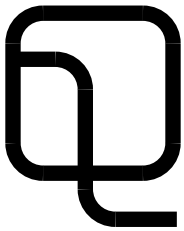

SoftQuad HoTMetaL
for Microsoft Windows

Standard identification

SoftQuad HoTMetaL is an SGML Application Conforming to International Standard ISO 8879 – Standard Generalized Markup Language.

Published by

SoftQuad Inc.
56 Aberfoyle Crescent
Suite 810
Toronto, Canada M8X 2W4
Telephone: (416) 239-4801
Fax: (416) 239-7105
Internet: hotmetal@sq.com
WWW: http://www.sq.com

Document version

SoftQuad HoTMetaL for Microsoft Windows
Second Edition (February 1995)

SoftQuad Inc. makes no warranty of any kind with respect to the completeness or accuracy of this book. SoftQuad may make improvements and/or changes to the product(s) and/or programs described in this book at any time and without notice.

Copyrights and trademarks

Notice

Agencies of the United States Government please note:

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 52.227-7013 and in similar clauses in the FAR and NASA FAR Supplement.

Contents

About this manual

This manual consists of:

1. *A guide for the perplexed,* an introduction and tutorial for HoTMetaL and HTML.
2. A chapter on each of the menus.
3. A chapter on the configuration mechanism.
4. An appendix on keyboard shortcuts.
5. An index

Your suggestions

SoftQuad welcomes your comments and suggestions on this documentation. These will be carefully considered for future versions of the HoTMetaL manual. You may contact us at the following address:

`hotmetal-doc@sq.com`

A guide for the perplexed

This chapter tells you how to start up HoTMetaL and gives the basic information you need to get going with creating and editing files. It contains the sections on the following topics:

- Purpose of HoTMetaL
- How to run HoTMetaL
- Creating and editing files
- Configuring HoTMetaL
- Text and markup
- Tutorial on HoTMetaL and creating web documents
- HTML quick reference
- Screen formatting in HoTMetaL
- Attributes

If you are new to HoTMetaL or HTML, you should certainly read this chapter, as it will help you get acquainted with the product and learn about the components and procedures you'll need to get your work done.

Purpose: editors and browsers

HoTMetaL is an *editor* for creating files that can be read by graphical *browsers* (such as *Mosaic*) that are connected to the World Wide Web (WWW). The file format for such files is called Hypertext Markup Language (HTML). The main difference between an editor like HoTMetaL and browsers is that HoTMetaL is for editing files and browsers are for retrieving, displaying, and reading files. Any text editor can create an HTML file (but we believe that HoTMetaL is a more convenient and pleasant way of doing it!) Browsers take a file saved by HoTMetaL, consisting of text and markup, and do things like screen formatting, generating graphical forms, issuing mail messages, and so forth. It's important to understand that the different kinds of programs do different things. There are many browsers available, and they can process the same HTML file in different ways—and these are outside the control of HoTMetaL! What is in HoTMetaL's control is creating correctly marked-up documents.

Note The HTML format is actually a type of file format based on the Standard Generalized Markup Language (SGML). All HTML files are SGML files (the converse is not true, however—there are many other file formats described by SGML, so most SGML files are not in HTML format.)

Some browsers have a command that lets you see what the HTML form of the current document looks like; you can also save the file in HTML format and view it with a text editor. HoTMetaL provides an easy-to-use, graphical, structured editor for creating files in this format.

An overview of the menus

This section provides a summary of the main features.

- File menu: file manipulation, e.g., opening and closing files; filtering input files.
- Edit menu: cutting and pasting; finding and replacing strings and patterns; spell checking.
- View menu: screen formatting; displaying different views of the document.
- Markup menu: inserting and changing markup; editing URLs; checking document conformance.
- Help menu.

How to run HoTMetaL

The usual way to run HoTMetaL is by double-clicking on the icon labeled 'HoTMetaL' in the HoTMetaL Program Group.

The icon must have a *command line*, consisting of the name and location of the executable file, followed by some command line "options" associated with it. Normally you won't have to worry about this because the installation program creates the command line for you. If you need to change something, note the following:

- You can check the command line associated with this icon by using the Properties... command in the Program Manager File menu.
- You can also change the HoTMetaL command line by using the Properties... command.
- You can create a new *program group* using the New command in the Windows File menu, then add HoTMetaL to that group (also using the New command).
- Alternatively, you could add HoTMetaL to an existing program group. When adding HoTMetaL to a group, associate it with a command line like

  ```
  c:\sqhm\sqhm.exe
  ```

  (Substitute the actual drive and directory for 'c:\sqhm'.

- An icon for the program item will be provided for you by HoTMetaL.

Two other ways you can launch HoTMetaL from within a Windows session are:

- Using the File Manager, move to the directory where you installed HoTMetaL. Now double-click on the file *sqhm.exe*.
- Use the Run... command in the Windows Program Manager with an explicit command line such as:

  ```
  c:\sqhm\sqhm.exe
  ```

Associating HoTMetaL with HTML files

If you want to associate HoTMetaL with your HTML files (thereby allowing you to invoke HoTMetaL by clicking on that file in the File Manager) do the following:

- Choose the Associate... command in the File Manager's File menu.
- In the File with Extension text box, enter 'htm'.
- In the Associate With text box, enter the command:

  ```
  c:\sqhm\sqhm.exe
  ```

  (Or click on Browse... and choose the program from a file chooser.)
- Click on the OK manual.

Setting the HoTMetaL directory

The directory where HoTMetaL is installed is referred to as "the HoTMetaL directory" throughout this documentation.

If you are running a copy of the HoTMetaL executable file (*sqhm.exe*) that *is not* in the installation directory, it will not be able to find automatically the various auxiliary files and directories that it needs to run. In this situation you must inform HoTMetaL of the location of the HoTMetaL directory explicitly. There are two ways to do this:

- Set the DOS environment variable called SQDIR to name the directory in which the software is installed. If the installation directory is *c:\sqhm*, for example, SQDIR should be set (from the DOS prompt) as follows:

  ```
  set SQDIR=c:\sqhm
  ```

  This setting could be added to your *autoexec.bat* file so that it will be executed every time your machine is booted. In any case, it must be done before you start up Windows.
- Specify the HoTMetaL directory on the HoTMetaL command line.
  - Click once on the HoTMetaL icon.
  - Invoke the Properties... command in the Windows File menu.
  - A dialog box will appear. In the Command line text box in this dialog, add the *-sqdir* option followed by the name of the HoTMetaL directory. For example:

```
d:\special\sqhm.exe -sqdir c:\sqhm
```

If you set the HoTMetaL directory using both methods, the value that you specify on the command line will take precedence. If you will be running more than one copy of HoTMetaL, and will have a different HoTMetaL directory for each, you should specify the directories on the command line.

Creating and editing files

This section gives the basic information needed to start editing files with HoTMetaL.

Creating a new file

HoTMetaL comes up with a new, empty HTML document ready for you to use. You can also create a new file as follows:

- Click on the New command in the File menu.

HoTMetaL brings up a new, empty file.

Editing an existing HTML file

If you already have an HTML file that you want to edit:

- Click on the Open... command in the File menu.
- In the dialog box that appears, choose the file that you want to edit.

Once you've done this, HoTMetaL opens the file and you can begin editing.

Some "legacy" HTML documents cannot be opened because they contain bad markup. See the section on the Open... command for information on strategies for dealing with this.

Configuring HoTMetaL

There are many aspects of HoTMetaL's behavior that you can configure to your personal needs or those of your site. For example, you can control default options in the Find and Replace dialog box, set options for the Save command, and specify the locations of various auxiliary files. You can run HoTMetaL without any problems using the default configuration, but at some point you may prefer to customize. This will be particularly true if several people will be using HoTMetaL on the same PC.

Throughout this manual you will see references to features that can be configured by setting a value for a particular *configuration variable*. These are set by editing files called *configuration files*, discussed below. The chapter *The configuration mechanism* lists and describes all of the configuration variables used by HoTMetaL.

Configuration files

The default configuration files read by HoTMetaL are the file *sqhm.ini* located in the directory where HoTMetaL is installed and the file *sqhm.ini* in the Microsoft Windows directory (usually *c:\windows*). These files contain configuration parameters called *configuration variables*. Variables set in the file in the Windows directory take precedence.

You can specify that different files are read by the configuration mechanism: see the chapter *The configuration mechanism* for details.

If a particular variable (parameter) has a setting in more than one file, the value in the file that is read last will take precedence. If a variable is defined more than once in the same file, the value that appears *last* in that file will take precedence over values that appear earlier in the file.

If a variable is not set in any configuration file, but is set in the environment, then the setting from the environment is used. If there is no setting in the configuration files or in the environment, then the built-in default value (if there is one) is used. If there is no default value, the variable is undefined.

In summary, the value of a configuration variable is taken from the following sources, in the order given below:

1. The configuration files.
2. The environment.
3. The built-in default.

Configuration variables read on start-up!

Configuration files (and configuration variables in the environment) are read by HoTMetaL on start-up, so the changes you make will take effect the next time you run HoTMetaL—they will have no effect on a currently-running HoTMetaL. If you need them to take effect immediately, you will have to exit HoTMetaL and restart it.

Setting parameters in the configuration files

You do not need to change any configuration variables unless you wish to customize the HoTMetaL configuration.

A variable is just a name that is assigned some value. You can change these variables by simply editing the configuration files, as appropriate, and making the desired changes.

Basic format for setting variables

Variables are assigned values by putting lines of the following form in the configuration files:

```
variable = value
```

For example:

```
undo_limit=50
```

(*undo_limit* is a configuration variable that specifies the number of successive commands that can be undone or reversed with HoTMetaL's Undo command. The default built in to HoTMetaL is 10; to raise this to 50, you would set the variable as in the example.)

You may put spaces or tabs on either side of the equal sign for readability. Also, if you prefer, you may substitute a colon (:) for the equal sign:

```
undo_limit:50
```

The effect is the same.

You should *not* have any "white space" (spaces or tabs) at the end of the line.

You should take care to use legal values for all the configuration variables. Otherwise, HoTMetaL may behave in unexpected ways.

Further information on setting configuration variables can be found in the chapter *The configuration mechanism*.

Launching browsers and viewers

Some HoTMetaL functionality (previewing the file with a browser and displaying graphics) relies on the existence of external applications. A *configuration file* (the file *sqhm.ini* in the Windows directory) tells HoTMetaL which programs to use to carry out these functions. The configuration variable *html_browser* in this configuration file specify the browser that will be used to preview your HTML files; *view_gif* and *view_bmp* specify the programs used to display GIF and BMP files.

You *must* decide which programs you want to use for these purposes and then modify the values of the variables in the *sqhm.ini* file.

You do this by opening the file with a text editor and making the desired changes.

For example, if you want to use *mosaic* as your HTML browser you would put a line such as the following in *sqhm.ini* (this example uses a typical location for the *mosaic.exe* file—you must use the actual location on your PC):

```
html_browser=c:\mosaic\mosaic.exe
```

The *view_gif* and *view_bmp* variables should specify a graphics viewing program such as *c:\windows\psp.exe* (again, you must use the actual location on your PC).

If you haven't already set these variables, you should do so now, because the tutorial below makes extensive use of the previewing feature.

Text and markup

Like most electronic documents, an HTML file consists of *text* and *markup*. (Markup is special codes inside the file that indicate how part of the file is to be processed: for example, a word-processor file would contain markup indicating typographical features such as the font and font size for various parts of the document.) In an HTML file the markup consists primarily of *elements*. Elements normally consist of a *start-tag* that is placed at the beginning of a section of the text, and an *end-tag* that is placed at the end of that section of text. In HoTMetaL, when you insert an element in the document, you are actually positioning its start- and end-tags. When you want words and phrases to be considered as distinct elements, you surround them with tags.

For example, a title in an HTML file would look like this:

```
<title>This is a title!</title>
```

As you will see, when you are editing documents with HoTMetaL, you don't have to deal with tags on this level: the start- and end-tags are represented on the screen by icons, and HoTMetaL will insert both tags for you when you select a portion of the text to be surrounded by an element. (In fact, HoTMetaL doesn't let you type tags literally—if you type the '<' character, HoTMetaL will replace it by a 'character entity' icon that looks like this: lt .)

The same file may look different when displayed with different browsers. When you are marking up a document in HTML format, you mark up parts of the document according to their function in the structure of the document. For example, there are different elements for headings, lists, list items, paragraphs, titles, and many other parts of a document's structure. One of the reasons for using HTML (and SGML) is that the files can be readily re-processed in a different format by other publishing, browsing, database, etc., applications.

In addition to describing the structure of a document, some elements also describe the links to other documents that can be accessed from an HTML document.

Because HTML documents are structured documents, the elements must be arranged according to specific rules: otherwise, the document is considered invalid. When you are using HoTMetaL, you don't have to keep track of these rules yourself—HoTMetaL does it for you. One of HoTMetaL's most important features is automatic *rules checking*, which ensures that you do not violate the required structure as you are creating a document. As well, when you open or save a document, HoTMetaL checks that the markup is correct and complete.

Many HTML browsers have permitted a very loose, unstructured document format. Therefore, if you are editing existing HTML files, you may find that the structure that HoTMetaL imposes on documents is somewhat constraining. If you need to, you can relax these constraints using the Turn Rules Checking Off command in HoTMetaL's Special menu. Because there is an emerging trend toward browsers that require a stricter document structure, we believe that you will find it to your advantage to create all your new

HTML documents with HoTMetaL's default rules in force. It will also be worthwhile to modify existing documents to conform to these rules.

The document-structuring rules built in to HoTMetaL are designed to be flexible while at the same time maintaining a useful document structure. If an existing 'legacy' document (one that was not created with HoTMetaL!) does not conform to these rules, HoTMetaL's Open... command will give you the opportunity to pass the document through a filter that will attempt to adjust the markup so that the document can be opened. You also have the choice of opening the document as a text document and editing it by hand. Once the errors are fixed, you can use the Interpret Document command to do the equivalent of Open... on the text file.

Templates

The following document templates are supplied with HoTMetaL, in the folder *Templates* in the HoTMetaL folder. You can open these with the Open... command in the File menu, or, more conveniently, with the Open Template... command in the File menu (this command opens the correct folder automatically).

- *custreg.htm* – example of a form.
- *deflist.htm* – example of a definition list.
- *img.htm*, *imgs.htm* – documents with images.
- *HomePage.htm* – a Home Page.
- *h1.htm*, *h2.htm*, *h3.htm* – examples of the use of heading elements.
- *lolist.htm*, *lulist.htm*, *solist.htm*, *sulist.htm* – examples of lists.
- *paras.htm* – a simple document with a few paragraphs.
- *Readme.htm* – a list of templates.

A tutorial on creating documents with HoTMetaL

If you're new to creating web (HTML) documents, you may want to use this short tutorial.

The tutorial covers the following topics:

- The basic document: titles, headers, and paragraphs.
- Character formatting: formatting inline text
- Block formatting
- Lists: ordered and unordered lists
- Links and URLs: anchors and images.

The tutorials don't cover each topic exhaustively, but give you enough information for you to master a topic after obtaining additional details from the HTML *Quick Reference* which you'll find after the tutorial.

The first part of the tutorial may be used as a "quick start" that shows you how to create a file with HoTMetaL. When you've finished that section, you can continue with the other sections or, if you feel comfortable using or

experimenting with HTML on your own, you can skip over the rest of the tutorial and refer to the HTML *Quick Reference* section when you need to find out something about HTML.

Getting started: a basic document

- Start up HoTMetaL.
- Choose Insert Element... in the Markup menu, or type Ctrl-I at the keyboard.

You will see a dialog box containing a list of *elements* that are valid at this point in the document. Elements are the "building blocks" of your document. Since the document is empty right now, any element is valid. However, we'll start by inserting the top-level element, which should be highlighted.

You will probably find it convenient to *pin* this dialog on the screen, by clicking on the button in the upper left corner of the dialog box, and choosing the Pin command from the menu that appears.

- Make sure that the Include Required Elements check box is turned on.
- Double-click on the element HTML in the list of elements, or, if this element is already highlighted, click on the Insert Element button.

HoTMetaL now inserts an HTML element by inserting *start-tag* <HTML> and *end-tag* </HTML> icons. (Sometimes tags are called "commands", but this isn't really accurate.) HTML surrounds all the other elements in the document.

Notice that HoTMetaL has also inserted a HEAD element inside HTML, and a TITLE element inside HEAD. These elements are required in this context in the document. The insertion point is inside TITLE. The words "Document Title" are not part of the text of the document: this is a prefix which is for screen display only.

- Inside the TITLE element, type a title for your sample document.

When you display this document in a browser, the contents of this element will be displayed in the title bar.

The main part of your document is the body, contained in the BODY element.

- Move the insertion point to the right of the </HEAD> end-tag.
- Insert a BODY element from the Insert Element dialog box (this is the only valid element at this point).

If you look at the Insert Element dialog box, you'll notice that you have many choices of elements to insert. However, it's normal to start your document with a heading. Web documents have six levels of headings, represented by the elements H1 through H6.

- Insert an H1 element.

An H1 header will be used for major divisions in your document.

- Type the following (or text of your choice) inside the H1 element:

```
George Orwell
```

Now you're ready to insert some text.

- Move the insertion point to the right of the </H1> end-tag.
- Insert a P (paragraph) element.
- Type some text such as the following:

```
George Orwell is best-known as the author of "Animal
Farm" and "1984", and these books gave the language the
overused adjective "Orwellian". However, these works were
written relatively late in his life, and followed an impressive
body of work that includes accounts of his experiences as
a soldier in the Spanish Civil War, and as a "down-and-out"
tramp roaming the English countryside.
```

You can begin smaller subdivisions of the document with lower-level (H2 through H6) headings. You can skip levels if you want, but your documents will usually look better if you don't.

- After the P element, insert an H2 element.
- Type the text:

```
Early life and education
```

- After the H2 element, insert a P element.
- Type the text:

```
Orwell, whose real name was Eric Arthur Blair, was born
in India in 1903. He was brought to England, along with his
mother and older sister, in 1907. His early education was at
a village school, and later at a private preparatory school.
At the age of 14 he won a scholarship to the prestigious
Eton College.
```

Now perhaps you'd like to see what this document will look like when it's published on the Web. But first, save the file:

- Choose Save from the File menu, or type Ctrl-S at the keyboard.
- Now, choose Preview from the File menu.

HoTMetaL will invoke a browser displaying the file you're editing. (if no browser is launched, then you should check that the *html_browser* configuration variable points to a browser program.)

As we suggested at the start of this section, you may wish to skip directly to the HTML *Quick Reference*, or continue with the extended tutorial.

Character formatting: Adding emphasis to inline text

The techniques in this section are for formatting *inline* text—text that's embedded a paragraph or some other block of text. Techniques for formatting blocks are covered in the next section. In an HTML document, you add emphasis to a piece of text by surrounding it with an element. This is a little different from the approach of many word-processors, in which you would, for example, highlight a piece of text and choose a type style from a menu.

- In the document you just created, highlight the words "Eric Arthur Blair".
- Choose Surround... from the Markup menu, or type Ctrl-U at the keyboard.

The list in the Surround dialog box gives all the elements that can surround the selection.

- Double-click on the element I (italic) in the list.

The selection is now surrounded by <I> and </I> tag icons, and, depending on which style sheet you've chosen, should be formatted in italic. In any case, a browser will format it in italic—you can use Preview to try this out.

Of course, you don't always have to surround the text *after* you've typed it—you can insert an I element with Insert Element... and just type the text between the tags.

For more information on character formatting, see the section *Character formatting* in the HTML *Quick Reference*.

Formatting blocks of text

There are several elements that you can use to surround parts of your document that require special formatting. For example, suppose you wish to add a block quote to the sample document you created above:

- Move the insertion point to the right of the last </P> end-tag.
- Choose Insert Element... in the Markup menu, or type Ctrl-I at the keyboard.
- Insert a BLOCKQUOTE element.
- Type the text:

```
Orwell portrayed his prep school days in harsh terms
in the essay "Such, Such Were the Joys...". This may have
been the result of interpreting his early experiences in
the light of his later political consciousness.
(B. R. Jones)
```

Notice that the text is indented slightly to set off the quotation. A browser will display a block quote with similar special formatting.

For more information on block formatting, see the section *Block formatting* in the HTML *Quick Reference*.

Lists

You can insert five different types of lists in your document.

Ordered (numbered) lists

Ordered lists are lists with numbered items. You don't have to add the numbers yourself—a browser will add these for you.

- Move the insertion point to the right of the </BLOCKQUOTE> end-tag in the sample document.
- Insert an H2 element.
- Type:

```
Orwell's Four Great Motives for Writing
```

- Move the insertion point to the right of the </H2> end-tag.
- Insert an OL (ordered list) element.

When you do this, HoTMetaL automatically inserts an LI (list item) element. With one exception, all lists consist of one or more LIs.

- Inside the LI element, type:

```
Sheer egoism
```

- Make sure the insertion point is just to the *right* of the </LI> end-tag.
- Insert a new LI element.
- Inside the new LI, type:

```
Aesthetic enthusiasm
```

- Insert another LI element after the last one.
- Type:

```
Historical impulse
```

- Insert another LI element after the last one.
- Type:

```
Political purpose
```

To really see how ordered lists work, you should preview the document:

- Save the document.
- Choose Preview from the File menu.

As you can see, the browser has inserted the list numbers automatically.

Unordered lists

An unordered list is one whose items aren't numbered, but instead start with bullets of some kind. Actually there are several kinds of unordered lists available to you: here we'll use the most common and general-purpose list element, UL (unordered list). In this section you'll also learn a new markup command.

- Move the insertion point to a position between the <OL> start-tag and the first <LI> start-tag.
- Choose Change... from the Markup menu, or type Ctrl-L at the keyboard.

The dialog box that appears is similar to the one you saw when you used the Insert Element... and Surround... markup commands. This time the list contains all the elements that can validly replace the current element. (The current element is OL.) Not surprising, your choices are all types of lists.

- Double click on UL in the list of elements.

The OL start- and end-tags have changed to UL tags.

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.

The browser now displays the list items with bullets rather than numbers.

You can *nest* lists by inserting a UL, OL, etc., inside a list item (LI). Some browsers will take account of this by changing the list bullet for the nested list.

For more information on lists, see the section *Lists* in the HTML *quick reference*. To see examples of lists, you can use the Open Template... command to open the templates *lolist.htm*, *lulist.htm*, *solist.htm*, and *sulist.htm*.

Links and URLs

It is normal for HTML documents to contain links to other documents, which can be located anywhere on the WWW. These links are provided by *Universal Resource Locators* (URLs), which name the location and filename of a document, and the protocol used to access it.

Anchors

When you want to create "hot text" that someone can click on in a browser and cause a document to be accessed, you use an "anchor" (A) element.

- Move the insertion point to a position just before the </BODY> end-tag.
- Type the text:

```
See also the
```

- Insert an A element.
- Inside the A, type the text:

```
bibliography
```

The word "bibliography" is hot text. In a browser it will be displayed in a different color than surrounding text.

The second step in creating an anchor is creating the URL. A URL is not part of the *content* of the element, like hot text is. A URL is an *attribute* of the element.

- Make sure the insertion point is inside the A element.
- Choose Edit SGML Attributes... from the Markup menu.

This gives you a dialog box with entries (text boxes) for each of the attributes. Right now you need to work with only one of these.

- In the text box labelled HREF, type the following (without any spaces between the parts):
  1. The characters "file:///".
  2. The name of the drive (without the colon) that the HoTMetaL software is located on.
  3. A vertical bar, '|'.
  4. The directory (full path) that the HoTMetaL software is located in, but instead of typing '\' (backslash) between the directories, as you normally do in Windows, type '/' (slash).
  5. The filename "works.htm".

  What you type should look something like this:

  ```
  file:///c|/sqhm/works.htm
  ```

The word "file" is a *protocol* (also called a "scheme"), which describes how the file referred to in the URL will be accessed by a web browser. You are using the *file* protocol because the file you're going to choose is on your local filesystem. If the document were on a web server, you would choose the protocol *http*.

The directory component of the URL is expressed in the "standard" format, which requires that the colon (:) be replaced by a vertical bar, and that directories be separated by slashes.

- Click on the Apply button.

What this all means is, when someone clicks on the hot text (the word "bibliography") in a browser, the browser will attempt to locate the file (*works.htm*) referred to in the URL.

To see how this works, you should view the document in a browser:

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.
- In the browser, double-click on the word "bibliography".

The browser will now display the file *works.htm*.

Links within the same document

It's possible to make a link between two anchors in the same document. Then, when you click on one of the anchors (call it the "source") the browser window will scroll to the location of the other anchor (the "destination").

- Move the insertion point to a position just before the first </P> end-tag.
- Choose Split from the Markup menu, or type Ctrl-P at the keyboard.
- In the new P element, type:

```
One of his well-known essays is
```

- Insert an A element.
- Type:

```
Why I Write
```

- Make sure the insertion point is inside the anchor you just created.
- Choose Edit SGML Attributes... from the Markup menu.

The dialog box that appears contains text boxes for each of the elements attributes, labeled with the attribute name.

- In the HREF text box, type:

```
#WHY
```

You have just set up the "source" anchor. This is the URL for this anchor, even though it doesn't look like the URL you created in a previous part of this tutorial. Now you have to set up the "destination" anchor:

- Highlight the words "Four Great Motives" in the second H2 element.
- Use the Surround... command to surround this text with an A element.

In this instance you're not going to create a URL for the anchor, rather, you're going to give this anchor a "name" by editing one of its attributes.

- Make sure the insertion point is inside the anchor you just created.
- Choose Edit SGML Attributes... from the Markup menu.
- In the NAME text box, type:

```
WHY
```

- Click on the OK button.

Now you're ready to see the effect of what you've just done.

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.
- If both of the anchors you just created are visible, resize the browser window so that you can see only the first one.
- In the browser, double-click on the words "Why I Write".

The browser window will now scroll so that the location of the "destination" anchor is visible.

In general, you can set up a "source" and "destination" anchor pair by setting the NAME attribute of the destination anchor to '*string*' and setting the HREF attribute (i.e., the URL) of the source anchor to '#*string*'. This sets up a one-way link. You can set up a two-way link by editing the two anchors so that each one's NAME attribute corresponds to the other's HREF.

Images

Web documents often include graphical images. Images are inserted in a document using an element that is similar to the A element.

- Click on the View menu.
- If it contains the command Show Inline Images, choose this command. If it contains the command Hide Inline Images, do nothing.
- Move the insertion point to a position just to the right of the </H1> end-tag.
- Insert an IMG (image) element. (If this element is not visible in the Insert Element dialog box, just type the letter 'i' and the list will scroll to the proper position.)
- Choose Edit SGML Attributes... from the Markup menu.

Now you are going to give the HREF attribute of the IMG element a value very similar to the first URL you created in the section on anchors. It will be the same except for the filename, which in this case is "author.gif". The URL will look something like this:

```
file:///c|sqhm/author.gif
```

- Click on the Apply button.

When you do this, a graphical image will be displayed inline, in the HoTMetaL document window. (This is not really George Orwell, but another writer whose work you know if you're reading this manual.)

The inline image will be displayed in the browser, too.

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.

See the section *Link elements* in the HTML *quick reference* for more information on images; in particular, you will find information on "hot images" and images with hot spots.

Forms

There are certain elements that a browser will display as graphical widgets, such as text fields or pop-up menus, that can accept input from a user. A form in an HTML document is a set of such elements that let the user enter some information and then call a program, located on a web server, that processes the information. For example, you could create a form that lets a user order a product that you're selling: you can set up the form so that when the user clicks on a "submit" button, the order is sent to your order-processing program.

To implement this, you have to:

- Create the form(s).
- Install on your server the program that will process the form's data.

The second of these two steps is beyond the scope of HoTMetaL. You will have to obtain supplementary documentation that explains this mechanism, which is known as the CGI (Common Gateway Interface). If you open the file *faq.htm* in the HoTMetaL folder you will find a reference to a document on this topic.

This tutorial explains how to properly set up a sample form.

We've noticed that many browsers still have bugs in their support for forms. If something that you create in this tutorial doesn't look the way it should when you display it with your favorite browser, the problem may be with the browser.

We suggest that for this exercise you create a new HTML document.

- Choose New from the File menu, or type Ctrl-N at the keyboard.
- Enter the HTML, HEAD, TITLE, and BODY elements as you've already learned to do.
- Now insert an H1 element, and type:

```
Buy my book!
```

- Insert a P element, and type:

```
Do I have a deal for you! Just click on the "Submit"
button in the form below to order any or all of these
best-sellers at a fraction of the regular cost!
```

(N.B.: This is not a tutorial on sales pitches!)

Actions

Now you're ready to start constructing a form.

- Insert a FORM element after the P element.
- With the insertion point inside the FORM element, choose Edit SGML Attributes... in the Markup menu.

This gives you a dialog box in which you will give a value for the ACTION attribute.

- In the ACTION text box, type:

```
http://www.sq.com/cgi-bin/quagmire
```

The "action" you've just specified refers to a program, located on SoftQuad's HTTP server, that can process the data entered in the form. At the end of the tutorial, you can submit the form to this program. In a "real-life" situation, you would probably specify a program on your own server, though in fact you can specify programs located anywhere on the Web.

- Set the attribute called METHOD to the value "GET".
- Click on the Apply button.

Another action that you can use is *mailto*: this causes the form to be e-mailed to a specified address. (You can try this later: for the purpose of this tutorial, please use the action described above.)

To make use of *mailto*:

- Inside the FORM element, choose Edit SGML Attributes... in the Markup menu.
- Set the ACTION attribute to the string "mailto:" followed by the e-mail address to which you want the form sent, e.g.,

```
mailto:charles@windsor.org
```

- Set the METHOD attribute to "POST".

Some browsers do not support *mailto*. Also, in order for this feature to work if your system is behind a firewall, you may need to configure your browser to use the correct proxy server.

Creating a text box

Now you'll enter the first element that generates a graphical widget in the browser:

- Enter a P element and type:

```
Name:
```

- Inside the paragraph, enter an INPUT element.

You can now preview the file to see what this looks like in a browser:

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.

Notice that the browser has placed a text box next to the word "Name:".

You're not done with this INPUT element yet.

- Move the insertion point inside the INPUT element.
- Choose the Edit SGML Attributes... command from the Markup menu, or type Ctrl-] at the keyboard.

This brings up a dialog box that lets you edit the *attributes* of the current element.

- In the text box for the attribute NAME, type:

```
cust-name
```

This value is used when the browser sends the form's data to the server, in order to identify which text box, drop-down list box, etc., a particular piece of data came from.

Another attribute, SIZE, is used if you want to specify the text box's length in characters.

Entering several lines of text

A text box in a form just lets you enter one line of text. If you need to allow your users to enter several lines of text at once (to enter an address, for example), you should use the TEXTAREA element.

- Insert a new P element and type:

  ```
  Address:
  ```

- After the text, insert a TEXTAREA element.
- Choose the Edit SGML Attributes... command from the Markup menu, or type Ctrl-] at the keyboard.
- Enter the following values for three of this element's attributes:

  NAME: cust_addr

  ROWS: 5

  COLS: 40

  ROWS and COLS specify the dimensions of the widget: 5 lines deep and 40 characters wide.
- Click on the Apply button.

  Note if you want a TEXTAREA to contain some default text, enter it between the start- and end-tags.

Now you may want to see how the browser renders this object:

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.

The browser generates a multi-line field, which may also have scroll bars.

Presenting a list of choices

Sometimes you will want the user to make one choice from a list of choices. In this example you'll see how to represent this with a drop-down list box.

- Insert a new P element, and type:

  ```
  Credit Card:
  ```

- After the text, insert a SELECT element.

HoTMetaL will ask you if you want to edit the attributes of this element.

- Click on the Stop and edit button.
- Enter "card-name" for the NAME attribute, and '1' for the SIZE attribute.
- Click on the Apply button.
- Inside the SELECT element, insert an OPTION element, and type:

  ```
  Visa
  ```

The OPTION element represents one choice in the a drop-down list box. The text you typed inside the element is a label that will appear in the list box.

- Insert two more OPTION elements after this one, containing the text "MasterCard" and "Amex", respectively.

To see what this looks like in the browser:

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.

Notice that the text you typed inside each of the OPTION elements appears as labels on the a drop-down list box.

If you wanted, you could have represented this list of choices as a scrollable list rather than a a drop-down list box. To do this, you would set the SIZE attribute to 2 or greater; this value would specify how many list items are shown at a time. If you want to be able to choose more than one item from this kind of list, set the MULTIPLE attribute to "MULTIPLE".

Just to complete this section of the form:

- Insert a new P element after the last one, and type:

```
Card number
```

- Insert an INPUT element.
- Choose Edit SGML Attributes... from the Markup menu, or type Ctrl-] at the keyboard.
- Set the NAME attribute to "card-num".
- Click on the Apply button.

Check boxes

So far you have used the INPUT element only to represent text boxes. Actually, this element can be used for a variety of purposes. One of these is representing *check boxes*: you would use this kind of box if you wanted the user to make a "yes/no" choice. (This is different from a *option button* (see below), which you would use when you wanted the user to choose one from a group of choices.)

- Insert a new P element after the last one, and type:

```
Check one or more titles:
```

- Insert a new P element after the last one, and type:

```
The Dentistry of Frederic Chopin
```

- Insert an INPUT element.
- Choose Edit SGML Attributes... from the Markup menu, or type Ctrl-] at the keyboard.
- Set the TYPE attribute to "CHECKBOX". (This is the attribute value that tells the browser that it should generate a check box.)
- Set the NAME attribute to "chopin".
- Click on the Apply button.

Now add two more titles in the same way:

- Insert a new P element after the last one, and type:

```
Motors and Such by Eddy Schneider
```

- Insert an INPUT element.
- Set the TYPE attribute to "CHECKBOX" and the NAME attribute to "motors".
- Insert a new P element after the last one, and type:

```
HTML for Travellers
```

- Insert an INPUT element.
- Set the TYPE attribute to "CHECKBOX" and the NAME attribute to "html".

To see what this looks like in the browser:

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.

Try clicking on the buttons with the mouse. You can turn on all, any, or none of the buttons.

Radio buttons

As we said above, you can also use the INPUT element to represent radio buttons. A group of radio buttons lets the user make one (and only one) choice from a group of choices.

- Insert a new paragraph after the last one and type:

```
Preferred language:
```

- Insert a new paragraph after the last one.
- Insert an INPUT element.
- Choose Edit SGML Attributes... from the Markup menu, or type Ctrl-] at the keyboard.
- Set the following attribute values:

  NAME: language

  VALUE: english

  TYPE: RADIO

  CHECKED: CHECKED

- Move the insertion point to the right of the </INPUT> end-tag and type:

```
English
```

Now add two more choices in the same way (keep all three choices in the same paragraph):

- Insert an INPUT element.
- Set the following attribute values:

  NAME: language

VALUE: french

TYPE: RADIO

- Move the insertion point to the right of the </INPUT> end-tag and type:

```
French
```

- Insert an INPUT element.
- Set the following attribute values:

  NAME: language

  VALUE: spanish

  TYPE: RADIO

- Move the insertion point to the right of the </INPUT> end-tag and type:

```
Spanish
```

Notice the following things about the attribute values you've just entered:

1. Each TYPE attribute is set to "RADIO". This tells the browser to generate a radio button.
2. Each NAME attribute has the same value, in this case, "language". This causes all three radio buttons to be in the same *group*, which means that the browser will allow only one of these three to be checked at once. If the form contains another group of radio buttons, the NAME attribute for all of its members must be the same, but different from the value for the current group.
3. The value of the VALUE attribute is sent to the server if the corresponding button is turned on when you submit the form, thus telling the server which button in this group was turned on.
4. The first INPUT element in this group has the CHECKED attribute set to "CHECKED". This tells the browser that this button should be turned on by default when the form is first displayed.

To see what this looks like in the browser:

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.

Try clicking on the buttons with the mouse. You can turn on only one of the radio buttons at a time. The button labeled "English" is initially turned on by default.

Reset

You can use the INPUT element to generate a button that restores all the form's controls (text boxes, radio buttons, etc.) to their default values:

- Insert a new P element after the last one.
- Insert an INPUT element.
- Choose Edit SGML Attributes... from the Markup menu, or type Ctrl-] at the keyboard.
- Set the following attribute values:

  TYPE: RESET

  VALUE: "Reset defaults"

To see what this looks like in the browser:

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.
- Enter some data in the form.
- Click on the Reset defaults button in the form.

The form's controls revert to their default values. The text boxes are all blank and the check boxes are turned off. The radio button labeled "English" will be turned on.

Submitting the form

Your form is almost complete. All you need to do is create a button that causes the browser to submit the form.

- Next to the INPUT element for the reset button, insert another INPUT element.
- Choose Edit SGML Attributes... from the Markup menu, or type Ctrl-] at the keyboard.
- Set the following attribute values:

  TYPE: SUBMIT

  VALUE: "Submit order"

To see what this looks like in the browser:

- Save the file.
- Choose Preview from the File menu, or type Ctrl-M at the keyboard.
- Enter some data in the form. (Don't enter a real credit card number!)
- Click on the Submit order button in the form.

The form will now be submitted to a program on SoftQuad's web server. This program doesn't actually process an order, it just echoes back the information that the browser sent it. This information will appear in the browser window: you can return to the form by clicking on the button that takes you to the previous document.

The information is presented in pairs containing a "name" (corresponding to the NAME attribute of the text box, check box, etc.) and a "value" (for text boxes or "text areas", this will be the data you typed in; for check boxes the

value "on" is submitted—check boxes that aren't turned on are ignored; for radio buttons the value of the VALUE attribute is sent).

You may notice a couple of unusual things about the format of the text: spaces are replaced by a '+' sign, and some special characters (notably newline, '=', and '&') are replaced by '%*nn*', where the *n*'s are digits from 0-9 and/or letters between 'A' and 'F'. This is the standard form that browsers use for submitting data to the server.

HTML quick reference

The authoritative source of information on the structure of HTML documents is the document *HyperText Markup Language (*HTML*) Version 2.0*. This section provides a short summary of this material. The rules governing the HTML format are quite flexible, and furthermore HoTMetaL will guide you through the document structure: therefore, the approach followed here will not be to enumerate all the possible combinations of elements. Rather, an overview of the structure will be presented, together with a discussion of the different groups of elements (emphasis, links, lists, etc.).

Overview

- An element called HTML surrounds the whole document.
- This element contains two sub-elements, HEAD and BODY. These elements are required.
- HEAD has sub-elements that define header material:
  - TITLE: document title. This element is required.
  - BASE: can be used to record the document's URL. The URL recorded here may be used to resolve a "partial URL", or used if the document is accessed "out of context".
  - ISINDEX: indicates to the browser that the document is an index document. This is used only if the document is on a server that does indexing.
  - LINK: indicates a relationship between this document and some other object.
  - META: gives information that appears in HTTP headers.
  - NEXTID: used to generate a unique identifier.
- Inside the BODY element, heading elements (H1 through H6) can be used to delimit major divisions of the document (headings are not mandatory, however). Headings are permitted to appear in any order, but you will obtain the best results when your documents are displayed in a browser if you follow these guidelines:
  - H1 should be used as the highest level of heading, H2 as the next highest, and so forth.
  - You should not skip heading levels: e.g., an H3 should not appear after an H1, unless there is an H2 between them.

Block formatting

- The major divisions of a document body's structure comprise the following elements:
  - ADDRESS: if you want to include the address of the author of the document, enter it inside this element.
  - BLOCKQUOTE: used for quotes from another source, requiring special block-style formatting.
  - CODE: code samples.
  - P: paragraphs.
  - PRE: pre-formatted text. You would use this text when you want the browser to use the same line breaks and spacing that you entered in the document. The text will be formatted by a browser using a fixed-width typewriter font.
  - DL, DIR, MENU, OL, UL: list elements (see *Lists*, below).

Character formatting

The following elements are used primarily for formatting inline text:

- B: bold.
- I: italic.
- EM: browsers *usually* represent this element in italic.
- STRONG: browsers *usually* represent this element in bold.
- TT: characters inside this element are formatted with a fixed-width typewriter font such as Courier.
- SAMP: literal characters.

Line breaks

If you want to force a browser to break the current line in the text, insert a BR element. You can't type inside this element: it just causes a line break.

Horizontal lines

To cause the browser to print a horizontal line (rule) in your document, insert an HR element. HoTMetaL will display a line in your document, but individual browsers may print lines of different widths and lengths.

List elements

HTML supplies five list elements. With the exception of DL, list elements are composed of one or more LI (list item) elements.

You can *nest* lists by inserting a UL, OL, etc., inside a list item (LI).

- OL: ordered list. Items in this list are numbered automatically by the browser. The numbering will reflect nesting levels.
- UL: unordered list. Items in this list are prefaced by a list mark such as a bullet. Browsers will usually vary the list mark to take account of nesting.
- DIR: directory list. This is an unordered list. Each LI element in this kind of list should no longer than 24 characters.
- MENU: menu list. This is an unordered list. Each LI element in this kind of list should be no longer than one line.

- DL: list of definitions. This is an unordered list. This kind of list is different from the others. Each "item" in this kind of list consists of one or more terms (DT elements), followed by a definition (DD element).

Link elements

It is normal for HTML documents to contain links to other documents, which can be located anywhere on the WWW. These links are provided by *Universal Resource Locators* (URLs), which are identifiers that name the location and filename of a document, and the protocol used to access it.

The following elements represent links to other documents:

- A: anchor. The HREF attribute of this element represents a URL. If this attribute has a value, the content of the element will be highlighted when the document is displayed in a browser window, and clicking on this content will cause the browser to attempt to open the file specified by the URL.
- IMG: image. This element represents a graphic image. It is typically used for inline images—you should be aware that some browsers may not be able to display such images. (In this case, the text, if there is any, given in the ALT attribute may be shown.) The SRC attribute represents a URL. See the sections on the Show Image and Show Inline Image commands in the View menu for information on displaying images in HoTMetaL.

HoTMetaL has three commands for working with URLs:

- Edit SGML Attributes... in the Markup menu provides a mechanism for editing the URL (if there is one) in the current element. See the tutorial section for an example of using this command.
- Publish... in the File menu is used if you want to change the URLs in the document from identifiers that refer to files on your local system to identifiers that refer to publicly-available files on one or more WWW servers.
- Show Link and Context View in the View menu is used to display the URL (if there is one) in the current element.

Links within the same document

This topic is covered in an extended example in the tutorial section.

In general, you can set up a "source" and "destination" anchor pair by setting the NAME attribute of the destination anchor to '*string*' and setting the HREF attribute (i.e., the URL) of the source anchor to '#*string*'. This sets up a one-way link. You can set up a two-way link by editing the two anchors so that each one's NAME attribute corresponds to the other's URL.

'Hot images'

A 'hot image' is used like an anchor—when you click on the image, the browser retrieves a document.

This is very easy to accomplish: you just need to insert an IMG element inside an A element. Each element will have a URL: the IMG's URL locates the image, and the A's URL locates the file that is retrieved when you click on the image.

Images with hot spots

Sometimes you will see images that have several "hot spots" that you can click on and cause different documents to be retrieved. This is accomplished by means of a *clickable image map*, a file that tells the browser where the hot spots are. To prepare such a file, you'll need other tools besides HoTMetaL, and so we won't discuss the procedure here. Rather, you should retrieve the document "Overview on using Clickable Image Maps " by opening the file *faq.htm*, located in the HoTMetaL installation folder, with a browser and clicking on the appropriate anchor.

Forms

The following elements are used to construct forms that the user can fill in and submit (e.g., via e-mail). When your document is browsed, the browser will generate the appropriate graphical widgets.

- FORM: the top-level element for a form.
- INPUT: generates a text field, button, radio button, or check box.
- SELECT: represents a group of choices that a user can make. Generates a pop-up menu or scrollable list.
- OPTION: one choice in a SELECT group.
- TEXTAREA: generates a field that allows the user to enter several lines of text.

For more information on forms, see the tutorial on forms in this chapter. You can also retrieve the document "Information on setting up form functionality" by opening the file *faq.htm*, located in the HoTMetaL installation folder, with a browser and clicking on the appropriate anchor. To see an example of a form, use Open Template... to open the template *custreg.htm*.

Other elements

The elements in this section do not fit into the categories described above.

- CITE: represents a document citation.
- KBD: used to display text that a user would enter at the keyboard. (This would be used in technical manuals, for example: it is not similar to an element such as INPUT, used in forms.)
- VAR: represents a variable name.

Obsolete elements

The elements PLAINTEXT, XMP, and LISTING are obsolete and are supported in this release of HoTMetaL only for compatibility with older documents. Using these elements in new content is not recommended. XMP and LISTING should be replaced by PRE.

The obsolete elements COMMENT and HP are not supported.

Proposed elements

The proposed elements DFN, STRIKE, and U are not supported by HoTMetaL.

If you see something you like...

If you see a web page that contains a typographical effect, form, etc., that you like, then the easiest way of achieving the same thing yourself is to save the file with the browser (make sure you save it in HTML format) and then open it with HoTMetaL.

For further information...

The file *faq.htm* in the HoTMetaL installation folder contains titles and URLs for documents that contain information on HTML usage. Open this file with a browser and retrieve the documents that you're interested in.

A number of HTML tutorials, of varying quality, are usually available on the web. The relevant Usenet newsgroups (those in the *comp.infosystems.www* hierarchy) are also a source of information.

Commands for inserting markup

The most common operations you will carry out in order to add or change markup are:

- The Insert Element... command in the Markup menu inserts a new, empty element in which you can type text or insert other elements.
- The Surround... command in the Markup menu lets you surround a selection with an element: if some part of the document should be contained in a particular element, then you can highlight that portion and select this command in order to choose an element to surround it with.
- The Change... command in the Markup menu lets you change the markup: if you want to tag part of the document with a different element you can select this command to get a list of valid elements to replace the current element.

Screen formatting

HoTMetaL provides screen-formatting capabilities that facilitate the document creation process by allowing you to assign distinctive styles to the elements in your document. *The purpose of these formatting features is only to improve the appearance of the document during the editing process. The formatting that you set with HoTMetaL does not affect how browsers such as Mosaic format the document.*

Because HTML files are structured documents, setting a style for an element means setting it for all elements of the same type.

The following kinds of typographical properties can be set:

- *Character-based properties*: font family, font size, font style, line height, justification, fill mode, and format type. All of these properties are set using the Character... command. (Font style allows you to adjust the font by making it bold, superscript, etc.; the fill mode determines how carriage returns are treated—in *fill mode*, they are treated like spaces, but in *no fill mode* they cause a line break; format type lets you choose whether an element appears inline or starts on a new line.)
- *Separation*: using the Separation... command, you can set off elements by adding space on top and on bottom. This command also lets you cause an element to be formatted as if it started with a tab.

Styles

HoTMetaL stores its formatting information in files called *styles files*: these contain the formatting information set with the Character... and Separation... commands.

HoTMetaL maintains two kinds of styles files. There is one *default styles file*, which is used whenever a file is created or opened with HoTMetaL. This is a *binary* file called *html.stl*. When an HoTMetaL file is created or opened, HoTMetaL looks for this styles file in the *styles path*, that is, the set of directories named by the *styles_path* configuration variable. If it finds the styles file, then the formatting information contained in that file is used for the current HoTMetaL file. If the styles file is not found, then a new styles file, containing the default formatting information, is created in the first directory in the styles path. Whenever the current HoTMetaL document is saved, this styles file is updated with whatever formatting parameters are in effect for the document.

HoTMetaL also uses styles files in text format. These styles file are loaded using the Load Styles... command. You can switch styles in the middle of a HoTMetaL session by choosing a new text styles file with this command. The formatting information from a text styles file will not be written to the default (binary) style unless the current file is saved. Text styles files are created using the Save Styles... command. You can maintain as many text styles files as you wish.

Displaying images

The Show Image command in the View menu lets you launch an external application (which you can choose by means of a configuration variable) to display a graphical image. By default, GIF images are displayed inline (in the HoTMetaL document window). See the Show/Hide Inline Images command in the View menu for more information

Previewing the document

In order to preview the current document, you can invoke a browser of your choice from within HoTMetaL. To do this, just invoke the Preview command in the File menu. In order for this to work, you *must* specify a browser in the HoTMetaL preferences file. See the section on this command for more information.

Displaying icons

Using the Show/Hide Tags command in the View menu, you can cause the special character icons and the tag icons that represent elements to be visible or invisible.

Displaying a document outline

The command Show Structure View in the View menu displays a nested outline view of the document. You can cut, copy, paste, and navigate in this outline. For more information, see this command's documentation in the chapter *The View menu*.

Newlines

Although pressing the Return or Enter key will put a "newline" into the file, Mosaic and other browsers follow the SGML whitespace rules, and therefore will ignore new line characters in some circumstances.

Attributes

Elements can have *content* (text and sub-elements contained in the element) and *attributes*. An attribute is a piece of information about the element which does not appear in the content of the element. Some common uses for attributes are: to represent a link (URL); naming an image file; choosing different types of graphical controls for an on-line form.

Viewing attribute values

There are a number of ways of telling whether or not an element has attributes, and of seeing what the attribute values are.

- If the current element has one or more attributes, then the Edit SGML Attributes... command in the Markup menu is activated (whether or not the attributes have been given values). If you invoke this command, you can see what the attribute values are and also edit the values.
- The *context view*, which is displayed by invoking the Show Link and Context View command in the View menu, displays the sequence of elements that contain the current element. It also displays the attribute values for any elements in the sequence that have attributes.
- The styles file supplied with HoTMetaL allows you to display attribute values inline, as prefixes displayed just to the right of the start-tag icon.

Editing attributes

Attribute values may be inserted or changed using the Edit SGML Attributes... command in the Markup menu. When you invoke this command, you will get a dialog box that contains a line for each attribute of the current element. Each line contains the attribute name followed by either a text box or a drop-down list box, depending on what kind of value the rules file says the attribute must have. The attribute value may be text (which could be subject to restrictions such as the length of the text or first character in the text), or a selection from a list of values.

Netscape support

We have included a rules file that allows you to use the extended features (as best as we can determine them) supported by the Netscape browser. You must modify the configuration file to name the alternate rules file if you want to use these features.

- Edit the configuration file, and locate the line assigning a value to the *rules_file* variable.
- Put a '#' character at the beginning of this line. This "comments it out" so that HoTMetaL will ignore it.
- Now insert the line:

```
rules_file=html-net.mtl
```

- Save the file and restart HoTMetaL.

If you want to restore the previous rules, "comment out" the line you just inserted and remove the '#' from the original line.

You should be aware that the Netscape extensions to HTML are not part of the HTML 2.0 specification. Web users viewing documents made with this alternate rules file will not notice the effects of the extensions when they view the file with a browser other than Netscape (e.g., text in the BLINK element will not blink, an IMG element with the attribute ALIGN=RIGHT will not align the image to the right, etc.)

For HTML markup that conforms to the HTML 2.0 specification, you should continue to use the default rules file (*html.mtl*). HTML markup created with this rules file can still be used with Netscape (but will not take advantage of the extended features).

The File menu

The File menu contains commands for creating, opening, closing, and saving files edited with HoTMetaL, previewing the current document with a browser, and modifying URLs.

New

When you invoke this command a new, empty HoTMetaL document is displayed in a document window.

Open...

Opens a previously saved file.

HoTMetaL presents you with a dialog box allowing you to open a file. The dialog box is called a *file selection dialog*; a similar dialog box appears when you select the Save As... command. The structure and function of the file selection dialog box for the Open... command is explained here.

The dialog box has several parts:

- A drop-down list box labeled List Files of Type. This list box lets you choose whether to display files with the default file extension (*.htm*) or display all files in the current directory.
- a text box labeled File Name

  This text box can contain a relative or absolute path name, which terminates in a file name or directory name, and can optionally start with a drive name. A file name (which may contain an extension) can contain the following "wildcard" (special) characters:

  - * (asterisk): matches any sequence of characters in a file name
  - *?* (question mark): matches any single character in a file name

  Such a pattern containing wildcard characters is used to filter the file names displayed in the list box directly below. You can type a pattern in the box directly or choose it from the List Files of Type list.

  If the File Name box contains a path name that ends in a file name without wildcard characters, clicking on [OK] will cause that file to be opened.

- an information field labeled Directories

  The current directory (including the drive name) appears underneath the label. The default directory is the one specified first by the *import_path* configuration variable.

- a list box underneath the File Name text box

  This list contains the files in the current directory that match the pattern in the File Name box. The list of files is updated whenever a new directory is chosen, and when you click on the OK button.

  Clicking once on a file in this list causes the name to go in the File Name box. Double-clicking causes the file to be opened. This has the same effect as clicking once on the file name and then clicking on the OK button.

- a list box, on the right side of the dialog, underneath the Directories information field

  This box allows you to navigate in the directory structure of a drive by double-clicking on the directories shown in the list. The top level "directory" displayed is the current drive; if you double-click on this directory, its subdirectories are displayed, slightly indented to indicate the nesting relationship. If you click on one of these directories, its subdirectories will be displayed, and so forth. At any particular time the list will display the sequence of directories that you have navigated along, ending with the subdirectories of the last directory you selected.

- A drop-down list box labeled Drives. This allows you to choose which drive to navigate.

By pressing Tab or Shift-Tab you can make the File Name box, the list of directories, the list of files, or either of the buttons the active item in the dialog box. When either of the lists is active, you can select a list item by pressing repeatedly on the first letter of the item until it is selected. When the File Name box is active, you can enter text in it.

In summary: you may select a directory from which a file may be opened by using the list of directories, or by typing the path name in the File Name text box. You may choose a file by doing one of the following:

- double-clicking in the list of files
- selecting a file in the list of files and then clicking on OK
- entering the file name in the File Name text box and clicking on OK

> **Note** In this dialog box, the default directory is the one specified first with the 'import_path' configuration variable.

Error checking

As the file is being opened, HoTMetaL checks for fatal SGML errors. Fatal errors include start-tags without matching end-tags, invalid element names, and many other SGML errors. In such cases, HoTMetaL displays a message describing the problem. It then gives you the following choices:

1. Import Through Filter, which allows you to run the file through a filter that may correct the errors. If you choose this option you will get a dialog box in which to give the name of an output file. Enter the output filename (including drive and directory) in the Output File text box, or click on the Choose File... button to get a file selection dialog box with which to choose a file. HoTMetaL will then invoke the filter: when the filter has finished, HoTMetaL attempts to open the output file.
2. Open the file as a text file so that you can correct the errors manually. When you've done this, you can run the Interpret Document command, which performs the equivalent of Open... on the text document.
3. Cancel the operation and correct the error(s) through other means.

If no errors are found, the file is formatted, and checked once more for errors as if the Turn Rules Checking On command in the Special menu had been selected for the new file. At this stage, non-fatal errors may be detected. Examples of these are incorrectly placed elements, and text at a point where no text is permitted. Errors of this kind do not prevent the file from being opened.

Finally the file is *validated*: this stage of error checking ensures that the HTML markup is correct and complete. The following example illustrates the difference between *rules checking* and *validation*: if you open a file that has an HTML element that does not contain a HEAD element, rules checking will not complain, because you have not yet violated the rules file. Validation, however, will alert you to the fact that the required HEAD element is missing.

> **Note** There will sometimes be a document type declaration (DOCTYPE) at the top of an HTML file, specifying which DTD to use. This declaration is ignored with files being opened with HoTMetaL, because all HTML files use the HTML DTD.

Open Template...

This command allows you to work with templates, which are pre-defined structures for documents. Templates are used as forms or document outlines that you can enter text into without having to insert any of the markup yourself.

Opening a template

To open a template, click on the Open Template... command. This brings up a file selection dialog box labeled Open Template. If you have a templates directory (see below) the dialog box will display the files from that directory Each file corresponds to a template: to open a template, just open it from this dialog box as you would any other file. The document name in the title bar will be the same as the template name, but with a number added to the first part of the filename: for example, the first time you open a template called *fax.htm*, the new document will be called *fax1.htm*, the second time the document will be called *fax2.htm*, and so forth.

When the template file is opened you can enter text or elements into it, and later save the file. When you save the file, you will have use the Save As... command and choose a new file name—the name in the title bar is not automatically adopted. If you save this file in the templates directory make sure you do not overwrite the original template file by mistake.

Creating your own templates

A number of templates are shipped with HoTMetaL, but it is expected that you will normally be working with templates that were created at your own site.

Templates directory

In order for you to work with templates successfully, a directory must be designated as the *templates directory*. This is a central location containing all of your template files, and it is the directory whose files are displayed when the Open Templates... dialog box comes up. By default, this is the directory called *tmplts* in the directory where HoTMetaL is installed.

If you want to use another directory for this purpose, you will have to name that directory using the *templates_path* configuration variable. For example:

```
templates_path=c:\susan\tmplts
```

If the default templates directory does not exist (perhaps someone has removed it), and no alternative directory is specified with the *templates_path* variable, the current directory will be used as the templates directory.

Creating templates

To create a template file with HoTMetaL, you should do the following:

- Create a document as you normally would.
- Invoke the Save As... command in the File menu.
- Choose a directory and filename. You can save the file directly in the templates directory or move it there later.
- Click on the [Save As] button.

Installing a template

When the template file has been created, it should be saved in the templates directory (or you can save it elsewhere and move it later) so that it will be easily accessible from the Open Template dialog box.

Save

This command saves the current file (that is, the file that is opened in the active document window) to the disk.

HoTMetaL saves the document in the file name shown in the title bar at the top of the window.

Save options

There are several *save options* that you may set if you need to do so. All of these options are set using configuration variables:

- You may choose to save a *document type declaration* (DOCTYPE), at the top of the file. The default may be set by setting the *export_doc_type_dec* configuration variable to YES or NO:

  ```
  export_doc_type_dec=YES
  ```

- You may choose to save an SGML declaration at the top of the file. The default may be set by setting the *export_sgml_dec* configuration variable to YES or NO.

  Note Normally you should use this feature only if your file will be used by some other SGML system. Some browsers may not be able to read an SGML declaration, or may display (unwanted) information from the declaration on the screen.

- If your file contains special characters (those outside the ASCII range 0-127), you may choose to have these converted to SGML *character references* when the file is saved. This situation arises only if your file was edited with another editor. Special characters that are entered in the document when it is being edited with HoTMetaL will be converted immediately to character entity icons. Character references are represented in a HoTMetaL document as a small, rectangular icon, similar to a character entity icon, labelled '#nnn', where the *n*'s are numbers. For example:

. Browsers should display a character reference as the character itself. The default setting for this option (YES or NO) may be set with the *export_convert_special_chars* configuration variable.
- You can choose the character(s) that HoTMetaL uses to mark the end of a line in the saved file. You may choose one of the values MAC, UNIX, and MSDOS, which will cause the end-of-line character to be carriage return, line feed, or carriage return and line feed, respectively The default end-of-line marker can be set with the *export_eol* configuration variable. If you don't provide a value for this variable, the default will be MSDOS.
- For elements that are formatted in *fill mode* (see the documentation on the Characters... menu item) you can set the length of lines in the saved file by telling HoTMetaL to insert end-of-line characters after a specified number of characters. This option can be turned on or off by

default by setting the *export_add_line_breaks* configuration variable to YES or NO. The number of characters in a line can be set with the *export_max_line_len* configuration variable, e.g.,

```
export_max_line_len=75
```

HoTMetaL will not cause a line break in the saved file to occur between an element and adjoining text.

If you have rules checking turned on, the file will be validated, and you will be warned if there are errors and asked if you still want to save. If you do, the file will be invalid and HoTMetaL may have trouble opening it in the future.

Save As...

This command lets you choose a new name for the current document. When you save a file with this command, HoTMetaL creates a *new* file whose content is the same as the current file, and closes the *current* file, leaving the new file open.

The document that was closed will look the same as it did the last time it was saved: any changes that were made since the last save will be saved in the newly created file.

HoTMetaL gives you the file selection dialog box with which to specify the name of the new file.

You should follow the same instructions for selecting a file or directory as were described in the section on the Open... command.

> **Note** In this dialog box, the default directory is the one specified first by the 'export_path' configuration variable.

The save options that were specified for Save will also apply to the Save As... command. Save As... will validate the file if rules checking is turned on, just as Save does.

Close File

This command closes the current file. If the file has had changes made to it since it was last saved, you will be prompted to save the changes before closing it.

Preview

When you invoke this command it will launch a browser to display the file. If the file hasn't been saved since changes were last made to it, HoTMetaL will prompt you to save the file. You then have the choice of saving the file or proceeding with the previewing operation without saving (in which case the document is saved to a temporary file automatically). The command line for the browser that is launched by this command is specified using the *html_browser* configuration variable. The default value for this variable (and therefore the default browser) is *c:\mosaic\mosaic.exe*.

Invoking this command does the same as if you had saved the file with HoTMetaL, launched the browser independently of HoTMetaL, and then opened the file with the browser.

Publish...

This command is a form of "find and replace" for URLs. Before a completed HTML document is moved to a WWW server, all URLs should refer to documents that are available on some WWW server. (While the document is being created, they may refer to documents on your local system.) The Publish... command gives you the opportunity to edit all the URLs, modifying them if necessary.

For example, when you are creating a document the URLs may consist of local filenames such as:

```
file:///c|/rodney/orwell/homage.htm
```

When the document is placed on your server, you must substitute URLs that refer to documents that are available on your server or some other server. For example:

```
http://www.sq.com/orwell/homage.htm
```

When you invoke Publish... you will get a dialog box containing two text boxes.

The first box (labeled Change URLs From) contains a part of the URL that you want to change; the second box (labeled To) contains the string that you want to change it to. The default values in these two boxes are specified by two configuration variables: *publish_change_from* specifies the part of the URL that should be changed; *publish_change_to* specifies the new value for this part of the URL.

If there were a large number of URLs for which you needed to change a local directory such as *file:///c|/rodney* to a directory on the server, such as *http://www.sq.com*, you could set your configuration variables as follows:

```
publish_change_from=file:///c|/rodney
publish_change_to=http://www.sq.com
```

This would cause the Change URLs From text box to contain "file:///c|/rodney" and the To text box to contain "http://www.sq.com".

Finding and Replacing URLs

When you click on the Find Next button, HoTMetaL finds the next element that has an attribute representing a URL (often the HREF attribute of the element A and the SRC attribute of the element IMG). The search starts at the insertion point (or selection).

When an element with a URL is found, the insertion point is placed inside that element, and the document scrolls to its location.

If the URL contains the text in the Change URLs From box, clicking on the Replace button will change it to the text in the To box.

Clicking on the Replace All button will make this change for all URLs in the document that contain the Change URLs From text. This also causes the dialog box to be dismissed.

You can edit the Change URLs From and To text if you want to perform substitutions other than the default one.

The searching performed by this command does *not* wrap around from the bottom to the top of the file.

Exit

Quits HoTMetaL. If an open file has been changed since the last time it was saved, you will be prompted to save the file before exiting.

The Edit menu

The Edit menu contains commands to cut, copy, and paste a selection, undo your last action, perform find and replace operations, and do spell checking.

Undo

Allows the effect of the *last* operation to be undone.

Most commands can be undone. There are, however, some HoTMetaL actions that cannot be undone:

- any command from the File menu, except Publish
- scrolling and windowing commands
- text selection
- Undo itself (it can be undone with Redo)
- Show Structure View, Show Link and Context View
- Any actions performed prior to the last time the document was saved cannot be undone.

If you imagine a sequence of undo-able commands as a list, successive Undo commands will proceed through the list, starting at the most recent. Therefore, if you execute two Undo commands in a row you will undo the most recent action, and then undo the second most recent action. Note that since Undo is not itself an undo-able command, one Undo cannot undo another. This function is reserved for Redo, which is the inverse of Undo. (See the section on Redo.)

If you undo a Copy or Cut command, the previous contents of the clipboard will be restored.

Undo limit

The number of previous commands that can be undone is not limitless but rather is controlled by an *undo limit*. Once this limit is reached, each successive command will cause the oldest undoable command to be committed, that is, it will no longer be possible to undo that command. For example, if the undo limit is set to 1 and you Cut a selection and then Paste what you just cut, you will be able to undo the paste but not the cut. If the undo limit were 2 or greater, both the cut and paste could be undone. The default value for the undo limit is 10, but this can be changed by setting the *undo_limit* configuration variable, e.g.:

```
undo_limit=20
```

Redo

This command allows the most recent undone command to be redone.

Redo operates in a way similar to Undo: a sequence of Redo commands re-does the most recent redo-able commands (i.e., commands that have been undone) in reverse order. Redo and Undo are inverses of each other: the net effect of an Undo and its corresponding Redo is to cause no change to the document.

If an undoable action is performed after a series of one or more Undo commands then the Undo commands will no longer be redo-able.

To illustrate how Redo works, suppose you Cut a selection in a document, and then Paste that selection somewhere else. If you perform two Undo commands, first the Paste and then the Cut will be undone. If you then execute a Redo, the Cut will be redone. A second Redo will then redo the Paste.

Cut

Removes the current selection from the document and places it in the clipboard. Any previous contents of the clipboard will be erased. The selection can then be pasted.

This command is used when you want to remove a section of text that will probably be pasted in elsewhere, in the same or another document.

Copy

Copies the contents of the current selection into the clipboard and erases any previous contents. The document is left unchanged.

This command is used when you want to duplicate a portion of the document without erasing it. The copied selection can be inserted elsewhere using the Paste command.

Paste

Transfers the contents of the clipboard to the document. If the document contains an insertion point, the clipboard is pasted at that point; if it contains a selection, then the contents of the clipboard overwrite the selection.

The clipboard can contain markup. If the paste would result in an incorrectly marked-up document, you may be prompted to either cancel the paste operation or continue with rules checking turned off. Some paste operations cannot be performed even with rules checking turned off.

Find and Replace...

Allows text, elements, and patterns to be found and replaced.

You are presented with a dialog box that allows you to enter various values and parameters.

Specifying the search and replace strings

The Find text box allows you to specify a *search string* of text characters, elements, character entities, or patterns. If the document contains a selection when you invoke Find and Replace... the selected text will automatically become the search string. If the selected text is longer than 255 characters, it will be truncated. If the selection contains a markup icon (an element or character entity) it will be truncated at the last character before the icon. A selection that starts with a markup icon will become a null search string, and therefore an existing selection cannot be used to cause an element to be the search string.

The Replace text box allows you to specify a *replace string* consisting of text characters, patterns, an element, or a character entity, with which you want to replace the search string.

The Find In text box allows you to restrict your search to a particular element.

The Find, Replace, and Find In strings are described in more detail below.

Command buttons

There are five buttons (including Cancel) along the bottom of the Find & Replace dialog box, which allow you to carry out several search and replace operations.

Find

Find causes HoTMetaL to search through the document for the search string according to the various search parameters chosen. If you click on Find and the search is successful, HoTMetaL selects the text, character entity, or element that was found and scrolls to the selection. If the search fails, HoTMetaL will beep. Also, a Not found message will be displayed at the bottom of the Find & Replace dialog box.

Text searches will not match if parts of the search string are found within different elements. If you are searching for 'Fred and Barney', but the word 'and' is in a separate element (emphasized, for example), the search string will not be matched.

Replace

Replace replaces the current selection in the document with the replace string. This command is enabled only when part of the document is selected.

Replace then Find

Replace then Find replaces the current selection in the document with the replace string, and then resumes the search procedure. This command will be enabled only when part of the document is selected.

You would use this option if you wanted to manually examine each occurrence of the search string before doing a replacement: if you decide to perform the replacement, click on Replace then Find again; otherwise, click on Find to go to the next occurrence of the search string.

Replace All

Replace All replaces all occurrences of the search string with the replace string. This command automates the whole find and replace process, not giving you the opportunity to choose individual cases. It is possible that some of the occurrences of the search string that are found cannot be replaced, because this would cause an incorrectly marked-up document. If so, these occurrences are skipped over. After the operation has been completed, a message will be displayed in the Find & Replace dialog box showing how many occurrences of the search string were found, and how many were replaced. The insertion point will now be at the end of the last replacement. Invoking the Undo command after Replace All will cause *all* of the replacements that were actually made to be undone.

Specifying search patterns

When the Find Patterns option is on (see below), the characters you type in the Find text box are interpreted as patterns by HoTMetaL: that is, the search string can contain certain special search characters that allow the search string to match a class of strings, or markup constructs. If your search string does not contain any special search characters, HoTMetaL will search for exactly the text you have typed. On the other hand, if the search string does contain special search characters, it defines a pattern of characters to be matched. For example, the search character '.' (period) is used in the pattern

```
m...y
```

to match a sequence of five characters beginning with 'm' and ending with 'y', e.g., the words 'money', 'marry', 'murky', etc. A complete description of search characters and expressions appears below.

The following characters are special search characters in a search pattern:

```
\ . * ? + ^ $ ]
```

In addition, the characters '&' and '<' are special when one or the other appears as the *first* character of the pattern.

If you want to search for any of these as ordinary characters when Find Patterns is turned on, it must be preceded by a backslash. For example,

```
\.
```

is used to match a period.

Search patterns are sequences of ordinary characters and special characters, combined according to certain rules.

The following list summarizes how these search characters are interpreted, and how search patterns are formed.

- An ordinary character represents itself.
- A string beginning with a '<', immediately followed by an element name (and possibly attributes and content, as discussed below), is used to match an element.
- A period or dot, '.', represents a single, arbitrary character (including a blank). So

  ```
  fo.d
  ```

  would match 'food', 'ford', 'fond', 'fold', etc. Similarly,

  ```
  s.o.
  ```

  matches 'stop', 'shot', 'snow', etc.
- A single character, or a string enclosed in parentheses, followed by an asterisk, '*', matches zero or more occurrences of that character or string. For example,

  ```
  l*ama
  ```

  would match 'ama', 'lama', 'llama', 'lllama', etc.

```
b(an)*a
```

would match 'ba', 'bana', 'banana', and so on. It is possible to combine the '*' with '.' to match arbitrary strings of characters. So

```
s.*ch
```

matches 'search', 'such', 'stretch', 'stopwatch', as well as 'sch' and 'skip lunch'. This search pattern represents strings that start with 's' followed by zero or more occurrences of an arbitrary single character (it doesn't have to be the same character over and over) followed by the characters 'ch'. Since the period can match a blank space, this pattern can match a multi-word string.

- A single character, or a string enclosed in parentheses, followed by a question mark, '?', matches zero or one occurrences of that character or string. For example, to search for instances of both 'color' and 'colour', you would use:

```
colou?r
```

- A single character, or a string enclosed in parentheses, followed by a plus sign, '+', matches one or more occurrences of that character or string. For example, the following expression matches 'ben', 'been', 'beeen', and so forth, but not 'bn'.

```
be+n
```

- Search patterns may be enclosed within parentheses for grouping.
- Search patterns separated by a vertical bar, '|', match any string that matches either of the patterns. For example, if you wanted to search for either 'love' or 'money', you would use the expression:

```
love|money
```

In a more complex example, you could combine two search patterns given above:

```
s.*ch|fo.d
```

- A caret, '^', at the very beginning of a search pattern means that text will match the pattern only if it immediately follows markup (a start- or end-tag, or a character entity). Such text must not be separated from the markup by white space. Anywhere else, the caret is not treated as a special search character (except in sub-strings, see below). For example, if you wanted to search for the word 'Note' immediately following markup, you could use:

```
^Note
```

- A dollar sign, '$', at the very end of a search pattern means that text will match the pattern only if it is immediately followed by markup. Such text must not be separated from the markup by white space. Anywhere else, the dollar sign is not treated as a special search character. For

example, if you wanted to search for the word ‘sub’ immediately preceding markup, you could use:

```
sub$
```

- A pair of square brackets, ‘[’ and ‘]’, around any string of characters defines a *sub-string* that matches any *one* of the characters between the brackets. For example,

  ```
  an[dy]
  ```

  matches ‘and’ and ‘any’.

  By contrast, a string of characters preceded by a caret, ‘^’, within brackets, matches any character *not* in the string. For example,

  ```
  th[^ei][a-z]*
  ```

  matches any word that begins with ‘th’, which is not followed by an ‘e’ or ‘i’. It would match ‘that’ and ‘thought’ but not ‘therefore’ or ‘this’.

- A sub-string, within square brackets, of the form

  ```
  [char1-char2]
  ```

  matches any character in the range of ASCII characters beginning at *char1* and ending at *char2*. For example, the sub-string

  ```
  [e-p]
  ```

  matches any lowercase letter between ‘e’ and ‘p’, inclusive. The substring

  ```
  [A-Za-z]
  ```

  matches any upper- or lower-case letter. Note that if searching is not in case-sensitive mode (see below), no distinction between lower case and upper case letters is made in character ranges. In this case, for example, the character range

  ```
  [a-z]
  ```

  would match any upper- or lower-case letter.

- An expression of the form

  ```
  [^char1-char2]
  ```

  matches any character *not* in the range of ASCII characters beginning at *char1* and ending at *char2*.

- A range can occur inside a sub-string. For example, the pattern:

  ```
  [ac-fh]
  ```

  matches any of ‘a’, ‘c’ through ‘f’, and ‘h’.

- If you wish to use any of the characters ‘^’, ‘]’, or ‘-’ as a literal character within a sub-string rather than as a special search character, there are certain rules you must follow.

- The '^' character is special only if it occurs as the first character in a sub-string. Otherwise, it's treated as a literal character. E.g.,

  ```
  [joy^]
  ```

  will match any of the characters 'j', 'o', 'y', and '^'.
- The character '-' is a special search character if it occurs *between* other characters in the sub-string. If it occurs at the beginning or end of the sub-string, it is a literal character. For example, the sub-string

  ```
  [a-]
  ```

  will match 'a' or '-'.
- The character ']' terminates a sub-string unless it occurs as the first character. For example,

  ```
  []ab]
  ```

  will match 'a' or 'b' or ']'. But

  ```
  [ab]]
  ```

  matches 'a' or 'b' followed by ']'.

None of the otherwise special search characters, including '\' and '[' has special meaning within a sub-string.

- If you surround a sub-expression in the search string by parentheses, '(' and ')', you can refer in the replace string to whatever this sub-expression matches. In general, an expression in the replace string of the form '\n', where *n* is a number from 1 to 9, means "replace this expression with whatever the *n*th expression in brackets in the search string has matched". For example, if the search string is

  ```
  (.*)read
  ```

  and the replace string is

  ```
  \1ox
  ```

  then if the search string matches 'bread', the found text will be replaced by 'box'. This is because the sub-expression '(.*)' matched the letter 'b'; the expression '\1' in the replace string means "replace this expression with whatever is matched by the *first* expression in parentheses in the search string". Therefore 'b' is substituted for '\1' and the replace string becomes 'box'.

  Here is a more complicated example: suppose the search string is

  ```
  (v.*e) (v.*a)
  ```

  and the replace string is

  ```
  \2 \1
  ```

  Now the search string may match the words 'vice versa'. The first sub-expression, '(v.*e)', matches 'vice' and the second sub-expression,

'(v.*a)', matches 'versa'. In the replace string, HoTMetaL replaces '\2' by what the second sub-expression in the search string matched, and replaces '\1' by what the first sub-expression matched. Therefore the replace string becomes 'versa vice'. The net effect of the operation is to replace an occurrence of 'vice versa' with 'versa vice'.

It is possible to nest sub-expressions. In this situation, the sub-expressions are numbered according to the order of occurrence of their left parentheses. For example, if the search string were

```
(a(bc)d)
```

and the replace string

```
\2 \1
```

the effect would be to find 'abcd' and replace it by 'bc abcd'.

The expression '\0' in a replace string refers to the entire string that was matched by the search string. E.g., if the search string were

```
fish
```

and the replace string were

```
gone \0ing
```

then an occurrence of 'fish' would be replaced by 'gone fishing'.

Elements and character entities as search patterns

A search string that begins with an open angle bracket, '<', followed by a valid element name matches an element of that name. If the search succeeds, the insertion point is positioned to the left of the start tag. It does not matter here or in other search options whether tags are visible or not (see Show/Hide Tags in the View menu). The name in the search string can optionally be followed by a closing angle bracket (>).

For example,

```
<META
```

matches the element that has the name META. Element names are not case sensitive in HoTMetaL, so '<meta' and '<META' would match the same elements.

In a replacement operation, if the search string and the replace string are both elements, one or more occurrences of the element in the search string will be changed to the type specified in the replace string. The contents of the element will be unchanged. The tag in the replace string cannot be followed by text; if it is, an error message will be displayed and the find operation will not be performed.

If only the replace string is an element, the text that is found will be removed and replaced. Replacement will not occur unless the document contains a selection.

Searching for text within an element

The search string can contain both an element name and, following it, some text (or a pattern) that must be matched within the element. In this case the element must end with a closing angle bracket. For example,

```
<P>The
```

would match the word 'The' anywhere within the element P. This is similar to the kind of restrictive searching that can be done using the Find In string but it can be used in conjunction with that feature to further restrict the search. In the last example, if the Find In string is set to:

```
<OL
```

the word 'The' would be matched if it appeared in a paragraph in a ordered list but not if it appeared in a paragraph in another context.

Error messages

If you have a badly-formed search or replace string, HoTMetaL will inform you of this with an error message at the bottom of the Find & Replace dialog box. This message will consist of a description of the error and the character position in the string at which the error occurred. Errors that will be reported include: invalid element or character entity names; unmatched parentheses and brackets in search patterns; '?', '*', or '+' not preceded by any character; invalid character ranges.

For example, if you use the search pattern:

```
<QUAGMIRE
```

you will get the error message:

> In Find string: Invalid element name at position 2

because the HTML files do not contain an element called QUAGMIRE. This message also indicates that the error was detected at the second character in the search string.

Other search options

There are five options that can be set in searches. You may want to search forward or backward through the file, match only whole words, match upper- and lower-case exactly, employ wrapping, or perform pattern searching. These options can be used in combination and are turned on or off by clicking in the five check boxes in the Find & Replace dialog box. The defaults for these options can be set with the appropriate configuration variable. In this case, the boxes will be selected, or not, according to the configuration setting. Values set with configuration variables may be overridden during your editing session by clicking in the check boxes in the dialog box.

Whole Words

A search string may be part of a word or it may represent an entire word. Turning on Whole Words means that the search will match a sequence of one or more whole words only. For example, if HoTMetaL were told to look for 'red' with Whole Words turned on, it would not find that string in 'Fred'.

With this option turned on, the pattern 'a.*z' will match 'a tough quiz' but not the first 12 characters (including the spaces) of a 'tough buzzard'. The default for this option can be set with the *find_whole_words* configuration variable.

Case Sensitive

When case sensitivity is turned on, HoTMetaL will look for the search string exactly as you've typed it—matching upper case to upper case and lower to lower. With case sensitivity off, the program will find any variation: a search string of 'alice' would match 'ALICE', 'alice' and 'AliCE'. This option applies to patterns as well as text. The default for this option can be set with the *find_case_sensitive* configuration variable.

Backwards Search

HoTMetaL normally starts its searches at the insertion point (or the end of the selection) and moves *towards the bottom* of the file. Backwards Search indicates that you want the search to move from the insertion point (or the start of the selection) *back to the top* of the file. If wrapping is not enabled (see below), HoTMetaL does not wrap around the beginning or end of the file, so you should always make sure that you begin your search from the appropriate place in the file. The default for this option can be set with the *find_backward* configuration variable.

Wrap

When Wrap is turned on, HoTMetaL will wrap around the top or bottom of the file, depending on whether you are doing a forward or backward search. The default for this option can be set with the *find_wrap* configuration variable.

Find Patterns

This option allows you to turn on or off HoTMetaL's ability to find patterns. If Find Patterns is turned off, any special characters that you type in the find or replace strings will be treated as ordinary characters. The default for this option can be set with the *find_patterns* configuration variable.

Find In

One of HoTMetaL's more powerful search features is its ability to restrict a search to the contents of particular type of element. This means, for example, that you could use Find and Replace... to check that a word is in uppercase letters whenever it appears in a paragraph, but in upper and lowercase when it's part of a title.

The Find In text box is used to specify the element that you want to restrict searching to. The format for this string is identical to that for the search string when searching for an element, except that the element name can't be followed by text.

Find Next

Performs a search for the search string specified in the previous Find & Replace dialog box. Once a search string has been specified, this command has the same effect as clicking the Find button in the Find & Replace dialog box.

The View menu

The View menu contains various commands to format the document window when editing an HoTMetaL document. This formatting affects the HoTMetaL display only: it doesn't affect how browsers will display the document.

Numerical values

The Character... and Separation... commands allow you to enter numerical values. These may be set from a menu or entered directly by the user in a text box. As appropriate, these values may be absolute, relative, or expressed as a percentage of some base value (this is explained below). The following units may be used:

- centimeters (cm)
- inches
- machine units
- millimeters (mm)
- picas
- pixels
- points

A pixel is the same as a point, and a machine unit is 1/16 of a pixel.

You may use any unit wherever you are allowed to enter values. For example, point size may be expressed in points, inches, picas, etc. Descriptions of individual commands will indicate which kinds of values (absolute, relative, percentage) may be used.

Units may be specified by giving the full unit name, the abbreviation (mm or cm), or the first few letters of the unit name, as long as that specification is unambiguous.

Examples

The following are examples of valid settings:

- 1 inches
- 1.45 i
- 2 mm
- 2 milli
- 3 pix
- 6 points
- 2 po
- 2 pica
- 2pica

Relative and percentage settings

Relative settings specify an amount to be added to or subtracted from a base setting: they have the same format as absolute settings, but are prefaced by a '+' or '-' sign. As well, you may set a value to be Adopt Current, which means that the value is to be inherited from the surrounding element. Some valid settings are:

- -2.67 inches
- +3 picas
- Adopt Current
- ac (same as Adopt Current)
- +0 (same as Adopt Current)

Values may also be expressed as a percentage by affixing a percent (%) sign, the word 'percent', or a suitable abbreviation, e.g.:

- 100% (same as the default value)
- 150 percent (1.5 times the default value)
- 243 per (2.43 times the default value)

Show/Hide Tags

If you select Show Tags, the start- and end-tags in your document will appear on the screen as small tag icons, and character entities will be represented as rectangular icons containing the character name; if you select Hide Tags, the tag icons will be hidden, and the character entities will appear in text form. This command is toggled: if you choose Show Tags, the menu item changes to Hide Tags, and vice versa. The tags that point to the right are *start-tags*, indicating the beginning of an element, while those that point to the left are *end-tags*, indicating the end of an element. When you create a new HoTMetaL document, tags will be visible by default.

Show/Hide Link and Context View

This feature allows you to display a window showing the sequence of nested elements that terminates at the current insertion point or selection.

This window will not show the structure of the entire document (see Show Structure View below) only the hierarchical context containing the current position. That is, it displays the sequence of open tags at the current position. Any attributes that have been set for elements in the sequence will be displayed.

When you invoke this command the name toggles to Hide Link and Context View; this command should be used to dismiss the context view.

You can bring up a different context view for each open HoTMetaL document.

Show/Hide Structure View

This command brings up a window that shows the structure of the entire document.

You can bring up a different structure view for each open HoTMetaL document. Once a structure view is displayed, it remains visible, even if you change documents, until it is dismissed.

The structure view shows the hierarchy of the document in a plain, unformatted manner. Each line in the structure view represents one element in the document. Each line shows a start tag, an end tag and possibly some text between them. The indention of the line indicates the level at which the element is nested. The text cannot be edited and typing is not recognized, but you can use all the editing commands from the Edit and Markup menus.

To select an element in the structure view, you can either click to the left of the start tag icon and drag the cursor across the icon, or click *twice* to the left of the start tag icon. Multiple elements can be selected by clicking the mouse to the left of a start tag and then dragging down to select successive elements.

When you invoke this command the name toggles to Hide Structure View; this command should be used to dismiss the structure view.

Editing in the structure view

Since text entry is not permitted in the structure view, almost all editing is done in the formatted view. The structure view is useful when rearranging whole elements or when creating an outline for your document. It can also be very useful when you are changing styles, for example, with the Character... command.

You may insert, remove, etc., elements while in the structure view. Any of the valid commands that you select will apply to the contents of the structure view.

Any editing you do in the structure view will appear in the formatted view provided it is displaying the same portion of the document.

Expanding the view of elements

Clicking on an element icon allows you to open and close elements, that is, show and hide their contents. When an element is open, all the elements it contains (but not their sub-elements) are shown as indented lines following that element's line. (To see the contents of sub-elements, open these in turn.) When an element is closed, access to its contents is not possible because the contents are not displayed in the window. Since you can insert elements in the structure view, you can use this feature to build an outline (structure) for your document before you begin to type in the text.

Show Image

This command allows you to display a file (usually a bitmapped graphic file) referred to by a URL in the current element (which must be one of IMG and LINK). To view a file, invoke this command when the insertion point or selection is inside the desired element.

Show Image uses the following mechanism to determine how to process the file—in this explanation, assume that the URL is:

```
file:///c|/rodney/orwell/george.gif
```

1. HoTMetaL reads the filename (the last part of the URL). In this case it's "george.gif"
2. HoTMetaL reads the file extension (the part after the dot). In this example the extension is "gif". This extension should indicate the format of the file: we would expect this file to be in GIF format.
3. HoTMetaL scans the defined configuration variables, looking for one called *view_extension*, where *extension* is the file extension read in the previous step. In this case, it would look for a variable called *view_gif*. The value of this variable (if it's defined) should be a command that processes the file, for example:

   ```
   view_gif=c:\windows\pbrush.exe $FILE
   ```

   If this variable is not found, Show Image terminates with an error message.
4. If the variable is found, the full filename specified in the URL (including the path, if specified) is substituted for the string '$FILE', and the resulting command line will be executed on your system. (If '$FILE' is not present in the command, the full filename is simply appended to the

command.) In this example, the resulting command line would be:

```
c:\windows\pbrush.exe c:\rodney\orwell\george.gif
```

(This invokes the Windows *Paintbrush* program.)

If you intend to use Show Image, you should ensure that the necessary configuration variables (*view_gif*, *view_tif*, *view_jpg*, etc.) are defined before you invoke HoTMetaL, and any programs that these variables refer to are available on your system.

Since Show Image uses only the filename part of a URL, the URL can specify a file on any server, provided that a file of the same name exists on your local system.

Show/Hide Inline Images

If you invoke Show Inline Images, all GIF files referred to by URLs of IMG elements in the current document will be displayed inline (i.e., in the HoTMetaL document window). The menu item will toggle to Hide Inline Images. Invoking Hide Inline Images will cause all such images to be hidden.

If the *show_inline_images* configuration variable is set to TRUE, GIF images will be displayed when a file is opened, and this menu item will be toggled to Hide Inline Images. Otherwise (if *show_inline_images* is FALSE, or not set at all), images *will not* be displayed when a file is opened, and the menu item will be toggled to Show Inline Images.

Show/Hide URLs

By default, HoTMetaL displays the URLs associated with relevant elements, in the prefix of the element's start-tag. If you want to hide the URLs, choose Hide URLs: the URLs will disappear from the display, and the menu item will toggle to Show URLs. Clicking on Show URLs will cause the URLs to be displayed again.

Character...

This item allows you to change character-related formatting properties such as font, font size, and justification. Remember that this formatting will affect the document in HoTMetaL only; it doesn't affect how the document is displayed by browsers.

Setting an element's formatting

You can change the format assigned to an element in the document at any time. When you invoke the Character... command, HoTMetaL gives you a dialog box allowing you to set formatting parameters for the current element (the one the insertion point or selection is in). When you change these parameters, they will change for all occurrences of the current element type.

The name of the element currently being formatted appears in the lower left corner of the dialog box. If you move the cursor to a different element while the Characters dialog box is on the screen, the dialog box changes to reflect the formatting of the new element.

Setting the default formatting

If the insertion point is not inside any element, (e.g., it's to the left of the HTML start-tag) then choosing Characters... gives you a dialog box to set the default format. The "element name" in the lower left corner of the dialog will read ".DEFAULT". Default formatting will ripple through the entire structure, provided that the elements in the document's hierarchy have their formatting parameters set to "Adopt Current".

Font Family

The drop-down list box for Font Family shows the current choice. You can specify the font family by name or adopt the font of a surrounding element (adopt current). To select a new font family, position the mouse on the Font Family arrow. Click and drag down the list of the font family names available on your system. Release the mouse button over your choice.

The default font family can be specified with the *default_font_family* configuration variable. A font family specified in this manner will be the default for all open documents. The built-in default font family is Helvetica.

Adopt Current

Sometimes you may want to change the font family you're using on the screen for many related elements. Changing the Font Family for every element would be time-consuming and tedious. You can allow changes to be inherited by taking advantage of HoTMetaL's ability to pass format settings from one element to the next.

For example, if the insertion point is surrounded by the following tags:

HTML BODY P

and you set the Font Family to Adopt Current in the BODY and P elements, then whatever font family you establish in the HTML element will ripple through the entire structure.

Font Size

Selecting the font size is similar to selecting the font family. You can open the Font Size drop-down list box and choose any size (including Adopt Current) shown. You may also enter a size in the text box to the left of the arrow. This size may be an absolute or relative value. If the size you chose is unavailable, HoTMetaL will choose the next smallest font size. Relative values (+2 points, -3 points, etc.) are added to the point size of the surrounding element.

The default font size can be specified with the *default_font_size* configuration variable. A font size specified in this manner will be the default for all open documents. The built-in default font size is 12 points.

Font Style

The check boxes under the Font Style heading allow you to add style variations to the font family. The check boxes can be set individually or in any combination. If you want the style to be the same as the style for the surrounding element, click on the Adopt Current check box only.

Bold, Italic

As an example, consider a document with three emphasis elements: B, I, and EM. So that the contents of these elements will stand out, you could set the font family and font size to Adopt Current for each of these elements, make them all Inline (see below) and then set B to Bold, I to Italic, EM to Bold and Italic. These emphasis elements would then cause their contents to be the same font and size as the surrounding text but each would have a different style.

Superscript, Subscript

Superscript and Subscript raise and lower, respectively, the baseline of the font. Superscripted text is raised so that its baseline is one-third of its ascent above the baseline of the containing element. (The *ascent* is roughly the distance from the baseline to the top of an uppercase letter.) Similarly, the baseline of subscripted text is lowered to be one-third of the ascent below the baseline of the surrounding element. To make effective use of these style options, either choose a font size that is smaller than the size of the containing element or ensure that the line spacing of the containing element is large enough that the tops of the superscripted characters and bottoms of the subscripted characters are not cut off.

Subscripts and superscripts are normally not supported by browsers.

Toggle

When Toggle is selected as part of the font style, the other style settings are *turned off* in the current element if they are *turned on* in the containing element. For example, an element whose font style is set to Bold and Toggle will appear as bold text within plain (Roman) surrounding text and as plain within bold surrounding text.

Adopt Current

Adopt Current means that the font style options of the containing element are adopted in addition to those explicitly set for this element type. For example, if Adopt Current is the only option selected, then the font style for an element of this type will be identical to the font style of its containing element. Another example would be an element whose font style is set to both Adopt Current and Italic. Then the element's text will appear with a font style of italic within an element containing plain text and as bold-italic in bold text.

Underline

Underline allows you to underline the text of the element.

Line Height

The Line Height specifies the vertical height from the bottom of one line to the bottom of the next line in the same element. This value should normally be set slightly larger than the font size so that the lines won't appear clipped.

You may select single, double, or triple spacing from the drop-down list box labeled Line Height. Double spacing gives twice the line height of single spacing; triple spacing gives three times the line height of single spacing. A value may also be entered directly in the text box to the right of the arrow. This value may be absolute, relative, or a percentage.

A percentage line height is interpreted as a percentage of single spacing. I.e., 100 percent is the same as single spacing, 150 percent is 1.5 times as high as single spacing, and so on.

If you give a relative value then the line height is equal to the single spaced line height, plus or minus the amount specified.

An absolute line height should be at least as large as the point size: otherwise, the lines will overlap. A value of about 1.2 times the point size would be normal.

If you select Adopt Current for the line height, then the element you are formatting will assume the absolute line height of its surrounding element. So if the surrounding element has a font size of 12 points and single line spacing, and you choose, say, a font size of 24 points for the current element, it will have the same line height as if it had a font size of 12 points, because that is the line height of the surrounding element. Line height defaults to Adopt Current.

Justification

HoTMetaL offers four styles of text justification plus the capability to inherit the justification style from the surrounding element. You can choose the one you want by clicking on the appropriate icon. The styles are mutually exclusive: you can choose just one.

The choices are:

- *Left* (flush left, ragged right): All the spaces between words are the same width, and the text is aligned on the left, leaving the right-hand edge uneven.
- *Right* (flush right, ragged left): All the spaces between words are the same width and the text is aligned on the right.
- *Centered*: all the spaces between words are the same width and the lines are centered on the midpoint of the line length.
- *Both* (flush right and left—often called *justified*): Spaces between words are adjusted to force both left- and right-hand edges of the text to be even.
- *Adopt Current*: the justification style is inherited from the surrounding element.

Fill

This group of choices allows you to specify whether HoTMetaL should treat carriage return characters as spaces or as line ends. When Fill is selected, the lines are broken at the last space before the right indent. If you type Return a return character is inserted but is treated as though it were a space. When No Fill is selected, HoTMetaL will not fill the lines—when you type Return , a new line will be started.

For example, poetry and centered headings would normally be in No Fill mode, while the contents of a paragraph would be in Fill mode.

Browsers typically display elements (with the exception of PRE) in fill mode.

Saving filled elements

When you save a file, you may choose to have line breaks inserted after a certain number of characters in all elements that are in fill mode. See the description of the Save command in the chapter on the File menu for more details.

Format Types

For the purposes of screen formatting, the elements are divided into two broad categories: Block and Inline. You can set the category for each element by clicking on the appropriate option button in the Format Type group.

Block

Block elements:

- Always start on a new line if there are any end-tags or text on the current line.
- Cause any element or text that follows them to begin on a new line.
- Can have all the formatting options set: justification, font family, size and style, vertical and horizontal spacing, and tab settings.

Inline

Inline elements:

- Cause no line break either before or after the element. This format might be used for an emphasized word or phrase or a short quotation.
- Can have only certain formatting options set: font family, size, and style. The other options are inherited from the surrounding element.

Separation...

This command lets you set the amount of vertical space that separates an element from neighboring elements. Remember that the separation values you set with this command will affect only how the document looks in HoTMetaL; they don't affect how the document is displayed by browsers.

To edit these values, move the insertion point inside an instance of an element you want to format, and click on the Separation... menu item, or click inside the desired element when the Separation dialog is already on the screen.

Top and bottom space

Top and bottom space control the vertical separation between elements. They are available only for block elements.

Top Space determines the minimum amount of vertical white space that must precede the element. If the element before this one has a Bottom Space value, the actual separation will be the greater of the current element's top space and the preceding element's bottom space.

The top space can be specified as an absolute amount or as a percentage of one line height at the current line height. For example, if you want 1 1/2 lines of white space between paragraphs, set the top space to 150%. The top and bottom space separations are added to the normal line spacing, so if two successive elements have a bottom separation of zero and top separation of zero, respectively, the baseline of the last line of the first element and the first line of the following element will be one line height apart.

Bottom Space determines the minimum amount of vertical white space that must follow the block. The actual separation is the greater of the current element's bottom space and the next element's top space. The value is specified in the same way as for Top Space, above.

Tabbed elements

An element that has Tabbed turned on is formatted as if it started with a tab. This command does not actually insert a tab into the text. Only inline elements may be designated as tabbed. This option is useful for displaying simple tabular material.

The Markup menu

The Markup menu contains commands to insert and edit HTML markup.

Interpret Document

Sometimes you will attempt to open a document that contains structural errors and HoTMetaL will be unable to correct them. In this situation HoTMetaL will give you the choice of passing the file through a filter or opening it as a text document so that you can correct the errors manually. In the latter case, when you have corrected the errors in a text document, and while the text document is still the current open document, you can use Interpret Document to convert it into an open document in HoTMetaL's internal format (which causes tag icons to be displayed and allows graphical editing). Alternatively, you can save the document and then open it using the Open... command in the File menu.

If Interpret Document discovers an error in the document, the error will be reported and the insertion point will move to its location. In this situation, the document will not be converted to HoTMetaL format.

Interpret Document is very similar to Open...: the difference is that Interpret Document does the equivalent of an open on the *current open text document*, rather than on a file that was chosen from a file selection dialog box.

This command will be enabled only of the current document is a text document. Normally HoTMetaL does not allow you to open a text document for editing as text: a file can be opened in this form only if it contains structural errors that HoTMetaL cannot resolve.

Insert Element...

This command creates a new element by inserting a new pair of start- and end-tag icons.

When you invoke this command, you are given the Insert Element dialog box, which contains a list of names of the elements permitted by the rules file at the location of the insertion point (or selection) in the document.

The list of elements

The left column of the list box contains an alphabetical list of element names; the right column contains a phrase describing the corresponding element. The name of the first element (if there is one) *required* by the HTML rules file at this point in the document will be followed by the '<' character.

Inserting an element

To select an element, click once on the line containing the element name, or type the first character of the element name repeatedly until the correct element is selected. Then click once on the [Insert Element] button to insert it. Alternatively, you can double-click on the line that contains the element name in the scroll box.

If the document contains a selection rather than an insertion point, the selection will be overwritten by the inserted element.

Restrictions

Insert Element... will be disabled and the menu item grayed-out if there are no elements that can be inserted at the insertion point or current selection without creating an incorrectly marked up document. Often the command will become enabled if rules checking is turned off (see Turn Rules Checking On/Off), but there are some elements in which you will never be allowed to insert an element: BASE, BR, HR, IMG, INPUT, ISINDEX, LINK, META, and NEXTID.

Required elements

The Insert Element dialog box includes a check box labeled Include Required Elements. When this box is selected, inserting an element causes the first required subelement (if there is one) to be inserted as well. This option may also be set with the *include_required_elements* configuration variable. This process is recursive, so that if the required subelement itself has a required subelement, this element will also be inserted, and so forth. Note that this does not mean that all required subelements of the current element will be inserted, only the first required subelement (and its own first required subelement, and so forth).

Surround...

This command lets you surround the current selection with a new element.

This command will be available only if the document contains a selection. When you invoke Surround..., the Surround dialog box appears: it contains a list of elements that can surround the selection and still leave the document correctly marked up. Choose an element in the same manner as when inserting an element.

The command will be disabled and the menu item grayed-out if there are no elements that can surround the selection without creating an incorrectly marked up document. Often the command will become enabled if rules checking is turned off (see Turn Rules Checking On/Off), but there are some circumstances in which you will never be allowed to surround a selection. In particular, you cannot surround *any* selection with one of the following elements: BASE, BR, HR, IMG, INPUT, ISINDEX, LINK, META, and NEXTID.

Change...

Changes the type of the current element. (The *current element* is the innermost element containing the insertion point or selection. If the selection consists of an entire element, including its start- and end-tags, then the current element is the one *containing the selected element*, not the selection itself.)

When you invoke this command, HoTMetaL presents you with a dialog box containing a list of elements that can replace the current element and still leave the document correctly marked up. You may choose the appropriate element from the list in the ways described above for the Insert Element... and Surround... commands.

The command will be disabled and the menu item grayed-out if there are no elements that can replace the current element without creating an incorrectly marked up document. Often the command will become enabled if rules checking is turned off (see Turn Rules Checking On/Off in the Special menu), but there are some circumstances in which you will not be allowed to change the current element: if the current element has *any* content (elements, text, or character entities) you cannot change it to any of: BASE, BR, HR, IMG, INPUT, ISINDEX, LINK, META, and NEXTID.

Edit SGML Attributes...

This command lets you view and edit the attributes of the current element.

To edit the attributes of an element, set the insertion point inside that element (i.e., make it the current element). Then invoke the Edit SGML Attributes... command. You will be presented with a dialog box that can be used to edit the attribute list.

The dialog box will change depending on the type and number of attributes that have been defined for this element in the rules file. In every case, however, the dialog box will consist of several lines, one per attribute. Each line will consist of:

- The *name* of the attribute.
- A drop-down list box or a text box, depending on the type of attribute. If the attribute value is defined as being one of a list of values, then the line in the dialog box line for that attribute will contain a drop-down list box giving the possible values. Otherwise, you will see a text box.

When you have entered the desired attribute values, click on the Apply button.

Insert Character Entity...

This command lets you insert a character entity into the document. Character entities represent special characters that you may not be able to enter into a document directly from your keyboard.

When you invoke this command you get a dialog box containing push buttons for the "special" characters in the ISO 8859-1 character set. To insert a character, just click on the button with the mouse.

If the character you want to insert is not found in this dialog box, you should click on the Other... button. When you do so you will get a dialog box that lets you choose character entities from a number of *entity sets*. The entity set is chosen from the drop-down list box labeled Entity Set. The character entities in the selected entity set are displayed in the list labeled Defined Entities. This list has has three columns: the first column gives the entity names; the second column will give some information about the kind of entity that has been defined (normally this will say SDATA, indicating that the character entity is a "specific character data entity" in SGML); the third column in the list gives the content of the character entity.

To choose an entity set, click on the Entity Set drop-down list box and make a selection from the list that appears. There are only two choices that are really useful:

- To choose the ISO 8859-1 character set, click on the line that reads "8879:1987//Entities Added Latin 1//EN".
- Local & Active: this set contains all the character entities that have been used in the current document so far. The effect of this is that character entities that are likely to be frequently used are conveniently grouped

together. If no entities have been used to this point, this list will be empty.

Inserting a character entity

You can insert a character entity by double-clicking on the entity name, or clicking once on the name and once on the Insert button.

If the document contains a selection (as opposed to an insertion point) before the character is inserted, the selection will be replaced by the entity reference.

You will never be allowed to insert a character entity into the elements BASE, BR, HR, IMG, INPUT, ISINDEX, LINK, META, and NEXTID.

Displaying character entities

If the Show Tags command is invoked (see the View menu) a character entity appears on the screen as an icon consisting of a box containing the character name. If Hide Tags is invoked, the character itself is displayed. (There are a few characters that HoTMetaL cannot display—in these cases the character name, surrounded by square brackets, will be displayed instead.) By default, character entities (and tags) are displayed as icons.

You cannot put an insertion point in the icon or the expanded text view of the character entity. You can only select the entire character. That is done by clicking the mouse to one side of it and then dragging across it.

Turn Rules Checking On/Off

This command toggles the state of rules checking in HoTMetaL.

When rules checking is on, HoTMetaL uses the rules file to ensure that the document being edited will be correctly marked up. While this checking is not complete, it will nevertheless catch and prevent most markup errors. Complete checking of the markup is done by the validation process that occurs when you open or save a file.

HoTMetaL prevents markup errors in a number of ways.

- The commands that could cause errors are disabled. For example, the Surround command in the Markup menu will be grayed-out if the document would not be correctly tagged after the selected content was surrounded by any element.
- A restricted list of elements is presented. For example, the Insert Element... command will only present a list of those elements that will leave the document correctly tagged after the insertion.
- An opportunity is given to cancel a command before any damage is done. For example, if a Paste operation would leave the document incorrectly tagged, HoTMetaL will present a warning giving the choice of canceling the paste or completing the command after first turning rules checking off.

Rules checking is normally desirable, since it greatly reduces the chance of making markup errors. However, there are occasions when rules checking can get in the way of the job at hand. Most commonly this happens when the operation that you are performing involves two or more steps, and one

of those intermediate steps will leave the document temporarily incorrectly marked up.

When the rules are not being checked, the commands that were previously disabled will usually become enabled; for the exceptions, see the documentation on the individual commands. This means that you will be able to create an incorrectly tagged document, and therefore you should leave the rules off only as long as you need to.

When you choose Turn Rules Checking On from the menu, the menu item changes to Turn Rules Checking Off to indicate that this state can be toggled. When you turn rules checking back on, HoTMetaL will quickly scan your document to make sure it is correctly tagged. If it isn't, HoTMetaL will present a warning describing the problem and the insertion point moves to the location of the error. Rules checking will remain off. Select Turn Rules Checking On again after the problem is corrected.

The rules checking state (On or Off) is displayed in the lower right corner of the HoTMetaL window.

The Help menu

About HoTMetaL...	Displays the HoTMetaL copyright notice.
SoftQuad Home Page	This command will invoke a browser, displaying the SoftQuad home page.

The Window menu

This menu contains commands for moving between open files and for arranging document windows on the screen.

Next

This command lets you cycle through the open files. HoTMetaL orders the open files according to the time they were opened. When you invoke Next, the file that was opened most recently after the current file, and has not yet been closed, becomes the current file. If the current file is the most recently opened file, then the least recently opened file becomes the current file.

You may also make a file active by clicking on its document window, or by selecting it from the list of open files that appears at the end of the Window menu (see below).

Previous

This command is similar to Next, but it makes the file that was opened most recently before the current file, and has not yet been closed, the current file.

Cascade

This command causes all the document windows (including those for context and structured views) to be cascaded, in the order in which the files were most recently active. That is, the active file is displayed at the front, the file that was next most recently active is displayed behind it, slightly above and to the left, and so forth.

Tile

This command causes all the document windows (including those for context and structured views) to be tiled. That is, the windows will be arranged in a column of non-overlapping, equal-sized windows down the frame. The active document will be on top. If there are more than three open documents, the windows will be arranged in several columns, with the active document at the top left.

Tile Vertically

This command is similar to Tile, but causes all the document windows to be arranged in a row of non-overlapping, equal-sized windows across the frame. The active document will be on the left. If there are more than three open documents, the windows will be arranged in columns, with the active document at the top left.

Arrange Icons

If some document windows have been iconified, this command will cause the icons to be arranged in a row at the bottom of the frame.

Filenames in the Windows menu

Every open document (including context and structure views) will cause a menu item consisting of the document name to be added to the Windows menu. Selecting the document name will cause that document to become active. Each document name is preceded by a number. This number is the *mnemonic* for the menu item, that is, typing the number while the Windows menu is open is the same as selecting the menu item.

The configuration mechanism

HoTMetaL's configuration mechanism lets you modify HoTMetaL's behavior to suit your needs and those of your site.

There are two types of information in the configuration files. There are variables that control HoTMetaL's behavior, and there are variables that give the location of files or directories.

Configuring HoTMetaL

There are many aspects of HoTMetaL's behavior that you can configure to your personal needs or those of your site. For example, you can control default options in the Find and Replace dialog box, set options for the Save command, and specify the locations of various auxiliary files. You can run HoTMetaL without any problems using the default configuration, but at some point you may prefer to customize. This will be particularly true if several people will be using HoTMetaL on the same PC.

Configuration files

The default configuration files read by HoTMetaL are the file *sqhm.ini* located in the directory where HoTMetaL is installed and the file *sqhm.ini* in the Microsoft Windows directory (usually *c:\windows*). These files contain configuration parameters called *configuration variables*. Variables set in the file in the Windows directory take precedence.

You can specify that different files are read by the configuration mechanism, according to the following procedure:

- If the HoTMetaL command line contains the option *-sqconfig* followed by a list of one or more files, these files will be used as the configuration files.

  To specify the files using the command line, you have to modify the HoTMetaL command line using the Properties... command in the Windows File menu.

  - Click once on the HoTMetaL icon.
  - Invoke the Properties... command.
  - A dialog box will appear. In the Command line text box in this dialog, add the option *-sqconfig* followed by the filenames. For example:

```
c:\sqhm\sqhm.exe –sqconfig c:\frances\sqhm.ini
```

You can specify more than one file by separating the files with semicolons:

```
c:\sqhm\sqhm.exe
–sqconfig c:\frances\sqhm.ini;${SQDIR}\sqhm.ini
```

(This example should be read as one line.) '${SQDIR}' means "replace this by the value of the SQDIR environment variable" (this variable, if it has a value, names the HoTMetaL directory). This setting causes two files to be used: the file *c:\frances\sqhm.ini*, as before, and the file *sqhm.ini* located in the HoTMetaL directory.

> **Note** the SQDIR (HoTMetaL directory) may also be set on the command line. If so, the expression '${SQDIR}' will represent the value from the command line. See the section 'Setting the HoTMetaL directory' in the 'A guide for the perplexed' chapter for full details.

- If there is no *-sqconfig* option in the HoTMetaL command line, but the DOS environment variable SQCONFIG names one or more files, then these will be used as configuration files. Here are examples of specifying configuration files using this method.

  ```
  set SQCONFIG=c:\frances\sqhm.ini

  set SQCONFIG=c:\frances\sqhm.ini;${SQDIR}\sqhm.ini
  ```

  The format for the value of the SQCONFIG variable is the same as for the value that can come after the *-sqconfig* command line option, as described above.

  Environment variables can be set at the DOS prompt, or if you want them to be set every time the PC is booted, you can put the same settings in the *autoexec.bat* file.

- If no configuration files are specified with the *-sqconfig* option or the SQCONFIG environment variable, HoTMetaL will try to read the default configuration files: *sqhm.ini* in the directory where HoTMetaL is installed and in the Windows directory (usually *c:\windows*).

If a number of configuration files are specified using the environment variable or the command line, HoTMetaL reads the list of files from right to left, that is, in the reverse of the order in which they are listed. If a particular variable (parameter) has a setting in more than one file, the value in the file that is read last will take precedence. If a variable is defined more than once in the same file, the value which appears *last* in that file will take precedence over values that appear earlier in the file.

If a variable is not set in any configuration file, but is set in the environment, then the setting from the environment is used. If there is no setting in the configuration files or in the environment, then the built-in default value (if there is one) is used. If there is no default value, the variable is undefined.

In summary, the value of a configuration variable is taken from the following sources, in the order given below:

1. The configuration files.
2. The environment.
3. The built-in default.

Configuration variables read on start-up!

Configuration files (and configuration variables in the environment) are read by HoTMetaL on start-up, so the changes you make will take effect the next time you run HoTMetaL—they will have no effect on a currently-running HoTMetaL. If you need them to take effect immediately, you will have to exit HoTMetaL and restart it.

A suggestion: base and personal configuration files

The following arrangement is one suggestion about how you can use configuration files. It involves using two files: a *base* file for variables that don't change very often, and a *personal* file for variables that change more frequently, are experimental, or are employed by a single user.

Base file

The base file should be used as a system configuration file. It should contain the settings that you want to be used as defaults. A parameter should be changed in this file only if you decide that its default value should change for everyone using HoTMetaL on a specific PC. The file *sqhm.ini* in the HoTMetaL directory should be used for this purpose.

Personal file

The personal configuration file should be used if you need to override some of the settings in the base file without modifying the base file. For example, you may want to make temporary changes to some of the parameters, or to specify parameter values that are used only by you (or another individual user), rather than by everyone who uses HoTMetaL on the computer. If your PC has only a single user, you may choose not to use the personal file at all. Or, you could use it only for making temporary changes to the configuration. The file *sqhm.ini* in the Windows directory should be used for this purpose.

Setting parameters in the configuration files

You do not need to change any configuration variables unless you wish to customize the HoTMetaL configuration.

A variable is just a name that is assigned some value. You can change these variables by simply editing the configuration files, as appropriate, and making the desired changes.

Basic format for setting variables

Variables are assigned values by putting lines of the following form in the configuration files:

```
variable = value
```

For example:

```
undo_limit=50
```

undo_limit is a configuration variable that specifies the number of successive commands that can be undone or reversed with HoTMetaL's Undo command. The default built in to HoTMetaL is 10; to raise this to 50, you would set the variable as in the example.

You may put spaces or tabs on either side of the equal sign for readability. Also, if you prefer, you may substitute a colon (:) for the equal sign:

```
undo_limit:50
```

The effect is the same.

You should *not* have any "white space" (spaces or tabs) at the end of the line.

If you don't want to set a particular variable, then you can do this by omitting or deleting any settings of that variable in the configuration files. Alternatively, you can "comment out" settings in these files by inserting the '**#**' character as the first character on all lines containing such settings: HoTMetaL will ignore such lines.

For example:

```
#undo_limit=50
```

You should *not* try to give variables a "null" value, e.g.,

```
undo_limit=
```

Or:

```
tag_font_name=""
```

You should take care to use legal values for all the configuration variables. Otherwise, HoTMetaL may behave in unexpected ways.

Referencing one variable from another

You can use the value of one configuration variable when assigning the value of another variable. For example:

```
my_name_is=rodney
templates_path= d:\${my_name_is}\tmplts
```

The expression:

```
${my_name_is}
```

is equal to the current value of the configuration variable (*my_name_is*) between the '{' and '}'. So this expression is equal to 'rodney'. When HoTMetaL evaluates *templates_path* in the last example, it substitutes 'rodney'. for '${my_name_is}', so that the value of *templates_path* becomes 'd:\rodney\tmplts'. HoTMetaL performs this substitution when it *uses the variable*, not when it reads the configuration files at start-up.

You can use the same notation to cause HoTMetaL to read a DOS environment variable. For example, you could set an environment variable (at the DOS prompt or in the *autoexec.bat* file):

```
set MY_NAME=rodney
```

A configuration file could have the following setting:

```
templates_path= d:\${MY_NAME}\tmplts
```

When *templates_path* is evaluated, 'rodney' is substituted for '${MY_NAME}', so that the value of *templates_path* becomes 'd:\rodney\tmplts'. As before, this substitution occurs when the variable is used by HoTMetaL, not at start-up.

The '$' symbol is used as a special character in configuration variables, so if you want to put a '$' in the value of an variable, you have to represent it with '$$'.

Appending and prepending to a variable

If a configuration variable has already been assigned a value, you may wish to *append* or *prepend* some characters to it. For example:

```
templates_path=c:\sqhm\tmplts;
templates_path += d:\jennifer\tmplts;

styles_path=c:\sqhm\styles;
styles_path =+ d:\jennifer\styles;
```

In the first example, the variable *templates_path* is first given the value 'c:\sqhm\tmplts;'. The '+=' in the expression

```
templates_path += d:\jennifer\tmplts;
```

causes 'd:\jennifer\tmplts;' to be *prepended* to the current value of *templates_path*. The value of *templates_path* becomes 'd:\jennifer\templates;c:\sqhm\templates;'.

In the second example, the variable *styles_path* is first given the value 'c:\sqhm\styles;'. The '=+' in the expression

```
styles_path =+ d:\jennifer\styles;
```

causes 'd:\jennifer\styles;' to be *appended* to the current value of *styles_path*. The value of *styles_path* becomes 'c:\sqhm\styles;d:\jennifer\styles;'.

(In these examples, the semi-colon, ';', between the file names, is *not* inserted automatically by HoTMetaL.)

If a variable does not already have a value, then assigning it a value using '=+' or '+=' has the same effect as just assigning a value using '='. For example, if *templates_path* does not currently have a value, then

```
templates_path+=c:\tmplts;
```

has the same effect as

```
templates_path=c:\tmplts;
```

Control variables

These variables control various aspects of HoTMetaL's behavior: save options, find and replace options, etc. Many of these variables take values of YES or NO; please note that YES, *true*, and 1 (one) are synonymous here, as are NO, *false*, and 0 (zero).

Save options

The following variables allow you to choose default *save options* for the Save and Save As... commands.

export_doc_type_dec

By default, HoTMetaL will save the *document type declaration* (DOCTYPE) when it saves a file. If this variable is set to NO, then HoTMetaL will *not* include the DOCTYPE with an exported file. If it is set to YES, or if it is not set at all, then the DOCTYPE will be saved with the file.

export_sgml_dec

By default, HoTMetaL will *not* export the SGML declaration when it exports a file. If this variable is set to YES, then the SGML declaration will be exported. If it is set to NO or omitted, then the SGML declaration is not exported.

export_add_line_breaks

By default, HoTMetaL will not impose any limit on the length of a line in an saved file, i.e., it will not add any explicit line breaks. If this variable is set to YES, HoTMetaL will add line breaks after the number of characters specified with the *export_max_line_len* variable (see below). If *export_add_line_breaks* is set to NO or omitted, no line breaks will be added.

export_max_line_len

If *export_add_line_breaks* is set to YES, lines will be broken after a certain number of characters. You can set this number with the variable *export_max_line_len* or in the dialog box, e.g.,

```
export_max_line_len=60
```

If this variable is not set to any value, the default is 72 characters.

export_convert_special_chars

If *export_convert_special_chars* is set to YES, then HoTMetaL will convert any special characters inserted directly in your document to SGML *character references*. (Special characters are those outside the ASCII range 0-127). This will apply only to special characters that were inserted if the file was modified or created by another editing package: special characters that are inserted using HoTMetaL are immediately converted into *character entity* icons. Character references are supported by browsers such as *Mosaic*. If the variable is omitted or set to NO, then special characters are not converted.

export_eol

This variable lets you choose the end-of-line marker that will be generated in your saved file. There are three choices: UNIX, which causes the end-of-line marker to be a line feed, MSDOS, which causes it to be a carriage return followed by a line feed, and MAC, which sets the marker to be a carriage return. The default value for this variable is MSDOS.

Find options

The next group of variables allow you to control the behavior of the commands in HoTMetaL's Find menu. You may override all these settings from the dialog box that accompanies the Find and Replace... command.

find_whole_words

By default, the Whole Words option is turned off in the Find & Replace dialog box. If this variable is set to YES, this option is turned on; if it is set to NO or undefined, Whole Words is turned off in the dialog box.

find_case_sensitive

By default, the Case Sensitive option is turned off in the Find & Replace dialog box. If this variable is set to YES this option is turned on; if it is set to NO, or undefined, Case Sensitive is turned off in the dialog box.

find_backward

By default, the Backwards Search option is turned off in the Find & Replace dialog box. If this variable is set to YES, this option is turned on; if it is set to NO, or undefined, Backwards Search is turned off in the dialog box.

find_wrap

By default, the Wrap option, which causes searches to encompass the entire file, starting at the current position, is turned on in the Find & Replace dialog box. If this variable is set to NO, Wrap will be turned off. If the variable is set to YES or not defined, wrapping will be turned on.

find_patterns

By default, the Find Patterns option is turned off in the Find & Replace dialog box. If this variable is set to YES, Find Patterns is turned on. If it is set to NO, or not defined, the Find Patterns option is turned off.

Markup options

These variables govern aspects of the markup process.

include_required_-elements

include_required_elements controls whether the Include Required Elements option is turned on for the Insert Element... command. If this variable is set to NO, the option will be turned off. If the variable is set to YES or undefined, the option will be on. For more information, see the section on Insert Element... in the Markup menu chapter.

prompt_for_attrs

This variable controls whether the Edit Attributes dialog box will be displayed each time an element with attributes is inserted in the document. If it is set to NO, or undefined, then users will be prompted with this dialog only if the element being inserted has *required* attributes. If the variable is set to YES, then the user will be prompted every time an element with attributes is inserted.

Display variables

The variables in this section pertain to HoTMetaL's screen display: invisible characters, fonts for icons, and colors.

default_font_name

This variable lets you choose the default font family for all documents opened with HoTMetaL. The value of this variable should be a font name, exactly as it appears in the Font Family drop-down list box. For example:

```
default_font_name=Times New Roman
```

The default font family is Helvetica.

default_font_size

This variable lets you choose the default font size for all documents opened with HoTMetaL. The value of this variable should be a positive number. For example:

```
default_font_size=14
```

The default font size is 12 points.

tag_font_name

This variable lets you choose the font used to display the element names in the tag icons in an HoTMetaL document. The list of available fonts is dependent on your system. The best way to find out which fonts you can use is to invoke HoTMetaL's Character... command and click on the arrow next to the drop-down list box labeled Font Family. The menu that appears contains the names of the available fonts. The value assigned to the *tag_font_name* variable should be the font name, not surrounded by quotes, exactly as it appears in the Font Family menu. E.g.,

```
tag_font_name=Avant Garde
```

The default font is Helvetica.

tag_font_size

This variable lets you choose the font size used to display the element names in the tag icons in an HoTMetaL document. The list of available font sizes is dependent on your system. The best way to find out which font sizes you can use is to invoke HoTMetaL's Character... command and click on the arrow next to the drop-down list box labeled Font Size. The menu that appears contains the available font sizes. The value assigned to the *tag_font_size* variable should be the font size in points, not surrounded by quotes. The default is 12 points.

Other options

html_browser

This variable specifies the application signature for an HTML browser that is to be invoked by HoTMetaL's Preview command.

publish_change_from

This variable specifies the text that appears in the Change From text box in the Publish... command's dialog box. The default value is 'file://'.

publish_change_to

This variable specifies the text that appears in the Change To text box in the Publish... command's dialog box. The default value is 'http://'.

show_inline_images

If this variable is set to TRUE, then any GIF images referred to by URLs in IMG elements will be displayed inline (in the HoTMetaL document window) when a file is opened. Otherwise (if the variable is set to FALSE or not set) such images are hidden. You can override the *show_inline_images* setting using the Show/Hide Inline Images command. This command toggles to Hide Inline Images by default if the variable is set to TRUE, and to Show Inline Images otherwise.

undo_limit

This variable sets the maximum number of commands that can be undone with the Undo command. By default, this value is 10. The maximum value is 65535. The minimum value is one; if you set it to a value less than one, it will be set to one anyway.

view_gif

This variable specifies a program that the Show Image command will invoke to display a file whose name ends with the *.gif* file extension (normally this file would be expected to be in GIF format).

You can add variables of your choice of the form *view_extension*, where *extension* is the file extension of the file you want to display: for example, you could have *view_tif*, *view_jpg*, etc., variables.

Location variables

The variables described below give the locations of directories or files that are used by HoTMetaL, and specify default file extensions.

Paths and directories

Paths are lists of directories that HoTMetaL searches to find files it needs to read, or uses to store files. The value of a variable that describes a path consists of a number of directory names, or paths, separated by semi-colons ';'. As well as giving specific directory names, it is possible to specify HoTMetaL's *working directory* by putting a period, '.', in the path.

> **Note** The working directory is set with the 'Properties...' command in the Microsoft Windows 'File' menu.

When giving a directory name (other than the working directory), you need to give the full DOS path. This may be fully or partly represented by a reference to another configuration or environment variable, as explained earlier in this chapter.

The following example shows how to set a path variable, in this case, *export_path*:

```
export_path=${SQDIR}\samples;.;c:\donald\samples
```

This setting is interpreted as follows:

- The expression '${SQDIR}' is replaced by the name of the HoTMetaL directory, so the expression '${SQDIR}\styles' will cause a directory such as *c:\sqhm\styles* to go into the path.
- The current working directory when the program was invoked, signified by ".", is included in the path.
- Lastly, the directory *c:\donald\samples* is specified explicitly.

The default value for all path variables, with the exception of *templates_path*, is ".".

export_path

This variable names the default directory for saving files. When you invoke the Export... command, the default directory that appears in the Directories list box is the first directory that is listed on the *export_path* variable.

import_path

This variable gives a default directory for opening files. In the file selection dialog box that appears when you invoke the Open... command, the Directories list box displays, by default, the directory that is listed first with the *import_path* variable.

styles_path

This variable describes the styles path, a list of directories where styles files are located. These are files used by HoTMetaL to describe the formatting for a document when it is displayed on the screen. HoTMetaL will create a styles file for a rules file the first time that rules file is used. It places the styles file in the first directory listed by the *styles_path* variable. On subsequent uses of the rules file, HoTMetaL will look for the styles file in the directories named by the variable. If the rules file has been changed since the styles file was created, HoTMetaL will ask you if you want to create a new styles file.

The format of this variable is the same as for *export_path*, above.

templates_path

This variable gives the directory for storing files that can be used as document templates with the Open Template... command.

Files

These variables give the names of specific files that HoTMetaL uses.

Except as noted, when giving a file name you need to give the full DOS path. This may be fully or partly represented by a reference to another configuration or environment variable, as explained above.

rgb_txt

This variable names the color map file, a file that associates color names with the red-green-blue values required to tell HoTMetaL how to produce the colors. The value of this variable should be a file name. If an absolute path is prepended to the file name, then that file is used; if the file name has a relative path, or no path, prepended to it, then HoTMetaL looks for a file relative to the SQDIR directory. The default value is ${SQDIR}*rgb.txt*.

File extensions

These configuration variables determine the default file extensions that appear in the file selection dialog boxes for different kinds of files. The file extensions appear in the Filename text box. The file extension consists of a dot followed by a sequence of characters (usually three). When you set a file extension, you must include the dot in the corresponding variable's value. E.g.,

```
dictionary_ext=.dct
```

styles_ext

This variable sets the default extension for binary styles files. When you create or open an HoTMetaL file, HoTMetaL will look in the styles path for a styles file that has the same name as the rules file's compiled-in system identifier, but with the file extension replaced by the styles extension. The default styles extension is *.stl*.

Tracing configuration variables

HoTMetaL allows you to trace exactly how the configuration settings are used. When tracing is turned on, you will be presented with a warning box when HoTMetaL reads the configuration files, or accesses any of the variables. Tracing is controlled by the SQTRACE environment variable. The possible values are ON (which is the same as *true* and 1), OFF (which is the same as *false* and 0) and FULL. The variable must be set before starting up HoTMetaL.

If SQTRACE has the value ON, you are notified whenever any of the following things happen:

- HoTMetaL reads the configuration files.
- the final value (i.e., after all configuration files have been read) of a variable is set. Furthermore, you will be told which of the files the value comes from.
- HoTMetaL looks for the value of a variable but does not find it in any configuration file.
- a duplicated variable is detected. If a variable is set in more than one configuration file, or more than once in the same file, you will be notified, and given the previous and new values and which files these values came from.
- an absolute file name or path name is encountered as the value of a variable.

If SQTRACE has the value OFF, or is not set, tracing will not be invoked.

Appendix 1: Keyboard shortcuts

This appendix discusses the keyboard shortcuts for invoking HoTMetaL commands and performing other windowing operations.

Shortcuts

Many of the shortcuts for invoking commands involve two keys: the Ctrl key and another key specific to the command.

To invoke the Open... command, for example, hold down the Ctrl key, and then press the O key while the other key is held down. This series of keystrokes is denoted Ctrl-O . In the menu, it is denoted '^O'. Other shortcuts consist of a function key, and some commands have two different shortcuts. Note that some older keyboards may not have the F11 and F12 keys.

Mnemonics

Like other Windows applications, HoTMetaL supports the use of *mnemonics* for accessing menus and commands from the keyboard. A mnemonic is a letter that is associated with a menu bar menu, command, and sometimes with a dialog box control. Usually the mnemonic will be the first letter of the menu or command name, but in cases where more than one menu, or more than one command within the same menu, starts with the same letter, the mnemonic will be a subsequent letter in the name. Since the mnemonic is always underlined, you can easily tell what it is.

You can bring down a menu at any time by pressing the Alt key, and then, while Alt is still depressed, pressing the key with the mnemonic letter. If a menu is visible, you can invoke a command from that menu just by pressing the key with the mnemonic letter. Some controls in file selection dialog boxes are associated with mnemonics. When the dialog box is active, press Alt plus the mnemonic to activate the control.

Menu Item	Keyboard Equivalent	Description
File Menu		
New...	Ctrl-N	Create a new HTML file.
Open...	Ctrl-O	Open an existing HTML file or text file.
Save	Ctrl-S	Save the current file.
Preview	Ctrl-M	Launch a browser, displaying the current document.
Quit	Ctrl-Q	Quit.
Edit Menu		
Undo	Ctrl-Z	Undo your last editing change; successive Undos will undo all changes back to the most recent save
Cut	Ctrl-X	Cut selected text or elements
Copy	Ctrl-C	Copy selected text or elements
Paste	Ctrl-V	Paste the contents of the clipboard at current insertion point
Find and Replace	Ctrl-F	Display the Find and Replace dialog
Find Next	F3	Repeat the most recent 'Find' operation.
View Menu		
Show/Hide Tags	Ctrl-W	Show or hide tags.
Character...	Ctrl-B	Select format for current element

Menu Item	Keyboard Equivalent	Description
Markup Menu		
Insert Element...	Ctrl-I	Insert allowed element.
Surround...	Ctrl-U	Surround the selection with another (permitted) element
Change...	Ctrl-L	Change selected element (if permitted)
Edit SGML Attributes...	F6	Edit attributes for current element.
Insert Character Entity...	Ctrl-E	Allow selection from list of character entities.
Turn Rules Checking On/Off	Ctrl-K	Turn automatic rules checking on/off

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X