

Arachnophilia

Definition of Arachnophilia

-- P. Lutus, Ashland, Oregon --

Arachnophilia is the descendant of WebThing, which is the descendant of Apple Writer, but I digress. I learned a lot writing and distributing WebThing, the most important being “Don’t write programs in Visual Basic.” WebThing had a number of shortcomings, all of which are addressed in Arachnophilia.

Arachnophilia’s purpose is to create Web pages. It does this in one of two general ways. ***The easy way*** is to drop a Rich Text Format (RTF) document onto the Arachnophilia program window and watch Arachnophilia turn it into a web page for you. ***The not so easy way*** is to write the HTML code yourself, which, although more work, produces the most professional-looking results.

Arachnophilia will help you create Web pages, no matter what method you choose. And, by just typing in a text file or a spreadsheet, *you can create new Arachnophilia toolbars for tags I haven’t thought of*. You can load hundreds of documents at once (memory permitting) and search through them at once for particular words. You can preview your work on up to six browsers, thus assuring your pages will look good no matter what browser your visitors own.

I hope you like Arachnophilia. I also hope you will read about CareWare, which is how you “pay” for Arachnophilia.

Web Resource Sites

Frequently Asked Questions

This web site contains answers to the most often asked questions about Arachnophilia.

www.arachnoid.com/arachnophilia

Use this link to get the most recent version of Arachnophilia.

www.microsoft.com,

Click these links to get the most recent versions of popular Web browsers.

www.netscape.com

Click this link to report bugs in Arachnophilia (after reading the Frequently Asked Questions site).

lutusp@arachnoid.com

Be sure to visit www.arachnoid.com, the home of [Arachnophilia](#).

Arachnophilia © Copyright 1996-1997 P. Lutus. All Rights Reserved.

Note: Function key F1 provides context-sensitive help throughout Arachnophilia.

Introduction

Arachnophilia is a full-featured HTML editor and workshop. With Arachnophilia, you can:

- Create Web pages and preview them on all available browsers.
- Interactively edit Web page resources and appearance.
- Import full-featured text, tables and outlines into your Web page, with table structure, indentation, font color, size and face preserved.
- Create interactive forms that launch programs to do things not possible in HTML alone.
- Organize a set of Web pages and associated graphics, sound files, and other resources.
- Maintain Web pages by examining their mutual references.

Arachnophilia will help you get started if you are a beginner, and it will help you organize larger projects as you acquire more experience. Arachnophilia will automatically copy resource files to your working directory as you select them, and will alert you if there are resources that are no longer being used.

Arachnophilia supports many of the HTML tags currently in use, and can be customized to include any specialized tags that the user desires, and any new tags that come into use in the future.

These help pages cannot teach you all you need to know about HTML to be a successful Web page writer, but there are many sources of detailed information that you should access. One very effective way to learn about HTML and Web pages in general is to browse the Web. There you will see any number of well-designed Web pages.

Another way to get more information is to search the Web itself for information about HTML. Example: Get online and access the Yahoo search engine at www.yahoo.com. Type HTML as your search string. Another good search engine is www.lycos.com. Also, bookstores and libraries have many excellent references on HTML.

These pages will help you with the particulars of Arachnophilia and its special features, and contain instructions on the use of the most common HTML tags.

The Internet and the Web

Computer networking has had a short and spectacular history. From the first effort to connect two computers together in the 1950s (rumor has it they deadlocked each other with "syntax error in line 12" messages) to the present, the value of sharing computer resources has become obvious. Regional networks became connected to each other, and eventually the Internet came into being.

There was always some way to navigate from place to place on networks. Earlier systems required you to type something cryptic at a keyboard, and to know something about UNIX. As time passed and more people tried to use the Internet for more kinds of things, the old ways were replaced with more intuitive graphic interfaces.

The most popular present method to interact with the Internet is by way of a Web Browser, a program that can read Web Pages that have been posted on the Internet. A modern Web page is a multimedia resource, containing text, images, sounds, and animations. Soon Web pages will contain links to objects that will offer you access to resources such as programs and information in ways not presently possible.

The "Web" in "Web page" refers to The World Wide Web, which is not so much a network as a way to interact with a computer network. The Web is in fact a combination of communication protocols that include a language called HTML, several kinds of programs called Browsers, and the Internet to connect it all together.

"Web pages" are written in HTML using a program like this one, generically known as a "Web Page Editor." After being written, Web pages are usually posted on a computer that is attached to the Internet. Then people who have network access to that computer can "read" the web page using their browser program.

Some Web pages contain graphics, sounds, and other things to increase their information content (translate: fun). But the most important single thing about a Web page is that it can contain links to other pages, resources, and locations. The behavior of these links is referred to as Hypertext, a method by which one can jump from place to place by activating the links.

In the original embodiment of Hypertext, one would move from paragraph to paragraph in a document using links, as in this help page. Now, using HTML, one may move from country to country by clicking a pointer device.

Individuals and institutions are therefore very interested in creating Web pages in the easiest possible way, and that is why Arachnophilia was written -- to make it easy to create your own Web page, add to it, and maintain it.

A good way to start is to browse the Web and see some of the pages that have been written by others, to get ideas about how you would like your page to look. Most browsers allow you to examine the original HTML code to see how a particular effect was created. This is a good way

to learn advanced techniques.

The Basics of Web Pages

In the simplest terms, the World Wide Web is a collection of web pages and other resources located on Internet server computers connected to web page browsers located on individual machines also connected to the Internet. What makes it a Web is that the Web pages communicate the information desired by their authors, and also they connect readers to other resources by way of links. The links in Web pages can refer to another Web page, a program, or some other kind of file, or another way of interacting with the Web such as Telnet.

Web pages have two aspects. One aspect is how the page looks to a web browser, colorful and interactive. The other aspect is the actual HTML code that tells the browser what to do. The fun part of browsing is seeing the result aspect, but the other part, the code, has to come first.

Fortunately, HTML was designed to be as obvious as possible, consistent with its purpose. You will be writing HTML in a very short time from starting out. Writing *great* HTML code will take longer.

The overall process for creating a Web page is to get the resources together -- a web editor (you've already taken care of that) and a browser. If you don't already have a Web browser, I recommend Microsoft Internet Explorer. It is available on the Internet (possibly a catch-22 if you don't have any browser right now) and, like Arachnophilia, it is free (but see the CareWare section for a better explanation of "free").

Once you have these basic tools, you can follow this strategy:

- Make a plan. Perhaps you want to have just one page that says "Hi, world, this is me!", or perhaps you want to create something more ambitious, several pages, lots of graphics. A broad outline of your intentions is a good starting point.
- Select a data directory to store your work as it proceeds. Arachnophilia will prompt you to do this the first time you create a link between two file resources.
- Create the first page (if you are planning more than one). This page is usually the first thing a visitor sees, and is often no more than a table of contents. Usually this page is named "index.htm" to be consistent with how Internet servers locate and activate pages.
- Create the child pages, and link them to the index page. As you work, launch your Web browser (Arachnophilia allows you to do this while you work) to see the result of your efforts.
- Finally, after testing all the pages and jumping between them and to any outside resources you reference, you can upload your page(s) to the Internet. This is in some ways the most specialized part of the process, and you may have to ask your Internet Service Provider for help.

Arachnophilia has many features to simplify this process. Arachnophilia will remind you to set up a working directory, then it will ask whether you want to automatically copy resource files to

this directory as you work. Using this feature, you will automatically assemble all the resources for your Web page in one place, and you will also use what are called "relative links" to them, meaning the entire package will work as expected after being moved onto the Internet server. And, using a cross-reference feature, Arachnophilia will find and list resources files that are no longer being used by your pages, for easy removal.

With Arachnophilia and a browser installed on your system, you can develop web pages without actually being connected to the Internet. You can write, test, debug, and modify your pages before posting them on the Internet. And, perhaps best of all, you can add your favorite tags to Arachnophilia, tags you use a lot, or tags that don't exist as this is being written.

The Basics of HTML

The most important thing to realize about HTML is that it is changing with time, and that different browsers support different features. Therefore, if you want to be a serious Web page developer, you should collect enough browsers to be sure your page looks acceptable on all of them. Arachnophilia supports up to six browsers simultaneously, allowing you to verify changes as you work.

The next most important thing to know about HTML is that it was designed to be as easy and transparent as possible. After a short time, you will be able to look at an HTML script and predict more or less how it will look when viewed with a browser. Naturally, Arachnophilia does not require you to guess -- you can just press a button to move between the HTML code display and the browser display.

The most basic element of HTML is called a "Tag." HTML tags usually, but not always, come in pairs, an opening tag and a closing tag.

Tags are enclosed in the greater than - less than symbols "<" ">."

The *opening tag* of a pair of tags appears this way: <TAG>.

The *closing tag* of a pair appears this way: </TAG>. Notice the slash.

Tags can contain modifiers within the "<" ">" symbols. Example:
The information *inside* the tags tell the browser to do something. Example: <BGSOUND SRC="music.mid"> tells the browser to play some music (if the browser in question is Microsoft Internet Explorer).

The information outside the tags is printed as text. Example: the HTML code "This should be Emphasized more than usual." will be shown by the browser as "This should be **emphasized** more than usual."

In this example, the browser understood the tag to mean "start printing in **boldface**." The browser then took the tag to mean "go back to the previous state." Most HTML tags are nested, which means they add to the formatting of any prior tags, even if the prior tags specify the same thing. This example of HTML:

"This is an important point to make."

Would most likely be rendered as:

"This is **an important point** to make."

because each pair of tags independently affect the text, and when the closing tag appears, the browser simply returns to whatever formatting had existed previously. This is important to understand and can save you from mysterious behavior in your Web pages.

NOTE: These structural issues are important to understand, but Arachnophilia will automatically convert text with many different styles, fonts, and colors for you, either by dragging text from another application using the Arachnophilia File Import methods, or by saving files in Rich Text Format, then simply loading them into Arachnophilia. Arachnophilia will automatically convert formatted text, tables, and outlines (indented structures) into the appropriate HTML code, as much as possible preserving the original appearance of your text (within the constraints imposed by HTML).

The most basic HTML document looks like this:

```
<HTML>

<Head>
<Title></Title>
</Head>

<Body>

|

</Body>

</HTML>
```

These pairs of tags are regarded as the basic structure of an HTML page, and all existing browsers support it. Arachnophilia will automatically create this structure for you when you open an HTML document from the File menu, or use the Structure/Lists toolbar.

The <HTML></HTML> tags simply identify this as an HTML document. The <Head><Title></Title></Head> tags basically allow you to place a title inside your document. This title is normally not displayed by the browser as part of the page, but appears at the top of the browser display on its title bar, and is also accessed by Internet search engines for descriptive information about your page.

Most Web Page development takes place within the <Body> </Body> tag pair. For example, if you typed "This is my page" between the <Body> and </Body> tags, your browser would display that phrase.

There are many structural elements in HTML, such as tables, lists, and provisions for graphic images and other resources, but by far the most important to understand is the Hyperlink or link.

The most common HTML link looks like this:

```
<A href="destination">An interesting place</A>
```

The browser will show this link as [An interesting place](#) and if you click on it (in a browser), the browser will locate and load the thing described by "destination". In creating a link, the text you type between the <A> and tags tells the *reader* what the link is, and the href="destination"

part of the link tells the *browser* where to look for the resource.

Links are the way you navigate around the World Wide Web. A link might contain another of your own pages:

```
<A href="mystory.htm">Click here to read my story</A>
```

A link might refer to another location in the same Web page

```
<A href="#somebookmark">Click here to move to the bookmark</A>
```

Elsewhere in the Web page would be the name to which this bookmark refers, and to which the browser would move:

```
<A name="somebookmark">Thanks for coming around!</A>
```

A link can refer to a file you want to allow access to:

```
<A href="myprogram.exe">Click here to run my program</A>
```

Or:

```
<A href="database.mdb">Click here to download my data file</A>
```

And (actually a very common example) you can refer to something else on the World Wide Web:

```
<A href="http://www.microsoft.com">Click here to visit Microsoft</A>
```

You can also place a graphic image on your page:

```
<IMG SRC="myhouse.gif">
```

You can even use a graphic image as the clickable area for a hypertext link:

```
<A href="http://www.beachstuff.com"><IMG SRC="sandybeach.gif">  
</A>
```

In this example, the visible part of the tag is a picture instead of text. When the user clicks on the picture "sandybeach.gif", the browser jumps to the Web location identified by "http://www.beachstuff.com".

Hypertext links can refer to many kinds of things, including files, Web sites, locations in the same document, other pages you have written, they can even launch applications and send E-mail.

This is just the barest outline of what HTML can do. Once again, there are many resources

available that can provide a full description and use of HTML, including resources on the Web itself.

How to make your own page (the easy way)

You always have the option of creating a page from scratch, by entering the HTML tags on your own. But, even with a lot of predefined tags, this takes a long time, and you have to learn a lot about HTML as you go along.

But there is an easier way -- Arachnophilia will allow you to import a word processing document that has been saved in Rich Text Format (RTF) and will then convert it automatically into a Web page.

Some word processing programs will export their documents in the RTF format, others won't, but you can still export them using one of the methods described here. Here are the ways to get your document from your word processor/spreadsheet/database to Arachnophilia:

- 1? Simply save the document in the Rich Text Format (RTF) document type. Then open it in Arachnophilia (File ... Open File ... RTF File). When you do this, Arachnophilia will ask whether it should convert it into HTML format. Choose "Yes" and you are done.
- 2? Open Windows Explorer and drag an RTF document icon onto the Arachnophilia program window (not onto an open document, but onto the background). In the same way as method (1), Arachnophilia will ask whether it should convert it into HTML format.
- 3? For programs that do not support the RTF file format, open your source program and Arachnophilia at the same time. Open a new, blank RTF document in Arachnophilia (File ... New File ... RTF File). Move to your source program and select the block of text you are interested in. Drag this block over to the open RTF document window in Arachnophilia, and drop it. Then select the menu option Tools ... Convert RTF to HTML.
- 4? Use the Windows clipboard and proceed as in method (3) above.

These methods apply to data sources such as spreadsheets and databases -- just use method 3 or 4 above if the program does not export RTF files.

Here are some suggestions to help Arachnophilia convert your document:

- 1? Always use *real* bulleted lists and *real* numbered lists, available in most word processing programs, instead of manually numbering a list of items. If you use the real versions of these features, Arachnophilia will create the HTML equivalent of these structures, which look great. If you simply manually number a list, it will not look nearly as good when viewed on a browser.
- 2? Avoid "outdenting" paragraphs in your document formatting -- Arachnophilia will interpret this as an outline. Instead, if you create a real outline, even one with multiple levels, Arachnophilia will translate it into HTML for you.

Here are some restrictions for this automatic conversion method:

- 1? Arachnophilia will automatically import tables and outlines, but cannot import any pictures that are included in your original document. These parts of your document have to be imported separately, using the HTML tags that are designed for that purpose.

- 2? Sometimes Arachnophilia will misinterpret a line with tab characters in it as a table row. If this is not what you intended and you want to prevent this behavior, choose Tools ... Options and select "Convert Tabs into Spaces." Remember later that you made this choice, because Arachnophilia won't create tables until the "Convert Tabbed Lines into Tables" option is enabled again.
- 3? In general, avoid the use of tabs, because this character is used in the RTF document format to identify table rows, and Arachnophilia relies on them for this purpose.

Be sure to review the [Arachnophilia File Import methods](#) for more information on these techniques.

How to make your own page (the hard way)

Remember: you can just import many types of documents directly into Arachnophilia. The methods described here are for finer control over the outcome than the HTML translator can provide. See the [Arachnophilia File Import methods](#) for more information.

Now we will create a small Web Page, just to get your feet wet. At this point you should have installed Arachnophilia on your computer, have an icon available to run Arachnophilia, and be in a somewhat adventurous frame of mind.

(If this is hard to follow the first time, you might want to print this help screen so you don't have to keep referring to it on the computer screen.)

Go ahead and run Arachnophilia, and select File ... New HTML Document. A new document will appear on the screen, and a dialog box will appear with some choices. Just to keep this simple, for now just press "OK" to go on.

This document outline will appear on your screen:

```
<HTML>

<Head>
<Title></Title>
</Head>

<Body Background="" BgColor=#FFFFFF Text=#000000 Link=#0000FF VLink=#800080
ALink=#FF0000>

|

</Body>

</HTML>
```

The vertical bar '|' above represents the location of the cursor in the Arachnophilia editor screen. Now type "This is my first Web Page!" and then press the *preview button*, the toolbar button that looks like this:



If you don't see your Web Page displayed, then a word of explanation is in order. When Arachnophilia runs for the first time, it looks around in your system for useful information, like whether Windows knows the whereabouts of a browser. If there is such a browser, Arachnophilia will use it to show your page.

If Windows doesn't know the location of a browser, Arachnophilia will ask you to locate one for it. If you actually don't have a browser on your system, you should stop this exercise, acquire one, and continue. There are several good ones posted on the web (a possible catch-22 situation if you don't already have a browser with which to look for it) and some are free.

If you do have a browser that Windows doesn't know about, simply tell Arachnophilia its location, and our little exercise can continue.

When you are done admiring your handiwork as displayed by your browser, just move the browser out of the way and Arachnophilia will reappear.

Now let's experiment. I can't know what your background is or how much you know about computers, so you may choose to take each of these steps, or skip ahead if you wish.

Drag the mouse across the word "first" to select it, so the display looks like this:

This is my [first](#) Web Page!

(note: the text shown in [color](#) here will look white-on-black on the Arachnophilia editor screen, i.e. selected)

Now press the toolbar button that looks like this:



This button will make the selection **bold**, just like in a normal word processor. The Arachnophilia display will look like this:

This is my **first** Web Page!

Remember: all formatting in the HTML language is by way of tags. You won't see the bold effect until you click the preview button on the toolbar and let the Browser display the result of your changes.

Now let's choose some more interesting colors for the display. Use your mouse to position the cursor as shown below:

```
<Body Background="" BgColor=#FFF|FFF Text=#000000 Link=#0000FF VLink=#800080
ALink=#FF0000>
```

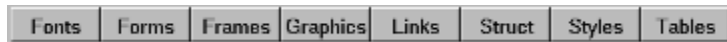
Notice the vertical bar in the middle of "BgColor=#FFF|FFF". This time, don't select anything, just position the cursor as shown. Now press the *right-hand mouse button*. A dialog box will appear asking you to select a color. Choose a color you like for a background. Now position the cursor in the middle of " Text=#000|000" and make a selection in the same way.

Now press the preview button again and look at the colors you have chosen. You can experiment until you are satisfied with your colors. You will notice that, if you have selected any of the text in the document and then click the right mouse button, a menu appears with some of the formatting tags that also appear on the toolbar, but if you just position the cursor in a color definition without selecting anything, the color selection dialog appears.

Now let's add an interesting graphic image for your page. Web Browsers use the background graphic as sort of wallpaper, i.e. they make repeated copies of the graphic to fill up the screen. Position the cursor as shown below:

```
<Body Background="|" BgColor=#FFFFFF Text=#000000 Link=#0000FF VLink=#800080
ALink=#FF0000>
```

Now look at the bottom of the screen. You will see a row of tabs that looks like this:



Each of these tabs launches a separate toolbar that can be placed in any convenient location. For this example, click "Graphics" and then select EditImg from the toolbar that will appear.

Arachnophilia will now ask you whether you want to save your file. This is a very good idea for several reasons. One reason is that you might otherwise lose your work in the event of a system crash or power failure. Another is that, if you locate your Web page in a particular directory, Arachnophilia can automatically move all your resource choices to that directory also. This makes it very easy to move your completed Web page (or pages) to the Internet later on.

After you have responded to the Save-File dialog box, hopefully by saving your file, Arachnophilia will ask you whether you want to automatically copy any files you select to this same directory. This is a good idea also, because (1) all your work will be located in one place, and (2) you are then allowed to use "relative links" to identify your resources, links that will work the same way no matter where you move your Web page.

These two dialog boxes are intended to guide you to a method that will seem desirable and obvious when you have had more experience.

Now navigate to a directory that contains some graphic files and select one. The two most-often used Web page file types are Graphic Image Format with the suffix .GIF, and Joint Photographic Experts Group (wow -- what a title!) files with the suffix .JPG. There are some others, but these are the best choices. If you don't have any graphic images with these formats, perhaps you have a paint program (Corel Photo-Paint is one example) that can convert some of your existing images to one of these two formats.

Having selected a graphic image, you may once again see the result of your work by previewing.

By the way, if you make a change you don't like, you can undo it by pressing this toolbar button:



Now let's do some rudimentary formatting to improve the appearance of our page. Let's make two horizontal bars, one above and one below our typing. Position the cursor as shown below:

```
<Body Background="|" BgColor=#FFFFFF Text=#000000 Link=#0000FF VLink=#800080
ALink=#FF0000>
```


|

This is my **first** web page!

</Body>

Now press the right-hand mouse button and select "Horizontal Rule" from the menu that will appear. Now the display should look like this:

```
<Body Background="" BgColor=#FFFFFF Text=#000000 Link=#0000FF VLink=#800080
ALink=#FF0000>
```

```
<Hr Width=95% Align=Center>
```

This is my **first** web page!

</Body>

Make another horizontal rule below your typing, so things look like this:

```
<Body Background="" BgColor=#FFFFFF Text=#000000 Link=#0000FF VLink=#800080
ALink=#FF0000>
```

```
<Hr Width=95% Align=Center>
```

This is my **first** web page!

```
<Hr Width=95% Align=Center>
```

</Body>

Now select the entire phrase "This is my **first** web page!", press the right mouse button, and select "Center" (there is also a toolbar button for this). Now look at your work in the browser.

Add paragraph breaks at the end of each line (some browsers will require this, some won't, but it is good HTML style). These are available from the right-click menu, or you can just type Ctrl-Enter:

```
<Body Background="" BgColor=#FFFFFF Text=#000000 Link=#0000FF VLink=#800080
ALink=#FF0000>
```

```
<Hr Width=95% Align=Center><P>
```

```
<Center>This is my first web page!</Center><P>
```

```
<Hr Width=95% Align=Center><P>
```

</Body>

This is important to understand: Normal line breaks don't mean anything to HTML -- there has to be an explicit tag that specifies a line break, such as `<P>`. There are two varieties of line break -- `<P>` (Ctrl-Enter), which separates paragraphs with a blank line(two line feeds), and `
` (Shift-Enter), which provides a single break between lines (one line feed). Experiment with these and don't expect normal blank lines (with no tags) to appear in the Browser view.

Using Arachnophilia

This section describes the general use of Arachnophilia, and some of the ways you can make Web page creation easier.

To create a blank document, not necessarily an HTML document, press File ... New Blank Document or press the corresponding toolbar button.

To create an HTML document that includes the basic structure of HTML, press File ... New HTML Document, or use the leftmost toolbar button, or use the right-click menu in an unoccupied part of the Arachnophilia program window. You will see the "Create HTML Document" dialog, which allows you to choose text and background colors and an optional background graphic as well as some other less-often-used options. You may also change these options after you have created the document by placing the cursor on the item of interest and pressing the right mouse button. You may then change the name of the background graphic or your color choices, as well as several other things.

To open an existing HTML document, select File ... Open, press the corresponding toolbar button, or use the right-click menu.

To save a document, choose either File ... Save or File ... Save as, or press the corresponding toolbar button, or use the right-click menu within the document display.

You may find and/or replace any text string you wish using either Edit ... Find or Edit ... Replace if you want to replace one text string with another, or the corresponding toolbar buttons. You can specify two non-printing characters using special symbols. The special symbols are ^p or ^j for line feed (the division between lines in a text editing document), and ^t for tab character.

You can Cut, Copy and Paste within or between Arachnophilia documents and between Arachnophilia and other Windows applications, although advanced features such as drag-and-drop are often easier to use.

You can test the result of your work at any time with up to six user-identified Web browser programs.

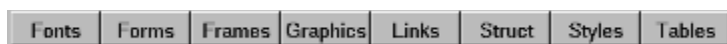
Beyond these normal behaviors, be sure to see Advanced Features of Arachnophilia for some spectacular capabilities that will really make your Web Pages stand out from the crowd!

Advanced features of Arachnophilia

Arachnophilia has many ways to make your work easier. You may drag a selection from a Arachnophilia document into another Arachnophilia document and drop it at a chosen location, or you may drag a Arachnophilia selection into another Windows application, or the reverse. But that is just the beginning.

Toolbar Customization

You can use the toolbar selections at the bottom of the main display to launch many independent toolbars that can be positioned in any convenient way. The default toolbar selector bar looks like this:



Each toolbar contains a group of related functions. You should explore these toolbars to gain an understanding of their functions.

After you have examined the toolbars, you can change them if you wish! All the commands on these toolbars come from special Toolbar Definition Files (with the suffix TBD), located in your Arachnophilia program directory. You can change the commands on the toolbars, or create entirely new toolbars, by editing the contents of these files.

There is an Excel 7.0 spreadsheet named “arachnophilia command sets.xls” located in your program directory that will create the entire set of TBD files for you. The most secure and efficient way to use this sheet is to make a backup copy of the original, and edit the contents of a copy.

If you make a copy of one of the worksheets within the Excel spreadsheet and give it a different name, the spreadsheet will create a new TBD file for you with the name you have chosen. If you just add new commands and edit existing commands, these changes will appear the next time you run Arachnophilia.

If you don’t have a copy of Excel, you may want to try loading the spreadsheet into another spreadsheet or database program, or you may edit the TBD file directly in a text editor, although this is more difficult.

Power Undo

Most of the commands related to the actions of Arachnophilia are reversible. If you don't like the outcome of your action, you may always undo it with the menu Power Undo command or toolbar button:



If you use this button, and you don't like the outcome, you can press the Redo button to go back to the original version. Arachnophilia will save 16 versions (16 Undo and 16 Redo) of your pages in this way, so you can recover from experiments.

After an undo, the HTML tag coloring will be lost. To get them back, simply press the tag recolor button on the toolbar, or select Tools ... Recolor Tags.

Smart Right-click

Some of the toolbar buttons allow you to edit a file, link, or color. To use these buttons, position the cursor in the area on interest and press the appropriate button. Another, faster way to accomplish this is to position the cursor on the link or color of interest and press the right mouse button. In most cases Arachnophilia will identify the target and launch the correct selection dialog. If the link contains the name of a graphic image, the most recent graphic directory is displayed and you can choose a replacement for the current graphic. If the link contains the name of Uniform Resource Link, you will see the directory that contains this file type, and so forth.

In general, if you prefer to right-click the mouse, first position the cursor without selecting anything to let Arachnophilia decide which action is appropriate, or select one or more characters to launch a menu of choices, or just use a toolbar function.

See also:

[Arachnophilia File Import Methods](#)

Arachnophilia File Import Methods

There are three principal ways to import a fully-formatted text document from another application:

- 1? Drag a Rich text Format (RTF) file icon from Windows Explorer to the Arachnophilia main window and “drop” it. Arachnophilia will then ask whether it should convert the document into HTML code.
- 2? Use the File ... Open File ... RTF File function to import Rich-Text formatted files which contain the formatted text of interest.
- 3? Drag a section of a document between the source program and a new, blank RTF document in Arachnophilia.

Arachnophilia will automatically convert your text into HTML code, keeping fonts and font colors, sizes, styles such as bold, italic and other formatting, centered titles, and many other characteristics. It will create HTML tables from your tables, preserving styles as above. It will create multilevel indented outline-format lists, both numbered and bulleted, also preserving the formatting of the original document.

This means you can work in a full-featured word processing program of your choice, and, when you are satisfied with the appearance of your text, tables and outlines, just move them into Arachnophilia using one of the methods above.

A special note about table formatting: In the RTF file format, the positioning of text within table cells is not preserved. Therefore, if you want to control the position of text *within* the table's cells, you must either:

- (1) convert the table into a tabbed list in the word processing program and then select left, center, or right positioning on a line-by-line basis before importing the table, or if this is not possible,
- (1) Set the positioning within Arachnophilia manually, after the table has been imported. In this example you want only the text in the middle cell of the row to be centered (you add the text in **red**):

```
<Tr><Td>data</Td><Td align=center>data</Td><Td>data</Td></Tr>
```

In this example you want the text of the entire row to be centered within the cells (you add the text in **red**):

```
<Tr align=center><Td>data</Td><Td>data</Td><Td>data</Td></Tr>
```

This is a feature that is supported in most modern browsers.

Custom Definitions

You have complete control over the content of the toolbar structure in Arachnophilia. When you run Arachnophilia, the toolbars are created from simple text files located in the program directory. These files have the suffix .TBD, which stands for Tool Bar Definition.

I have provided a spreadsheet program compatible with Excel 7.0, also located in the program directory, that contains the content of these files and is an easy way to add new toolbars or edit the contents of those already defined.

The file is named “arachnophilia command sets.xls,” and, if you own Excel 7.0, you may simply click on the program icon to run the program. You may add any tags you wish, or change those that exist to suit your personal preferences. If you copy one of the worksheets within the workbook and give it a new name, this will become a new toolbar in Arachnophilia.

The user-command entries may contain some special tokens -- the tokens “\n” or “^p” which result in a line feed when the command is carried out, the token “\t” which results in a tab (you cannot use actual tabs because tabs are used in the .TBD file format to separate fields), and the vertical bar “|” which identifies the user's selection before the command is issued. For example, let us say that a new HTML tag is invented called the SHOUT and it has this syntax: <SHOUT>this phrase</SHOUT>. To define a command to handle this new tag, you would enter:

Button name: “Shout” (this should be relatively short).

Command: <SHOUT>|</SHOUT>

Tool Tip Text: “Shout Tag”

Status Bar: "This is the new SHOUT Tag"

After you have entered this definition, a new command button will appear on the specified toolbar, and if you select a phrase and press the button, the tags <SHOUT> and </SHOUT> will enclose you selection.

Beyond these formatting procedures, there are some special commands, enclosed in braces, that execute Arachnophilia program commands. Here is an example:

One of the standard tags (from the “Graphics-Misc” toolbar) creates a new graphic image tag, and (by launching the graphic selection dialog) fills it with something you have chosen. Here is the tag:

```
<IMG src="[NewGraphic]" alt="">
```

This tag creates an editing entry just like the others, but, because of the special entry [NewGraphic], causes a file selection dialog to appear as the tag is being constructed. If you choose a graphic named “mygraphic.jpg,” the resultant tag would look like this:

```
<IMG src="mygraphic.jpg" alt="">
```

After you have defined this tag, you may simply right-click it and Arachnophilia will know what to do, but the first time, when there is no filename to offer guidance, this tag is a way to automatically launch a file selection dialog.

Here are all the currently defined special tags and their meanings:

<i>Tag</i>	<i>Meaning</i>
[newcolor]	Selects a color, inserts it into a tag if one is specified.
[newhtml]	Selects a local HTML file, inserts it into a tag if one is specified.
[newurl]	Selects an URL file type, inserts its contents into a tag if one is specified.
[newgraphic]	Selects a graphic file type, inserts it into a tag if one is specified.
[newsound]	Selects a multimedia file type, inserts it into a tag if one is specified.
[newcgi]	Selects a CGI file type, inserts it into a tag if one is specified.
[cut]	Performs the clipboard Cut function, disregards any text that may accompany the tag.
[copy]	Performs the clipboard Copy function, disregards any text that may accompany the tag.
[paste]	Performs the clipboard Paste function, disregards any text that may accompany the tag.
[save]	Saves the current document.
[saveas]	Saves the current document under a new name.
[date]	Insert current date at cursor position.
[datetime]	Insert current date and time at cursor position.

One of the worksheets in the Excel workbook doesn't become a toolbar, instead it becomes the right-click menu. This sheet has the internal name "RightClickMenu," which is how Arachnophilia identifies it. The commands you see when you press the right-click menu are located in this sheet.

If you do not own Excel 7.0, you can still add to the toolbars and edit their contents using a word processor or even a simple text editor -- this is just less convenient. In this case, if you want to add a new toolbar, simply copy one of the TBD files in the

Arachnophilia program directory, change its file name and its internal name, and create the commands that meet your requirements.

The Right-Click Functions and Menu

When you are editing text, you may press the right mouse button to gain access to some special functions and a shortcut menu. Here is how:

To let Arachnophilia choose the appropriate action, simply place the editing cursor at the desired location, without selecting anything, and press the right mouse button. If Arachnophilia finds itself pointing at a color definition, the color dialog will appear. If Arachnophilia detects a file name, a link, a resource, or one of many other things, the appropriate dialog box will appear, allowing you to make a new choice.

Example: if you position the cursor this way (the cursor is marked with the vertical bar character "|"):

```
This is a <Font Color=#FFF|F80>Color Block</Font> to emphasize it.
```

Arachnophilia will detect the color definition and display the color dialog box. If you point at this:

```
<IMG src="book|case.jpg" alt="">
```

Arachnophilia will open a "Change Graphic Image" dialog box, directed to the most recently accessed graphic file directory. In general, if you place the cursor on a color or a resource specifier string, Arachnophilia will automatically sort out your intentions.

If Arachnophilia cannot detect any of these things, or if you select some characters before pressing the right mouse button, a shortcut menu will appear which allows you easy access to many of the most-used HTML tags.

The File Menu

The File menu has the normal commands for opening, saving, and closing documents, plus some special commands uniquely related to Arachnophilia's purpose.

The File ... New File command opens one of many kinds of document. Some document types are treated in special ways, so be sure that you choose the correct document type.

The File ... Open File command allows you to choose which kind of document you are opening. Arachnophilia remembers which directory contains which file type, so this makes it easy to keep track of file locations.

The Close All command closes, and if necessary saves, all your documents at once. This is an easy way to close part of a project and clear your workspace for another task.

The Save All Changed Files command simultaneously saves all open documents that have been changed in any way, with just one mouse press.

The Print and Print Preview commands allow you to print your document, but not like a word processing program would, with page numbers and margin control. This command is primarily used to acquire a scratch-pad style hard copy for programming purposes. For more control of the printed output, I recommend that you transfer your document to a word processing program for printing.

The Send Mail command allows you to e-mail a copy of your work.

The Edit Menu

The edit menu contains two commands to Undo or Repeat an action as well as the usual Cut, Copy, and Paste commands.

Undo works in an obvious way -- it simply copies the entire document into temporary storage at each issued command (not during keyboard text entries), and, if you decide you don't like the outcome of a command, you can simply undo it. If you don't like the undo, you can repeat the action. This method works because most Web pages are relatively short and therefore occupy a small amount of memory.

That's the good news. The bad news is, if you are working on large documents, you will run out of memory by using Undo. So you have the option of turning off this feature (see [Tools Menu](#)). if you find yourself running out of memory in seemingly normal circumstances, try turning off Power Undo.

The Find and Replace commands will search the entire document, or from the cursor to the end of the document, or even all open documents, depending on your choices. These dialogs accept two special characters that represent non-printing characters in your document: ^p or \n represent a line feed, and ^t represents a tab character. This means if you type '^p' (without the ' characters) as part of your entry, Arachnophilia will search for line feeds (the character that separates lines in your document). The replace entry also accepts these special characters.

The Tools Menu

The Tools menu contains some explicit commands that are normally carried out automatically.

Convert RTF to HTML

This command, enabled only for RTF documents, converts them into HTML pages using the Arachnophilia RTF Translator Module. Options for this translation are located in the Tools ... Options menu (see below).

Insert HTML Tag

This submenu is a backup for other ways to insert tags that rely on the launching of a file dialog box or a color dialog box.

Recolor Tags

This command, for which there is also a main toolbar button, recolors the tags in an HTML document. After extensive editing, or the use of Undo, the HTML tag colors may not be correct any more. This button will restore tag colors.

Analyze Site

This command will analyze your site for internal consistency. See [Arachnophilia Site Analyzer](#) for more information.

Beautify Script

This command will find and correctly indent program code listings, such as Perl/CGI scripts and C/C++ programs. While indenting the code listing, it finds any unmatched pairs of the structural characters “{,},(,)”, thus performing a simple syntax check on your program. If it finds any unmatched structural characters, it will alert you to this fact with an error message. For more on this function, see [The Arachnophilia Beautifier](#).

The **Tools ... Options** menu takes care of several Arachnophilia setup functions.

Choose Default Font

Default Font allows you to choose a different font and size for the editing display. I recommend a fixed-pitch font such as Courier New, because some Arachnophilia functions look much better in such a font is used.

Choose Background Color

This option allows you to select a background color that is different than the Windows default color. This function permits you to see (to some extent) what your HTML page will look like

when viewed on a browser. Just choose the same color for your HTML editing screen that you have chosen for your browser display.

Use Word Wrap

This feature conveniently keeps all the text in view, but sometimes you want to allow the lines to extend off the screen to the right. For example, when you analyze your site using the [Arachnophilia Site Analyzer](#), you will want to use a fixed pitch font and turn off word wrap for best results.

Enable Power Undo Feature

This function enables or disables the Undo feature. If you are experiencing out-of-memory errors or are working with large documents, disable the Power Undo feature to conserve memory.

Convert tabbed lines into Tables

This function controls how Arachnophilia interprets tabs while importing Rich Text Format (RTF) documents and converting them into HTML. If your document contains tabs that are not part of tables, Arachnophilia may get confused and turn them into tables you didn't intend. To control this behavior, simply select "Convert tabs into spaces," but remember that you did this, because later you may want to create tables. In order to do this, you must re-enable the default option "Convert tabbed lines into Tables."

Convert Extended Chars into HTML

This function converts extended characters (characters with numerical codes greater than 127) into the equivalent HTML representation, which is `&#nnn;` for a given character code `nnn`. This method guarantees that the extended characters will appear correctly on any server system, including those that will not accept character codes greater than 127. If your server accepts extended characters and you want to reduce the size of your page, or if you want to be able to see and edit the original characters in the Arachnophilia HTML file, disable this function.

Convert "<" and ">" into Visible Chars

This function controls how Arachnophilia interprets these two special characters. Normally these characters enclose HTML tags, so if they appear in normal text, they must be treated in a special way. If this function is enabled (the default), these characters will be made visible in the HTML page, so a line reading "5 is > 3" will be displayed correctly. Conversely, if you have an import document that contains HTML tags, and you want to preserve these tags in their literal form, disable this function.

Show full pathnames on title bar

In large projects that span multiple directories, this option allows you to see the entire directory path to your document, thus avoiding confusion when files have similar names.

HTML Tag coloring options

Arachnophilia will color the tags in an HTML document, making it easy to distinguish among normal text, tags and the definitions within tags. These buttons allow you to customize the color choices. Your choices are preserved between Arachnophilia work sessions.

On slower computers, you have the option of turning off tag coloring.

The Arachnophilia Site Analyzer

This function, available at Tools .. Analyze Site, create a cross-reference list of your site's files, organized as a tree structure.

The basic structure of a Web site is similar to a tree, with a file (usually named "index.html") as its root. A visitor to your site first sees index.html, then branches out from there through other pages you offer and to other places on the Web by way of hypertext links, which form the branches of the tree.

As development progresses, some things may happen that you might prefer did not. Here is a short list:

1. You may decide against using a particular resource in your pages, and you may then change the identifying tag to point at another resource instead. But, being human, you forget that the resource is still stored in your site directory.
2. You may make a typographical error in identifying a resource, and then not test that tag to verify that it is working.
3. Someone on the Internet may change their identifying Uniform Resource Locator address without visiting your house and telling you personally, and delivering a box of candy to take you over the disappointment. Thus, that tag in your pages will no longer point to anything desirable.

The Arachnophilia Site Analyzer will list your site's resources in three sublists, organized as tree structures:

1. Referenced on-site resources, meaning resources that both exist and are connected to the tree.
2. Unreferenced on-site resources, meaning resources that exist but *do not appear to be* connected to any part of the tree.
3. Calls to unresolved or off-site resources, which are references to Internet Uniform Resource Location specifications, and also any calls that cannot be connected to a known resource.

List (1) is a reassurance that there are resources that both exist on your site and that are referenced in your pages.

List (2) show any resources that do not appear to have references. The Site Analyzer cannot detect all uses of a resource, so don't just delete all the items on this list without first trying to determine if they are in use.

- Site Analyzer won't detect references to resources inside CGI scripts.
- Site Analyzer also cannot detect the content of any subdirectories that follow the Web naming convention that only requires the name of the directory. In this case, the browser detects a file named "index.html" on its own and launches it. To solve this problem, just add the page name to the end of your calling tag -- instead of calling for "mysubprogram," call instead for "mysubprogram/index.html." The browser will treat these two calls the same way, but Site Analyzer is then able to add that "branch" to its tree.

List (3) generated by Site Analyzer contains all off-site references in your pages (mostly Internet sites), sorted alphabetically, so you can easily test their validity using your browser.

When the analysis is complete, a resizable dialog box opens. You may click on one of the categories in the same way you open a directory in Explorer. The first tier (or level) consists of the categories mentioned above. The second tier consists of actual resources or resource names. The third tier, by far the most interesting, lists the HTML pages that called the resource.

The third tier is by far the most useful aspect of the Site Analyzer. If you want to see the original resource reference, just click on the page's name in the third tier, and that page will be opened and the cursor will be placed next to the original call. This arrangement is particularly useful in list (3) (calls to unresolved or off-site resources), because this is where typographical errors tend to wind up. This feature makes it easy to correct such errors interactively, by opening each page and correcting links that have no destination.

In the "Unreferenced Resource" list there obviously is no third tier, so if you click on a resource in this list, you are offered the option of deleting the resource. Be careful, for the reasons outlined above -- there are many reasons why Site Analyzer might not find a reference call.

After you have made a number of editorial corrections, you will want to regenerate the list -- just close the Site Analyzer dialog and re-select Tools ... Analyze Site.

The Arachnophilia Beautifier

Nothing looks worse than a computer program with ragged indentation. Have you ever wondered how programmers get their program listings to look so orderly? Most use a program available on UNIX called “cb,” meaning “C Beautifier.”

But this program isn’t generally available outside the UNIX world, so I decided to add one to Arachnophilia. If you have one of several file types on display, you can clean up the code with a click. Here are the valid file types:

- HTML pages with JavaScripts in them.
- Perl programs.
- CGI programs.
- C and C++ programs.

The beautifier behaves differently for each of these, but it will produce a nice-looking result for all. Just put your program on display and select Tools ... Beautify Script. And, if you don’t like the result, just Undo it.

There is one error message that can be generated by the Beautifier -- “Indentation Error.” This error arises when there are unmatched braces or parentheses. There must be an equal number of “{ and “}” symbols, and “(“ and “)” symbols in the script to avoid this error. Therefore this consistency check is a simple check of the script’s overall syntactical correctness. Any errors detected by the Beautifier would also be caught by the language compiler or interpreter later, so this saves time and effort.

Locating the syntax error is quite simple -- just follow the script until the text seems to be too far from, or too close to, the right margin. Directly above will be the site of the error.

The Commands Menu

Most of this menu is a duplicate of the functions offered by the user-defined and default toolbars. It is mostly included for compatibility with keyboard-only operations (which I confess I haven't tried), and for those who prefer to browse a menu tree.

In each submenu you have the option of launching the associated toolbar, so it is an easy way to review the contents of those toolbars, as well as gaining access to commands you don't use very often.

The Preview Menu

The preview menu allows you to activate the selected browser, or quickly switch browsers. The Launch Selected Browser function is duplicated on the toolbar.

At the bottom of this menu is an option to identify browsers. This is how you add browsers to your Arachnophilia installation. You may identify up to six browsers. Using multiple browsers is good practice, because no two browsers act exactly the same. You may want to change the coding of your pages to accommodate these differences.

Arachnophilia uses a feature of Windows called DDE (Dynamic Data Exchange) to switch from the main Arachnophilia display to the browser display without requiring the user to relaunch the browser each time. Basically, after launching your browser the first time, you may simply press the main toolbar preview button and the browser's display is updated with the new information. This system works on nearly all browsers -- on one older version of Microsoft Internet Explorer (2.0), because of a coding error, you have to touch the browser's title bar to refresh the display. For owners of this browser, I recommend an upgrade, free at the time of writing.

The Window Menu

The Windows menu allows you to arrange opened documents in several ways. It also lists open documents and allows you to switch between them.

Creating Frames

A complete explanation of frames is far beyond the scope of these help pages, but you can get started with a simple example, and then you may complete your education using the resources available in bookstores and the Web itself.

The use of frames makes it possible to break the browser window into several smaller windows, thus allowing you to present complex information in a flexible, accessible way. You may wish to display an index of resources while showing each selected resource in a separate window -- frames let you do this.

The Frames toolbar (of the default toolbars) has the most important tags for frame creation. At the time of writing, there are still many browsers that do not support frames, so it is a good idea to include a no-frames page along with your frames-capable page (see example below).

The first tag to understand is called `<Frameset>`. This tag defines either rows or columns, depending on which option is chosen. If you write `<Frameset rows="20%,20%,*">`, you will define three rows, the first two each occupying 20% of the browser's height, and the third row (marked with *) occupying the remainder of the space.

A command of `<Frameset cols="20%,20%,*">` does exactly the same thing, except in this example, columns are defined instead of rows.

You may create multiple columns and rows by nesting the `<Frameset>` tags. Here is an example:

```
<HTML>

<Head>
<Title>Frames Example</Title>
</Head>

<Frameset rows="20%,*">
  <Frame src="toprow.htm" name="title_area">
    <Frameset cols="20%,*">
      <Frame src="leftcol.htm" name="index_area">
        <Frame src="display.htm" name="display_area">
      </Frameset>
    </Frameset>
  </Frameset>

<Noframes>

(Put a no-frames version of your page here)

</Noframes>

</HTML>
```

On a frames-capable browser, this example would produce a display with a title row at the top, an index column at the left, and a large display area at the lower right. By the way, most frames-capable browsers will refuse to display anything until the subsidiary pages have been written.

A typical use for this frame setup is to change the contents of the *display area* by clicking links in the *index area*, while keeping both in view. Each link would look more or less like this:

```
<A href="choice.htm" Target="display_area">Make this choice!</A>
```

Remember when we set up the frames in the above example? We remembered to use the "name" option to give each window a name. Now we can refer to the windows by name from any other pages we create. This greatly increases the control you have over the frame system and also reminds you, with a well-chosen name, which window is the target for each page.

Obviously, using frames means that you have many more pages to write, but each will probably be smaller. The first page sets up the frame system and usually provides for a non-frame version of its contents. All subsidiary pages use the frame system that is set up in the first page.

This has been only a brief introduction to the subject of frames -- there are more frame tutorials and resources at www.netscape.com and www.microsoft.com as well as elsewhere on the Web.

An Introduction to JavaScript

Java is a programming language that can be used to deliver platform-independent applications across the Web. At present, the language has two forms: Java, which is a compiled, highly structured language for large projects, and JavaScript, which is a scripting language that requires much less effort than full Java. In exchange for this convenience, JavaScript has some limitations that exist mostly for security.

With JavaScript, you can create interactive programs that provide immediate results without requiring separate CGI scripts or multiple pages -- the results appear right on the page that carries the program! Or, if you prefer, you can launch separate pages and applications for more involved tasks.

The JavaScript program source is delivered along with the Web page that carries it, so JavaScript is presently regarded as public domain -- you cannot protect the source code for your JavaScript program. JavaScript also cannot open, read or write files or make system calls -- this is why it is relatively secure.

Once again, there is much more to the JavaScript language than can be covered in these help pages, but I will provide a few examples to get you started, then you can complete your training using textbooks and web resources.

The Frames toolbar (of the default toolbars) has several shortcuts to aid JavaScript program development. The first is "script," which, when activated, looks like a combination of HTML tags:

```
<Script LANGUAGE="JavaScript">
<!-- Hide this from older browsers

(the JavaScript program goes here)

// end hide -->
</Script>
```

The `<Script></Script>` tags set the JavaScript program apart from the rest of the HTML code. But, since there are many browsers that do not support JavaScript, you may want to hide the JavaScript code from them. The comment tags accomplish this, so the JavaScript program listing won't spoil the appearance of your page.

Now let's just do something silly to get started. Position the cursor in the middle of the Script block and press the "Alert" button on the Frames/Java toolbar. This will be the result:

```
<Script LANGUAGE="JavaScript">
<!-- Hide this from older browsers

alert("|");

// end hide -->
</Script>
```


As in previous examples, the vertical bar "|" shows the position of the cursor. Now Type "Hello World!" and press the Arachnophilia preview button. You will see a dialog box with the message "Hello World!"

Now let's write a more complex program. Move the cursor above the "alert" line and press the "prompt" button on the Frames/Java toolbar. You will see this:

```
<Script LANGUAGE="JavaScript">
<!-- Hide this from older browsers

var response = prompt("|", "y/n");

alert("Hello World!");

// end hide -->
</Script>
```

Now type "Enter a phrase:" where the cursor is located and then change the "alert" line to read:

```
alert("You Typed: " + response);
```

Then press the Arachnophilia preview button. You will see a prompt box that asks you to type something, then, after you have, you will see an alert dialog that will show what you typed.

There is nothing sacred about what the "prompt" button places in your program -- it is just a typical entry. For example, you could use any variable name instead of "response," and you might want to say "Type your response here" instead of "y/n" in the default area. But as you learn more about JavaScript, you will understand this -- and you will find many more things that JavaScript can do!

JavaScript has been designed to resemble the C++ programming language as much as possible, to accommodate the many people who have learned that language. Thus, if you are familiar with C++ you should feel right at home with JavaScript.

JavaScript can do many things not covered in this brief tutorial, so I recommend that you browse your local bookstore and the web for references to JavaScript. Also, because the source code of JavaScript-equipped Web pages is located right in the page, you can search for well-designed JavaScript applications and simply download them to see how they were written.

A sample of JavaScript applications can be found at <http://www.arachnoid.com> (the author's site) and tutorials and resources are available (or should become available shortly) at www.netscape.com and www.microsoft.com. Or you can simply use your favorite web search tool with the search string "JavaScript."

Working with Front Page

Microsoft Front Page® does some things very nicely. There are also some things it won't do at all, and finally (in version 1.1) there are some syntactically correct HTML tags that Front Page will simply destroy.

When one imports existing HTML code into Front Page, one must change many parts of otherwise acceptable HTML code to prevent Front Page from arbitrarily changing or losing parts of the original code.

Arachnophilia can work with Front Page and minimize some of the latter's undesirable behaviors. The first step is to make Arachnophilia one of the editors that Front Page uses for special tasks. To do this, run Front Page Explorer, select Tools ... Configure Editors, click Add, enter a file type of "*.htm", Type "Arachnophilia" for "Editor Name," then locate Arachnophilia using the provided browser function.

The second part of the process is to import your existing HTML pages into the Front Page environment, a complex process beyond the scope of this document. Refer to the Front Page help screens for more on this process.

Once having imported your pages, you will need to find some types of HTML code that Front Page will object to and then change in ways you probably won't like. Here is an example:

I use an image tag to produce a horizontal line. It looks like this:

```
<IMG src="redchip.gif" alt="" height=2 width=95%>
```

This tag uses a small image (that can be as small as 1 by 1 pixel) to make a color horizontal line. My horizontal line automatically adjusts to the width of the screen, just like the default HTML horizontal line tag, but it can be any color you desire. This tag is completely acceptable HTML code, it works in all recent browsers, but Front Page will find and eat this tag, forcing the image to have the width and height of the original graphic file, thus destroying its usefulness as a line.

To solve this problem, I have included a special tag called a "bot" (Microsoft's term) that essentially hides selected HTML code from Front Page. The Bot tag is located on Arachnophilia's "Struct" default toolbar. When the Bot tag is used, the result looks like this:

```
<!--VERMEER BOT=HTMLMarkup StartSpan -->
<IMG src="redchip.gif" alt="" height=2 width=95%>
<!--VERMEER BOT=HTMLMarkup EndSpan -->
```

This tag can be applied to any HTML code that Front Page objects to, but that you want to keep. And believe me when I tell you, there are plenty of tags that Front Page will have for lunch. At the time of writing I have used Front Page for only a few hours, but most of that time has been spent trying to keep Front Page from disabling some of the best parts of my Web pages.

Editorial comment: Front Page is a very ambitious project that will make HTML page creation

available to many people otherwise daunted by the complexity of HTML page development. But, like so many applications, it overemphasizes the designer's wishes and almost completely ignores the user's wishes. In many cases, Front Page will take syntactically correct HTML code and render it nonfunctional in attempting to fit the code into Front Page's narrow conception of what is acceptable HTML code.

Arachnophilia emphasizes the opposite -- it takes into account the needs of the user by providing field-customizable toolbars, as just one example.

Basically, with Front Page, Microsoft's software designers are in charge. With Arachnophilia, *you are in charge* (if you want to be). But Arachnophilia can't do some of the things Front Page can do.

The methods on this page provide a way for Arachnophilia to work with Front Page and (to some extent) keep Front Page under control.

The terms "Microsoft" and "Microsoft Front Page" are registered trademarks of Microsoft Corporation.

Autocopy Mode

Arachnophilia takes care of a difficult issue in Web page development automatically. This feature, called Autocopy, allows Arachnophilia to automatically copy your resource selections to your working directory. This has two effects:

- (1) All the selected resources for your page end up in one directory. This is a reliable way to maintain control over those resources.
- (1) Because all the resources are in the same directory as your Web pages, you may use relative links instead of absolute links. This is a way to make your pages more reliable and more portable, because you can move the entire package to any site and it will work as intended.

Instead of a link that, because it is located in another directory, looks like this:

```
<IMG SRC="c:\data\network\mywebpage\mypicture.gif" >
```

, using Autocopy, yours will look like this

```
<IMG SRC="mypicture.gif" >
```

because the resource is located in the same directory as the page that refers to it.

The Autocopy mode requires that two conditions be met:

- (1) You must choose a working directory for your HTML pages, and
- (2) You must enable Autoocopy when Arachnophilia asks you to.

The first time you try to attach a resource file to your page, if you haven't yet saved your Web page, Arachnophilia will ask you if you want to save it. If you respond with Yes, (highly recommended) then Arachnophilia will ask whether it can enable Autocopy.

After Autocopy has been enabled, no matter where you find your resources, they are copied into your working directory and their names are converted to the relative form.

The best part of Autocopy is that, when you have created and tested your Web page(s) at your personal computer, you can upload the entire package to your Internet Service Provider and it may work perfectly the first time. This is not an everyday occurrence.

The single exception to the Autocopy method is a link to a local HTML page. This type of resource will not be copied, because the page you have linked to may have its own links, and if the page is moved its links will stop working. For this reason, you should create all your HTML pages within the directory structure that you intend to maintain when you have uploaded your site onto the Internet.

HTML Page Setup

This dialog box appears when you create a new HTML page. You can use it to choose the color of text and all the HTML tags. You can choose a title for your page, the title that appears on the title bar of the browser that visits your site.

Your choices are preserved between uses of Arachnophilia, so you only really have to state your preferences once.

All the choices in this dialog can be edited later, so it won't hurt to pass this up by pressing "OK" if you aren't interested in sorting out the meanings of these options.

CareWare - What is it?

Arachnophilia is one example of a new software distribution system developed by the author. In the CareWare system, you don't owe any money to anyone. By the way, if someone made you pay for Arachnophilia, you were cheated. You should go back and demand a refund. Arachnophilia is freely available on the Internet and, because I own the copyright, no one has the right to make you pay for it except me, and I'm not going to.

But you are not off the hook yet. The "Payment" for a CareWare program is not monetary. You have to make a different kind of payment altogether. Let me explain.

Most Americans are totally dissatisfied with everything. It is too hot, too cold, too wet, too dry. If we have a free day, we are unhappy because we don't have two free days. And just about the time we figure out that we are supposed to appreciate the world as it is, we fall over and die.

So here's your payment for Arachnophilia:

Imagine you have only two hours to live:

- Is there something important you have to say to someone, something you might regret not having said? If you were to die, would that person always wonder what you really thought or felt?
- Is there a pretty spot you have always wanted to visit, sit under a tree, whatever?
- Have you ever experienced the shock of noticing how beautiful ordinary things are, once it dawns on you that you might not be around very long?

If you are an old person (like me):

- Do you speak to young people in a way that they will be encouraged to grow up and expect to be happy and productive?
- When you correct a young person, do you ask yourself "Is this mostly for my benefit, or mostly for his?"

If you are a young person:

- Do you try to be patient with old people, even though most of us are complete morons?
- Do you try to live in the world as though you belonged here, as though what you do matters to everyone, to the world itself, to you?
- Do you appreciate the small, free beauties of life, and not expect to buy anything very important?

Look at this list. If you already belong to this list, if this list already reflects your behavior and values, then you already own your copy of Arachnophilia. In a sense, you owned it before it was written.

If you don't feel a kinship with the statements in the list, then please do one or more of the things listed there. Maybe change how you talk to a young person, or someone whose life would be improved if you related to him or her differently. Or just allow a sense of wonder to re-enter your life, a sense that nothing is deserved and everything contains hidden beauty. And that sometimes beauty is not so much hidden as unobserved.

I would like it if you lived your entire life as though each day was your last, as though every small action mattered, in the way that it does when you've run out of time. But I am a realist -- if you do that for just *one day*, one day of saying the important things, of performing the kindnesses that naturally occur to us when each day might be our last, **then you will have paid me for Arachnophilia.**

I don't ask this because there is some definition of good behavior, some correct religious or philosophical viewpoint. I ask it precisely because there *isn't* such a viewpoint. We are all free agents, we get to choose. In fact, we *must* choose -- it's dangerous to let others choose for us. And no one gets to tell anyone else how to behave -- unless, of course, one is "selling" software using the CareWare system.

Paul Lutus, Ashland, Oregon

Arachnophilia Distribution Policy

Arachnophilia may be freely distributed and copied, so long as no fee is charged. Arachnophilia may not be made part of a product for which a fee is charged.

Notwithstanding the foregoing, Arachnophilia is not in the public domain and is © Copyright 1996-1997, Paul Lutus, who retains all present and future rights in and to Arachnophilia.

The determination of fitness of Arachnophilia for any particular purpose shall be solely with the recipient, and Paul Lutus shall not be responsible for any damages, direct, indirect, or consequential, that may arise from use of Arachnophilia.

HTML (Hypertext Markup Language) is the language with which Web pages are written. Originally invented by Tim Berners-Lee while at CERN, HTML is undergoing changes and improvements as time passes.

Browser: a generic term used to describe a program that can read and display Web Pages.

Examples: [Microsoft Internet Explorer](#), [Netscape](#) (click these links to acquire these browsers).

Web: a slang term for the Word Wide Web, the interacting entity composed of the Internet, HTML pages posted there, and browsers.

Web Page: A document written in HTML, that allows one to interactively gain access to other resources by way of active links.

Internet: A worldwide network of computers and people.

Unix: a very popular computer operating system. Much of the present ability of Windows and other similar operating systems was derived from Unix. Unix allows multiple users to run multiple processes on a single computer, and facilitates interconnection between computers.

Object: In computer parlance, an object is a package consisting of (1) data and (2) computer program elements designed to act on the data.

Hypertext: a scheme by which one can navigate through an information resource by activating links.

No, no. Don't confuse this with REALITY. This is WINDOWS. Reality requires a Browser.

Telnet: an older Internet communication protocol that resembles a text editor screen, except that you can issue commands to a remote system in what is called a "Unix shell session."

Internet Service Provider: A company that provides Internet access to individuals and businesses. Most good providers allow several megabytes of storage on their server per subscriber, thus allowing one to post a Web page with plenty of graphic images and other resources.

Relative Link: a Web page link that presumes that its target is a local resource, rather than spelling out exactly where it is located. Also see [Absolute Link](#).

Relative Link:

Absolute Link:

Absolute Link: a Web page link that presumes that its target is not a local resource, therefore spelling out exactly where it is located. Also see Relative Link.

Absolute Link:

Relative Link:

Tag: a term used in HTML programming to describe the fundamental unit of communication between the HTML script and the browser that tries to read it. Example: in an HTML script, "This is <I>very</I> important" will be shown by a browser as "This is *very* important" because the browser understands "<I>phrase</I>" to mean "Italicize all between <I> and </I>."

Rich text Format: a file format supported by many Microsoft applications that preserves the appearance of the document, but can be easily transported to another application. Arachnophilia supports Rich Text Format.

Hyperlink or Link: a reference in a Web Page that refers to another Web page or some other kind of resource. When you click on the link, you jump to what the link describes. Also see [Absolute Link](#), [Relative Link](#).

Select, Selection: A method to choose a part of a document. You most often select text in one of these ways: (1) hold down the shift key while moving the cursor across the desired text with the arrow keys, or (2) press the left mouse button and drag the mouse cursor across the desired text.

Pixel: Short for Picture Element. One of the matrix of graphic elements that make up a graphic image. A 640 by 480 VGA graphic image has 640 horizontal pixels per row and 480 vertical pixels per column, for a total of 30,720 pixels. Pixels are the basic units out of which graphic images are created -- each displays a color dot.

Arachnophilia: A fondness for spiders and Webs of all kinds. One having this condition is called an “Arachnophiliac.”

