

# NetTalk 5 - Compiling your own Multi-Site host

Document last updated: 25 May 2010, NetTalk 5 PR 17

---

## Contents

|  |    |
|--|----|
| Requirements:.....                             | 1  |
| Background .....                               | 2  |
| Making your own Host app.....                  | 2  |
| Configuring the HOST Exe .....                 | 3  |
| Connecting to the Host for the first time..... | 4  |
| Registering a Static Site.....                 | 5  |
| Compiling an App into a DLL .....              | 7  |
| Data Table Location (ISAM / TPS etc) .....     | 10 |
| Data Table Location (SQL).....                 | 10 |
| Deploy the DLL .....                           | 11 |
| Registering a Dynamic Site.....                | 11 |
| Updating a Dynamic Site.....                   | 12 |

## Requirements:

Clarion 6 or later

WinEvent

xFiles

GPFReporter (highly recommended for crash protection)

HyperActive (required by GPF Reporter)

SelfService (recommended)

MessageBox (recommended)

File Manager 3 (recommended)

Insight Graphing (recommended)

### Special Notes:

It is very important that the Host program (Host.Exe) and all the DLL's it uses are compiled with the same version of NetTalk.

## Background

Only one program can listen on an IP/Port combination at a time. Since the default port number for web sites is 80, and since IP addresses are in short supply, it's a good idea for sites to be able to share this resource if possible.

With NetTalk 5 this is possible by using the Host example app. The source to this app is shipped so you can modify it, and extend it as you like.

The basic idea is that instead of compiling your NetTalk site as an EXE, you compile it as a DLL. Then you can "register" the DLL with the Host exe (at runtime). Using the URL host name the Exe examines each incoming request and passes it off to the appropriate DLL for processing. If it does not recognize the URL Host name then the web interface to the host itself is displayed.

The URL host name is the first part of the URL address. So for example, in the URL `http://www.capesoft.com/accessories/index.htm` the URL host name is `www.capesoft.com`. The global DNS system might resolve this to the IP address `64.207.29.214`, but the URL itself is passed to the server. The DNS system might also point `www.faswin.com` at the same IP address, and therefore requests for that site will also go through the same host server.

It should be pointed out that within a domain any number of sub-domains are possible. For example `www.capesoft.com`, `news.capesoft.com` and `forums.faswin.com` can all point to the same IP Address, but ultimately can be handled by different DLL's.

The reverse is also true, it's possible to register a DLL against multiple host URL's. `http://capesoft.com`, `http://www.capesoft.com`, and `http://www.capesoft.co.za` all are different host URL's but all go to the same DLL for processing.

This technique does not work for secure sites, because with a secure site the host name is encrypted. Thus the certificate is needed to decrypt the host name, but without the host name it's impossible to know which certificate to use.

The NetTalk Host app is able to load, and unload, DLL's at runtime, without closing the listening IP port. Thus it is possible to update a site's DLL with no down-time. Also, the Session Queue is held by the EXE, so updating a site in this way does not result in dropped sessions.

Because the Server object is in the Host, custom code embedded in the WebServer procedure in your DLL does not run. (The exception to this is the INIT method in the WebServer procedure in the DLL). So any code in your WebServer procedure will need to either be moved to the Host exe, or handled in some alternate way.

## Making your own Host app

1. Make a new development folder.
2. Copy all the app, dct and ico files from the MultiSite6 (59) example
3. Create a Web folder, and copy in the files from `\clarion6\3rdParty\Libsrc\NetWeb\web` or `\clarion7\accessory\libsrc\win\NetWeb\web`
4. Open the app and compile. This makes a program called **HOST.EXE**.

## Configuring the HOST Exe

The Host Exe program has a “Settings” Tab. Before you can begin you need to set these settings to appropriate values.

The screenshot shows the 'Settings' tab of the CapeSoft Website Manager. The 'Server' section includes fields for Login (login), Password (password), Listen On Port (80), Bind To IP (1.2.3.4), and Max Errors (200). The 'Windows Service Settings' section includes Service Name (WebServer) and Service Description (CapeSoft NetTalk Web Server Host). The 'Logging' section includes checkboxes for Log All Pages and Log Spiders (both unchecked), a text field for Log Command (logger), and a text field for Log Extensions (empty). Buttons for 'Save', 'Close', and 'Import IP2Location' are visible at the bottom.

**Login / Password:** The Login that will be used when logging in to manage the Host.

**Listen on Port:** The port number for this server to listen on.

**Bind to IP:** if the server has multiple IP addresses (which it probably does) then (optionally) set this to the IP Address you want this server to listen on.

Note that all the sites registered with this host will share this IP address and port.

**Max Errors:** The number of Errors that will be stored in the Errors table. Old errors are deleted to make way for new ones.

**Service Name:** If you have SelfService in the host app, then you can set the service name here. The service name needs to be unique for each service, and is used when the host program is registered with the Windows Service Manager. This name should contain no spaces.

**Service Description:** The description that will be displayed in the Windows Service Manager.

See the SelfService documentation (<http://www.capesoft.com/docs/SelfService/selfservice.htm>) for more background information on Services in general, and the SelfService functionality in an app.

**Log all Pages:** Tick this on to log all page requests. Warning, this can result in very large logs very quickly.

**Log Command:** The way to turn logging for an individual page on, or off, is to set a parameter for that page. For example, if the log command is set to LOGME then logging for a page (or graphic etc) can be turned on with a url like <http://www.whatever.com/somepage.htm?logme=1> and off with <http://www.whatever.com/somepage.htm?logme=0>.

**Log Extensions:** You can log all pages with a specific extension by adding a space-separated list of extensions to log here.

**Log Spiders:** By default requests by web spiders (Google, Yahoo etc) are not logged. Tick this on if you want to log those requests.

When done click on the **Save** button. Settings are saved in the same folder as the Host.Exe, with the name Settings.Xml.

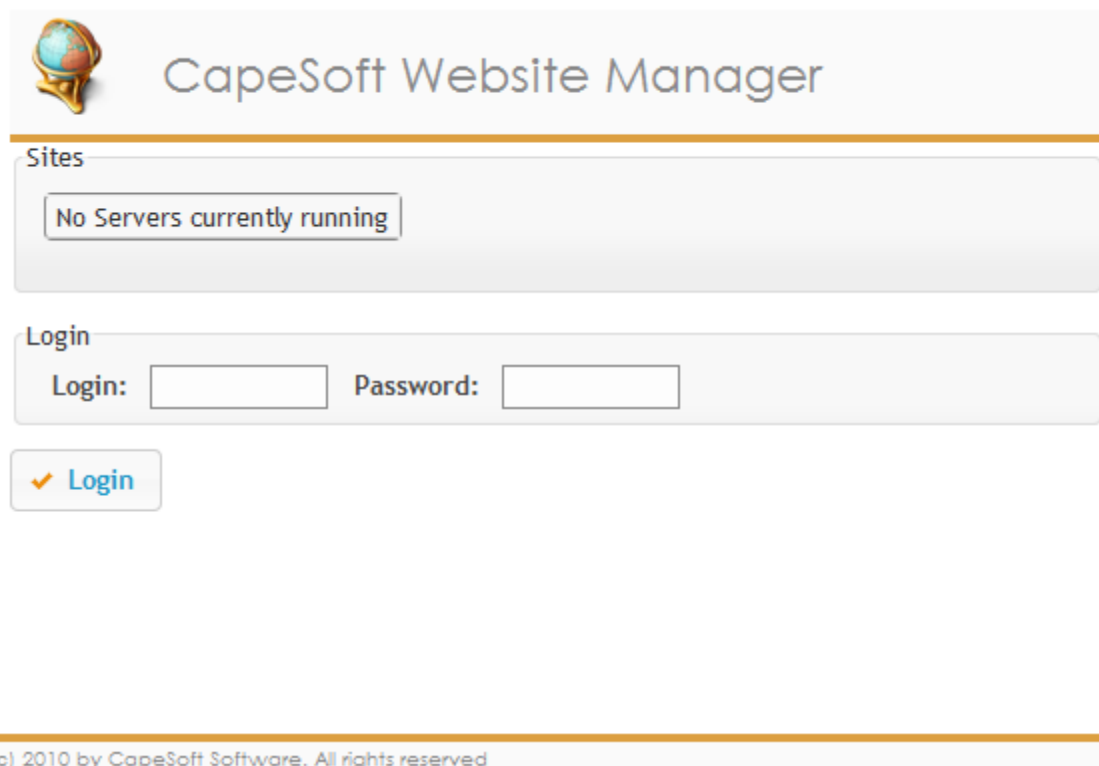
## Connecting to the Host for the first time


In order to use the Host program effectively you will need at least 2 different URL Host names pointing to this computer. One for the host itself, and one for each site that will be registered.

Note that most machines have a number of alternate host URL's already active. For example, on the machine itself, <http://127.0.0.1> and <http://localhost> will both work. (If the server is listening on all interfaces). These would be considered as two different URL hosts.

In the long run you will need a permanent, fixed, DNS entry which is never assigned to anything, so that you can manage the Host exe. For example manage.whatever.com.

Connecting to the server gives you a page that looks something like this;




 CapeSoft Website Manager

Sites

No Servers currently running

Login

Login:  Password:

 Login

© 2010 by CapeSoft Software. All rights reserved

Enter the Login and Password (as they're set on the Settings tab) and click the Login button.

[Manage Servers](#)[Errors Log](#)[Page Analysis](#)[Site Analysis](#)[Referers](#)[Log](#)

## Servers

No Records found

[+ Insert](#)

## Registering a Static Site

Some sites consist purely of static files. These sites do not need a DLL, they can be served directly by the host app. An example of this is the site2 folder in the MultiSite6 example.

Inside the site2 folder are a handful of static html pages. If these pages needed css, or JavaScript files then they would need to be here. In this sense, the folder layout is whatever the site requires it to be.

The static site folder can be anywhere on the hard drive. It does not need to be below the Host folder. The normal web server rules apply though - only files in, or below, that static folder can be served.

Click on the Insert button, under the servers list, to register a new site.

The Form looks something like this;



## Update WebServers

### Host

Host Name:  eg www.capesoft.com - don't prefix with http://

Host Inactive:

### Site Pages

Web Folder:

Default Page:

### DLL (optional)

DLL Path:

DLL Name:

Web Server Procedure:

Save

Cancel

### Host Name

This is the Host part of the URL which will identify this site. Do not prefix this with the http:// part, and don't put any page name in here. This is just for the Host part of the URL. Ultimately the Host Exe will use this name to know that a request belongs to this site.

### Host Inactive

If you want to make the site inactive for any reason, tick this on. If the site is a DLL, and the Host.Exe has a problem with the DLL, then the Host Exe may make the site Inactive, by setting this option.

### Web Folder

The location of the site's web folder anywhere on the disk. A fully qualified path name (c:\something\somewhere\web) is ideal.

### Default Page

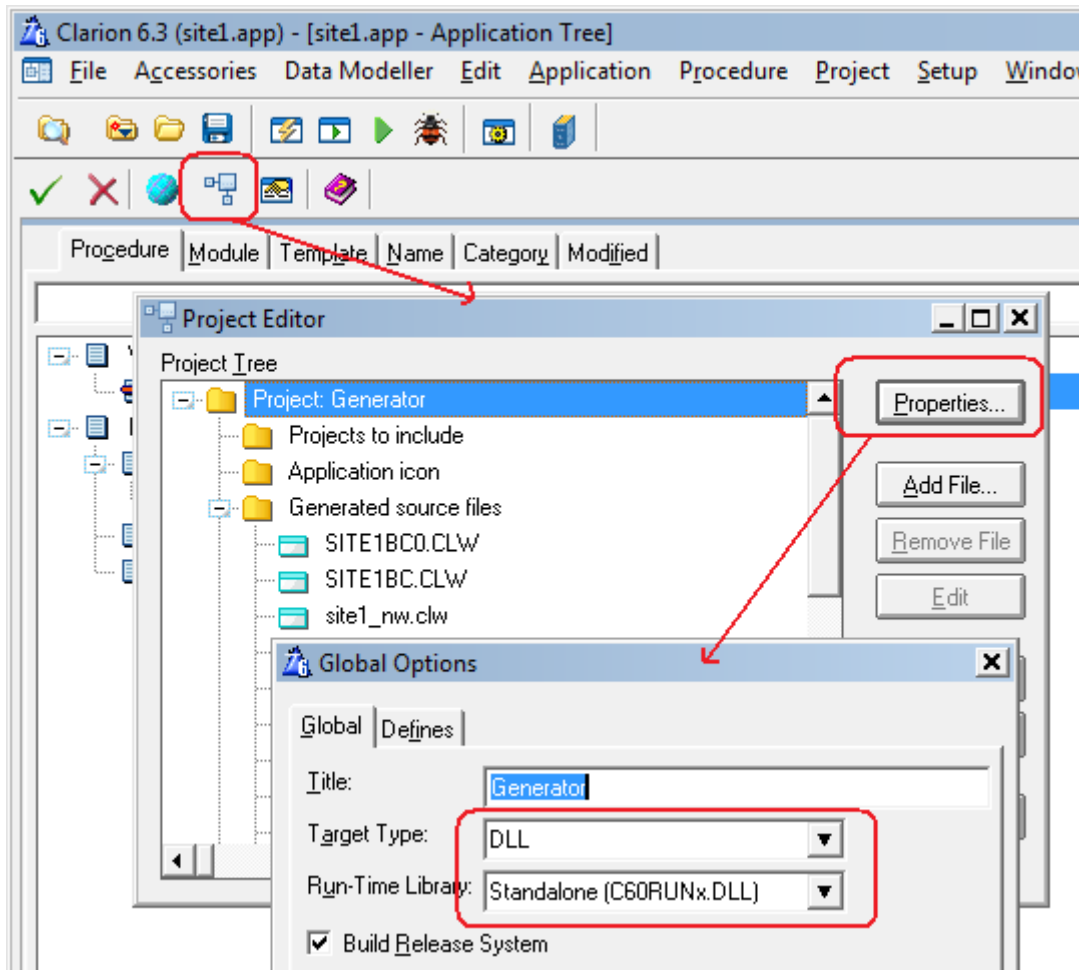
The "first page" of the site – the page that will be called if just the URL is used without a page name.

Since this is a Static site registration, the DLL items are not required, and will be discussed later on.

## Compiling an App into a DLL

In order to register a NetTalk app with the Host, it has to be compiled as a DLL, not an EXE. This is fortunately a very simple thing to do;

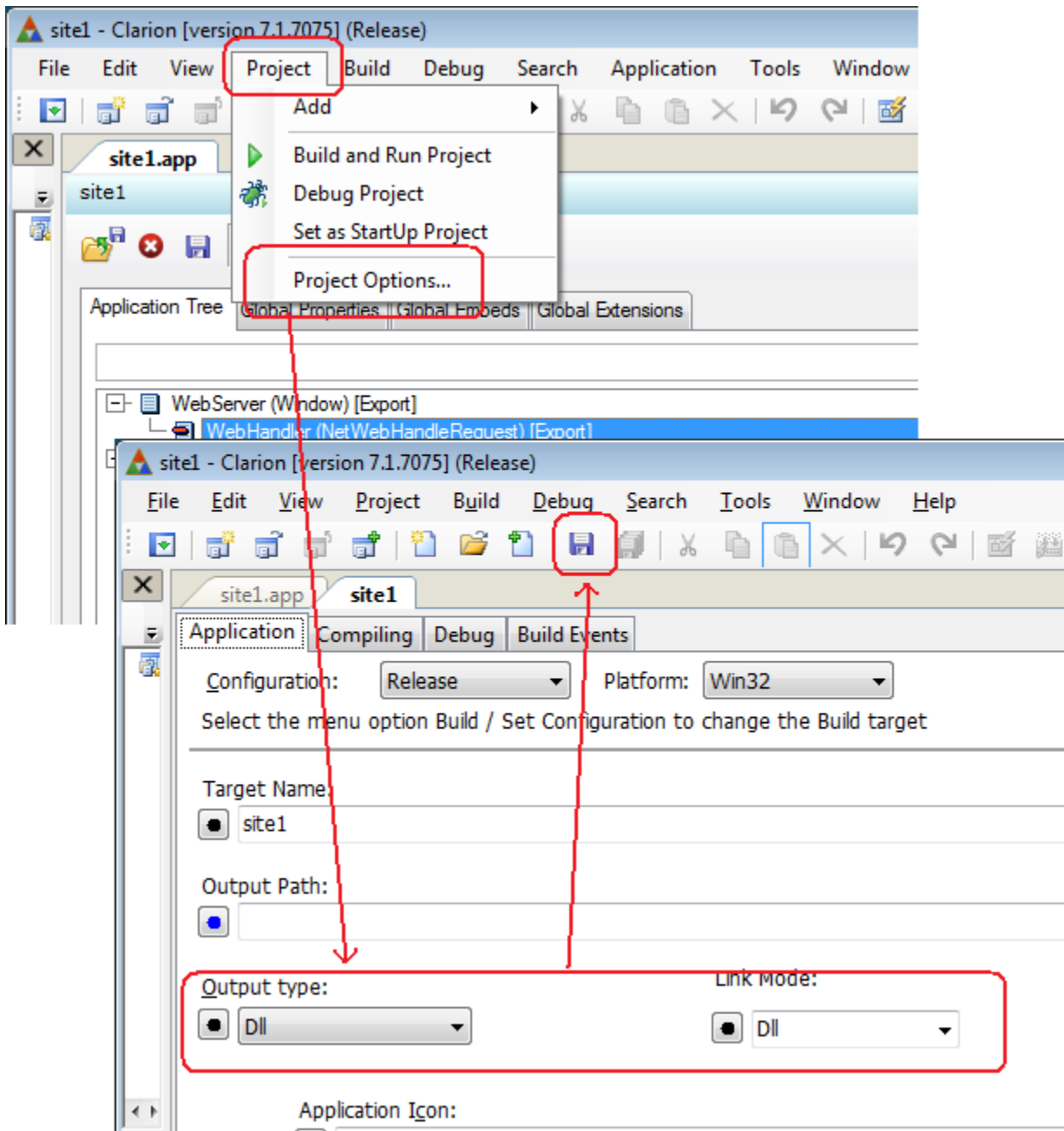
1. Clarion 6: Go to the Project settings for the App and Change the Target Type to DLL. Make sure the Runtime Library is set to Stand Alone.



Or

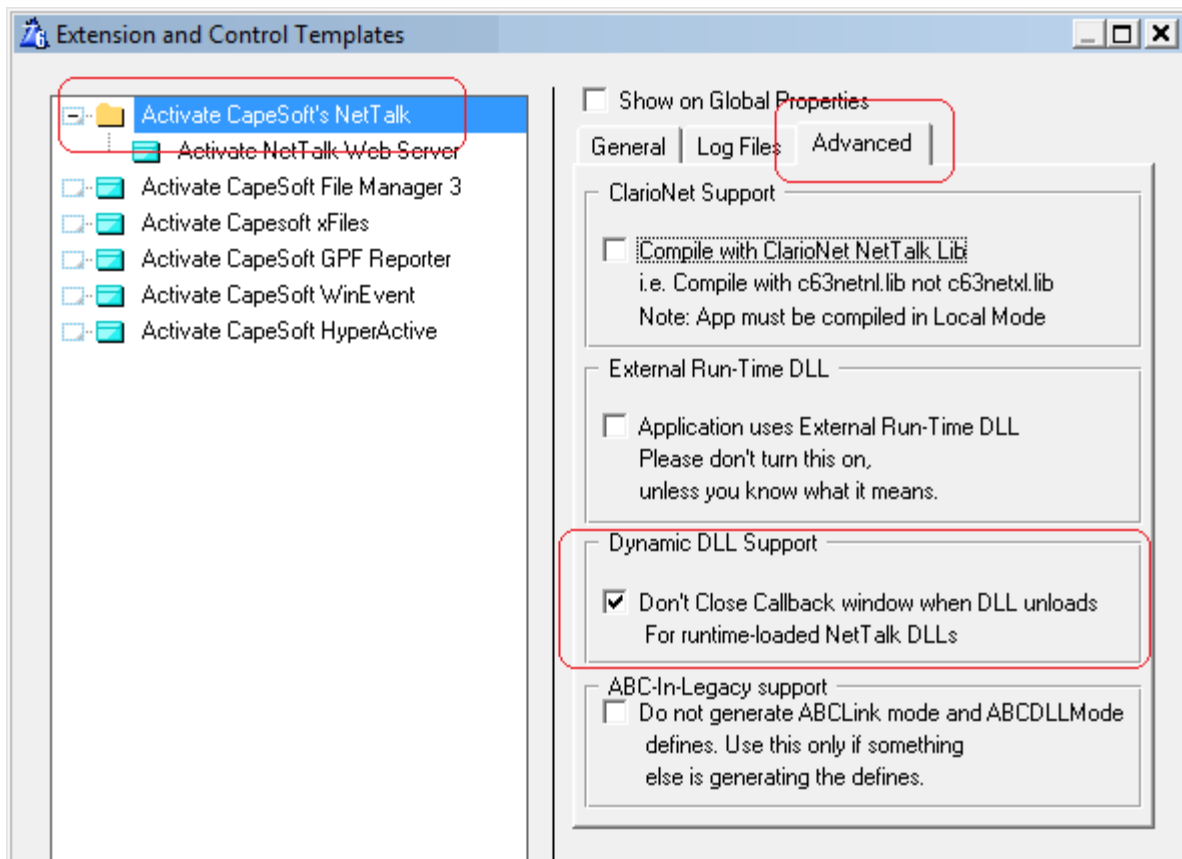
1. Clarion 7: Go to the Project settings for the App and Change the Output Type to DLL. Make sure the Link Mode is set to DLL.

After making changes it's VERY important to click on the SAVE button.



2. Go to the Global NetTalk Extension, to the Advanced Tab and tick on the option for Dynamic DLL Support. This is a really important step. If you fail to do this, and your DLL is deactivated by the host, then the Host will stop listening on its port.





3. If it's not already set, set the Prototype of the WebServer procedure to

```
(<NetWebServer pServer>), name('WebServer')
```

Also set the Parameters List to

```
(<NetWebServer pServer>)
```

4. Make sure the Prototype of the WebHandler procedure is

```
(String p_String), Name('WebHandler')
```

And the Parameters List is

```
(String p_String)
```

**Really useful Tip:** Steps 2,3 and 4 are compatible with compiling the app as an EXE, so if you wish to run a test copy of the site as an Exe, then all you need to do is reverse Step 1, and set the options back to Exe.

## Data Table Location (ISAM / TPS etc)

The main practical difference between running as an Exe, or a DLL, is the location of the data tables. With an Exe the data tables are located by default in the application folder. Where data tables are elsewhere simple techniques like SETPATH or Prop:Datapath are commonly used. For purposes of using the App as an Exe, or a DLL, a slightly more advanced technique is required.

1. Each table needs a distinct Full Pathname variable set in the dictionary. For example;

Customers has Full Pathname set to !glo:customersFileName

Invoices has Full Pathname set to !glo:invoicesfilename

and so on.

2. Recommended, but not essential, is that these variables are threaded. By making them threaded you allow different users to use different data sets, from the same server. If the server will only have one data set then the variables can be non-threaded.

3. If the variables are unthreaded you can set them in the WebServer procedure, in the **NetTalk Object After s\_web assigned** embed point. At this point the variable

`s_web._SitesQueue.defaults.appPath` contains the location of the DLL. And you can (and probably should) use it in assigning the other table name variables. For example

```
glo:CustomersFileName = clip(s_web._SitesQueue.defaults.appPath) &
'customers.tps'
```

4. If the variables are threaded, then you can set them in the WebHandler procedure ProcessLink method, before the parent call. At this point the variable `self.site.AppPath` contains the location of the DLL. For example;

```
Glo:CustomersFileName = clip(self.site.AppPath) & 'customers.tps'
```

The session is also active at this point, so you can use `GetSessionValue` to retrieve information you've stored about the current session. This can be useful in linking specific users to specific data sets.

**Tip:** It is likely that you will, at some point, wish to START threads from inside your app, which are not started by a request from the browser. So moving all this code to a separate procedure, and then simply calling the procedure from the WebHandler, and any other procedure that is the first procedure in a thread, reduces the need to repeat this code.

## Data Table Location (SQL)

Setting the data table location (ie Owner variable) for SQL tables in a web app is pretty much the same as setting it in a normal Clarion windows app. Use the same embed points as described in the section above (Data Table Location (ISAM / TPS) ) and set the connection string and table names appropriately.

Note that for SQL it is not necessary to have a separate variable for each table name, since the connection string is common.

## Deploy the DLL

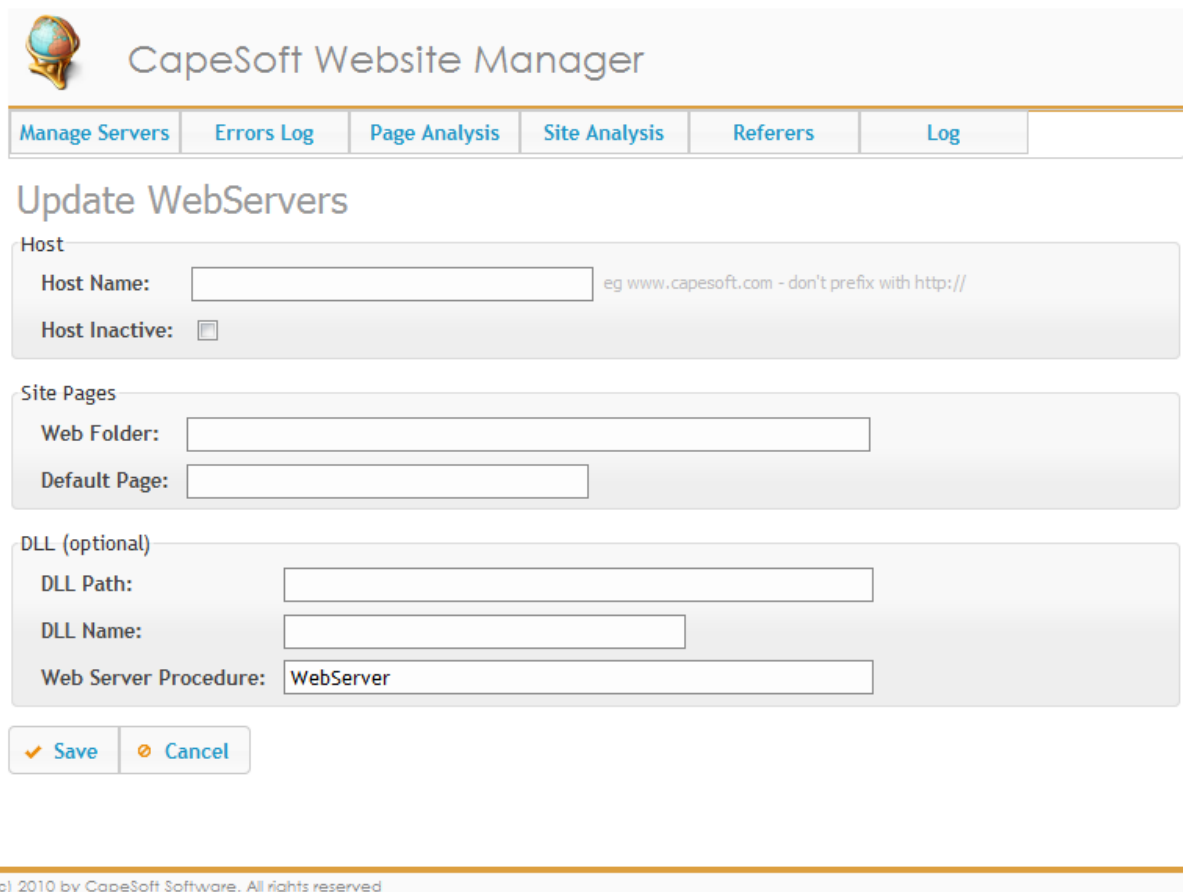
Because the app is compiled in DLL Mode, it will need the Clarion Runtime DLL's for the file drivers, RTL and so on. Any DLL's required by this DLL should be in the HOST.EXE folder.

Although DLL's can share a web folder (the location of the web folder will be set in a moment) it is likely that each DLL will have its own web folder, containing images, styles and scripts unique to that site.

**Special Note:** It is very important that the Host program (Host.Exe) and all the DLL's it uses are compiled with the same version of NetTalk.

## Registering a Dynamic Site

For the most part registering a dynamic site is the same as registering a static site. There are only three extra fields to fill in.



The screenshot shows the 'Update WebServers' dialog box in CapeSoft Website Manager. The dialog has a title bar with the CapeSoft logo and the text 'CapeSoft Website Manager'. Below the title bar is a navigation menu with buttons for 'Manage Servers', 'Errors Log', 'Page Analysis', 'Site Analysis', 'Referers', and 'Log'. The main content area is titled 'Update WebServers' and contains three sections: 'Host', 'Site Pages', and 'DLL (optional)'. The 'Host' section has a 'Host Name' text box with a placeholder 'eg www.capesoft.com - don't prefix with http://' and a 'Host Inactive' checkbox. The 'Site Pages' section has a 'Web Folder' text box and a 'Default Page' text box. The 'DLL (optional)' section has a 'DLL Path' text box, a 'DLL Name' text box, and a 'Web Server Procedure' text box with 'WebServer' selected. At the bottom of the dialog are 'Save' and 'Cancel' buttons. A footer at the bottom of the window reads '(c) 2010 by CapeSoft Software. All rights reserved'.

### Site Pages

The Web Folder, and Default Page options if set here, override the settings in the DLL.

### DLL Path

The fully qualified path name to the DLL. Ie something like  
c:\here\there\everywhere or \\machine\here\there\everywhere  
rather than  
..\..\here\there\everywhere

### DLL Name

The name of the DLL

### **Web Server Procedure**

The name of the procedure in the DLL containing the NetTalk WebServer object. This is almost always called WebServer.

## **Updating a Dynamic Site**

To update a Dynamic site the host has to first unload the DLL.

Then the new DLL is placed in the correct folder, and the host reactivates the DLL.

For small, incremental changes, users should not even notice the downtime (if it is done fast enough). Sessions are stored in the Host Exe, not the DLL, so logged in users and sessions are not lost during this process.

Always remember though to update any support files (images, .js, .css and so on) at the same time.

[end]