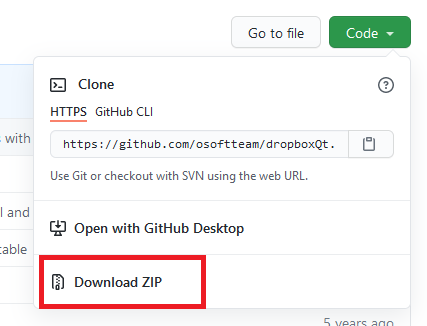
Integrating DropboxQt into Qt project

DropboxQt is available on GitHub:

<https://github.com/osoftteam/dropboxQt>

Download the zip file of library and extract it into somewhere.



We extracted the folder in below directory:

C:\Projects\QtDropbox\dropboxQt-master-temp

This directory includes below subfolders:

* autotest
* bin
* examples
* prj
* src

And below files:

* .gitignore
* LISENCE
* README.md

We can delete the files and keep the folders.

We start with **/prj/** folder. This folder has a dropboxQt.pro file for Qt. this .pro file can be used by qmake. Click on address bar of the folder:

C:\Projects\QtDropbox\dropboxQt-master-temp\prj

and type “cmd”. Command window opens. Give the command “qmake”. Below results will be printed:

|  |
| --- |
| C:\Projects\QtDropbox\dropboxQt-master-temp\prj>qmake  Info: creating stash file C:\Projects\QtDropbox\dropboxQt-master-temp\prj\.qmake.stash |

Below files will be generated:

* .qmake.stash
* Makefile
* Makefile.Debug
* Makefile.Release
* object\_script.libdropboxQt.Debug
* object\_script.libdropboxQt.Release

We will work with two of these files:

* Makefile.Debug
* Makefile.Release

And two folders are generated. Both empty:

* debug
* release

With the next commands, we create lib files with .o ending for debug and release (these commands will take several minutes to generate all files):

|  |
| --- |
| C:\Projects\QtDropbox\dropboxQt-master-temp\prj>mingw32-make -f Makefile.Debug  C:\Projects\QtDropbox\dropboxQt-master-temp\prj>mingw32-make -f Makefile.Release |

422 files in each folder will be created.

# Creating executables for examples

We start with the example “authorize” in folder:

C:\Projects\QtDropbox\dropboxQt-master-temp\examples\authorize

We click on the address bar and type “cmd” and enter the command “qmake”:

|  |
| --- |
| C:\Projects\QtDropbox\dropboxQt-master-temp\examples\authorize>qmake  Info: creating stash file C:\Projects\QtDropbox\dropboxQt-master-temp\examples\authorize\.qmake.stash |

Below files in folder “authorize” will be generated:

* .qmake.stash
* Makefile
* Makefile.Debug
* Makefile.Release

We will work with two of these files:

* Makefile.Debug
* Makefile.Release

And two folders are generated. Both empty:

* debug
* release

Open the file “authorize.pro” and notice the line:

|  |
| --- |
| include( ../examples-common.pri ) |

open the .pri file:

"C:\Projects\QtDropbox\dropboxQt-master-temp\examples\examples-common.pri"

Notice the line:

|  |
| --- |
| LIBS += -L../../prj/debug/ -ldropboxQt |

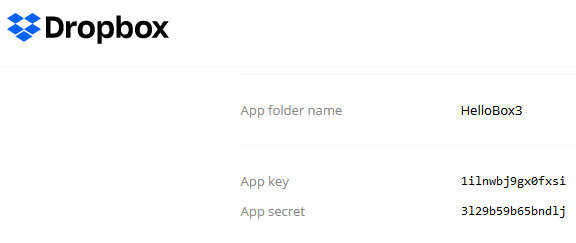
examples-common.pri file is ready to be used for debug mode. Enter below command to build executable for debug mode:

|  |
| --- |
| C:\Projects\QtDropbox\dropboxQt-master-temp\examples\authorize>mingw32-make -f Makefile.Debug |

The size of the .exe file must be around 4.3 MB. For whatever reason, generating release executable doesn’t work well and I don’t know why.

## Running example “authorize”

First we need to register our app in Dropbox website. After registration, we get a key and a secret code for our app:



I set:

* Allow public users: Allow
* Access token expiration: No expiration

With this key and secret code, we generate the file:

"C:\Projects\QtDropbox\dropboxQt-master-temp\examples\authorize\debug\app.info"

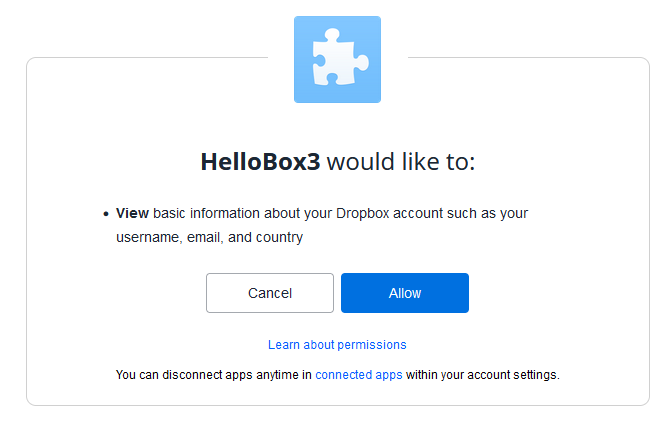
With the content of the file:

|  |
| --- |
| {  "key": "1ilnwbj9gx0fxsi",  "secret": "3l29b59b65bndlj"  } |

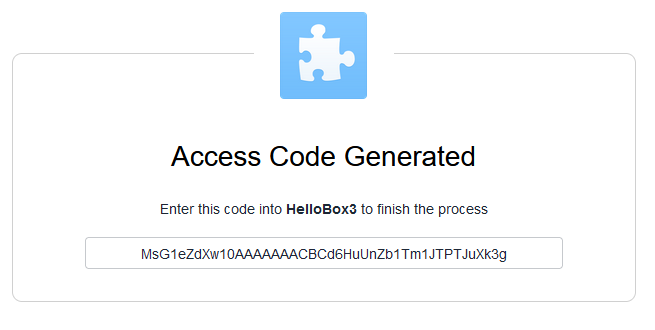
Now we click on the address bar of the folder containing the executable and type:

|  |
| --- |
| C:\Projects\QtDropbox\dropboxQt-master-temp\examples\authorize\debug>authorize.exe app.info token.info  1. Go to https://www.dropbox.com/oauth2/authorize?response\_type=code&client\_id=1ilnwbj9gx0fxsi  2. Click "Allow" (you might have to log in first).  3. Copy the authorization code.  Enter the authorization code here:  > |

Copy the link highlighted in green to web browser. We will be directed to Dropbox website for authorization:



We click on “allow” and are given a code:



Copy this code and paste in command window and press Enter:

|  |
| --- |
| C:\Projects\QtDropbox\dropboxQt-master-temp\examples\authorize\debug>authorize.exe app.info token.info  1. Go to https://www.dropbox.com/oauth2/authorize?response\_type=code&client\_id=1ilnwbj9gx0fxsi  2. Click "Allow" (you might have to log in first).  3. Copy the authorization code.  Enter the authorization code here:  >MsG1eZdXw10AAAAAAACBCd6HuUnZb1Tm1JTPTJuXk3g  finished, token stored in file token.info |

The last line shows that the operation was successful. We notice the new file has been created:

"C:\Projects\QtDropbox\dropboxQt-master-temp\examples\authorize\debug\token.info"

with contents of a token in it:

|  |
| --- |
| {  "access\_token": "AuhyJD1pHNwAAAAAAAAAAcuUh3hfENk3M0hJ4wjPr-Wu9cFG0WELH9Q5IPJKJmUZ"  } |

## Problem with getting dropboxQt to work

Getting this step to work was not easy for me. I got an exception error with the message:

“TLS initialization failed on GET Request”

I looked at the page:

<https://stackoverflow.com/questions/53805704/tls-initialization-failed-on-get-request/53809217>

I realized that OpenSSL hasn’t been installed with my Qt. Hence, I executed the file with double clicking on it and then I chose “add or remove components”:

"C:\Qt\Qt5.13.0\MaintenanceTool.exe"

Under “Qt > Developer and designer tools” , I checked “OpenSSL toolkit” and installed it. Below folder will appear in my Qt installation directory:

C:\Qt\Qt5.13.0\Tools\OpenSSL

Then I added two directories to my “PATH” environment variable:

C:\Qt\Qt5.13.0\Tools\OpenSSL\Win\_x64\bin

C:\Qt\Qt5.13.0\Tools\OpenSSL\Win\_x86\bin

In order to test SSL working fine, open command window and type:

|  |
| --- |
| C:\Projects\QtDropbox\dropboxQt-master-temp\examples\authorize\debug>openssl genrsa 2048 > website-file.key  Generating RSA private key, 2048 bit long modulus (2 primes)  .....................................................................................................................................+++++  ............................................+++++  e is 65537 (0x010001) |

A file must be generated:

"C:\Projects\QtDropbox\dropboxQt-master-temp\examples\authorize\debug\website-file.key"

## Example “account-info”

Creating executables for debug and for release is exactly like example “authorize”. When the executable is ready, copy the file “token.info” from example “authorize” in the folder of “account-info.exe”. Then open command window in folder and give below command and see the result:

|  |
| --- |
| C:\Projects\QtDropbox\dropboxQt-master-temp\examples\account-info>qmake  Info: creating stash file C:\Projects\QtDropbox\dropboxQt-master-temp\examples\account-info\.qmake.stash  C:\Projects\QtDropbox\dropboxQt-master-temp\examples\account-info>mingw32-make -f Makefile.Debug  g++ -c -fno-keep-inline-dllexport -g -Wall -W -Wextra -fexceptions -mthreads -DUNICODE -D\_UNICODE -DWIN32 -DMINGW\_HAS\_SECURE\_API=1 -DQT\_NETWORK\_LIB -DQT\_CORE\_LIB -I. -I..\..\src -IC:\Qt\Qt5.13.0\5.13.0\mingw73\_32\include -IC:\Qt\Qt5.13.0\5.13.0\mingw73\_32\include\QtNetwork -IC:\Qt\Qt5.13.0\5.13.0\mingw73\_32\include\QtCore -Idebug -IC:\Qt\Qt5.13.0\5.13.0\mingw73\_32\mkspecs\win32-g++ -o debug\main.o main.cpp  g++ -Wl,-subsystem,console -mthreads -o debug\account-info.exe debug/main.o -L..\..\prj\debug -ldropboxQt C:\Qt\Qt5.13.0\5.13.0\mingw73\_32\lib\libQt5Networkd.a C:\Qt\Qt5.13.0\5.13.0\mingw73\_32\lib\libQt5Cored.a  C:\Projects\QtDropbox\dropboxQt-master-temp\examples\account-info>cd debug  C:\Projects\QtDropbox\dropboxQt-master-temp\examples\account-info\debug>account-info.exe token.info  {  "account\_id": "dbid:AAAxY7g2fbKD-\_tB4JlgI4QAE-baCKppeG0",  "account\_type": "basic",  "country": "DE",  "disabled": false,  "email": "robodaam@yahoo.com",  "email\_verified": true,  "is\_paired": false,  "locale": "en",  "name": {  "abbreviated\_name": "DE",  "display\_name": "Damon Emami",  "familiar\_name": "Damon",  "given\_name": "Damon",  "surname": "Emami"  },  "referral\_link": "https://www.dropbox.com/referrals/AABc7f-DLO3EGRcyswmYfJ3wAPdaRMBKroA?src=app9-1857969",  "team": {  "sharing\_policies": {  "shared\_folder\_join\_policy": "from\_team\_only",  "shared\_folder\_member\_policy": "team",  "shared\_link\_create\_policy": "default\_public"  }  }  }  Finished, press ENTER to exit |

The part highlighted in green is the response of our request for account information from Dropbox.

# Adding dropboxQt to Qt project

In this example we want to print the account information of the Dropbox-given token in a Qt widget application. First we create a very simple QMainWindow Qt project and make sure it works. We assume that the project name is HelloBox3

Then below steps:

1. Make a copy of “examples-common.pri” in Qt project folder. I renamed the copy to “dropbox-common.pri”.
2. Edit “dropbox-common.pri”.
3. Edit “HelloBox3.pro” project file.
4. Write dropbox code in source file of the project.
5. Run the project and check the result.

## Steps 1 & 2: Making copy of “examples-common.pri” file

After copying and renaming “examples-common.pri” file, we make the highlighted changes in the file. The file is located in:

"C:\Users\diana\OneDrive\Dokumente\QtProjects\HelloBox3\HelloBox3\dropbox-common.pri"

|  |
| --- |
| QT += core network  QT -= gui  CONFIG += console  CONFIG -= app\_bundle  TEMPLATE = app  DROPBOX\_DIR=C:/Projects/QtDropbox/dropboxQt-master-temp  SRC\_DIR = $${DROPBOX\_DIR}/src  #ENDPOINT\_DIR = $${SRC\_DIR}/endpoint  #STONE\_GENERATED\_DIR = $${SRC\_DIR}/stone  #INCLUDEPATH += $${STONE\_GENERATED\_DIR} $${SRC\_DIR}  INCLUDEPATH += $${SRC\_DIR}  ##############################################################  # specify path to the dropboxQt lib, on windows it can be #  # ../../prj/debug/ on unix - shadow build path #  ##############################################################  unix {  LIBS += -L../../prj/ -ldropboxQt  }  win32 {  LIBS += -L$${DROPBOX\_DIR}/prj/debug/ -ldropboxQt  } |

So basically what I do is that I define a new variable, “DROPBOX\_DIR” and I use it to locate libs and source files.

## Step 3: Edit project .pro file

It is just “including” the file “dropbox-common.pri” by adding one cute little line:

|  |
| --- |
| #For dropbox:  include(dropbox-common.pri) |

Notice that I don’t specify the full path to .pri file because the project file and .pri file are in the same folder.

## Step 4: Source code for getting account information

We can put the code in constructor of MainWindow class (notice the highlighted line which is complete path to “token.info”):

|  |
| --- |
| #include <QtCore/QCoreApplication>  #include "dropbox/DropboxClient.h"  #include "dropbox/users/UsersRoutes.h"  #include "dropbox/endpoint/DropboxAppInfo.h"  #include "dropbox/endpoint/DropboxAuthInfo.h"  MainWindow::MainWindow(QWidget \*parent) :  QMainWindow(parent),  ui(new Ui::MainWindow)  {  ui->setupUi(this);  QString argAuthFile = "C:/Projects/QtDropbox/dropboxQt-master-temp/examples/account-info/debug/token.info";  dropboxQt::DropboxAuthInfo authInfo;  // Read auth info file.  if(!authInfo.readFromFile(argAuthFile))  {  ui->label->setText("Error reading <auth-file>");  }  dropboxQt::DropboxClient c(authInfo.getAccessToken());  try  {  std::unique\_ptr<dropboxQt::users::FullAccount> accountInfo = c.getUsers()->getCurrentAccount();  ui->label->setText(accountInfo->toString());  }  catch(dropboxQt::DropboxException& e)  {  ui->label->setText("Exception: " + QString::fromStdString((e.what())));  }  } |

## Step 5: Running the application

We have placed a QLabel object in main window and display account information in that label. Here is how it looks like:

