<https://blog.csdn.net/adriano119/article/details/5878169>

在QT中，有一个工具qmake可以生成一个makefile文件，它是由.pro文件生成而来的，.pro文件的写法如下：

1. 注释  
从“#”开始，到这一行结束。

2.模板变量告诉qmake为这个应用程序生成哪种makefile。下面是可供使用的选择：  
TEMPLATE = app  
A> app -建立一个应用程序的makefile。这是默认值，所以如果模板没有被指定，这个将被使用。  
B> lib - 建立一个库的makefile。  
C> vcapp - 建立一个应用程序的VisualStudio项目文件。  
D> vclib - 建立一个库的VisualStudio项目文件。  
E> subdirs -这是一个特殊的模板，它可以创建一个能够进入特定目录并且为一个项目文件生成makefile并且为它调用make的makefile。

#指定生成的应用程序放置的目录  
DESTDIR += ../bin

#指定生成的应用程序名  
TARGET = pksystem

#配置信息  
CONFIG用来告诉qmake关于应用程序的配置信息。

    CONFIG+= qt warn\_on release

在这里使用“+=”，是因为我们添加我们的配置选项到任何一个已经存在中。这样做比使用“=”那样替换已经指定的所有选项是更安全的。  
A> qt部分告诉qmake这个应用程序是使用Qt来连编的。这也就是说qmake在连接和为编译添加所需的包含路径的时候会考虑到Qt库的。  
B> warn\_on部分告诉qmake要把编译器设置为输出警告信息的。  
C> release部分告诉qmake应用程序必须被连编为一个发布的应用程序。在开发过程中，程序员也可以使用debug来替换release

#指定uic命令将.ui文件转化成ui\_\*.h文件的存放的目录  
UI\_DIR += forms

#指定rcc命令将.qrc文件转换成qrc\_\*.h文件的存放目录  
RCC\_DIR += ../tmp

#指定moc命令将含Q\_OBJECT的头文件转换成标准.h文件的存放目录  
MOC\_DIR += ../tmp

#指定目标文件(obj)的存放目录  
OBJECTS\_DIR += ../tmp

#程序编译时依赖的相关路径  
DEPENDPATH += . forms include qrc sources

#头文件包含路径  
INCLUDEPATH += .

#qmake时产生的信息,【

a读取变量a的字符串】,【a读取变量a的字符串】,【

(PATH)读取环境变量PATH】  
#message($$(PATH))

#源文件编码方式  
CODECFORSRC = GBK

#工程中包含的头文件  
HEADERS += include/painter.h  
#工程中包含的.ui设计文件  
FORMS += forms/painter.ui  
#工程中包含的源文件  
SOURCES += sources/main.cpp sources/painter.cpp  
#工程中包含的资源文件  
RESOURCES += qrc/painter.qrc

LIBS += -L folderPath  //引入的lib文件的路径  -L：引入路径

Release:LIBS += -L folderPath // release 版引入的lib文件路径

Debug:LIBS += -L folderPath // Debug 版引入的lib 文件路径

DEFINES += XX\_XX\_XXX  //定义编译选项，在.h文件中就可以使用 ：#ifdefine xx\_xx\_xxx

RC\_FILE = xxx.icns

7. 平台相关性处理  
我们在这里需要做的是根据qmake所运行的平台来使用相应的作用域来进行处理。为Windows平台添加的依赖平台的文件的简单的作用域看起来就像这样：

win32 {  
SOURCES += hello\_win.cpp  
}  
====================================================================================================================  
当你已经创建好你的项目文件，生成Makefile就很容易了，你所要做的就是先到你所生成的项目文件那里然后输入：

Makefile可以像这样由“.pro”文件生成：

    qmake -oMakefile hello.pro

对于VisualStudio的用户，qmake也可以生成“.dsp”文件，例如：

    qmake -tvcapp -o hello.dsp hello.pro

++++++++++++++++++++++++一个PRO文件实例++++++++++++++++++++++++++++++++++++++++

TEMPLATE = app     #模块配置  
LANGUAGE = C++   #C++语言

CONFIG += qt warn\_on debug release

#引入的lib文件,用于引入动态链接库  
LIBS += qaxcontainer.lib

#头文件包含路径  
INCLUDEPATH += ../../qtcompnent/qtchklisten/inc ../../qtcompnent/qtclearfile/inc ../../validator/inc/validerrcode ../../qtcompnent/qtdir/inc ../inc ../../utillib/inc/xmlapi ../../utillib/inc/util ../../xercesc ../../qtcompnent/qteditor/inc ../../qtcompnent/qtfunreview/inc ../../qtcompnent/qttable/inc ../../qtcompnent/qtversion/inc ../../qtcompnent/qtini/inc ../../icdtool/icdservices/inc ../../icdtool/dataset/inc ../../icdtool/doi/inc ../../icdtool/reportcontrol/inc ../../icdtool/GSEconctrol/inc ../../icdtool/inputs/inc ../../icdtool/SMVconctrol/inc ../../icdtool/logcontrol/inc ../../scdpreview/inc/scdpreviewtoollib ../../scdpreview/form ../../icdtool/sclcontrol/inc ../../icdtool/log/inc ../../icdtool/settingcontrol/inc ../../qtcompnent/qteditor/inc ../../qtcompnent/qttreeview/inc ../../qtcompnent/qttabwidget/inc ../../communication/inc ../../qtcompnent/qtabout/inc ../iedmanage/inc ../ldmanage/inc ../foriecrun/inc ../../qtcompnent/validset/inc

#工程中包含的头文件  
HEADERS += ../inc/exportstable.h /  
 ../inc/maintabwidget.h /  
 ../inc/outputtab.h /  
 ../inc/strutil.h /  
 ../inc/treeeditview.h /  
 ../inc/MainForm.h /  
 ../inc/recenfileini.h /  
 ../inc/ExportCIDFunction.h

#工程中包含的源文件  
SOURCES += ../src/main.cpp /  
 ../src/exportstable.cpp /  
 ../src/maintabwidget.cpp /  
 ../src/outputtab.cpp /  
 ../src/treeeditview.cpp /  
 ../src/MainForm.cpp /  
 ../src/recenfileini.cpp /  
 ../src/ExportCIDFunction.cpp

#工程中包含的.ui设计文件  
FORMS = ../form/scdmainform.ui /  
 ../form/exportiedform.ui /  
 ../form/Exportsedform.ui /  
 ../form/Importsedform.ui /  
 ../form/formiminputs.ui

#图像文件

IMAGES = images/substation.png /  
 images/communication.png /  
 images/autocom.png /  
 images/reportcfg.png /  
 images/comcfg.png /  
 images/filetrans.png /  
 images/review.png /  
 images/setting.png

#工程中包含的资源文件  
RESOURCES   = Scintilla.qrc

#CONFIG -= release  
CONFIG -= debug

RC\_FILE = scdtool.rc

BINLIB = ../../bin ../../xercesc/lib

UI\_HEADERS\_DIR = ../inc  # .ui文件转会为\*\*.h   存放的目录  
UI\_SOURCES\_DIR = ../src  # .ui文件转会为\*\*.cpp 存放的目录  
QMAKE\_LIBDIR = $${BINLIB}

release {  
TARGET = scdtool       #指定生成的应用程序名  
OBJECTS\_DIR = ../../obj/scdtool/release #指定目标文件(obj)的存放目录  
}  
debug {  
TARGET = scdtool\_d     #指定生成的应用程序名  
OBJECTS\_DIR = ../../obj/scdtool/debug #指定目标文件(obj)的存放目录  
}

MOC\_DIR = $${OBJECTS\_DIR}  
DESTDIR = ../../bin    #指定生成的应用程序放置的目录