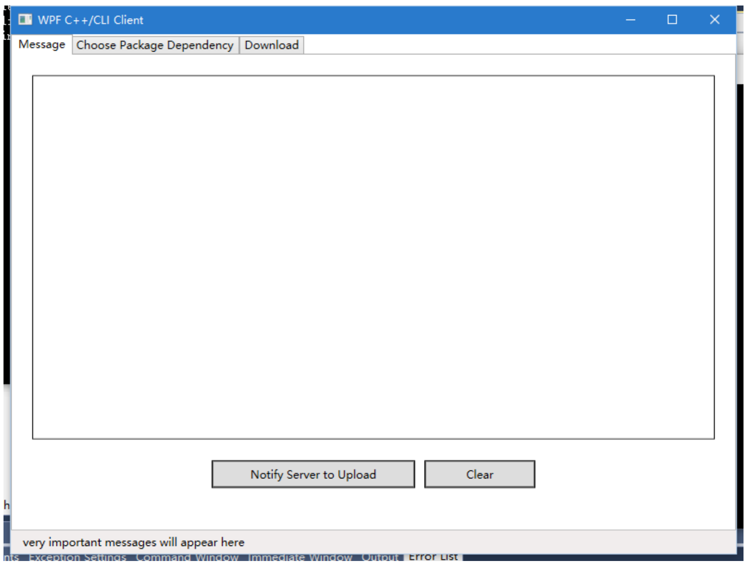
ReadMe of Dependency-Based Remote Code Repository

Jiayu Li

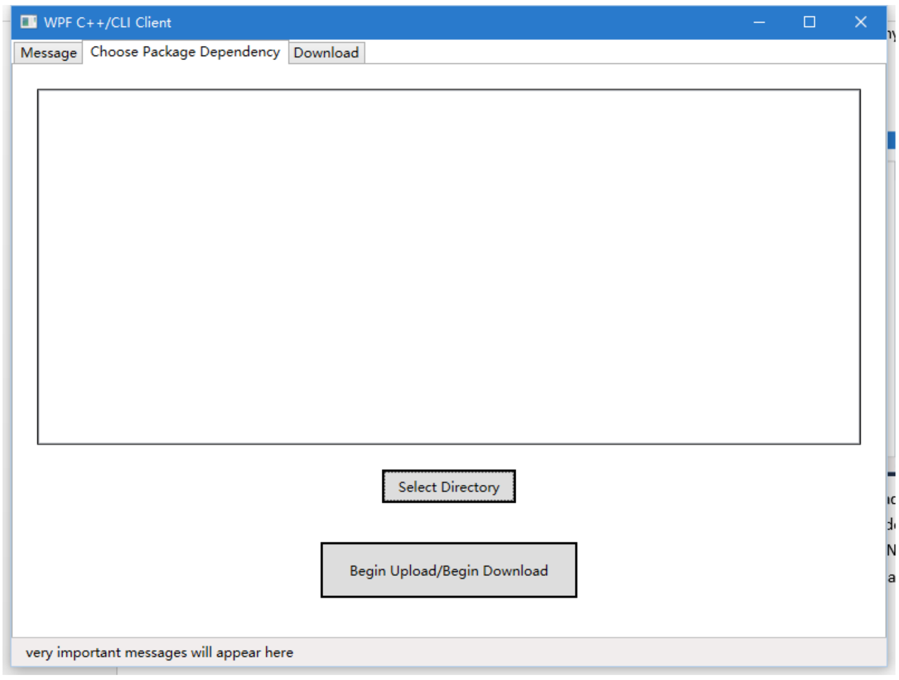
I have implement the project and match the requirements. I will explain how to use my project and how my project match all the requirements. And my project can process the types of files including .cpp, .h and .xml.

I design some steps to use my project. You can follow steps to use it. Otherwise, it may have some problems. Now, I start my project you can see a window:

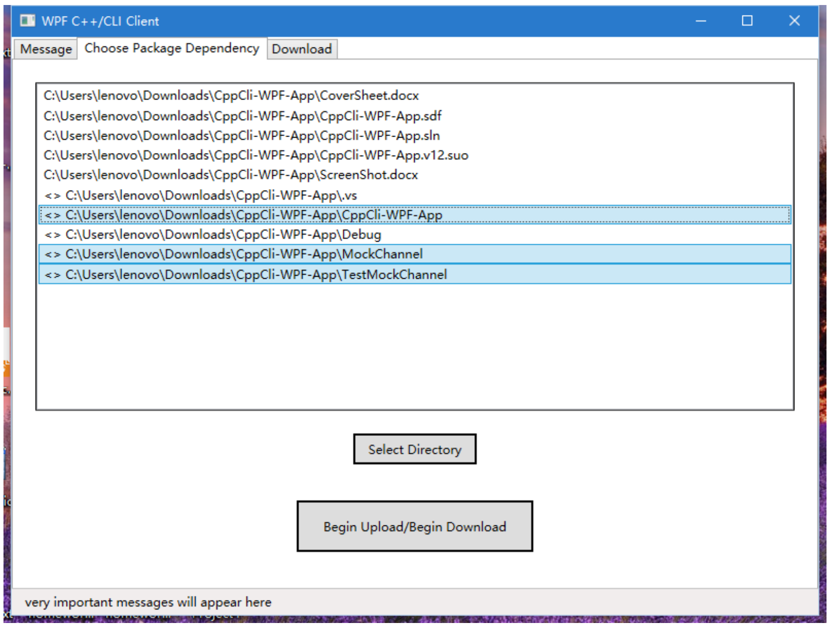


very important messages will appear here

(1) When you want to upload (or check in) file, you must press the button "Notify Server to Upload" firstly. After pressing the button, you can go click the head tab "Choose Package Dependency" to decide which packages have dependencies and you want to store them in server. Note that you should choose package instead of individual files like .cpp or .h because in this project, professor asks us to process the package or module dependencies. I set it just can process the dependency of package.

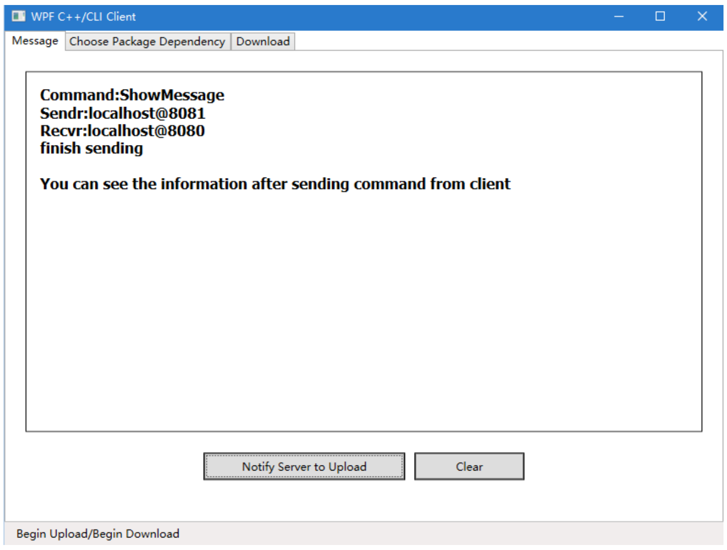

WPF C++/CLI Client
Message
Choose Package Dependency
Download
Select Directory
Begin Upload/Begin Download
very important messages will appear here

After you press the head tab and enter the window "Choose Package Dependency", you can press the button "Select Directory" to choose packages.


WPF C++/CLI Client
Message
Choose Package Dependency
Download
Select Directory
Begin Upload/Begin Download
very important messages will appear here

```
C:\Users\lenovo\Downloads\CppCli-WPF-App\CoverSheet.docx
C:\Users\lenovo\Downloads\CppCli-WPF-App\CppCli-WPF-App.sdf
C:\Users\lenovo\Downloads\CppCli-WPF-App\CppCli-WPF-App.sln
C:\Users\lenovo\Downloads\CppCli-WPF-App\CppCli-WPF-App.v12.suo
C:\Users\lenovo\Downloads\CppCli-WPF-App\ScreenShot.docx
<> C:\Users\lenovo\Downloads\CppCli-WPF-App\.vs
<> C:\Users\lenovo\Downloads\CppCli-WPF-App\CppCli-WPF-App
<> C:\Users\lenovo\Downloads\CppCli-WPF-App\Debug
<> C:\Users\lenovo\Downloads\CppCli-WPF-App\MockChannel
<> C:\Users\lenovo\Downloads\CppCli-WPF-App\TestMockChannel
```

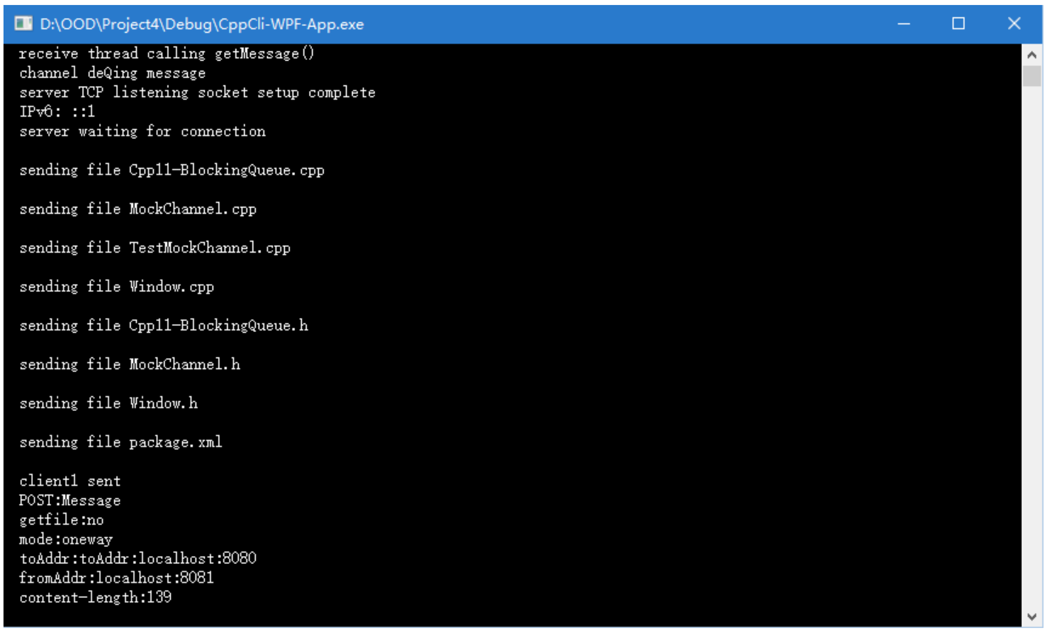
Now, you can see I choose the package CppCli-WPF-App, MockChannel and TestMockChannel. When I press the button "Begin Upload/Begin Download", it will send the files with dependencies to the server including a corresponding an xml file. Note: important steps: 1. Notify server to upload; 2. Go into "Choose Package Dependency" and choose the packages with dependencies. 3. Press button "Begin Upload/Begin Download" to upload the files to server. And you can see the result when you go back to the head tab "Message":



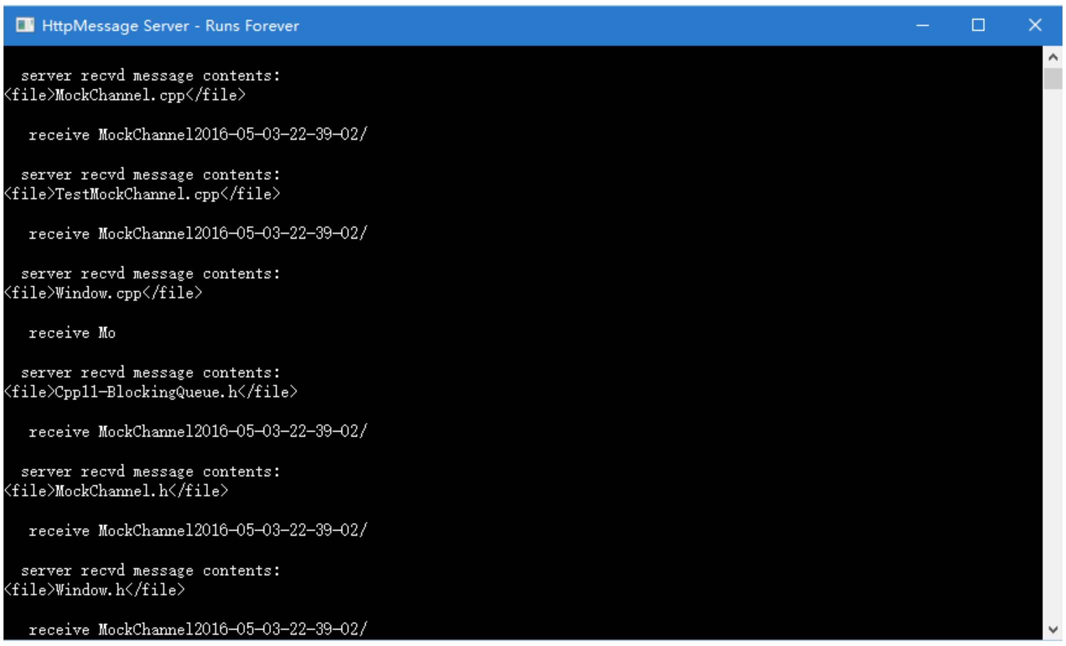
```
Command:ShowMessage
Sendr:localhost@8081
Recvr:localhost@8080
finish sending

You can see the information after sending command from client
```

Begin Upload/Begin Download

In the window, you can see client gives user a notification that it has sent packages to server according to you command. Also, you can see the result in console of client and server. Client sending files and sending POST message to server. And server it receives message from the client and the name of the files.

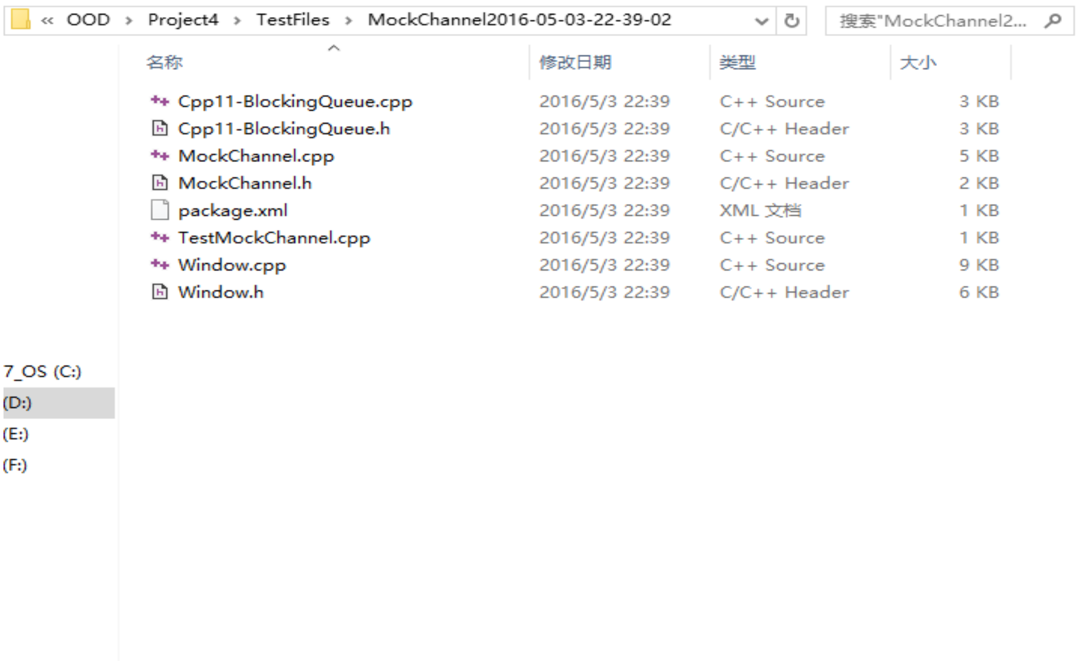


```
receive thread calling getMessage()
channel deQing message
server TCP listening socket setup complete
IPv6: ::1
server waiting for connection

sending file Cpp11-BlockingQueue.cpp

sending file MockChannel.cpp

sending file TestMockChannel.cpp

sending file Window.cpp

sending file Cpp11-BlockingQueue.h

sending file MockChannel.h

sending file Window.h

sending file package.xml

client1 sent
POST:Message
getfile:no
mode:oneway
toAddr:toAddr:localhost:8080
fromAddr:localhost:8081
content-length:139
```



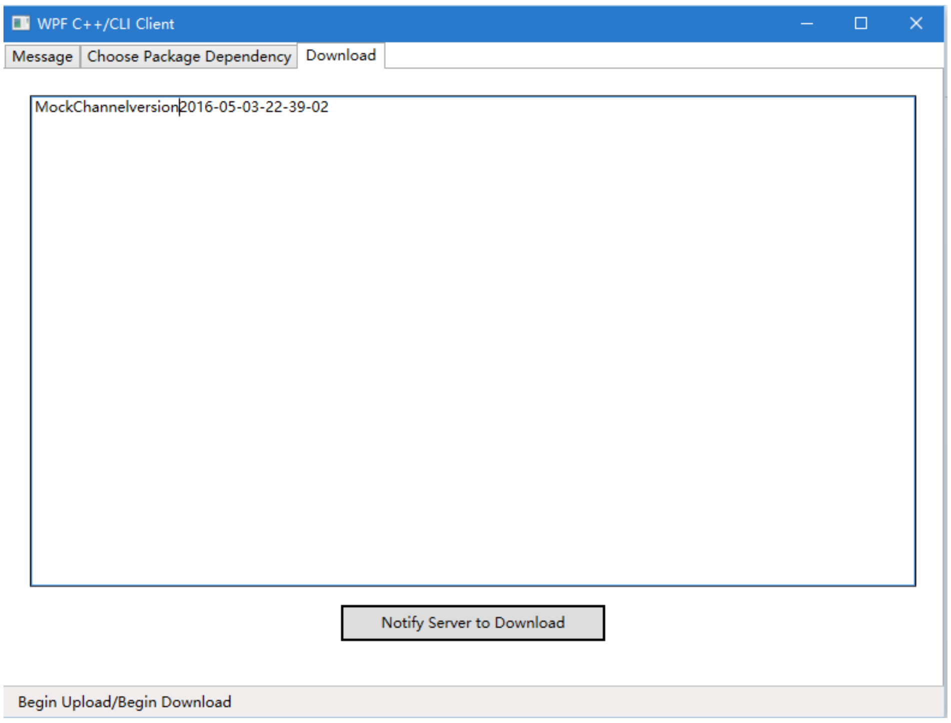
```
 server recvd message contents:
<file>MockChannel.cpp</file>

  receive MockChannel2016-05-03-22-39-02/

 server recvd message contents:
<file>TestMockChannel.cpp</file>

  receive MockChannel2016-05-03-22-39-02/

 server recvd message contents:
<file>Window.cpp</file>

  receive Mo

 server recvd message contents:
<file>Cpp11-BlockingQueue.h</file>

  receive MockChannel2016-05-03-22-39-02/

 server recvd message contents:
<file>MockChannel.h</file>

  receive MockChannel2016-05-03-22-39-02/

 server recvd message contents:
<file>Window.h</file>

  receive MockChannel2016-05-03-22-39-02/
```

Also in the path you can see the files has been received in the server. (I use TestFiles as server to store the files from clients). And you can check the package name which consists of check-in data and the first package name. (e.g. in the example, check-in data is 2016-05-03-22-39-02 and first package name is MockChannel)


« OOD › Project4 › TestFiles › MockChannel2016-05-03-22-39-02
搜索"MockChannel2...
7_OS (C:)
(D:)
(E:)
(F:)

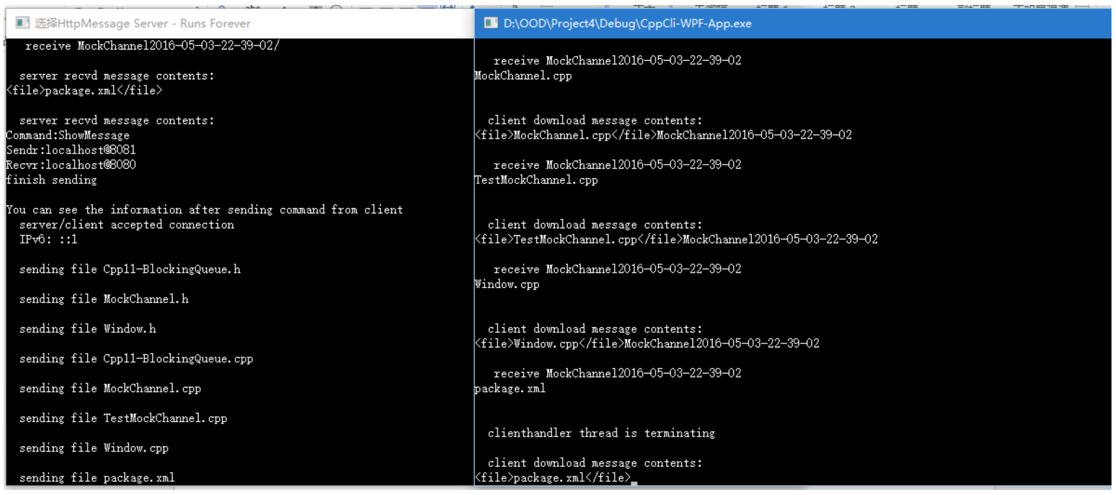
名称	修改日期	类型	大小
Cpp11-BlockingQueue.cpp	2016/5/3 22:39	C++ Source	3 KB
Cpp11-BlockingQueue.h	2016/5/3 22:39	C/C++ Header	3 KB
MockChannel.cpp	2016/5/3 22:39	C++ Source	5 KB
MockChannel.h	2016/5/3 22:39	C/C++ Header	2 KB
package.xml	2016/5/3 22:39	XML 文档	1 KB
TestMockChannel.cpp	2016/5/3 22:39	C++ Source	1 KB
Window.cpp	2016/5/3 22:39	C++ Source	9 KB
Window.h	2016/5/3 22:39	C/C++ Header	6 KB

(2) When we finish the upload packages (also calls check-in), we want to download some packages with dependencies based on the version and the package name. (extraction)


WPF C++/CLI Client
Message
Choose Package Dependency
Download
Notify Server to Download
Begin Upload/Begin Download

```
MockChannelversion2016-05-03-22-39-02
```

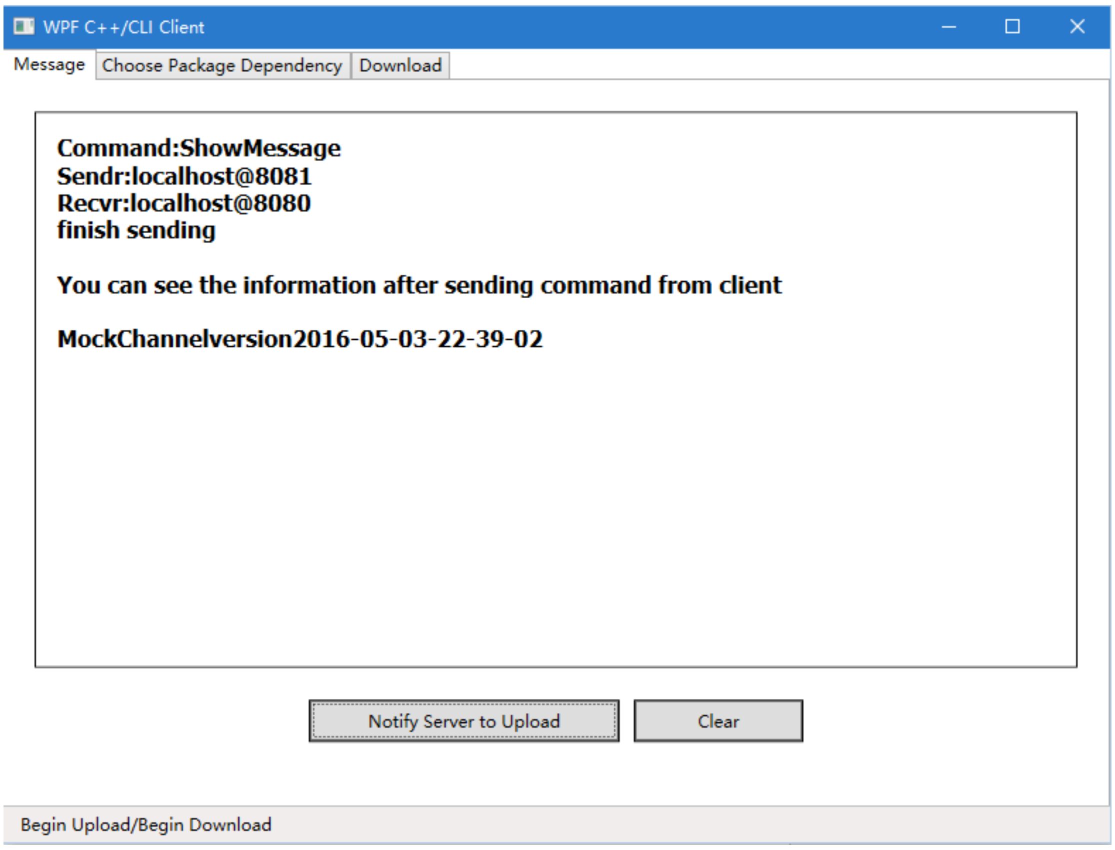
So, you can go into the window "Download" and input the package name and version you want to download. Note that the input should follow a format: package name + version + check-in data like I write above. When finishing input, you can press the button "Notify Server to Download" and then you jump back to the window "Choose Package Dependency" and press the button "Begin Upload/Begin Download". Now, you can see the result: (for example, I want to download the packages I uploaded before. The package name is MockChannel and the check-in data is 2016-05-03-22-39-02))



```
选择HttpMessage Server - Runs Forever
  receive MockChannel2016-05-03-22-39-02/

  server recvd message contents:
<file>package.xml</file>

  server recvd message contents:
Command:ShowMessage
Sendr:localhost@8081
Recvr:localhost@8080
finish sending

You can see the information after sending command from client
  server/client accepted connection
  IPv6: ::1

  sending file Cpp11-BlockingQueue.h

  sending file MockChannel.h

  sending file Window.h

  sending file Cpp11-BlockingQueue.cpp

  sending file MockChannel.cpp

  sending file TestMockChannel.cpp

  sending file Window.cpp

  sending file package.xml
```

```
D:\OOD\Project4\Debug\CppCli-WPF-App.exe

   receive MockChannel2016-05-03-22-39-02
MockChannel.cpp


  client download message contents:
<file>MockChannel.cpp</file>MockChannel2016-05-03-22-39-02

   receive MockChannel2016-05-03-22-39-02
TestMockChannel.cpp


  client download message contents:
<file>TestMockChannel.cpp</file>MockChannel2016-05-03-22-39-02

   receive MockChannel2016-05-03-22-39-02
Window.cpp


  client download message contents:
<file>Window.cpp</file>MockChannel2016-05-03-22-39-02

   receive MockChannel2016-05-03-22-39-02
package.xml


  clienthandler thread is terminating

  client download message contents:
<file>package.xml</file>_
```

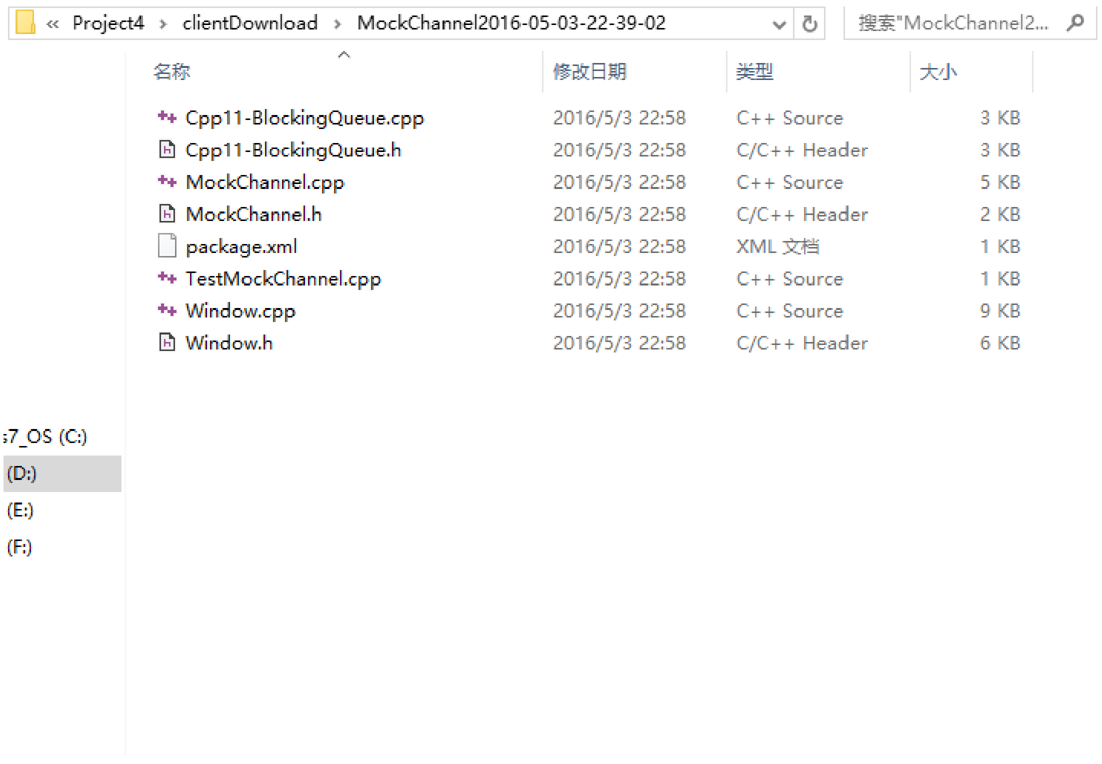
In the console, Server sends files to the client and client receives the message and files from the server. Also in the GUI, you can see it notify you have received the files from server:



```
Command:ShowMessage
Sendr:localhost@8081
Recvr:localhost@8080
finish sending

You can see the information after sending command from client

MockChannelversion2016-05-03-22-39-02
```

Begin Upload/Begin Download

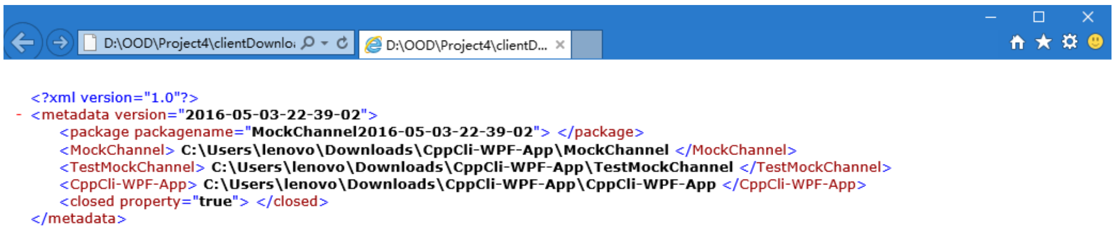
Now you can see in the path we can find the packages. I use the file clientDownload to store packages from server.


« Project4 › clientDownload › MockChannel2016-05-03-22-39-02
搜索"MockChannel2...
;7_OS (C:)
(D:)
(E:)
(F:)

名称	修改日期	类型	大小
Cpp11-BlockingQueue.cpp	2016/5/3 22:58	C++ Source	3 KB
Cpp11-BlockingQueue.h	2016/5/3 22:58	C/C++ Header	3 KB
MockChannel.cpp	2016/5/3 22:58	C++ Source	5 KB
MockChannel.h	2016/5/3 22:58	C/C++ Header	2 KB
package.xml	2016/5/3 22:58	XML 文档	1 KB
TestMockChannel.cpp	2016/5/3 22:58	C++ Source	1 KB
Window.cpp	2016/5/3 22:58	C++ Source	9 KB
Window.h	2016/5/3 22:58	C/C++ Header	6 KB

(3) About the xml file. I use the example of package MockChannel2016-05-03-22-39-02. When I open the xml file of the package, you can see I set the check-in data as

its version and package name will be the parent of the packages with dependencies. And the property of package is set to closed, which the property of closed is ture. In my code, I have set the properties of all the files are closed. I think if we want to change it with GUI, we must add some textboxes into the GUI to change the property of the files. But, I think it will spend some days. So I just set the properties of all the files are closed.



```
  <?xml version="1.0"?>
- <metadata version="2016-05-03-22-39-02">
      <package packagename="MockChannel2016-05-03-22-39-02"> </package>
      <MockChannel> C:\Users\lenovo\Downloads\CppCli-WPF-App\MockChannel </MockChannel>
      <TestMockChannel> C:\Users\lenovo\Downloads\CppCli-WPF-App\TestMockChannel </TestMockChannel>
      <CppCli-WPF-App> C:\Users\lenovo\Downloads\CppCli-WPF-App\CppCli-WPF-App </CppCli-WPF-App>
      <closed property="true"> </closed>
  </metadata>
```