MATLAB Web Server

For Use with MATLAB®

Computation

Visualization

Programming

MATLAB Web Server
Version 1
How to Contact The MathWorks:

- www.mathworks.com  Web
- comp.soft-sys.matlab  Newsgroup
- support@mathworks.com  Technical support
- suggest@mathworks.com  Product enhancement suggestions
- bugs@mathworks.com  Bug reports
- doc@mathworks.com  Documentation error reports
- service@mathworks.com  Order status, license renewals, passcodes
- info@mathworks.com  Sales, pricing, and general information

- 508-647-7000  Phone
- 508-647-7001  Fax
- The MathWorks, Inc.  Mail
  3 Apple Hill Drive
  Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

MATLAB Web Server
© COPYRIGHT 1999 - 2001 by The MathWorks, Inc.
The software described in this document is furnished under a license agreement. The software may be used
or copied only under the terms of the license agreement. No part of this manual may be photocopied or repro-
duced in any form without prior written consent from The MathWorks, Inc.
FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by
or for the federal government of the United States. By accepting delivery of the Program, the government
hereby agrees that this software qualifies as "commercial" computer software within the meaning of FAR
252.227-7014. The terms and conditions of The MathWorks, Inc. Software License Agreement shall pertain
to the government's use and disclosure of the Program and Documentation, and shall supersede any
conflicting contractual terms or conditions. If this license fails to meet the government's minimum needs or
is inconsistent in any respect with federal procurement law, the government agrees to return the Program
and Documentation, unused, to MathWorks.
MATLAB, Simulink, Stateflow, Handle Graphics, and Real-Time Workshop are registered trademarks, and
Target Language Compiler is a trademark of The MathWorks, Inc.
Other product or brand names are trademarks or registered trademarks of their respective holders.

Printing History:  January 1999  First printing  New for Version 1.0 (Release 11)
September 2000  Revised for Version 1.2 (Release 12) Online only
January 2001  Reprinted with minor changes
June 2001  Minor changes (online only)
Contents

Preface

About This Book ................................................. vi
Organization of the Document .............................. vi

Typographical Conventions ................................. vii

Related Products ............................................. viii

MATLAB on the Web

1

Introduction .................................................... 1-2
MATLAB Web Server Environment ...................... 1-2
Building MATLAB Web Server Applications .......... 1-3

Product Requirements ....................................... 1-5
Web Requirements ........................................... 1-5

Installation ..................................................... 1-6
Availability ....................................................... 1-6
Installation Procedure ...................................... 1-6
General Post-Installation Procedures ................. 1-6
Solaris/Linux Post-Installation Procedures .......... 1-8
Windows NT Post-Installation Procedures .......... 1-11

Graphics Display ............................................. 1-13
Downloading and Installing VNC ......................... 1-13
Downloading and Installing Perl ......................... 1-14
Starting and Stopping VNC ............................... 1-14
Using VNC with the MATLAB Web Server ............ 1-15
Getting Started

Introduction ...................................................... 2-2
Templates ......................................................... 2-2

Creating Input Documents ................................. 2-4
Input Template .................................................. 2-4

Creating MATLAB Web Server M-Files ............... 2-7
M-File Template ................................................ 2-7

Creating Output Documents ............................... 2-10
Output Template ............................................... 2-10

Debugging Your Application ............................. 2-13
Debugging Procedure ....................................... 2-13

Additional Application Examples ...................... 2-15
Data Display ................................................... 2-15
MATLAB Graphics .......................................... 2-16
Stock Price Simulation ....................................... 2-20

Inside the MATLAB Web Server

MATLAB Web Server Components ...................... 3-2
File Locations .................................................. 3-4

Understanding matlabserver ............................ 3-5
matlabserver.conf .......................................... 3-5
Using matlabserver ......................................... 3-7
Returning Results via the Web ......................... 3-10
Function Summary ........................................... 4-2
  htmlrep ...................................................... 4-3
  matweb ...................................................... 4-6
  wscleanup .................................................. 4-7
  wsprintjpeg ............................................... 4-8
  wssetfield ............................................... 4-9

Directory Structure

Troubleshooting Web Server

  General Troubleshooting ................................. B-2
  Additional Troubleshooting for Windows NT .......... B-4
  Additional Troubleshooting for Solaris and Linux ... B-6

Selected Bibliography
Preface

About This Book ........................................ vi
Organization of the Document ........................ vi

Typographical Conventions .............................. vii

Related Products ....................................... viii
About This Book

This book describes the MATLAB® Web Server. The MATLAB Web Server lets you deploy MATLAB based applications over a network using standard Web technology.

Organization of the Document

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 1</td>
<td>Chapter 1, “MATLAB on the Web” provides basic information about how to develop MATLAB applications in a Web environment. Discusses how to install the MATLAB Web Server. Also describes how to obtain and install two related products: Virtual Network Computing (VNC) and Perl.</td>
</tr>
<tr>
<td>Chapter 2</td>
<td>Chapter 2, “Getting Started” illustrates the creation of MATLAB Web applications with several demonstration programs, and provides templates to simplify the creation of user applications.</td>
</tr>
<tr>
<td>Chapter 3</td>
<td>Chapter 3, “Inside the MATLAB Web Server” discusses the various components of the MATLAB Web Server and provides needed configuration information.</td>
</tr>
<tr>
<td>Chapter 4</td>
<td>Chapter 4, “Reference” describes the functions used to create and manipulate MATLAB Web Server applications.</td>
</tr>
<tr>
<td>Appendix B</td>
<td>Appendix B, “Troubleshooting Web Server” discusses solving problems and responding to error messages.</td>
</tr>
</tbody>
</table>
## Typographical Conventions

This manual uses some or all of these conventions.

<table>
<thead>
<tr>
<th>Item</th>
<th>Convention Used</th>
<th>Example</th>
</tr>
</thead>
</table>
| Example code | Monospace font | To assign the value 5 to A, enter  
A = 5 |
| Function names/syntax | Monospace font | The cos function finds the cosine of each array element. Syntax line example is MLGetVar ML_var_name |
| Keys | Boldface with an initial capital letter | Press the Return key. |
| Literal strings (in syntax descriptions in reference chapters) | Monospace bold for literals | f = freqspace(n,'whole') |
| Mathematical expressions | Italics for variables  
Standard text font for functions, operators, and constants | This vector represents the polynomial  
p = x^2 + 2x + 3 |
| MATLAB output | Monospace font | MATLAB responds with  
A = 5 |
| Menu titles, menu items, dialog boxes, and controls | Boldface with an initial capital letter | Choose the File menu. |
| New terms | Italics | An array is an ordered collection of information. |
| Omitted input arguments | (...) ellipsis denotes all of the input/output arguments from preceding syntaxes. | [c,ia,ib] = union(...) |
| String variables (from a finite list) | Monospace italics | sysc = d2c(sysd,'method') |
Related Products

The MATLAB Web Server requires MATLAB Release 11 or later.

No other MathWorks-supplied software is required.

See “Product Requirements” on page 1-5 for additional requirements specific to Web-based software applications.

The MathWorks provides numerous products that can be used in conjunction with the MATLAB Web Server.

For more information about any of these products, see either:

- The online documentation for that product, if it is installed or if you are reading the documentation from the CD
- The MathWorks Web site, at http://www.mathworks.com; see the “products” section

**Note** The toolboxes listed below all include functions that extend MATLAB’s capabilities.

<table>
<thead>
<tr>
<th>Product</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATLAB Compiler</td>
<td>Compiler for automatically converting MATLAB M-files to C and C++ code</td>
</tr>
<tr>
<td>MATLAB C/C++ Math Library</td>
<td>Library for automatically converting MATLAB applications that contain math and graphics to C and C++ code for stand-alone applications</td>
</tr>
<tr>
<td><strong>Product</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MATLAB C/C++ Graphics Library</td>
<td>Library for automatically compiling MATLAB programs that contain graphics and graphical user interfaces (GUIs) into complete stand-alone applications</td>
</tr>
<tr>
<td>MATLAB Runtime Server</td>
<td>MATLAB environment in which you can take an existing MATLAB application and turn it into a stand-alone product that is easy and cost-effective to package and distribute. Users access only the features that you provide via your application’s graphical user interface (GUI). They do not have access to your code or the MATLAB command line.</td>
</tr>
</tbody>
</table>
MATLAB on the Web

Introduction .............................................. 1-2
MATLAB Web Server Environment ................. 1-2
Building MATLAB Web Server Applications ...... 1-3

Product Requirements ................................. 1-5
Web Requirements ...................................... 1-5

Installation .............................................. 1-7
Availability .............................................. 1-7
Installation Procedure ................................ 1-7
General Post-Installation Procedures .......... 1-7
Solaris/Linux Post-Installation Procedures ... 1-9
Windows NT Post-Installation Procedures .... 1-11

Graphics Display ....................................... 1-13
Downloading and Installing VNC ................... 1-13
Downloading and Installing Perl .................... 1-14
Starting and Stopping VNC ......................... 1-14
Using VNC with the MATLAB Web Server ...... 1-15
Introduction

MATLAB Web Server Environment

The MATLAB® Web Server enables you to create MATLAB applications that use the capabilities of the World Wide Web to send data to MATLAB for computation and to display the results in a Web browser. The MATLAB Web Server depends upon TCP/IP networking for transmission of data between the client system and MATLAB. The required networking software and hardware must be installed on your system prior to using the MATLAB Web Server.

In the simplest configuration, a Web browser runs on your client workstation, while MATLAB, the MATLAB Web Server (matlabserver), and the Web server daemon (httpd) run on another machine.
In a more complex network, the Web server daemon can run on a machine apart from the others.

Building MATLAB Web Server Applications
MATLAB Web Server applications are a combination of M-files, Hypertext Markup Language (HTML), and graphics. Knowledge of MATLAB programming and basic HTML are the only requirements.

The application development process requires a small number of simple steps:

1. Create the HTML documents for collection of the input data from users and display of output. You can code the input documents using a text editor to input HTML directly, or you can use one of the commercially available HTML authoring systems, such as Front Page from Microsoft, PageMill from Adobe, or HoTMetaL from SoftQuad.

2. List the application name and associated configuration data in the configuration file matweb.conf. (See “matweb.conf” for a description of this file.)
3 Write a MATLAB M-file that:
   a Receives the data entered in the HTML input form.
   b Analyzes the data and generates any requested graphics.
   c Places the output data into a MATLAB structure.
   d Calls htmlrep to place the output data into an HTML output document template. (See htmlrep for a description of this process.) The maximum amount of HTML data you can receive from MATLAB is 256 KB.
Product Requirements

The MATLAB Web Server has the same supporting hardware and software requirements as MATLAB itself, except for memory required. MATLAB hardware and software requirements are documented in the MATLAB Installation Guide for your computer.

Memory requirements while running the MATLAB Web Server vary with the number of MATLABs configured. Each MATLAB running under the MATLAB Web Server consumes 256 KB of memory.

The MATLAB Web Server requires that TCP/IP networking software must be installed on your computer.

Web Requirements

Web Browsers
To submit input to and receive output from the MATLAB Web Server, you must install a Web browser suitable for your platform. Current versions of the MATLAB Web Server have been tested with Netscape Communicator Release 4.7 and Microsoft Internet Explorer 5.0. Earlier browser releases will probably also work but have not been tested.

The MathWorks does not redistribute these products. You can obtain them directly from the companies that developed them. You can find additional information at http://www.netscape.com or http://www.microsoft.com.

Web Server
You need to install Web server software (HTTPD or similar) on the system where MATLAB is running or on a machine that has network access to the machine where MATLAB is running. There are numerous sources for obtaining this software, including:

- Pre-installed Microsoft Peer Networking Services on your PC
- Netscape Enterprise Server, available by purchase from Netscape Communications, Inc.
- Free distribution over the Internet (Apache: http://www.apache.org)

The Web server software must be capable of running Common Gateway Interface (CGI) programs.
Installation

Availability
The MATLAB Web Server is available on UNIX (Solaris) workstations and IBM PC compatible computers running Microsoft Windows NT or Linux.

Installation Procedure
To install the MATLAB Web Server, follow the normal MATLAB installation procedure for your platform, as documented in the MATLAB Installation Guide for PC and the MATLAB Installation Guide for UNIX. The MATLAB Web Server appears as one of the installation choices you can select as you proceed through the installation screens.

General Post-Installation Procedures
After installing MATLAB and the MATLAB Web Server, you must perform a number of steps that regulate communication between MATLAB and your Web browser.

Note Throughout this document the notation <matlab> represents the MATLAB root directory, the directory where MATLAB is installed on your system. For notational consistency in UNIX command syntax, the notation $MATLAB is used to represent the MATLAB root directory.

Note in particular:

1 To get the demonstration programs discussed in this chapter to work, you need to create a matweb.conf file in the <matlab>/toolbox/webserver/wsdemos directory. The Readme file shows the format for matweb.conf. Replace the notation <matlab> with the name of the root directory where you installed MATLAB. Use the matlabroot command to determine this directory. Also, replace matlabserver_host_name with the TCP/IP hostname for your machine.

2 The installation procedure creates the file matlabserver.conf in the <matlab>/webserver directory. The file contains the notation
-m 1

This number represents the number of MATLABs that can run concurrently. After testing that everything is working properly, you can change this number to something more convenient. On Windows NT edit matlabserver.conf directly with a text editor.

**Note** To see any changes made to matlabserver.conf, you need to stop and restart the MATLAB Web Server. See the section “Understanding matlabserver” on page 3-5 for more information on matlabserver.conf.

On Solaris/Linux use the webconf script to initialize the matlabserver.conf file. Then edit this file to change options further, particularly those that webconf cannot set on the command line.

3. Follow the directions provided by your Web server (httpd) to create the needed aliases:

   a. The home or default directory
   b. /cgi-bin
   c. /icons

Point each of these aliases to `<matlab>/toolbox/webserver/wsdemos` to get the demonstration programs to work.

If your application creates graphic (jpeg) files, you need to provide a location where MATLAB can write these for httpd access, e.g., /icons. The mldir entry associated with each application in the matweb.conf file indicates the location to MATLAB.

If you do not have permission to set up or change these aliases, you must place copies of some files in locations where the httpd can find them.

- Copy matweb (matweb.exe on Windows NT), found in `<matlab>/webserver/bin/arch`, to the directory aliased by /cgi-bin or equivalent. The supported architectures are Windows NT (win32), Solaris (sol2), and Linux (glnx86).
• Copy matweb.conf in <matlab>/toolbox/webserver/wsdeemos to the
directory aliased by /cgi-bin or equivalent.
• Copy all demo HTML files in <matlab>/toolbox/webserver/wsdeemos to the
directory where the httpd keeps all HTML files (often referred to as the
home or default alias).

Note that when aliases are different from those provided in the demo HTML
files, you will have to make the corresponding changes in those HTML files. For
example, if you use the Apache Web Server (http://www.apache.org), the
aliases listed above are called:
   a DocumentRoot
   b ScriptAlias /cgi-bin/
   c Alias /icons/

You edit these in the conf/http.conf file in the main Apache directory. Note
that any alias with a trailing ‘/’, such as ‘/icons/’, must have a trailing
slash in its value, e.g.,
Alias /icons/ ”/local/matlab/toolbox/webserver/wsdeemos/”.

Solaris/Linux Post-Installation Procedures
The MATLAB Web Server installation procedure places five scripts into the
<matlab>/webserver directory:
• webconf: Builds matlabserver configuration file (matlabserver.conf). See
  Chapter 3, “Inside the MATLAB Web Server”, for a discussion of
  matlabserver and the matlabserver.conf file. Use this script to specify the
  number of simultaneous MATLABs to run, the nondefault TCP/IP port, and
  other variables.
• webstart: Stops and restarts matlabserver via calls to webdown and
  webboot. These three scripts must all reside in the same directory.
• webdown: Stops running matlabserver.
• webboot: Starts matlabserver.
• webstat: Displays matlabserver status information.

Enter the command
   script_name -h
at the command prompt to see detailed information about a specific script. After completing the MATLAB and MATLAB Web Server installation process, run the webconf script to generate the matlabserver.conf file. Then run webstart to start matlabserver. Run webdown at any time to stop matlabserver execution.

**Automatic Startup at System Boot**

1 To start matlabserver automatically at system boot, create the following links and file while logged in as root (superuser).

   ln -s $MATLAB/webserver/webboot /etc/webboot$WEBSERVER_MARKER
   ln -s $MATLAB/webserver/webdown /etc/webdown$WEBSERVER_MARKER

   $WEBSERVER_MARKER is a marker string that uniquely identifies this release of the MATLAB Web Server. It is defined in the matlabserver.conf file. (See “matlabserver.conf” on page 3-5.) The default is _TMW$RELEASE, where $RELEASE is a string like 'R12', also set in matlabserver.conf.

   **Note** Add the -c configuration file option to webboot and webdown if the matlabserver.conf file is not in <matlab>/webserver or in the directory where the script is located. For example: webboot -c $CONFIGURATION_FILE

   $CONFIGURATION_FILE is the path to the file matlabserver.conf.

2 In the directory $MATLAB/webserver are two initialization scripts:

   • rc.web.sol2 (Solaris)
   • rc.web.glnx86 (Linux)

   Solaris users should copy the script as shown below.

   cp $MATLAB/webserver/rc.web.sol2 /etc/init.d/webserver

   Linux users should copy the appropriate script as shown below.

   cp $MATLAB/webserver/rc.web.glnx86 /etc/init.d/webserver (Debian)
   cp $MATLAB/webserver/rc.web.glnx86 /etc/rc.d/init.d/webserver (Red Hat)
Open the copied file in a text editor and follow the directions for modifying the file. Save and close the file when you are done.
4 Solaris users should create a link in the `rc` directory associated with run level 3.
   `cd /etc/rc3.d; ln -s ../init.d/webserver S20webserver`

Linux users should look in `/etc/inittab` for the default run level. Create a link in the `rc` directory associated with that run level. For example, if it is 5
   `cd /etc/rc5.d; ln -s ../init.d/webserver S95webserver` (Debian)
   `cd /etc/rc.d/rc5.d; ln -s init.d/webserver S95webserver` (Red Hat)

5 You can test the changes you have made without rebooting your system. To start the MATLAB Web Server on Solaris, enter
   `cd /etc/init.d
   ./webserver start`

   On Linux, enter
   `cd /etc/init.d (Debian)
   cd /etc/rc.d/init.d (Red Hat)
   ./webserver start`

6 To check that the MATLAB Web Server is operational on any system, enter

   `cd $MATLAB/webserver
twebstat -c $CONFIGURATION_FILE`

   `$CONFIGURATION_FILE` is the path to the file `matlabserver.conf`.

**Windows NT Post-Installation Procedures**

After installation, you must reboot your machine to start MATLAB Server as a Windows NT service. The service starts automatically at system boot.

**Startup Sequence**

If you install a new version of MATLAB and the MATLAB Web Server, start MATLAB before starting Web Server. MATLAB performs some system updates required for successful Web Server operation.
Deinstallation
To remove MATLAB Server from the Windows NT Registry, open a command prompt (MS-DOS) window. Enter the command sequence

```
    cd <matlab>/webserver/bin/win32
    matlabserver -remove
```
Graphics Display

We recommend that Solaris/Linux users use Virtual Network Computing (VNC) for their X display. VNC can be used even on systems where there is no hardware frame buffer. It is easy to use, can be easily started with the boot scripts, provides complete user control, and has good performance. For information about use of the VNC software with the MATLAB Web Server, see:

- “Starting and Stopping VNC” on page 1-14.
- “Using VNC with the MATLAB Web Server” on page 1-15.

The VNC software requires installation of Perl. Both VNC and Perl are available in binary form over the Web for free distribution.

Linux users should find Perl already installed on their systems.

Solaris users need to download the Perl distribution and decompress it with the zcat version of the gunzip utility. This utility is found in $MATLAB/webserver/bin/sol2/zcat. Read “Downloading and Installing Perl” on page 1-14 for directions.

### Downloading and Installing VNC

The VNC software is available at the Web site

http://www.uk.research.att.com/vnc/download.html

Select the latest distribution for either or both of the UNIX platforms

**Linux 2.x for x86** (for glnx86)

**Solaris 2.5 (SPARC)** (for Solaris2; it should work on 2.6 and higher)

and download the software. To decompress run

```
zcat file | tar -xvf -
```

where file is either a .tgz file on glnx86 or a .Z file on Solaris.
The directory created contains a README file that discusses how to install the software. The directory contains four files:

- vncviewer
- vncserver
- vncpasswd
- Xvnc

that should be copied to a standard directory on your UNIX path.

**Downloading and Installing Perl**

The Perl software is available at the Web site http://www.activestate.com. Under **Products** select **ActivePerl**; then select **Download Now** next to the **ActivePerl** description on the **Products** page.

There are two formats for Solaris, one using Solaris packages and the other using the generic installer. Select **Solaris 2.6 - AS Package** to use the generic installer.

Download the distribution to a temporary directory, extract the files, change directory to the **ActivePerl** directory, and run the **install.sh** script.

```bash
$MATLAB/webserver/bin/sol2/zcat ActivePerlxxx.tar.gz | tar -xvf -
  cd ActivePerlxxx
  ./install.sh
```

The installation script will ask for the installation directory.

**Starting and Stopping VNC**

Before starting VNC be sure the **vncserver** Perl script has the correct path to the Perl executable at the top of the script. Fix the path if it is incorrect. The default script makes reasonable assumptions about the geometry and color requirements of your virtual X display. If you need to change the assumptions, you can specify different geometry and color requirements on the command line or edit the **vncserver** script. See the README file with the VNC distribution for additional details or the online documentation at the Web site. Type

```bash
vncserver -help
```

for information about arguments to the script.
The first time you run the `vncserver`, you will be prompted for a password. This password controls access to the VNC viewer, which is not used with the Web Server.

You do not have to specify a display number when starting the VNC server, but it is best to specify one to prevent any potential conflict with the main X display, which is normally 0.

**Starting VNC**

To start the VNC server, enter

```
    vncserver :number
```

where `number` is something other than 0. 1 is a good value. Set the DISPLAY variable to `:<number>` in your configuration file or use `-display :<number>` when starting the Web Server via the `webstart` or `webboot` script.

If your Web Server application requires MATLAB to render better color than the default, set the `depth` argument when you start the VNC server

```
    vncserver :<number> -depth <depth>
```

where `:<depth>` is either 16 or 24.

**Stopping VNC**

To stop the VNC server, enter

```
    vncserver -kill :number
```

where `number` is the same as that used to start the server.

After stopping the VNC server, it takes about 30 seconds for the socket to time out and clear, so wait a while after stopping the server before restarting it.

**Using VNC with the MATLAB Web Server**

Use the same value for `display` with the MATLAB Web Server that you used when starting the VNC server.

For example, suppose you want to start the Web Server by passing the display on the command line. Here is some sample output from such a command.
webstart -display :1
[webstart]: Calling webdown to take down MATLAB Web Server...
  No server to take down . . .
[webstart]: Calling webboot to start MATLAB Web Server . . .
  Waiting for MATLAB Web Server to come up . . .
  Type your interrupt character (usually CTRL-C) to quit.
Time = 10 secs : still waiting for 1 of 2 MATLAB sessions . . .
Time = 20 secs : still waiting for 1 of 2 MATLAB sessions . . .
Time = 30 secs : still waiting for 1 of 2 MATLAB sessions . . .
ALL MATLAB sessions started, but . . .
Time = 10 secs : Web Server startup still not complete . . .
Time = 20 secs : Web Server startup still not complete . . .
MATLAB Web Server is up . . .

The display can also be indicated in the matlabserver.conf file.
Getting Started

Introduction .......................................................... 2-2
Templates .............................................................. 2-2

Creating Input Documents ............................................ 2-4
Input Template ......................................................... 2-4

Creating MATLAB Web Server M-Files ............................ 2-7
M-File Template ....................................................... 2-7

Creating Output Documents ........................................... 2-10
Output Template ....................................................... 2-10

Debugging Your Application ......................................... 2-13
Debugging Procedure ............................................... 2-13

Additional Application Examples ................................. 2-15
Data Display .......................................................... 2-15
MATLAB Graphics ................................................... 2-16
Stock Price Simulation .............................................. 2-20
Introduction

The process of creating a MATLAB Web Server application involves the creation of:

- An HTML input document for data submission to MATLAB. See “Creating Input Documents” on page 2-4.
- An HTML output document for display of MATLAB’s computations. See “Creating Output Documents” on page 2-10.
- A MATLAB M-file to process input data and compute results. See “Creating MATLAB Web Server M-Files” on page 2-7.
- A test file to validate code before distributing the application over the Web. See “Debugging Your Application” on page 2-13.

The process of creating a MATLAB Web Server application can be simplified through the use of a set of templates that has been provided. These are discussed in “Templates” on page 2-2.

Templates

Four templates found in the directory `<matlab>/toolbox/webserver/wsdemos` simplify the process of creating a MATLAB Web Server application:

- input_template.html
- output_template.html
- mfile_template.m
- tmfile_template.m

Each template provides actual code that you need to incorporate into your application plus instructions on how to modify the template where necessary. If you follow the directions in these templates, you should be able to create MATLAB Web Server applications with reasonable effort.

Additionally provided in `<matlab>/toolbox/webserver/wsdemos` is `webmagic`, a magic squares demonstration program. A magic square produces the same sum along any row, column, or either of the two main matrix diagonals. There are four files associated with `webmagic`:
• *webmagic1.html*: the *webmagic* input document
• *webmagic2.html*: the *webmagic* output document
• *webmagic.m*: the *webmagic* MATLAB M-file
• *twebmagic.m*: the *webmagic* stand-alone test file

To learn how the four templates were modified to create the *webmagic* application, we will examine templates and note the specific changes applied.

If you want to look at some other applications created with these templates, see the section “Additional Application Examples” on page 2-15.
Creating Input Documents

**Input Template**
The file `input_template.html` provides the code needed to create a MATLAB Web Server input document. An abbreviated version looks like

```html
<!-- STEP 1
Choose either the NT version or the Unix version of the form tag (depending on which platform the matweb client program will be run):
-->
<!-- NT version: -->
<form action="/cgi-bin/matweb.exe" method="POST">
<!-- Unix version: -->
<form action="/cgi-bin/matweb" method="POST">

<!-- STEP 2
Create a hidden field naming your M-file. Replace MY_M_FILE with the name of main MATLAB function of your application. (An HTML input field of type "hidden" is commonly used to pass variables to a web server. It is not displayed by the browser.)
-->
<input type="hidden" name="mlmfile" value="my_m_file">

<!-- STEP 3
Add all your other HTML form tags here. Replace MY_INPUT_VARIABLE_1 with the name of an input variable in your application.
-->
<p>My input variable 1: <input type="text" name="my_input_variable_1"></p>

<!-- STEP 4
Create a "submit" input tag for the user to click to send the input to your program.
-->
```
webmagic Input
Examine the significant part of the source for webmagic1.html.

```html
<form action="/cgi-bin/matweb.exe" method="POST">
    <input type="hidden" name="mlmfile" value="webmagic">
    <p>Magic square size (minimum is 3, maximum is 20):
        <input type="text" size="2" maxlength="2" name="msize"></p>
    <p><input type="submit" name="Submit" value="Submit"></p>
</form>
```

The line

```html
<form action="/cgi-bin/matweb.exe" method="POST">
```

calls matweb, the entry point to the MATLAB Web Server. matweb.exe is the Microsoft Windows NT name of the program used by the MATLAB Web Server to extract data from HTML forms. On Solaris/Linux this program is called just matweb. We refer to the program as matweb throughout this document except when the platform distinction is important. matweb is described more thoroughly in the next chapter.

The next line

```html
<input type="hidden" name="mlmfile" value="webmagic">
```

provides the name of the MATLAB M-file (mlmfile) to run. In this application the M-file is named webmagic.

Lastly, the input

```html
<input type="text" size="2" maxlength="2" name="msize"></p>
```

passes to webmagic.m a two-character field named msize, which contains the size of the magic square to compute.
To display the input document and run the magic squares demonstration locally on your computer, start your Web browser and set the URL to http://<your_domain>/webmagic1.html.

The magic squares input document, webmagic1.html, is displayed in your browser.

Enter the size of the magic square matrix you want to compute and press Submit.
Creating MATLAB Web Server M-Files

M-File Template
You use the M-file template to code your MATLAB application as you normally do. The template provides the additional code you need to accept input from your HTML input document and to return results to your HTML output document. An abbreviated version looks like

```matlab
function retstr = mfile_template(instruct, outfile)
    % STEP 1
    % Initialize the return string.
    retstr = char('');

    % STEP 2
    % Set working directory.
    % The variables INSTRUCT.MLDIR and INSTRUCT.MLID are provided
    % automatically to all MATLAB Web Server applications that use
    % the matweb program.
    cd(instruct.mldir);

    % STEP 3
    % Get the HTML form input variables
    my_input_variable_1 = instruct.my_input_variable_1;

    % STEP 4
    % Perform your MATLAB computations, graphics file creations, etc., here:

    % STEP 5
    % Put variables that you want to put into your HTML output document in an output structure. You create an HTML output document from OUTPUT_TEMPLATE.HTML.
    outstruct.my_output_variable_1 = More MATLAB computations creating ...
        scalars, matrices, cell arrays, graphics files, etc.;

    % STEP 6
    % Call the function HTMLREP with the output structure you just
```
% created and the filename you created from OUTPUT_TEMPLATE.HTML.
% Replace <OUTPUT_TEMPLATE.HTML> with the name of the HTML output
% file you created using OUTPUT_TEMPLATE.HTML.
% This call fills the string RETSTR to return and optionally
% writes the output as a file if a valid filename is given
% as the
% second argument to the present function.
templatefile = which('<OUTPUT_TEMPLATE.HTML>');</nargin == 1)
    retstr = htmlrep(outstruct, templatefile);
elseif (nargin == 2)
    retstr = htmlrep(outstruct, templatefile, outfile);
end

webmagic M-File
The data entered on the webmagic1.html input document is automatically passed to MATLAB, which then runs the webmagic function. Notations in **boldface** refer to steps in the M-file template.

% Initialize the return string. **(Step 1)**
retstr = char('');

**(Step 2. Not needed. No generated graphics.**)

% Get the msize (string) variable. Convert to a number. **(Step 3)**
% Check the range.
if(~length(instruct.msize))
    msize = 3; % Default empty field.
else
    msize = str2double(instruct.msize);
    if (msize > 20), msize = 20; end % Max size.
    if (msize < 3), msize = 3; end % Min square.
end

% Save size as a char string in structure OUTSTRUCT. **(Step 4, 5)**
outstruct.msize = msize;
% Create magic square in output structure OUTSTRUCT.
outstruct.msquare = magic(msize);

% Get column, row, and diagonal sum. Put in OUTSTRUCT.
d = sum(outstruct.msquare,1);
outstruct.msum = d(1,1);

% Output the results and optionally write as a file if the
% filename was given as the second argument to WEBMAGIC.

(Step 6)
templatefile = which('webmagic2.html');
if (nargin == 1)
    retstr = htmlrep(outstruct, templatefile);
elseif (nargin == 2)
    retstr = htmlrep(outstruct, templatefile, outfile);
end
Creating Output Documents

Output Template
The file output_template.html provides the code needed to create a MATLAB Web Server output document. An abbreviated version looks like

<!-- Modify this file to create your own HTML output document and save it as <MY_OUTPUT>.html, where <MY_OUTPUT> is replaced by a name that has meaning within the context of your application. -->

<!-- STEP 1
Display a MATLAB scalar or character string. Replace <MY_OUTPUT_VARIABLE_1> in the following line with the name of the MATLAB variable you want to display. Change the other text to something meaningful within the context of your application.
--> My output variable 1 has been computed to be $<my_output_variable_1>$

<!-- STEP 2
Put all your other HTML tags here.
-->

webmagic Output
The webmagic output document contains three variables.

$msquare$ -- the completed magic square

$msize$ -- the size of the magic square

$msum$ -- the magic square sum along its rows, columns, or diagonals

Using htmlrep the webmagic function replaces these variables with actual values, using the input obtained from webmagic1.html.
The source for the webmagic2.html template document looks like

```html
<!--
HTML output template used by webmagic.m in call to the
function HTMLREP. (HTMLREP replaces variable names delimited
by dollar signs, e.g., $msquare$, with their values. It also
generates HTML tables and select lists dynamically from
matrics,
cell arrays, and vectors.)
--> 
<html>
<head>
<title>Magic Square in an HTML Table</title>
</head>
<div align="center">
<strong>Magic Square in an HTML Table</strong></p>
<!--
Use the MATLAB "AUTOGENERATE" HTML attribute to generate a
table dynamically from the variable "msquare", which is a
matrix (in program webmagic.m).
--> 
<table border="1" cellspacing="1" autogenerate="$msquare$">
<tr>
<td align="right">
</td>
</tr>
</table>

<!--
Replace $msize$ and $msum$ with the contents of MATLAB
variables
named "msize" and "msum" respectively as computed in
webmagic.m.
--> 
<p>
Size: $msize$<br>
Sum of all columns, rows, and diagonals: $msum$
</div>
```
When displayed in a browser, webmagic2.html looks like
Debugging Your Application

Debugging Procedure
You can use the MATLAB debugging facility plus your Web browser to debug your application before making it available to all users.

An effective method of debugging is to write your application M-file to accept two arguments, as in

```
retstr = webmagic(structure, testfile)
```

You can use the second argument to create an HTML file that displays test output.

As a first debugging step, create a driver program that sets up the input variables and calls the main function. You can use the MATLAB debugging facilities to check the logic of your test program. The file twebmagic.m located in <matlab>/toolbox/webserver/ws demos is an example of such a driver program. (Note the use of the wsetfield function to create the test input variables. This is optional. Elements of the structure s could be created directly, e.g., s.msize = '5'.)

```
function twebmagic()
%TWEBMAGIC Example standalone test of webmagic function.
%TWEBMAGIC Does setup and calls webmagic. Creates the output file, %twebmagic.html.

% Set up input variables.
s = {};
s = wsetfield(s, 'mlmfile', 'webmagic');
s = wsetfield(s, 'msize', '5');
s = wsetfield(s, 'mldir', '.');

% Create an output test file.
str = webmagic(s, 'twebmagic.html');

Because the driver program calls webmagic with two arguments, webmagic writes its output to the file twebmagic.html. The call to htmlrep within the webmagic function handles this.
```

```
retstr = htmlrep(outstruct,'webmagic2.html', outfile)
```
outfile in this case is twebmagic.html, the second argument passed to webmagic.

The second step in debugging is to use your Web browser to examine your test file (outfile) and make appropriate changes until the output is displayed as you intend.

**Debugging Template**

To assist you in debugging, we have provided the template tmfile_template.m, shown below in abbreviated form.

```matlab
function tmfile_template()

% STEP 1
% Set up input variables as they would come in from
% the HTML input form created from INPUT_TEMPLATE.HTML.
outstruct.my_input_variable_1 = some appropriate test value;

% STEP 2
% Call your application function that was created from
% <MFILE_TEMPLATE.M>. Replace <MFILE_TEMPLATE> with the
% name of your application M-file. Provide a test output
% file name for the optional argument by replacing
% <TEST_OUTPUT.HTML> with your test output HTML file name.
retstr = <MFILE_TEMPLATE>(outstruct, '<TEST_OUTPUT.HTML>');

% STEP 3
% Examine the file you supplied for <TEST_OUTPUT.HTML> in
% your web browser.
```
Additional Application Examples

The easiest way to learn how to create MATLAB applications that send and receive data over the Web is to analyze the sample programs included with your MATLAB Web Server distribution. To access these sample programs, open the file `matlab/toolbox/webserver/ws demos/index.html` in your browser. By analyzing the code used to create these examples, you can expand upon them to create more complex MATLAB Web Server applications.

Data Display

The `players` demonstration function illustrates a basic use of the MATLAB Web Server to display data over the Web. The file `players.txt` contains a data base with information about The MathWorks softball team.

<table>
<thead>
<tr>
<th>PLAYER</th>
<th>POSITION</th>
<th>AVERAGE</th>
<th>AT BATS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ron Berg</td>
<td>First Base</td>
<td>.337</td>
<td>30</td>
</tr>
<tr>
<td>John Theytaz</td>
<td>Second Base</td>
<td>.294</td>
<td>22</td>
</tr>
<tr>
<td>Charles Carmody</td>
<td>Third Base</td>
<td>.261</td>
<td>34</td>
</tr>
<tr>
<td>Debby Oldman</td>
<td>Shortstop</td>
<td>.287</td>
<td>29</td>
</tr>
<tr>
<td>Jack Pirrotta</td>
<td>Right Field</td>
<td>.256</td>
<td>27</td>
</tr>
<tr>
<td>Josh Tillson</td>
<td>Center Field</td>
<td>.301</td>
<td>31</td>
</tr>
<tr>
<td>Eugene Goldbrick</td>
<td>Left Field</td>
<td>.281</td>
<td>31</td>
</tr>
<tr>
<td>Donna Navillus</td>
<td>Pitcher</td>
<td>.222</td>
<td>32</td>
</tr>
<tr>
<td>Anna Alman</td>
<td>Catcher</td>
<td>.290</td>
<td>30</td>
</tr>
</tbody>
</table>

`players` uses a few MATLAB commands to read the tab-delimited file, convert the data to an HTML file, and display the output in a Web browser. No input form is needed for this application, only a URL. The URL can be entered directly in your browser or as a live link in another page.
The URL for the players example is

http://<your_domain>/cgi-bin/matweb.exe?mlmfile=players&

on Windows NT. (On Solaris/Linux use matweb instead of matweb.exe.)

The output looks like

<table>
<thead>
<tr>
<th>Name</th>
<th>Position</th>
<th>Stats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ron Berg</td>
<td>First Base</td>
<td>337/30</td>
</tr>
<tr>
<td>John Thelaz</td>
<td>Second Base</td>
<td>284/22</td>
</tr>
<tr>
<td>Anna Almar</td>
<td>Catcher</td>
<td>230/30</td>
</tr>
<tr>
<td>Debby Oldan</td>
<td>Shortstop</td>
<td>297/29</td>
</tr>
<tr>
<td>Eugene Goldmark</td>
<td>Left Field</td>
<td>281/31</td>
</tr>
<tr>
<td>Charles Carmonny</td>
<td>Third Base</td>
<td>261/34</td>
</tr>
<tr>
<td>Jack Pirrota</td>
<td>Right Field</td>
<td>256/27</td>
</tr>
<tr>
<td>Donna Navilli</td>
<td>Pitcher</td>
<td>242/32</td>
</tr>
<tr>
<td>Josh Tillson</td>
<td>Center Field</td>
<td>221/31</td>
</tr>
</tbody>
</table>

If you would like to experiment on your own with a similar simple application, use players as a model to create an M-file that reads your own text file, e.g., myfile.txt, and places data into a MATLAB structure. To display the result in your Web browser, use the above URL, changing the value of the mlmfile argument to the name of your new M-file. Also, copy the entry for players in matweb.conf and change the name of the application within the brackets [] to the one you have chosen.

**MATLAB Graphics**

The `webpeaks` function, included as a demonstration program, creates a peaks plot and returns the output to your Web browser. In examining portions of the `webpeaks` code, you will see how to include MATLAB graphics as part of a MATLAB Web Server application.

To start the `webpeaks` demonstration, set the URL in your browser to http://<your_domain>/webpeaks.html, the `webpeaks` input document.
This input document allows you to set the characteristics of the peaks plot you want to generate. This is a more complex input document than the one we used with webmagic, as it makes use of HTML frames. The code in the source file

```html
<form action="/cgi-bin/matweb.exe" method="POST"
     target="outputwindow">
    <input type="hidden" name="mlmfile" value="webpeaks">
</form>
```

calls the webpeaks function and targets the output to a frame on the lower portion of the input document itself.

In addition to the code necessary to compute and display the peaks function, the file webpeaks.m contains additional code specific to the transmission of graphics data across the Web. In webpeaks.m the code

```matlab
mlid = getfield(h,'mlid')
```

extracts mlid from the structure h.

mlid is a unique identifier that matlabserver provides. Using the value of mlid to construct filenames ensures that filenames are unique. It can
also be used to maintain contexts among the different connections in an application. You can see this in the code:

```matlab
s.GraphFileName = sprintf('%speaks.jpeg',mlid)
```

which creates a name for a jpeg file. If `mlid` has the value `ml00277`, for example, the jpeg file will be named `ml00277peaks.jpeg`.

The function `htmlrep` replaces MATLAB variable names it finds in the HTML output template file `webpeaks2.html` with the values in the input structure `s`:

```matlab
rs = htmlrep(s,'webpeaks2.html')
```

 `$GraphFileName$`, the variable that represents the graphic output, is found in the line:

```html
<img border=0 src="$GraphFileName$">
```

in `webpeaks2.html`.

The final output document shows both the input and output frames.
Additional Application Examples

Note When the input form contains a “clickable” image created by an `<input>` tag of the form `<type = "image" name = "mymap">`, the variable names for the x and y coordinates are normally passed to the program as `mymap.x` and `mymap.y`. The MATLAB Web Server converts these to `mymap_x` and `mymap_y`. For example, the input `<input type = image src = "mymap.jpeg" name = "mymap">` results in storing the x and y coordinates `mymap_x` and `mymap_y` in the structure passed to your program.
Stock Price Simulation

Now that you are familiar with the use and operation of the MATLAB Web Server, look at webstock, a program that uses MATLAB to simulate possible stock price scenarios based on user assumptions about estimated return and price volatility.

Simulation of Future Stock Prices

This is a Monte-Carlo simulation of the price of a stock over the next year. Input today's price, the expected rate of return, and the volatility of the stock. You can plot a number of possible price scenarios at once.

The prices are generated by sampling a lognormal stock price process. See, for example, N. Chriss "Black-Scholes and Beyond", Irwin, 1997. The lognormal stock price model is used in finance to value stock options.

Stock symbol: IBM
Current Stock Price: 104
Annualized Expected Return (percent): 90
Annualized Volatility (percent): 30
Number of Simulated Paths: 10

Set your browser to http://<your_domain>/webstock1.html to begin the simulation. When you are finished, examine the source of the input (webstock1.html) and output (webstock2.html) documents, the MATLAB M-file (webstockrnd.m), and the stand-alone test M-file twebstockrnd.m.
Inside the MATLAB Web Server

<table>
<thead>
<tr>
<th>MATLAB Web Server Components</th>
<th>3-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>File Locations</td>
<td>3-4</td>
</tr>
</tbody>
</table>

| Understanding matlabserver                        | 3-5 |
| matlabserver.conf                                 | 3-5 |
| Using matlabserver                                | 3-6 |
| Returning Results via the Web                     | 3-9 |
MATLAB Web Server Components

The MATLAB Web Server consists of a set of programs that enable MATLAB programmers to create MATLAB applications and access them on the Web:

• **matlabserver**: Manages the communication between the Web application and MATLAB.

  *matlabserver* is a multithreaded TCP/IP server. It runs the MATLAB program (M-file) specified in a hidden field named *mlmfile* contained in the HTML document. *matlabserver* invokes *matweb.m*, which in turn runs the M-file.

  *matlabserver* can be configured to listen on any legal TCP/IP port by editing the *matlabserver.conf* file on Windows NT or running *webconf* on Solaris/Linux. The number of simultaneous MATLABs is specified here.

• **matweb**: A TCP/IP client of *matlabserver*. This program uses the Common Gateway Interface (CGI) to extract data from HTML documents and transfer it to *matlabserver*.

• **matweb.m**: Calls the M-file that you want the Web application to run.

Two configuration files are used in conjunction with the MATLAB Web Server programs:

• **matweb.conf**: A configuration file that *matweb* needs for connecting to *matlabserver*. Applications must be listed in *matweb.conf*.

• **hosts.conf**: An optional file providing additional security. If *hosts.conf* is present, only listed machines can connect to the MATLAB Web Server. Machines are listed by name in a single column, e.g.,

  
  parrot.mathworks.com
  bluebird.mathworks.com

  Machines must be listed by name, not by IP number. The operating system resolves the name into a valid IP address.

Figure 3-1, MATLAB on the Web, is diagram showing how MATLAB operates over the Web.
Figure 3-1: MATLAB on the Web
**File Locations**

Any M-files used in conjunction with a Web application, including `matweb.m`, must appear on the MATLAB path. The `matweb` and `matweb.conf` files must appear under a `/cgi-bin` alias. Any generated graphics must be located where the Web Server can find them and programs can write them.
Understanding `matlabserver`

`matlabserver` is designed to run continuously in the background as a Windows NT service or as a background process on other systems. (Administrator privileges are normally required to install `matlabserver`.) For testing you can turn on terminal logging by entering the command `matlabserver -t` at a Windows NT command prompt (`webstart -t` on Solaris and Linux) when starting `matlabserver`. (See “General Troubleshooting” on page B-2.)

- To learn about how the `matlabserver.conf` file controls `matlabserver`, see “`matlabserver.conf`” on page 3-5.
- To learn about the `matweb` program and additional files and programs that run in conjunction with `matlabserver`, see “Using `matlabserver`” on page 3-7.
- To learn how the MATLAB Web Server returns output, see “Returning Results via the Web” on page 3-10.

`matlabserver.conf`

When `matlabserver` starts up, it looks in the file `matlabserver.conf` for its initial setting data. On Windows NT the installation procedure creates this file in the `<matlab>/webserver` directory while installing the MATLAB Web Server. On other systems you must first run the `webconf` script to establish values for some of the arguments in the file before starting the MATLAB Web Server.

Configuration settings must appear on the first line of the `matlabserver.conf` file. Basic options that can go on the first line include:

- Port number (port that `matlabserver` listens on)
- Threads (maximum number of simultaneous MATLABs)
- Timeout (how long in seconds to wait for `matlabserver` to start)
- Additional MATLAB path (path prepended to the MATLAB path at startup)

You can also specify additional settings related to troubleshooting. See Appendix B, “Troubleshooting Web Server” for details.

Additional lines appear in the Solaris/Linux `matlabserver.conf` file, starting on line 2, to specify various shell variables that fully specify the environment.
Note that the values for the variables can contain references to other shell variables using the normal shell $ notation.

### Table 3-1: Additional Environment Variables for Solaris/Linux

<table>
<thead>
<tr>
<th>Variable</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERVER</td>
<td>Distinguishes multiple servers in one configuration file. Usually blank.</td>
</tr>
<tr>
<td>SERVER_HOST</td>
<td>Hostname where <code>matlabserver</code> will run.</td>
</tr>
<tr>
<td>START_USERNAME</td>
<td>Username of user who will start the server. (Do not start as root. It causes a security problem.)</td>
</tr>
<tr>
<td>RELEASE</td>
<td>String normally obtained from <code>$MATLAB/.VERSION</code> file. Used in constructing <code>matlabserver</code> log files.</td>
</tr>
<tr>
<td>ARCH</td>
<td>Architecture of <code>matlabserver</code> to run.</td>
</tr>
<tr>
<td>MATLAB</td>
<td>Location of MATLAB to be started by <code>matlabserver</code>.</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>X Window display. Use <code>-nodisplay</code> if no MATLAB graphical display is required.</td>
</tr>
<tr>
<td>LM_LICENSE_FILE</td>
<td>MATLAB license file.</td>
</tr>
<tr>
<td>WEBSERVER_MARKER</td>
<td>String used to construct various log files.</td>
</tr>
<tr>
<td>SLOG_FILE</td>
<td><code>matlabserver</code> log file used for debugging. To turn on add <code>-f $SLOG_FILE</code> on first line.</td>
</tr>
<tr>
<td>ELOG_FILE</td>
<td><code>matlabserver</code> error log file. Turned on automatically. Use <code>webstat</code> to view.</td>
</tr>
<tr>
<td>PID_FILE</td>
<td>File containing the process ID of the <code>matlabserver</code> process.</td>
</tr>
<tr>
<td>CHECK_XHOST</td>
<td>Time in seconds to check for a valid X Window display.</td>
</tr>
</tbody>
</table>
Changing Initial Data

Edit `matlabserver.conf` on Windows NT. On Solaris/Linux run the `webconf` script to initialize the `matlabserver.conf` file. Edit this file if you want to change options further, particularly those that `webconf` cannot set on the command line. To see any changes you have made to `matlabserver.conf`, you need to stop and restart the MATLAB Web Server.

<table>
<thead>
<tr>
<th>Option</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-p [n]</code></td>
<td>Port <code>matlabserver</code> listens on. Default is 8888.</td>
</tr>
<tr>
<td><code>-m [n]</code></td>
<td>MATLABs to run. Default is 1.</td>
</tr>
<tr>
<td><code>-o timeout</code></td>
<td>Number of seconds to wait for <code>matlabserver</code> to start (range from 1 to 32000). Default is 300.</td>
</tr>
<tr>
<td><code>-a path</code></td>
<td>Path prepended to the MATLAB path at startup No default.</td>
</tr>
</tbody>
</table>

Type `matlabserver -h` on Windows NT (`webconf -h` on Solaris/Linux) for a list of additional editable `matlabserver.conf` options.

If `matlabserver` cannot locate a `matlabserver.conf` file, it uses the defaults.

Using `matlabserver`

When we viewed the source of the `webmagic1.html` file, we observed the line of HTML code

```html
<FORM ACTION="/cgi-bin/matweb.exe" METHOD="POST">
```

and we noted that this line establishes communication with MATLAB. `matweb` is a program that resides on the HTTP server and communicates with `matlabserver`. `matweb` requires information found in `matweb.conf` to locate `matlabserver` (which could be running on a different machine).

matweb Program

`matweb` is a client of `matlabserver` that uses Common Gateway Interface (CGI) to get data from HTML forms. It transfers the information to `matlabserver`, which then runs applications written in M-files to produce responses.
When the MATLAB Web Server is installed, `matweb` is placed in `<matlab>/webserver/bin/arch` for all architectures. This placement allows you to run `matweb` on a machine having the HTTPD but not having MATLAB installed and being of a different architecture from the server.

For HTTP server access you must also place a copy of `matweb` in the directory denoted by the `/cgi-bin` alias. The installation process places a copy in `<matlab>/toolbox/webserver/wsdemos` to run the sample applications.

`matweb.conf`

To connect with `matlabserver`, `matweb` requires information stored in the configuration file `matweb.conf`. Create this file inside the directory denoted by `/cgi-bin`, along with the `matweb` program. Use the copy in `<matlab>/toolbox/webserver/wsdemos` as a guide.

An instance of `matweb.conf` looks like

```plaintext
[webmagic]
mserver=parrot

[webpeaks]
mserver=parrot
mdir=/matlab/toolbox/webserver/wsdemos
```

Multiple application configurations must appear in the same file. Each variable appears on a separate line followed by an equal sign `=`, which is then followed by a value, e.g., `mserver=parrot`. Applications are delineated by the main application entry point name (M-file) in square brackets `[ ]`. For example, `[webpeaks]` is on one line followed by all its variables and corresponding values. Note that `webmagic` does not require an `mdir` entry because it does not generate any graphics or save any files. Use the `%` or `#` character at the beginning of any line to comment it out.
All the fields that can be contained in `matweb.conf` are described in Table 3-2.

### Table 3-2: `matweb.conf` Fields

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>[application]</td>
<td>Name of the MATLAB application to run</td>
<td>webmagic</td>
</tr>
<tr>
<td>(required)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mldir</td>
<td>Working directory for reading or writing files. If specified, this directory is automatically added to the MATLAB path.</td>
<td><code>&lt;matlab&gt;/toolbox/webserver/wsdemos</code></td>
</tr>
<tr>
<td>(optional)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mllog</td>
<td>Produces an application-specific log file that records all exchanges between the application and MATLAB. Turn off logging when the program is running because logging has a negative impact on performance.</td>
<td><code>&lt;matlab&gt;/toolbox/webserver/wsdemos/webmagic.log</code></td>
</tr>
<tr>
<td>(optional)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mlserver</td>
<td>Name of host running matlabserver</td>
<td>parrot</td>
</tr>
<tr>
<td>(required)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mlport</td>
<td>Port that matlabserver listens on. This value must correspond to the port number set in the <code>matlabserver.conf</code> file or on the command line (the <code>p</code> argument).</td>
<td>8888 (default)</td>
</tr>
<tr>
<td>(optional)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mltimeout</td>
<td>Seconds to wait for matlabserver before timing out</td>
<td>180 (default)</td>
</tr>
<tr>
<td>(optional)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>my_var</td>
<td>User-created configuration variable</td>
<td>value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

After you create a new MATLAB Web Server application and enter its configuration data into `matweb.conf`, you need to restart `matlabserver` before you can use the application.
matweb M-File
Looking again at the source from the webmagic1.html file (see “webmagic Input” on page 2-5), observe that the line

```html
<input type="hidden" name="mlmfile" value="webmagic">
```

sets argument mlmfile to the value webmagic. The mlmfile argument contains the name of the MATLAB M-file to run.

matlabserver uses the value of mlmfile obtained from the matweb M-file, matweb.m (webmagic in this example) to run the MATLAB application.  webmagic takes the input data from webmagic1.html, computes the magic square of the requested dimensions, and outputs the results using webmagic2.html as a template.

Returning Results via the Web
The MATLAB Web Server distribution kit contains the file webmagic2.html, which serves as an example of an HTML output document template. The webmagic function uses the htmlrep command to place the computed values into the webmagic2.html output template using the code

```matlab
str = htmlrep(s,'webmagic2.html');
```

In this example s is a MATLAB structure containing the results of the webmagic magic squares computation. htmlrep extracts data from s and replaces variable fields in webmagic2.html with the results of MATLAB computation. The completed webmagic2.html form is transmitted to the user’s browser.
Reference
**Function Summary**

This chapter provides detailed descriptions of the functions in the MATLAB Web Server.

<table>
<thead>
<tr>
<th>Function</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>htmlrep</td>
<td>Replace variable names with values in HTML document.</td>
</tr>
<tr>
<td>matweb</td>
<td>MATLAB Web Server main entry point.</td>
</tr>
<tr>
<td>wscleanup</td>
<td>Purge stale files from directory.</td>
</tr>
<tr>
<td>wsprintjpeg</td>
<td>Create JPEG file.</td>
</tr>
<tr>
<td>wsetfield</td>
<td>Add new field or append to existing field.</td>
</tr>
<tr>
<td>Purpose</td>
<td>Substitute values for variable names in HTML document</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>Syntax</td>
<td>outstring = htmlrep(instruct, infile)</td>
</tr>
<tr>
<td></td>
<td>outstring = htmlrep(instruct, infile, outfile)</td>
</tr>
<tr>
<td></td>
<td>outstring = htmlrep(instruct, infile, outfile, attributes)</td>
</tr>
<tr>
<td>Description</td>
<td>htmlrep(instruct, infile) replaces all MATLAB variables in infile, an HTML document, with corresponding values of variables of the same name in instruct. Variables can be character strings, matrices, or cell arrays containing strings and scalars. String and scalar variables are replaced by straight substitution. Output is returned in outstring. Variable names in infile must be enclosed in dollar signs, e.g., $varname$.</td>
</tr>
<tr>
<td></td>
<td>outstring = htmlrep(instruct, infile, outfile) additionally writes output to the HTML document outfile (for stand-alone testing).</td>
</tr>
<tr>
<td></td>
<td>instruct is a MATLAB structure containing variable names (field names) and corresponding values.</td>
</tr>
<tr>
<td></td>
<td>infile is an HTML template file with MATLAB variable names enclosed in dollar signs.</td>
</tr>
<tr>
<td></td>
<td>outfile is the name of an output file for optional standalone testing.</td>
</tr>
<tr>
<td></td>
<td>outstring = htmlrep(instruct, infile, outfile, attributes) provides additional directives to htmlrep. The third argument in this form of the command must be present for the attributes argument to be recognized. Use an empty string ' ' for the third argument if you do not want to direct output to a file. The attributes argument is a MATLAB string (enclosed in ') with the listed attributes separated by spaces. Two attributes are allowed.</td>
</tr>
<tr>
<td></td>
<td>noheader Suppresses the output of the HTML header 'Content-type: text/html\n\n' to outfile and outstring.</td>
</tr>
<tr>
<td></td>
<td>extendmemory Enables dynamic memory extension beyond 256 KB.</td>
</tr>
</tbody>
</table>
Note The extendmemory attribute is designed only for use with htmlrep independently of the MATLAB Web Server. Using it with the MATLAB Web Server will cause unpredictable results with output larger than 256 KB.

HTML tables and select lists can be generated dynamically from matrices or cell arrays containing strings and scalars:

1 Tables can be generated using the special MATLAB AUTOGENERATE HTML table attribute with the matrix or cell array name as the value. For example, the following code automatically generates all the HTML needed to display the entire matrix, msquare, in an HTML table.

```
<TABLE BORDER="1" CELLSPACING="1" AUTOGENERATE="$msquare$">
  <TR>
    <TD ALIGN="RIGHT">
      </TD>
  </TR>
</TABLE>
```

At least one of each of the tags listed above is required. htmlrep uses the HTML code from the <TABLE> tag to the </TABLE> tag as a template for generating the entire table. If different column attributes are required, additional pairs of cell tags (<TD> and </TD>) can be included up to the number of columns in the matrix or cell array. For example, adding these tags

```
<TD ALIGN="CENTER">
</TD>
```

after the </TD> tag above causes the second column to be center-justified.

If there are more columns in the matrix or cell array than <TD> </TD> pairs, the last pair is used for all subsequent columns.

2 SELECT lists are generated using the special MATLAB AUTOGENERATE HTML SELECT attribute with the vector, matrix or cell array name as the value. For example, the following code automatically generates all the HTML needed to display the entire vector, mylist, in an HTML SELECT list. (SELECT lists must appear inside HTML <FORM> and </FORM> tags.)
<SELECT NAME="NAMELIST" SIZE=10 AUTOGENERATE=$mylist$ MULTIPLE>
  <OPTION SIZE=6>
</SELECT>

htmlrep uses the HTML code from the <SELECT> tag to the </SELECT> tag as a template for generating the entire SELECT list. One of each of the tags shown above is required.

If mylist is a matrix or cell array, htmlrep uses only the first column vector to construct the select list.
matweb

Purpose
MATLAB Web Server main entry point

Syntax
matweb(instruct)

Description
matweb is an M-file that in turn calls a MATLAB application M-file stored in the mlmfile field of MATLAB structure instruct. It also passes instruct to the application. The matweb function (M-file) is invoked by matlabserver. instruct contains the fields:

- All the data from the HTML input document
- mlmfile, which stores the name of the M-file to call
- mldir, the working directory specified in matweb.conf
- mlid, the unique identifier for creating filenames and maintaining contexts

If a MATLAB warning or error is encountered, the text is captured and returned to the user's browser. You can disable error and warning notification if you want.

See Also
eval, lasterr, lastwarn, warning
**Purpose**

Purge stale files from directory

**Syntax**

```plaintext
deletecount = wscleanup(filespec, timewindow, direc)
deletecount = wscleanup(filespec, timewindow)
```

**Description**

deletecount = wscleanup(filespec, timewindow, direc) deletes all files matching filespec in the directory direc that are older than the number of hours specified in timewindow. deletecount is the number of files actually deleted.

deletecount = wscleanup(filespec, timewindow) deletes all files matching filespec in the current default directory that are older than the number of hours specified in timewindow. deletecount is the number of files actually deleted.
wsprintjpeg

Purpose
Create JPEG file

Syntax
status = wsprintjpeg(fig, jpegfilename)

Description
status = wsprintjpeg(fig, jpegfilename) creates a JPEG file called jpegfilename. wsprintjpeg attempts to create the JPEG file using the MATLAB print command with the -djpeg argument. If this fails, it creates a temporary PCX file and then calls imread and imwrite to create the JPEG output.

See Also
imread, imwrite, print
Purpose  Add new field or append to existing field

Syntax  

\[ s = \text{wssetfield}(s, \text{name1}, \text{value1}, \ldots) \]

Description  \[ s = \text{wssetfield}(s, \text{name1}, \text{value1}, \ldots) \] sets the contents of the field \text{name1} to \text{value1} and returns the result in the changed structure \( s \). A single value is stored as a character array. Items with multiple values have the values stored in a cell array of strings. Multiple calls serve to add values to an existing field. Either use the MATLAB \text{getfield} function to retrieve the values or reference the structure fields directly.

See Also  \text{getfield}
Directory Structure
MATLAB is distributed in compressed format on CD. The installation procedure moves the files to your hard disk, decompresses them, and installs them into your MATLAB root directory. The installation provides a version of the CGI client program `matweb` for all platforms, regardless of which platform `matlabserver` is licensed for. This allows your `matweb` client to run on a different machine and platform from your copy of `matlabserver`. The supported platforms are:

- PC running Windows NT (`win32`)
- Sun workstation running Solaris (`sol12`)
- PC running Linux (`glnx86`)

In the directory structure shown below, replace the pathname component `arch` with the appropriate identifier for your platform.

After installation of the MATLAB Web Server, your MATLAB directory should include these additional files and subdirectories.

<table>
<thead>
<tr>
<th>File</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>matlabserver.conf</code></td>
<td><code>matlabserver</code> options</td>
</tr>
<tr>
<td><code>webboot</code></td>
<td>Start <code>matlabserver</code> script</td>
</tr>
<tr>
<td><code>webconf</code></td>
<td><code>matlabserver</code> configuration file script</td>
</tr>
<tr>
<td><code>webdown</code></td>
<td>Stop <code>matlabserver</code> script</td>
</tr>
<tr>
<td><code>webstart</code></td>
<td><code>matlabserver</code> restart script</td>
</tr>
<tr>
<td><code>webstat</code></td>
<td>Report <code>matlabserver</code> status</td>
</tr>
</tbody>
</table>

Note: If you create a `hosts.conf` file to control which machines can access the MATLAB Web Server, you must place that file in this directory.
You must also place a copy of the appropriate **matweb** executable and **matweb.conf** into the `/cgi-bin` aliased subdirectory of the HTTP server root directory.

**Table A-2: `<matlab>/ webserver/ bin/ arch`**

<table>
<thead>
<tr>
<th>File</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>matlabserver</td>
<td>The MATLAB Web Server binary</td>
</tr>
<tr>
<td>matlabserver.exe</td>
<td>The MATLAB Web Server binary (Windows NT)</td>
</tr>
<tr>
<td>matweb</td>
<td>MATLAB TCP/IP client of matlabserver (Solaris and Linux)</td>
</tr>
<tr>
<td>matweb.exe</td>
<td>MATLAB TCP/IP client of matlabserver (Windows NT)</td>
</tr>
</tbody>
</table>

**Table A-3: `<matlab>/ toolbox/ webserver/ webserver`**

<table>
<thead>
<tr>
<th>File</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>htmlrep.mexsol</td>
<td>Replace variable names with values in HTML document (Solaris)</td>
</tr>
<tr>
<td>htmlrep.mexglx</td>
<td>Replace variable names with values in HTML document (Linux)</td>
</tr>
<tr>
<td>htmlrep.dll</td>
<td>Replace variable names with values in HTML document (Windows NT)</td>
</tr>
<tr>
<td>matweb.m</td>
<td>Web Server main entry point</td>
</tr>
<tr>
<td>wscleanup.m</td>
<td>Purge stale files from directory</td>
</tr>
<tr>
<td>wsprintjpeg</td>
<td>Create JPEG file</td>
</tr>
<tr>
<td>wssetfield.m</td>
<td>Add new field or append to existing field</td>
</tr>
</tbody>
</table>
### Table A-4: `<matlab>/toolbox/webserver/wsdemos`

<table>
<thead>
<tr>
<th>File</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>dummy.html</td>
<td>Temporary HTML document in bottom frame of <code>webpeaks1.html</code></td>
</tr>
<tr>
<td>index.html</td>
<td>List of demos</td>
</tr>
<tr>
<td>input_template.html</td>
<td>HTML input template</td>
</tr>
<tr>
<td>matweb.conf</td>
<td>Sample <code>matweb.conf</code> file</td>
</tr>
<tr>
<td>mfile_template.m</td>
<td>M-file creation template</td>
</tr>
<tr>
<td>output_template.html</td>
<td>HTML output template</td>
</tr>
<tr>
<td>peaksplot.html</td>
<td>HTML document (input form) in top frame of <code>webpeaks1.html</code></td>
</tr>
<tr>
<td>players.html</td>
<td>Softball players HTML output form</td>
</tr>
<tr>
<td>players.m</td>
<td>Softball statistics file</td>
</tr>
<tr>
<td>players.txt</td>
<td>Softball text data file</td>
</tr>
<tr>
<td>tplayers.m</td>
<td>Stand-alone test driver for <code>players</code></td>
</tr>
<tr>
<td>tmfile_template.m</td>
<td>Test file template</td>
</tr>
<tr>
<td>thtmlrep.m</td>
<td>Test of <code>htmlrep</code> function</td>
</tr>
<tr>
<td>thtmlrep1.html</td>
<td>HTML input form</td>
</tr>
<tr>
<td>webmagic.m</td>
<td>Convert magic square into HTML table</td>
</tr>
<tr>
<td>webmagic1.html</td>
<td>Magic square input form</td>
</tr>
<tr>
<td>webmagic2.html</td>
<td>Magic square output template</td>
</tr>
<tr>
<td>twebmagic.m</td>
<td>Example stand-alone test of <code>webmagic</code> function</td>
</tr>
<tr>
<td>webpeaks.m</td>
<td>Web peaks plot</td>
</tr>
</tbody>
</table>
### Table A-4: `<matlab>/toolbox/webserver/wsdemos` (Continued)

<table>
<thead>
<tr>
<th>File</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>webpeaks1.html</td>
<td>HTML frame</td>
</tr>
<tr>
<td>webpeaks2.html</td>
<td>peaks plot output form</td>
</tr>
<tr>
<td>webstock.html</td>
<td>Stock price simulation input form</td>
</tr>
<tr>
<td>webstock1.html</td>
<td>Stock price simulation main frame</td>
</tr>
<tr>
<td>webstock2.html</td>
<td>Stock price simulation output template</td>
</tr>
<tr>
<td>webstockrnd.m</td>
<td>Stock future price path simulation</td>
</tr>
<tr>
<td>webstocktemp.html</td>
<td>HTML output place holder</td>
</tr>
<tr>
<td>twebstockrnd.m</td>
<td>Stand-alone test driver for <code>webstockrnd</code></td>
</tr>
<tr>
<td>wstextread.m</td>
<td>Place a delimited file into a cell array of strings</td>
</tr>
</tbody>
</table>
Troubleshooting Web Server

General Troubleshooting . . . . . . . . . . . . . . . . . . . B-2
Additional Troubleshooting for Windows NT . . . . . . B-4
Additional Troubleshooting for Solaris and Linux . . . . . B-6
This section provides three categories of troubleshooting information:

• General Troubleshooting
• Additional Troubleshooting for Windows NT
• Additional Troubleshooting for Solaris and Linux

**General Troubleshooting**
This information is relevant to all operating systems that support the MATLAB Web Server.

**Event and Error Logging**
The MATLAB Web Server provides a logging facility that may be useful in diagnosing operational problems. This facility supplements capabilities such as the Windows NT Event Viewer provided by the operating system. Logs may record all events or error events only.

Logging can be controlled using various logging options. Logging options may be set in the `matlabserver.conf` file or specified on the command line. Any option you specify on the command line overrides the value found in `matlabserver.conf`.

Use the `matlabserver` command on Windows NT and the `webconf` and `webstart` scripts on Solaris and Linux to set these options.

**Table B-1: matlabserver Logging Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f [log_filename]</td>
<td>Event log file. Required when logging to a file. File content determined by -v setting. (File logging is inefficient. Use when debugging only.)</td>
</tr>
<tr>
<td>-l [errorlog_filename]</td>
<td>(Solaris and Linux only). Error log file. Default is <code>matlabserver_error.log</code> in current directory. On Windows NT use the Event Viewer to see a list of error events.</td>
</tr>
</tbody>
</table>
Table B-1: matlabserver Logging Options (Continued)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-t</td>
<td>Terminal logging. The logging level is set to 2 (transaction and buffer logging). (If you use this option with the matlabserver command on Windows NT, it must precede all other options on the command line.)</td>
</tr>
<tr>
<td>-v [n]</td>
<td>Verbosity. Controls file logging. Default is 0 (no logging). Additional values are 1 (transaction logging) and 2 (transaction and buffer logging).</td>
</tr>
</tbody>
</table>

Application-Specific Log File Not Produced

If present, the mllog variable in matweb.conf (see “matweb.conf” on page 3-8) creates an application-specific log file. This file is not a MATLAB file; it is controlled by the operating system. If this file is not produced, check that the slashes in the pathname of your file are in the correct orientation for the operating system you are using.

Network bind Error (Port in Use)

If matlabserver fails to start because of a bind error, as noted in matlabserver_error.log or in the Event Viewer on Windows NT, the port you are attempting to run matlabserver on is busy. To fix this problem, you need to change the port number that matlabserver listens on. See the section “matlabserver.conf” on page 3-5 for a discussion on how to change the port number. You may need to ask your system administrator to provide you with a valid unused port number.

M-File Programming Considerations

Make certain that each line of your M-file application is terminated with a ; character. Otherwise, the HTML output will be corrupted.

Connect() failure Error

There are two probable reasons why you may receive a Error: connect() failure message:
1 matlabserver is not running.

On Solaris or Linux run the webstat script. On Windows NT click the Start button, open the Control Panel, and choose Services. The status of MATLAB Server should be Started.

2 Port mismatch between matlabserver.conf and matweb.conf.

The default TCP/IP port that matlabserver listens on is set at 8888. You may change this setting in the matlabserver.conf file with the -p option. The port setting for each application configuration in the matweb.conf file must agree with the port setting in matlabserver.conf. (See “Table 3-2: matweb.conf Fields”.) If you have changed the port setting in matlabserver.conf, you must similarly change the port setting in matweb.conf using the mlport option. If mlport is not explicitly set, the default of 8888 is assumed.

Locating matweb.conf

In some network configurations, it is not possible to give users access to the /cgi-bin directory of the HTTP server. In such cases a matweb.conf file should be created with only one entry, containing the actual location of a configuration file that users can edit. This entry must appear inside angle brackets < >. An example of this type of matweb.conf file is

</apps/projects/strategy/matweb.conf>

where /apps/projects/strategy/matweb.conf is a valid accessible file.

Additional Troubleshooting for Windows NT

Using the Windows NT Event Viewer

The Windows NT Event Viewer captures data that can be useful for debugging matlabserver operations even if you have not requested matlabserver logging through the command options. To access the Event Viewer, choose Start -> Programs -> Administrative Tools -> Event Viewer. When the Event
Viewer appears, click on **Log** on the menu bar and choose **Application** from the pull-down menu.

Double-click on a `matlabserver` entry to receive additional details that may be useful in determining the cause of a `matlabserver` problem.
Startup Sequence
If you install a new version of MATLAB and the MATLAB Web Server, you need to start MATLAB before starting Web Server. MATLAB performs some system updates required for successful Web Server operation.

Additional Troubleshooting for Solaris and Linux

Error Logging
For error logging information, look in matlabserver_error.log in <matlab>/webserver or in the file specified with the -l option in the matlabserver.conf file.

Creating New Applications
After you create a new MATLAB Web Server application and enter its configuration data into matweb.conf, you will need to restart matlabserver before you can use the application.
Selected Bibliography
Numerous books have been published about HTML programming and the World Wide Web. Two that we have found both useful and readable are:


On the Web, the World Wide Web Consortium (W3C) publishes comprehensive information about current and future directions of the Web and the HTML language. You will find their home page at http://www.w3.org. Look at http://www.w3.org/MarkUp for a comprehensive discussion of HTML.
Index

A
alias 1-8
aliases 1-7
applications 1-3
authoring systems 1-3
availability 1-6

B
bind error B-3
browsers 1-5

C
configuration files 3-2
configuration settings 3-5

data display 2-15
debugging 2-13
  procedure 2-13
  template 2-14
deinstallation 1-12
directory structure A-2

E
Event Viewer B-2, B-4
examples 2-15

F
file locations 3-4

G
graphics 2-16

H
hosts.conf
  purpose 3-2
htmlrep 4-3
  purpose 1-4
  with webmagic 2-10
  with webpeaks 2-18

I
input template 2-4
input_template.html 2-2
installation 1-6

L
Linux B-6

M
magic square 2-2
MATLAB graphics 2-16
matlabserver 3-7
  purpose 3-2
  system boot 1-9
matlabserver basic options 3-5
matlabserver design 3-5
matlabserver.conf 3-9
  built by webconf 1-8
  configuring ports 3-2
  creation 1-6
  initial setting data 3-5
  overriding B-2
matweb 4-6
  configuration file 3-8
  M-file 3-10
Index

nomenclature 2-5
single entry B-4
matweb program 3-7
matweb.conf 3-8
application development 1-3
format 1-6
locating graphic files 1-7
mldir 3-9
mllog 3-9
mlport 3-9
mlserver 3-9
mltimeout 3-9
network configurations B-4
purpose 3-2
matweb.exe
client of matlabserver 3-2
matweb.m 3-2
finding mlmfile value 3-10
purpose 3-2
M-file template 2-7
mfile_template.m 2-2
mldir 4-6
purpose 1-7
mlid 4-6
mlmfile 3-2
with matweb 4-6
with webmagic 2-5
mlmfile argument 3-10

P
Perl 1-13, 1-14
players function 2-15
post-installation procedures
general 1-6
Solaris/Linux 1-8
Windows NT 1-11
product requirements 1-5

R
references C-2
requirements 1-5

S
scripts
webboot 1-8
webconf 1-8
webdown 1-8
webstart 1-8
webstat 1-8
Solaris, troubleshooting B-6
stock price simulation 2-20

T
TCP/IP 1-2, 1-3
template
input 2-4
M-file 2-7
output 2-10
tmfile_template.m 2-2
displayed 2-14
troubleshooting
configuration options B-2
Linux B-6

N
NT Event Viewer B-2, B-4

O
output template 2-10
output_template.html 2-2
Solaris B-6
Windows NT B-4
twebmagic.m 2-13
typographical conventions (table) vii

V
virtual network computing (VNC) 1-13
VNC
  virtual network computing 1-13
VNC server
    starting 1-15
    stopping 1-15

W
Web requirements 1-5
Web Server applications 1-3
webboot script 1-8
webconf 3-5, 3-7
  changing settings 1-7
  configuring ports 3-2
webconf script 1-8
webdown script 1-8
webmagic 2-2
  input 2-5
  output 2-10
  processing 2-8
webmagic1.html 2-6
  setting mlmfile 3-10
  source 3-7
webmagic2.html 3-10
  displayed 2-12
  source 2-11
webpeaks 2-16
webstart script 1-8
webstat script 1-8
webstock 2-20
Windows NT
  deinstallation 1-12
  event viewer B-2, B-4
  service 1-11, 3-5
  troubleshooting B-4
wscleanup 4-7
wsprintjpeg 4-8
wssetfield 4-9
  test input 2-13