Themen
Startseite
Rezepte:
4k Eintopf
Dippekuchen
Brot backen
Körnerbrot
Waffeln
Elektronik:
Klipp-Klapp, Experimente mit Relais
Leuchtdioden
Transistoren
Optokoppler an 230 Volt
Gießmelder
Mikrofon Vorverstärker
Audio Level Limiter
Office Makros für Uni-Trend Oszilloskope
Mikrocontroller:
Einblick in die moderne Elektronik ohne viel Theorie
Multithreading in C und Arduino
Protothreads und Protosockets
STM32 Anleitungen
ESP8266 und ESP8285 WLAN Module
Windows Treiber für PL2303
Binäruhr
YupStick
AVR:
Einstieg in die Elektronik mit Mikrocontrollern
Assembler Workshop

QtWebApp HTTP Server in C++

QtWepApp is a HTTP server library in C++, inspired by Java Servlets. For Linux, Windows, Mac OS and many other operating systems that the Qt Framework supports.

QtWebApp contains the following components:

- HTTP(S) 1.0 and 1.1 server
- Template engine
- Buffered logger

These components can be used independently of each other. A very small usage example:

```
// The main program starts the HTTP server
int main(int argc, char *argv[])
{
    QCoreApplication app(argc,argv);

    new HttpListener(
        new QSettings("configfile.ini", QSettings::IniFormat, &app),
        new MyRequestHandler(&app),
        &app);

    return app.exec();
}


// The request handler receives and responds HTTP requests
void MyRequestHandler::service(HttpRequest& request, HttpResponse& response)
{
    // Get a request parameters
    QByteArray username=request.getParameter("username");

    // Set a response header
    response.setHeader("Content-Type", "text/html; charset=UTF-8");

    // Generate the HTML document
    response.write("<html><body>");
    response.write("Hello ");
    response.write(username);
    response.write("</body></html>");
}
```

The small memory requirement of about 2MB qualifies the web server to be used in embedded systems. For example the Beer brewing machine of Sebastian Düll. But it's also powerful enough for larger web services.

The logger improves disk space and performance by retaining debug messages in memory until an error occurs. No debug messages are written as long everything works fine. Changes to the configuration of the logger become active automatically without program restart.

Deutsche Version dieser Seite

Last changes

1.9.0 07.09.2023
Some methods return now const references instead of objects to improve the performance a little bit.
Chunked transfer is not used anymore if the Content-Length is known.

1.8.8 24.01.2023
Better HTTP status code (400 bad request) if the first line of request is malformed.
Connection closes more quickly after an Error.

1.8.7 23.01.2023
Fix possible endless loop with high CPU load after receiveing a broken HTTP request.

1.8.6 30.09.2022
Fix compile error under Windows: sslCaCertFileName was not declared

1.8.5 19.03.2022
Add support for SSL peer verification and CA certificate.

1.8.4 29.10.2021
Add Content-Length header to static file controller.

1.8.3 21.03.2021
The minLevel for logging can now be configured as string.
Info messages are now positioned between DEBUG and WARNING.
I also added an example for HTTP Basic authorization.

Source code, Tutorial, API documentation .

The library runs with Qt version 4.7 to 6.x. In case of Qt 6 you need to install the Qt5Compat library. It contains support for a lot of 8 Bit character encodings, that Qt6 does not support anymore by default. You may use the software under the conditions of the LGPL License.

How it started

Many years ago a very experienced Java developer insisted that Java is **the** internet language, because doing internet communication is much more complicated in other programming languages. We started a challenge to verify this. The quest was:

- Program a HTTP server with some basic features using only the Java 6 runtime Library versus Qt 4.

After a weekend, we both presented what we achieved so far. Both programs were quite similar, so he admitted that he was wrong. By the way, the Qt/C++ program was much smaller and a bit faster than his Java version.

A few years later I made a library from that prototype and used it in some few private projects. Collegues encouraged me to publish the code. Since then I had no use of the project anymore, but I applied some improvements anyway to make people happy. Interestingly the Qt makers are working on a standard HTTP server since many years, but in 2023 it is still not included in the Qt library. That might explain why many people use my library instead.

Thanks

The current high quality of the library is the result of the collaboration of many people. I thank the helpers for testing QtWebApp extensively in a real productive environment and thus contributing to the improvement.