

Qt –Dialog-based Networking Server Application

Yih-Chuan Lin

CSIE Windows Programming Class

National Formosa University

Qt- Networking

- 學習目的
 - 認識Qt網路連線類別功能
 - 練習使用Qt TcpServer類別
 - 學習如何整合網路連線功能於對話盒視窗應用程式

Qt- Networking

- **QTcpServer Class provides for TCP-based server:**

Header:	<code>#include <QtNetwork></code>
qmake:	<code>QT += network</code>
Inherits:	<code>QObject</code>

Qt- Networking

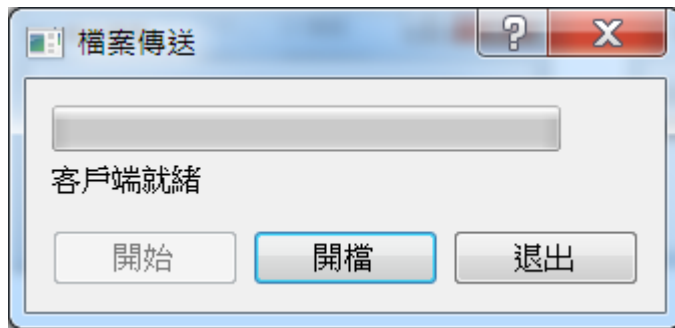
- **QTcpServer Class : a Server for accepting TCP connection**



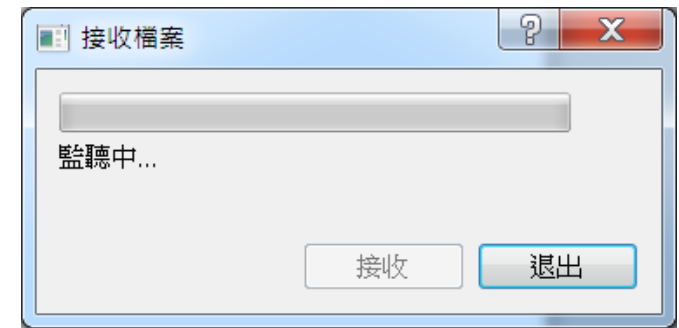
Qt- Networking

- The application data format (user-defined)

Client

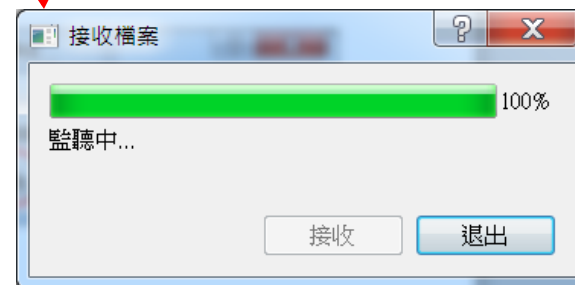
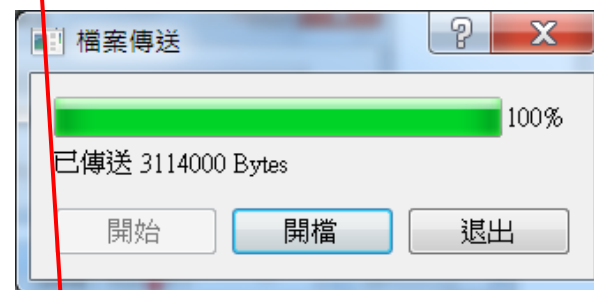
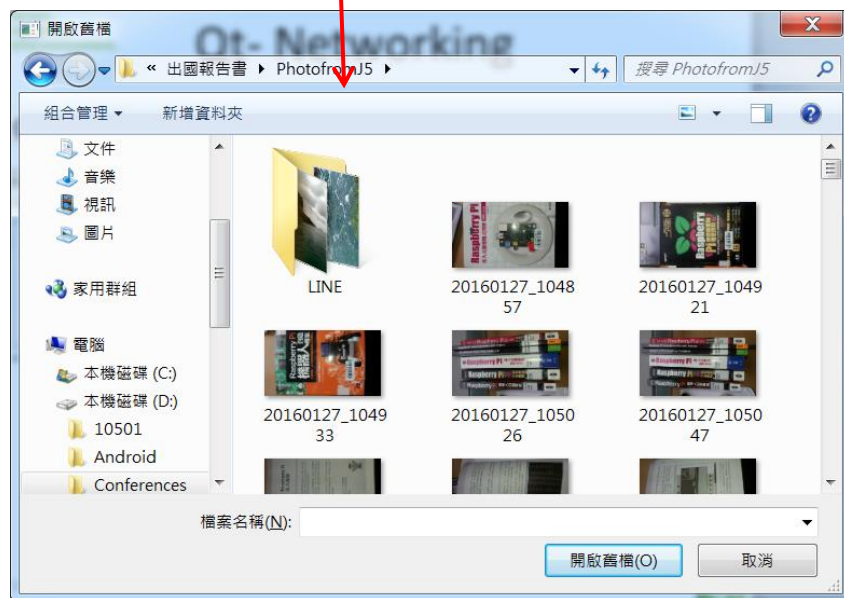
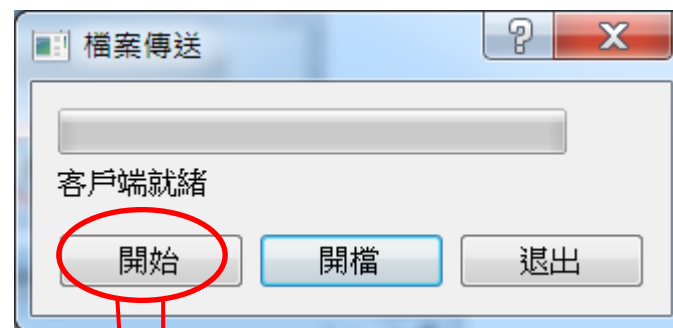
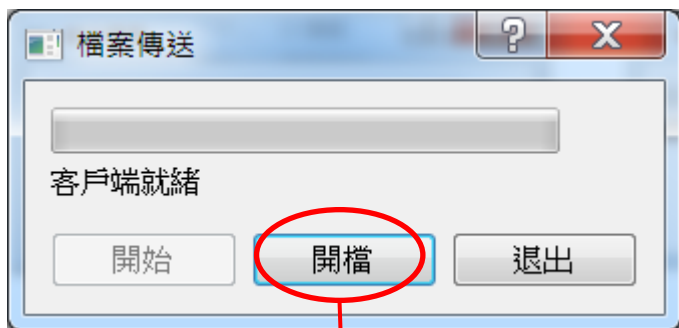


Server



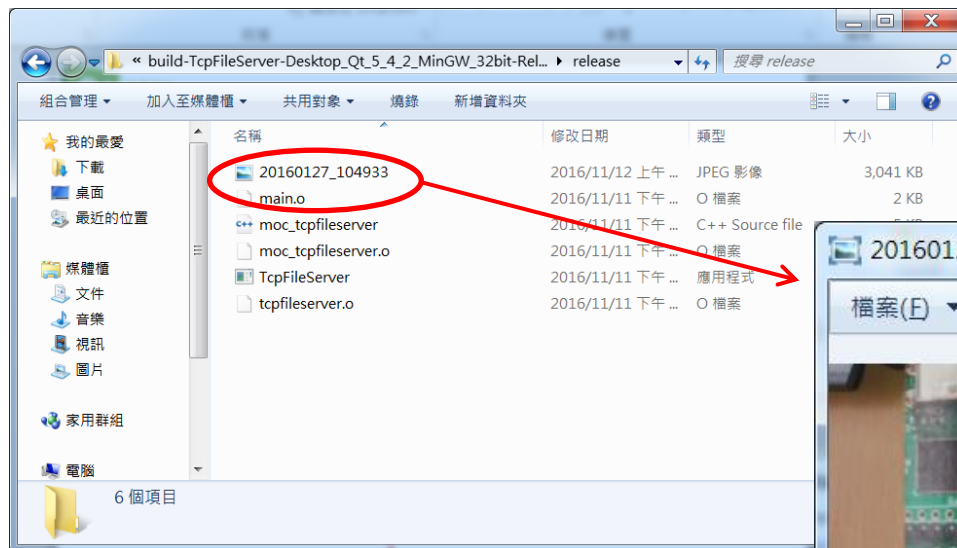
Qt- Networking

- **QTcpServer Class : a Server for accepting TCP connection**



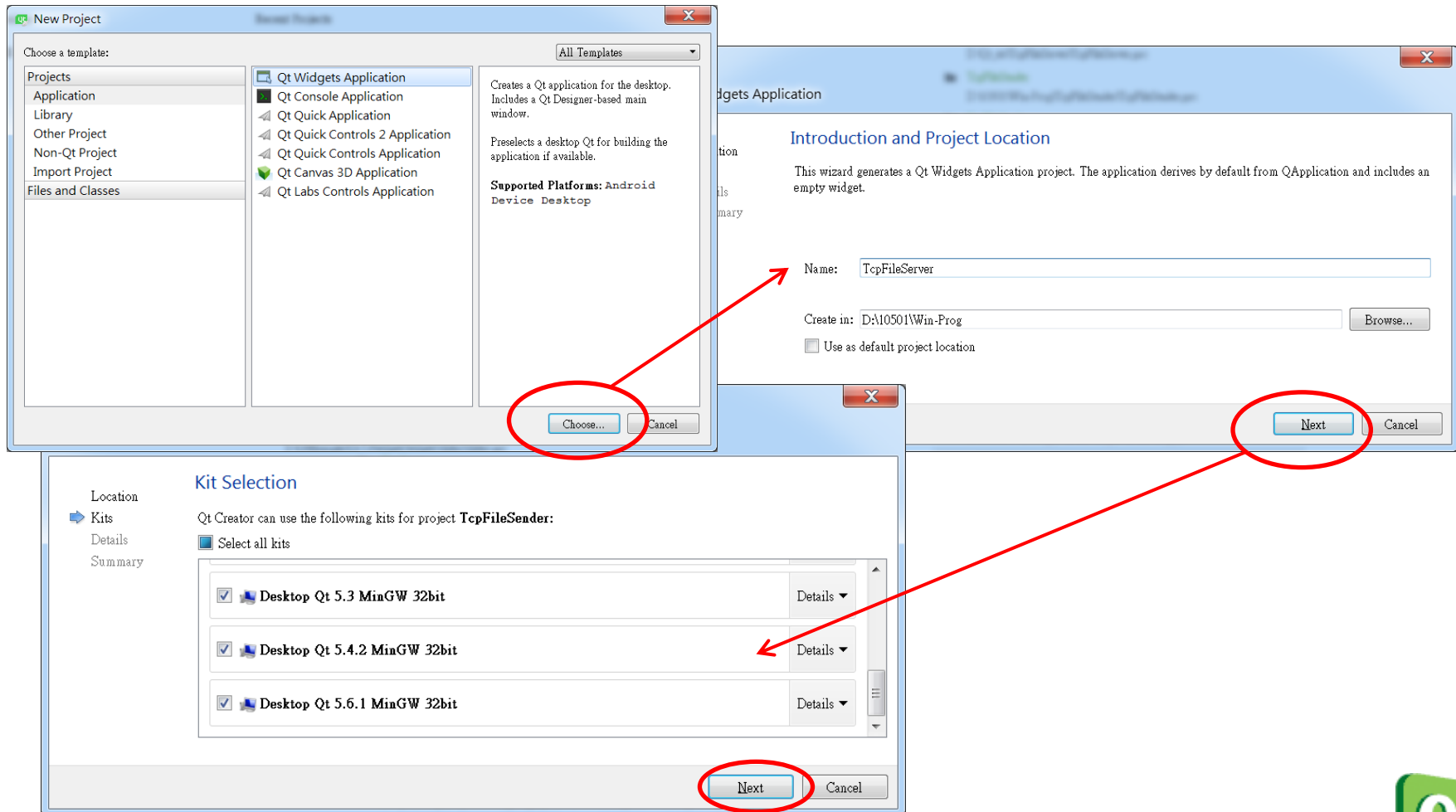
Qt- Networking

- **QTcpServer Class : a Server for accepting TCP connection**



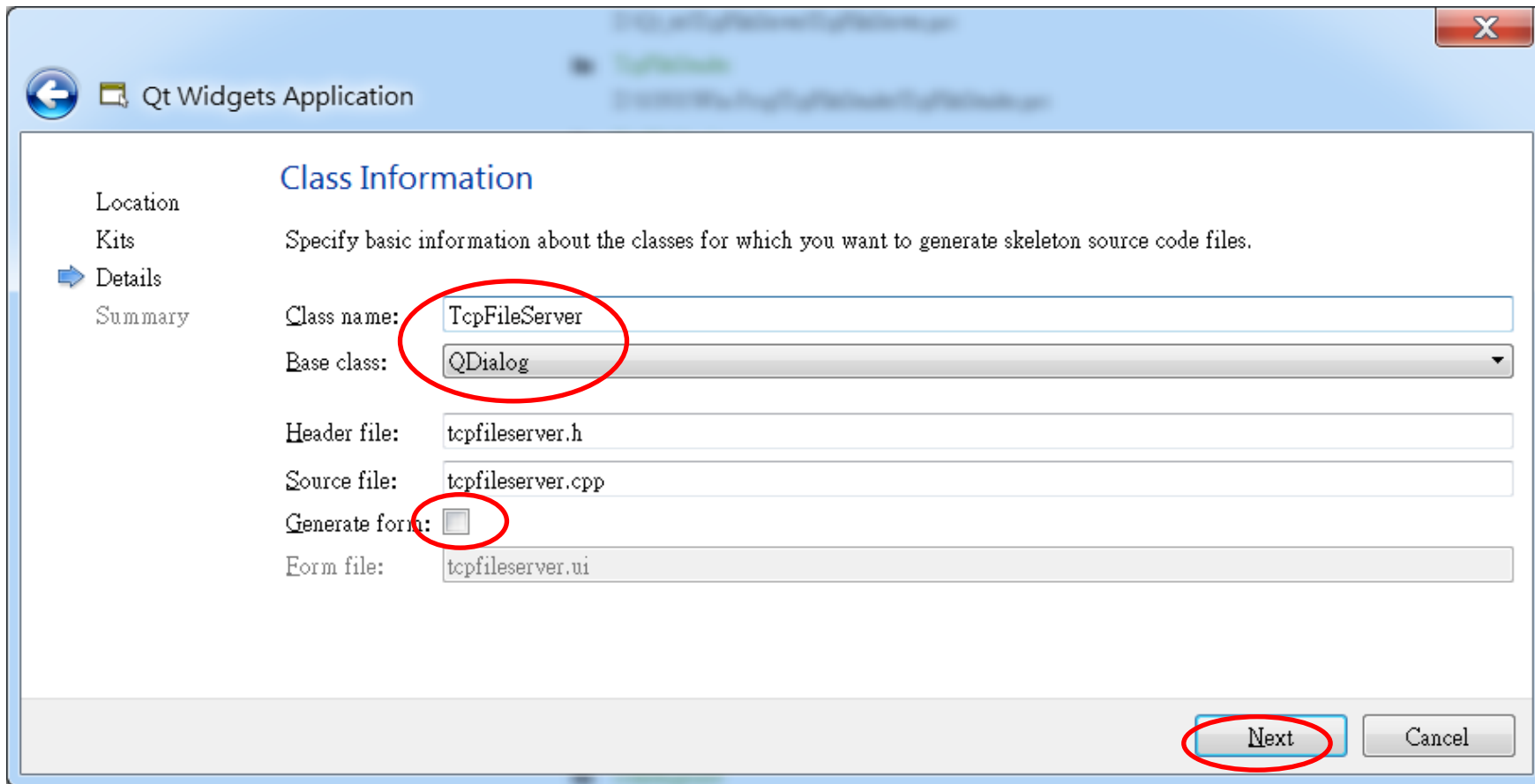
Qt- Networking

- Create a file server dialog-based application:



Qt- Networking

- Create a file server dialog-based application:



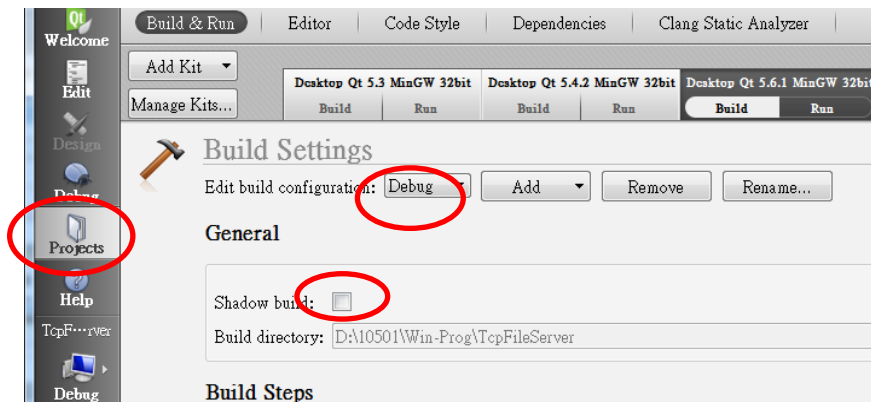
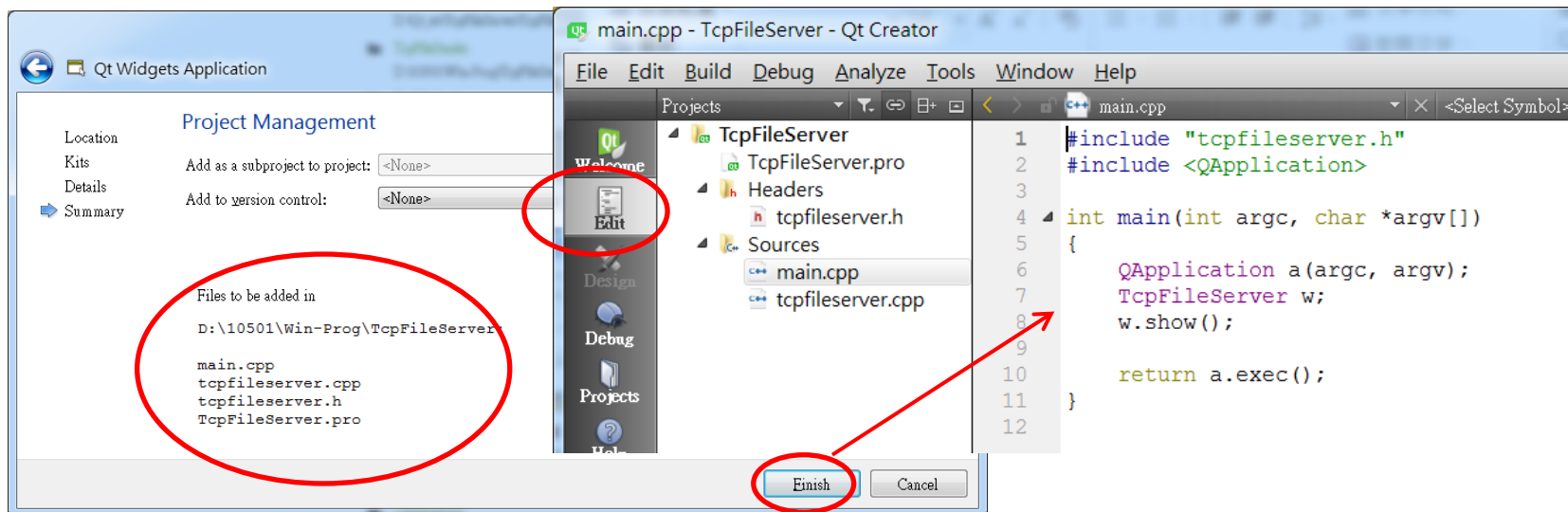
The image shows the 'Class Information' dialog box in Qt Creator. The dialog is titled 'Qt Widgets Application' and has a sidebar with 'Location', 'Kits', 'Details' (selected), and 'Summary'. The main area is titled 'Class Information' and contains the following fields:

- Class name: (circled in red)
- Base class: (circled in red)
- Header file:
- Source file:
- Generate form: ☐ (circled in red)
- Form file:

At the bottom right, there are 'Next' and 'Cancel' buttons. The 'Next' button is circled in red.

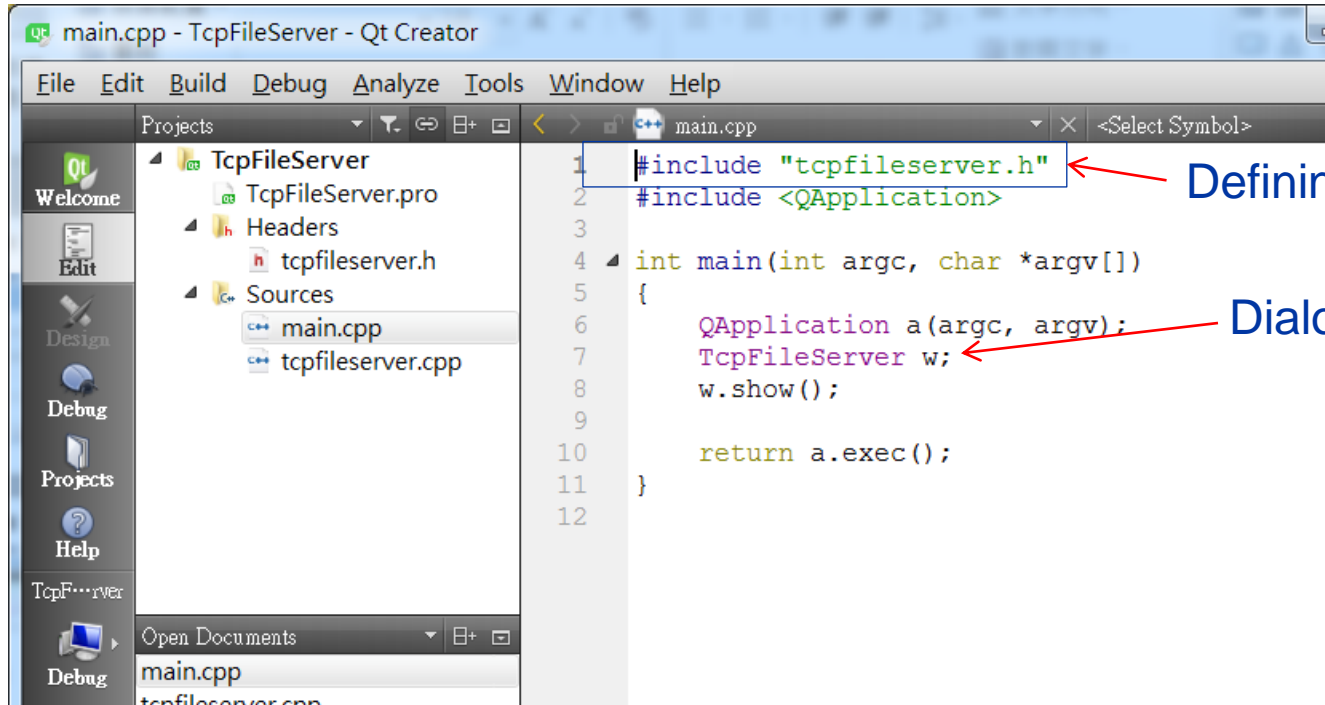
Qt- Networking

● Create a file server dialog-based application:



Qt- Networking

- Check out the main function:



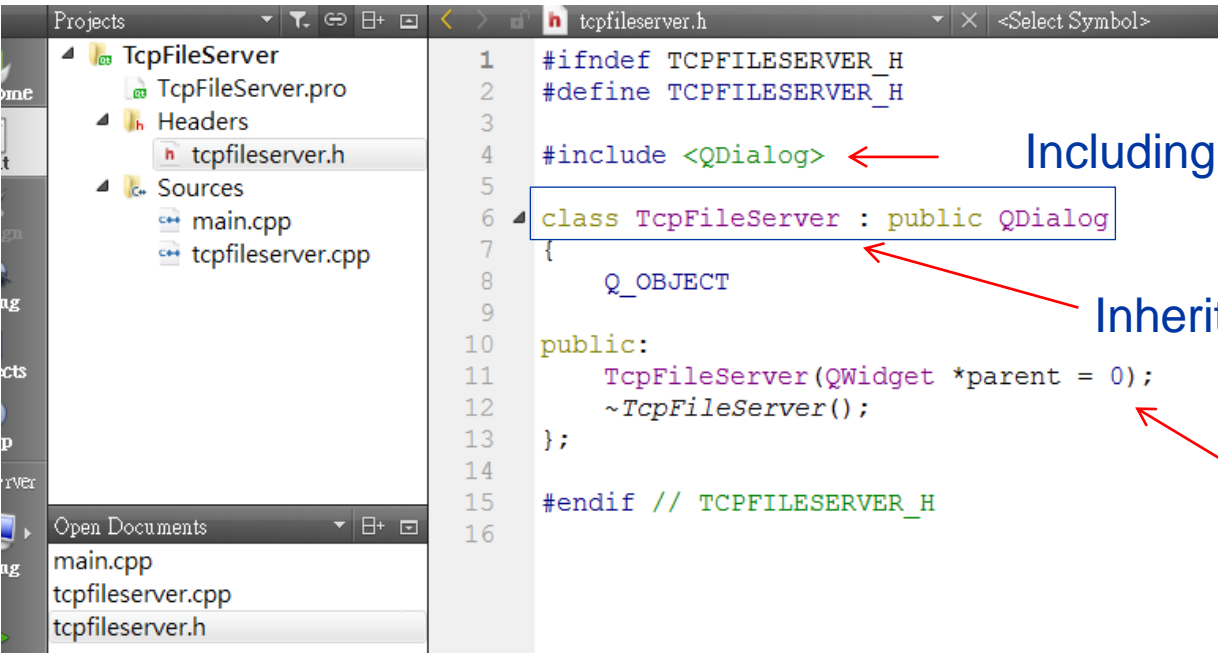
```
1 #include "tcpfileserver.h"
2 #include <QApplication>
3
4 int main(int argc, char *argv[])
5 {
6     QApplication a(argc, argv);
7     TcpFileServer w;
8     w.show();
9
10    return a.exec();
11 }
12
```

Defining Dialog window!

Dialog window object!

Qt-Networking

- Check out the definition of TcpFileServer class:



```
1  #ifndef TCPFILESERVER_H
2  #define TCPFILESERVER_H
3
4  #include <QDialog>
5
6  class TcpFileServer : public QDialog
7  {
8      Q_OBJECT
9
10 public:
11     TcpFileServer(QWidget *parent = 0);
12     ~TcpFileServer();
13 };
14
15 #endif // TCPFILESERVER_H
16
```

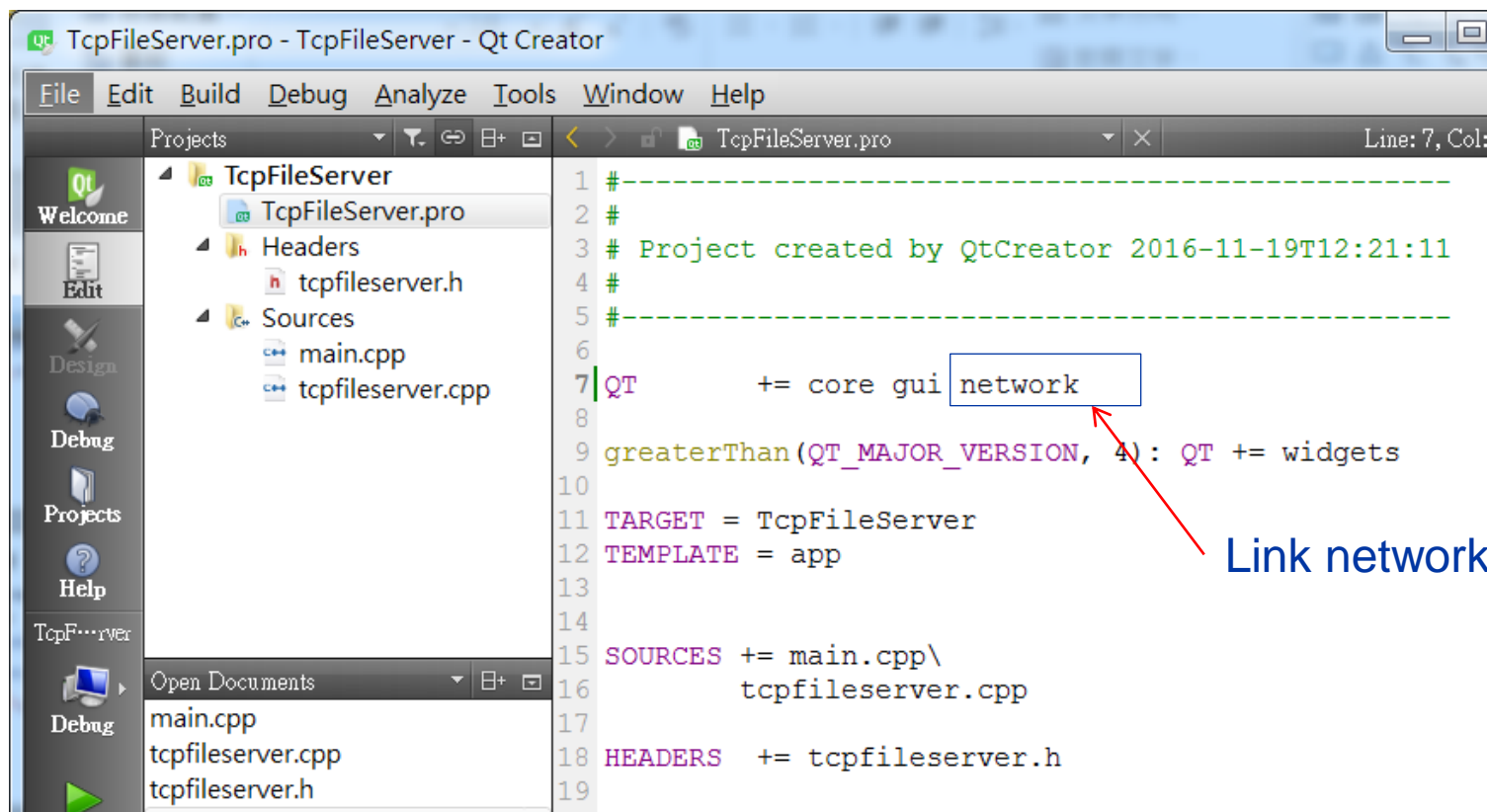
Including QDialog header file!

Inheriting from QDialog Class!

Constructor!

Networking

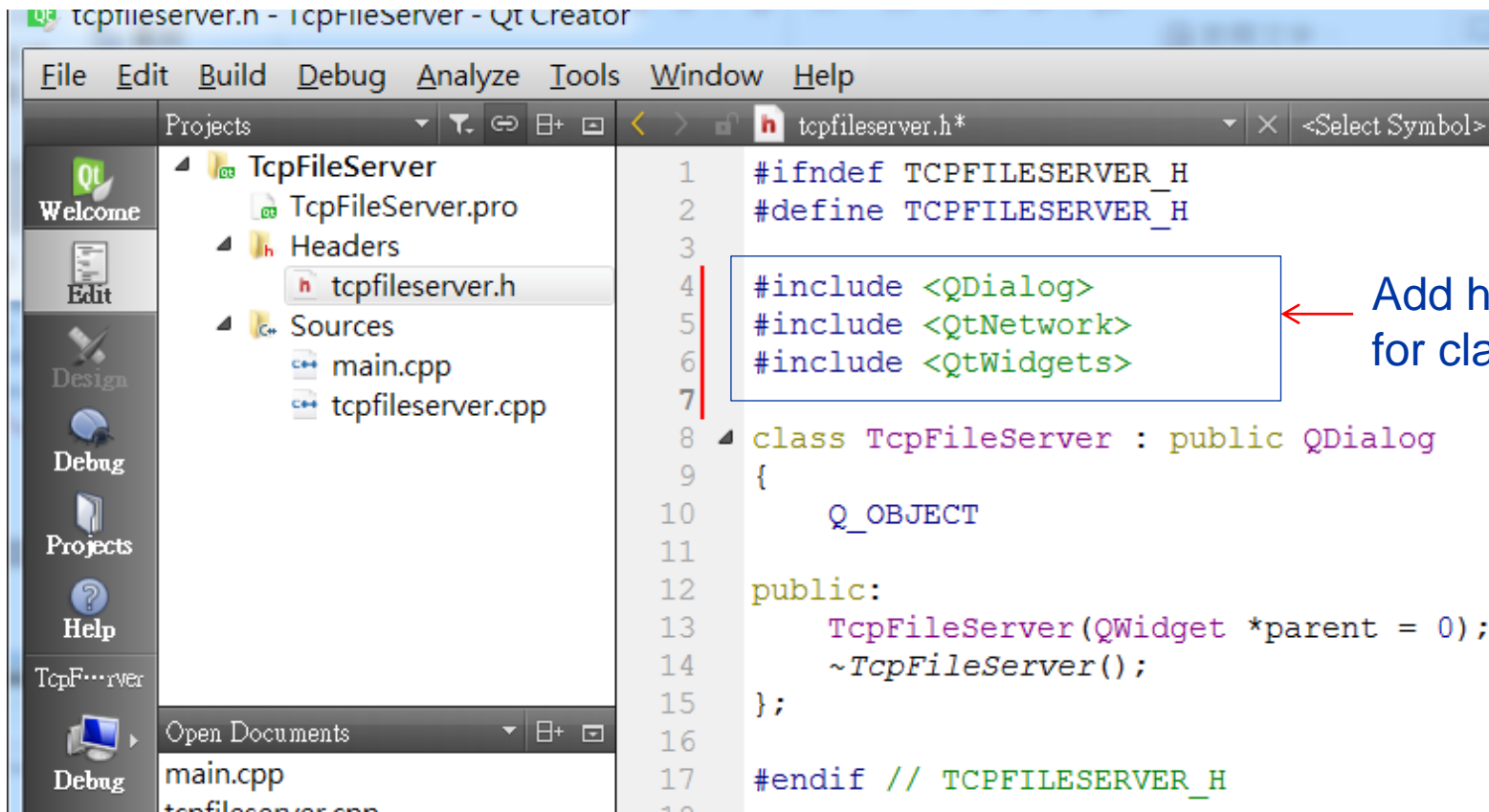
- Check out the qmake project file:



Link network module!

Qt- Networking

- **Modify the definition of TcpFileServer class:**



```
1  #ifndef TCPFILESERVER_H
2  #define TCPFILESERVER_H
3
4  #include <QDialog>
5  #include <QtNetwork>
6  #include <QtWidgets>
7
8  class TcpFileServer : public QDialog
9  {
10     Q_OBJECT
11
12     public:
13         TcpFileServer(QWidget *parent = 0);
14         ~TcpFileServer();
15 };
16
17 #endif // TCPFILESERVER_H
```

← Add header files
for classes used.

Qt- Networking

- **Modify the definition of TcpFileServer class:**

```

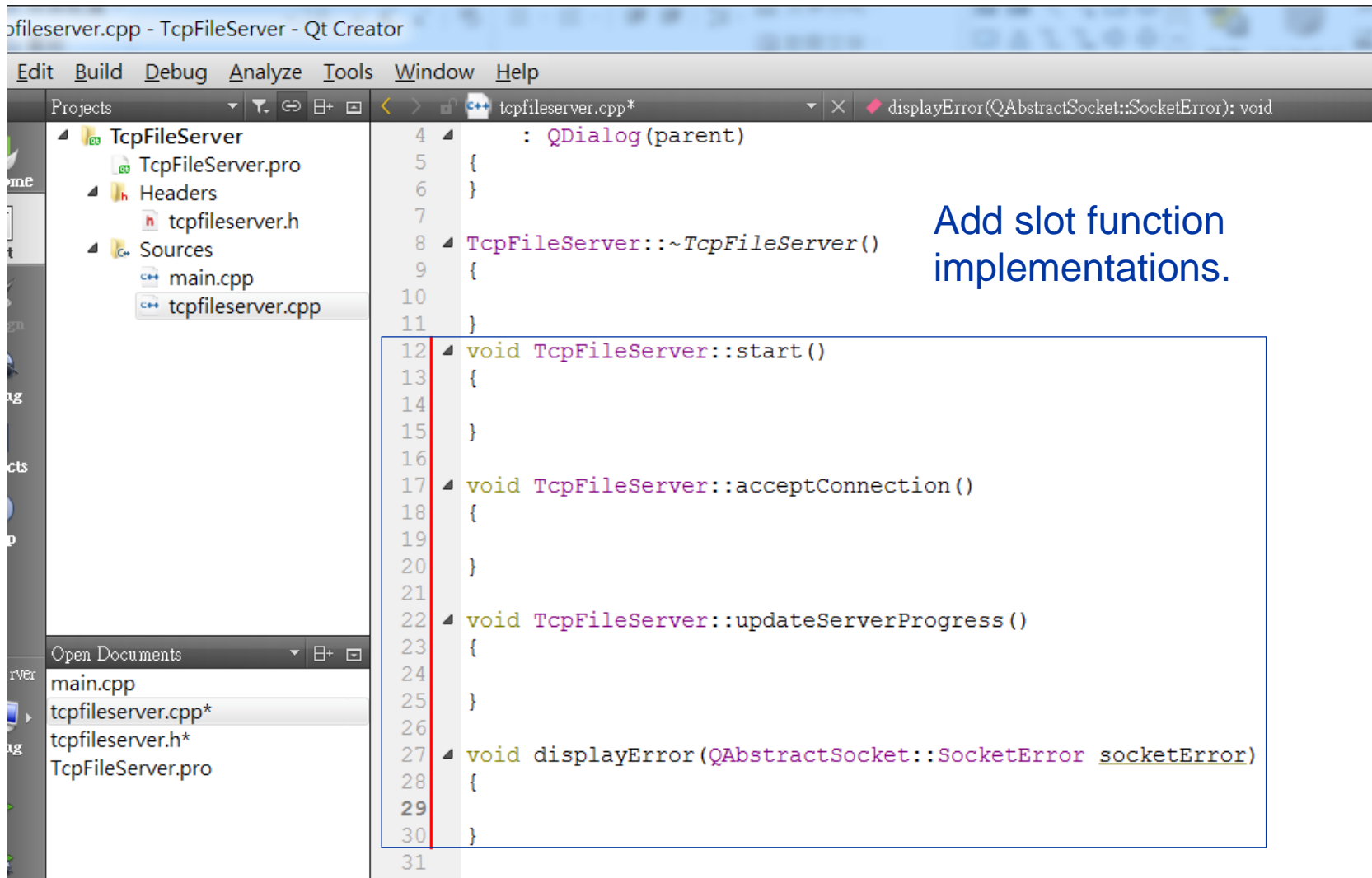
1  #ifndef TCPFILESERVER_H
2  #define TCPFILESERVER_H
3
4  #include <QDialog>
5  #include <QtNetwork>
6  #include <QtWidgets>
7
8  class TcpFileServer : public QDialog
9  {
10     Q_OBJECT
11
12 public:
13     TcpFileServer(QWidget *parent = 0);
14     ~TcpFileServer();
15 public slots:
16     void start();
17     void acceptConnection();
18     void updateServerProgress();
19     void displayError(QAbstractSocket::SocketError socketError);
20 };
21
22 #endif // TCPFILESERVER_H
23

```

Add slot functions.

Qt- Networking

- Add slot function implementation to tcpfileserver.cpp:



Qt Creator interface showing the implementation of slot functions in `tcpfileserver.cpp`. The code is as follows:

```

4      : QDialog(parent)
5  {
6  }
7
8  TcpFileServer::~TcpFileServer()
9  {
10 }
11
12 void TcpFileServer::start()
13 {
14 }
15
16
17 void TcpFileServer::acceptConnection()
18 {
19 }
20
21
22 void TcpFileServer::updateServerProgress()
23 {
24 }
25
26
27 void displayError(QAbstractSocket::SocketError socketError)
28 {
29 }
30
31

```

A red box highlights the implementation of the `displayError` slot function, which is a static void function taking a `QAbstractSocket::SocketError` parameter.

Add slot function
implementations.

Qt- Networking

- **Modify the definition of TcpFileServer class:**

```

8  class TcpFileServer : public QDialog
9  {
10     Q_OBJECT
11
12 public:
13     TcpFileServer(QWidget *parent = 0);
14     ~TcpFileServer();
15
16 public slots:
17     void start();
18     void acceptConnection();
19     void updateServerProgress();
20     void displayError(QAbstractSocket::SocketError socketError);
21
22 private:
23     QProgressBar      *serverProgressBar;
24     QLabel            *serverStatusLabel;
25     QPushButton       *startButton;
26     QPushButton       *quitButton;
27     QDialogButtonBox  *buttonBox;
28
29     QTcpServer        tcpServer;
30     QTcpSocket         *tcpServerConnection;
31     qint64             totalBytes;
32     qint64             byteReceived;
33     qint64             fileNameSize;
34     QString            fileName;
35     QFile              *localFile;
36     QByteArray         inBlock;
37 };
  
```

Add private variables.

Qt- Networking

- **Modify the constructor of TcpFileServer class:**

Generating user interface fields!

```

1  #include "tcpfileserver.h"
2
3  TcpFileServer::TcpFileServer(QWidget *parent)
4      : QDialog(parent)
5  {
6      totalBytes = 0;
7      byteReceived = 0;
8      fileNameSize = 0;
9      serverProgressBar = new QProgressBar;
10     serverStatusLabel = new QLabel(tr("伺服器端就緒"));
11     startButton = new QPushButton(tr("接收"));
12     quitButton = new QPushButton(tr("退出"));
13     buttonBox = new QDialogButtonBox;
14     buttonBox->addButton(startButton, QDialogButtonBox::ActionRole);
15     buttonBox->addButton(quitButton, QDialogButtonBox::RejectRole);
16 }
17
18 TcpFileServer::~TcpFileServer()
19 {
20 }
21
22 void TcpFileServer::start()
23 {
24 }
25
26
27 void TcpFileServer::acceptConnection()
28 {

```

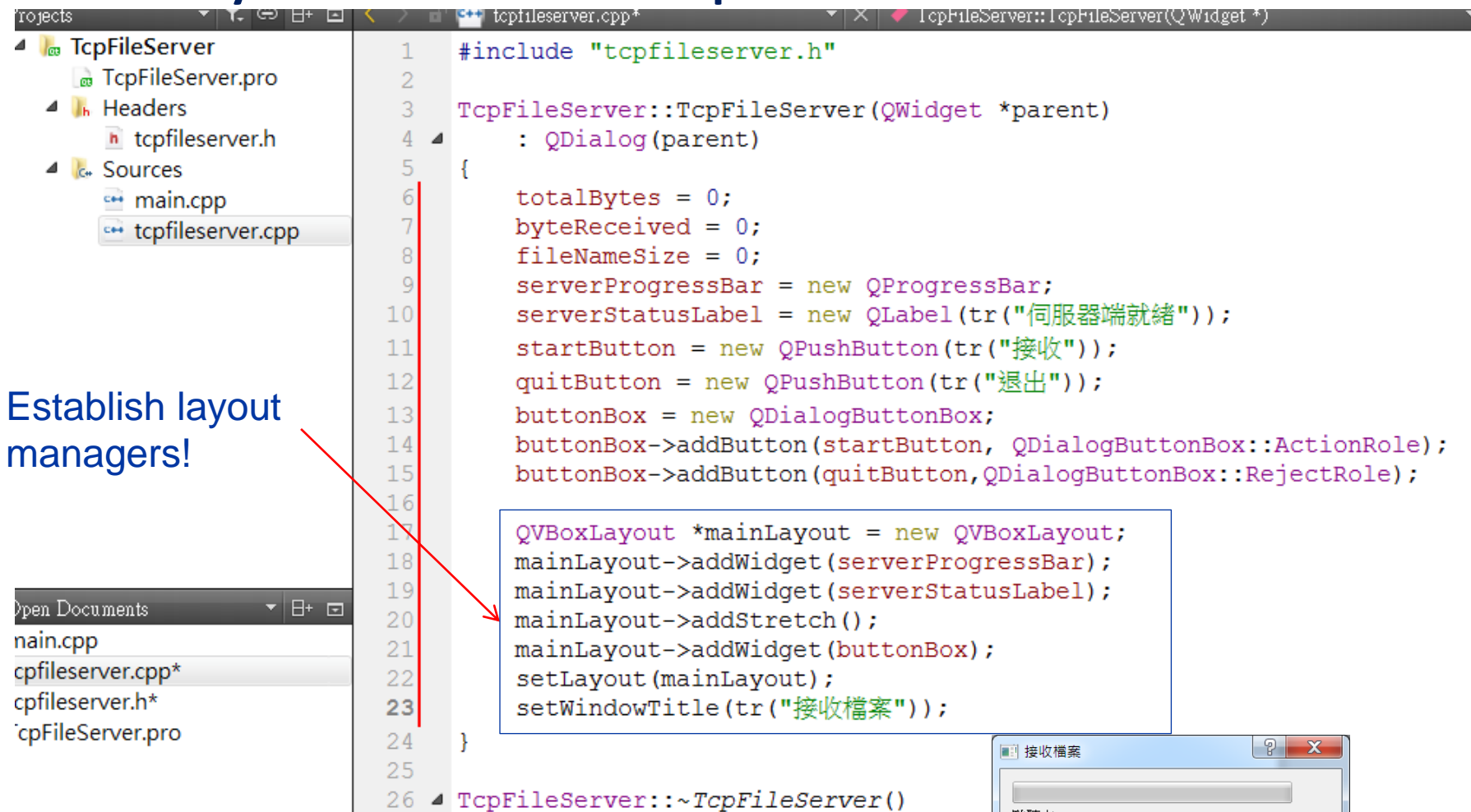
接收檔案

監聽中...

接收 退出

Qt- Networking

● Modify the constructor of TcpFileServer class:



Establish layout managers!

```

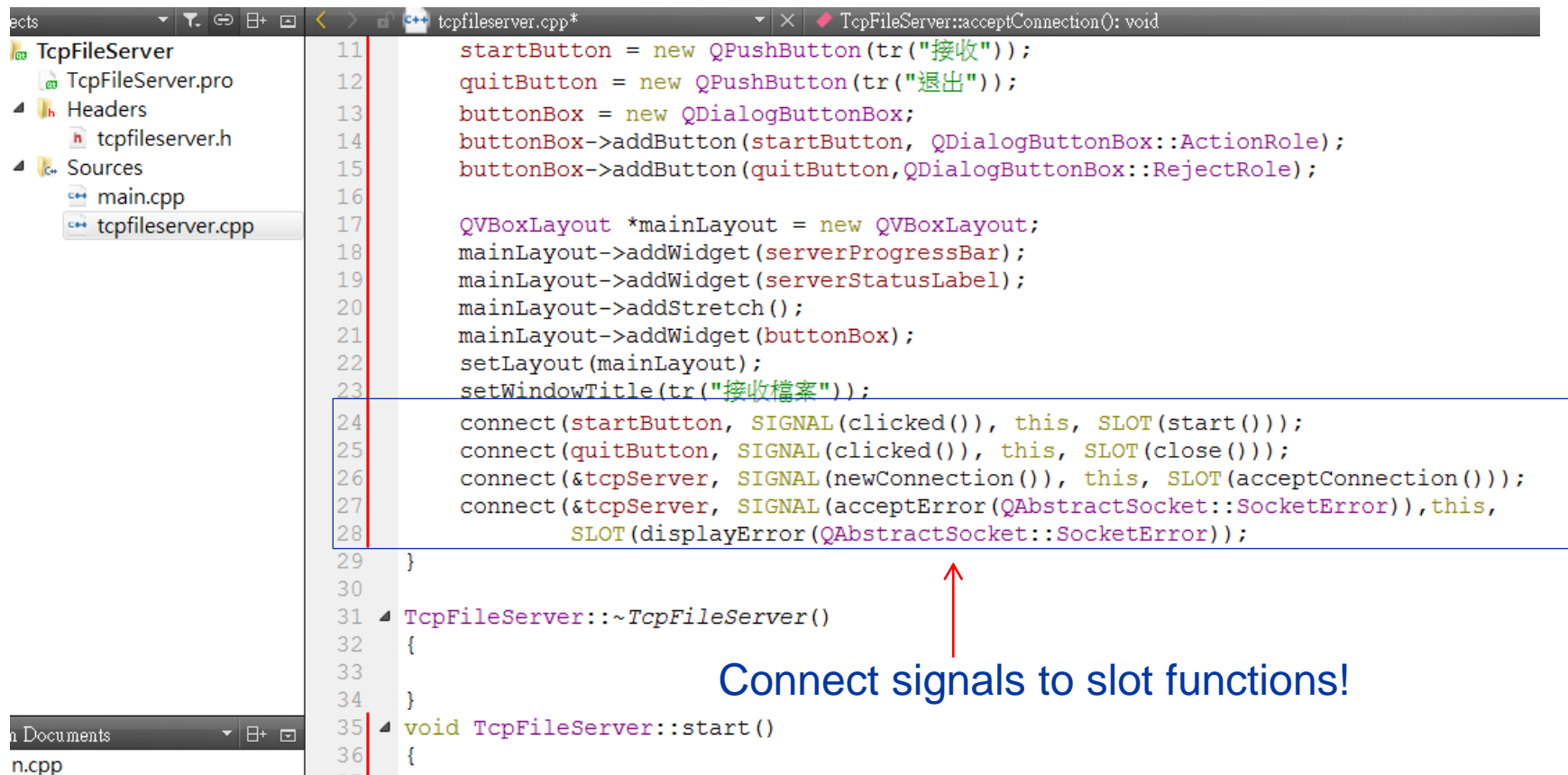
1  #include "tcpfileserver.h"
2
3  TcpFileServer::TcpFileServer(QWidget *parent)
4      : QDialog(parent)
5  {
6      totalBytes = 0;
7      byteReceived = 0;
8      fileNameSize = 0;
9      serverProgressBar = new QProgressBar;
10     serverStatusLabel = new QLabel(tr("伺服器端就緒"));
11     startButton = new QPushButton(tr("接收"));
12     quitButton = new QPushButton(tr("退出"));
13     buttonBox = new QDialogButtonBox;
14     buttonBox->addButton(startButton, QDialogButtonBox::ActionRole);
15     buttonBox->addButton(quitButton, QDialogButtonBox::RejectRole);
16
17     QVBoxLayout *mainLayout = new QVBoxLayout;
18     mainLayout->addWidget(serverProgressBar);
19     mainLayout->addWidget(serverStatusLabel);
20     mainLayout->addStretch();
21     mainLayout->addWidget(buttonBox);
22     setLayout(mainLayout);
23     setWindowTitle(tr("接收檔案"));
24 }
25
26 TcpFileServer::~TcpFileServer()

```

Qt

Qt- Networking

- **Modify the constructor of TcpFileServer class:**



```

11  startButton = new QPushButton(tr("接收"));
12  quitButton = new QPushButton(tr("退出"));
13  buttonBox = new QDialogButtonBox;
14  buttonBox->addButton(startButton, QDialogButtonBox::ActionRole);
15  buttonBox->addButton(quitButton, QDialogButtonBox::RejectRole);
16
17  QVBoxLayout *mainLayout = new QVBoxLayout;
18  mainLayout->addWidget(serverProgressBar);
19  mainLayout->addWidget(serverStatusLabel);
20  mainLayout->addStretch();
21  mainLayout->addWidget(buttonBox);
22  setLayout(mainLayout);
23  setWindowTitle(tr("接收檔案"));
24  connect(startButton, SIGNAL(clicked()), this, SLOT(start()));
25  connect(quitButton, SIGNAL(clicked()), this, SLOT(close()));
26  connect(&tcpServer, SIGNAL(newConnection()), this, SLOT(acceptConnection()));
27  connect(&tcpServer, SIGNAL(acceptError(QAbstractSocket::SocketError)), this,
28          SLOT(displayError(QAbstractSocket::SocketError)));
29  }
30
31  TcpFileServer::~TcpFileServer()
32  {
33  }
34
35  void TcpFileServer::start()
36  {

```

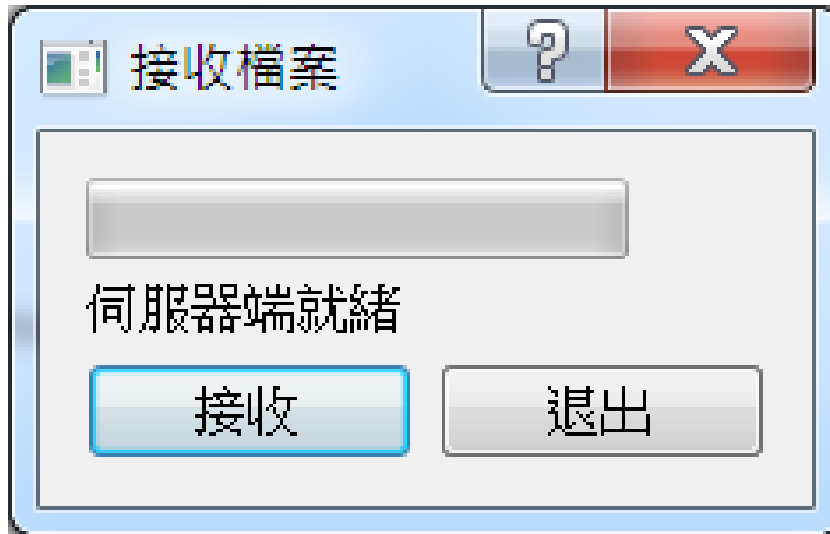
Connect signals to slot functions!



Code less.
Create more.
Deploy everywhere.

Qt- Networking

- **Build and run the project:**



Try to interact with the dialog window displayed!
Do you see any reaction when you click on the buttons?



Qt- Networking

- Implement slot functions start() :

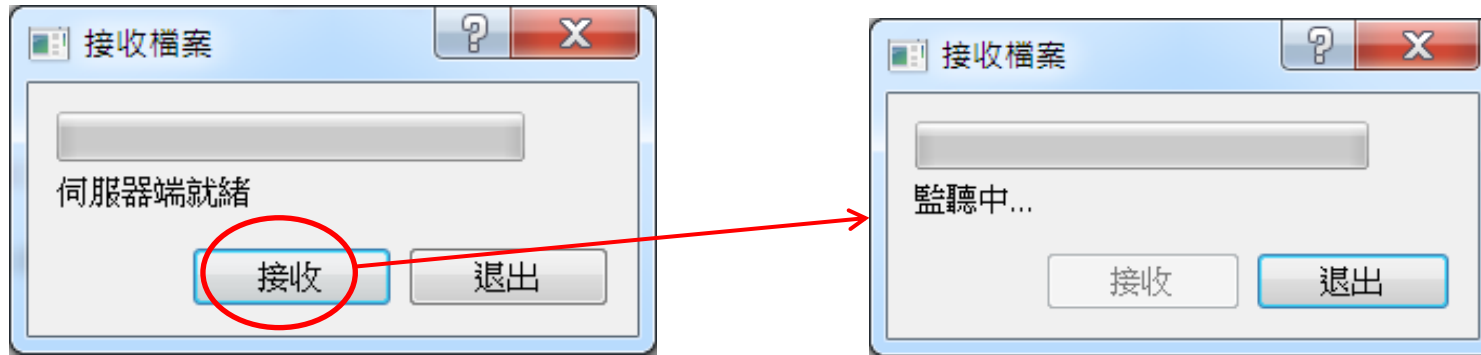
```
server
leServer.pro
ers
pfileserv.h
ces
ain.cpp
pfileserv.cpp

31  TcpFileServer::~TcpFileServer()
32  {
33
34  }
35  void TcpFileServer::start()
36  {
37      startButton->setEnabled(false);
38      byteReceived = 0;
39      fileNameSize = 0;
40      while(!tcpServer.isListening() &&
41          !tcpServer.listen(QHostAddress::AnyIPv4, 16689))
42      {
43          QMessageBox::StandardButton ret = QMessageBox::critical(this,
44                                                                  tr("迴圈"),
45                                                                  tr("無法啟動伺服器: %1.").arg(tcpServer.errorString()),
46                                                                  QMessageBox::Retry | QMessageBox::Cancel);
47          if (ret == QMessageBox::Cancel) return;
48      }
49      serverStatusLabel->setText(tr("監聽中..."));
50  }
51
52  void TcpFileServer::acceptConnection()
53  {
54
55  }
56
```

Check to see if the server is in listening?
Make the server in listening if it is not.

Qt- Networking

- **Build and run the project:**



Try to interact with the “接收” button!

Qt- Networking

● Implement slot function acceptConnection():

```

43         QMessageBox::StandardButton ret = QMessageBox::critical(this,
44                                                                 tr("迴圈"),
45                                                                 tr("無法啟動伺服器: %1.").arg(tcpServer.error()),
46                                                                 QMessageBox::Retry | QMessageBox::Cancel);
47
48         if (ret == QMessageBox::Cancel) return;
49         serverStatusLabel->setText(tr("監聽中..."));
50     }
51
52     void TcpFileServer::acceptConnection()
53     {
54         tcpServerConnection = tcpServer.nextPendingConnection();//取得已接受的客戶端連線
55         connect(tcpServerConnection, SIGNAL(readyRead()),
56               this, SLOT(updateServerProgress()));//連接readyRead() 訊號至updateServerProgress() 槽函數
57         connect(tcpServerConnection, SIGNAL(error(QAbstractSocket::SocketError)),
58               this, SLOT(displayError(QAbstractSocket::SocketError)));//連接error() 訊號至displayError() 槽函數
59         serverStatusLabel->setText(tr("接受連線"));
60         tcpServer.close(); //暫停接受客戶端連線
61     }
62
63     void TcpFileServer::updateServerProgress()
64     {
65

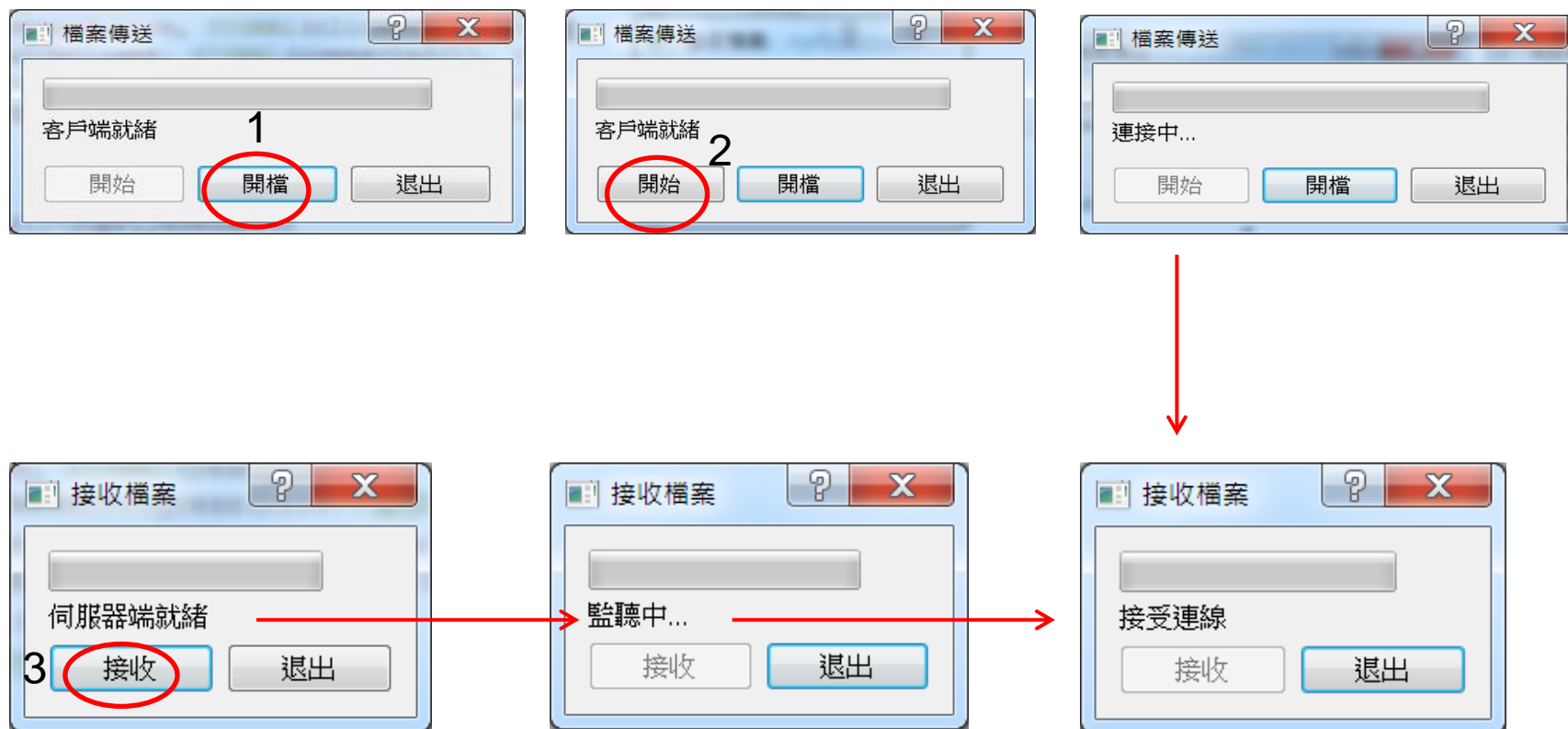
```

Called by **newConnection()** signal from **tcpServer** object!

1. Get the QTcpSocket object '**tcpServerConnection**', which has connected to the client.
2. Connect signals **readyRead()** and **error()** to the slot functions **updateServerProgress()** and **displayError()**.

Qt- Networking

- **Build and run the project:**



Server has accepted the connection.

Qt- Networking

● Implement slot function updateServerProgress():

```

TcpFileServer
  TcpFileServer.pro
  Headers
    tcpfileserver.h
  Sources
    main.cpp
    tcpfileserver.cpp
  Documents
    cpp
    eServer.cpp*
    eServer.h
    eServer.pro
  tcpfileserver.cpp*
  TcpFileServer::updateServerProgress(): void

63 void TcpFileServer::updateServerProgress()
64 {
65     QDataStream in(tcpServerConnection);
66     in.setVersion(QDataStream::Qt_4_6);
67     if(byteReceived <= sizeof(qint64)*2)
68     {
69         if((fileNameSize == 0) && (tcpServerConnection->bytesAvailable() >=
70             sizeof(qint64)*2))
71         {
72             in >> totalBytes >> fileNameSize;
73             byteReceived += sizeof(qint64)*2;
74         }
75         if((fileNameSize != 0) && (tcpServerConnection->bytesAvailable() >=
76             fileNameSize) )
77         {
78             in >> fileName;
79             byteReceived += fileNameSize;
80             localFile = new QFile(fileName);
81             if(!localFile->open(QFile::WriteOnly))
82             {
83                 QMessageBox::warning(this, tr("應用程式"),
84                     tr("無法讀取檔案 %1:\n%2.").arg(fileName)
85                     .arg(localFile->errorString()));
86                 return;
87             }
88         }else{
89             return;
90         }
91     }
92 }
93

```

Step 1: Read header procedure

Qt- Networking

- Implement slot function `updateServerProgress()`:

```

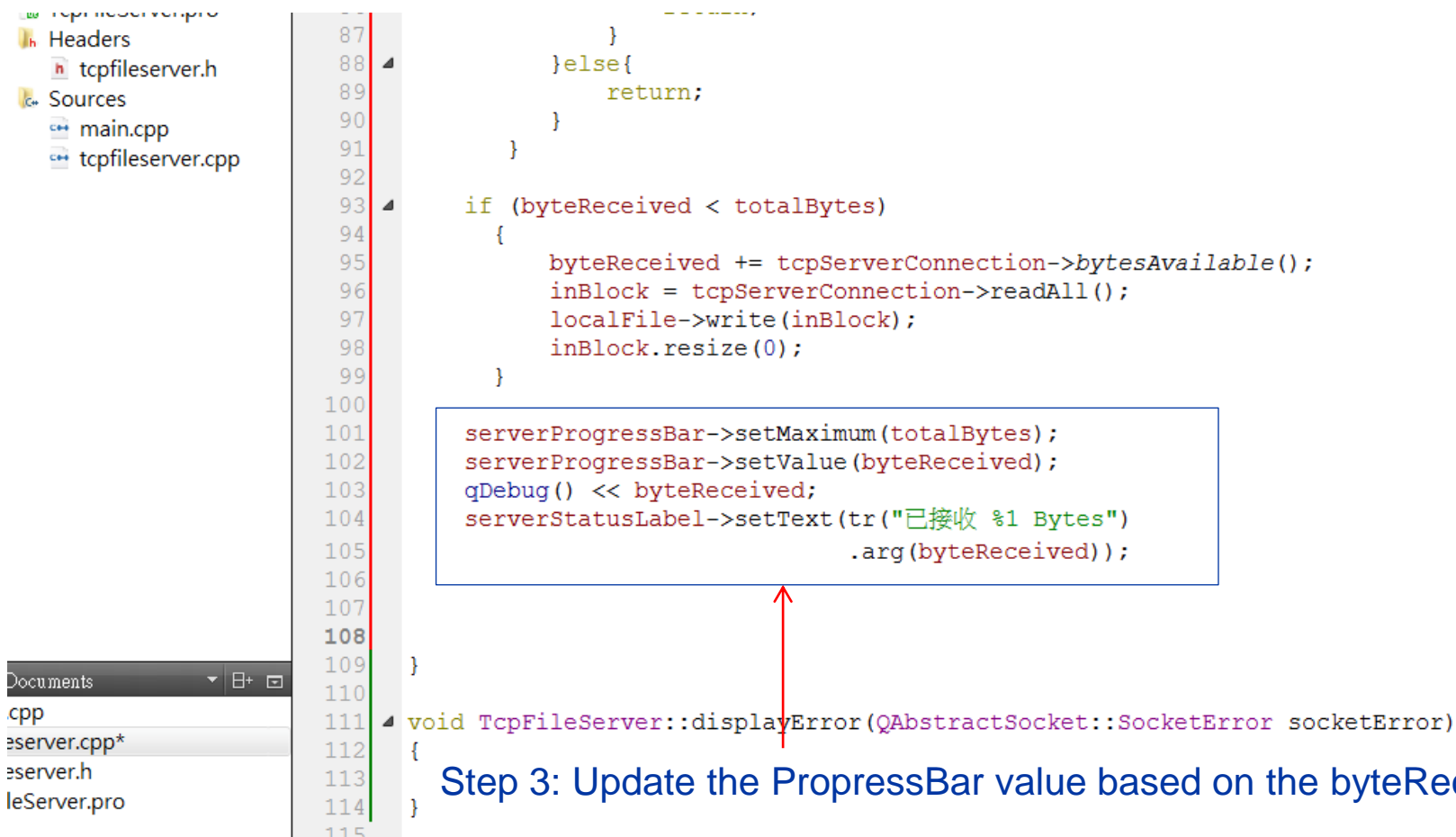
79         byteReceived += fileNameSize;
80         localFile = new QFile(fileName);
81         if(!localFile->open(QFile::WriteOnly))
82         {
83             QMessageBox::warning(this, tr("應用程式"),
84                                   tr("無法讀取檔案 %1:\n%2.").arg(fileName)
85                                   .arg(localFile->errorString()));
86             return;
87         }
88     }else{
89         return;
90     }
91 }
92
93 if (byteReceived < totalBytes)
94 {
95     byteReceived += tcpServerConnection->bytesAvailable();
96     inBlock = tcpServerConnection->readAll();
97     localFile->write(inBlock);
98     inBlock.resize(0);
99 }
100 }
101
102 void TcpFileServer::displayError(QAbstractSocket::SocketError socketError)
103 {
104
105 }
106

```

Step 2: Read all the available data .

Qt- Networking

● Implement slot function updateServerProgress():



```

87         }
88     }else{
89         return;
90     }
91 }
92
93 if (byteReceived < totalBytes)
94 {
95     byteReceived += tcpServerConnection->bytesAvailable();
96     inBlock = tcpServerConnection->readAll();
97     localFile->write(inBlock);
98     inBlock.resize(0);
99 }
100
101 serverProgressBar->setMaximum(totalBytes);
102 serverProgressBar->setValue(byteReceived);
103 qDebug() << byteReceived;
104 serverStatusLabel->setText(tr("已接收 %1 Bytes")
105                             .arg(byteReceived));
106
107
108 }
109
110
111 void TcpFileServer::displayError(QAbstractSocket::SocketError socketError)
112 {
113
114 }
115

```

Step 3: Update the PropressBar value based on the byteReceived.

Qt- Networking

● Implement slot function updateServerProgress():

```

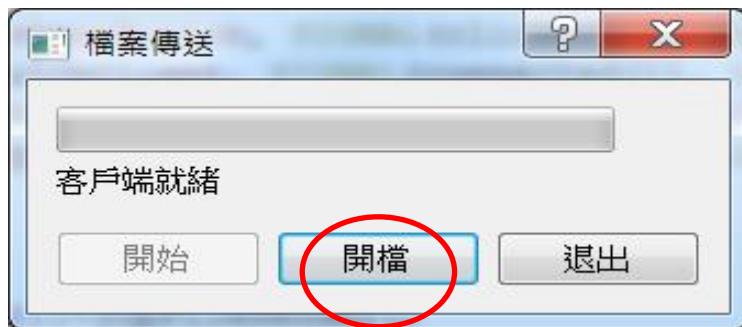
FileServer
TcpFileServer.pro
headers
tcpfileserver.h
Sources
main.cpp
tcpfileserver.cpp
94      {
95          byteReceived += tcpServerConnection->bytesAvailable();
96          inBlock = tcpServerConnection->readAll();
97          localFile->write(inBlock);
98          inBlock.resize(0);
99      }
100
101      serverProgressBar->setMaximum(totalBytes);
102      serverProgressBar->setValue(byteReceived);
103      qDebug() << byteReceived;
104      serverStatusLabel->setText(tr("已接收 %1 Bytes")
105                               .arg(byteReceived));
106
107      if (byteReceived == totalBytes)
108      {
109          tcpServerConnection->close();
110          startButton->setEnabled(true);
111          localFile->fileName();
112          localFile->close();
113          start();
114      }
115
116
117
118  }
119
120  void TcpFileServer::displayError(QAbstractSocket::SocketError socketError)
121  {

```

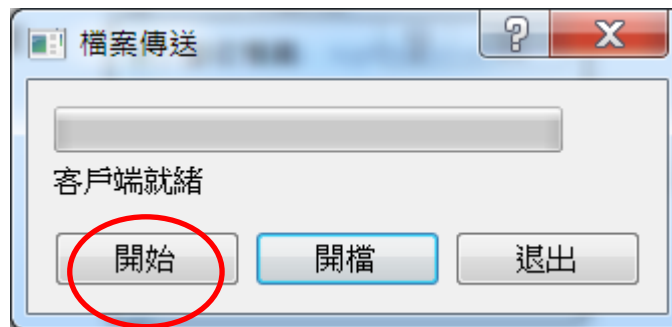
Step 4: Receiving completed procedure.

Qt- Networking

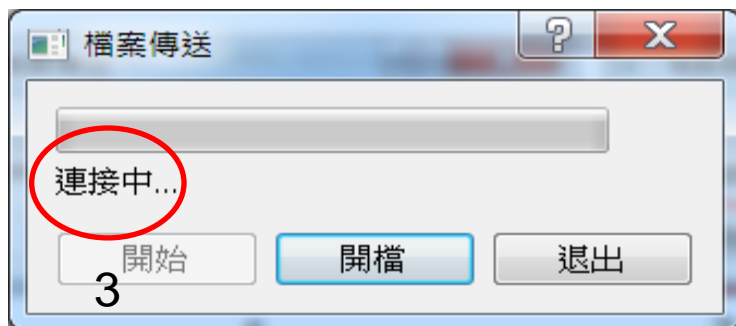
- Build and run the project:



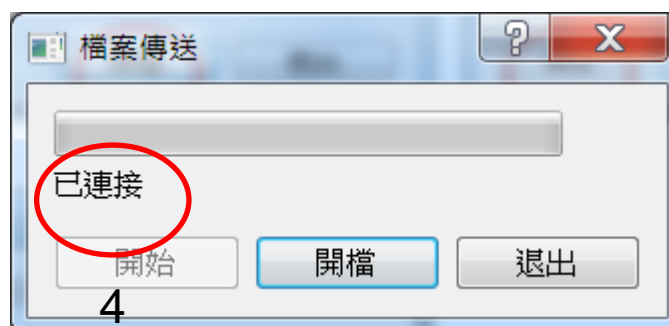
1



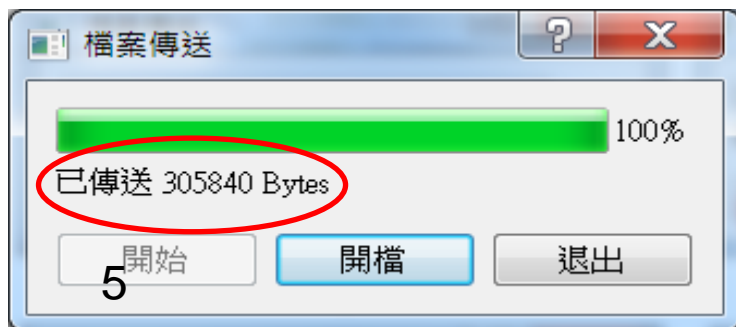
2



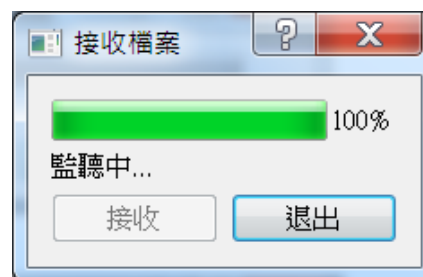
3



4



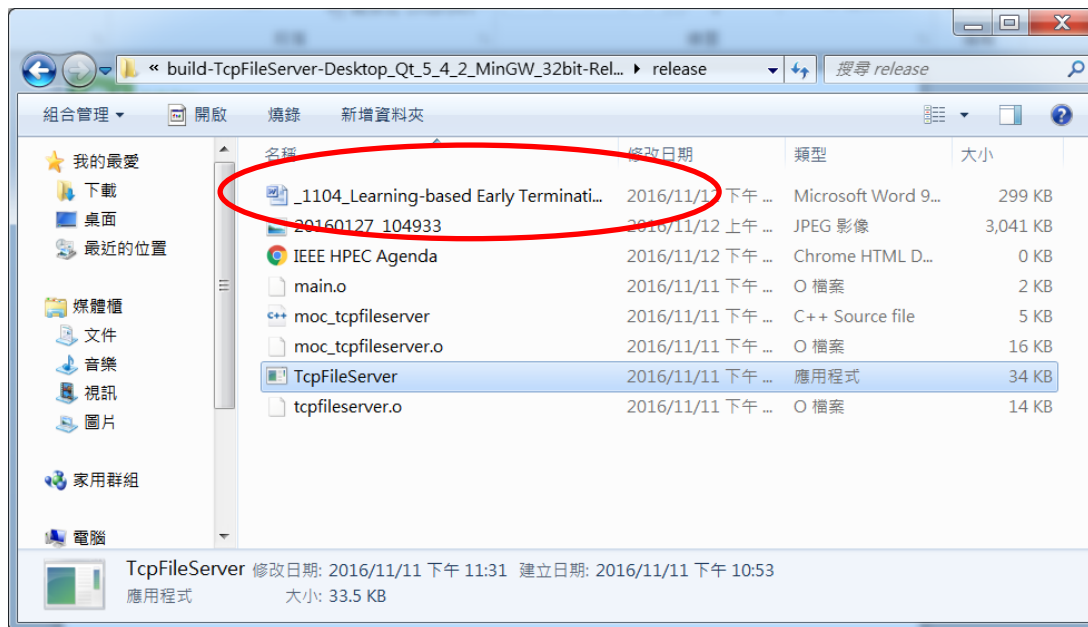
5



Server has received the header part.

Qt- Networking

- Build and run the project:



Open the File Explore to see if the file has been received successfully.

Qt- Networking

● Implement slot function displayError():



```

110         startButton->setEnabled(true);
111         localFile->fileName();
112         localFile->close();
113         start();
114     }
115
116
117
118 }
119
120 void TcpFileServer::displayError(QAbstractSocket::SocketError socketError)
121 {
122     if (socketError == QTcpSocket::RemoteHostClosedError)
123         return;
124
125     QMessageBox::information(this, tr("網絡錯誤"),
126                             tr("產生如下錯誤: %1.")
127                                 .arg(tcpServerConnection->errorString()));
128     tcpServerConnection->close();
129     serverProgressBar->reset();
130     serverStatusLabel->setText(tr("伺服器就緒"));
131     startButton->setEnabled(true);
132 }
133
134
135

```

Error handling procedure!

Qt- Networking

- **Signals of TcpServer used in the project:**

