Facebook Connect

News

QFaceBookConnect is available for download here Updated version of this wiki is available at: http://gitorious.org/qfacebookconnect/pages/Home

A more comprehensive example on how to use qfacebookconnect is facebrick, facebrick aims to implement a facebook client for Nokia N900, It should also compile on Symbian with minor modifications.

Introduction

QFacebookconnect is a port of the Facebook connect for iPhone to Qt so enabling Symbian, Maemo, and any Qt supported platform to make API calls to Facebook's platform.

How to use

There is a sample application in the sample/ folder, which can be a good starting point.

Session

The first thing that you need to do is to create a session, like this:

```
FBSession* session = FBSession::sessionForApplication("<YOUR APP KEY>", "<YOUR APP SECRET>", QString());
```

The third parameter is for specifying a session proxy, read more about it on facebook developer pages. The FBSession::sessionForApplication creates a new instance of the FBSession and it's your job to delete it later when its not needed any more.

Once you have a session object, you can connect to the signals you are interested in:

```
void sessionDidLogin (FBUID aUid);
void sessionDidNotLogin ();
void sessionWillLogout (FBUID aUid);
void sessionDidLogout ();
```

I think the method names are self explanatory. You can always use the sender() function from QObject to find out who sent the signal.

Unlike the objective-c version of Facebook connect, you don't have to worry about disconnecting signals when u delete your session. Qt will take care of that.

Login

To login to facebook you can use the FBLoginDialog like so:

```
if (iFBSession->resume() == false)
{
    iLoginDialog = new FBLoginDialog();
    iLoginDialog->show();
}
```

The FBSession::resume() function checks if it is possible to restore a previously saved session and thus avoid logging in again, if it is possible the sessionDidLogin() signal will be emitted before it returns true.

How to make a request

Use the FBRequest::request() static function to get an FBRequest object. Then use the call method supplying it with the request name and parameters, like so:

```
FBRequest* request = FBRequest::request();
Dictionary params;
QString query = "select name,pic_big, status,birthday_date, timezone from user where uid in (select uid2 from friend where uid1=
params["query"] = query;
connect (request, SIGNAL(requestDidLoad(QVariant)), this, SLOT(requestDidLoad(QVariant)));
connect (request, SIGNAL(requestFailedWithFacebookError(FBError)), this, SLOT(requestFailedWithFacebookError(FBError)));
request->call("facebook.fql.query",params);
```


(C) Copyright Nokia 2013. All rights reserved.

and then in the signal method:

```
void MainWindow::requestDidLoad(const QVariant& aContainer)
{
if (aContainer.type() == QVariant::List)
{
    QVariantList list = aContainer.toList();
    for (int i = 0 ; i < list.count(); i ++)
    {
        QVariantHash dictionary = list.at(i).toHash();
        QHashIterator<QString, QVariant> iterator(dictionary);
        QString name = dictionary.value("name").toString();
        ui->listWidget->addItem(name);
    }
    sender()->deleteLater(); // delete the request !
}
}
```

QVariantList, QVariantHash and QVariant

I decided to keep the code simple and therefore I opted to use the QVariants, but as the qt document on QVariant says: "You can even store QList<QVariant> and QMap<QString, QVariant> values in a variant, so you can easily construct arbitrarily complex data structures of arbitrary types. This is very powerful and versatile, **but may prove less memory and speed efficient than storing specific types in standard data structures."** QVariant

May be later on this can be improved.

Getting Permissions

Use FBPermissionDialog to do that, read more on the Facebook connect pages on Facebook about it.

How to report bugs

Please file bug reports on Google Code @ QFacebook on Google Code

Contact me at My blog

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.