**跨平台技术方案**

**众智科技西安分公司**

**魏旭博**

**宋准**

**2021/06**

目录

[1. 背景 1](#_Toc75966410)

[2. 技术需求 1](#_Toc75966411)

[3. 具体技术要求 1](#_Toc75966412)

[4. 可选方案 1](#_Toc75966413)

[5. Linux平台文件类型说明 1](#_Toc75966414)

[6. 方案1--使用QtCreator+vs2019+qtVsTools 2](#_Toc75966415)

[6.1 window平台配置 2](#_Toc75966416)

[6.1.1 安装vs2019（16.10.2版本） 2](#_Toc75966417)

[6.1.2 安装Qt5.12.10版本 3](#_Toc75966418)

[6.1.3 安装vsQtTools 3](#_Toc75966419)

[6.2 Linux（ubuntu）平台配置 5](#_Toc75966420)

[6.2.1安装ubuntu 18.04 6](#_Toc75966421)

[6.2.2安装相关程序包 6](#_Toc75966422)

[6.2.3安装QT 5.12.10 6](#_Toc75966423)

[6.3 使用vsTools创建DLL项目 8](#_Toc75966424)

[6.3.1创建项目 8](#_Toc75966425)

[6.3.2配置Debug 10](#_Toc75966426)

[6.3.3配置Release（参考 Debug） 12](#_Toc75966427)

[6.3.4配置Debug-Linux 12](#_Toc75966428)

[6.3.5配置Release-Linux（参考Debug-Linux） 13](#_Toc75966429)

[6.4 使用vsTools创建exe项目 13](#_Toc75966430)

[6.4.1创建项目 13](#_Toc75966431)

[6.4.2配置Debug 15](#_Toc75966432)

[6.4.3配置Release（参考Debug） 16](#_Toc75966433)

[6.4.4配置Debug-Linux 16](#_Toc75966434)

[6.4.5配置Release-Linux （参考Debug-Linux） 18](#_Toc75966435)

[6.5 项目编译 18](#_Toc75966436)

[6.5.1编译报错 18](#_Toc75966437)

[6.6 项目调试 18](#_Toc75966438)

[6.6.1 window平台调试 18](#_Toc75966439)

[6.6.2 linux平台调试 19](#_Toc75966440)

[6.7 打包安装 19](#_Toc75966441)

[7.方案2--使用QtCreator+vs2019+CMake 19](#_Toc75966442)

# 背景

随着国家对国产化操作系统的支持和推广，原有公司的技术平台只能运行在window系统中，国产化操作系统大多不会另起炉灶，多基于unix/linux系统内核进行开发。为了公司产品长久的发展，需要指定一套跨平台的开发方案，将现有产品进行升级，可以支持不同的操作系统。主要目标为window，linux操作系统。

# 技术需求

目前公司内部没有开展跨平台项目的经验，内部员工比较熟悉的开发环境仍然局限在window上的visual studio开发工具。激进的切换开发工具链，对整个团队来讲并非容易，需要逐步过渡，或者可以找到更加兼容的方式来实现，让团队快速上手开发，而不必过多关注跨平台本身。

# 具体技术要求

- 可以和visual studio开发环境进行结合

- 开发人员可以不关注linux开发配置

- 便于构建linux和window平台下的产品

- 便于调试linux和window平台下的产品

- 使用qt作为跨平台界面

# 可选方案

1. 使用QtCreator+vs2019+qtVsTools作为开发工具链
2. 使用QtCreator+vs2019+CMake作为开发工具链
3. 使用QtCreator

# Linux平台文件类型说明

|  |  |  |
| --- | --- | --- |
| 系统  说明 | Linux | Windows |
| 文件后缀 | .so | .dll |
| .a | .lib |
| .o | .obj |
| .out | .exe |

在Linux系统中路径是 “/”，如/home/zzsoft/projects

在Windows 路径是“\”,如D:\dev\qt\QtCp

# 方案1--使用QtCreator+vs2019+qtVsTools

## window平台配置

Window平台下开发方案：

使用vs2019IDE+vs2015编译器（或vs2017编译器）+ Qt.5.12.10进行开发

### 安装vs2019（16.10.2版本）

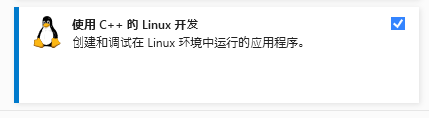
#### 首次安装vs2019

启动安装界面后-》启动负荷中必选的模块包括：

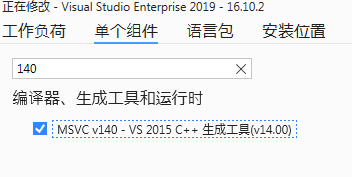
使用c++的桌面开发

使用C++的Linux开发





对于没有安装过vs2015，需要安装vs2015生成工具组件



#### 已经安装vs2019

A. 内网开发人员，大多数已经有在使用vs2019的早期版本，要升级vs2019到16.10.2，可以从ftp下载离线安装包进行升级安装：

10.0.30.50/ftp/Custom/Soft/Visual Studio/vs2019/vs2019offline-16.10.2.rar

解压程序，运行cache目录下的vs\_Enterprise.exe进行安装。启动后会提示更新，点击更新进行升级，等待更新完成，提示重启即完成升级。

注释：可能会出现某些组件需要重新安装，本人安装，出现勾选了netcore组件，对于其他用开发中用不到的组件可以去掉

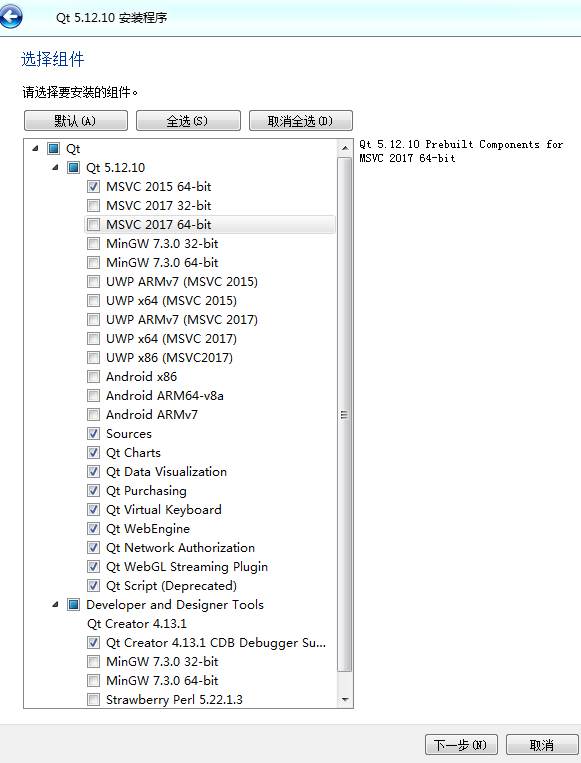
1. 更新完后，需要根据项目需要安装相应的C++、linux和vs2015支持组件

### 安装Qt5.12.10版本

首先，从ftp以下路径获取安装包：

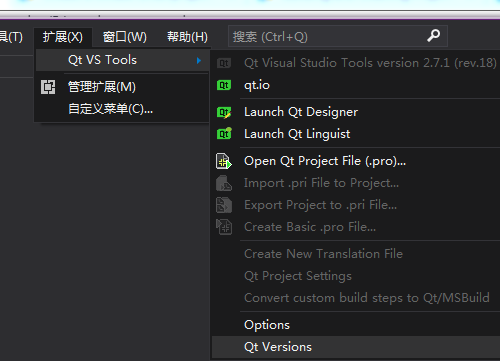
10.30.50/Custom/Soft/Qt/qt-opensource-windows-x86-5.12.10.exe

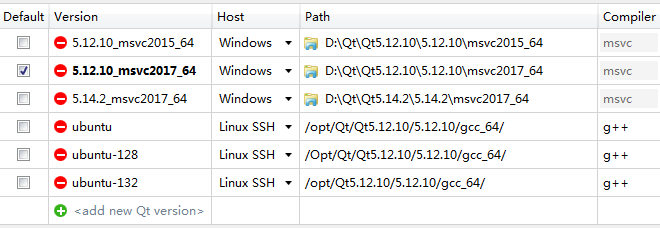
其次，外网安装需要断网安装，否则需要登录，安装运行后，勾选如下组件：



### 安装vsQtTools

从ftp服务器：10.0.30.50/ftp//Custom/Soft/Qt/qt-vsaddin-msvc2019-2.7.1.vsix获取插件进行安装，安装完后运行vs编辑器，找到扩展->Qt VS Tools->Qt Versions,配置Qt运行版本

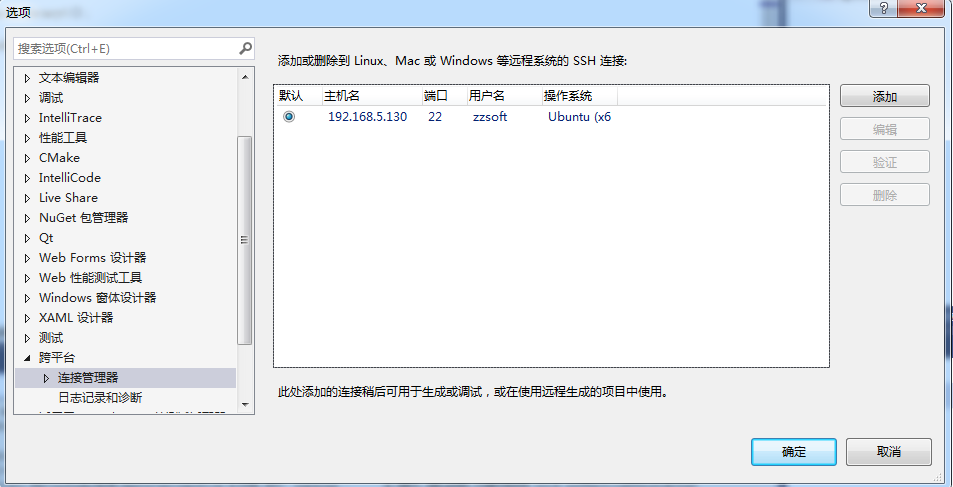




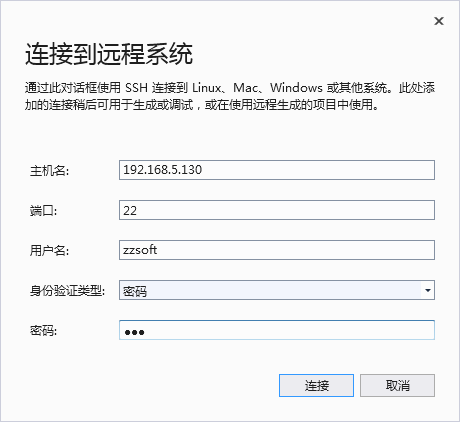
上图显示的windows表示本机安装的Qt编译版本，Linux SSH表示通过SSH工具远程连接进行交叉编译和调试的主机qt安装目录。

**配置SSH服务：**

工具——选项——跨平台——连接服务器

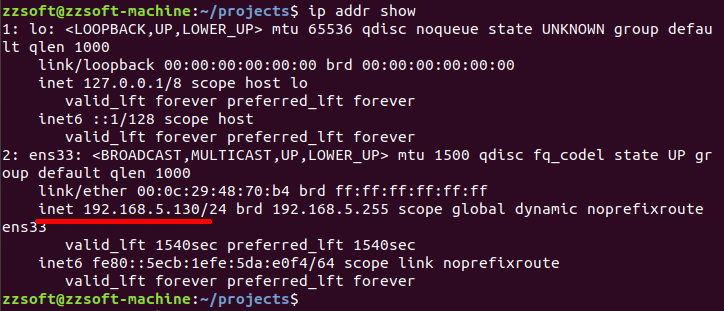


点击 添加，输入下列信息



第一个IP地址可在linux系统查看

输入命令 ip addr show



端口默认22不用改

输入用户名和密码进行连接

## Linux（ubuntu）平台配置

Linux平台使用ubuntu + QT + SSH

### 6.2.1安装ubuntu 18.04

### 6.2.2安装相关程序包

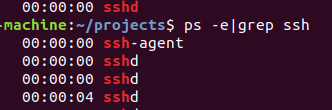
**（1）安装SSH**

sudo apt-get install openssh-server

sudo apt-get install openssh-client

启动SSH服务： sudo /etc/init.d/ssh start

检查是否开启： ps -e|grep ssh



ssh-agent表示ssh-client启动，sshd表示ssh-server启动

重启SSH服务： service ssh restart

开机自启动SSH： sudo systemctl enable ssh

**（2）安装G++**

sudo apt-get install g++

失败的话 sudo apt-get update 再试

**（3）安装qmake**

sudo apt-get install make

### 6.2.3安装QT 5.12.10

从ftp以下路径获取安装包：

10.0.30.50/Custom/Soft/Qt/qt-opensource-linux-x64-5.12.10 .run

放在任意目录并打开控制台

配置权限

chmod +x qt-opensource-linux-x64-5.12.10 .run

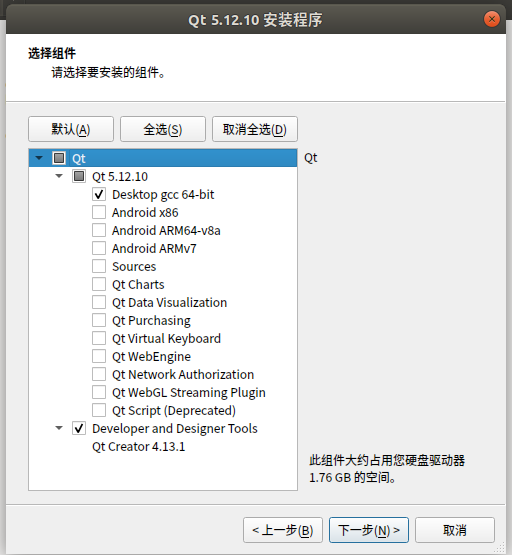
运行安装包（断网状态）

./qt-opensource-linux-x64-5.12.10 .run

安装包统一装在opt目录



勾选如下组件：



卸载：在qt目录运行sudo ./MaintenanceTool

若QT 不能运行： Sudo apt-get install libxcb\*

若缺少GL： Sudo apt-get install libgl1-mesa-dev

设置QT环境变量

任意目录运行： gedit ~/.bashrc

在最末尾加上：

QTDIR=/opt/Qt5.12.10/5.12.10/gcc\_64

PATH=$QTDIR/bin:$PATH

LD\_LIBRARY\_PATH=$QTDIR/lib:$LD\_LIBRARY\_PATH

export QTDIR PATH LD\_LIBRARY\_PATH

重启或 source ~/.bashrc生效

也可使用 $PATH 查看

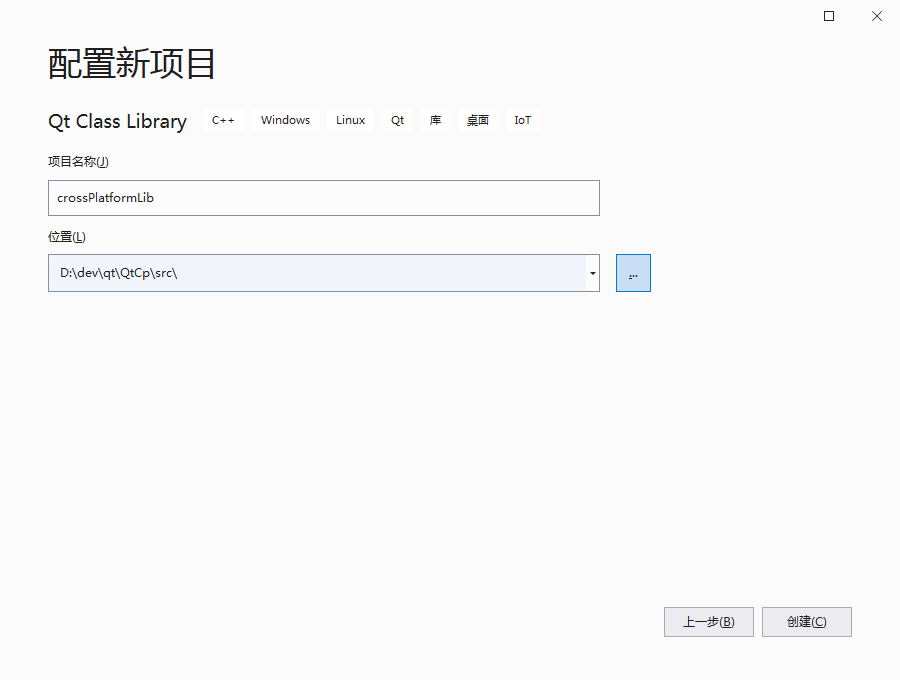
## 使用vsTools创建DLL项目

### 6.3.1创建项目

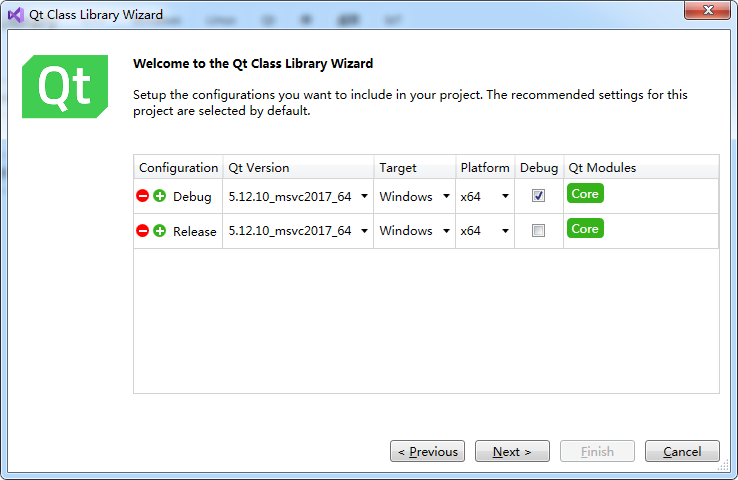
文件——新建——选择 Qt Class Library



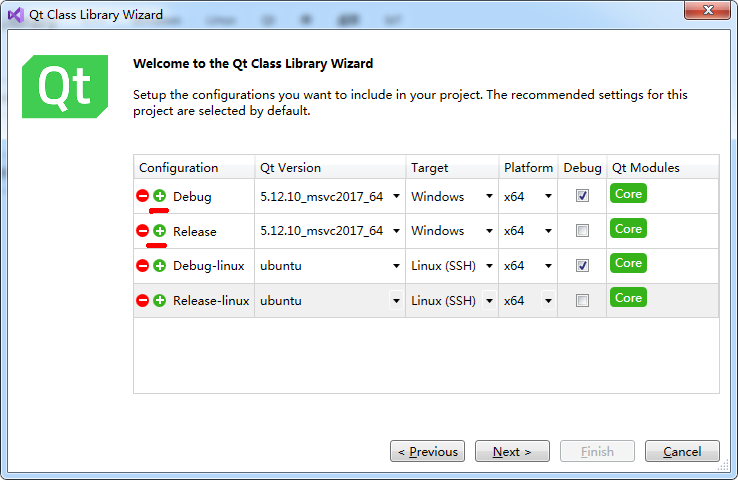
修改项目名称，放在src目录



创建——Next

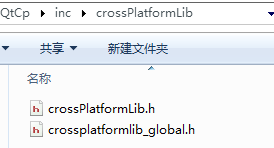


点击两个+号，并依照图所示 配置linux



Next——Finish 新建完成

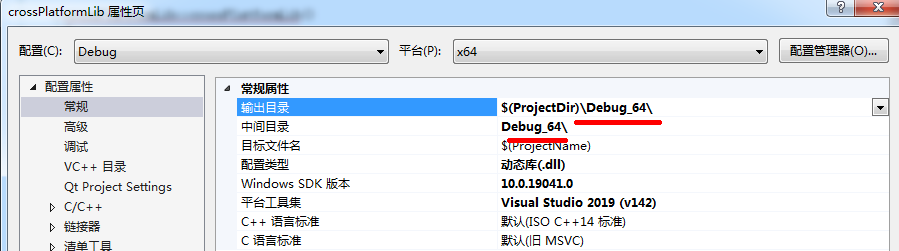
在工程inc目录下新建项目文件夹，调整.h文件位置。



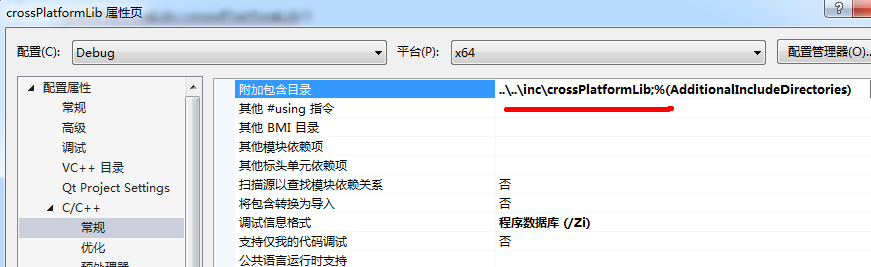
### 6.3.2配置Debug

项目——属性：

$(ProjectDir)\Debug\_64\

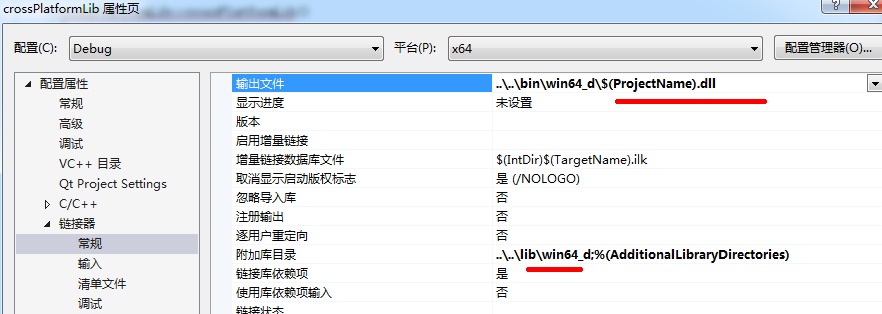


..\..\inc\crossPlatformLib

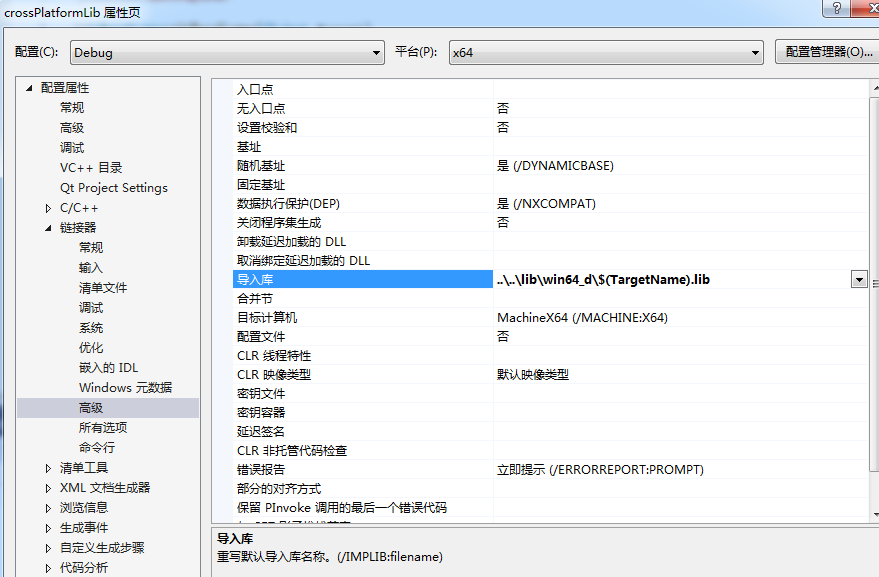


..\..\bin\win64\_d\$(ProjectName).dll

..\..\lib\win64\_d



..\..\lib\win64\_d\$(TargetName).lib

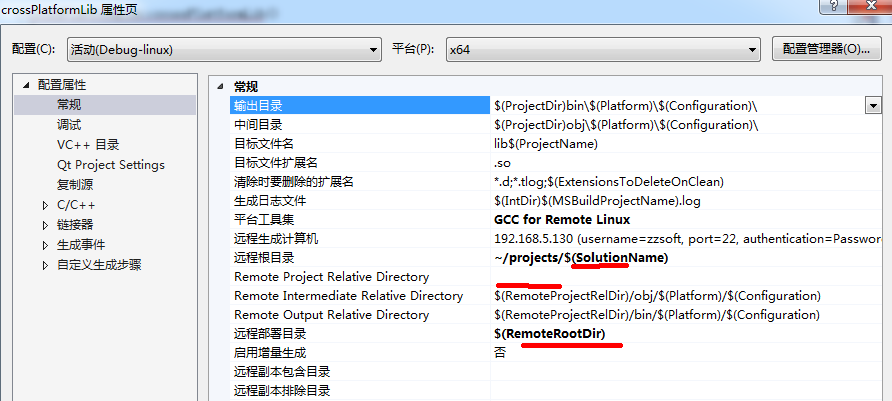


### 6.3.3配置Release（参考 Debug）

### 6.3.4配置Debug-Linux

项目——属性：

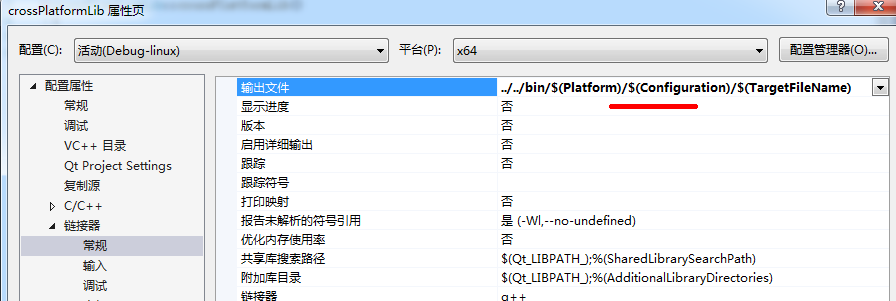
~/projects/$(SolutionName)



../../inc/$(ProjectName)



../../bin/$(Platform)/$(Configuration)/$(TargetFileName)



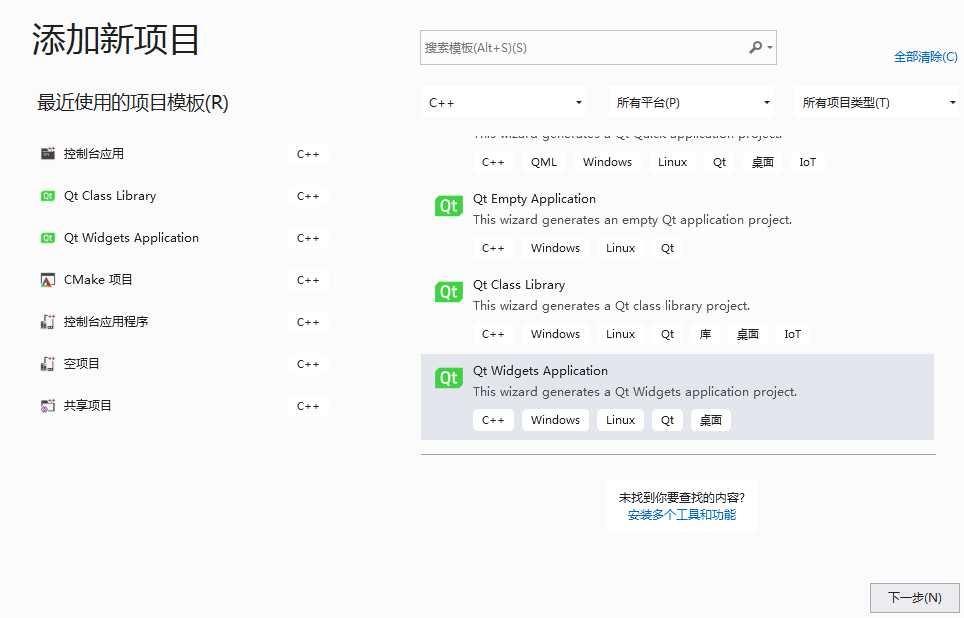
项目——生成，可在Linux系统的 /home/zzsoft/projects/QtCp/bin/x64/Debug-linux目录找到so文件

### 6.3.5配置Release-Linux（参考Debug-Linux）

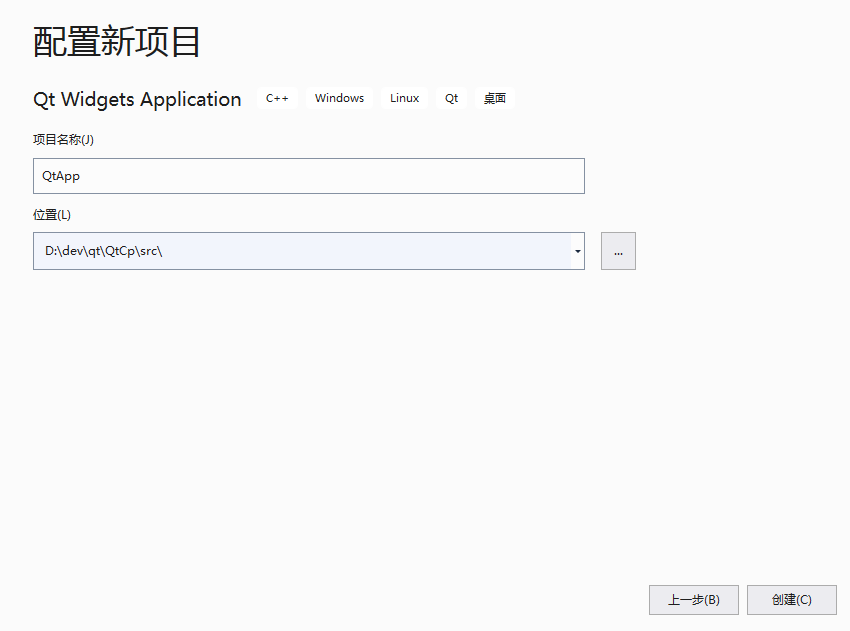
## 使用vsTools创建exe项目

### 6.4.1创建项目

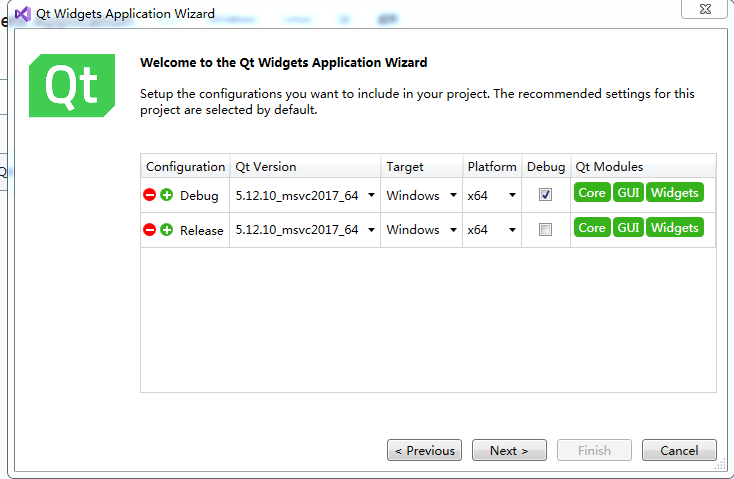
文件——新建——选择QT Widgets Application



修改项目名称，放到src目录

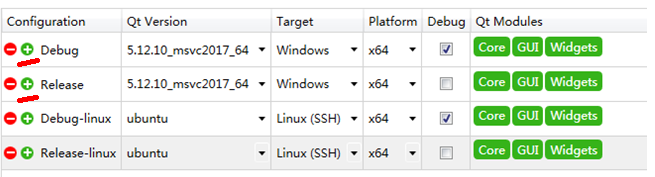


创建——Next



这里需要配置linux远程目录 （前提是配好vsTools ，分别指定Windows和linux系统的Qt库目录）

点击两个+号，并依照图所示 配置linux

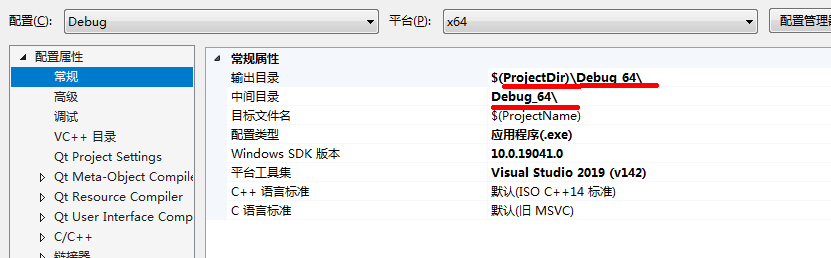


Next——Finish 新建完成

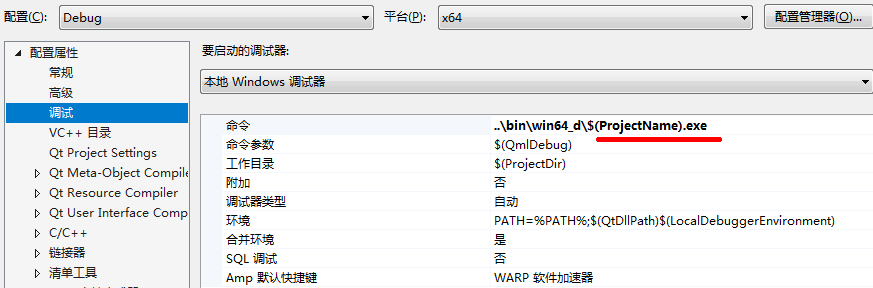
### 6.4.2配置Debug

项目——属性：

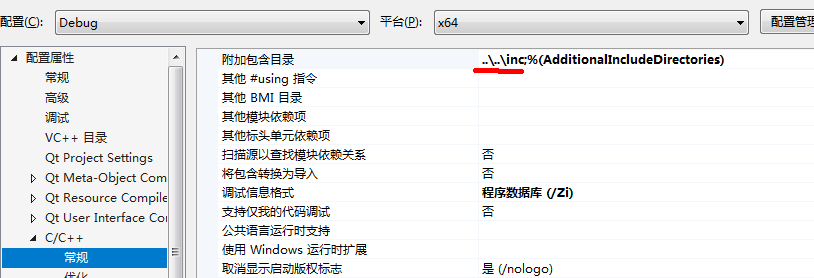
$(ProjectDir)\Debug\_64\



..\bin\win64\_d\$(ProjectName).exe

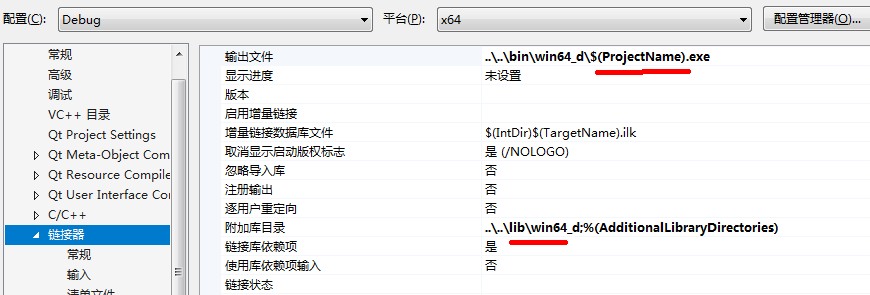


..\..\inc

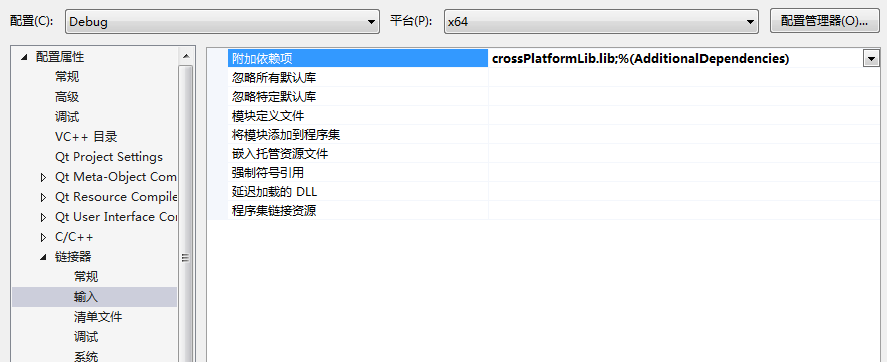


..\..\bin\win64\_d\$(ProjectName).exe

..\..\lib\win64\_d



crossPlatformLib.lib

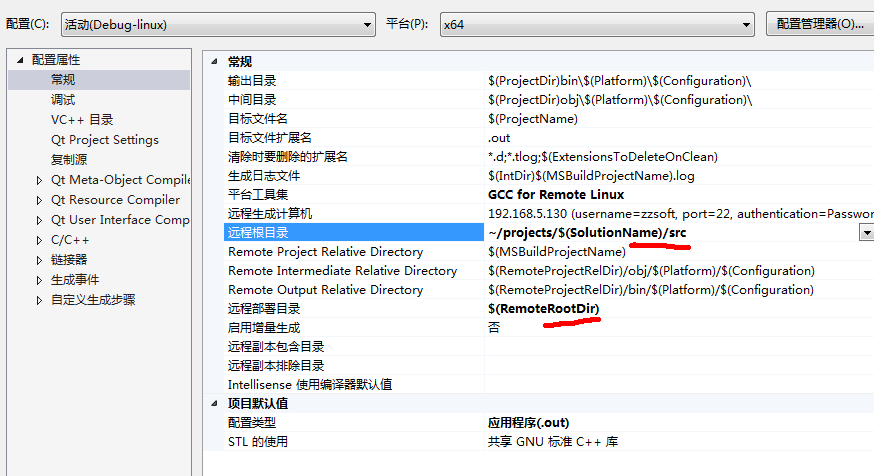


### 6.4.3配置Release（参考Debug）

### 6.4.4配置Debug-Linux

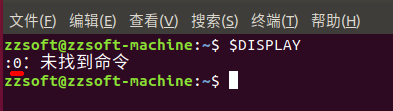
项目——属性：

~/projects/$(SolutionName)/src



export DISPLAY=:0.0

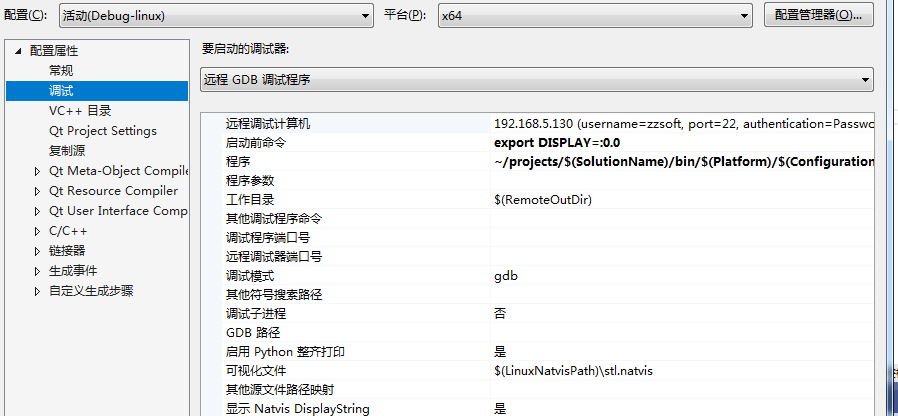
注：在Linux系统任意终端输入 $DISPLAY查看端口



是0就配置export DISPLAY=:0.0

是1就配置export DISPLAY=:1.0

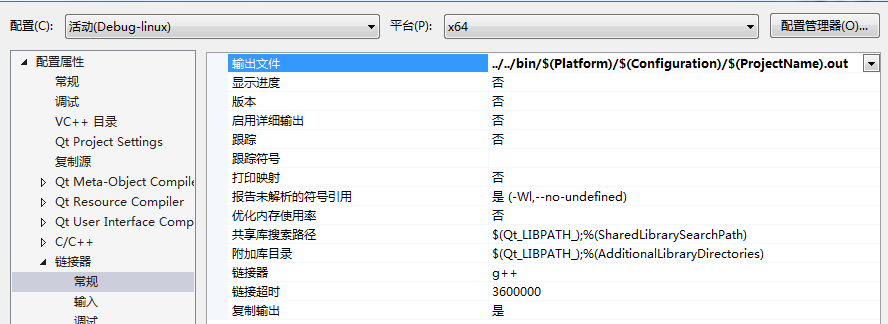
~/projects/$(SolutionName)/bin/$(Platform)/$(Configuration)/$(ProjectName).out



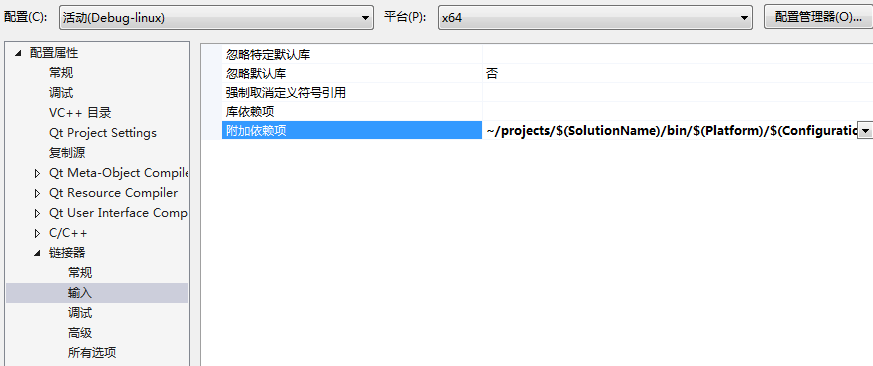
../../inc



../../bin/$(Platform)/$(Configuration)/$(ProjectName).out



~/projects/$(SolutionName)/bin/$(Platform)/$(Configuration)/libcrossPlatformLib.so



### 6.4.5配置Release-Linux （参考Debug-Linux）

## 项目编译

配置好项目，进行编译

### 6.5.1编译报错

若报错：\*\*vs props 有多个根元素

则删除所有生成项，重新生成就好了

## 项目调试

### window平台调试

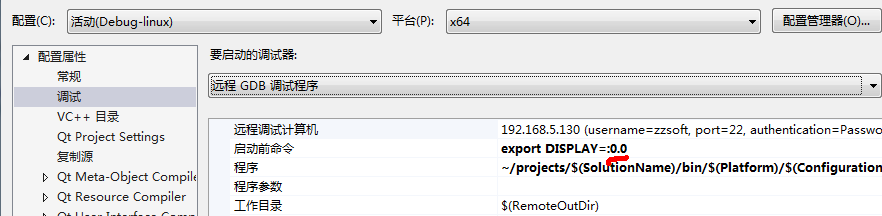
（1）检查远程连接

工具——选项——跨平台

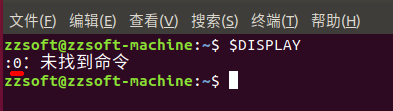
检查Linux远端IP地址及用户名密码

（2）检查Linux端口

export DISPLAY=:0.0



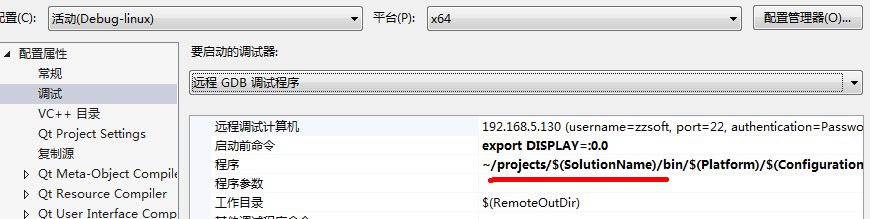
在Linux系统任意终端输入 $DISPLAY查看端口



是0就配置export DISPLAY=:0.0

是1就配置export DISPLAY=:1.0

（3）检查远程.out（相当于exe）路径



### linux平台调试

## 打包安装

https://blog.csdn.net/weixin\_40774605/article/details/105913281

# 7.方案2--使用QtCreator+vs2019+CMake