Code less.
Create more.
Deploy everywhere.

Qt –Dialog-based Networking 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 TcpSocket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

- **QTcpSocket Class is used for TCP communication end-point:**

Header:	#include <QtNetwork>
qmake:	QT += network
Inherits:	QAbstractSocket
Inherited By:	QSslSocket.

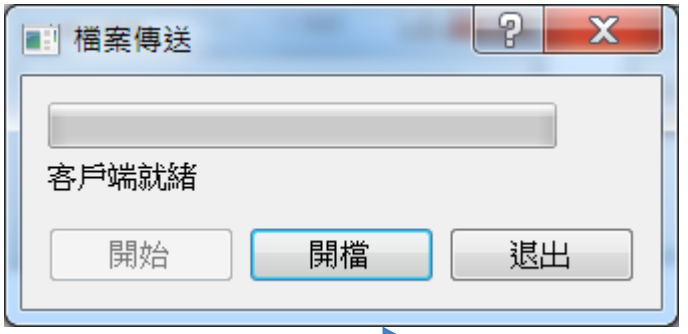
Qt- Networking

- **QTcpSocket Class : end-points of TCP connection**

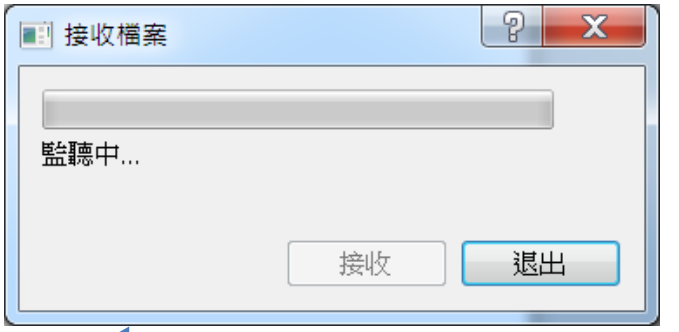
Qt- Networking

- **The application data format (user-defined)**

Client

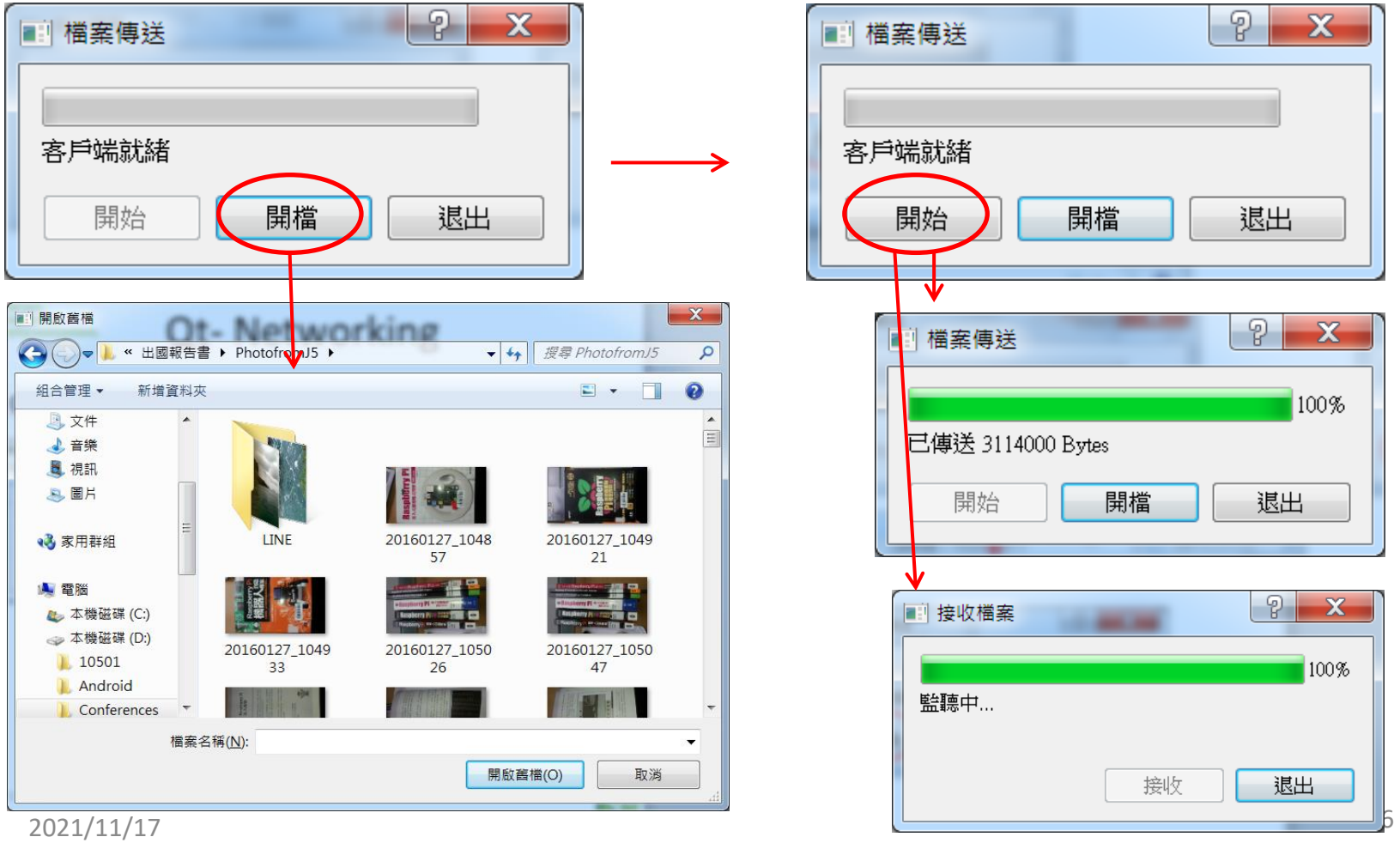


Server



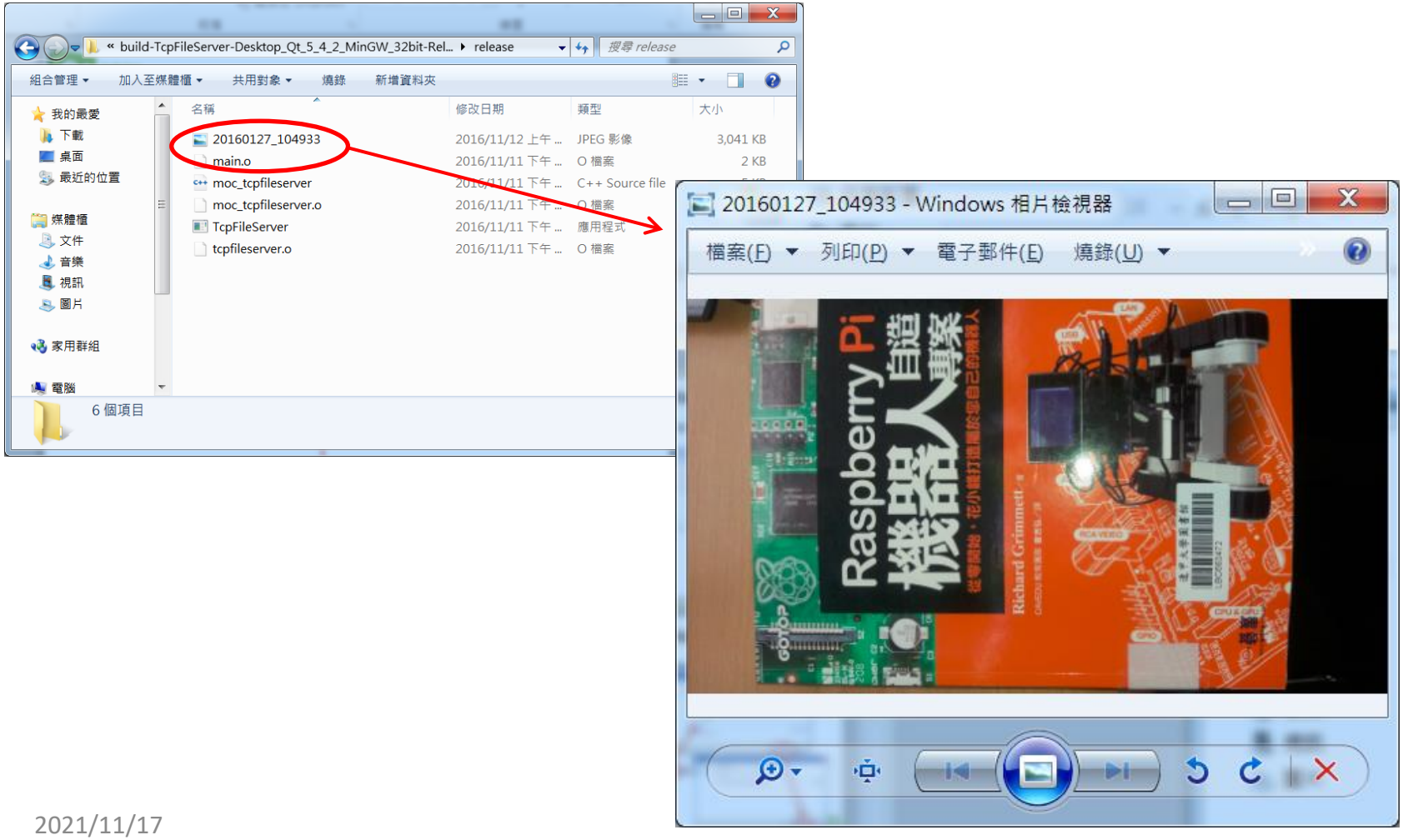
Qt- Networking

- **QTcpSocket Class : end-points of TCP connection**



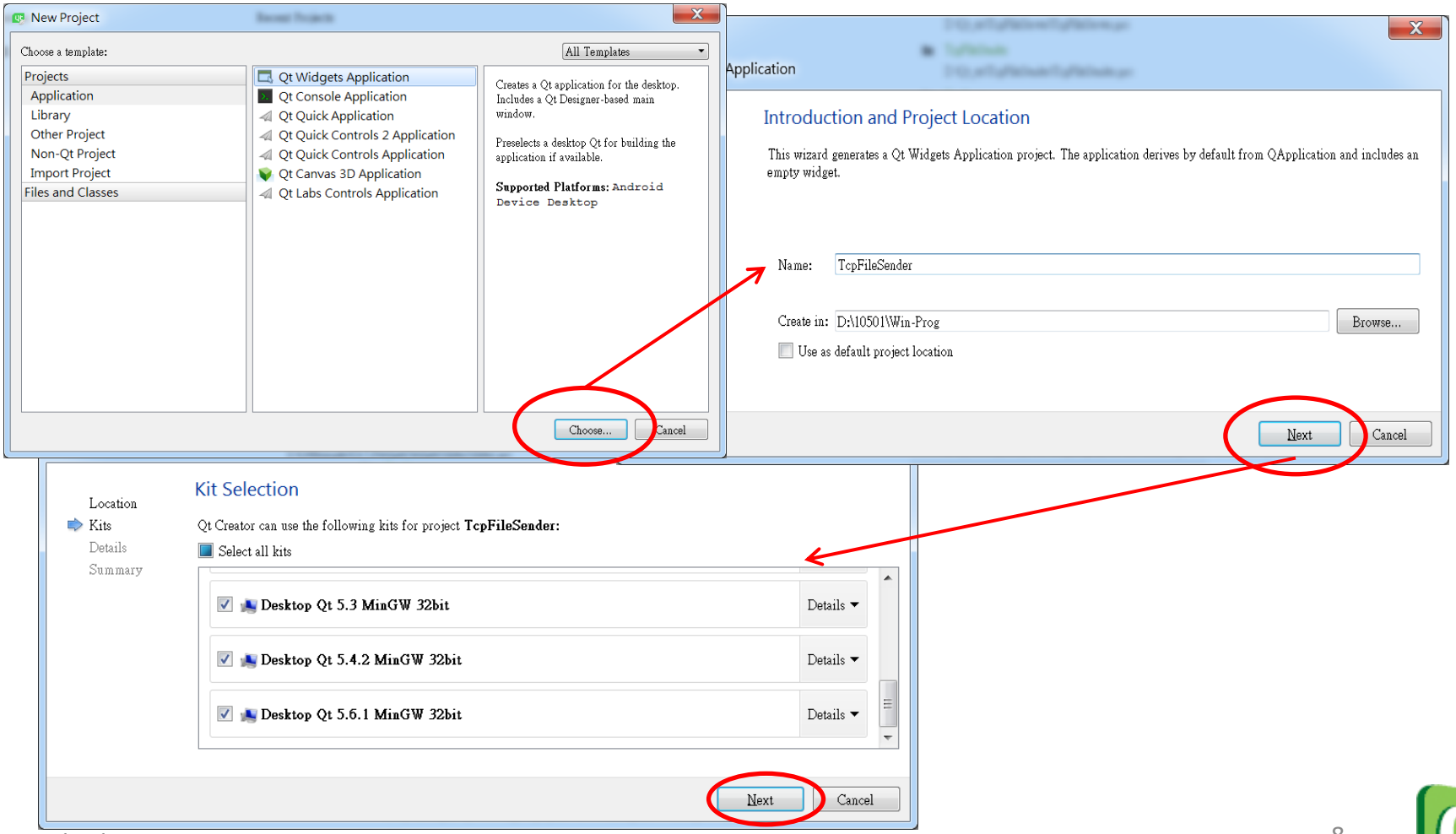
Qt- Networking

- **QTcpSocket Class : end-points of TCP connection**



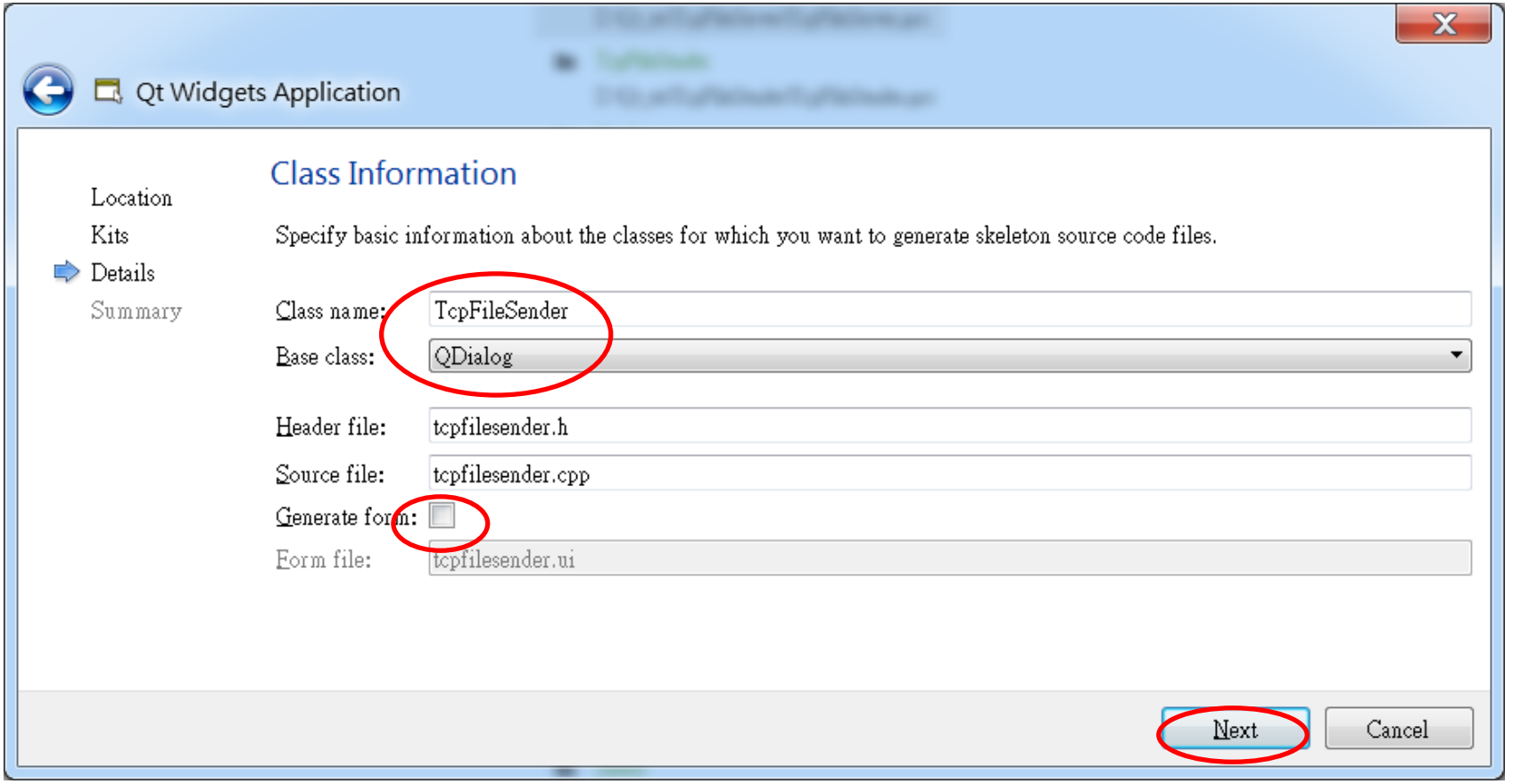
Qt- Networking

- **Create a file sender dialog-based application:**



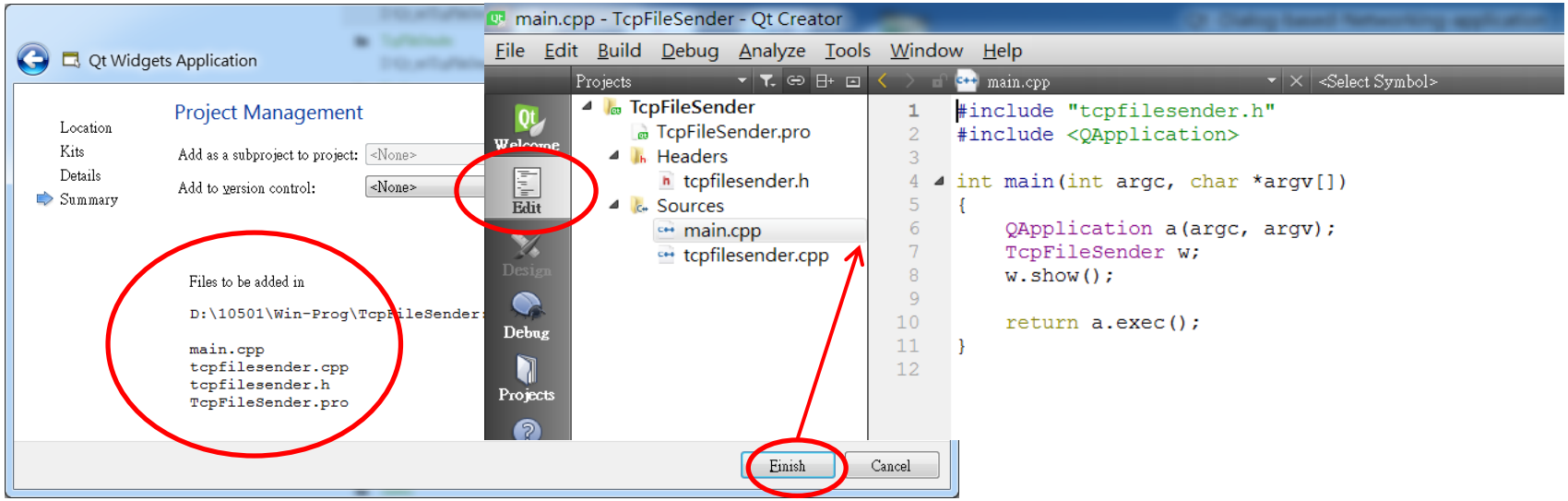
Qt- Networking

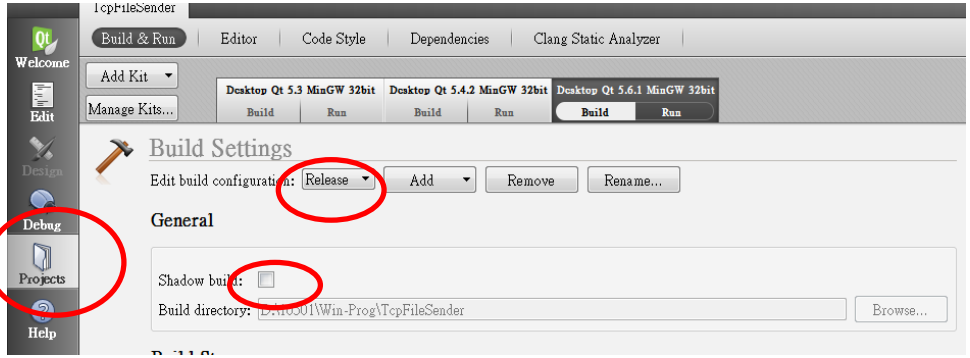
- **Create a file sender dialog-based application:**



Qt- Networking

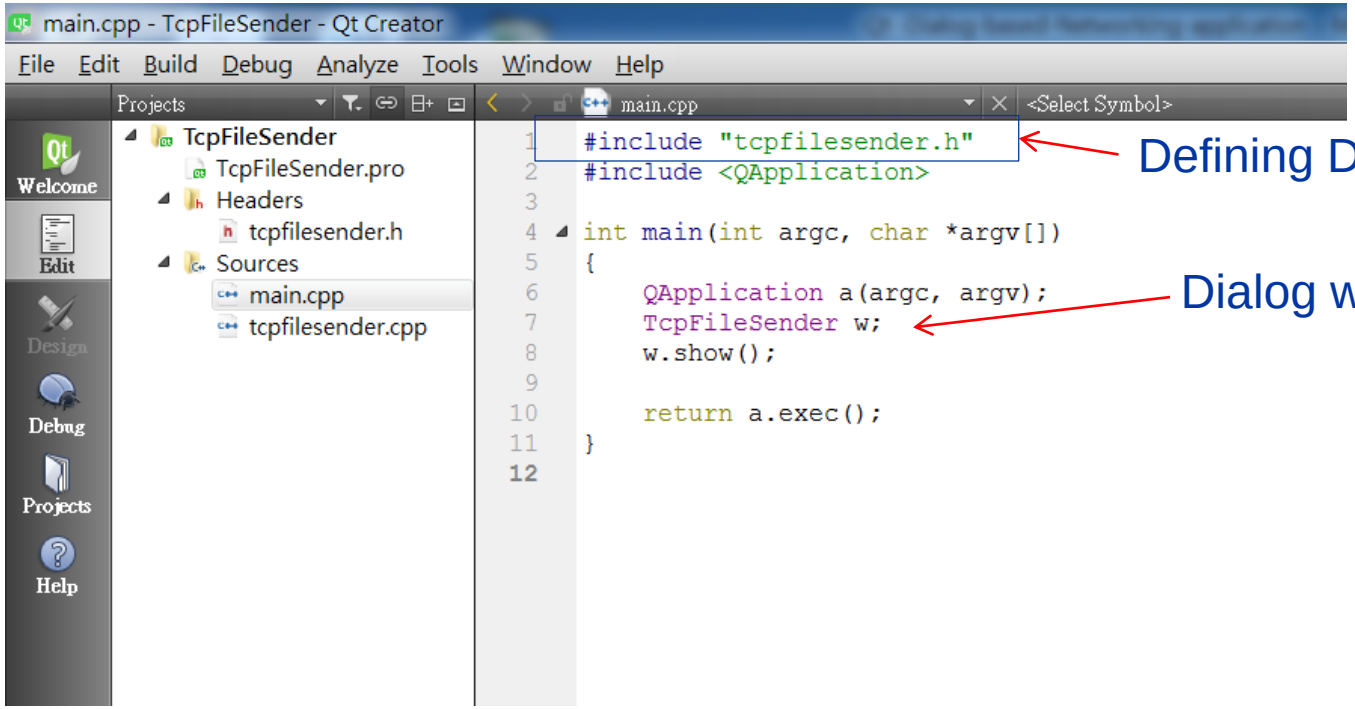
- **Create a file sender dialog-based application:**





Qt- Networking

- **Check out the main function:**

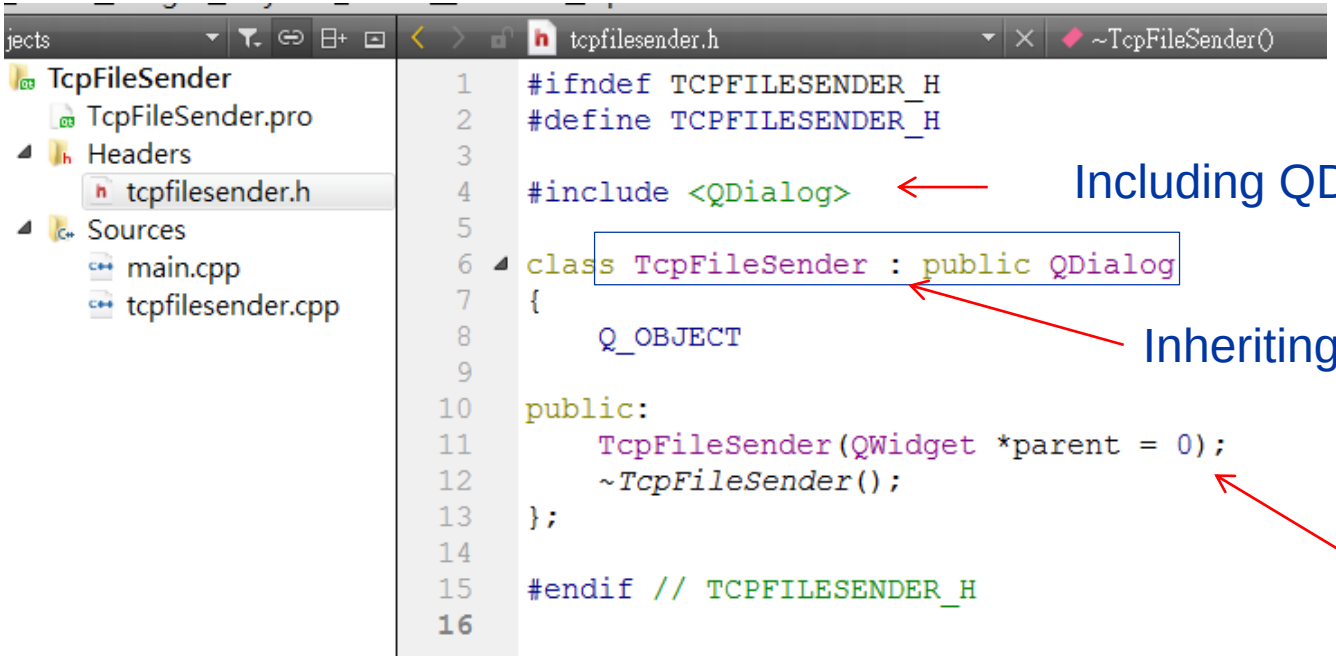


Defining Dialog window!

Dialog window object!

Qt-Networking

- **Check out the definition of TcpFileSender class:**



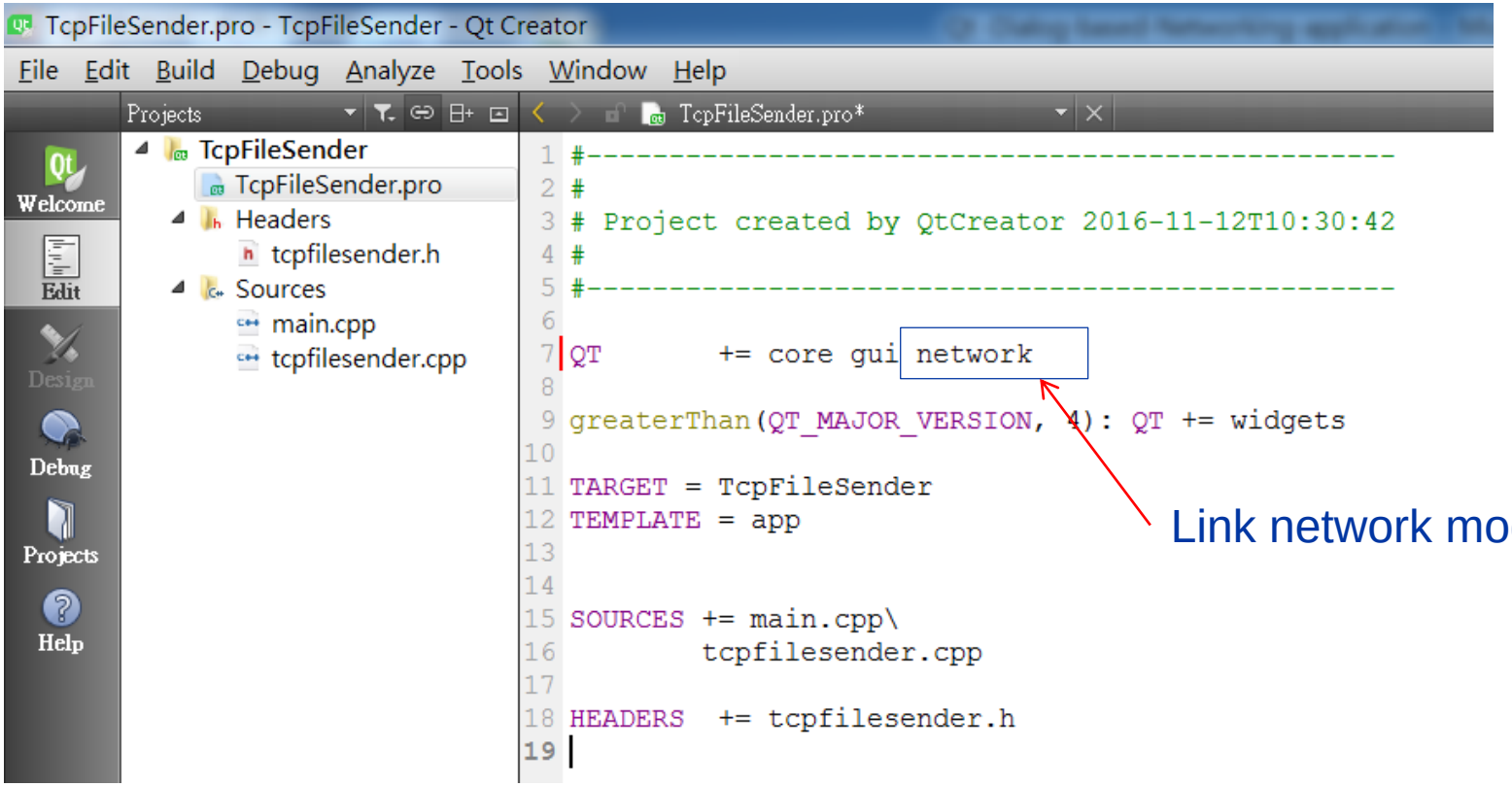
Including QDialog header file!

Inheriting from QDialog Class!

Constructor!

Networking

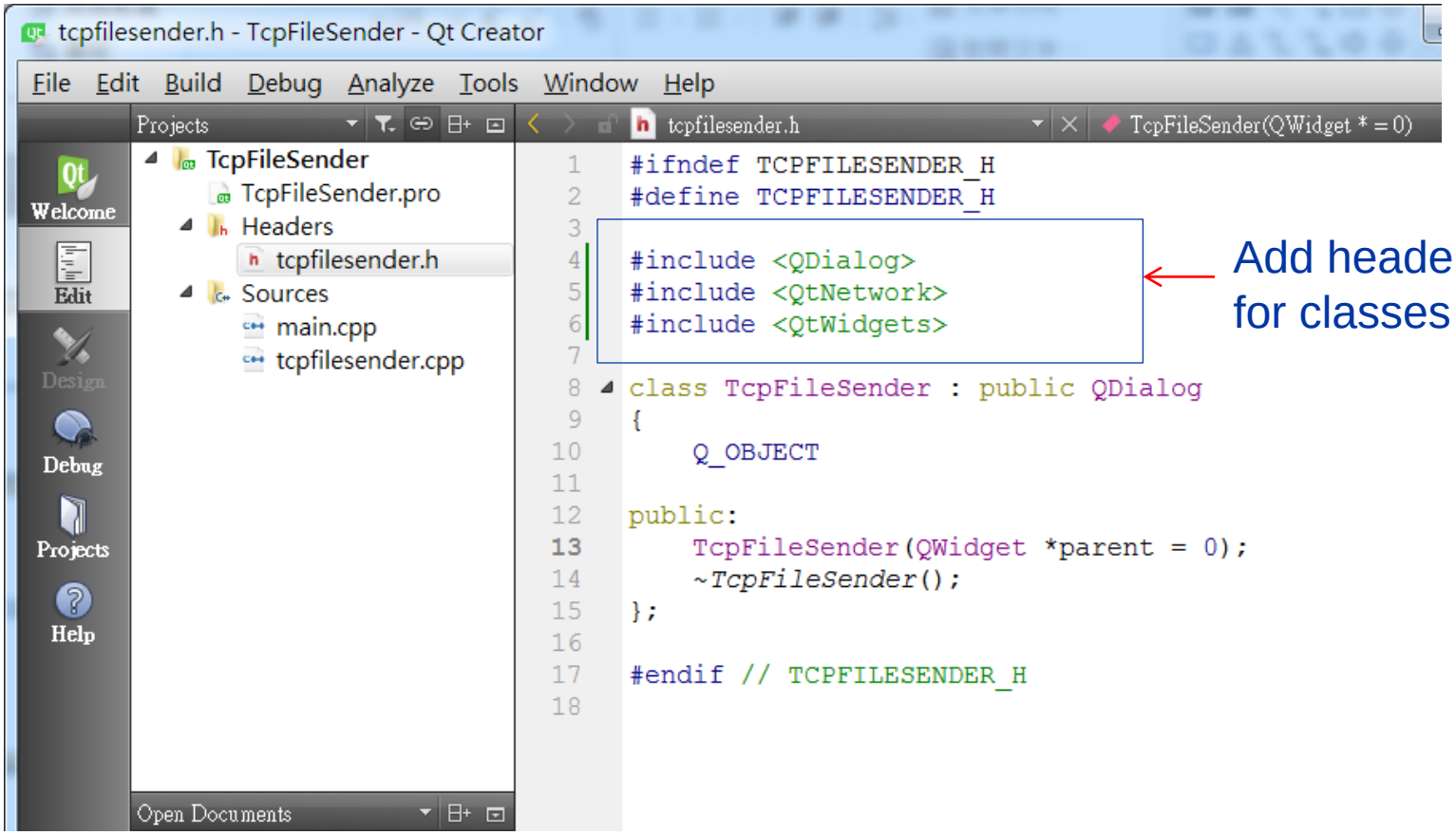
- **Check out the qmake project file:**



Link network module!

Qt- Networking

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



Add header files for classes used.

Qt- Networking

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

ender.h - TcpFileSender - Qt Creator

Build Debug Analyze Tools Window Help

Projects | tcpfilesender.h* | openFile(): void

- TcpFileSender
  - TcpFileSender.pro
  - Headers
    - tcpfilesender.h
  - Sources
    - main.cpp
    - tcpfilesender.cpp

D:\10501\Win-Prog\TcpFileSender\tcpfilesender.h

```
1  #ifndef TCPFILESENDER_H
2  #define TCPFILESENDER_H
3
4  #include <QDialog>
5  #include <QtNetwork>
6  #include <QtWidgets>
7
8  class TcpFileSender : public QDialog
9  {
10     Q_OBJECT
11
12 public:
13     TcpFileSender(QWidget *parent = 0);
14     ~TcpFileSender();
15 public slots:
16     void start();
17     void startTransfer();
18     void updateClientProgress(qint64 numBytes);
19     void openFile();
20 };
21
```

Forward declaration.

Add slot functions.

Qt- Networking

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

ender.h - TcpFileSender - Qt Creator

Build Debug Analyze Tools Window Help

Projects | tcpfilesender.h* | openFile(): void

- TcpFileSender
  - TcpFileSender.pro
  - Headers
    - tcpfilesender.h
  - Sources
    - main.cpp
    - tcpfilesender.cpp

D:\10501\Win-Prog\TcpFileSender\tcpfilesender.h

```
1  #ifndef TCPFILESENDER_H
2  #define TCPFILESENDER_H
3
4  #include <QDialog>
5  #include <QtNetwork>
6  #include <QtWidgets>
7
8  class TcpFileSender : public QDialog
9  {
10     Q_OBJECT
11
12 public:
13     TcpFileSender(QWidget *parent = 0);
14     ~TcpFileSender();
15 public slots:
16     void start();
17     void startTransfer();
18     void updateClientProgress(qint64 numBytes);
19     void openFile();
20 };
21
```

Forward declaration.

Add slot functions.

Qt- Networking

- **Add slot function implementation to tcpfilesensder.cpp:**

tcpfilesender.cpp - TcpFileSender - Qt Creator

File Edit Build Debug Analyze Tools Window Help

Projects

- TcpFileSender
  - TcpFileSender.pro
  - Headers
    - tcpfilesender.h
  - Sources
    - main.cpp
    - tcpfilesender.cpp

Open Documents

- main.cpp
- tcpfilesender.cpp*
- tcpfilesender.h*

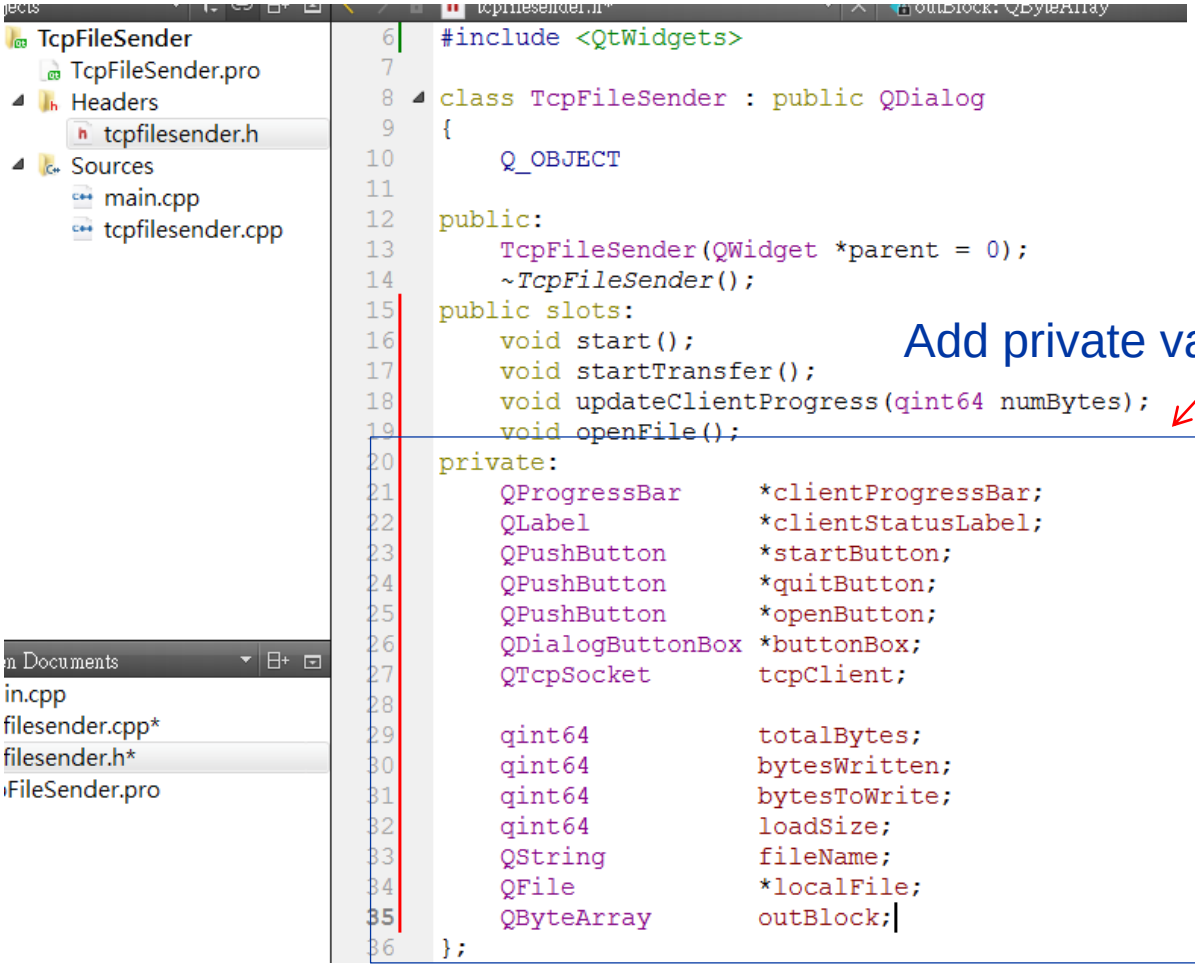
tcpfilesender.cpp* | TcpFileSender::openFile(): void

```
1  #include "tcpfilesender.h"
2
3  TcpFileSender::TcpFileSender(QWidget *parent)
4      : QDialog(parent)
5  {
6  }
7
8  TcpFileSender::~TcpFileSender()
9  {
10
11 }
12 void TcpFileSender::start()
13 {
14 }
15
16 void TcpFileSender::startTransfer()
17 {
18 }
19 void TcpFileSender::updateClientProgress(qint64 numBytes)
20 {
21 }
22 void TcpFileSender::openFile()
23 {
24 }
```

Add slot function implementations.

Qt- Networking

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

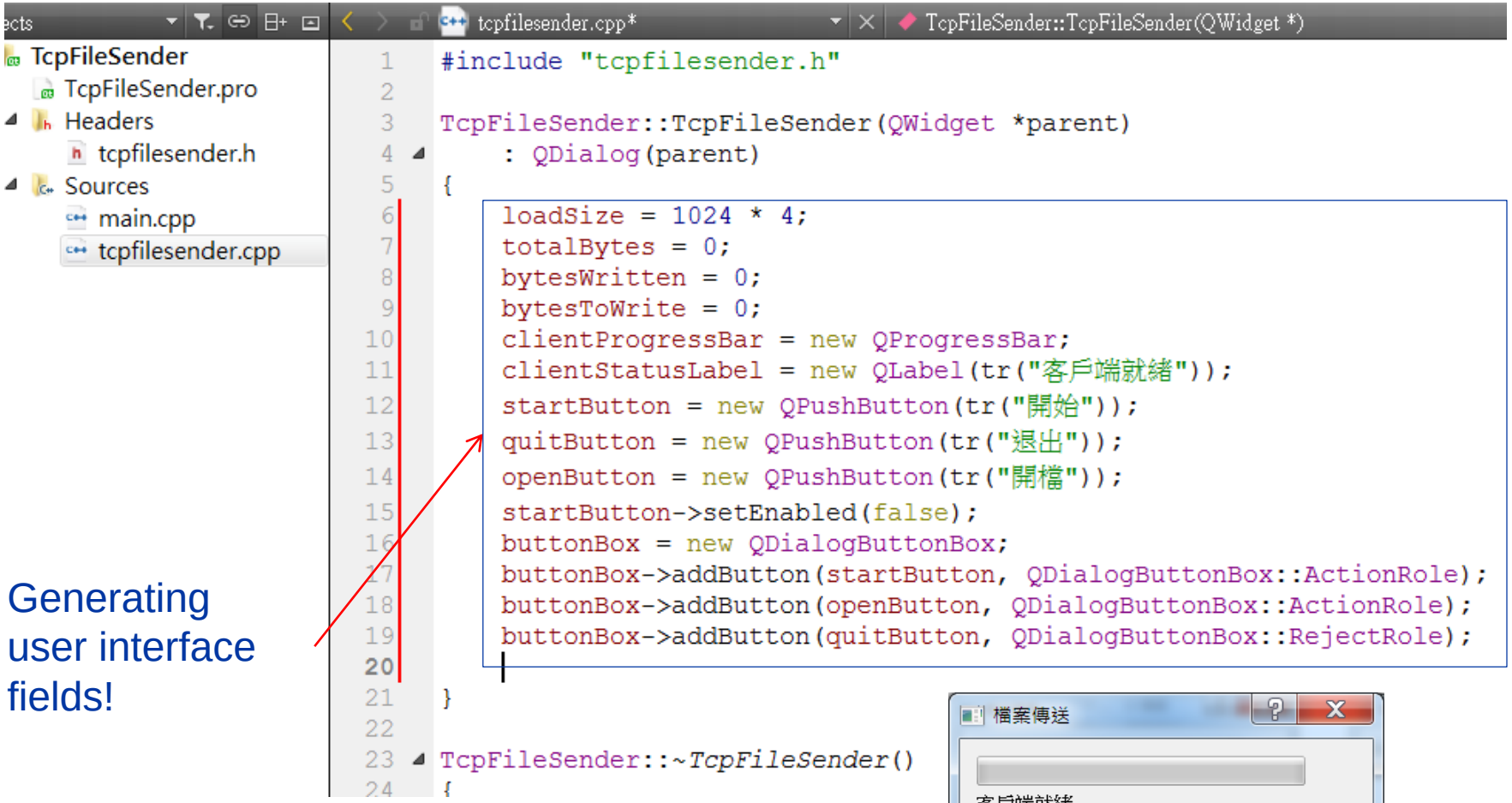



```
#include <QtWidgets>

class TcpFileSender : public QDialog
{
    Q_OBJECT

public:
    TcpFileSender(QWidget *parent = 0);
    ~TcpFileSender();
public slots:
    void start();
    void startTransfer();
    void updateClientProgress(qint64 numBytes);
    void openFile();
private:
    QProgressBar     *clientProgressBar;
    QLabel           *clientStatusLabel;
    QPushButton      *startButton;
    QPushButton      *quitButton;
    QPushButton      *openButton;
    QDialogButtonBox *buttonBox;
    QTcpSocket       tcpClient;

    qint64           totalBytes;
    qint64           bytesWritten;
    qint64           bytesToWrite;
    qint64           loadSize;
    QString          fileName;
    QFile            *localFile;
    QByteArray       outBlock;
};
```

Add private variables.

Qt- Networking

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




```
#include "tcpfilesender.h"

TcpFileSender::TcpFileSender(QWidget *parent)
    : QDialog(parent)
{
    loadSize = 1024 * 4;
    totalBytes = 0;
    bytesWritten = 0;
    bytesToWrite = 0;
    clientProgressBar = new QProgressBar;
    clientStatusLabel = new QLabel(tr("客戶端就緒"));
    startButton = new QPushButton(tr("開始"));
    quitButton = new QPushButton(tr("退出"));
    openButton = new QPushButton(tr("開檔"));
    startButton->setEnabled(false);
    buttonBox = new QDialogButtonBox;
    buttonBox->addButton(startButton, QDialogButtonBox::ActionRole);
    buttonBox->addButton(openButton, QDialogButtonBox::ActionRole);
    buttonBox->addButton(quitButton, QDialogButtonBox::RejectRole);

}

TcpFileSender::~TcpFileSender()
{
```

Generating user interface fields!

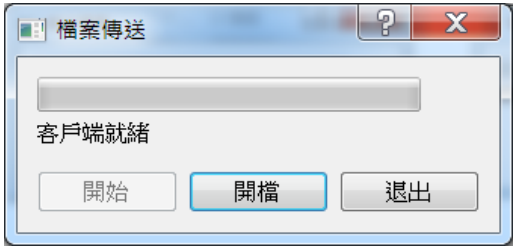
Qt- Networking

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

TcpFileSender
- TcpFileSender.pro
- Headers
  - tcpfilesender.h
- Sources
  - main.cpp
  - tcpfilesender.cpp

```cpp
#include "tcpfilesender.h"

TcpFileSender::TcpFileSender(QWidget *parent)
    : QDialog(parent)
{
    loadSize = 1024 * 4;
    totalBytes = 0;
    bytesWritten = 0;
    bytesToWrite = 0;
    clientProgressBar = new QProgressBar;
    clientStatusLabel = new QLabel(tr("客戶端就緒"));
    startButton = new QPushButton(tr("開始"));
    quitButton = new QPushButton(tr("退出"));
    openButton = new QPushButton(tr("開檔"));
    startButton->setEnabled(false);
    buttonBox = new QDialogButtonBox;
    buttonBox->addButton(startButton, QDialogButtonBox::ActionRole);
    buttonBox->addButton(openButton, QDialogButtonBox::ActionRole);
    buttonBox->addButton(quitButton, QDialogButtonBox::RejectRole);

    QVBoxLayout *mainLayout = new QVBoxLayout;
    mainLayout->addWidget(clientProgressBar);
    mainLayout->addWidget(clientStatusLabel);
    mainLayout->addStretch(1);
    mainLayout->addSpacing(10);
    mainLayout->addWidget(buttonBox);
    setLayout(mainLayout);
    setWindowTitle(tr("檔案傳送"));

}
```

Establish layout managers!



Qt- Networking

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

tcpfilesender.cpp* | TcpFileSender::TcpFileSender(QWidget *) | Line: 30, Col: 66

```
eSender
FileSender.pro
aders
tcpfilesender.h
urces
main.cpp
tcpfilesender.cpp
```

```cpp
    buttonBox->addButton(quitButton, QDialogButtonBox::RejectRole);

    QVBoxLayout *mainLayout = new QVBoxLayout;
    mainLayout->addWidget(clientProgressBar);
    mainLayout->addWidget(clientStatusLabel);
    mainLayout->addStretch(1);
    mainLayout->addSpacing(10);
    mainLayout->addWidget(buttonBox);
    setLayout(mainLayout);
    setWindowTitle(tr("檔案傳送"));
    connect(openButton,SIGNAL(clicked()), this, SLOT(openFile()));
    connect(startButton, SIGNAL(clicked()), this, SLOT(start()));
    connect(&tcpClient, SIGNAL(connected()), this, SLOT(startTransfer()));
    connect(&tcpClient, SIGNAL(bytesWritten(qint64)), this, SLOT(updateClientProgress(qint64)));
    connect(quitButton, SIGNAL(clicked()), this, SLOT(close()));

}

TcpFileSender::~TcpFileSender()
{

}
void TcpFileSender::start()
{
}
```

Connect signals to slot functions!

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 openFile() :**

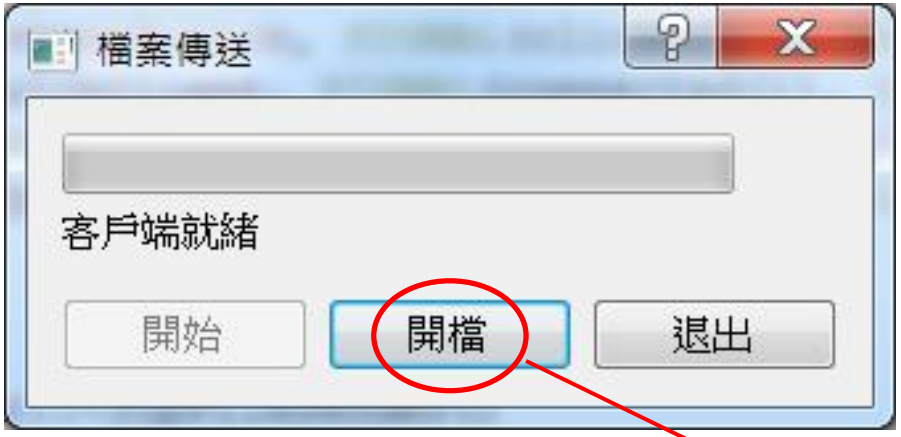
TcpFileSender
- TcpFileSender.pro
- Headers
  - tcpfilesender.h
- Sources
  - main.cpp
  - tcpfilesender.cpp

```
    mainLayout->addWidget(clientProgressBar);
    mainLayout->addWidget(clientStatusLabel);
    mainLayout->addStretch(1);
    mainLayout->addSpacing(10);
    mainLayout->addWidget(buttonBox);
    setLayout(mainLayout);
    setWindowTitle(tr("檔案傳送"));
    connect(openButton,SIGNAL(clicked()), this, SLOT(openFile()));
    connect(startButton, SIGNAL(clicked()), this, SLOT(start()));
    connect(&tcpClient, SIGNAL(connected()), this, SLOT(startTransfer()));
    connect(&tcpClient, SIGNAL(bytesWritten(qint64)), this, SLOT(updateClientPr
    connect(quitButton, SIGNAL(clicked()), this, SLOT(close()));

}
void TcpFileSender::openFile()
{
    fileName = QFileDialog::getOpenFileName(this);
       if (!fileName.isEmpty()) startButton->setEnabled(true);
}
TcpFileSender::~TcpFileSender()
{

}
void TcpFileSender::start()
```

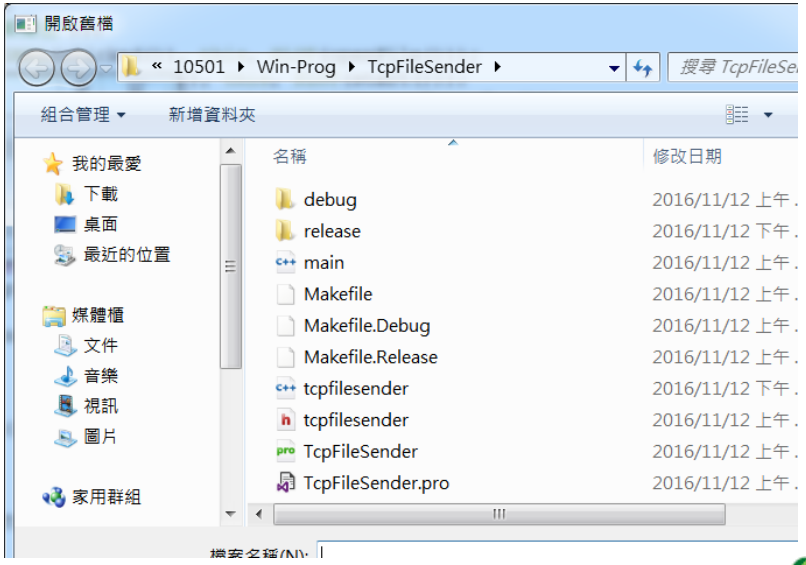
Get the filename the user selects from the QFileDialog.
Check if the filename is gotten successfully.
If yes, set the start button enabled!

Qt- Networking

- **Build and run the project:**



Try to interact with the “開檔” button!



Qt- Networking

- **Implement slot function start():**

```
TcpFileSender
  TcpFileSender.pro
  Headers
    tcpfilesender.h
  Sources
    main.cpp
    tcpfilesender.cpp
```

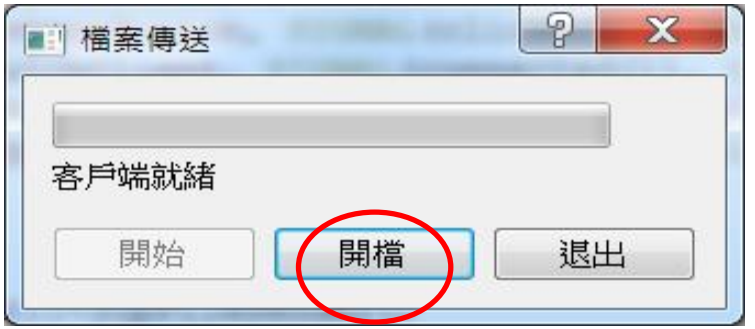
```
    mainLayout->addWidget(clientStatusLabel);
    mainLayout->addStretch(1);
    mainLayout->addSpacing(10);
    mainLayout->addWidget(buttonBox);
    setLayout(mainLayout);
    setWindowTitle(tr("檔案傳送"));
    connect(openButton,SIGNAL(clicked()), this, SLOT(openFile()));
    connect(startButton, SIGNAL(clicked()), this, SLOT(start()));
    connect(&tcpClient, SIGNAL(connected()), this, SLOT(startTransfer()));
    connect(&tcpClient, SIGNAL(bytesWritten(qint64)), this, SLOT(updateClientProgres
    connect(quitButton, SIGNAL(clicked()), this, SLOT(close()));

}
void TcpFileSender::openFile()
{
    fileName = QFileDialog::getOpenFileName(this);
    if (!fileName.isEmpty()) startButton->setEnabled(true);
}
void TcpFileSender::start()
{
    startButton->setEnabled(false);
    bytesWritten = 0;
    clientStatusLabel->setText(tr("連接中..."));
    tcpClient.connectToHost(QHostAddress::LocalHost, 16689);
}
```

```
Documents
.cpp
esender.cpp*
esender.h
ileSender.pro
```

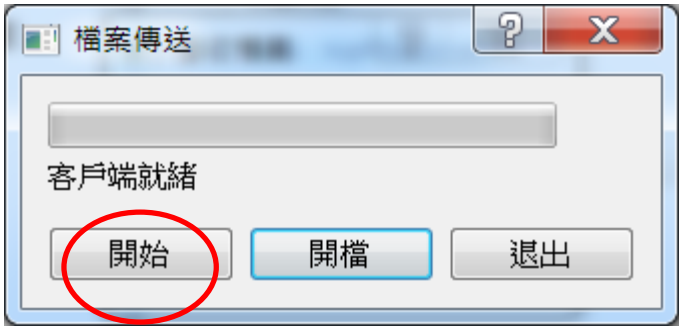
Use connectToHost() to initiate a tcp connection to a host with ip (127.0.0.1) running a server listen on port (16689).

Qt- Networking

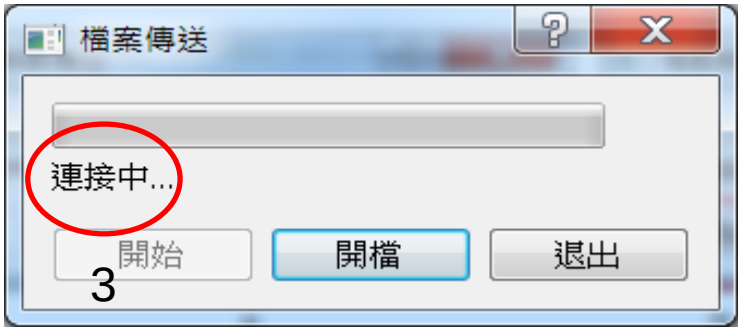
- **Build and run the project:**



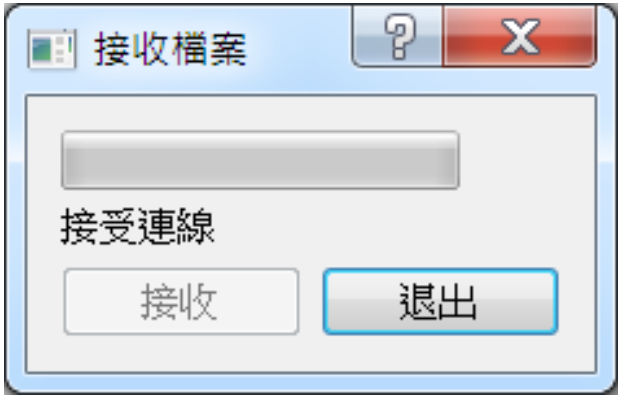
1



2



3



Server has accepted the connection.

Qt- Networking

- **Implement slot function startTransfer():**

er.cpp - TcpFileSender - Qt Creator

uild Debug Analyze Tools Window Help

ects | tcpfilesender.cpp* | TcpFileSender::startTransfer(): void

TcpFileSender
- TcpFileSender.pro
- Headers
  - tcpfilesender.h
- Sources
  - main.cpp
  - tcpfilesender.cpp

```
47  }
48  void TcpFileSender::startTransfer()
49  {
50      localFile = new QFile(fileName);
51      if (!localFile->open(QFile::ReadOnly))
52       {
53          QMessageBox::warning(this,tr("應用程式"),
54                               tr("無法讀取 %1:\n%2.").arg(fileName)
55                               .arg(localFile->errorString()));
56          return;
57       }
58
59
60
61
```

Step 1: Open the file the user has selected.

Qt- Networking

- **Implement slot function startTransfer():**

eSender.pro
rs
filesender.h
es
in.cpp
filesender.cpp

```
48  void TcpFileSender::startTransfer()
49  {
50      localFile = new QFile(fileName);
51      if (!localFile->open(QFile::ReadOnly))
52       {
53          QMessageBox::warning(this,tr("應用程式"),
54                                  tr("無法讀取 %1:\n%2.").arg(fileName)
55                                  .arg(localFile->errorString()));
56          return;
57       }
58
59      totalBytes = localFile->size();
60      QDataStream sendOut(&outBlock, QIODevice::WriteOnly);
61      sendOut.setVersion(QDataStream::Qt_4_6);
62      QString currentFile = fileName.right(fileName.size() -
63                                          fileName.lastIndexOf("/")-1);
64      sendOut <<qint64(0)<<qint64(0)<<currentFile;
65      totalBytes += outBlock.size();
66
```

Step 2: Fill the filename field in the header.

Qt- Networking

- **Implement slot function startTransfer():**

Headers
- tcpfilesender.h

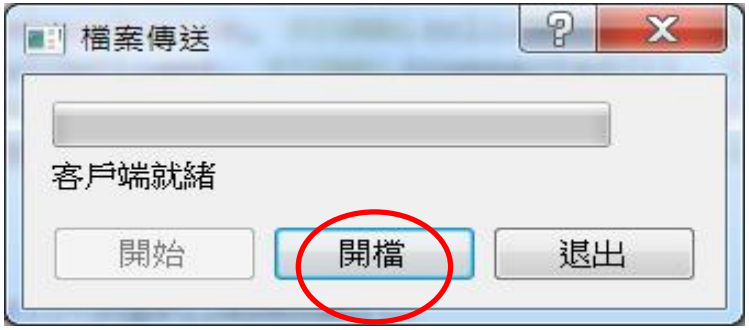
Sources
- main.cpp
- tcpfilesender.cpp

```
57          }
58
59      totalBytes = localFile->size();
60      QDataStream sendOut(&outBlock, QIODevice::WriteOnly);
61      sendOut.setVersion(QDataStream::Qt_4_6);
62      QString currentFile = fileName.right(fileName.size() -
63                                           fileName.lastIndexOf("/")-1);
64      sendOut <<qint64(0)<<qint64(0)<<currentFile;
65      totalBytes += outBlock.size();
66
67      sendOut.device()->seek(0);
68      sendOut<<totalBytes<<qint64((outBlock.size()-sizeof(qint64)*2));
69      bytesToWrite = totalBytes - tcpClient.write(outBlock);
70      clientStatusLabel->setText(tr("已連接"));
71      qDebug() << currentFile <<totalBytes;
72      outBlock.resize(0);
73  }
```

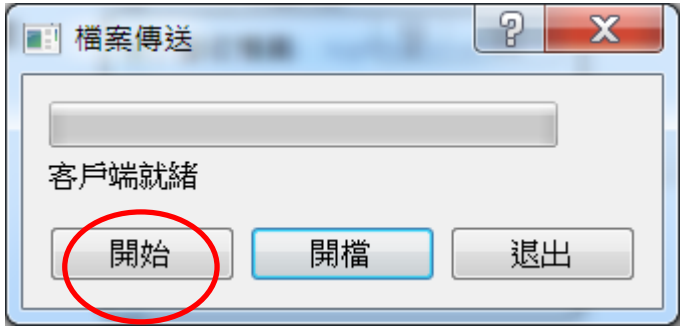
Step 3: Fill the total size and filename size fields in the header.
And send the header first using write() method.

Qt- Networking

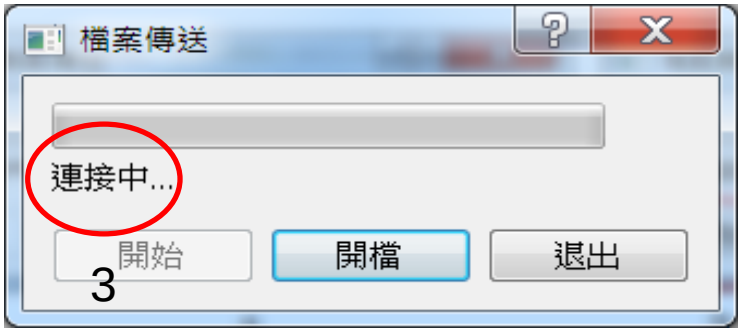
- **Build and run the project:**



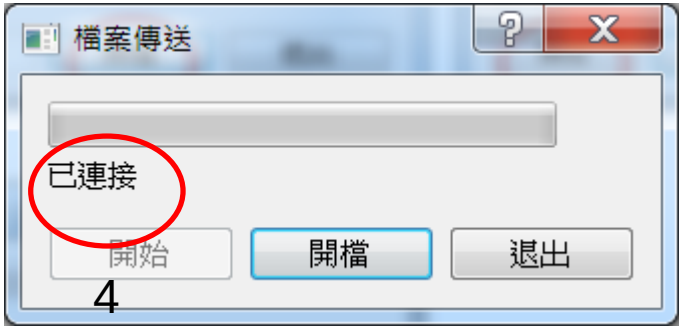
1



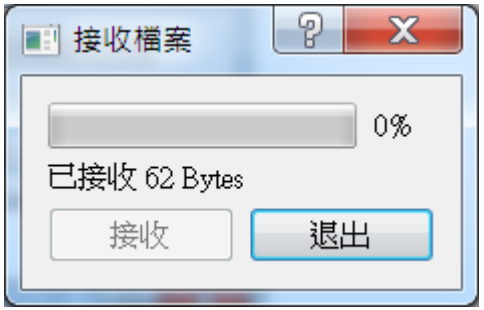
2



3



4



Server has received the header part.

Qt- Networking

- **Implement slot function updateClientProgress():**

tcpfilesender.h
ources
main.cpp
tcpfilesender.cpp

```
    qDebug() << currentFile <<totalBytes;
    outBlock.resize(0);
}
void TcpFileSender::updateClientProgress(qint64 numBytes)
{
    bytesWritten += (int) numBytes;
    if(bytesToWrite > 0)
    {
        outBlock = localFile->read(qMin(bytesToWrite, loadSize));
        bytesToWrite -= (int) tcpClient.write(outBlock);
        outBlock.resize(0);
    }else
    {
        localFile->close();
    }
```

Step 1: Read payload (at most 4 KB) from the file.
Send the payload (in outBlock) using write().

Qt- Networking

- **Implement slot function startTransfer():**

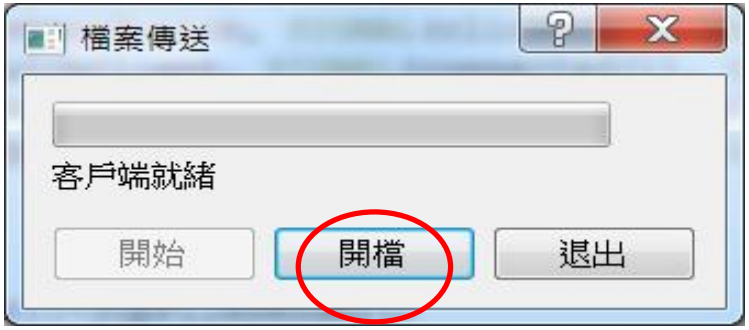
TcpFileSender
- TcpFileSender.pro
- Headers
  - tcpfilesender.h
- Sources
  - main.cpp
  - tcpfilesender.cpp

```cpp
    outBlock.resize(0);
}
void TcpFileSender::updateClientProgress(qint64 numBytes)
{
    bytesWritten += (int) numBytes;
    if(bytesToWrite > 0)
    {
        outBlock = localFile->read(qMin(bytesToWrite, loadSize));
        bytesToWrite -= (int) tcpClient.write(outBlock);
        outBlock.resize(0);
    }else
    {
        localFile->close();
    }

    clientProgressBar->setMaximum(totalBytes);
    clientProgressBar->setValue(bytesWritten);
    clientStatusLabel->setText(tr("已傳送 %1 Bytes").arg(bytesWritten));
}
```

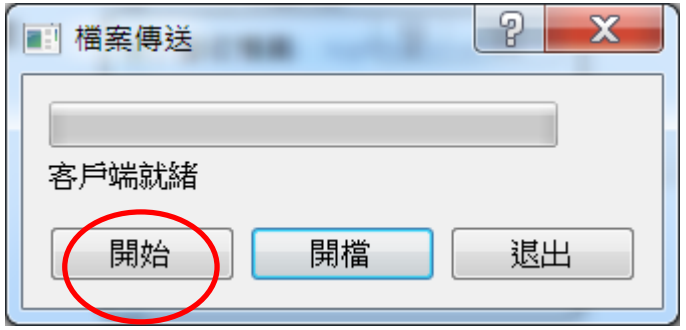
Step 2: Update the progress bar and the status label shown in the window.

Qt- Networking

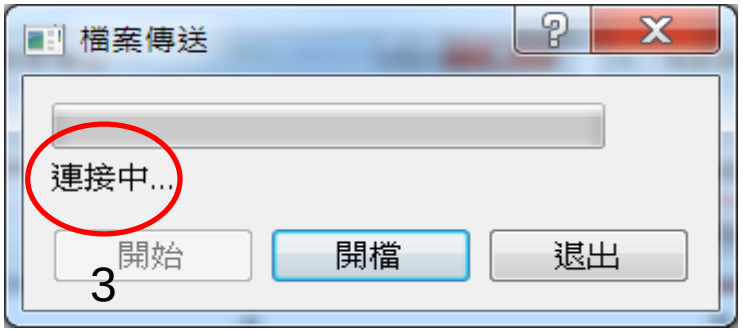
- **Build and run the project:**



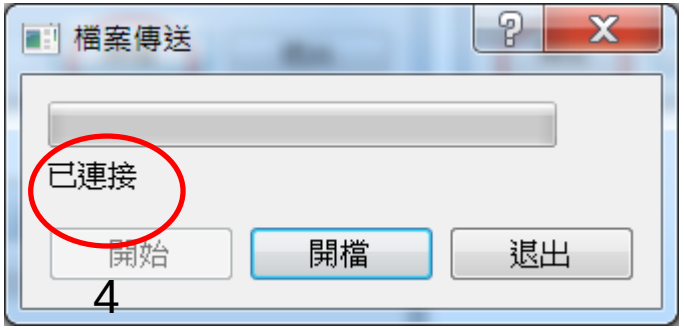
1



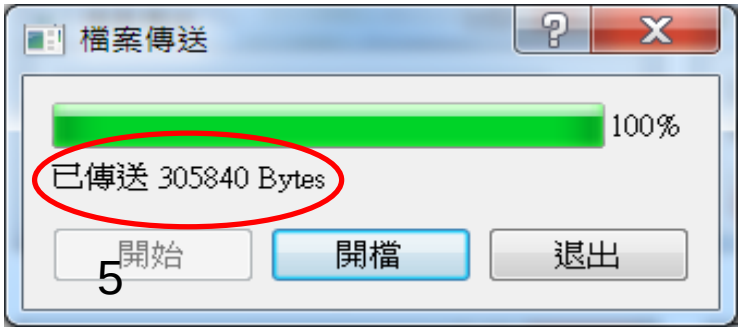
2



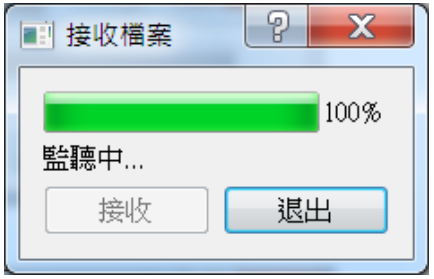
3



4



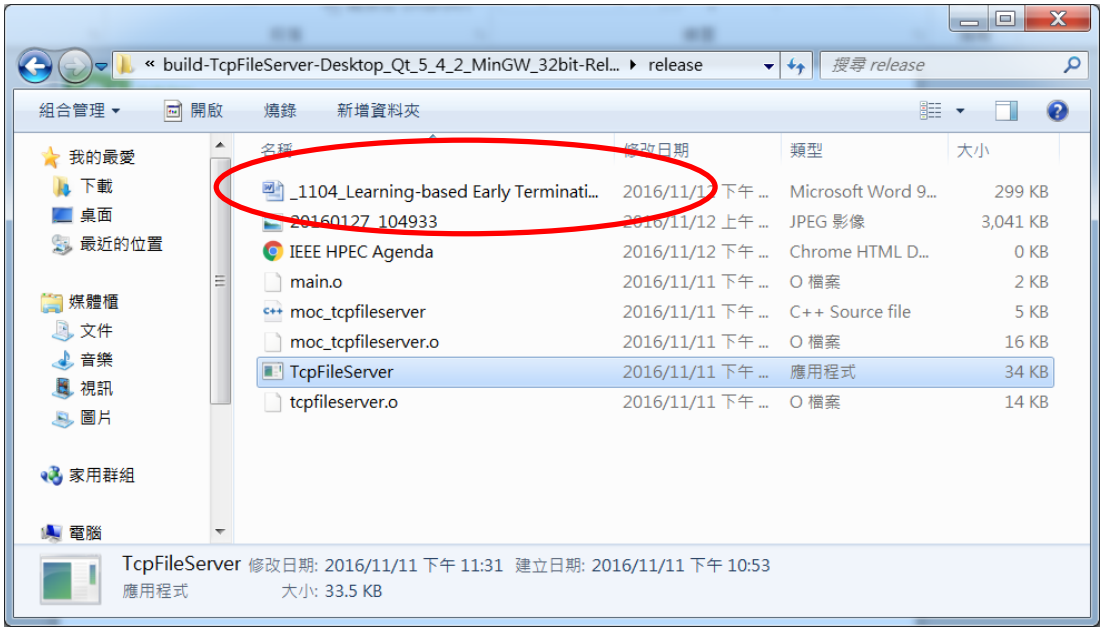
5



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

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