

KQOAuth – Easy and Powerful OAuth library for Qt

Introduction

kQOAuth is a powerful yet easy way to integrate OAuth authentication to your Qt application. kQOAuth handles OAuth request signing, request submitting and reply parsing for you. It also provides an easy way to retrieve user authorization to protected resources with a built in HTTP server. All this is done with Qt programming in mind, so you can use Qt's signals to react to OAuth events in your own application. This also means that the library works fully asynchronously. kQOAuth is licensed under the LGPL license.

kQOAuth is written by Johan Paul (johan.paul@gmail.com). The web site of kQOAuth is at [1]. The content on this wiki page is taken from this web page.

Main Components

The two main classes of kQOAuth are "KQOAuthManager" and "KQOAuthRequest" . KQOAuthManager executes the request and keeps track of the authorization process, while KQOAuthRequest describes the next step of the process and takes care of formatting the headers for the request.

You can find more info on it with example:[here](#)

If you are unfamiliar with "OAuth", it would be nice to checkout the tutorials below:

[Beginner's Guide to OAuth – Part II : Protocol Workflow](#)

[<https://dev.twitter.com/docs/auth#intro> Authenticating Requests with OAuth]

Features

kQOAuth is written in C++ and its main features include:

- Provides easy, "automated", OAuth authentication but also more advanced detailed control of the authentication process.
 - OAuth request handling (request signing, submitting and result parsing).
 - No dependencies to external libraries.
 - Works with signals and slots - asynchronously.
 - Handles protected resource owner authentication by taking care of opening temporary token retrieval web page and parsing the reply with a built in HTTP server.
-

Download and source code

kQOAuth library source code is available on Gitorious.org at [this page](#)

Usage

kQOAuth can be used in two different ways. If you don't want to control the authentication process in detail, you can use the convenience APIs to get access to protected resources with a few lines of code. The only pieces of information you will need are

- your application's consumer key
 - your application's secret consumer key
 - URL for retrieving the temporary request tokens
 - URL for the user to authenticate the application to access the protected resources
 - URL for sending the authenticated request.
-

Disclaimer

kQOAuth is at version 0.92. Use it at your own risk. It has so far been successfully tested on Mac OS X 10.6 and Ubuntu Linux 10.10. Only Qt 4.7 and newer are supported. kQOAuth currently supports only HMAC-SHA1 signing method. kQOAuth should be Symbian compliant, but feedback regarding this is needed.

