MinaliC

A minimal web server developed in C

Specification

Version 2.0

Date: 2011-03-21
Author: Hans Alshoff, Sweden

1 Introduction

MinaliC is a small web server that is developed in the C programming language.
Currently it has been developed for the Windows platform.
It is intended to be a minimal server to be used for small web applications, when full scale web server functionalities is not needed. The server has support for calling CGI scripts (Common Gateway Interface). It also has support for calling PHP and Perl scripts.
MinaliC can be run either as a windows service or as a console application.
The server only has one executable binary - minalic.exe.
MinaliC can be expanded by building plug-ins that implements certain protocols, so the server can be more than just a web server. It can be a server for your own protocol.

2 Directory structure

The MinaliC binary can be placed in any folder on the disk. In this document that folder will be called <Binary folder>. In that folder there must be a subfolder called wwwroot. That is the root folder of all web applications.
In the wwwroot you can create subfolders that will act as web-folders.
Example of directory structure:

```
<Binary folder>
        |
        minalic.exe
        minalic.ini
        minalic.log
        <wwwroot>
            |
            index.htm
            logo.bmp
            <subdir>
                |
                index.cgi
                <subdir2>
                    |
                    index.php
```

The web default file - the file chosen if no filename is specified- is index.cgi, index.php index.pl and index.htm (in priority order from the highest).

3 Configuration

The configuration file is called minalic.ini and is placed in the <Binary folder>.
Configuration variables are:

- HTTP_PORT, sets the port that the server listens to. Default value is 80.
- SERVER_LOGLEVEL is the level of how much of the server activities that is logged. Default value is 1. With a value of 2 the server will log http request and response headers.
- SESSION_TIMEOUT is the number of seconds that a session is valid. Default value is 1800 (30 minutes).
- SERVER_HANDLERS is the number of simultaneous client handler threads that are allowed. Default value is 100.
- CGI_TIMEOUT is the number of seconds that a CGI module can be running without returning any data. Default value is 90 seconds.
- SERVER_PLUGIN loads a plug-in module and binds it to a port number.
- DIRECTORY_BROWSING_DEFAULT, decides if directory browsing allowed for all directories (1) or only directories with a dir.ini file (0). Default value is 0.

For example : HTTP_PORT = 8080.
A line is commented with a beginning asterisk (*).

4 Logging

The name of the log file is minalic.log and it will be created in the <Binary folder>.
When the server is started as a console application, all logging will be done through the standard output.

5 Running

To install the server as a windows service type minalic.exe –install. The created service will be started and set to start automatically. To uninstall the service type minalic.exe –uninstall. With the flags –start and –stop the service can be set to start or stop.
To specify the name of the service when installed type minalic.exe –install –name:*my_name*.
The –name: option can be used on all of the commands –install, -uninstall, -start and –stop.
Example: minalic.exe –stop –name:*my_service.*

Starting the binary minalic.exe with no parameters will start the server as a console application.

6 Directory browsing

By default directory browsing is turned off. It can be turned on by setting the configuration value DIRECTORY_BROWSING_DEFAULT to 1. If this value is set to 0, directory browsing is by default not allowed in any directory. But directory browsing can be turned on in a specific directory by putting an empty file called dir.ini in the specific directory.

7 HTTP specific

7.1 Requests

MinaliC supports the request methods GET and POST. For other requests the server will respond with a 501 Not implemented.

7.2 Response

A response from the server can be any of these:

- 200 OK
- 501 Not implemented
- 404 Not Found
- 400 Bad Request
- 301 Moved Permanently
- 302 Redirect
- 500 Internal Server Error
- 403 Not Modified

HTTP header fields that MinaliC sends in a response are:

- The HTTP status line (example HTTP/1.1 200 OK)
- Date:
- Content-Length:
- Content-Type:
- Last-Modified:
- Server:
- Connection:
- Set-Cookie:

The server can recognize 46 file extensions and decide the correct mime type to return in the Content-Type header.
If a request includes the header If-Modified-Since it looks to see if the requested file has been updated since last call.

8 Sessions

MinaliC handles sessions by help of a sessionid that is included in a domain cookie. For each request the server gets it will respond with a valid sessionid. The timeout of a session occurs when there has not been any request for a specific amount of time. That time is set in the configuration file with the variable SERVER_TIMEOUT. If the server finds out that the sessionid returned by the browser is timed out it will create a new sessionid. A sessionid is a 30 byte alfa numeric string.

9 CGI

A CGI module is a file with the extension .cgi. The server will try to load the corresponding file with the extension .exe. CGI modules can be put in the same folder as the HTML pages. For example the request http://localhost/subfolder/myscript.cgi.will invoke the executable myscript.exe that is put in the wwwroot\subfolder directory.

When a CGI module is invoked the server sends the input content to the module through standard input and it receives the output from the CGI module back from the modules standard output.
The CGI module can send the following headers back to the server:

- Location:
- Content-Type:
- Set-Cookie:

If the script sends an absolute location address in a Location header (starting with http:/) the server will send a 302 Redirect to the client. If the script sends a local address the server will handle the redirect by itself by responding with the new local page that was requested.
The CGI module will receive the following headers from the server through the environment variables:

REQUEST_METHOD
QUERY_STRING
SERVER_SOFTWARE
SERVER_NAME
SCRIPT_NAME
SCRIPT_FILENAME
GATEWAY_INTERFACE
SERVER_PROTOCOL
REMOTE_ADDR
SESSION_ID
WWWROOT (physical path to the wwwroot directory for the server)
SERVER_ROOT (physical path to the bin directory for the server)
HTTP_ACCEPT
CONTENT_TYPE
CONTENT_LENGTH
HTTP_COOKIE

10 PHP and Perl

MinaliC also has support for running PHP and Perl scripts. To run this kind of scripts from the server, PHP or Perl first must be installed on the server. MinaliC calls PHP and Perl as a CGI module and communicates with the interpreter as with ordinary CGI modules. The PHP executable must be named php.exe, and the Perl executable must be named perl.exe.

11 Internal structure and source code

MinaliC is a multi threaded server. Each request will end up in a handler thread for that request. For safety reasons the number of simultaneous threads can be limited by the configuration parameter SERVER_HANDLERS. The requests over that limit will be rejected.

The source code is developed in an object oriented manner – but in C.
Each class has a header file (.h) and an implementation file (.c).

The server has been tested with Webserver Stress Tool 7.0.

12 Plug-ins

MinaliC has support for loading custom plug-ins to handle all communication on a specific port. In the configuration file it is possible to declare several plug-ins on the form: PLUGIN = {plugin_name}:{port}. A plug-in is a dll file that implements some predefined functions. These functions are called from MinaliC at certain points. For example you could build a plug-in that implements some protocol that is based on TCP/IP. The {plugin_name} is the name of the plug-in dll with the .dll excluded. The {port} is the TCP/IP port to listen on. It is possible to declare several plugging.
For example:
PLUGIN = FTP:21
PLUGIN = LDAP:78.
In this example the server looks for the two dlls´ ftp.dll and ldap.dll. These dlls' should be put in the <Binary folder>.

The implementation of a plug-in dll is based on exporting three functions. These functions are:

1. `char *get_version()`
2. `char *get_build()`
3. `void handle_request(SOCKET s,struct sockaddr_in addr)`

Functions 1 and 2 are very simple. The only purpose of these functions is to return the version and the build id of the plug-in. These values are stings and can be decided by the implementer. Function number 3 is the actual implementation of the protocol handler for the protocol that the plug-in is supposed to implement. MinaliC sends two parameters to the function. These parameters are the socket to read from and send to for the connection and also an address parameter including the clients address.

The function is called from within a handler thread in MinaliC, and each connection on the specific port will end up in a call to this function. This makes the plug-in dll multi threaded. By implementing this function the plug-in therefore can handle several multiple connections at the same time.
Here is a very simple implementation of a plug-in dll in Visual Studio:

```
__declspec(dllexport) char *get_version()
{
            return "2.1";
}



__declspec(dllexport) char *get_build()
{
            return "PL0023";
}

/*This protocol is very simple. It waits for 4 characters from the client
and then sends the string “My return string” back to the client.*/
__declspec(dllexport) void handle_request(SOCKET s,struct sockaddr_in addr)
{
            int         len;
            char        header[256];
            int         max;
            char        *answ="My return string";

            max = 0;
            while (max < 4)
            {
                        len = recv(s,&header[max],1,0);
                        if (len == 0)
                                    return ; /* No data found*/

                        if (len < 0)
                                    return; /* Socket closed*/
                        max += len;
            }
            header[max+1] = 0;
            send(s,answ,strlen(answ),0);
}
```

It is by help of plug-ins possible to make MinaliC handle your own protocol. Several plug-ins can be loaded in one server instance. You can still have the web server running in parallel with the plug-in. It is also possible to turn of the web server and only run your protocol. That is done by specifying the configuration HTTP_PORT = 0.

13 SDK

When building plug-ins you can make use of some functions that helps integrating with the server. MinaliC exports a library (minalic.lib) that it is possible to link your plug-ins towards. These functions are:

```
/*Performs logging*/
void echo(char *txt);

/*Performs logging of certain level*/
void echoex(char *txt,int level);

/*Reads parameters in the configuration file*/
char *get_config_value(char *key, char *value);
```

The functions are declared in the source file minalic.h. You need to include this file in you project to make use of the functions.