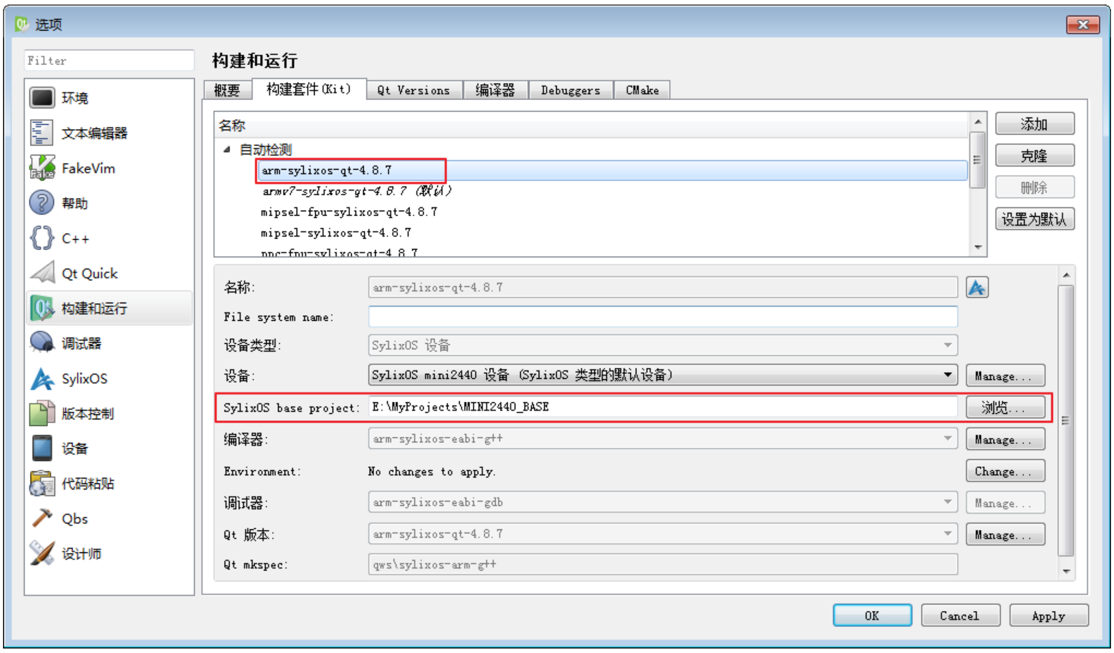
名称: arm-sylixos-qt-4.8.7
File system name:
设备类型: SylixOS 设备
设备: SylixOS mini2440 设备 (SylixOS 类型的默认设备)
SylixOS base project:
编译器: arm-sylixos-eabi-g++
Environment: No changes to apply.
调试器: arm-sylixos-eabi-gdb
Qt 版本: arm-sylixos-qt-4.8.7
Qt mkspec: qws\sylixos-arm-g++

Figure 37 “Kit” option card

Select the “arm-sylixos-qt-4.8.7” Kit in the “automatic inspection”, the device selects “SylixOS mini2440 device (default device of SylixOS type)”; “SylixOS base project” selects “CPU type” as the directory of SylixOS base project finished compilation in the “arm920t” by clicking the “browse...” button in the right side, as shown by figure 3.8.

Note: “device” option would automatically configure the device (device has been created successfully) of the same type with SylixOS compiler

Note: Refer to the *RealEvo-IDE User Manual* for the creation method of SylixOS base engineering（project）.



名称

- 自动检测
  - arm-sylixos-qt-4.8.7
  - armv7-sylixos-qt-4.8.7 (默认)
  - mipsel-fpu-sylixos-qt-4.8.7
  - mipsel-sylixos-qt-4.8.7
  - ppc-fpu-sylixos-qt-4.8.7

名称: arm-sylixos-qt-4.8.7

File system name:

设备类型: SylixOS 设备

设备: SylixOS mini2440 设备 (SylixOS 类型的默认设备)

SylixOS base project: E:\MyProjects\MINI2440_BASE

编译器: arm-sylixos-eabi-g++

Environment: No changes to apply.

调试器: arm-sylixos-eabi-gdb

Qt 版本: arm-sylixos-qt-4.8.7

Qt mkspec: qws\sylixos-arm-g++

Figure 3.8 SylixOS base project setup and device selection

Select the “armv7-sylixos-qt-4.8.7” Kit in the “automatic detection”. The device selects “SylixOS smart210 device (the default device of SylixOS type)” (if available), “SylixOS base project” selects your “CPU type” SylixOS base engineering’s directory under which “corex-a8” has been compiled, by clicking the “browse...” at the right side.

Click “OK” button at last to complete the setup (if the Kit configuration is successful, the red exclamation in front of the corresponding Kit would disappear. As shown by figure 3.8, all of the Kits have been successfully configured).

3.5 Kit selection

“arm-sylixos-qt-4.8.7” Kit uses the “-march=armv4” soft floating point parameter for compilation, being applicable for all ARM processors (such as ARM9, ARM11, Cortex-A series).

“armv7-sylixo-qt-4.8.7” Kit uses “-march=armv7-a-mfpu=neon” parameters for compilation, being applicable for all ARMv7-A architecture processors equipped with NEON co-processors (such as Cortex-A8, Cortex-A9, and so on). Such Kit uses the VFP-v3 of ARM and the SIMD order set of NEON. Its performance is better than the Kit of ‘arm-sylixos-qt-4.8.7”.

The Kit of “mipsel-sylixos-qt-4.8.7” uses the “-mips32 -msoft-float” parameter for compilation, being applicable for all of the mips32, mips64 processors without hard floating point co-processors (such as loongson 1b)

The Kit of “mipsel-fpu-sylixos-qt-4.8.7” uses the “-mips32 -mhard-float” parameters for compilation, being applicable for all of the mips32 and mips64 processors with hard floating co-processors (such as loongson1a, loongson2x, loongson3x, jz4780 and 24kf and so on)

“ppc-sylixos-qt-4.8.7” Kit uses the “-msoft-float” parameters for compilation, being applicable for the PowerPC processor without the hard floating co-processor.

“ppc-fpu-sylixos-qt-4.8.7” Kit uses the “-mhard-float” parameters for compilation, being applicable for PowerPC processor with hard floating point co-processor (such as the MPC750 and MPC83xx and so on).

“x86-sylixos-qt-4.8.7”Kit is applicable for all 32-bit processor with x86 system structure.

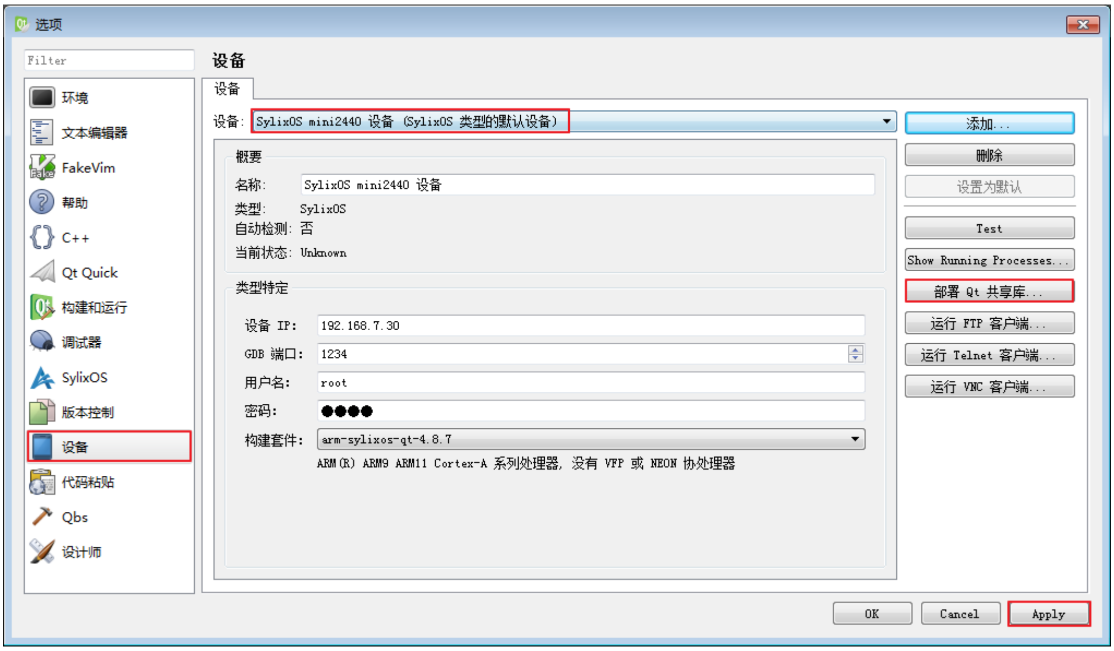
“x64-sylixos-qt-4.8.7” Kit is applicable for all 64-bit processor with X86 system structure.

Chapter 4 Deploy Qt Shared Library

4.1 Deploy Qt Shared Library

Note: If the SylixOS device has already deployed the Qt shared library, skip this chapter and go to chapter 5 directly.

Click the menu “tool →option...”, the option dialog box would pop out, click the “device” in the left column, it would switch to the “device” subpage, as shown by figure 4.1.



设备: SylixOS mini2440 设备 (SylixOS 类型的默认设备)

概要

名称: SylixOS mini2440 设备

类型: SylixOS

自动检测: 否

当前状态: Unknown

类型特定

设备 IP: 192.168.7.30

GDB 端口: 1234

用户名: root

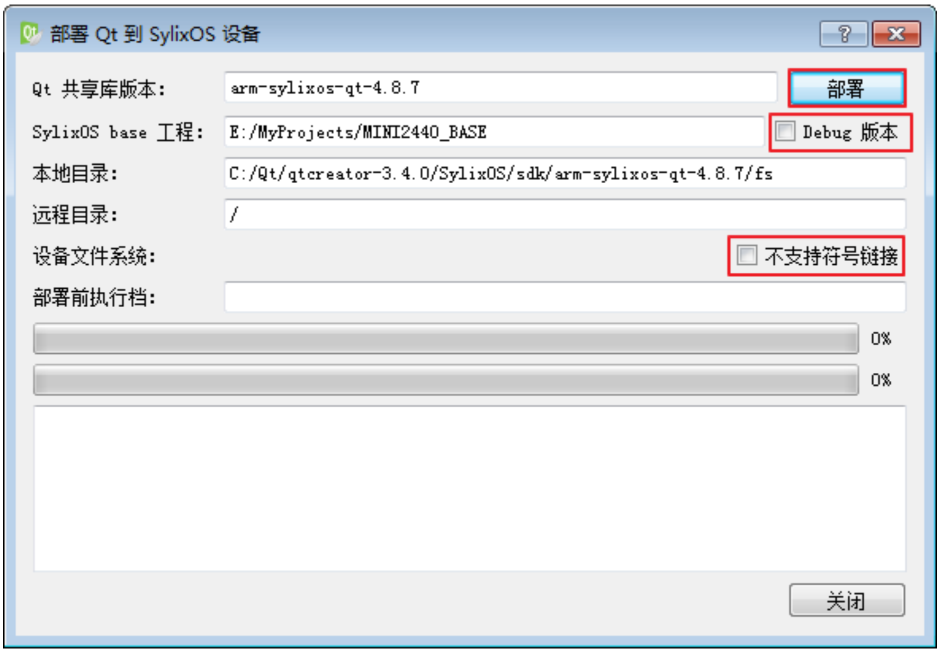
密码: ●●●●

构建套件: arm-sylixos-qt-4.8.7

ARM(R) ARM9 ARM11 Cortex-A 系列处理器, 没有 VFP 或 NEON 协处理器

Figure 4.1 “Device” subpage

Select the “SylixOS mini2440 device (the default device of SylixOS type) in the drop-down box of the device, then click “deploy Qt shared library...” button, the “deploy Qt to SylixOS device” box would pop out, as shown by figure 4.2.



部署 Qt 到 SylixOS 设备

Qt 共享库版本：arm-sylixos-qt-4.8.7

SylixOS base 工程：E:/MyProjects/MINI2440_BASE

□ Debug 版本

本地目录：C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs

远程目录：/

设备文件系统：□ 不支持符号链接

部署前执行档：

0%

0%

Figure 4.2 “Deploy Qt TO SylixOS device” dialog box

If you would like to deploy the shared library and kernel module of the Debug version in the SylixOS base project, please tick the “Debug version” check box; otherwise it would deploy the shared library and kernel module of the “Release” version.

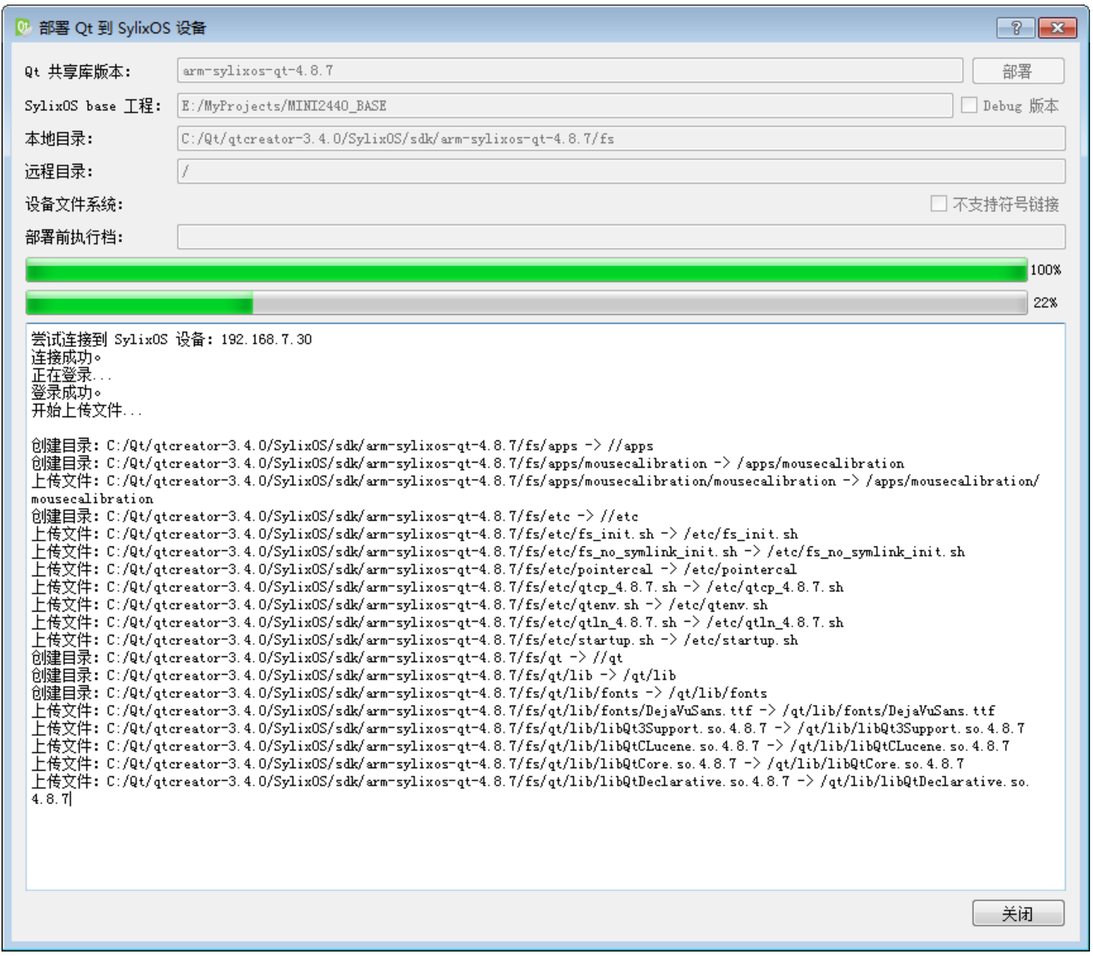
Note: If the SylixOS device usage does not support the file system linked by the symbol (such as FAT32) to serve as the file system of the storage medium, tick the check box of “do not support the symbolic link”.

At last, click the “deploy” button to start deploy the Qt shared library, as shown by figure 4.3.

If the user has designated an executable program in the “execute before deploying Qt” input box in the “SylixOS” option subpage, such a Windows executable program would be run before deployment.

Note: Pay attention to the case that the over-small partition of the hard disk might lead to the deployment failure when the hard disk is used as the storage medium for deployment.

Note: Over the deployment process of Qt shared library, the four modules of libsylixos, libcextern, libsalsa, and openssl of the SlixOS base project are depended on. Therefore, creating the SylixOS base project necessitates compiling the above mentioned 4 modules in parallel.



Qt 共享库版本: arm-sylixos-qt-4.8.7
SylixOS base 工程: E:/MyProjects/MINI2440_BASE
□ Debug 版本
本地目录: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs
远程目录: /
设备文件系统:
□ 不支持符号链接
部署前执行档:

100%
22%

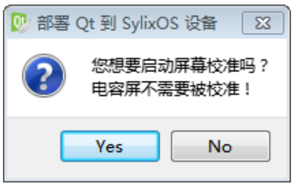
```
尝试连接到 SylixOS 设备: 192.168.7.30
连接成功。
正在登录...
登录成功。
开始上传文件...

创建目录: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/apps -> //apps
创建目录: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/apps/mousecalibration -> /apps/mousecalibration
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/apps/mousecalibration/mousecalibration -> /apps/mousecalibration/
mousecalibration
创建目录: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/etc -> //etc
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/etc/fs_init.sh -> /etc/fs_init.sh
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/etc/fs_no_symlink_init.sh -> /etc/fs_no_symlink_init.sh
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/etc/pointercal -> /etc/pointercal
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/etc/qtcp_4.8.7.sh -> /etc/qtcp_4.8.7.sh
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/etc/qtenv.sh -> /etc/qtenv.sh
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/etc/qtln_4.8.7.sh -> /etc/qtln_4.8.7.sh
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/etc/startup.sh -> /etc/startup.sh
创建目录: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/qt -> //qt
创建目录: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/qt/lib -> /qt/lib
创建目录: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/qt/lib/fonts -> /qt/lib/fonts
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/qt/lib/fonts/DejaVuSans.ttf -> /qt/lib/fonts/DejaVuSans.ttf
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/qt/lib/libQt3Support.so.4.8.7 -> /qt/lib/libQt3Support.so.4.8.7
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/qt/lib/libQtCLucene.so.4.8.7 -> /qt/lib/libQtCLucene.so.4.8.7
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/qt/lib/libQtCore.so.4.8.7 -> /qt/lib/libQtCore.so.4.8.7
上传文件: C:/Qt/qtcreator-3.4.0/SylixOS/sdk/arm-sylixos-qt-4.8.7/fs/qt/lib/libQtDeclarative.so.4.8.7 -> /qt/lib/libQtDeclarative.so.
4.8.7
```

g

Figure 4.3 Start to deploy Qt shared library

4.2 Screen calibration

When the deployment and initialization of file system is done, the screen calibration inquiry dialog box would pop out, as shown by figure 4.4.



部署 Qt 到 SylixOS 设备

您想要启动屏幕校准吗？
电容屏不需要被校准！

Figure 4.4 Screen calibration inquiry dialog box

If your SylixOS device uses the resistive screen, please click the “Yes” button to start the screen calibration; if your SylixOS device uses the capacitive screen, please click “No” button to skip the screen calibration.

Click the “+” symbol in the LCD screen to complete the screen calibration in accordance with the reminders in the LCD screen, as shown by figure 4.5.

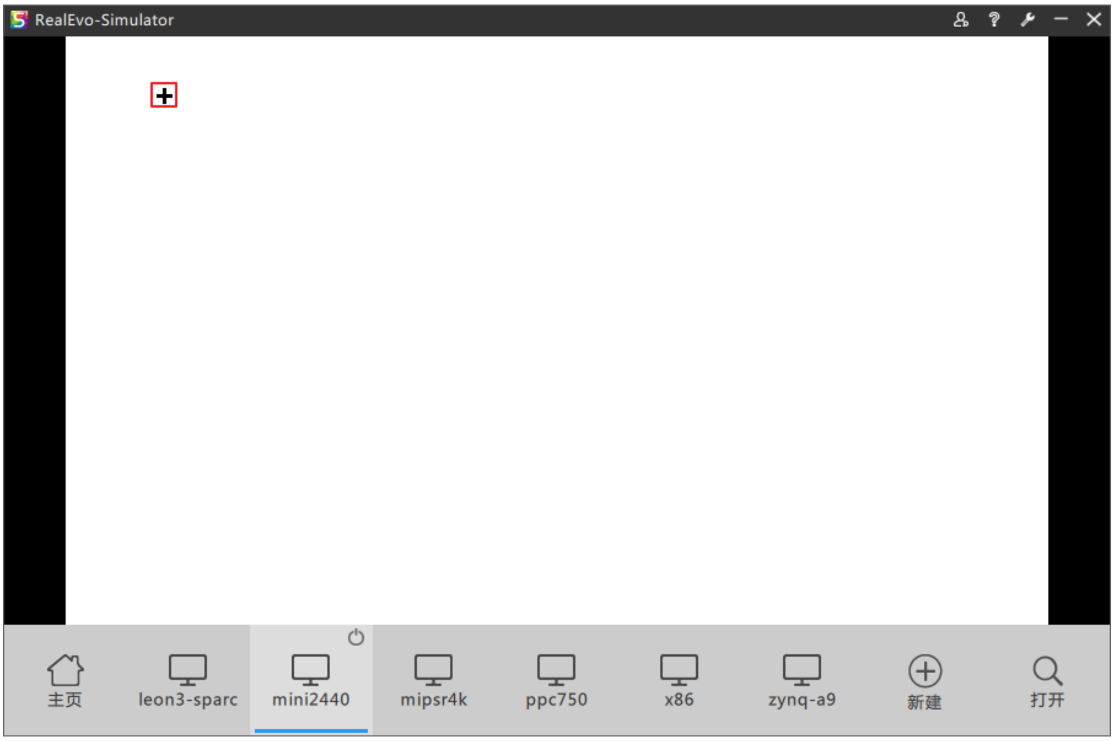


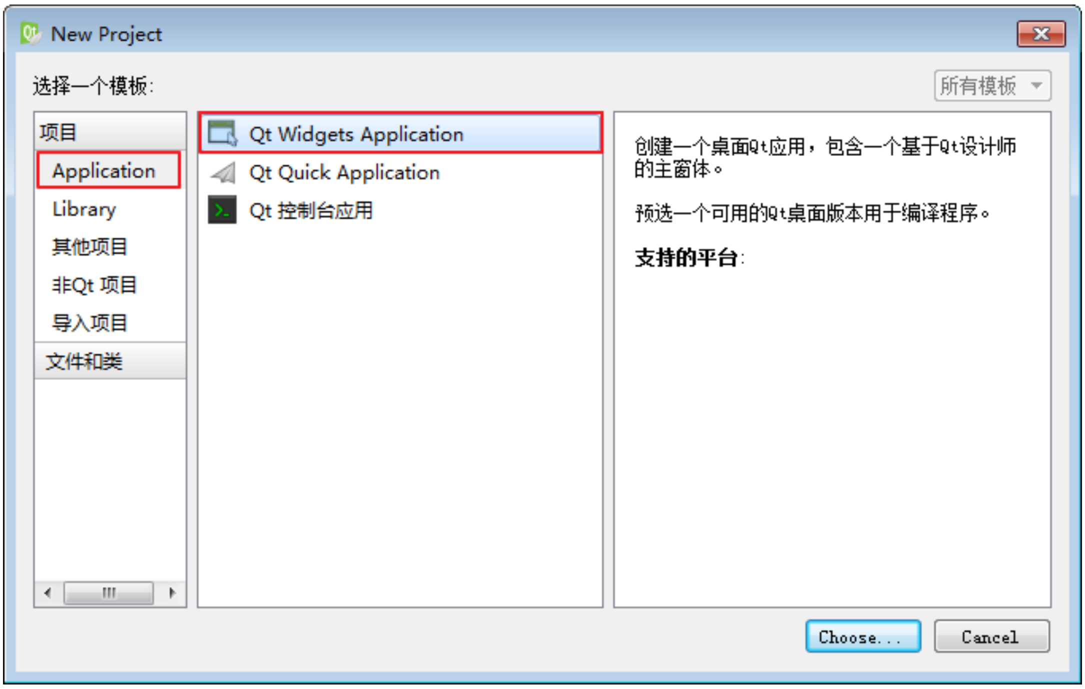
Figure 4.5 Screen calibration

Note: The “leon3-sparc”, “mipsr4k” and “ppc750” in the simulator platform do not support the display function and “x86” does not support the mouse function.

Chapter 5 Develop Qt Widgets Application Program

5.1 New Qt Widgets Application Program

Click the menu “file → new file or project..”, the new-built dialog box would pop out, as shown by figure 5.1.



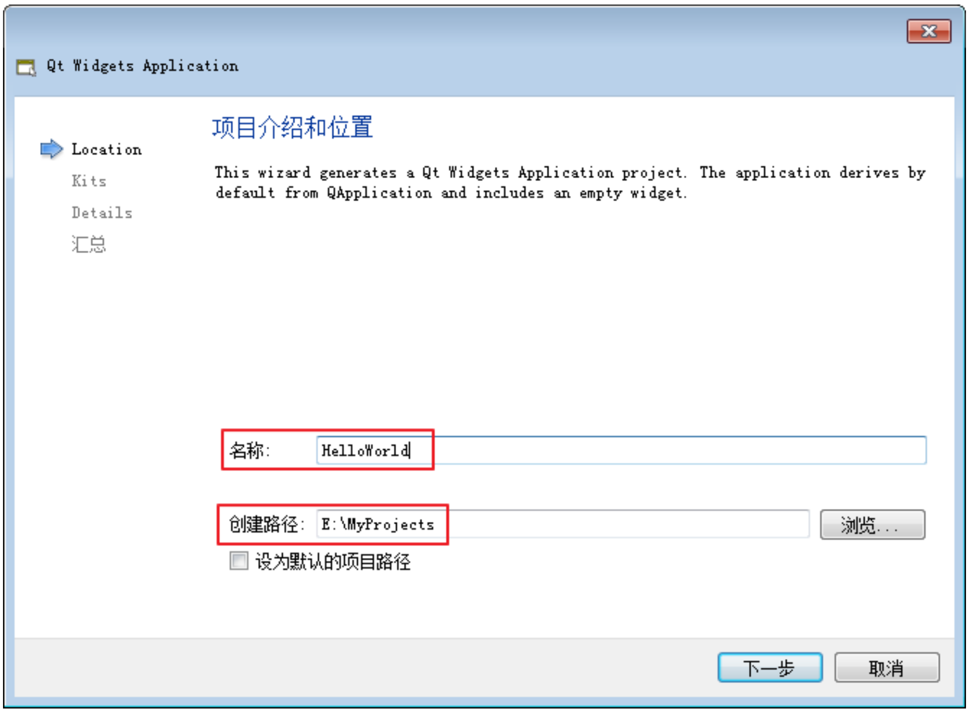
创建一个桌面Qt应用，包含一个基于Qt设计师的主窗体。

预选一个可用的Qt桌面版本用于编译程序。

支持的平台:

Figure 5.1 Newly-built dialog box

Select the “Qt Widgets Application” template in the “Application” of project type, and then click “Choose...” button to enter the “Qt Widgets Application” guide, as shown by figure 5.2.



Qt Widgets Application

项目介绍和位置

This wizard generates a Qt Widgets Application project. The application derives by default from QApplication and includes an empty widget.

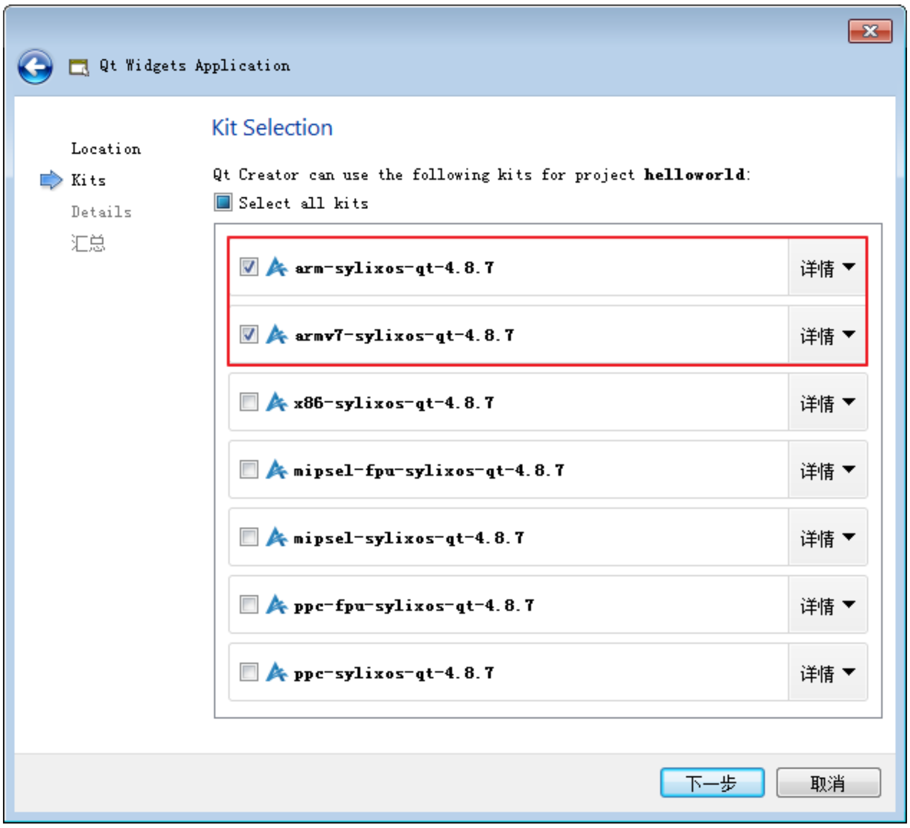
名称: HelloWorld

创建路径: E:\MyProjects

□ 设为默认的项目路径

Figure 5.2 “Qt Widgets Application” guide

Input the project name of “HelloWorld”; create path: click “browse...” button to select an appropriate project storage file folder, such as “E:\MyProjects” folder, then click “next” button to enter the next step, as shown by figure 5.3.



Qt Widgets Application

Kit Selection

Qt Creator can use the following kits for project **helloworld**:

■ Select all kits

☑ arm-sylixos-qt-4.8.7 详情

☑ armv7-sylixos-qt-4.8.7 详情

□ x86-sylixos-qt-4.8.7 详情

□ mipsel-fpu-sylixos-qt-4.8.7 详情

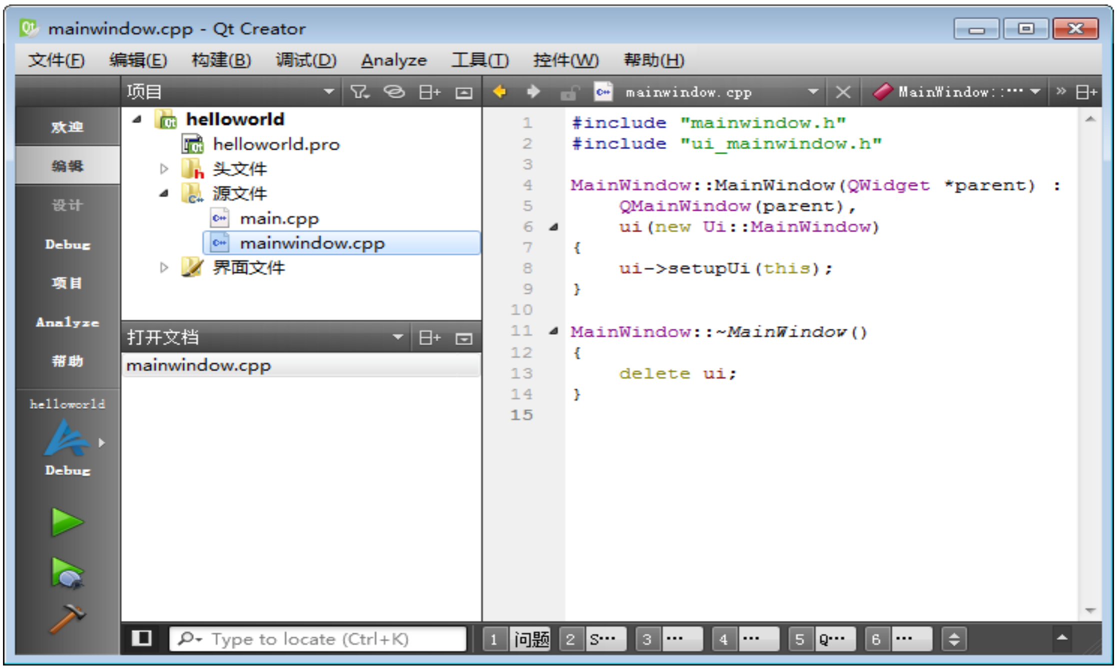
□ mipsel-sylixos-qt-4.8.7 详情

□ ppc-fpu-sylixos-qt-4.8.7 详情

□ ppc-sylixos-qt-4.8.7 详情

Figure 5.3 Kit selection

Kit could select “arm-sylixos-qt-4.8.7” and “armv7-sylixos-qt-4.8.7”, or all select, then click “next” button to finish the project creation as per the default process. After finishing the creation, HelloWorld project is shown by figure 5.4.



```
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);
}

MainWindow::~MainWindow()
{
    delete ui;
}
```

Figure 5.4 HelloWorld project

5.2 Revise Qt Widgets Application Program

Double click the mainwindow.ui file (as shown by figure 5.5) in the project file browser interface file direcctory, it could open the interface designer, as shown by figure 5.6.

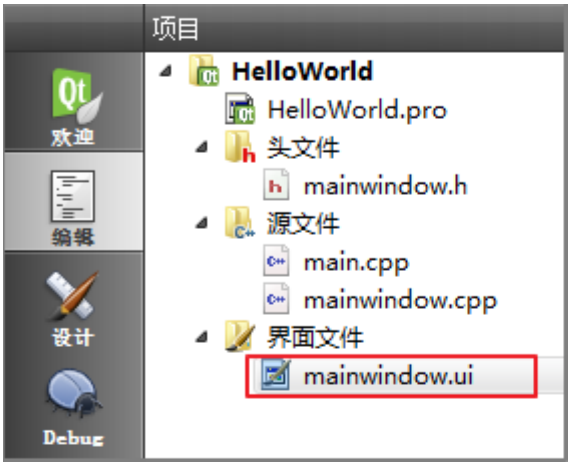


Figure 5.5 Mainwindow.ui file

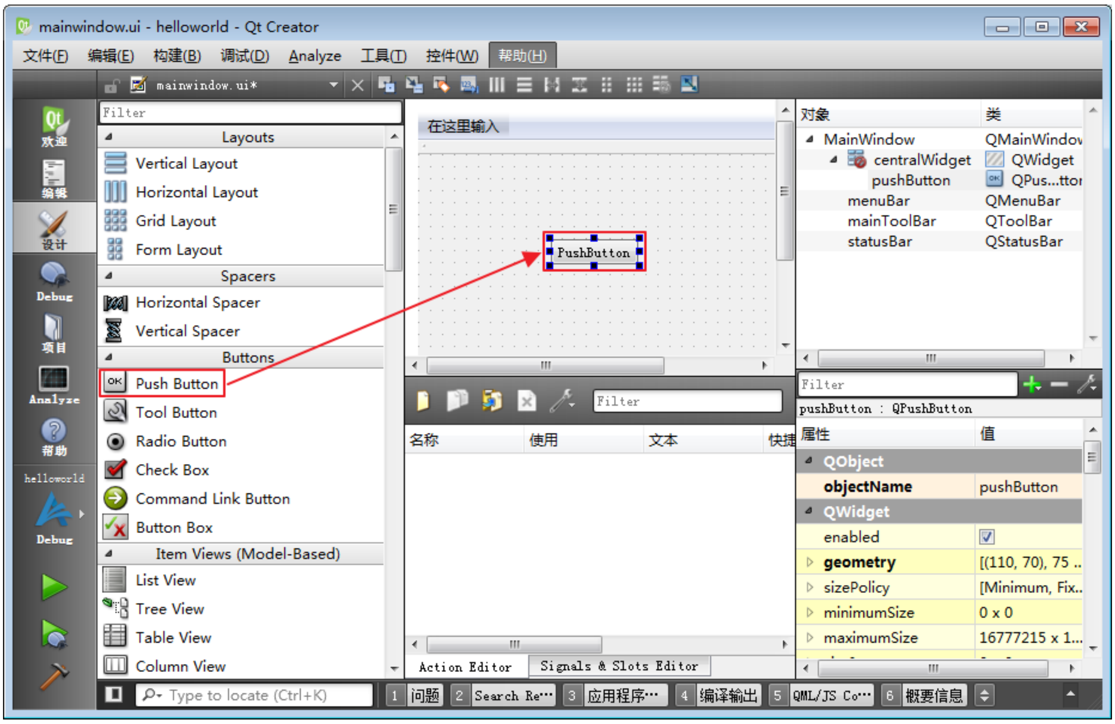
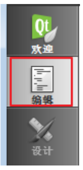


Figure 5.6 Interface designer

Draw a button of “Push Button” type to the middle of the window and store the file (Ctrl +S), click the compilation button  to switch back to the code compilation window.

Revise the mainwindow.h file as shown by the program list 5.1 (red bold font is the revised part, similarly hereinafter):

Program list 5.1 mainwindow.h file

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>

namespace Ui {
class MainWindow;
}

class MainWindow : public QMainWindow
{
   Q_OBJECT

public:
   explicit MainWindow(QWidget *parent = 0);
```

```
    ~MainWindow();

private slots:
    void btnClick();

private:
    Ui::MainWindow *ui;
};

#endif // MAINWINDOW_H
```

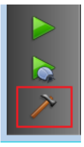
Revise mainwindow.cpp file as shown by the program list 5.2.

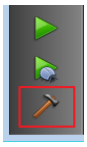
Program list 5.2　mainwindow.cpp file

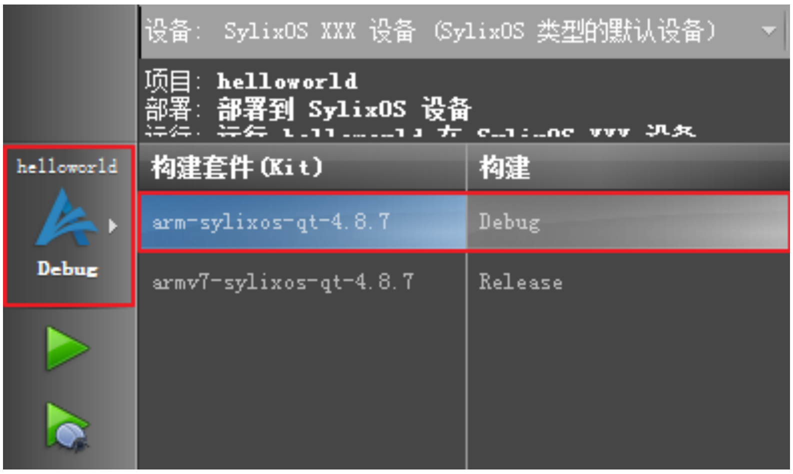
```
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    connect(ui->pushButton, SIGNAL(clicked()), this, SLOT(btnClick()));
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::btnClick()
{
    printf("hello Qt\n");
}
```

The code function is simple. There is only one button in the interface. When the button is pressed, the “hello Qt” in the console would be printed.

5.3 Build Qt Widgets Application Program

点击左下角的构建按钮，即可完成项目构建。当前使用的构建套件（Kit）是“arm-sylixos-qt-4.8.7”，构建的是 Debug 版本，点击按钮 可以完成构建套件（Kit）及版本切换，如图 5.7 所示。

Click the build button  at the left lower corner to build the project. The current building Kit is “arm-sylixos-qt-4.8.7” and the built version is Debug version. Click the button to switch between Kit and version, as shown by figure 5.7.



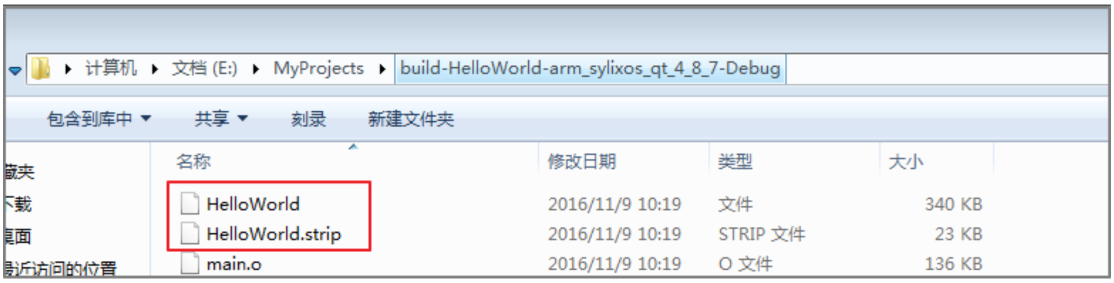
设备： SylixOS XXX 设备 (SylixOS 类型的默认设备)

项目： helloworld
部署： 部署到 SylixOS 设备

构建套件(Kit)	构建
arm-sylixos-qt-4.8.7	Debug
armv7-sylixos-qt-4.8.7	Release

Figure 5.7 Kit and version

The outcome of building is stored in the subfolder of “build-HelloWorld-arm_sylixos_qt_4_8_7-Debug” in the project storage folder. The subfolder name depends on your currently selected Kit and version. The folder content is shown by the figure 5.8.



计算机 ▸ 文档 (E:) ▸ MyProjects ▸ build-HelloWorld-arm_sylixos_qt_4_8_7-Debug

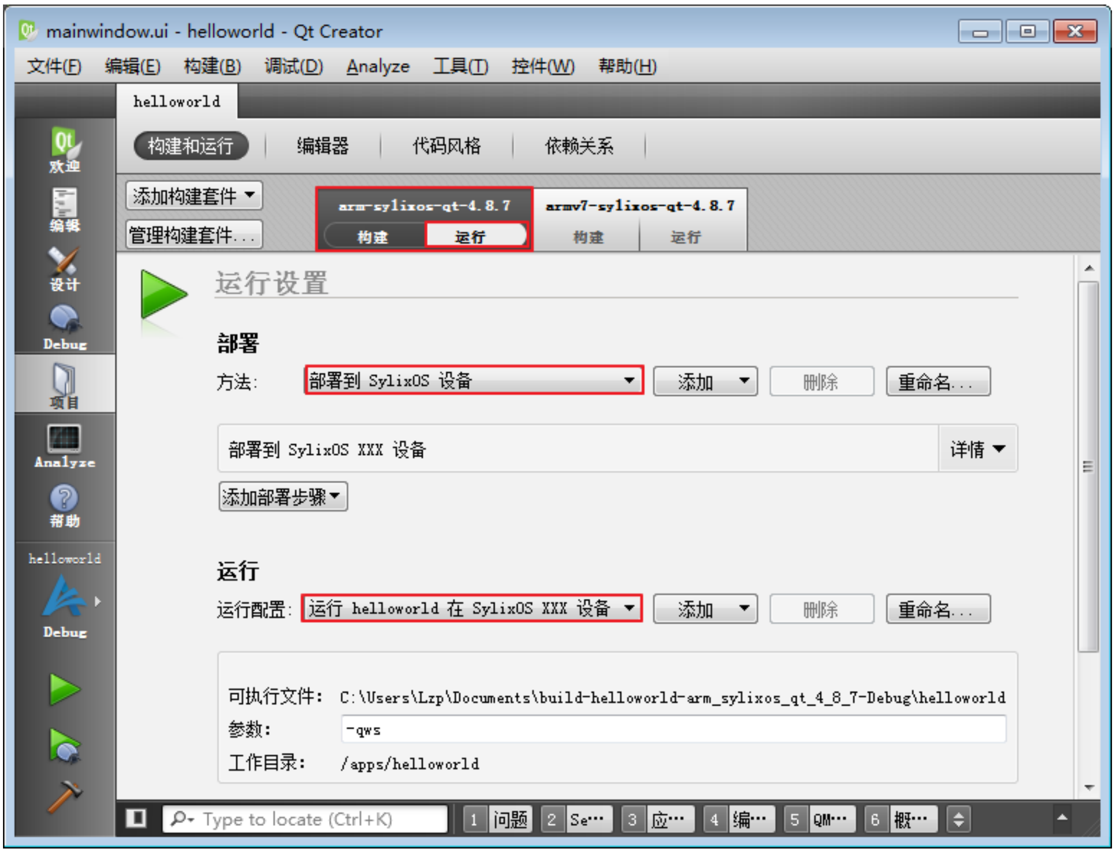
名称	修改日期	类型	大小
HelloWorld	2016/11/9 10:19	文件	340 KB
HelloWorld.strip	2016/11/9 10:19	STRIP 文件	23 KB
main.o	2016/11/9 10:19	O 文件	136 KB

Figure 5.8 Build result contents

Note: As shown by the figure 5.8, HelloWorld. Strip file is the HelloWorld executable file after unnecessary information is removed. This file is much smaller than HelloWord file and is applicable for being deployed to the device file system.

5.4 Run Qt Widgets Application Program

We should confirm the deployment configuration of the project is "deploy to the SylixOS device", the running configuration is "run HelloWorld in the SylixOS mini2440 device". Click the project button in the left side to enter the project setup page, as shown by figure 5.9.



运行设置

部署

方法: 部署到 SylixOS 设备

部署到 SylixOS XXX 设备

运行

运行配置: 运行 helloworld 在 SylixOS XXX 设备

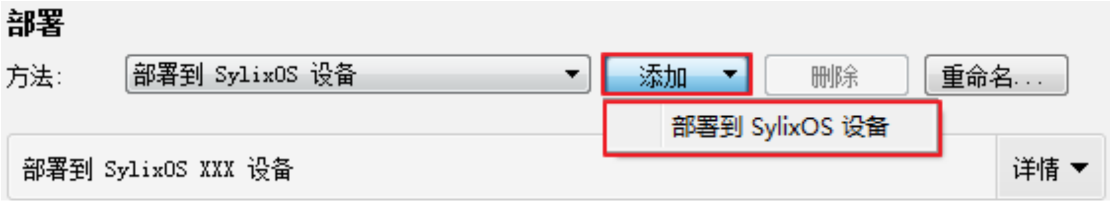
可执行文件: C:\Users\Lzp\Documents\build-helloworld-arm_sylixos_qt_4_8_7-Debug\helloworld

参数: -qws

工作目录: /apps/helloworld

Figure 5.9 Project setup page

Select the "run" button in the Kit "arm-sylixos-qt-4.8.7", confirm the deployment configuration of the project is "deploy to SylixOS device", the running configuration is "run the HelloWorld in the SylixOS mini2440 device"; othwerwise, click the "add" button, as shown by figure 5.10 and 5.11.

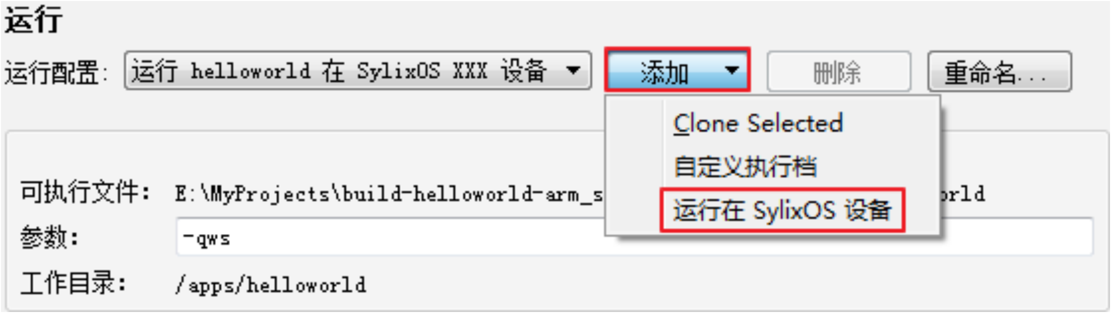


部署

方法: 部署到 SylixOS 设备

部署到 SylixOS XXX 设备

Figure 5.10 Add the deployment method



运行

运行配置: 运行 helloworld 在 SylixOS XXX 设备

可执行文件: E:\MyProjects\build-helloworld-arm_s orld

参数: -qws

工作目录: /apps/helloworld

Figure 5.11 Add running configuration

It could be found that the working directory of Qt application program is "/apps/HelloWorld" directory, the parameter is "-qws", if your Qt application program in the future has other parameters, it could be added to after "-qws" in the "parameter" input box.

It shall be noticed that the deployment procedures could also deploy the file or directory designated by the "INSTALLS" variable in the project file (such as the HelloWorld.pro file in the HelloWorld project) besides the executable file of the Qt application program. If your Qt application software has to deploy other file or directory, please add or revise the "INSTALLS" variable. "INSTALLS" variables' grammar is shown by program list 5.3.

Program list 5.3 "INSTALLS" variable grammar

```
sylixos {
XXX.files    = 本地文件或目录名
XXX.path     = SylixOS 设备文件系统路径
INSTALLS     += XXX
}
```

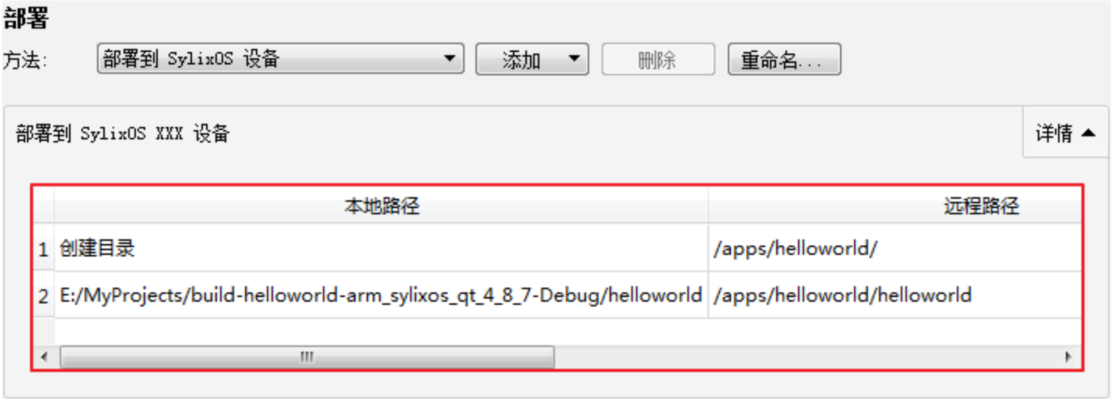
If an image directory imagedir and a configuration file config.xml in the HelloWorld project (the image directory and configuration file in the folder of HelloWorld program) shall be deployed to the "/apps/HelloWorld" directory in the SylixOS device; we should add the content shown by the program list 5.4 in the file of HelloWorld. pro.

Program list 5.4 "INSTALLS" variable example

```
sylixos {
imagedir.files      = imagedir
imagedir.path       = /apps/HelloWorld
configfile.files    = config.xml
configfile.path     = /apps/HelloWorld
INSTALLS            += imagedir configfile
}
```

Note: “sylixos {” in the starting line and “}” in the end line are not the must, but the content inside (especially the SylixOS device file system path) is only related with the SylixOS system platform. In order to make the Qt application program code cross platform better, it is suggested to add “sylixos {”in the starting line and “}” in the end line.

If it is needed to check which files and directories would be deployed to SylixOS device, click “details ▼” button, as shown by figure 5.12.

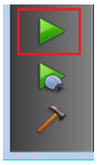


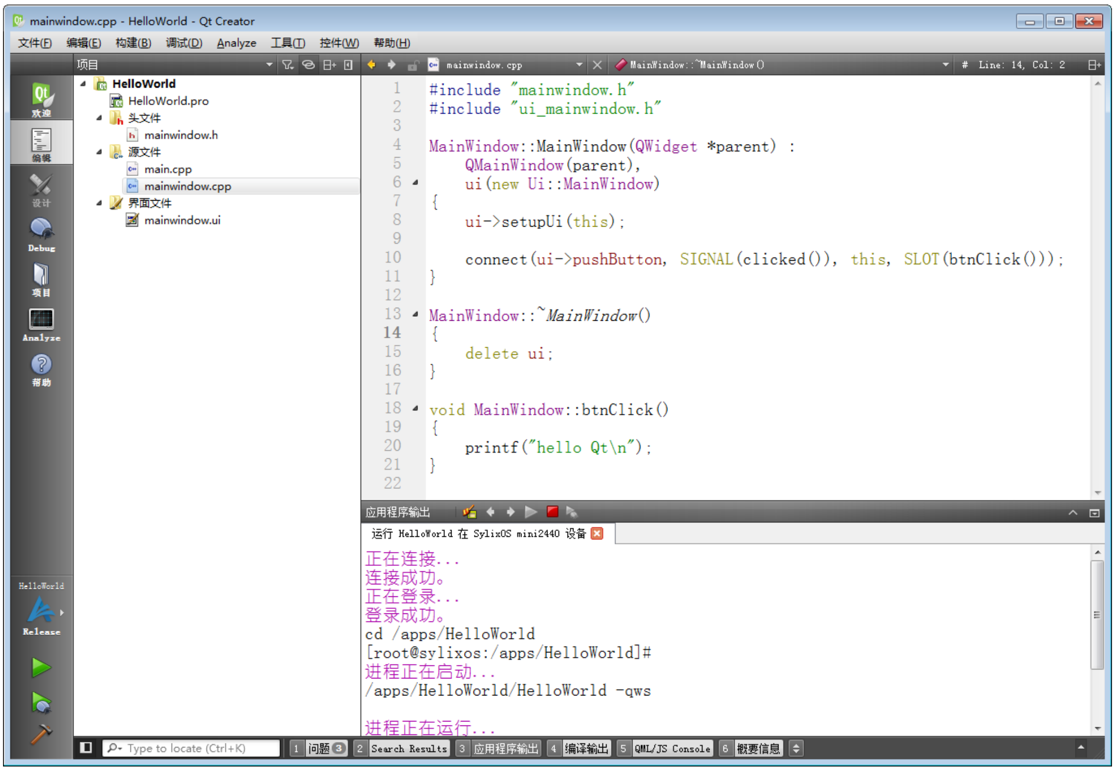
方法: 部署到 SylixOS 设备

部署到 SylixOS XXX 设备

	本地路径	远程路径
1	创建目录	/apps/helloworld/
2	E:/MyProjects/build-helloworld-arm_sylixos_qt_4_8_7-Debug/helloworld	/apps/helloworld/helloworld

Figure 5.12 Deployment file list

Click the run button  to deploy and run the application program, as shown by figure 5.13.



```
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    connect(ui->pushButton, SIGNAL(clicked()), this, SLOT(btnClick()));
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::btnClick()
{
    printf("hello Qt\n");
}
```

```
正在连接...
连接成功。
正在登录...
登录成功。
cd /apps/HelloWorld
[root@sylixos:/apps/HelloWorld]#
进程正在启动...
/apps/HelloWorld/HelloWorld -qws

进程正在运行...
```

Figure 5.13 Deploy application program

The LCD screen of the SylixOS device would display the main window of HelloWorld application program, as shown by figure 5.14.

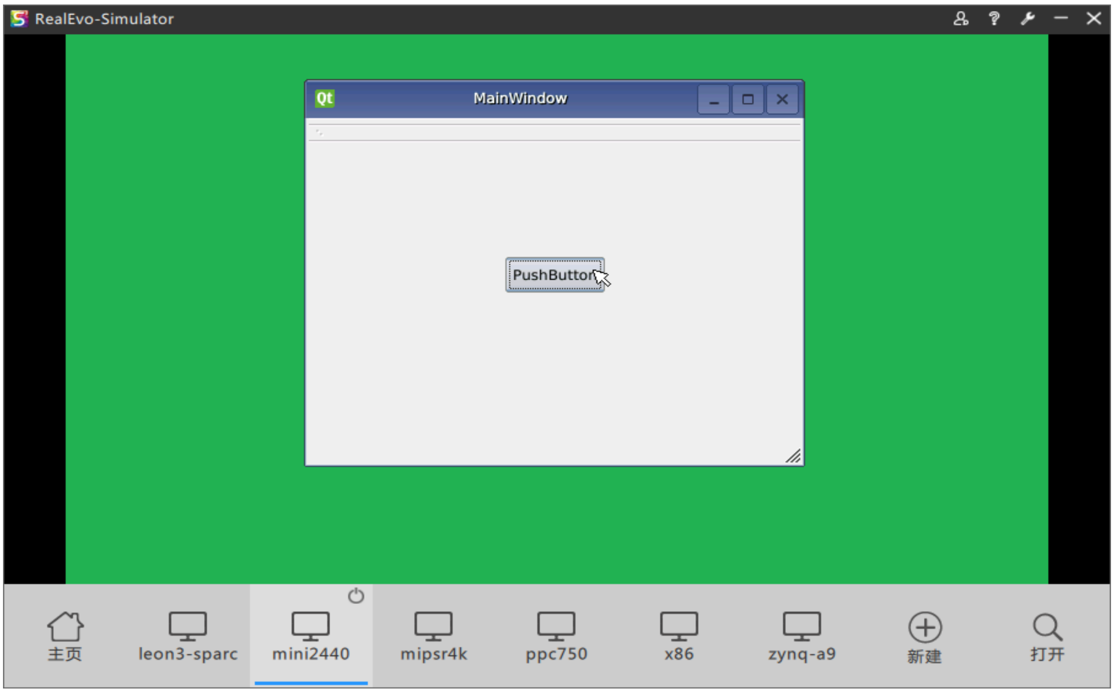
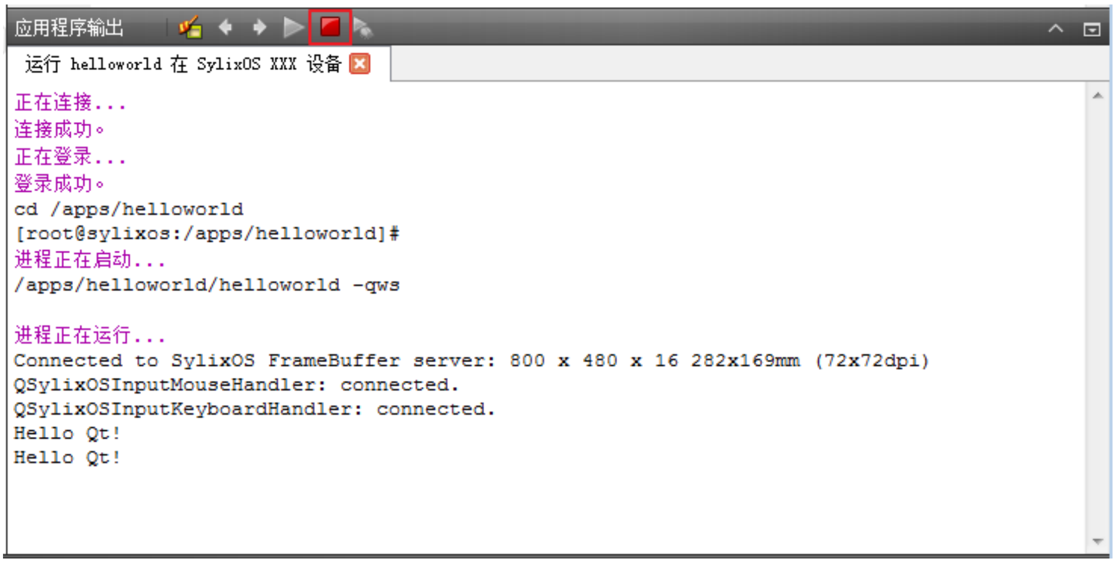


Figure 5.14 HelloWorld application program

When the button in HelloWorld application program is pressed, “hello Qt” would be outputted in the console of the HelloWorld application program of Qt Creator, as shown by figure 5.15.



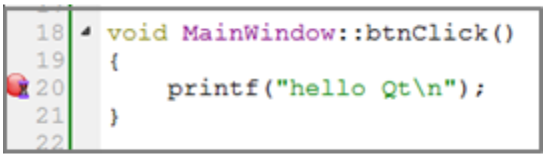
```
正在连接...
连接成功。
正在登录...
登录成功。
cd /apps/helloworld
[root@sylixos:/apps/helloworld]#
进程正在启动...
/apps/helloworld/helloworld -qws

进程正在运行...
Connected to SylixOS FrameBuffer server: 800 x 480 x 16 282x169mm (72x72dpi)
QSylixOSInputMouseHandler: connected.
QSylixOSInputKeyboardHandler: connected.
Hello Qt!
Hello Qt!
```

Figure 5.15　HelloWorld application program console

Click the red “stop” button in the console of HelloWorld application program to close the HelloWorld application program, as shown by figure 5.15.

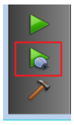
5.5 Debug Qt Widgets Application Program

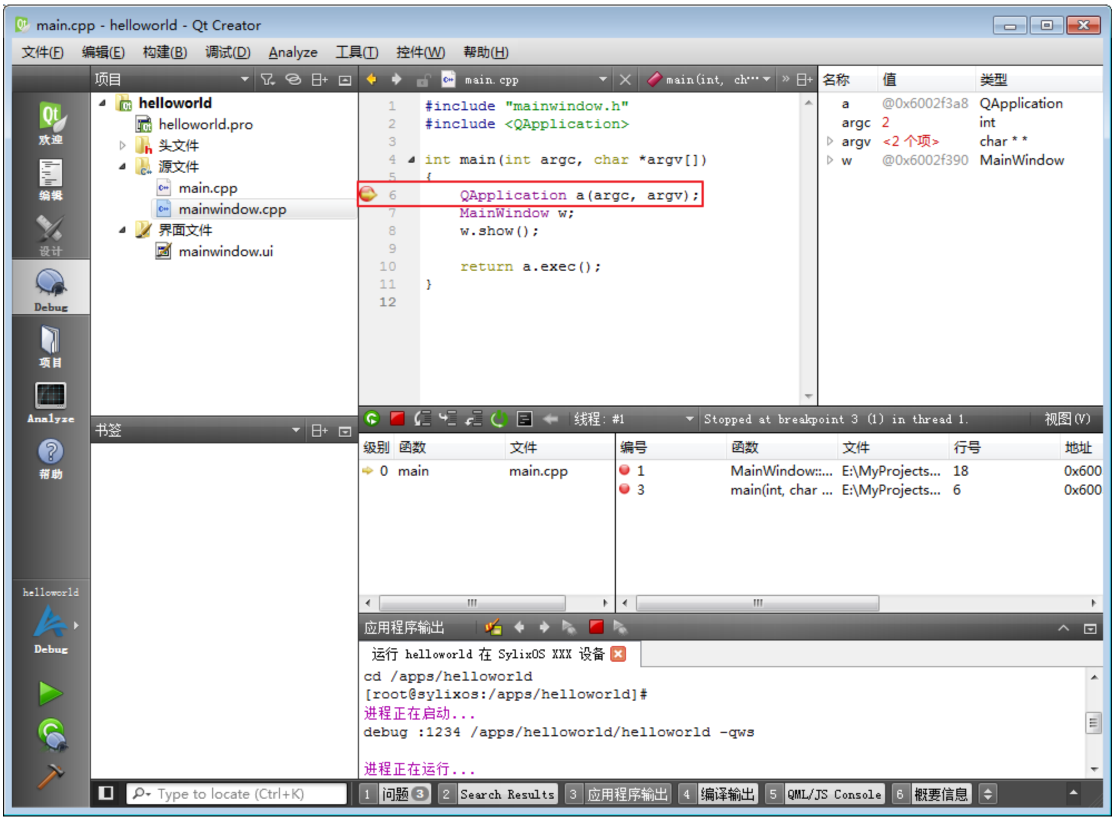
Add a breakpoint in the MainWindow::btnClick function of the mainWindow.cpp file (click the location of the small red ball in the figure 5.16 to add the breakpoint), as shown by figure 5.16.



```
void MainWindow::btnClick()
{
    printf("hello Qt\n");
}
```

Figure 5.16　Add breakpoint

Click the debugging button  (keyboard F5) to start deploy and debug the application program, as shown by figure 5.17.



```
#include "mainwindow.h"
#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;
    w.show();

    return a.exec();
}
```

名称	值	类型
a	@0x6002f3a8	QApplication
argc	2	int
argv	<2 个项>	char * *
w	@0x6002f390	MainWindow

线程: #1 Stopped at breakpoint 3 (1) in thread 1.

级别	函数	文件
0	main	main.cpp

编号	函数	文件	行号	地址
1	MainWindow::...	E:\MyProjects...	18	0x600
3	main(int, char ...	E:\MyProjects...	6	0x600

运行 helloworld 在 SylixOS XXX 设备

```
cd /apps/helloworld
[root@sylixos:/apps/helloworld]#
进程正在启动...
debug :1234 /apps/helloworld/helloworld -qws

进程正在运行...
```

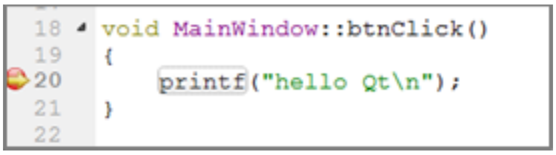
Figure 5.17 Debugging interface

The program stops at the entry point of the main function, we can see the local variable value in the variable window at the right side, and the call stack of current thread in the call stack window at the bottom.

Keyboard F10 is for step over, F11 is for step into, “Shift + F11” is for step out, “Ctrl + F10” is to run to the cursor line, F5 is to continue at full speed. Or click the debugging control tool bar at the top of the call stack window to control the debugging process.

Press the keyboard F5 to run with full speed, the LCD screen of SylixOS device would show the main window of HelloWorld application program, as shown by figure 5.14.

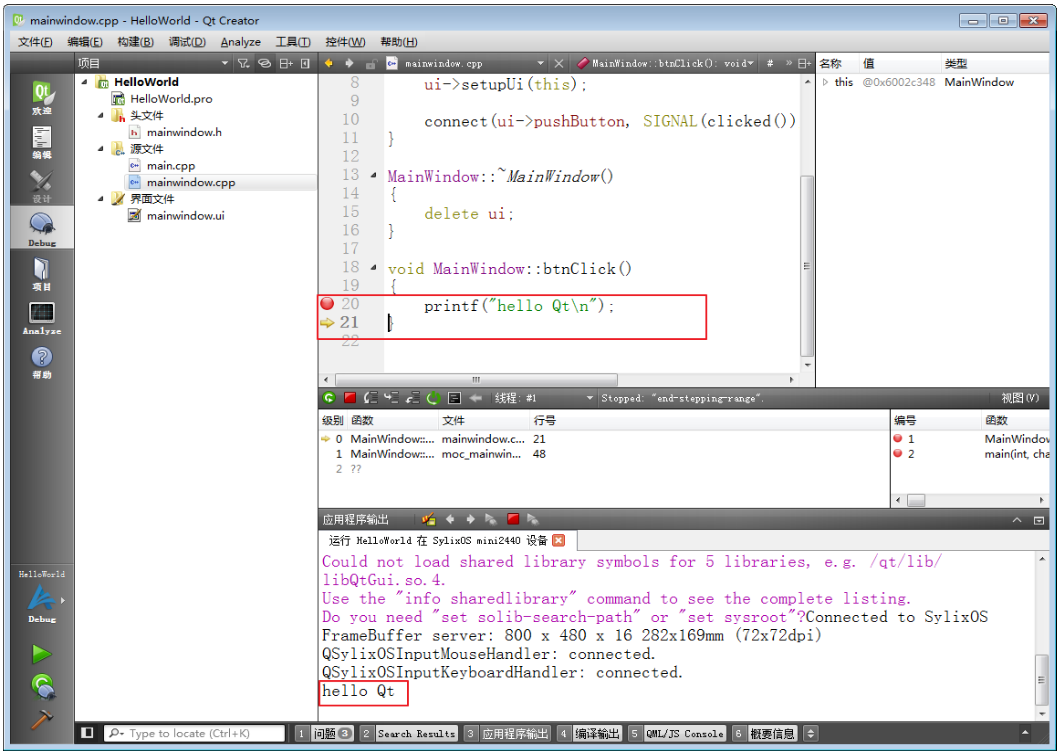
As we have added a breakpoint in the function of MainWindow::btnClick, when the button in the HelloWorld application program is pressed, the program would stop at this breakpoint, as shown by figure 5.18.



```
void MainWindow::btnClick()
{
    printf("hello Qt\n");
}
```

Figure 5.18 Breakpoint arrives

Press the keyboard F10 for step over, “hello Qt” would be outputted on the console of the HelloWorld application program in Qt Creator, as shown by figure 5.19.



```
    ui->setupUi(this);

    connect(ui->pushButton, SIGNAL(clicked())
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::btnClick()
{
    printf("hello Qt\n");
}
```

```
this  @0x6002c348  MainWindow
```

```
Stopped: "end-stepping-range".
```

级别	函数	文件	行号
0	MainWindow::...	mainwindow.c...	21
1	MainWindow::...	moc_mainwin...	48
2	??		

编号	函数
1	MainWindov
2	main(int, cha

```
Could not load shared library symbols for 5 libraries, e.g. /qt/lib/
libQtGui.so.4.
Use the "info sharedlibrary" command to see the complete listing.
Do you need "set solib-search-path" or "set sysroot"?Connected to SylixOS
FrameBuffer server: 800 x 480 x 16 282x169mm (72x72dpi)
QSylixOSInputMouseHandler: connected.
QSylixOSInputKeyboardHandler: connected.
hello Qt
```

Figure 5.19　Single-step run

Click the menu “debugging → stop debugging” to stop debugging and close the HelloWorld application program, as shown by figure 5.20.

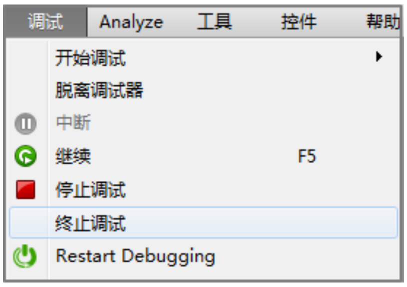


Figure 5.20　Stop debugging

Note: There is problem for the current QtCreator restarting “start debugging” after “stop debugging”, please always use “terminate debugging” approach to end a debugging process.

5.6 Attach Approach to Debug Qt Widgets Application Program

Over the development process, we might come across such a case: one Qt Widgets application program has run a period of time in the SylixOS device, but there is problem in the running logic of this application program due to the problems in the code compilation of this application program. We hope that this application program can be connected for debugging through which the defects in the code of the application program can be found.

RealEvo-QtSylixOS supports to debug a running Qt Widgets application program with the Attach approach.

To illustrate the usage of debugging the Qt Widgets application program with Attach, we assume the following several points:

- SylixOS device has run the HelloWorld application program of the Debug version of the HelloWorld project;
- The current configuration of the HelloWorld project is the Debug version;
- The Kit's SylixOS device used by HelloWorld project is the SylixOS device in need of debugging;
- RealEvo-QtSylixOS has opened the HelloWorld project and set the HelloWorld project as current active project, as shown by the figure 5.21.

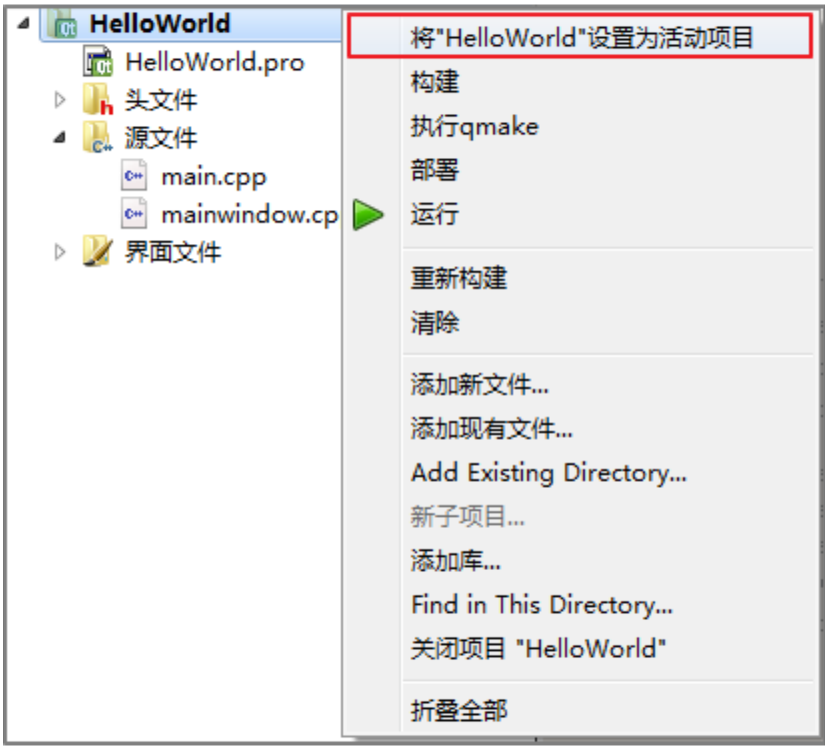


Figure 5.21 Set active project

Click the menu “debugging → start debugging → Attach to the remote SylixOS application program...”, as shown by figure 5.22.

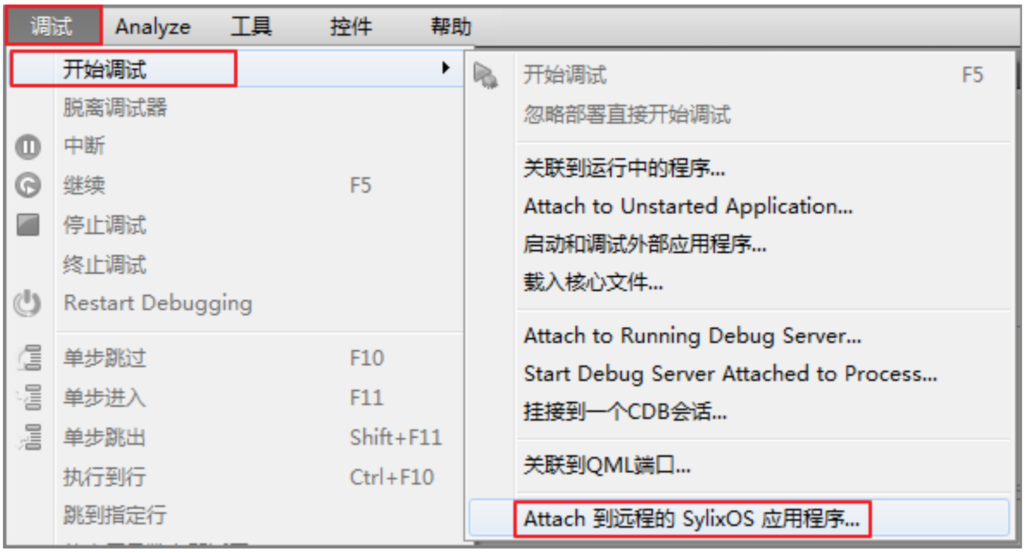
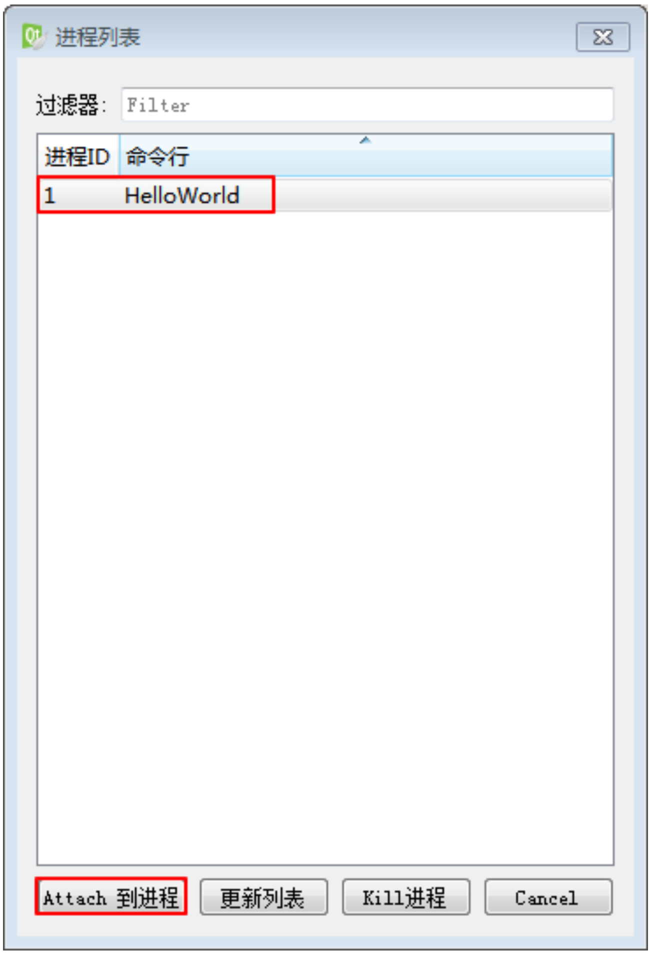


Figure 5.22 Attach application program

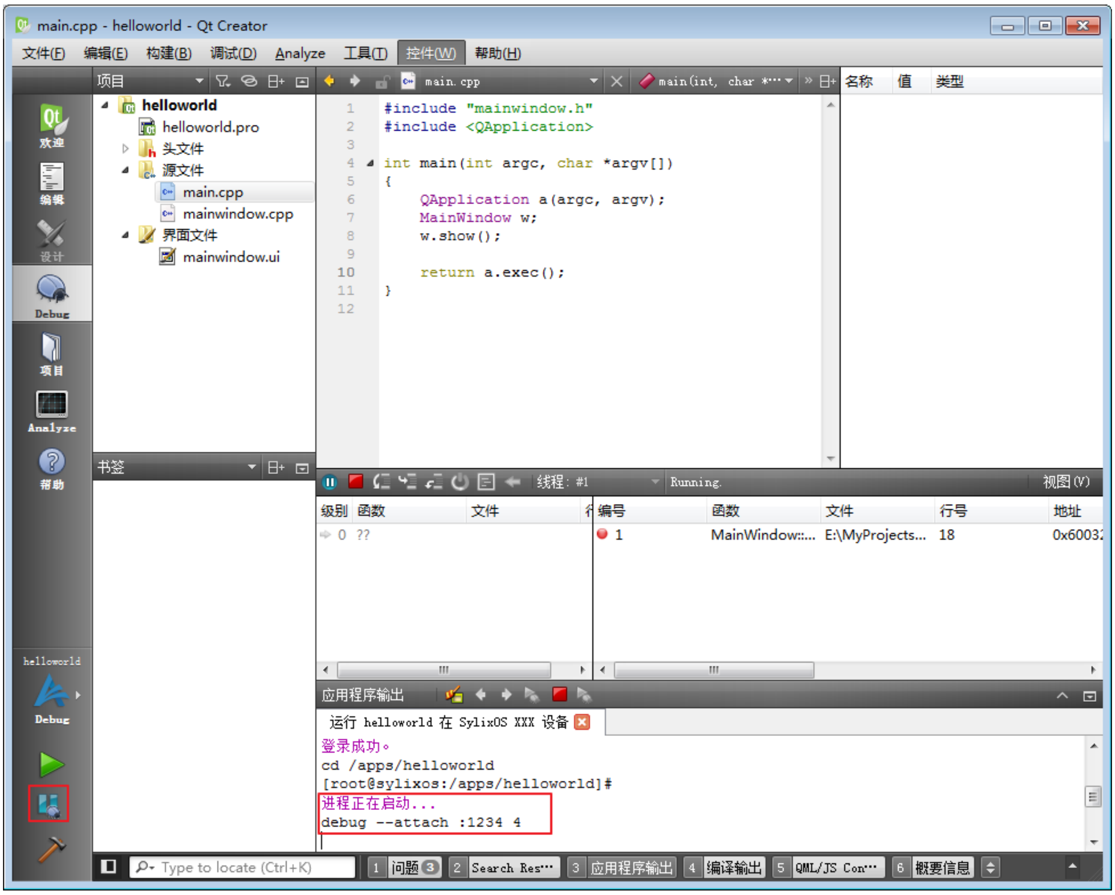
The process list dialog box would pop out, as shown by figure 5.23.



进程ID	命令行
1	HelloWorld

Figure 5.23 process list dialog box

The process currently run in the SylixOS device would be shown in the process list. Select the target process “HelloWorld” and then click “Attach to process” button to start the debugging, as shown by figure 5.24.



```
#include "mainwindow.h"
#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;
    w.show();

    return a.exec();
}

```

级别	函数	文件
0	??	

编号	函数	文件	行号	地址
● 1	MainWindow::...	E:\MyProjects...	18	0x6003:

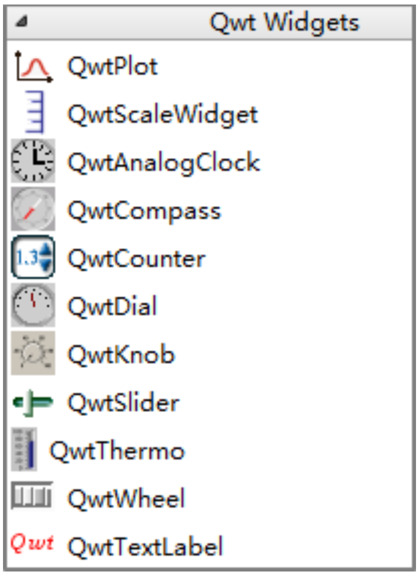
```
登录成功。
cd /apps/helloworld
[root@sylixos:/apps/helloworld]#
进程正在启动...
debug --attach :1234 4
```

Figure 5.24 Start debugging

The debugging method is the same as that in section 5.5, which would not be illustrated here. The different point is: click the menu “debugging → Terminate debugging” to terminate debugging. However, the HelloWorld application program would not close and would continue to run in the SylixOS device.

5.7 Add the Qwt in the Qt Widgets Application Program

If the Qwt window widgets shown by the figure 5.25 is used in the interface file (such as the mainwindow.ui in the HelloWorld project) of Qt Widgets application program,



Qwt Widgets

QwtPlot

QwtScaleWidget

QwtAnalogClock

QwtCompass

QwtCounter

QwtDial

QwtKnob

QwtSlider

QwtThermo

QwtWheel

QwtTextLabel

Figure 5.25 Qwt window widgets

It's requied to revise the project file (such as the HelloWorld.pro file in the HelloWorld project) by adding the content as shown by the program list 5.5.

Program list 5.5 link parameter

```
sylixos {
LIBS        += -lqwt
INCLUDEPATH += $(QTDIR)/../qwt
}
```

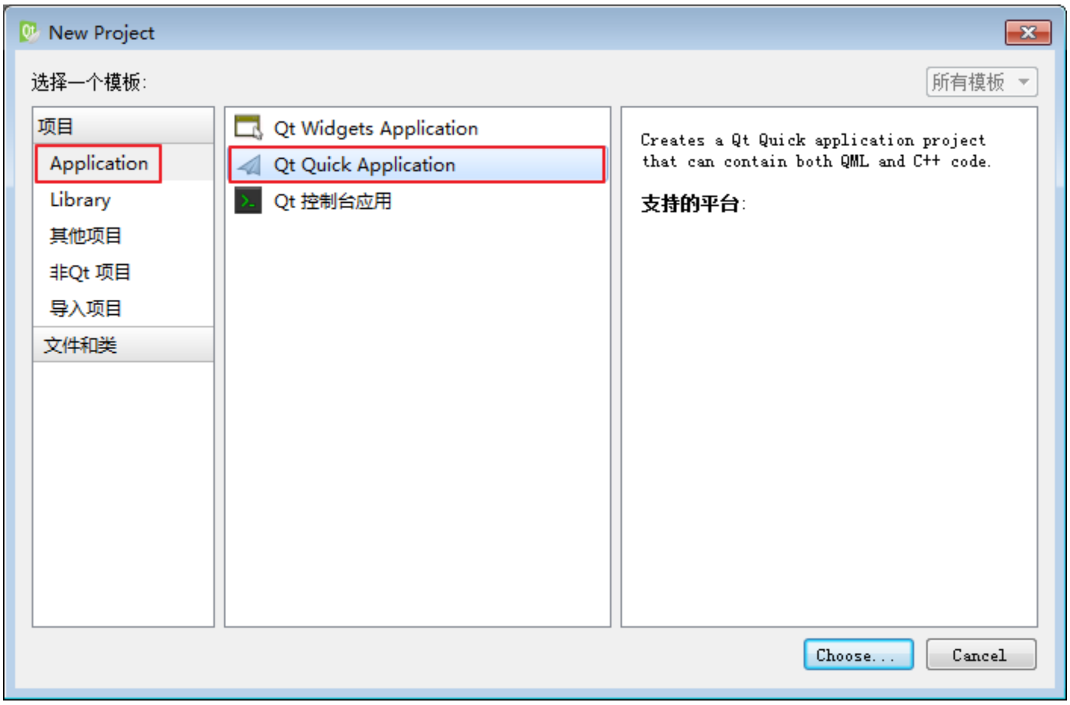
Among which, "LIBS" variable designates the link parameters of the Qt application program: -lxxx denotes the linked libxxx.so shared library, -Lyyy denotes searching shared library in the yyy directory. "INCLUDEPATH" variable designates the search path for the header file.

Note: If your Qt application program needs linking to other shared library or static library, revision can be made in accordance with the above instructions.

Chapter 6 Develop Qt Quick Application Program

6.1 New Qt Quick Application Program

Click menu “File → new file or project...”, the new dialog box would pop out, as shown by figure 6.1.

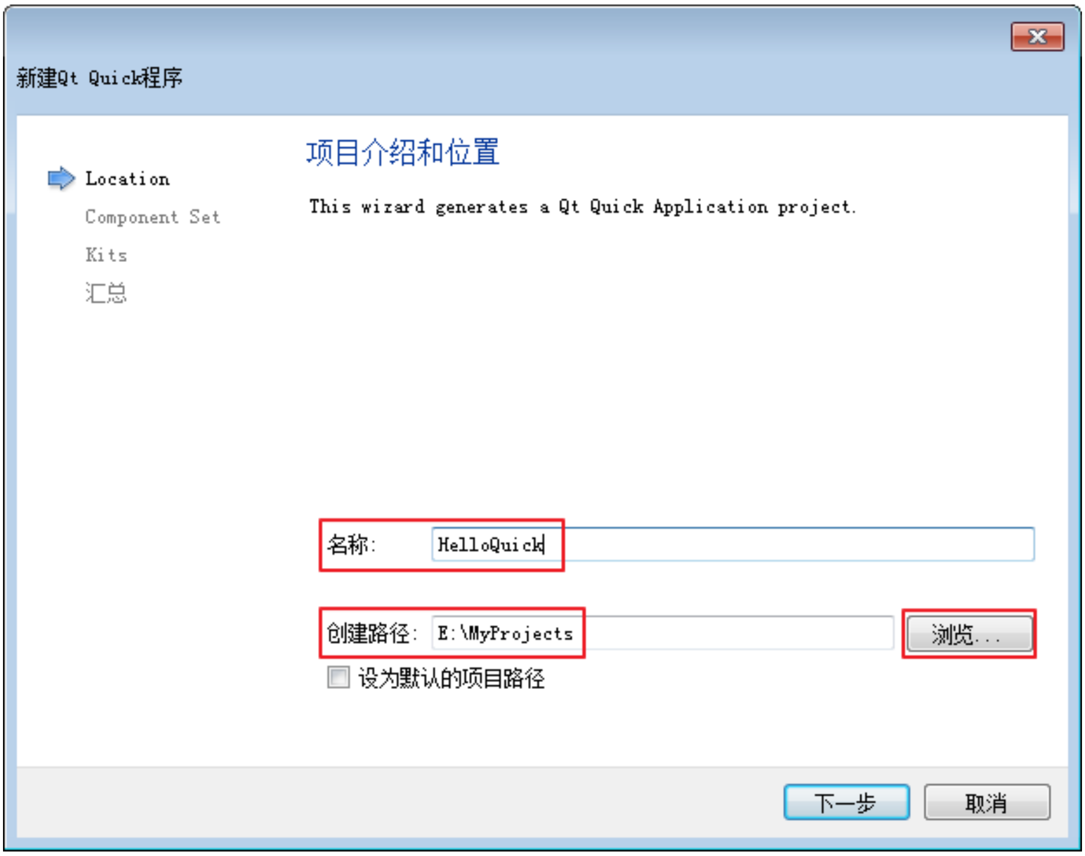


Creates a Qt Quick application project that can contain both QML and C++ code.

支持的平台:

Figure 6.1 New dialog box

Select the “Qt Quick Application” template in the project type “Application”, and then click “Choose...” button to enter “Qt Quick Application” guide, as shown by figure 6.2.



新建Qt Quick程序

项目介绍和位置

This wizard generates a Qt Quick Application project.

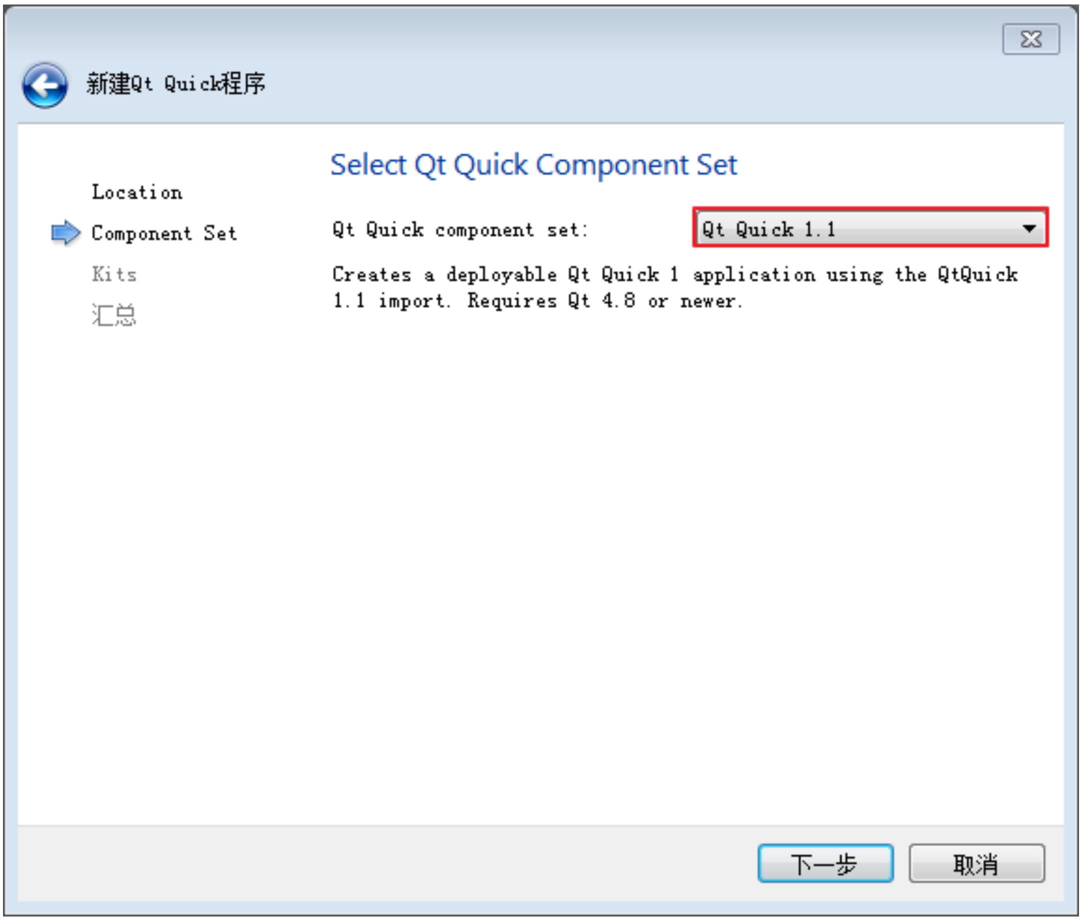
名称: HelloQuick

创建路径: E:\MyProjects

□ 设为默认的项目路径

Figure 6.2 “Qt Quick Application” guide

Input project name as “HelloQuick”, create path by clicking “browse...” button to select an appropriate project storage location, such as “E:\MyProjects” directory, and then click “next” button to enter into the next step, as shown by figure 6.3.



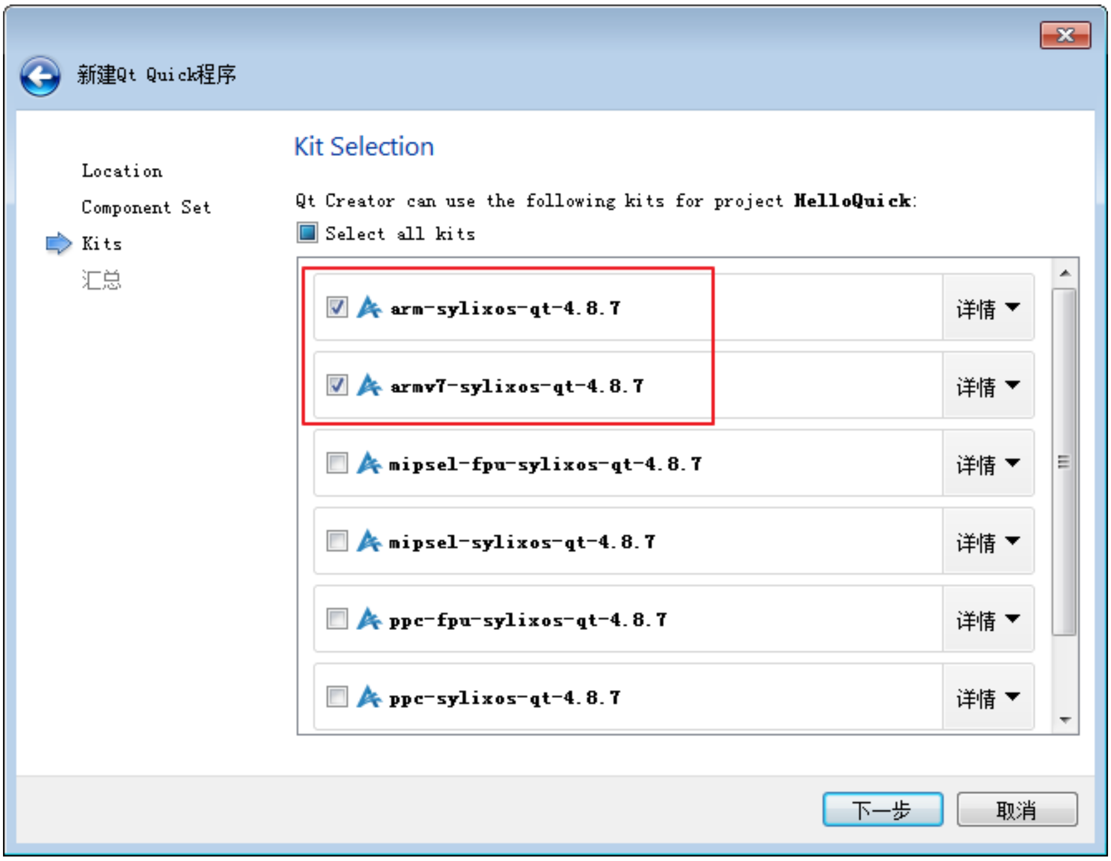
Select Qt Quick Component Set

Qt Quick component set: Qt Quick 1.1

Creates a deployable Qt Quick 1 application using the QtQuick 1.1 import. Requires Qt 4.8 or newer.

Figure 6.3 Qt Quick module selection

Select “Qt Quick component set” as “Qt Quick 1.1”, and then click “next” button to enter the next step, as shown by figure 6.4.



Kit Selection

Qt Creator can use the following kits for project **HelloQuick**:

□ Select all kits

☑ arm-sylixos-qt-4.8.7

☑ armv7-sylixos-qt-4.8.7

□ mipsel-fpu-sylixos-qt-4.8.7

□ mipsel-sylixos-qt-4.8.7

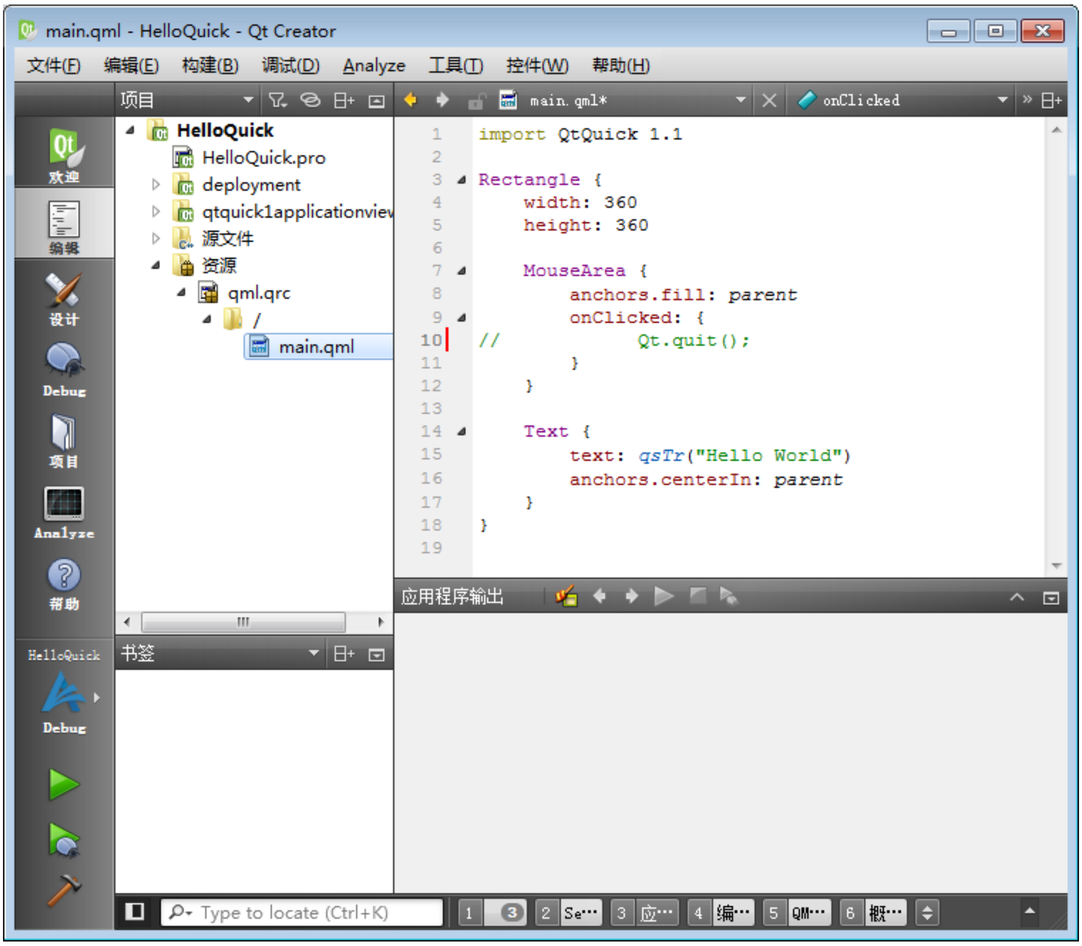
□ ppc-fpu-sylixos-qt-4.8.7

□ ppc-sylixos-qt-4.8.7

Figure 6.4 Kit selection

For Kit, you may select “arm-sylixos-qt-4.8.7” or “armv7-sylixos-qt-4.8.7”, or select all, then click “next” button to finish the project creation as per default procedures. After the project creation, HelloQuick project is shown as the figure 6.5.

Note: If no Kit of SylixOS could be found, it’s usually because that the “Qt Quick component set” is not selected as “Qt Quick 1.1”.



```
import QtQuick 1.1

Rectangle {
    width: 360
    height: 360

    MouseArea {
        anchors.fill: parent
        onClicked: {
//          Qt.quit();
        }
    }

    Text {
        text: qsTr("Hello World")
        anchors.centerIn: parent
    }
}
```

Figure 6.5 HelloQuick project

6.2 Revise Qt Quick Application Program

To avoid the problems that the HelloQuick application program exits actively when clicking the main interface of HelloQuick application program, it's required to revise the main.qml file in the resources, comment out the “Qt.quit();” statement in the “onClicked” processing, as shown by figure 6.5.

The compilation, running and debugging methods of Qt Quick application program fit the Qt Widgets application program. You could refer to sections 5.3, 5.4, 5.5 respectively, and no iteration here. The analysis method of Qt Quick application program would be described in the following text.

6.3 Analyze Qt Quick Application Program

Click the menu “analyze → QML Profiler”, as shown by figure 6.6, to start deploy and analyze application program, as shown by figure 6.7.

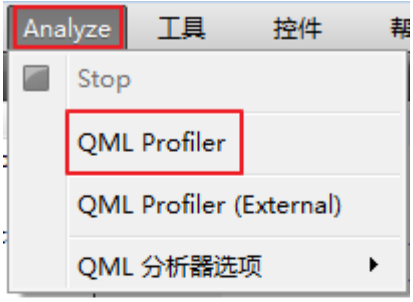
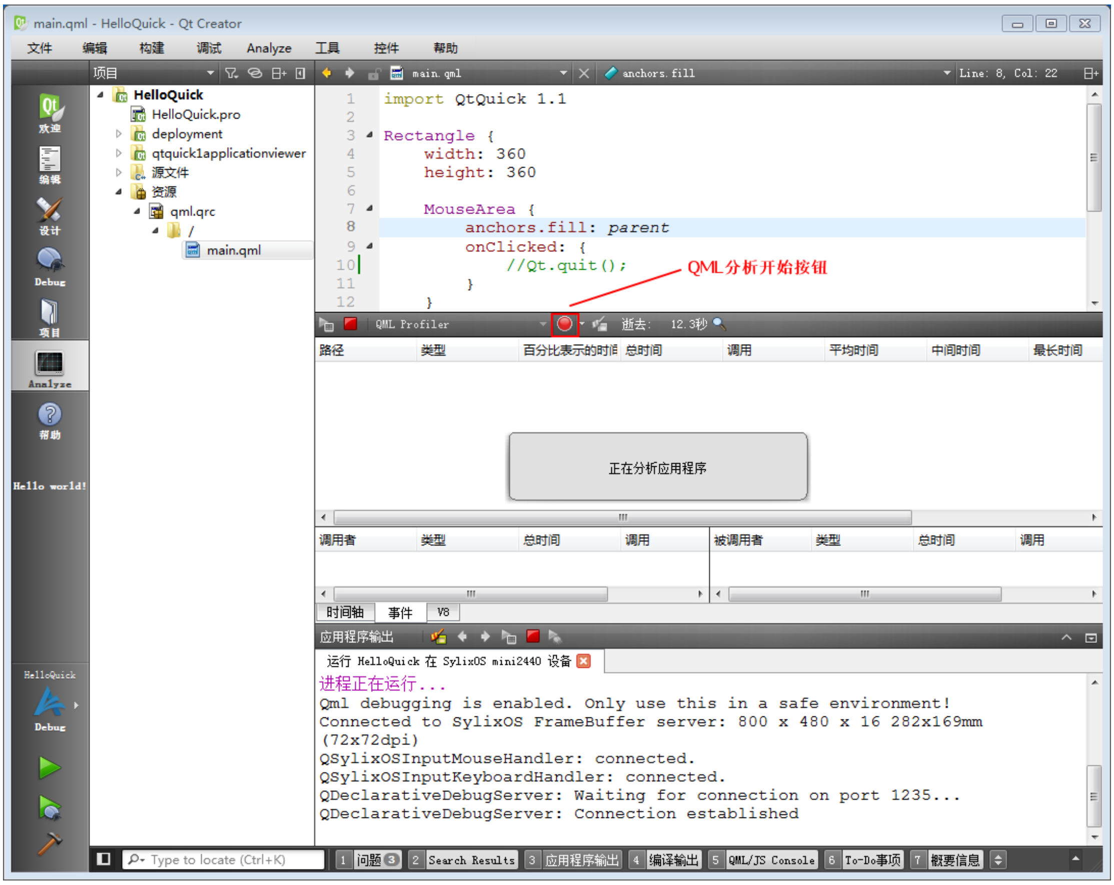


Figure 6.6 QML Profiler menu



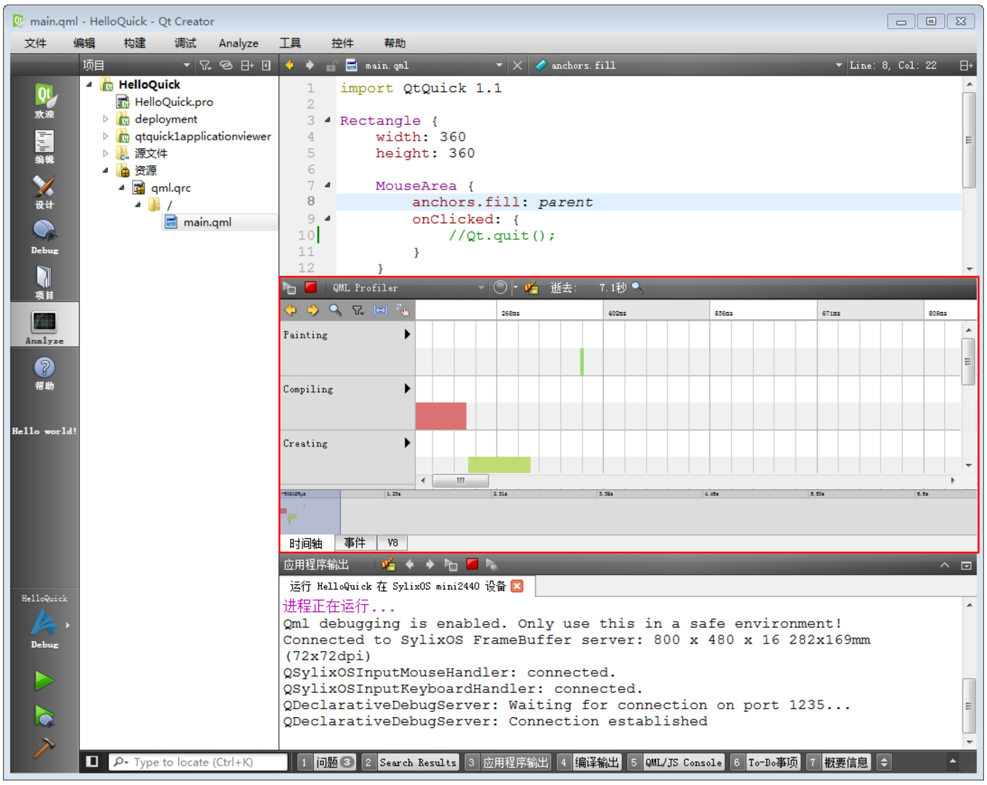
```
import QtQuick 1.1

Rectangle {
    width: 360
    height: 360

    MouseArea {
        anchors.fill: parent
        onClicked: {
            //Qt.quit();
        }
    }
```

```
进程正在运行...
Qml debugging is enabled. Only use this in a safe environment!
Connected to SylixOS FrameBuffer server: 800 x 480 x 16 282x169mm
(72x72dpi)
QSylixOSInputMouseHandler: connected.
QSylixOSInputKeyboardHandler: connected.
QDeclarativeDebugServer: Waiting for connection on port 1235...
QDeclarativeDebugServer: Connection established
```

Figure 6.7 QML analysis interface

When the application program runs, if QML analyzer does not start to work, click the “start” button in the QML analyzer, then the QML analyzer would start work.

Operate the HelloQuick application program in the LCD screen of SylixOS device. After a while, such as 10 seconds later, click the “start” button in the QML analyzer to stop analysis, QML analyzer would display the time consuming data and the occurred events of the HelloQuick application program over the past 10 seconds in graphic mode, as shown by figure 6.8.

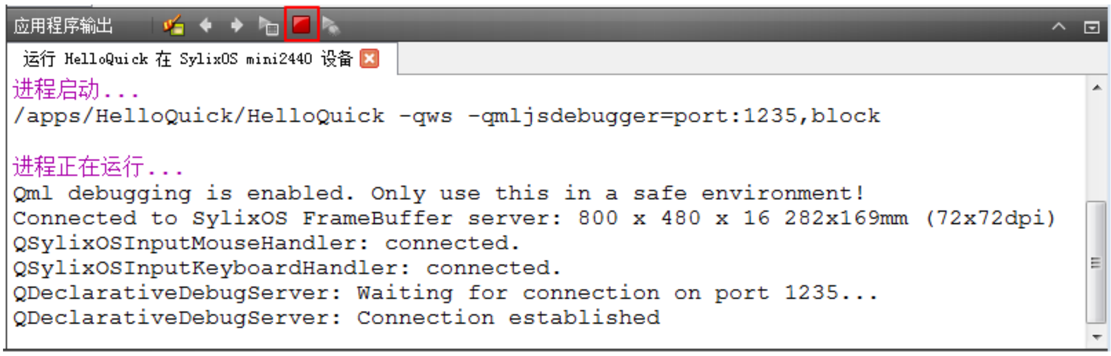


```
import QtQuick 1.1

Rectangle {
    width: 360
    height: 360

    MouseArea {
        anchors.fill: parent
        onClicked: {
            //Qt.quit();
        }
    }
```

```
运行 HelloQuick 在 SylixOS mini2440 设备
进程正在运行...
Qml debugging is enabled. Only use this in a safe environment!
Connected to SylixOS FrameBuffer server: 800 x 480 x 16 282x169mm
(72x72dpi)
QSylixOSInputMouseHandler: connected.
QSylixOSInputKeyboardHandler: connected.
QDeclarativeDebugServer: Waiting for connection on port 1235...
QDeclarativeDebugServer: Connection established
```

Figure 6.8 QML analysis result

Click the red “stop” button in the HelloQuick application program console window, the HelloQuick application program would be closed, as shown by figure 6.9.



```
运行 HelloQuick 在 SylixOS mini2440 设备
进程启动...
/apps/HelloQuick/HelloQuick -qws -qmljsdebugger=port:1235,block

进程正在运行...
Qml debugging is enabled. Only use this in a safe environment!
Connected to SylixOS FrameBuffer server: 800 x 480 x 16 282x169mm (72x72dpi)
QSylixOSInputMouseHandler: connected.
QSylixOSInputKeyboardHandler: connected.
QDeclarativeDebugServer: Waiting for connection on port 1235...
QDeclarativeDebugServer: Connection established
```

Figure .9 Close button

Chapter 7 Develop Projects of Other Types

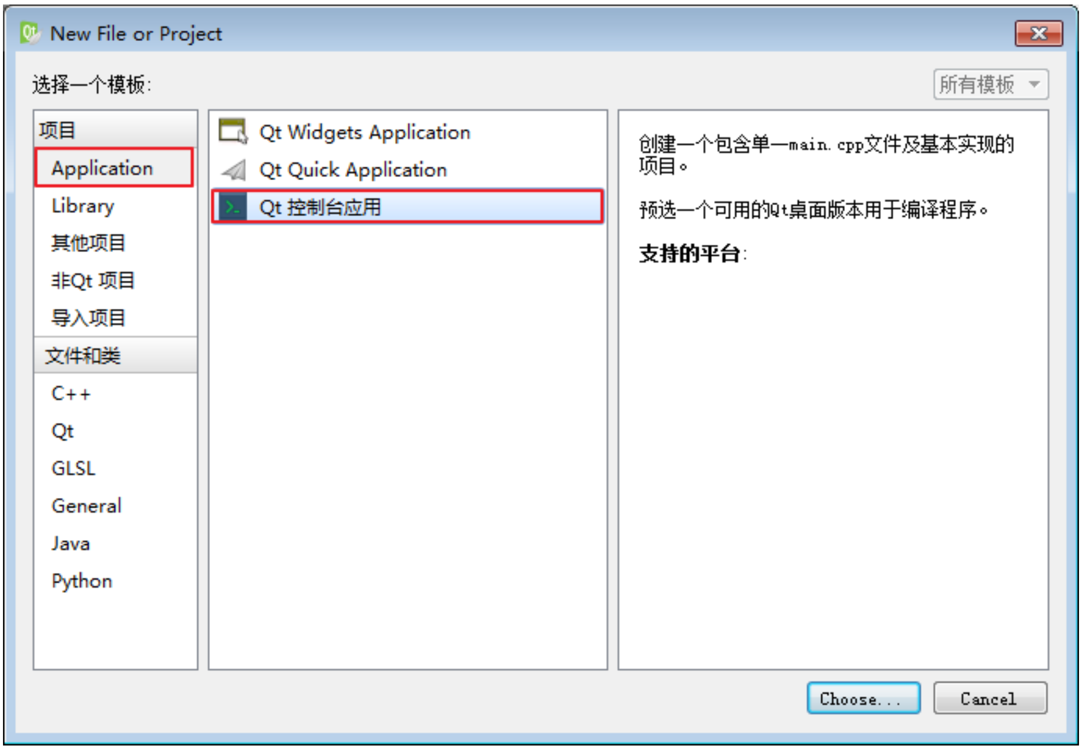
7.1 Develop Qt Console Application Program

RealEvo-QtSylixOS supports the development of Qt console application program. The differences between Qt console application program and Qt Widgets application program are: Qt console application program could not use QtGui module (such as: Qt widgets QWidget, Qt dialog QDailog, Qt main window QMainWindow), but it could still use modules of QtCore, QtNetwork, Qt database QtSql.

Qt console application program new processes are as follows:

- Click menu “file → new file or project..”, the new dialog box would pop out, select the “Qt console application” template in the file type “Application”, as shown by figure 7.1.
- Select the Kit of SylixOS in the follow-up “Kit Selection” dialog box (such as “arm-sylixos-qt-4.8.” and “armv7-sylixos-qt-4.8.7”)

The compilation, running and debugging methods of Qt console application program fit the Qt Widgets application program. Details can be found in section 5.3, 5.4, and 5.5.



创建一个包含单一main.cpp文件及基本实现的项目。

预选一个可用的Qt桌面版本用于编译程序。

支持的平台:

Figure 7.1 Qt console application

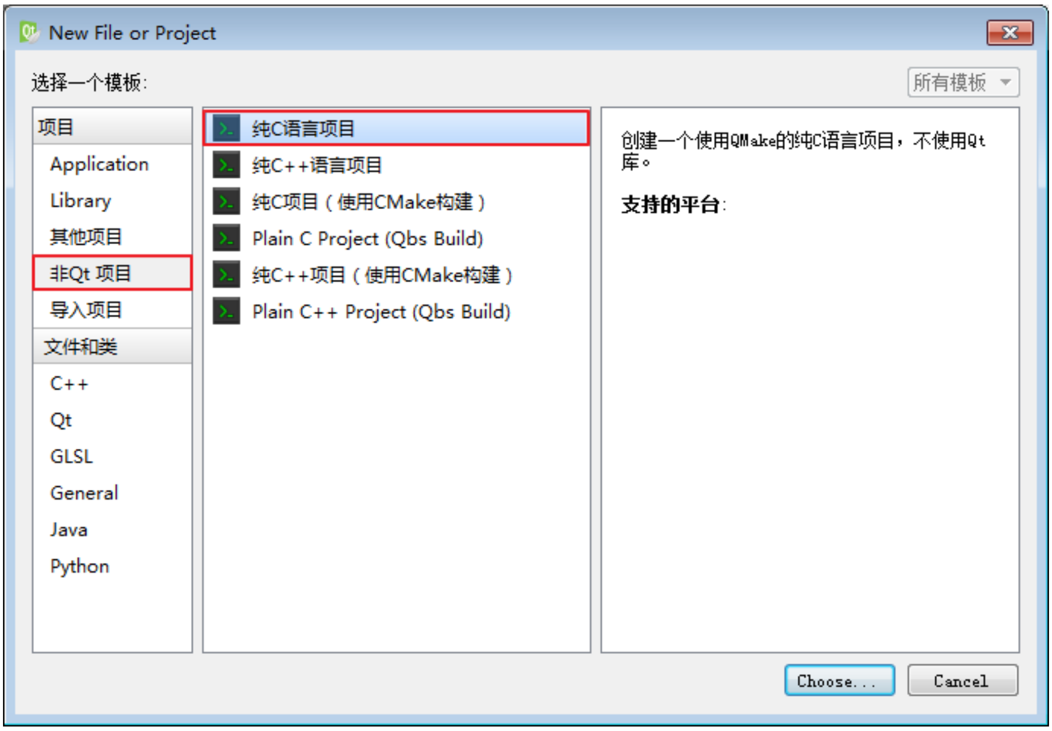
7.2 Develop SylixOS Pure C Language Program

RealEvo-QtSylixOS supports the development of SylixOS pure C language program. SylixOS pure C language program uses qmake to generate Makefile, so it is exempted from the compilation of Makefile. SylixOS pure C language program could not use any Qt module.

Process for the new SylixOS pure C language project is as follows:

- Click the menu “file → new file or project...”, the new built dialog box would pop out, select the “pure C language project” template in the project type “non-Qt project”, as shown by figure 7.2.
- Select the Kit of SylixOS in the following up “Kit Selection” dialog box (such as “arm-sylixos-qt-4.8.7” and “armv7-sylixos-qt-4.8.7”).

The compilation, running and debugging methods of SylixOS pure C language project fit the Qt Widgets application program development. Details can be found in section 5.3, 5.4, and 5.5.



创建一个使用QMake的纯C语言项目，不使用Qt库。

支持的平台:

Figure 7.2 Pure C language project

7.3 Develop SylixOS pure C++ Language Project

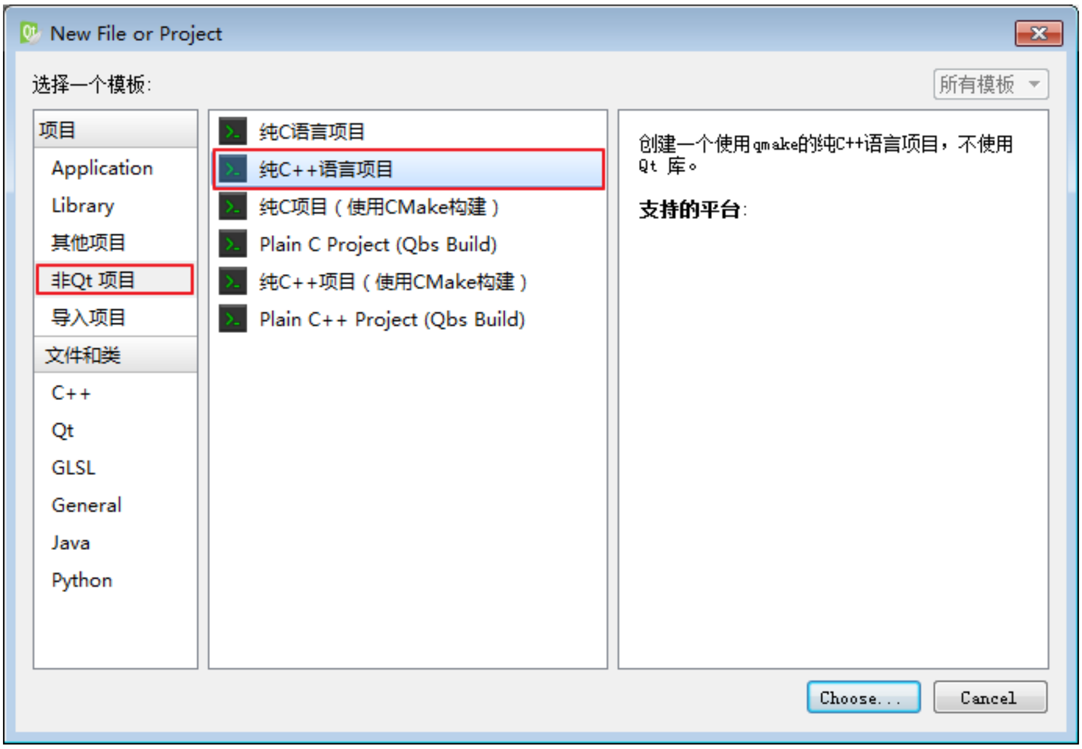
RealEvo-QtSylixOS supports the development of SylixOS pure C++ language project. SylixOS pure C++ language project uses qmake to generate Makefile, so it is exempted from the compilation of Makefile. SylixOS pure C++ language project could not use any Qt module.

The process for the building the new SylixOS pure C++ language project is as follows:

- Click the menu "file → new file or project...", the new-built dialog box would pop out. Select the "pure C++ language project" template in the project type "non-Qt project", as shown by figure 7.3.

- Select the SylixOS Kit in the follow-up "Kit Selection" dialog box (such as "arm-sylixos-qt-4.8.7" and "armv7-sylixos-qt-4.8.7")

The compilation, running and debugging methods of SylixOS pure C++ language project fits the Qt Widgets application program development. Details can be found in section 5.3, 5.4, and 5.5.



创建一个使用qmake的纯C++语言项目，不使用Qt 库。

支持的平台:

Figure 7.3 Pure C++ language project

7.4 Develop SylixOS Shared Library Project

RealEvo-QtSylixOS supports the development of SylixOS shared library project. SylixOS shared library project uses qmake to generate Makefile, so it is exempted from the compilation of Makefile. SylixOS shared library project could use any Qt module.

The process for building the new SylixOS shared library project is as follows:

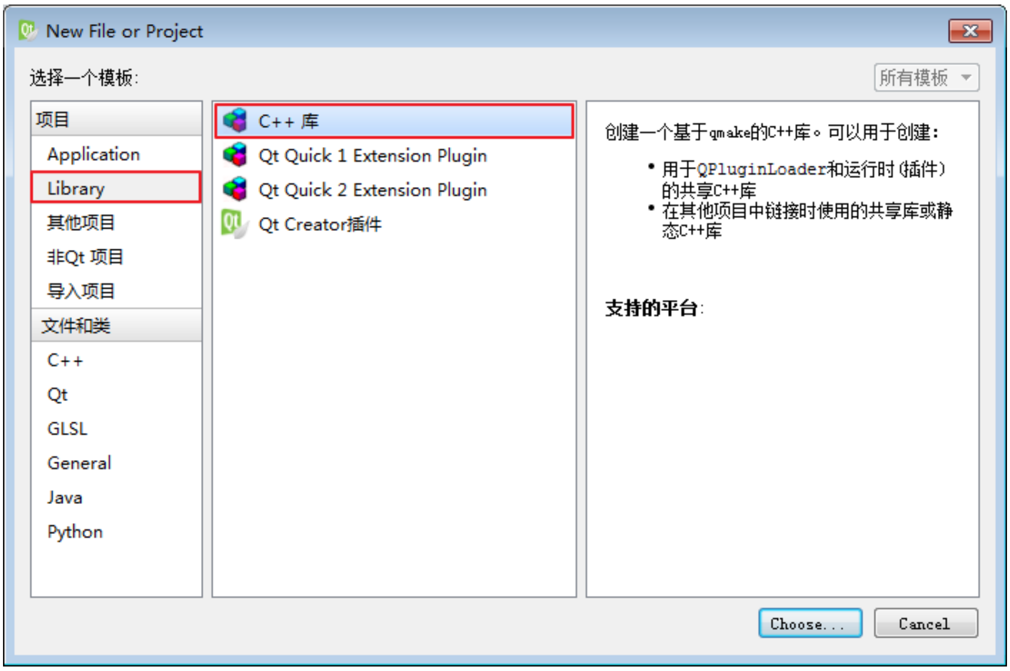
- Click the menu "file → new file or project...", the newly built dialog box would pop out, select the "C++ library" in the project type "Library", as shown by the figure 7.4.

- Select the "shared library" in the type combo box in the follow-up project introduction and location dialog box.

- Select the SylixOS Kit in the "Kit Selection" dialog box (such as "arm-sylixos-qt-

4.8.7”and “armv7-sylixos-qt-4.8.7”).

The compilation method of SylixOS shared library project fits the Qt Widgets application program development. Details can be found in section 5.3. The deployment directory of SylixOS shared library project is “/usr/lib”, click the “run” button to deploy the SylixOS shared library to the “/usr/lib” directory in the SylixOS device.



创建一个基于qmake的C++库。可以用于创建:

- 用于QPluginLoader和运行时(插件)的共享C++库
- 在其他项目中链接时使用的共享库或静态C++库

支持的平台:

Figure 7.4 C++ library

7.5 Develop SylixOS Static Library Project

RealEvo-QtSylixOS supports the development of SylixOS static library project. The SylixOS static library project uses qmake to generate Makefile, so it is exempted from the compilation of Makefile. The SylixOS static library project could use any Qt module.

The process for building the new SylixOS static library project is as follows:

- Click the menu “file → new file or project...”, the new built dialog box would pop out, select the “C++ library” template in the project type “Library”, as shown by figure 7.4.
- Select the “static library” in the type compo box in the follow-up project introduction and location dialog box.
- Select the SylixOS Kit in the follow-up “Kit Selection” (such as “arm-sylixos-qt-4.8.7” and “armv7-sylixos-qt-4.8.7”).

The compilation method of SylixOS static library project fits the Qt Widgets application program. Details can be found in section 5.3. SylixOS static library project does not support the deployment function.

Chapter 8 FAQ

8.1 How to Use QtSerialPort Class Library

It shall revise the project file (such as the HelloWorld.pro file in the HelloWorld project), add the content shown by the program list 8.1.

Program list 8.1 QtSerialPort class library usage

```
sylixos {
LIBS            += -lQtSerialPort
INCLUDEPATH     += $(QTDIR)/../QtSerialPort
}
```

Note: In terms of the further usage of QtSerialPort class, please refer to the Qt official wiki: http://wiki.qt.io/QtSerialPort

8.2 How to Use VNC

If you need to use VNC, please input the order as shown by the program list 8.2 in the Shell of SylixOS, in order to set and store the “QWS_DISPLAY” environment variable.

Program list 8.2 use VNC

```
QWS_DISPLAY=VNC:sylixosfb:$DISPLAY
varsave
```

Then restart Qt application program, at last click “run VNC client...” button in the “device option page” shown by figure 4.1.

8.3 How to Rotate Screen

If you need to rotate the screen by 90 degree, please input the order as shown by the program list 8.3 in the Shell of SylixOS, in order to set and store “QWS_DISPLAY” environmental variable.

Program list 8.3 rotate screen

```
QWS_DISPLAY=Transformed:Rot90
varsave
```

Then restart Qt application program.

If you need to rotate the screen by 90 degrees and use VNC in parallel, please input the order as shown by the program list 8.4 in the Shell of SylixOS and store the “QWS_DISPLAY” Environmental variable.

Program list 8.4 rotate screen and use VNC

```
QWS_DISPLAY=Transformed:Rot90:VNC:sylixosfb:$DISPLAY
varsave
```

Then restart Qt application program.

8.4 In Cases of Deployment Failure

When it comes across the deployment failure when using the Qt for deployment, it shall go through the following steps for confirmation:

1. Check whether the SylixOS device is set up correctly in figure 3.2 (the corresponding compilation tools chain is correct);
2. Click the “Test” button in the figure 3.6 to check whether the network is available (as shown by figure 3.5);
3. Check whether it has successfully set up the SylixOS BASE path of the device in the figure 3.8;
4. Check whether it selects the “Debug version” option correctly in figure 4.2 (it shall make right choice according to the actual compiled SylixOS BASE);
5. Check whether it selects “do not support symbol link” option in the figure 4.2 (in the SylixOS system, TpsFs file system, and Yaffs file system support symbol link, Fat32 file system does not support symbol link).