软件部署（Software Deployment）

软件部署概述

软件部署是为将一个软件系统投入使用而进行的所有活动，包括硬件配置、软件的安装、环境变量设置等。通常软件部署需要考虑一下的功能模块：软件释放，软件安装及激活，软件停用及解除安装，软件内置更行和版本追踪等。简单说来，软件部署就是将程序员开发出来的软件部署到客户计算机上面，供其他人使用的一些列活动。需要说明的是，由于不同计算机的具体运行环境存在很大的差异，所以在软件的部署过程中有可能存在很多繁琐和复杂的细节。接下来的一部分，将用具体的实例展示软件在Windows平台部署的常规流程，使用C++和Python混合编程的开发环境。

QT软件部署示例

简介

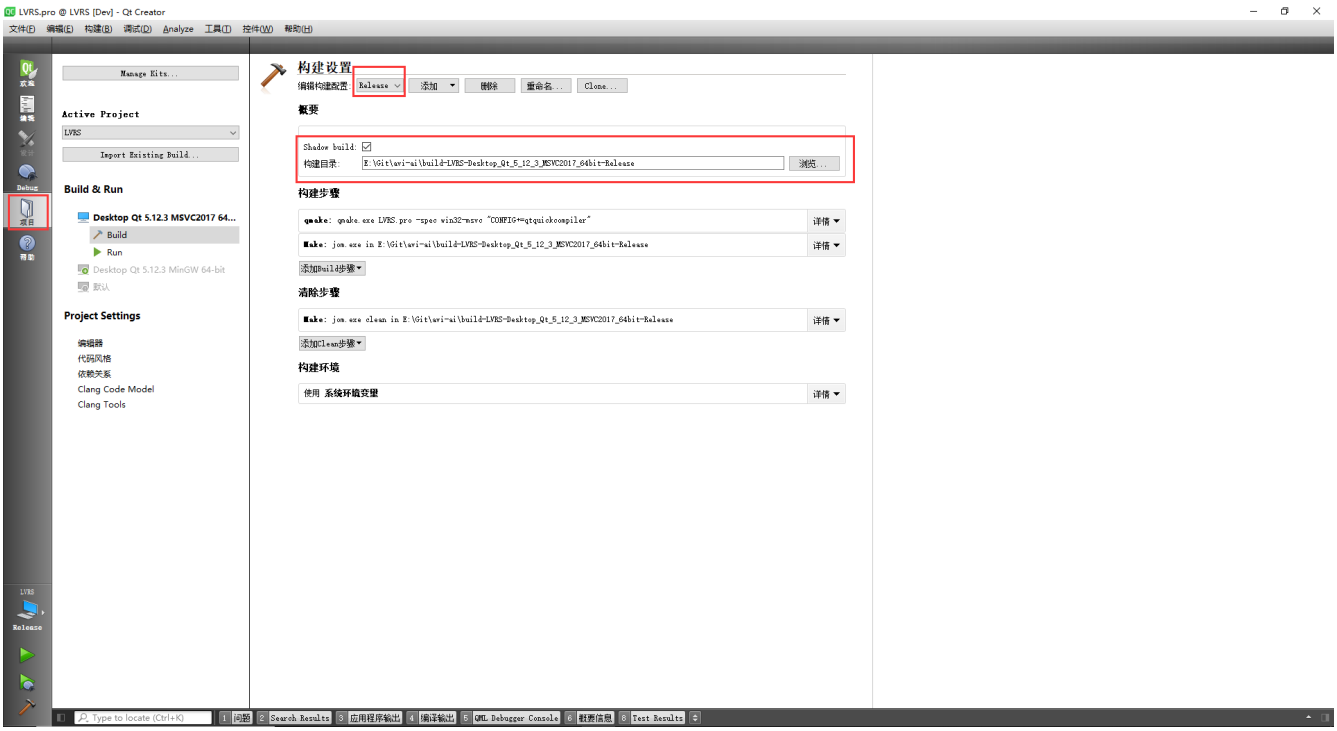
你是否遇到过这样的问题呢？ 当我们用QT写好了一个软件，要把你的程序分享出去的时候，不可能把编译的目录拷贝给别人去运行。编译好的程序往往应该是一个主程序，加一些资源文件，再加一些动态链接库。当然，更好的方式也可以将所有文件打包成一个安装文件（例如Windows平台.exe文件，Linux平台的.rpm文件）。

静态编译及动态链接库

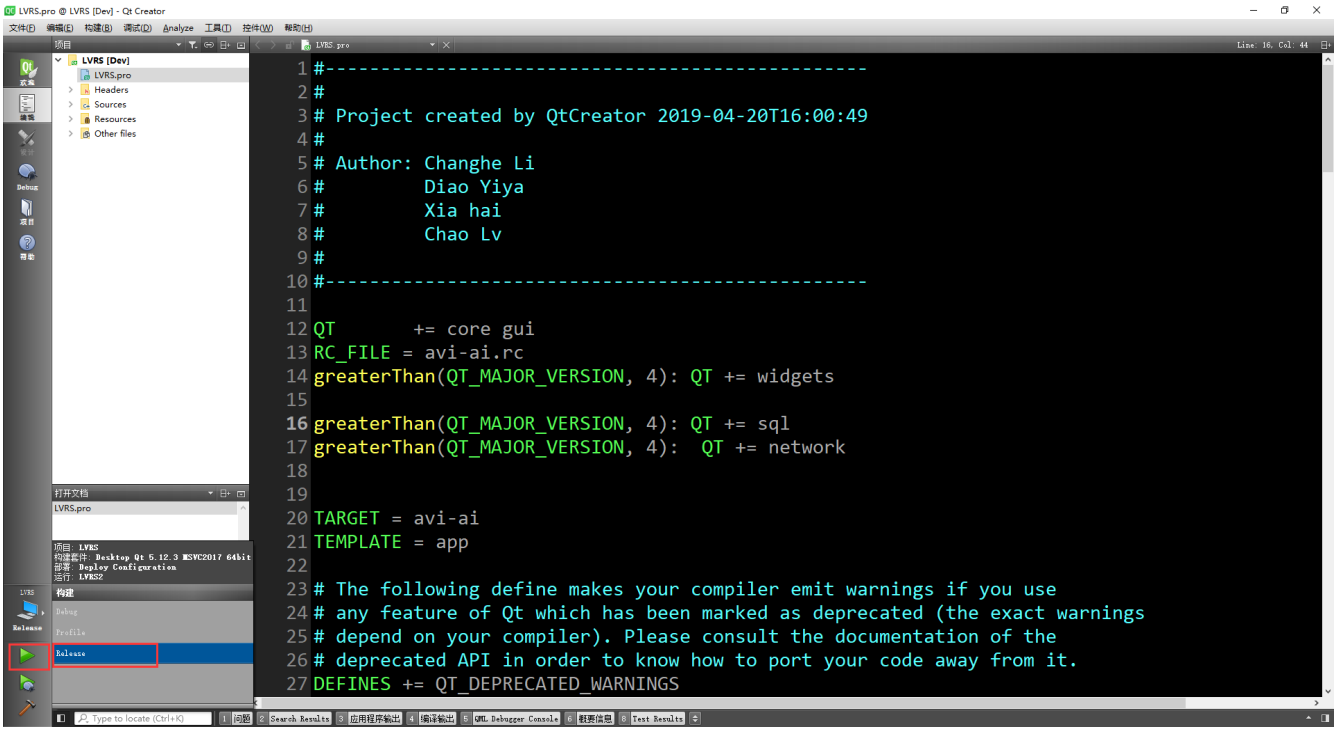
安装QT后，默认采用的是动态链接库的编译方式，如果需要发布程序，需要在可执行的文件中添加必须的动态链接库，然而有些动态链接库文件很大，这并不是我们想要的结果。最好的办法是提交一个静态链接的程序。但是安装的Qt是动态编译的，要生成静态的版本，就需要自己重新进行编译。而且，静态发布需要符合特定的授权问题，可见Qt LGPL授权文件。但是附带动态链接库则相当于直接使用了QT的DLL，直接包含于安装目录之下，使用者明确知道自己使用了LGPL授权版本。当然，使用动态链接库的方式部署软件也会让安装目录看起来比较杂乱，有时候也会出现动态链接库缺失的情况，需要耐心检测。

生成主程序

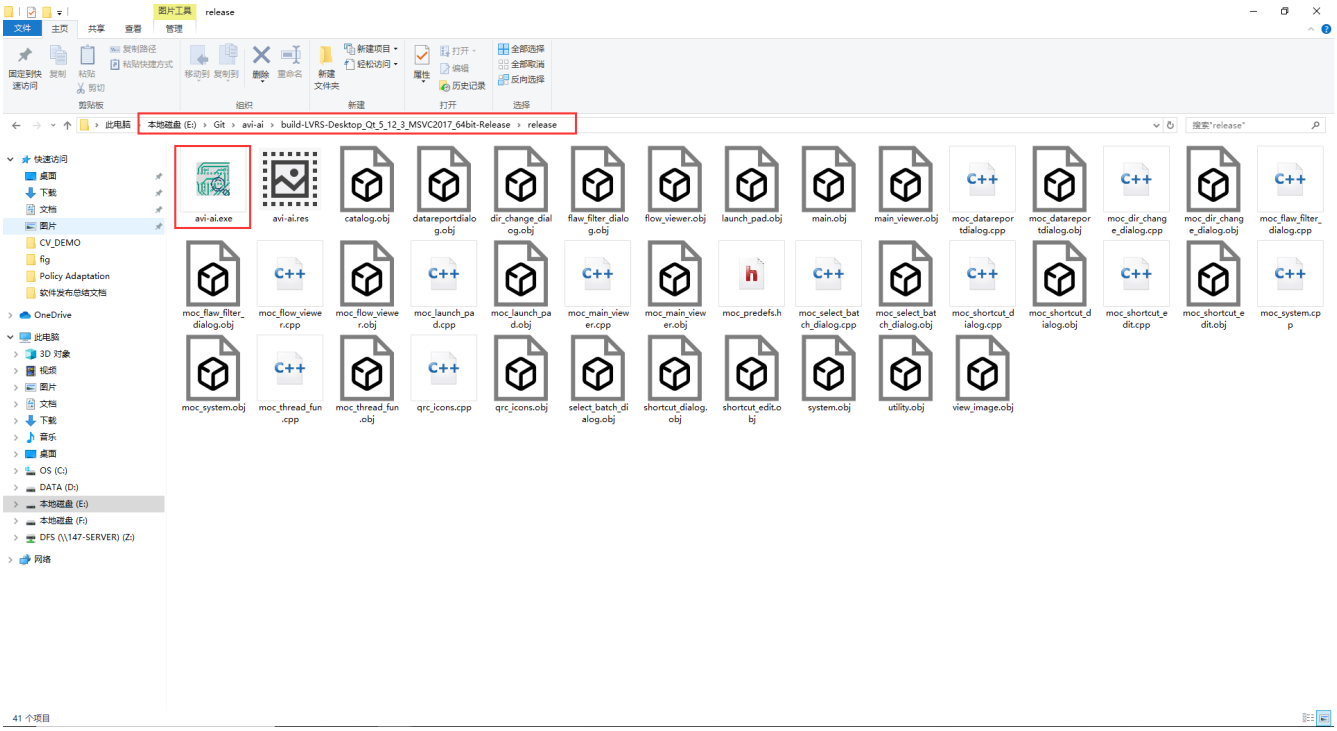
程序Debug方式测试完毕之后，将项目的构建套件改为Release版本，在相应的项目生成路径下找到生成的.exe文件。例如下图的构建方式，及项目构建套件的设置示例。



设置release构建套件



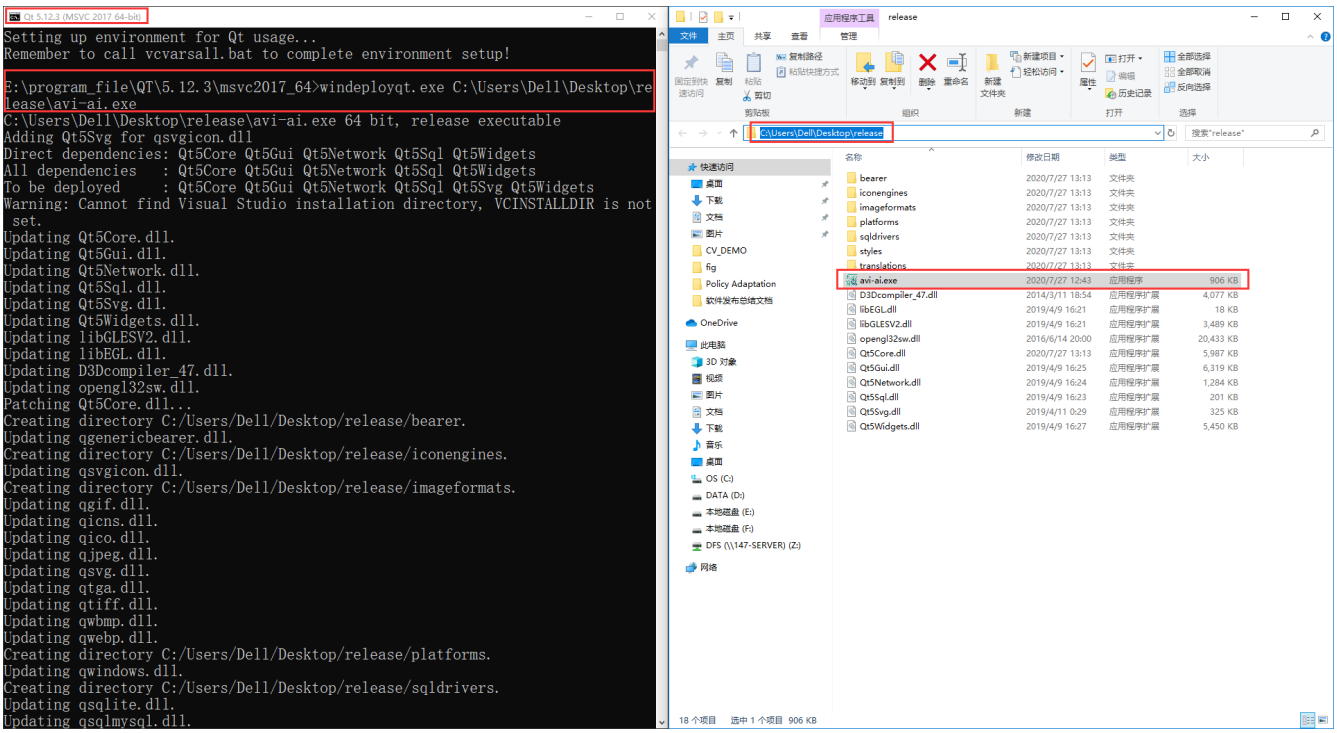
用release套件构建程序



主程序生成结果

抽取DLL文件

生成的Release版本的主程序成功后，只能够在QT的内置环境中成功运行，因为存在很多的依赖关系，因此需要将其依赖的DLL及资源文件等放到主程序能够找到的地方。这里，可以使用QT内置的开发工具windeployqt，寻找其依赖的DLL。需要注意的是若之前选择的是MSVC编译器这里就应该选择MSVC的命令行工具，若选择MinGW编译器这里就应该选择MinGW命令行工具部署。同时，32或者64位的环境应该也要对应上。可以先将生成的主程序置于一个空文件夹中，然后（这里MSVC命令行开发工具作为示例）使用`windeployqt.exe [主程序所在路径]`抽取DLL文件。



DLL抽取

打包为安装文件

抽取DLL文件完成后，主程序所在的文件夹中会包含所有的依赖，至此，该文件夹其实已经成为了一个绿色版本的软件，无需安装既可以使用。但是很多时候发布起来并不方便，因此可以将上述的安装文件打包成一个安装文件。可以借助NSIS等软件完成。这里以NSIS软件为例，可以自己新建一个脚本文件打包，可以在向导的帮助下建立模板。在向导帮助下建立模板的方式这里不做赘述，可以根据提示框手动选择相应的文件和方式。以下对NSIS脚本中常用的一些字段简要描述以下。

```
; 安装程序初始定义常量
!define PRODUCT_NAME "avi-ai"
!define PRODUCT_VERSION "1.0.69.0"
!define PRODUCT_PUBLISHER "China University of Geosciences (Wuhan)"
!define PRODUCT_WEB_SITE "http://www.mycompany.com"
!define PRODUCT_DIR_REGKEY "Software\Microsoft\Windows\CurrentVersion\App Paths\avi-
ai.exe"
!define PRODUCT_UNINST_KEY
"Software\Microsoft\Windows\CurrentVersion\Uninstall\${PRODUCT_NAME}"
!define PRODUCT_UNINST_ROOT_KEY "HKLM"


; 软件的属性信息

VIProductVersion "${PRODUCT_VERSION}" ;版本号，格式为 X.X.X.X (若使用则本条必须)
VIAddVersionKey  "ProductName" "avi-ai" ;产品名称
VIAddVersionKey  "Comments" " " ;备注
VIAddVersionKey  "CompanyName" "China University of Geosciences (Wuhan)" ;公司名称
VIAddVersionKey  "LegalTrademarks" " " ;合法商标
VIAddVersionKey  "LegalCopyright" "Copyright (c) 2019 China University of Geiosciences
(Wuhan)" ;合法版权
VIAddVersionKey  "FileDescription" " " ;文件描述(标准信息)
VIAddVersionKey  "FileVersion" "${PRODUCT_VERSION}" ;文件版本
VIAddVersionKey  "ProductVersion" "${PRODUCT_VERSION}" ;产品版本

SetCompressor lzma

; ------ MUI 现代界面定义 (1.67 版本以上兼容) ------
!include "MUI.nsh"

; MUI 预定义常量
!define MUI_ABORTWARNING
!define MUI_ICON "C:\Users\Dell\Desktop\1.0.69.0\rc\icon.ico"
!define MUI_UNICON "${NSISDIR}\Contrib\Graphics\Icons\modern-uninstall.ico"

; 欢迎页面
!insertmacro MUI_PAGE_WELCOME
; 安装目录选择页面
!insertmacro MUI_PAGE_DIRECTORY
; 安装过程页面
!insertmacro MUI_PAGE_INSTFILES
; 安装完成页面
!define MUI_FINISHPAGE_RUN "$INSTDIR\avi-ai.exe"
!define MUI_FINISHPAGE_SHOWREADME "$INSTDIR\description.html"
```

```
!insertmacro MUI_PAGE_FINISH

; 安装卸载过程页面
!insertmacro MUI_UNPAGE_INSTFILES

; 安装界面包含的语言设置
!insertmacro MUI_LANGUAGE "SimpChinese"

; 安装预释放文件
!insertmacro MUI_RESERVEFILE_INSTALLOPTIONS
; ------ MUI 现代界面定义结束 ------

Name "${PRODUCT_NAME} ${PRODUCT_VERSION}"
OutFile "avi-ai1.0.69.0.exe"
InstallDir "$PROGRAMFILES\avi-ai"
InstallDirRegKey HKLM "${PRODUCT_UNINST_KEY}" "UninstallString"
ShowInstDetails show
ShowUnInstDetails show


/******************************
 *  以下是安装程序的解压缩部分示例代码（包括文件夹和文件），注册表信息，快捷方式等  *
 ******************************/
Section "MainSection" SEC01
  SetOutPath "$INSTDIR"
  SetOverwrite ifnewer
  File "C:\Users\Dell\Desktop\1.0.69.0\avi-ai.exe"
  CreateDirectory "$SMPROGRAMS\avi-ai1.0.69.0"
  CreateShortCut "$SMPROGRAMS\avi-ai1.0.69.0\avi-ai1.0.69.0.lnk" "$INSTDIR\avi-ai.exe"


  SetOutPath "$INSTDIR"
  ; additional dll
  File "C:\Users\Dell\Desktop\1.0.69.0\ucrtbased.dll"
  File "C:\Users\Dell\Desktop\1.0.69.0\msvcp140.dll"
SectionEnd

Section -AdditionalIcons
  WriteIniStr "$INSTDIR\${PRODUCT_NAME}.url" "InternetShortcut" "URL"
"${PRODUCT_WEB_SITE}"
  CreateShortCut "$SMPROGRAMS\avi-ai1.0.69.0\Website.lnk" "$INSTDIR\${PRODUCT_NAME}.url"
  CreateShortCut "$SMPROGRAMS\avi-ai1.0.69.0\Uninstall.lnk" "$INSTDIR\uninst.exe"
SectionEnd

Section -Post
  WriteUninstaller "$INSTDIR\uninst.exe"
  WriteRegStr HKLM "${PRODUCT_DIR_REGKEY}" "" "$INSTDIR\avi-ai.exe"
  WriteRegStr ${PRODUCT_UNINST_ROOT_KEY} "${PRODUCT_UNINST_KEY}"
SectionEnd

/******************************
 *  以下是安装程序的卸载部分示例代码（包括文件夹和文件），注册表信息，快捷方式等  *
 ******************************/
```

```
Section Uninstall

  Delete "$INSTDIR\${PRODUCT_NAME}.url"
  Delete "$INSTDIR\uninst.exe"


  RMDir "$INSTDIR"

  DeleteRegKey ${PRODUCT_UNINST_ROOT_KEY} "${PRODUCT_UNINST_KEY}"
  DeleteRegKey HKLM "${PRODUCT_DIR_REGKEY}"
  SetAutoClose true
SectionEnd

#-- 根据 NSIS 脚本编辑规则，所有 Function 区段必须放置在 Section 区段之后编写，以避免安装程序出现未可预知的问题。--#

Function un.onInit
  MessageBox MB_ICONQUESTION|MB_YESNO|MB_DEFBUTTON2 "您确实要完全移除 $(^Name) ，及其所有的组件? " IDYES +2
  Abort
FunctionEnd

Function un.onUninstSuccess
  HideWindow
  MessageBox MB_ICONINFORMATION|MB_OK "$(^Name) 已成功地从您的计算机移除。"
FunctionEnd
```

添加资源文件

以上步骤对于简单的软件部署已基本够用。但是对于稍微复杂的软件来说，除了抽取出DLL文件之外，主程序若存在资源文件，也应该一并置于该目录下面。这里需要特别注意的是资源文件和主程序之间的相对路径。因为QT默认生成的主程序、资源文件的相对位置若在部署的时候发生改变，资源文件的打开就会出现错误。例如：若按照以上的QT构件套件设置，生成的主程序会放在`E:\Git\avi-ai\build-LVRS-Desktop_Qt_5_12_3_MSVC2017_64bit-Release`目录下，QT会将当前程序所在的路径默认为`E:\Git\avi-ai\build-LVRS-Desktop_Qt_5_12_3_MSVC2017_64bit-Release`，若资源文件在调试的时候放在`E:\Git\avi-ai`目录之下，若使用相对路径，则程序里面会使用`../资源名称`的方式获取路径。这样部署的时候，资源文件就不能够放在和主程序的同级目录，而应该放在上级目录。当然，为了解决这个问题也可以在生成主程序的时候根据自己安装目录的理想层级结构自由修改，进而和资源的路径匹配。

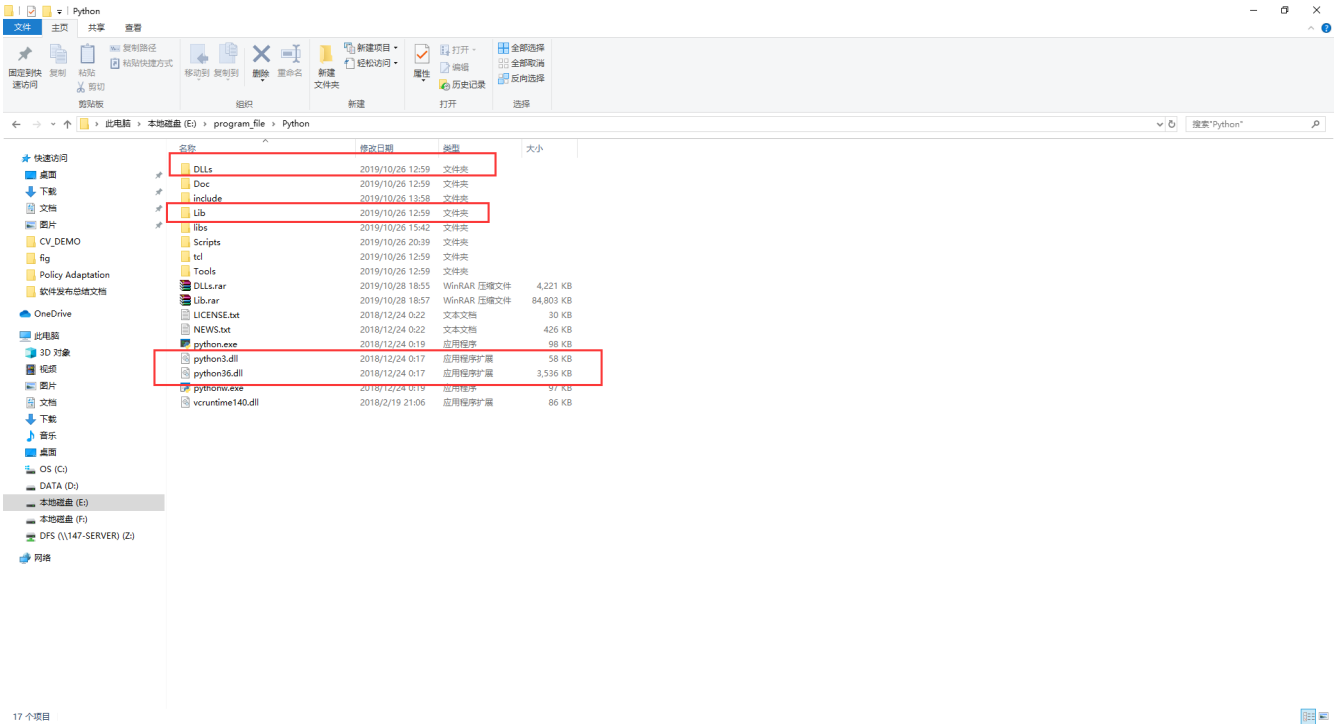
混合编程的软件部署

简述

不同程序开发语言都有自己的特色，若一个软件的开发过程中用到了多种语言，那么使用单一的语言开发环境抽取出来的依赖库难以保全。因此，这样的软件在客户的计算机上面运行面临更大的挑战。此处以C++和Python在QT环境下的混合编程软件部署为例，当使用`windeployqt.exe`抽取完了DLL数据之后，Python的解释环境并没有部署。因此，还需要将相应版本的Python环境添加到软件安装包中，这样才能支持软件中Python部分的调用。

抽取Python的解释环境

找到相应版本的Python安装路径，将如下的文件（夹）置于主程序的打包目录中。需要注意的是，目录中的文件较多，若直接放到主程序的目录中使用NSIS脚本打包，往往NSIS软件会卡死，因此可以先将文件夹压缩，而后在进行打包操作。



Python环境抽取