Creating an HTTP network request in Qt using QNetworkAccessManager, QNetworkRequest, QNetworkReply

This article demonstrates how to prepare a **HTTP** request and to make use of the response.

The example code demonstrates usage of QNetworkAccessManager, QNetworkRequest, QNetworkReply, QUrl and QSettings. `HttpCommunicator` and `RawRequestUi` are important classes which makes use of several Qt services to implement example application.

UI for Entering the URL

RawRequestUi File:HttpConsole.zip class implements simple UI where user can enter URL such as http://beer.com RawRequestUi provides following user input elements

- Select type of request i.e. get, post, head, put or delete.
- Choose request body. Body can be taken from file or can be entered in text input field provided.
- Response area displays headers and response body received from **HTTP** request.
- Header area where user can insert **HTTP** header into the request.

Screen on Nokia 5800 using a Wifi Network

Screen on Nokia 5230 using a 3G Network

Creating Request

(c) Copyright Nokia 2013. All rights reserved.

`HttpCommunicator` class makes use of QNetworkAccessManager, QNetworkRequest, QNetworkReply, QUrl to get its job done. `QNetworkRequest::setUrl()` is all you have to do to prepare a **HTTP** request. User entered headers are inserted using the API `QNetworkRequest::setRawHeader()`.

Communicating with Server

QNetworkAccessManager encapsulates nitty-gritty of actual communication. QNetworkAccessManager has APIs such as `get()`, `post()`, `deleteResource()`, `put()`, `head()` to make **HTTP** request. `HttpCommunicator` hooks onto QNetworkAccessManager by providing slot for `QNetworkAccessManager::finished()`.

Catching the Response

QNetworkReply makes the job easier by representing the network reply through simple and intuitive APIs. `QNetworkAccessManager::get()` and other request making functions returns pointer to QNetworkReply. Same pointer is returned in the parameter to `QNetworkAccessManager::finished()`. It is job of caller to delete QNetworkReply object but beware that you have to use `QNetworkReply::deleteLater()` if you want to delete the object inside the slot `QNetworkAccessManager::finished()`. `QNetworkReply::rawHeaderList()` is useful function which returns header list returned by server. `RawRequestUi::processSngResponse()` makes use of `QNetworkReply::rawHeaderList()`, `QNetworkReply::rawHeader()` to read **HTTP** reply headers.

Miscellaneous

Example code also makes use of QSettings to remember data entered in user input fields.

Example project

Download from Gitorious.org

SISX File Media:HttpConsole template.sis

(c) Copyright Nokia 2013. All rights reserved.

(c) Copyright Nokia 2013. All rights reserved.