TECHNICAL NOTE TNOI33

Title: Using the Web Server custom web site

Product(s): G3, MC, and DSP

ABSTRACT

Crimson 2.0 Web Server provides embedded tools for application remote viewing and control. No extra development is required to be able to remotely access, view and modify information on the process or machine using the embedded Virtual HMI. However, for custom purposes, the G3 series, Modular Controller Master and Data Station Plus SX or GT provide the ability to develop custom web pages.

INTRODUCTION

The following technical note will present how to create a custom web site on a G3 and what tools are available to the developer to view and/or manipulate data. Development on a Modular Controller Master and Data Station Plus SX or GT is exactly the same. We will use the following steps to develop a custom web site:

1. Installation of a CompactFlash card on the unit.
2. Access to the CompactFlash using the USB connection.
3. Creation of the WEB folder and default.htm page.
4. Review an example of how to view data on a custom web page.
5. Review an example of how to change data on a custom web page.

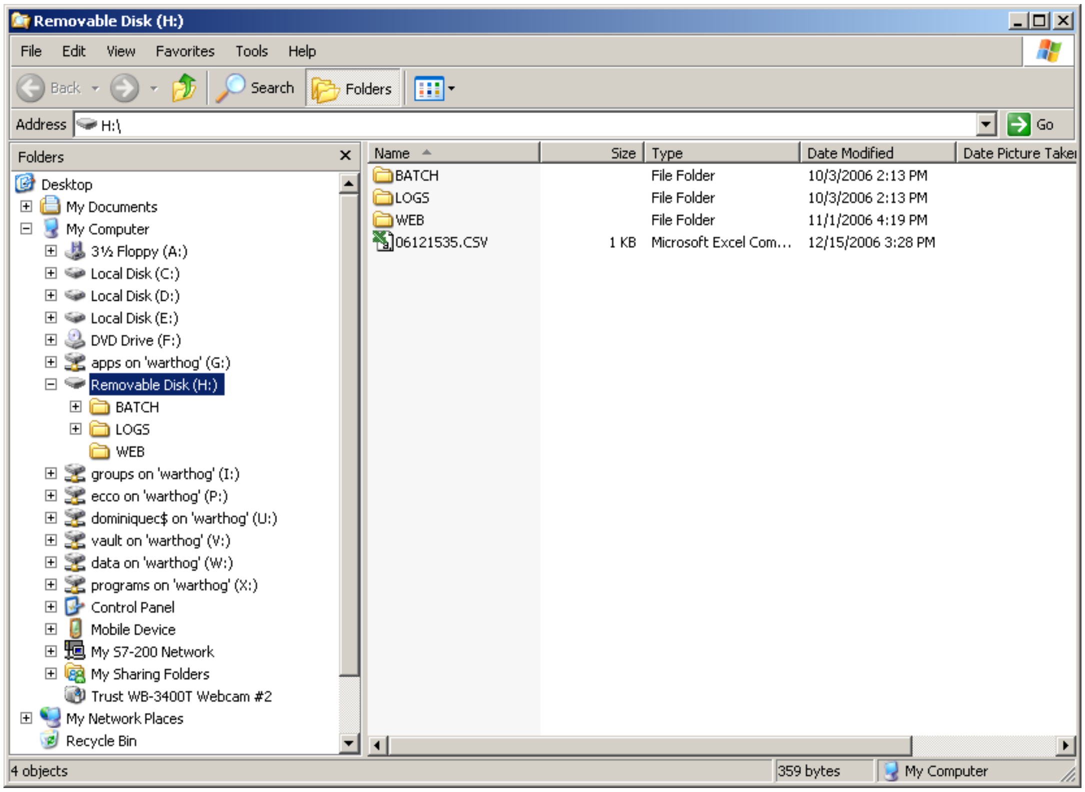
Although the HTML code will be explained or commented through this document, a basic understanding is required to create a proper web site.

NOTE: These features only work with Crimson 2.0 build 299 or above and is thus only available to Windows 2000 or above users. Please update your version of Crimson 2.0 to the latest available online on www.redlion.net

PREPARING THE COMPACTFLASH CARD

A CompactFlash card is required to use the custom web site capability for all the units. Please insert the card in the available slot and double check that the orange LED is lit and solid. If the LED blinks slowly, please refer to the Crimson 2.0 manual for troubleshooting.

Once the card is inserted, connect the USB cable between your G3 and PC. If the required drivers were not installed, your computer should prompt you to install them. Refer to the technical note G3, DSP and Modular Controller -- USB Installation Guide for details on the installation procedure. Once the drivers are installed, the CompactFlash card can be accessed from Windows Explorer as shown in Figure 1.



Name	Size	Type	Date Modified	Date Picture Take
BATCH		File Folder	10/3/2006 2:13 PM	
LOGS		File Folder	10/3/2006 2:13 PM	
WEB		File Folder	11/1/2006 4:19 PM	
06121535.CSV	1 KB	Microsoft Excel Com...	12/15/2006 3:28 PM	

Figure 1

Create a folder called "WEB" on the root of the CompactFlash card. This folder will contain all the custom web site files, i.e. html pages and pictures. Subfolders can be created under the WEB folder if required.

CREATING A WEB PAGE AND ACCESSING DATA

When Internet Explorer will access the custom web site, the first page it will load will be "default.htm". Therefore, the welcome page or main page of the custom web site should be called "default.htm" and created just under the folder WEB.

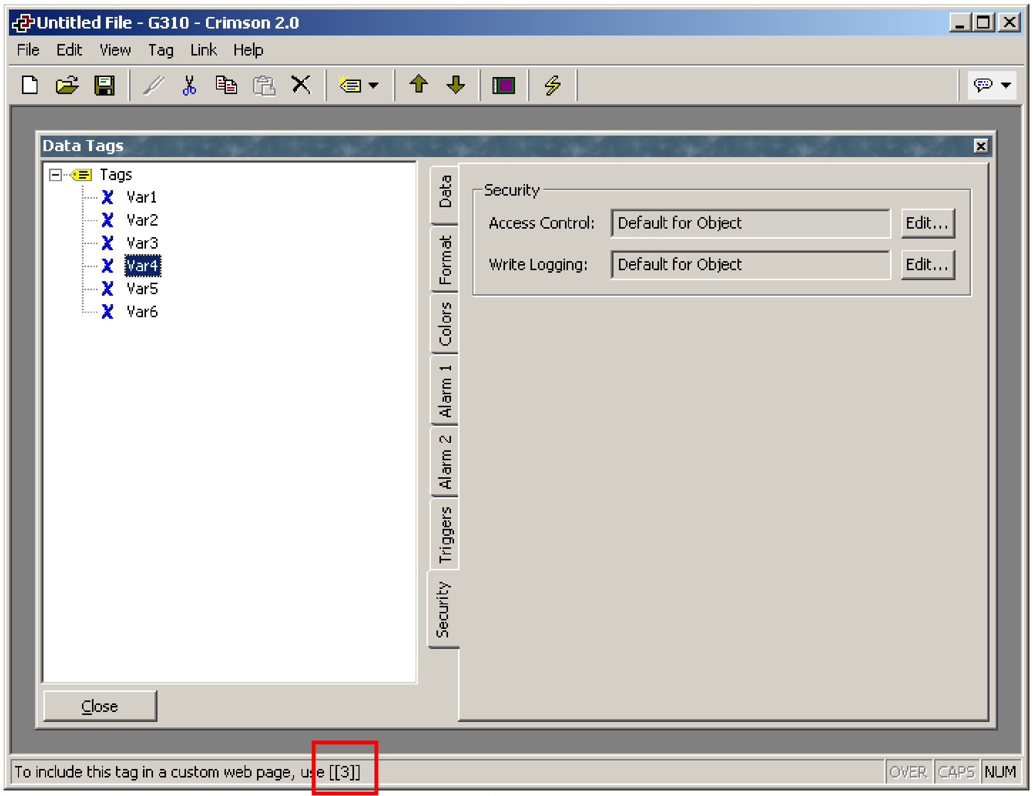
You can then generate or create pages under any HTML or web page editor such as Front Page, Mozilla or Notepad. In this tutorial, the html code will be displayed as shown in Notepad.

The sample code below is a very basic "default.htm" that will display a sentence and the value of the tag with index 0. Code comments are displayed in green.

As shown in the code, the index is entered in the form **[[x]]** where **x** is the tag's index number. The Web Server will then replace this code with the value of the corresponding tag. Moreover, this value will be formatted as per the tag format in Crimson 2.0. A flag will therefore show the on or off text and decimal points will be displayed.

```
<html>

 <!-- Head information for the page including page title -->
 <head>
  <title>My first page</title>       <!-- Your page title -->
 </head>

 <body color=#FFFFFF>                 <!-- Start of your page body -->

  <!-- This code displays the current tag value for index 0
  [[0]] will be replaced by the tag value a the time the page is loaded -->
  The value of the tag with index 0 is [[0]].

 </body>                              <!-- End of your page body -->

</html>
```

The tag index number is defined by Crimson and is available in the tag window status bar as shown in Figure 2. The index displayed in the status bar corresponds to the tag selected in the tree. Here Var4 has index 3.



Security

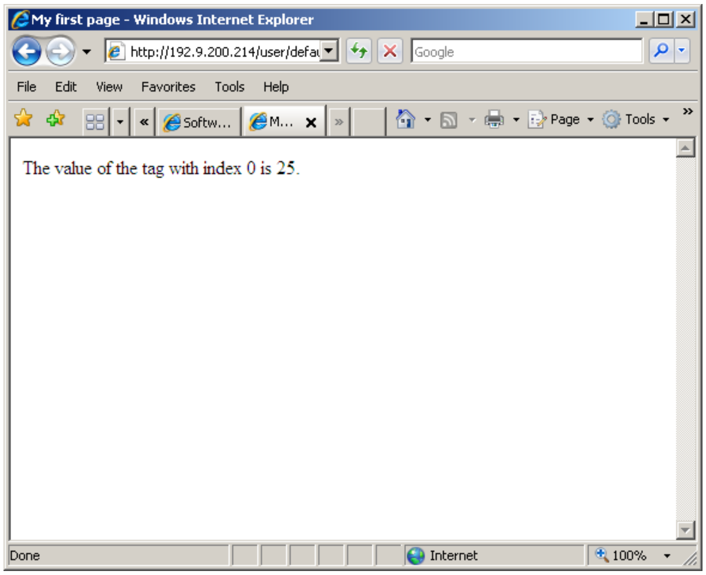
Access Control: Default for Object

Write Logging: Default for Object

To include this tag in a custom web page, use [[3]]

Figure 2

The resulting custom web page accessed from Windows Internet Explorer is shown in Figure 3.



My first page - Windows Internet Explorer

http://192.9.200.214/user/defau

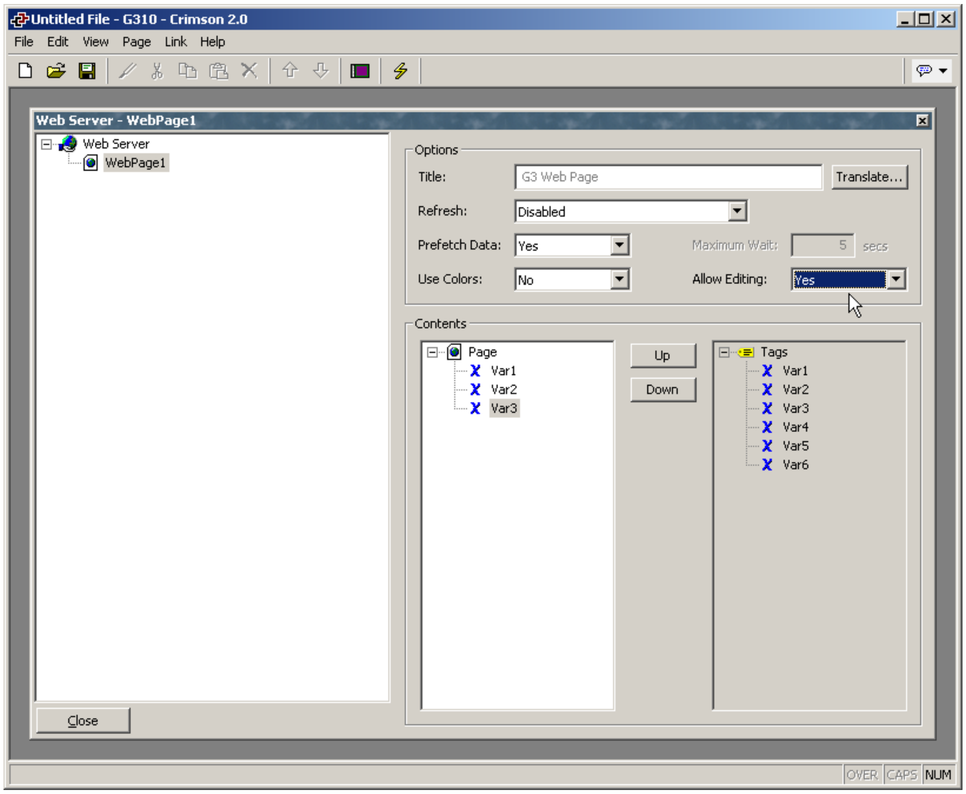
The value of the tag with index 0 is 25.

Figure 3

CHANGING DATA FROM A CUSTOM WEB PAGE

This section is a bit more complicated for those who do not know HTML and also implies some Crimson 2.0 set up for security reasons.

Crimson also offers an easier way to give data access by using Web Pages created from the Web server module as shown in Figure 4. Each page can display a set of tags and edit them if Allow Editing is set to "Yes".



Untitled File - G310 - Crimson 2.0

Web Server - WebPage1

Options

Title: G3 Web Page

Refresh: Disabled

Prefetch Data: Yes

Maximum Wait: 5 secs

Use Colors: No

Allow Editing: Yes

Contents

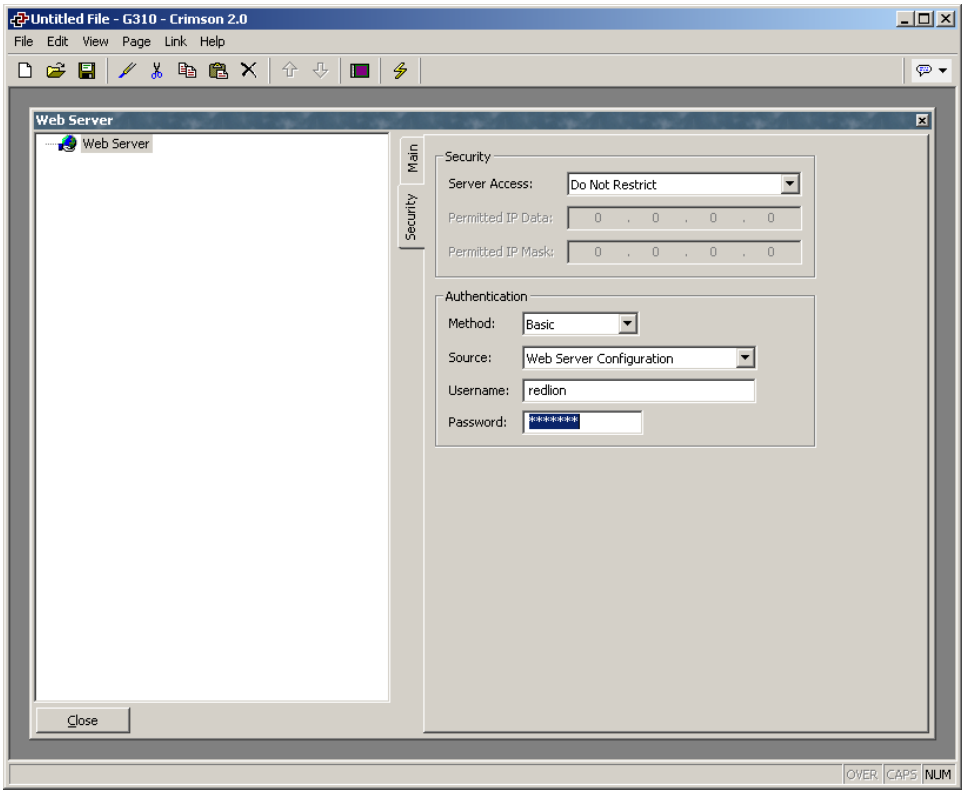
Page: Var1, Var2, Var3

Tags: Var1, Var2, Var3, Var4, Var5, Var6

Figure 4

Setting up Crimson to accept data from custom web pages.

Crimson needs to be set up to accept data coming from the custom web site. First of all, it is strongly advised that the security is activated on the web server as shown in Figure 5.



Security

Server Access: Do Not Restrict

Permitted IP Data: 0 . 0 . 0 . 0

Permitted IP Mask: 0 . 0 . 0 . 0

Authentication

Method: Basic

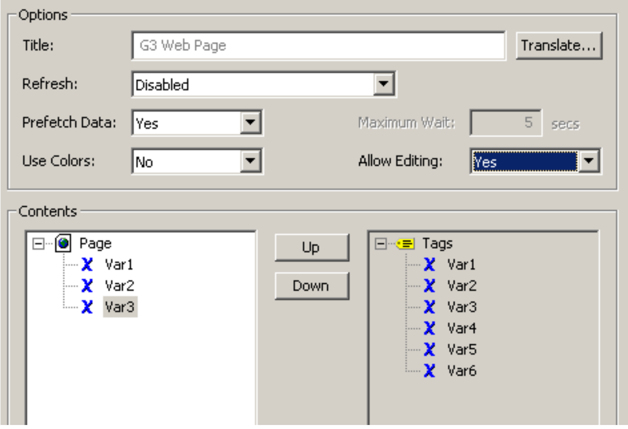
Source: Web Server Configuration

Username: redlion

Password: *******

Figure 5

Then, create a data web page from the Web Server Main tab and insert the variables that should be editable in the custom pages. Change Allow Editing to "Yes" as shown in Figure 6.



Options

Title: G3 Web Page

Refresh: Disabled

Prefetch Data: Yes

Maximum Wait: 5 secs

Use Colors: No

Allow Editing: Yes

Contents

Page
- Var1
- Var2
- Var3

Tags
- Var1
- Var2
- Var3
- Var4
- Var5
- Var6

Figure 6

This informs Crimson that the above selected variables can be changed from web pages.

NOTE: Security can be achieved from the Web Server configuration, as shown above in Figure 5, or from the Security Manager. If the security manager is used, one of the users has to possess the Web Server permissions.

Writing the HTML code.

Data entry is done using an html form and a postscript to transfer the information to the G3. A form is a single or a group of data entry fields in an html page as shown in Figure 7. Upon entry, the web browser will send the data to the Red Lion unit using the POST function. The unit will reply with a confirmation page for success or failure. To disable this confirmation page, please see the section “Required information by the unit” below.

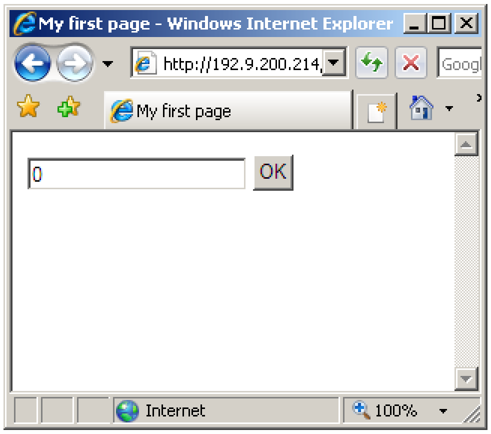


Figure 7

Single data entry

A data entry form as shown above uses the following HTML code.

For numeric tags

```
<FORM name=form1 method=post action=script autocomplete=off>
 <INPUT type=text name=data value=0>               <!-- Data Entry field -->
 <INPUT type=submit name=submit value=OK>          <!-- Button OK -->
</FORM>
```

For ASCII string tags

```
<FORM name=form1 method=post action=script autocomplete=off>
 <INPUT type=text name=data maxlength=8 value=0>   <!-- Data Entry field -->
 <INPUT type=submit name=submit value=OK>          <!-- Button OK -->
</FORM>
```

Forms parameters

Values in red are the parameters changed to target the different information.

- *form1* – Form name, usually different for each form on a single html page.
- *script* – The script to launch when the OK button is pressed.
- *data* – the html tag the data entry is referencing.
- *0* – The default value to display in the data entry field.

Values in blue are the extra parameters required for ASCII string data entry.

- *maxlength* – Maximum number of characters. It should be identical to the number defined in the targeted tag format in Crimson 2.

Required Information by the Unit

In the case of the G3, the script transferring data from the web page to the HMI is “write-tags” and is located in “**`/scripts/write-tags`**” in the unit server. In order to transfer the correct information, extra fields are required in the form but will be hidden to the user.

The following extra information has to be sent to the script using html **`INPUT`** tags:

- Parameter tagX – defines a tag name for the index given in value. Here Tag1 for index 0.

  ```
  <input type=hidden name=tag1 value=0>
  ```

- Parameter back – defines the page the script redirects you after the data entry. Here default.htm.

  ```
  <input type=hidden name=back value=/user/default.htm>
  ```

- Parameter page – to be left with default value 1

  ```
  <input type=hidden name=page value=1>
  ```

- Optional parameter silent – removes confirmation page and replies HTTP 204 “No Content” for a non-zero value. Here no confirmation page will be sent. The parameter default is 0.

  ```
  <input type=hidden name=silent value=1>
  ```

A complete form to send the value of one tag (not an ASCII string) will thus be as follow. Here changing the tag with index 0 and confirmation page active:

```
<FORM name=form1 method=post action=/scripts/write-tags autocomplete=off>
 <INPUT type=hidden name=tag1 value=0>      <!-- Define tag index 0 -->
 <INPUT type=hidden name=back value=/user/default.htm>
 <INPUT type=hidden name=page value=1>
 <INPUT type=text name=data1 value=[[0]]>   <!-- Data Entry field -->
 <INPUT type=submit name=submit value=OK>
</FORM>
```

If the form is designed for only one data entry item, each form can use the couple **`tag1`**, **`data1`** to reference the variable.

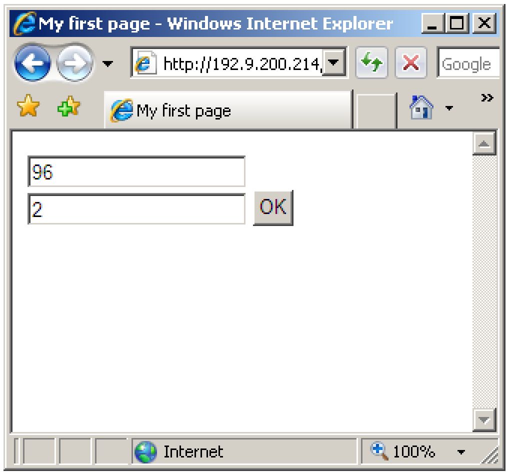
NOTE: These tag names have no relation with Crimson tag names and are used by the postscript only. Their name cannot be changed, only the number at the end, for example tag2, data2.

Please see Appendix A for the complete formatted html code to send two individual tags to a G3.

Multiple data entry simultaneously

In the case the user wishes to enter multiple tags at once, the form has to be modified to accept multiple data. A form including two tags is shown in figure 8.

Note that clicking on OK in this form will send both values simultaneously to the postscript. For this reason, it is advised to display the current value in the field so clicking OK does not change it.



96

2

Figure. 8.

To access multiple tags in a form, the following two html INPUT tags have to be inserted for each new tag included in the form, one hidden to define the tag index and one for the data entry field.

```
<INPUT type=hidden name=tag1 value=0>

<INPUT type=text name=data1 value=[[0]]>
```

The names **`tag1`** and **`data1`** represent a couple. If multiple tags entries are required within a form, the second tag entry will use a hidden input **`tag2`** and a data entry name **`data2`**. A third one would use **`tag3`**, **`data3`** and so on…

The form code for two tags would look like this:

```
<FORM name=form1 method=post action=/scripts/write-tags autocomplete=off>
 <INPUT type=hidden name=tag1 value=0>      <!-- Define tag index 0 -->
 <INPUT type=hidden name=tag2 value=1>      <!-- Define tag index 1 -->
 <INPUT type=hidden name=back value=/user/default.htm>
 <INPUT type=hidden name=page value=1>
 <INPUT type=text name=data1 value=[[0]]>   <!-- Data Entry field for 0 -->
 <INPUT type=text name=data2 value=[[1]]>   <!-- Data Entry field for 1-->
 <INPUT type=submit name=submit value=OK>
</FORM>
```

NOTE: These tag names have no relation with Crimson tag names and are used by the postscript only. Their name cannot be changed, only the number at the end, for example tag2, data2.

Please see Appendix B for the complete formatted html code to send multiple tags to a G3.

APPENDIX A

The following code includes html format tags. Comments are in green.

Copy/Pasting this code in an empty default.htm will work as long as Crimson’s web server is set up correctly.

```
<html>
 <head>
  <title>Custom Edit</title>
 </head>
 <body color=#FFFFFF>

<!-- Text on top of the page -->
 <p><font face=tahoma size=4 color=#FF0000>
  <b>Custom Web Site: A page where data tags value can be changed </b>
 </font></p>

<!—INDEX 0 -->
 <font face=tahoma size="2">
  <b>Tag with Index 0 in Crimson 2.0 Data Tags.</b><br><br>

<!-- Display the current value for index 0 -->
  Current Value:&nbsp; [[0]]
 </font>

<!-- Form to enter the value for index 0 and submit to the HMI -->
 <form name=form1 method=post action=/scripts/write-tags autocomplete=off>
   <font face=tahoma size="2">
     <input type=hidden name=tag1 value=0>
     <input type=hidden name=back value=/user/default.htm>
     <input type=hidden name=page value=1>
     New Value:&nbsp;
     <input type=text name=data1 value="[[0]]" style="font-face: tahoma;
font-size: 12">
     <input type=submit name=submit value=&nbsp;OK&nbsp; style="font-face:
tahoma; font-size: 12">
   </font>
</form>
<br>

<!-- INDEX 1-->
<font face=tahoma size="2">
<b>Tag with Index 1 in Crimson 2.0 Data Tags.</b><br><br>

<!-- Display the current value for index 1 -->
Current Value:&nbsp; [[1]]
</font>
```

Code continues next page…

```
<!-- Form to enter the value for index 1 and submit to the HMI -->
<form name=form2 method=post action=/scripts/write-tags autocomplete=off>
   <font face=tahoma size="2">
     <input type=hidden name=tag1 value=1>
     <input type=hidden name=back value=/user/default.htm>
     <input type=hidden name=page value=1>
     New Value:&nbsp;
     <input type=text name=data1 value="[[1]]" style="font-face: tahoma;
font-size: 12">
     <input type=submit name=submit value=&nbsp;OK&nbsp; style="font-face:
tahoma; font-size: 12">
   </font>
</form>
<br>

<!-- Link to go back to web server home page -->
<font face=tahoma size="2">
<a href="/">Web Server Home</a><br>
</font>

<p><font face=tahoma size=1>© 1993-2006 <a href=http://www.redlion.net>Red
Lion Controls</a></font></p>

</body>
</html>
```

APPENDIX B

The following code includes html format tags. Comments are in green.

Copy/Pasting this code in an empty default.htm will work as long as Crimson's web server is set up correctly.

```
<html>
 <head>
  <title>Custom Edit</title>
 </head>
 <body color=#FFFFFF>

<!-- Text on top of the page -->
  <p><font face=tahoma size=4 color=#FF0000><b>
Custom Web Site: A page where data tags value can be changed in groups
  </b></font></p>

<!-- This code displays the current tags values for indexes 0 and 1 -->
  <font face=tahoma size="2">
   <b>Change a group of tags.</b>
   <br><br>
   Current Value:&nbsp; [[0]],&nbsp;[[1]]
  </font>

<!-- Form to enter the value for index 0 and 1 and submit to the HMI -->
  <form name=form1 method=post action=/scripts/write-tags autocomplete=off>
    <font face=tahoma size="2">
      <input type=hidden name=tag1 value=0>
      <input type=hidden name=tag2 value=1>
      <input type=hidden name=back value=/user/default.htm>
      <input type=hidden name=page value=1>
      Tag with index 0: <input type=text name=data1 value="[[0]]" size=4
style="font-face: tahoma; font-size: 12"><br>
      Tag with index 1: <input type=text name=data2 value="[[1]]" size=4
style="font-face: tahoma; font-size: 12"><br>
      <br>
      <input type=submit name=submit value=&nbsp;Valid&nbsp; style="font-
face: tahoma; font-size: 12">
    </font>
  </form>
  <br>

<!-- Link to go back to web server home page -->
  <font face=tahoma size="2">
   <a href="/">Web Server Home</a><br>
  </font>

  <p><font face=tahoma size=1>© 1993-2006 <a href=http://www.redlion.net>Red
Lion Controls</a></font></p>

 </body>
</html>
```