GoAhead WebServer 2.5 Getting Started

Compiling With Supported Build Systems

GoAhead WebServer includes maintained project files for Windows and standard GNU command line environments. Windows build system support includes Visual Studio 2010 Express project files. Build support for Linux and Mac OS X is provided using GNU Makefiles and C code compilers. Example Makefiles are also provided for WindRiver VxWorks, Windows CE, LynxOS, Netware, QNX4 and eCos but are not maintained and might require edits. Users are encouraged to submit updates and feedback regarding unmaintained platforms to the WebServer forum (http://webserver.goahead.com/forum).

Systems using GNU tools for Makefiles (LINUX or MAC OS X)

The Makefiles provided for Linux and Mac OS X are self-contained and do not rely on Autoconf or Configure scripts. Ensure a standard C source code compiler is installed on the platform along with the make utility.

```
$> cd webs-2-5/<OS_TYPE>
$> make
```

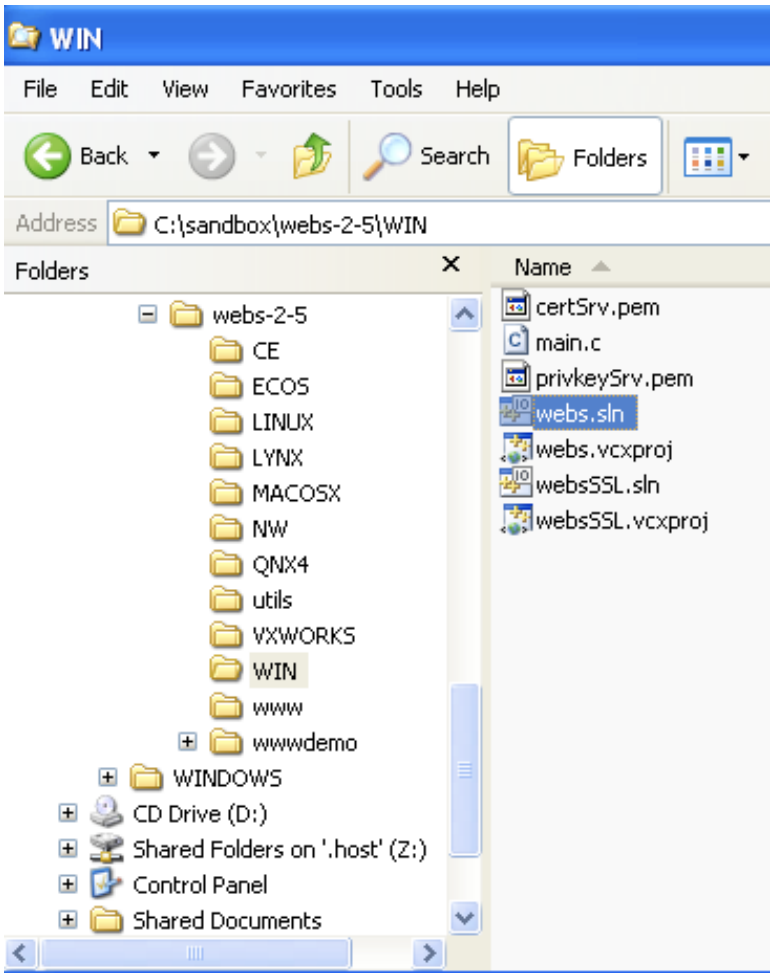
A successful compile will result in a *webs* executable file in the current directory.

Windows (Visual Studio)

Solution and project files for Visual Studio C++ 2010 Express are provided in the package. Visual Studio Express is free evaluation software available from Microsoft's Web site.

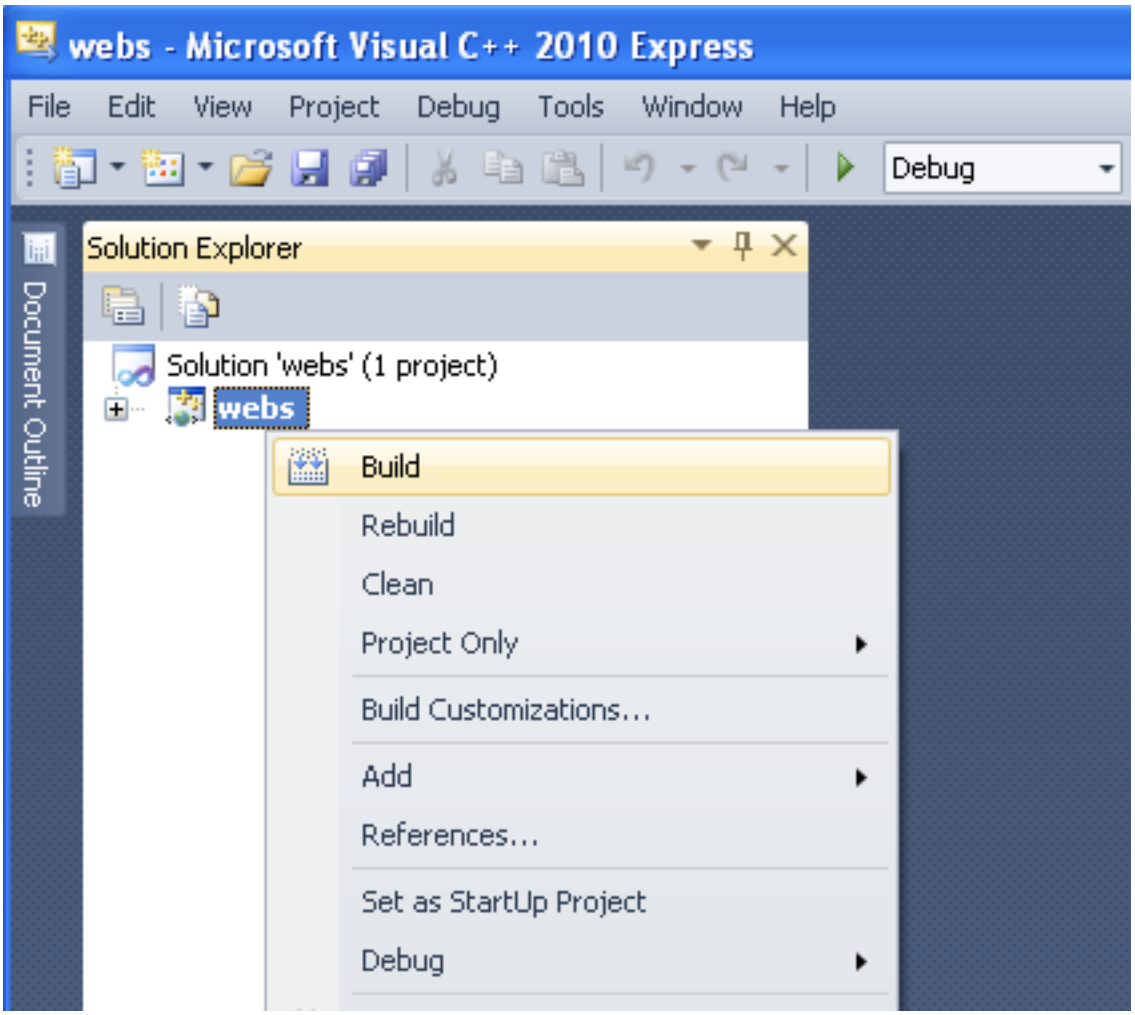
To compile with Windows

1. Open the WebServer Visual Studio solution file in the WIN directory by double-clicking the *webs.sln* file.



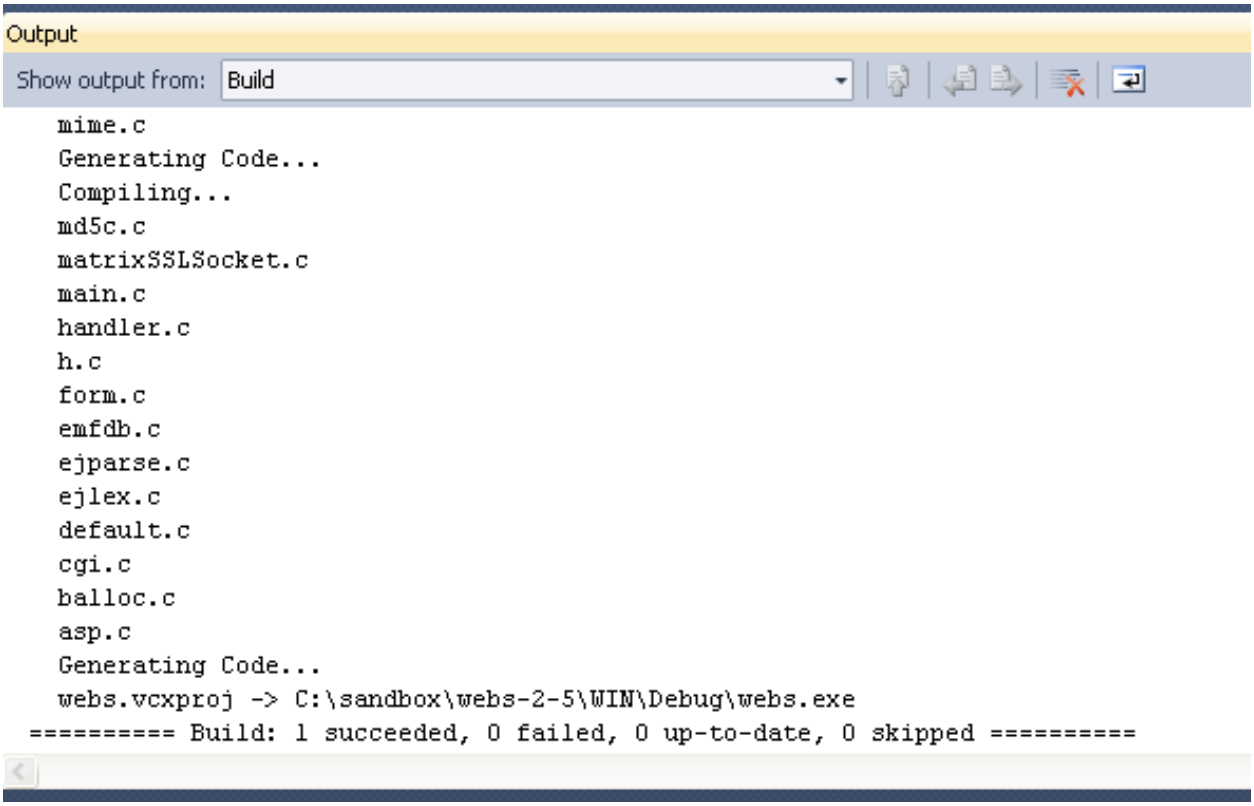
Note: Alternatively, open this file from within Visual Studio. On the File menu, click Open, click Project/Solution, and then locate the webs.sln file.

2. Right-click the **webs** project and select **Build** from the drop-down menu.



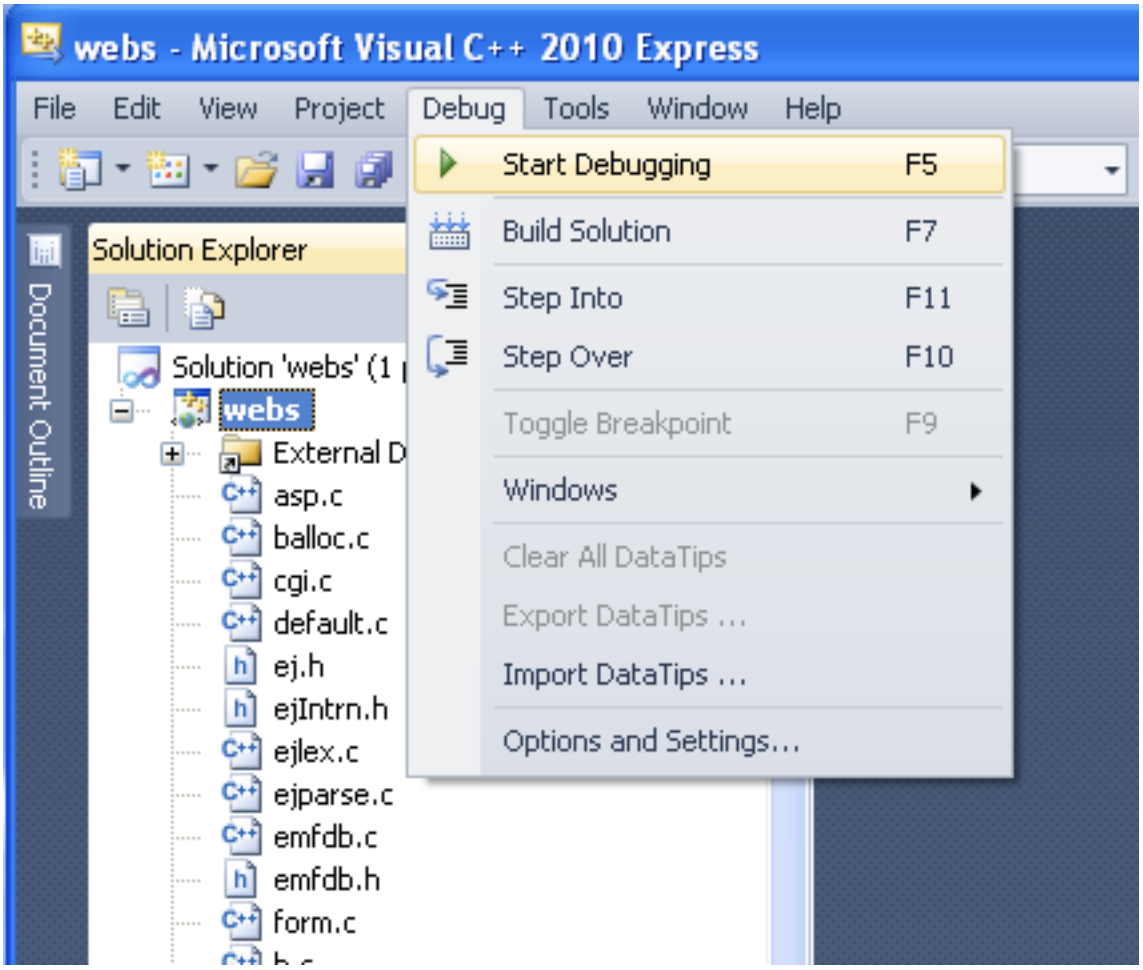
Note: This menu can also be used to clean or rebuild the source code. Alternatively, from within Visual Studio on the Debug menu, click Build Solution.

3. Confirm through the Output window that the compile did not have any errors.



4. If a compile error occurs, double-click the error text to open the file in the Visual Studio editor.

5. On the Debug menu, click Start Debugging to run WebServer in the debugger.



Compiling For Other Platforms

Generic Build Instructions (Eclipse, XCode, Tornado, etc.)

In general, the process of building WebServer on any platform is straightforward.

Source Code

Every C file in the main directory of the package must be compiled. This involves adding all C files to your platform's standard project file mechanism. In addition, the C files specific to your operating system (for example LINUX/main.c) must be added.

Typically, each header file is also added to the project for dependency calculations.

Compile Flags

Compile flags for WebServer are minimal. More information on each of the defines can be found in the configuration section below. The following must always be defined (example using VxWorks):

```
-DWEBS -DOS="VXWORKS" -DVXWORKS
```

Optional defines are described in detail under "Compile Flags" in the "Optional Features and Configurations" section, below. For example, enable the following for full access management control:

```
-DUSER_MANAGEMENT_SUPPORT
-DDIGEST_ACCESS_SUPPORT
```

Linking

WebServer uses basic operating system functions only, such as sockets and optional file system access. On most platforms, additional include libraries are not required for linking. On Windows platforms, ws2_32.lib must be linked to provide some of the sockets functionality.

Optional Features and Configurations

Secure Sockets Layer

PeerSec Networks MatrixSSL is the preferred SSL provider for use with WebServer and a fully supported layer is included in the source. Building an SSL-enabled server using MatrixSSL is described in the document Webs25MatrixSSL.pdf included with WebServer 2.5; however, the **quick steps for LINUX and MAC OS X** are:

1. Download the latest version of MatrixSSL from http://www.matrixssl.org.
2. Unzip the MatrixSSL source code package to the root directory of the WebServer source:

   ```
   $> cd webs-2-5
   $> tar -xzvf matrixssl-3-1-2-open.tgz
   ```

3. Change directory to the unpacked MatrixSSL source directory and build:

   ```
   $> cd matrixssl-3-1-2-open
   $> make
   ```

4. Return to the webs directory and compile:

   ```
   $> cd ../<OS_TYPE>
   $> make
   ```

   The Makefiles for LINUX and MACOSX will detect the MatrixSSL package, apply the necessary preprocessor defines, and link with the SSL library.

Example CGI Implementation

The wwwdemo/cgi-bin/cgitest.c source file is a sample CGI program that creates an HTML page that displays the CGI environment variables. This source-code module is stand-alone and no other source files are required to generate the executable. Typically, it is built by the simple command line:

```
cc -o cgitest cgitest.c
```

Web Content Compilation for ROM (GoAhead WebCompile™)

The utils/webcomp.c source file generates a binary that will package a list of Web content files into a single C file suitable for compiling into WebServer for platforms without a file system. The files can include text or binary types:.html, .jpg, .asp, .css, etc. This source-code module is stand-alone and no other source files are required to generate an executable. Typically, it is built by the simple command line:

```
cc -o webcomp webcomp.c
```

Compile Flags

CFlag	Effect
-DWEBS_SSL_SUPPORT	Enables MatrixSSL for WebServer. Allows all data between the browser and WebServer to be encrypted under the HTTPS protocol. Secure access is via a different port on the Web server (typically 443, and default 4433 in WebServer). See the WebServer25MatrixSSL document for details.
-DDIGEST_ACCESS_SUPPORT	Enables digest authentication for Web password requests, rather than plaintext authentication. Adds some measure of security, although less than if SSL is used. Password requests may be initiated by setting a global password, using User Management, or by custom code. Enabled by default.

CFlag	Effect
-DUSER_MANAGEMENT_SUPPORT	Enables access control lists for URL resources based on user/group policies. Disabled by default.
-DWEBS_IF_MODIFIED_SUPPORT	Adds if-modified-since support. Optimizes bandwidth by only sending pages that have changed since the client's last request. Requires WebServer to be linked with the standard math library "libm".
-DB_STATS -DB_FILL -DDEBUG -DDEV -DASSERT	Enable debugging features. B_STATS prints memory usage/leak stats to a file named "leak.txt". B_FILL zero-fills all memory upon allocation. DEBUG and DEV enable various additional code sections to run. ASSERT enables asserts to be displayed.

Header File Options

The following options are configurable in webs.h in the section, "User Configurable Defines".

CFlag	Effect
Default Index Page WEBS_DEFAULT_HOME	Defines the default page used as an index page when a directory URL is specified. By default the initial setting is "home.htm". Multiple default home page names are not supported.
Server Port (Non Secure) WEBS_DEFAULT_PORT	The Internet standard insecure HTTP port is 80. By default, the WebServer port is set to 8080.
Server Port (SSL secure) WEBS_DEFAULT_SSL_PORT	The Internet standard secure HTTPS port is 443. By default, the WebServer SSL port is set to 4433.

CFlag	Effect
Whitelist Support `WEBS_WHITELIST_SUPPORT`	Validates URLs based on a list of files found under the web root directory. This provides an additional layer of filtering to disable malformed and malicious URLs from being processed by the operating system. It is **highly recommended** that this setting remain enabled.
Logging Support `WEBS_LOG_SUPPORT`	Provides URL access and error logging to a file. By default, the file is “log.txt”, and is hardcoded in webs.c The log output format is Apache Common Log Format and can be parsed by standard log analysis tools.
Keep-Alive Support `WEBS_KEEP_ALIVE_SUPPORT`	Enables HTTP/1.1 keep-alive support. This optimizes speed by holding a connection open for more than one HTTP request.
Proxy Support `WEBS_PROXY_SUPPORT`	Experimental option to track local versus remote requests.
Compiled Web Content `WEBS_PAGE_ROM`	Enable support for WebCompile. Web content is loaded from ROM rather than a file system. For use with data packaged by utils/webcomp.c

Main Executable Options

The following options are set in <*OS*>/main.c for each operating system.

CFlag	Effect
Document Root `*rootWeb = T("www");`	Defines the directory that contains Web-accessible content. Two sample document roots are included in the WebServer package - www and wwwdemo. Only a single document root can be active at any one time.
CGI-BIN Root	The location of the cgi-bin directory is hardcoded in several files in WebServer to “cgi-bin” directly under the Document Root.
Global Password `*password = T("");`	The global password for access to any Web pages. Blank password means pages are not protected. SSL or digest authentication should be enabled as a minimum if using this password mechanism.
Server Port (non-secure)	The server port is set within main.c; by default, the value is WEBS_DEFAULT_PORT as defined in webs.h.
Server Port (SSL secure)	This is set automatically within WebServer to WEBS_DEFAULT_SSL_PORT.
Directories requiring SSL	Specified by the calls to websRequireSSL() within main.c.

Running WebServer

Running the webs file

Running from the command line:

```
$> cd webs-2-5/<OS_TYPE>
$> ./webs
```

Running in demo/documentation mode:

```
$> cd webs-2-5/<OS_TYPE>
$> ./webs -demo
```

Running in Windows

Because command line options are not typically used in Windows applications, enabling demo/documentation mode can also be initiated at compile-time by editing WIN/main.c and uncommenting the line:

```
#define USE_DEMO_MODE 1
```

Either double-click the webs icon, or run webs.exe from a command shell to view any debug or error trace. If building with Visual Studio, the executable is typically located in webs-2-5/WIN/Release or webs-2-5/WIN/Debug.

Stopping WebServer

Operating system-specific methods for stopping processes vary by platform, but typically the “kill” command is used to send a signal to stop the process. In Windows, to stop the Web server, right-click the GoAhead WebServer icon in the taskbar and select Close.

Accessing the Web Server with a Browser

To access the running Web server, open a browser to http://localhost:8080/.

To access the running Web server over an encrypted connection, open a browser to https://localhost:4433/.

Browsers such as Internet Explorer and Safari will present a warning about unsigned certificates and allow you to continue loading the page. Some browsers, such as Firefox 3.0 and above, will return an error message indicating that the certificate is not trusted:

```
Alert
localhost:4433 uses an invalid security certificate.
The certificate is not trusted because the issuer certificate is unknown.
The certificate is only valid for Sample Server Cert

(Error code: sec_error_unknown_issuer)
```

This error message is given because the sample certificate provided with the WebServer distribution has not been issued by a Certificate Authority that the browser recognizes (such as Verisign or GoDaddy). Of course, in this case you are running a server on your local network, so it is okay to add an exception for this certificate in your browser. The exception should be removed once you have generated or obtained your own certificate and private key for the server.

In Firefox, you can provide an exception for the site while testing under:

```
Preferences->Advanced->Encryption->View Certificates->Servers->Add Exception
```

The dialog will ask for the host URL, which in this example is "https://localhost:4433". Enter the name of your host and port, and click "Get Certificate". The certificate is loaded and you have the option to have an exception for this certificate on this host. Click Ok, and you will be able to access the Web server via https.

Increasingly, browsers are making it difficult to accept certificates that are self-signed, or ones where the host name in the certificate doesn't match the host name you are accessing. This adds a layer of security and requires the extra measures described above when testing with sample certificates.

Copyright Information

Trademarks

Copyright

Copy Restrictions

GoAhead Software, Inc.

10900 NE 8th Street Suite 1200 Bellevue, WA 98004 +1 (425) 453-1900 www.goahead.com
info@goahead.com