Building CyberDragon from source

What you need:

1. Qt Creator IDE with Qt 5.4.1.

For Windows users get it from:
http://download.qt.io/official_releases/online_installers/qt-opensource-windows-x86-online.exe

Remember to select MinGW from Tools section when installing (I only have build with MinGW so I don't have a clue if it works with Visual Studio).

For Linux users: Either use your Linux package manager or if none is provided by your Linux distro (horror!) then get it from:
https://download.qt.io/archive/qt/5.4/5.4.1/
'

2. Perl.

For Windows users get it from: http://strawberryperl.com/
For Linux users it is probably already installed. If not then do ”yum install perl”, ”apt-get perl” or whatever package manager/command your distro is using.

3. zlib, GeoIP and OpenSSL

For Windows users I have provided pre-built zlib, GeoIP and OpenSSL binaries (just take a look at ”include” and ”libs” folders) and they will be automatically used when building under Windows. Linux users must install the above mentioned dependencies with their package manager. Usually the ”yum install zlib-devel GeoIP-devel openssl-devel” or something similar should do the trick.

Next step: Setting up your build environment.

Before launching Qt Creator we need to install one addon module that CyberDragon needs, namely the qtftp add-on that provides FTP support.

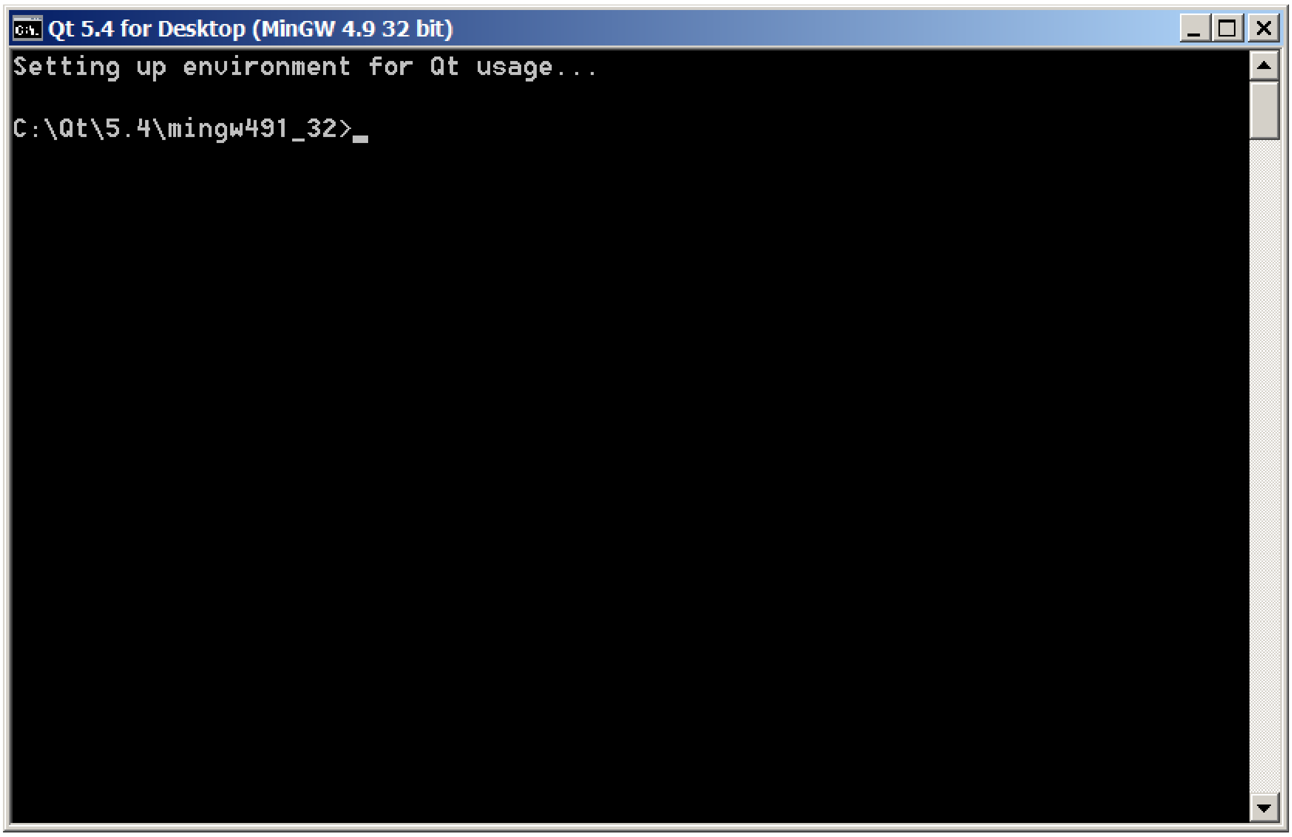
You have to ask Digia (or maybe Nokia) why they decided in their infinite wisdom to remove FTP support from the Qt network core in 5.x.....

Without qtftp you will get ”Project ERROR: Unknown module(s) in QT: ftp” and won't be able to compile CyberDragon.

I have provided qtftp zip file with the source package so you don't have to go hunting for it.

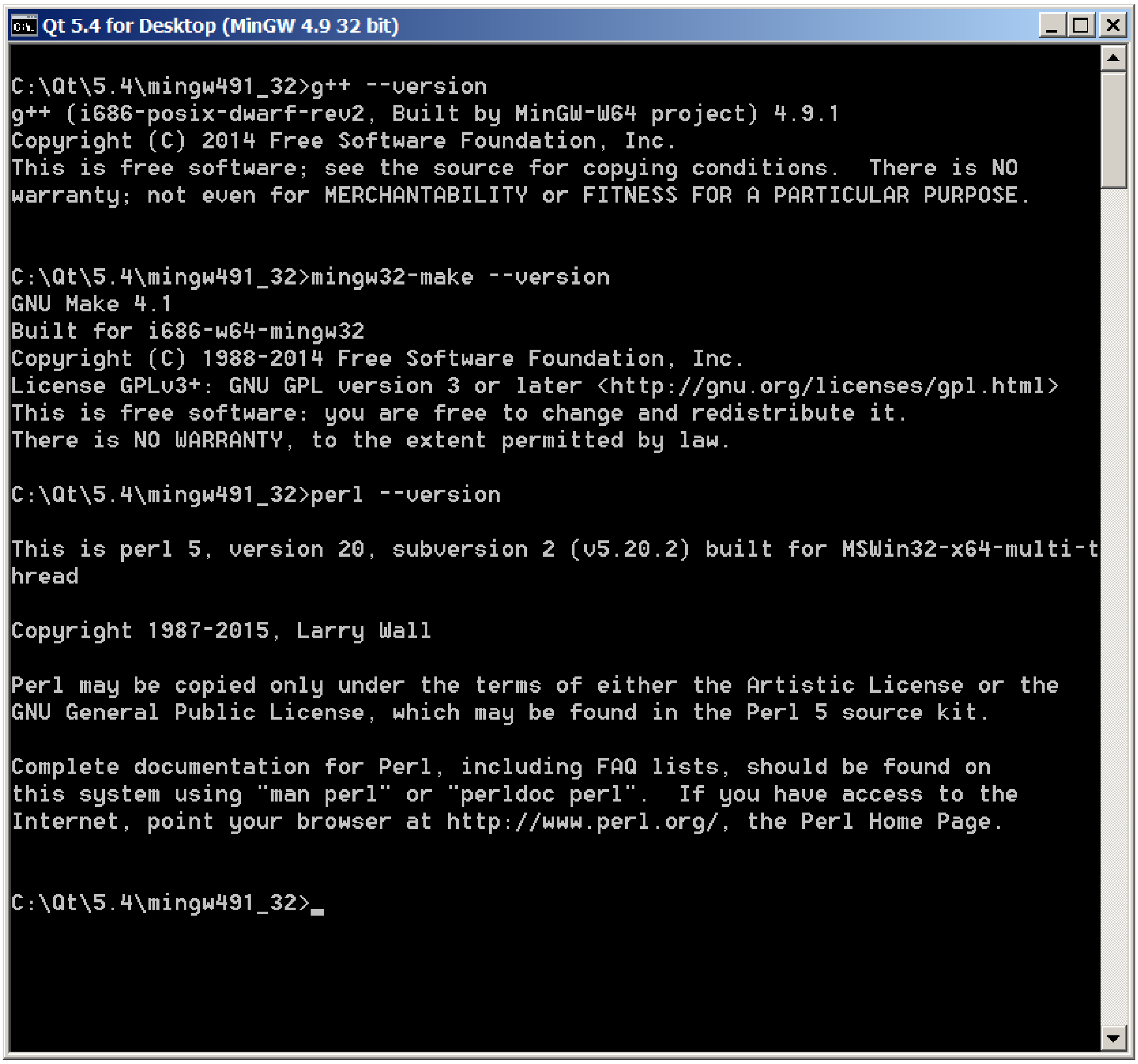
Now, the following steps are for Windows users. Linux users can skip to step 4.

Step 1. Open your Qt 5.4 command prompt (look under from Qt | 5.4 | MinGW (32-bit))
This is what you should see.



```
Setting up environment for Qt usage...

C:\Qt\5.4\mingw491_32>
```

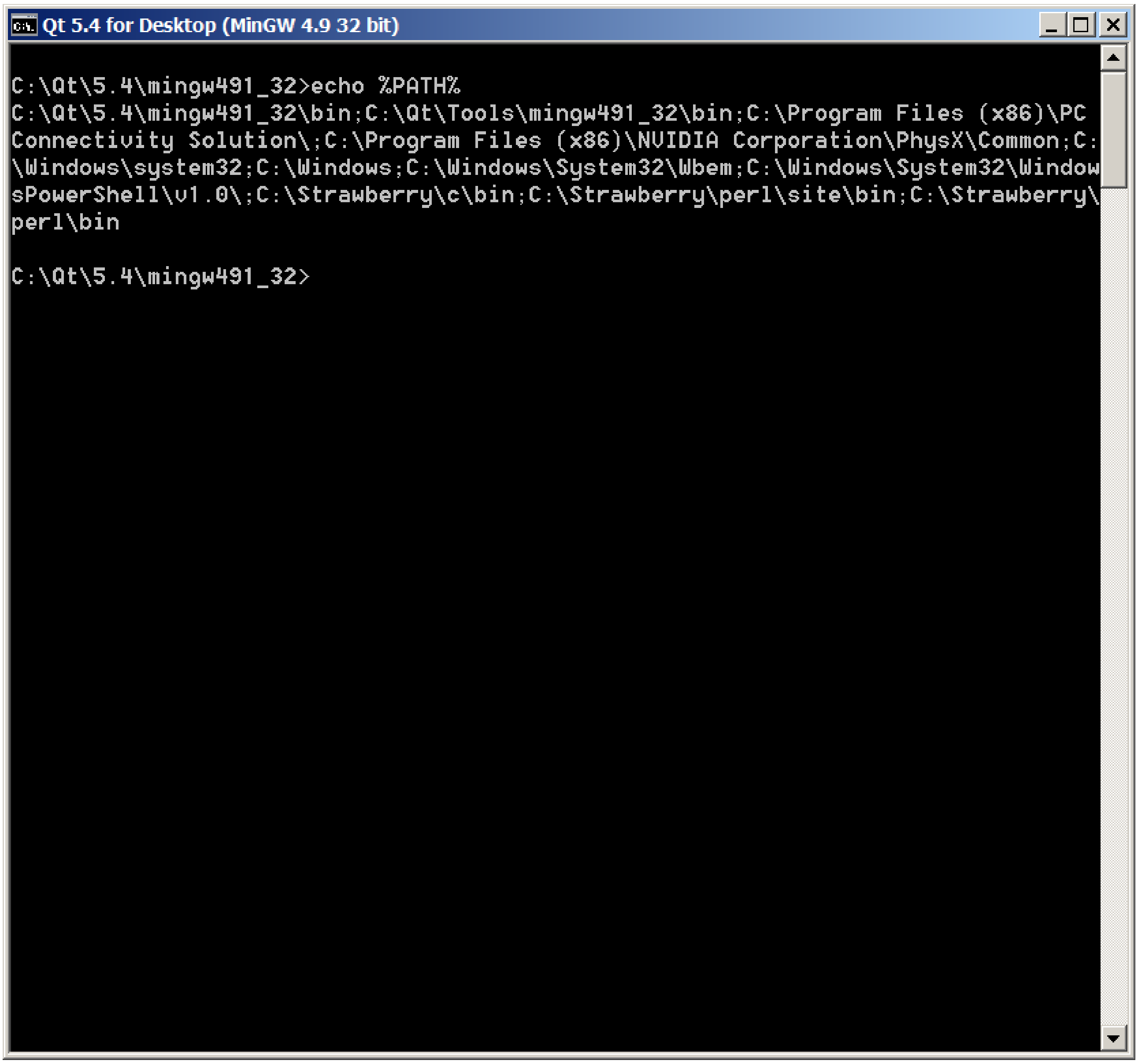
Step 2. Ensure that you have g++, mingw32-make and perl working.



```
C:\Qt\5.4\mingw491_32>g++ --version
g++ (i686-posix-dwarf-rev2, Built by MinGW-W64 project) 4.9.1
Copyright (C) 2014 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.


C:\Qt\5.4\mingw491_32>mingw32-make --version
GNU Make 4.1
Built for i686-w64-mingw32
Copyright (C) 1988-2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

C:\Qt\5.4\mingw491_32>perl --version

This is perl 5, version 20, subversion 2 (v5.20.2) built for MSWin32-x64-multi-t
hread

Copyright 1987-2015, Larry Wall

Perl may be copied only under the terms of either the Artistic License or the
GNU General Public License, which may be found in the Perl 5 source kit.

Complete documentation for Perl, including FAQ lists, should be found on
this system using "man perl" or "perldoc perl".  If you have access to the
Internet, point your browser at http://www.perl.org/, the Perl Home Page.


C:\Qt\5.4\mingw491_32>_
```

Step 3. Just in case check your path (this is only needed if you other development tools installed to your machine, like cygwin or older 32-bit MinGW). Type **echo %PATH%**

Make sure that C:\Qt\5.4\mingw491_32\bin, C:\Qt\Tools\mingw491_32\bin, C:\Strawberry\perl\site\bin and C:\Strawberry\perl\bin are listed before any other development tools you might have.

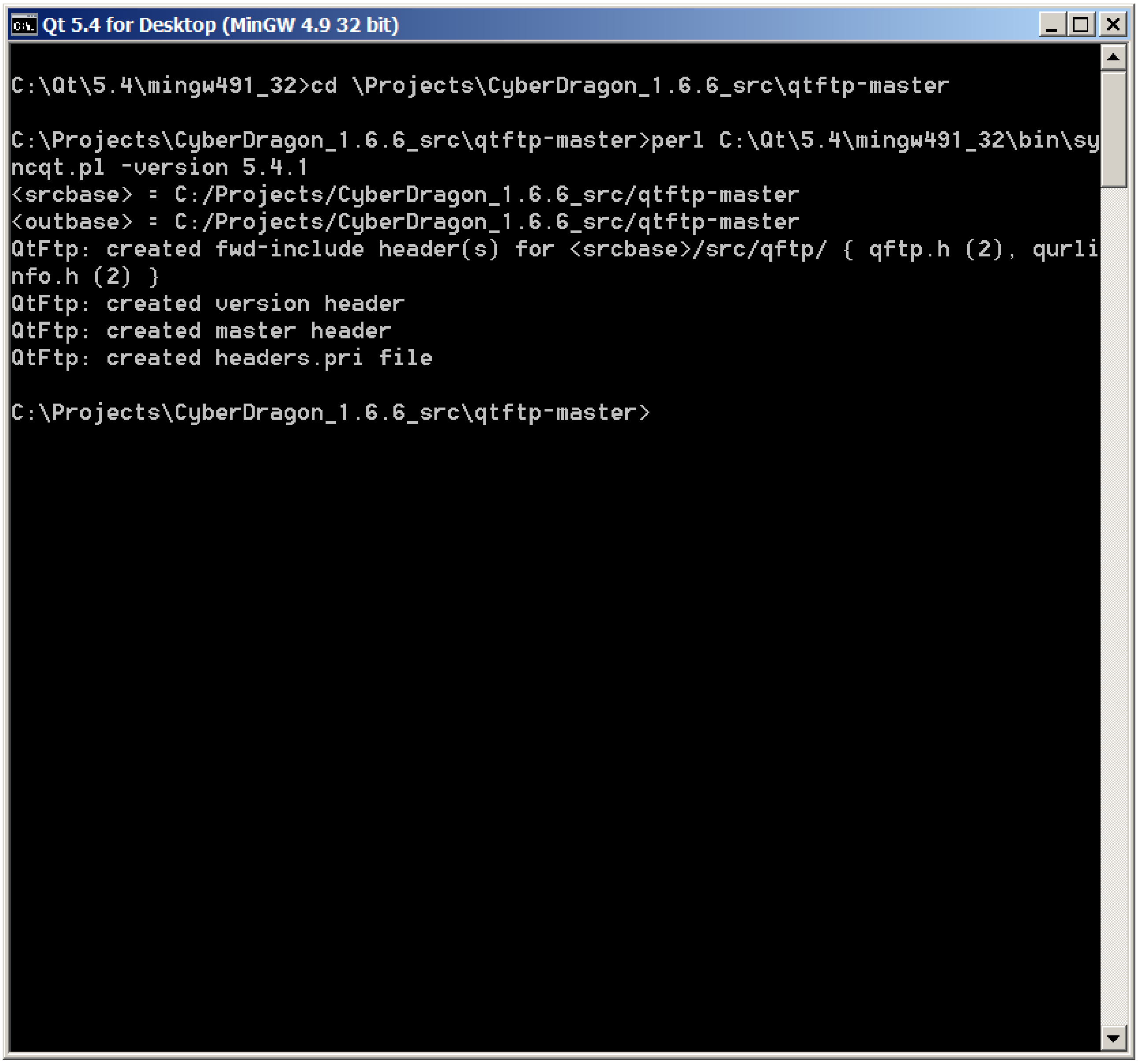


```
C:\Qt\5.4\mingw491_32>echo %PATH%
C:\Qt\5.4\mingw491_32\bin;C:\Qt\Tools\mingw491_32\bin;C:\Program Files (x86)\PC
Connectivity Solution\;C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common;C:
\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windows\System32\Window
sPowerShell\v1.0\;C:\Strawberry\c\bin;C:\Strawberry\perl\site\bin;C:\Strawberry\
perl\bin

C:\Qt\5.4\mingw491_32>
```

Step 4. Unzip the qtftp-master.zip that is included with the source. Windows has builtin support for zip files. Linux users should have some GUI tool for that or just use unzip from command line. (and if you don't have unzip installed do the ”yum install unzip” or ”apt-get unzip” dance again) Alternatively, there is also qt-qtftp.tar.gz for Linux users that you can extract with simple tar -xavf qt-qtftp.tar.gz command.

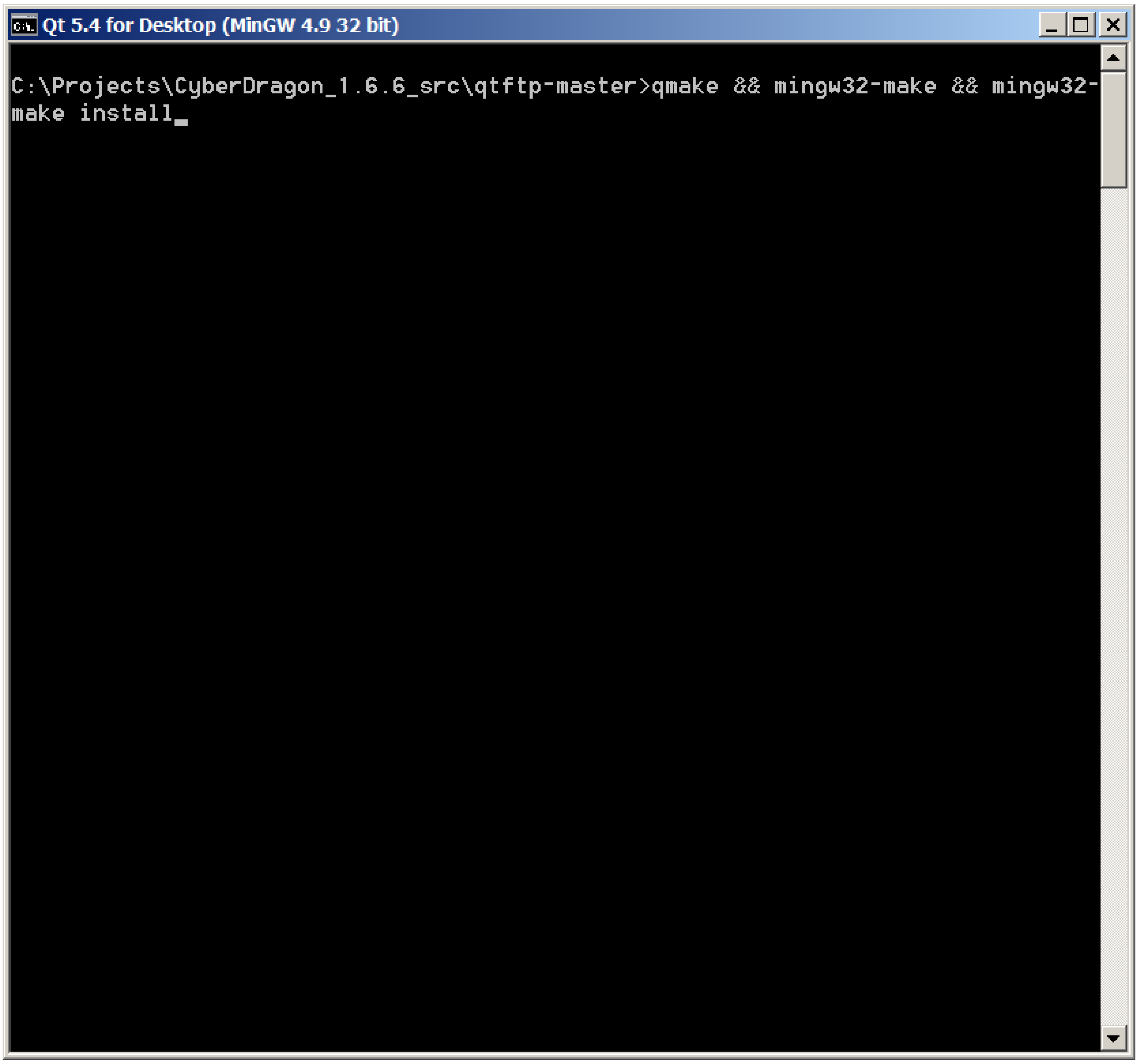
Then go to the directory where you unzipped qtftp and execute **perl C:\Qt\5.4\mingw491_32\bin\syncqt.pl -version 5.4.1** for window users.

Linux users could try perl /usr/bin/syncqt.pl -version 5.4.1.



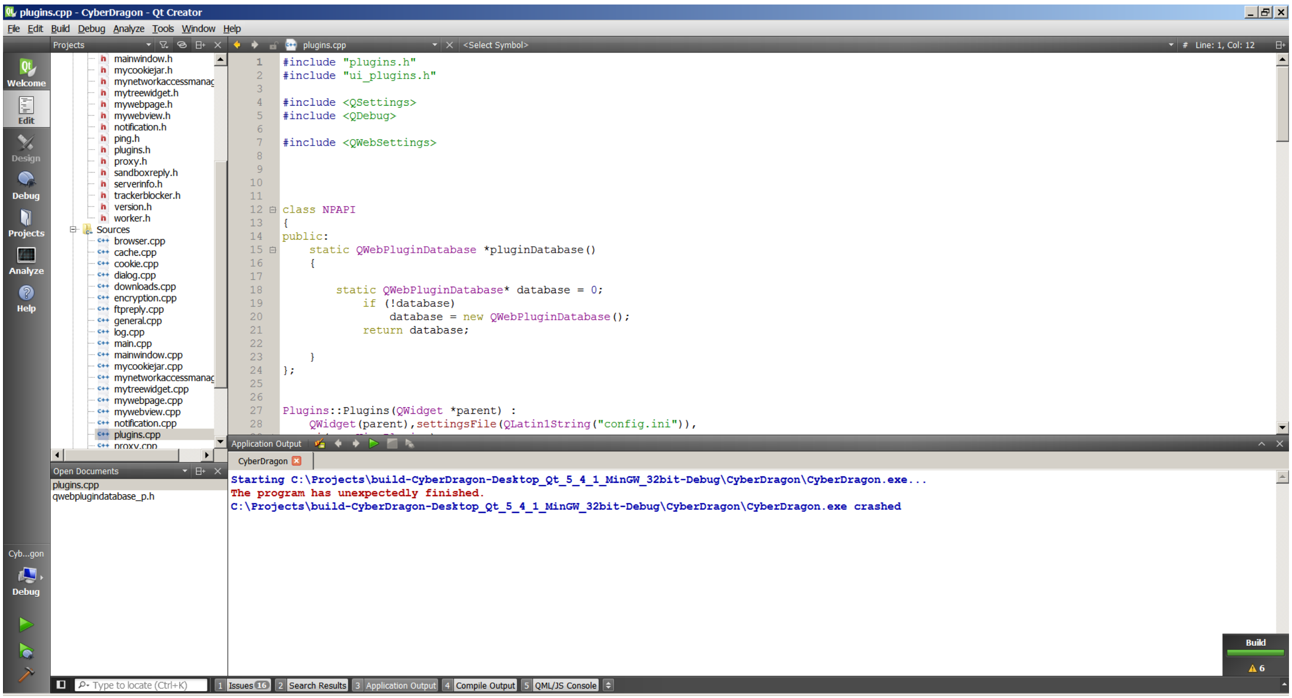
```
C:\Qt\5.4\mingw491_32>cd \Projects\CyberDragon_1.6.6_src\qtftp-master

C:\Projects\CyberDragon_1.6.6_src\qtftp-master>perl C:\Qt\5.4\mingw491_32\bin\sy
ncqt.pl -version 5.4.1
<srcbase> = C:/Projects/CyberDragon_1.6.6_src/qtftp-master
<outbase> = C:/Projects/CyberDragon_1.6.6_src/qtftp-master
QtFtp: created fwd-include header(s) for <srcbase>/src/qftp/ { qftp.h (2), qurli
nfo.h (2) }
QtFtp: created version header
QtFtp: created master header
QtFtp: created headers.pri file

C:\Projects\CyberDragon_1.6.6_src\qtftp-master>
```

Step 5. Compile and install.
For Windows users do: qmake && mingw32-make && mingw32-make install
For Linux users do: qmake (or sometimes qmake-qt5) && make && make install



```
C:\Projects\CyberDragon_1.6.6_src\qtftp-master>qmake && mingw32-make && mingw32-
make install_
```

You are now ready to launch Qt Creator. Open CyberDragon.pro file and it will automatically open it with Qt Creator.

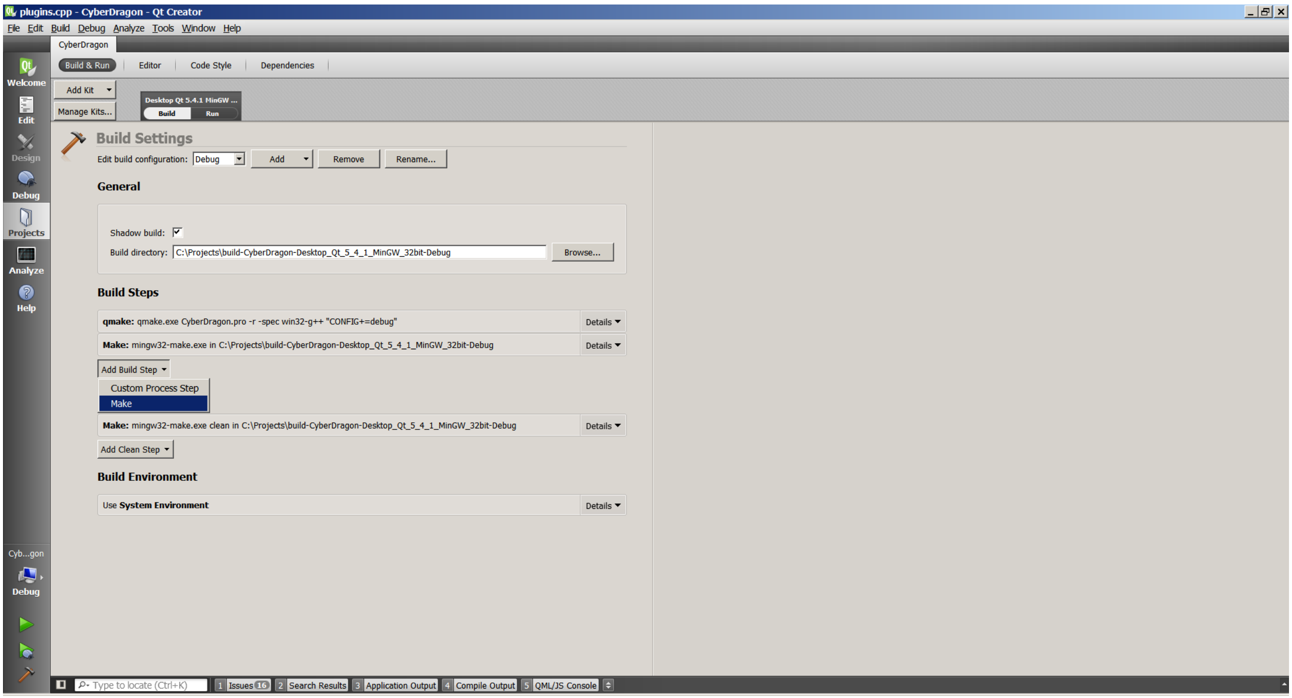


```
#include "plugins.h"
#include "ui_plugins.h"

#include <QSettings>
#include <QDebug>

#include <QWebSettings>




class NPAPI
{
public:
    static QWebPluginDatabase *pluginDatabase()
    {

        static QWebPluginDatabase* database = 0;
            if (!database)
                database = new QWebPluginDatabase();
            return database;

    }
};


Plugins::Plugins(QWidget *parent) :
    QWidget(parent),settingsFile(QLatin1String("config.ini")),
```

```
Starting C:\Projects\build-CyberDragon-Desktop_Qt_5_4_1_MinGW_32bit-Debug\CyberDragon\CyberDragon.exe...
The program has unexpectedly finished.
C:\Projects\build-CyberDragon-Desktop_Qt_5_4_1_MinGW_32bit-Debug\CyberDragon\CyberDragon.exe crashed
```

At the bottom left corner you can choose either **Debug build** or **Release build**. To build just click the upper green arrow or alternatively press Ctrl + R.

One more step before you start compiling: click the **Projects** button from the left and then click **Add Build Step** and from there choose **Make**



Build Settings

Edit build configuration: Debug

General

Shadow build: ☑

Build directory: C:\Projects\build-CyberDragon-Desktop_Qt_5_4_1_MinGW_32bit-Debug

Build Steps

qmake: qmake.exe CyberDragon.pro -r -spec win32-g++ "CONFIG+=debug"

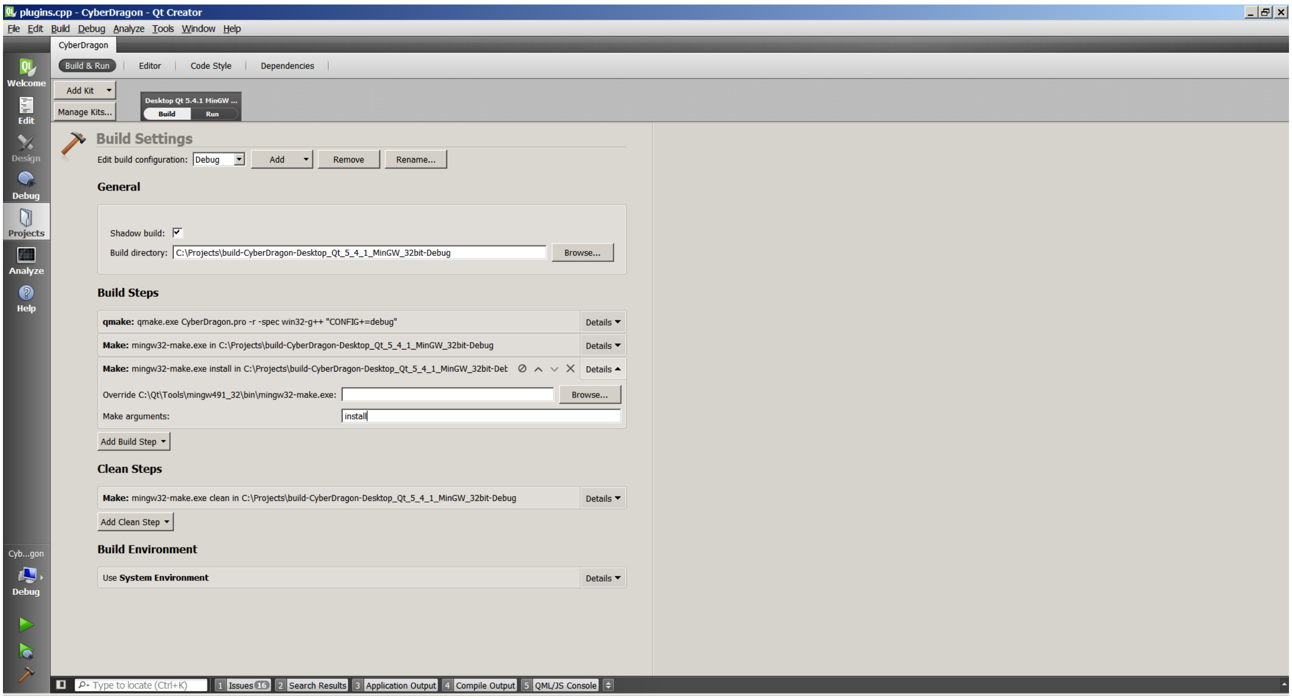
Make: mingw32-make.exe in C:\Projects\build-CyberDragon-Desktop_Qt_5_4_1_MinGW_32bit-Debug

Make: mingw32-make.exe clean in C:\Projects\build-CyberDragon-Desktop_Qt_5_4_1_MinGW_32bit-Debug

Build Environment

Use **System Environment**

Then type **install** to **Make arguments** field.
You have to do this step for both Debug build and Release build.



Edit build configuration: Debug

General

Shadow build: ☑

Build directory: C:\Projects\build-CyberDragon-Desktop_Qt_5_4_1_MinGW_32bit-Debug

Build Steps

qmake: qmake.exe CyberDragon.pro -r -spec win32-g++ "CONFIG+=debug"

Make: mingw32-make.exe in C:\Projects\build-CyberDragon-Desktop_Qt_5_4_1_MinGW_32bit-Debug

Make: mingw32-make.exe install in C:\Projects\build-CyberDragon-Desktop_Qt_5_4_1_MinGW_32bit-Det

Override C:\Qt\Tools\mingw491_32\bin\mingw32-make.exe:

Make arguments: install

Clean Steps

Make: mingw32-make.exe clean in C:\Projects\build-CyberDragon-Desktop_Qt_5_4_1_MinGW_32bit-Debug

Build Environment

Use **System Environment**

Now click the **Edit** button from left and then the **Compile Output** at the bottom.
Finally you can press that green arrow (or Ctrl + R) and watch stuff scroll ;-)
If everything went ok, it will automatically start CyberDragon after successfull compile.

Have Fun!