Contents

Chapter 1

Introduction

1.1 Introduction

AWeb is a World Wide Web browser for the Amiga computer. It offers the following features:

- AWeb doesn't use Magical User Interface (MUI) (see section 1.1.1). It uses BOOPSI classes instead, which results in less memory usage and increased speed. The BOOPSI classes are partially custom designed for AWeb, and partially from the ClassAct kit (see section 1.1.2).

- AWeb can use a wide range of TCP-stacks: AmiTCP/IP, I-Net225, AS-225 or compatible. Without TCP stack running, you can still view local files.

- AWeb uses extensive internal multitasking, which leads to total asynchroneous and parallel network access. Images are starting to load while the document is still loading. It is possible to follow a link while the previous document is still loading. A separate network status window shows all pending network and local file accesses. All network accesses can be interrupted *immediately*.

- The HTML-2 standard is fully supported, including forms. Some of the most important so-called HTML-3 extensions are also supported. For buggy pages not conforming to the standard, AWeb offers a compatibile mode.

- AWeb supports the use of proxies for HTTP, FTP, Gopher and Telnet. Because some proxies can't handle forms or other special cases, the use of proxies can be temporarily disabled from the menu.

- The HTTP and Gopher protocols are supported internally. By using external programs, FTP and Mailto can be used too. HTTP user authorization is supported.

that action. The former must be immediate, while the latter in general may take a short time, up to a second or so, without confusing the user. (Depending of the kind of application, of course.) MUI applications treat visual feedback as being an application response, thereby failing miserably.

It should be obvious that blocking user input entirely and showing a busy pointer, like some people suggest, is not an option at all in the above examples.

I realize some gadtools gadgets have the same behaveour as MUI gadgets, and that's one of the reasons why I don't like gadtools very much, either. But at least gadtools *buttons* give immediate feedback.

Intuitivity

Gadtools offers a cycle gadget. Click anywhere in the gadget, and the next option is selected. You may like or may not like this kind of gadget, but the behaveour exists and is part of the Amiga look and feel.

MUI offers a gadget that looks exactly the same, but behaves differently. Click in the text part of the gadget, and nothing happens. Just a quick flash of a pop-up menu. I have clicked many times on gadgets of this kind and wondered why nothing changes before I realized it was a MUI application so this kind of gadget behaves differently.

The most important rule in GUI design is: be consistent. Gadgets that look the same must behave the same. Gadgets that behave differently must look different. MUI fails to be consistent with the Amiga OS look and feel.

I am aware that there are commodities that change the behaveour of the Gadtools cycle gadget into a pop-up menu. Basically these commodities suffer from the same fault: it changes the behaveour of the gadget but not its appearance.

More intuitivity

Fortunately, MUI 3.x has left the silly behaveour of configuring all your MUI applications from one preference program. But still, configuring your MUI application can be very difficult and not obvious for the casual MUI user. The best example of this is the ever returning question in many Amiga newsgroups: How do I get AMosaic to run on its own screen? I believe that the 18 (eighteen!) steps to perform to accomplish this, are far too many.

And even if the way configuring MUI aspects of an application is improved, there are still many MUI applications offering *two* settings requesters - one for the MUI aspects, and one for the application's own parameters. This is also confusing. For the user, a system like MUI should be transparent. The user should not need to know which aspect is controlled by MUI, and which by the application itself.

1.2.3 Licence

This licence does not apply to parts of the distribution not covered by the AWeb copyright. For the licence of these parts, refer to the respective documentation.

1.2.4 Distribution

The AWeb-II package, containing the AWeb browser, is distributed by AmiTrix Development.

Distribution otherwise than via AmiTrix Development is not allowed without prior written permission from both the author and AmiTrix Development.

configurations. Note that you can always change the configuration afterwards, using the settings requester.

- First, it asks if you have 2 MB of memory in your Amiga, or more. If you have only 2 MB of memory, then a different setup is needed to get the most out of AWeb. Have look at these tips to see how you should set up AWeb for a 2 MB Amiga.
- Also, you will be asked if you prefer small fonts or large fonts. The large fonts option will make AWeb look as closely as possible to the standard Netscape setup, using only standard Workbench fonts. Although the HTML standard doesn't impose specific fonts in any way, many pages on the World Wide Web are designed to look best using these fonts. If you plan to use AWeb on a small screen (less than 640 x 400), you might want to use the small fonts, or else the window will contain very little text. **Note:** The *large fonts* settings makes use of the *CGTimes* scalable font found on the Workbench Fonts diskette. Make sure you have this font properly installed if you choose *large fonts*.

2.2.2 Configuring the JFIF datatype

If you use the JFIF datatype (by Christoph Feck, TowerSystems), then please read this:

The JFIF datatype doesn't seem to handle shareable pens on public screens correctly under all circumstances. You have to install AWeb in the datatype or else AWeb will not be able to show inline JPEG images.

In the JFIF preference editor, you must add an application named **AWebIP** (AWeb Image Processing), and then select *Single-Pass Quantization* (in the GadTools version of the preference editor) or *One-pass* (in the MUI version).

Chapter 3

Working with AWeb

3.1 Starting AWeb

3.1.1 From the Workbench

You can start AWeb by a double-click on its icon.

AWeb can be specified as *default tool* in a project icon. You can use extended selection (shift-click) to select one or more project icons. AWeb will load the projects selected as local documents.

In addition, AWeb supports the following *tool types*:

URL=*url_or_filename*
: Specify this tool type one or more times to open these documents when AWeb is started. If you specify a local filename, use the LOCAL tool type too.

LOCAL
: If this tool type is present, the names in the URL tool types will be interpreted as local file names, rather than network URLs. This tool type is not needed if you select documents by their project icon, as mentioned above.

CONFIG=*settings_filename*
: Use this file as settings file instead of the default, AWeb.prefs. If the file doesn't exist, some default settings are used. Saving the settings will save to this file name.

HOTLIST=*hotlist_filename*
: Use this file as AWeb's hotlist instead of the default, AWeb.hotlist. If the file doesn't exist, it will be created the first time you add an entry to the hotlist.

The `Project / Open WWW` menu function or its shortcut, **AW**, will activate this field and preload it with "http://www." for even more convenience.

3.2.3 Status indicator

This field serves two purposes.

First, when browsing through a page, it shows the URL "behind" the link currently pointed to with the mouse. That is, when you click this URL will be retrieved.

Second, when a page is being loaded for this window, is shows the current state. If actual data is retrieved, a progress bar will appear showing how far the load process is. The progress bar will not appear if the final size of the document is not known on forehand.

3.2.4 Background status indicator

When one or more load operations are in progress, this indicator will show a little square. On every connection made, and on every block retrieved, the square will advance one step.

This indicator lets you know if there are still things loading in the background, and how rapidly they progress. If you want more detail, the network status window will tell you everything.

3.2.5 Back button

This button lets you walk back through the window history. The window history contains all pages viewed before *in this window*.

The `Navigate / Back` menu function, or its shortcut, **AB**, or the **Alt + cursor left** key combination, will do the same.

3.2.6 Forward button

This button lets you walk forward through the window history. The window history contains all pages viewed before *in this window*. Note that the window history is currently linear, without branches. That means that if you walk back a few steps, then retrieve another page, the window history "after" that page will be forgotten.

The `Navigate / Forward` menu function, or its shortcut, **AF**, or the **Alt + cursor right** key combination, will do the same.

3.2.7 Home button

This button retrieves the URL that is configured as your home page.

3.3 The menus

3.3.1 Project menu

The *Project menu* offers functions to open or close windows, fetch or save documents, or quit AWeb.

New window
Open a new window. Only available in the registered version.

Close window
Close the current window. Only available in the registered version.

Open URL
Clear the URL field () and activate it so you can type a new URL.

Open WWW
Preset the URL field () with "http://www." and activate it for maximum convenience if you want to load a WWW page.

Open local...
Opens a standard file requester. After you select a HTML file, the file will be loaded in the current window.

Search engines
Opens the local page extras/search.html which contains a quick interface to several search engines on the Internet.

View source...
Show the HTML source of the current page, using the viewer program that was installed as HTML source viewer.

Save source...
Opens a standard file requester. After you type a file name or select a file, the HTML source of the current page is saved. If the selected file already exists, you have the choice to:

- overwrite the old file,
- append the source to the old file,
- select another name, or
- cancel the save alltogether.

About...
Opens a window with version information. If the current window has an ARexx port, the name is shown here.

Quit
Quit AWeb after confirmation. If you confirm, all pending network operations are cancelled.

3.3.3 Cache menu

The *Cache menu* offers functions to flush the cache, and to save or flush the cached authorizations.

Flush images in current
Inlined images that appear in the current document are deleted from both the memory and disk caches. Note that if the image also appears in another document, it is undisplayed in that document, too.

Flush old images
Inlined images that do *not* appear in any currently displayed document are deleted from both the memory and disk caches.

Flush all images
The image cache is cleared completely. After this function, no images are left in the cache.

Flush documents
All documents that are not currently displayed are deleted from the cache.

Save authorizations
Save the current authorization details.

Flush authorizations
Forget all the current authorization details. Note that this flushes the internal authorizations cache only. To flush the disk cache too, you have to select `Cache / Save authorizations` afterwards.

3.3.4 Navigate menu

The *Navigate menu* offers functions to navigate in the document history.

Back
Walk back one document through the window history. The window history contains all pages viewed before *in this window*. Pressing the Back button, or using the **Alt + cursor left** key combination, will do the same.

Forward
Walk forward one document through the window history. The window history contains all pages viewed before *in this window*. Note that the window history is currently linear, without branches. That means that if you walk back a few steps, then retrieve another page, the window history "after" that page will be forgotten. Pressing the Forward button, or using the **Alt + cursor right** key combination, will do the same.

Home document
Retrieve the URL that is configured as your home page. Pressing the Home button will do the same.

Window history
Show the window history requester (see section 3.8).

Change settings...
Brings up the settings requester (see section 3.9).

Save settings
Save the current settings. This function will take a snapshot of your window positions first.

3.3.7 Help menu

The *Help menu* offers you more information.

Documentation
Shows the AWeb manual in this window.

AWeb home page
Retrieves the AWeb Home page. A TCP stack must be running, and you must be connected to the Internet for this function to work.

AWeb FAQ
Retrieves the AWeb FAQ. A TCP stack must be running, and you must be connected to the Internet for this function to work.

3.3.8 ARexx menu

The *ARexx menu* offers functions to start ARexx macros.

Start ARexx macro...
Opens a standard file requester. After you select an ARexx macro, that macro will be executed with the original window as default port.

User-configurable items
The remainder of this menu contains a list of ARexx macros you can configure yourself. Use the ARexx macro menu settings page for this.

3.4 The browser window

The browser window is the most important window of AWeb. It is the place where World Wide Web pages are displayed. On top of the window there are some gadgets, already explained in section 3.2.

3.4.1 Scrolling the page

You can scroll the window contents horizontally and vertically, provided the size of the document is larger than the window.

Of course you can use the scroll bars and arrow buttons to scroll, but AWeb also understands the following keys:

image is retrieved, and a standard save requester will pop up to let you specify a file name.

If the document or image is already in cache, it will only be saved, not retrieved again over the network.

Note you can also save a displayed image in this way. Just press the **Shift** key and click the image. This will work even for background images (only if background images are displayed): just shift-click somewhere in the background and you will be asked for a filename to save the background image.

If the image is also a link, shift-clicking the image could be abmiguous; therefore AWeb will save the image in this case. If you want to download the document "behind" the link, you can either select `Cache / Flush images in current` from the menu, and shift-click the upper left half of the icon, or click the image to load and display the document, and then select `Project / Save as HTML` from the menu to save the document.

3.5 Starting your TCP connection

Before you can access documents from the World Wide Web, you must start your *TCP connection* with the Internet. The connection is maintained by the TCP package you are using, also known as *TCP stack*.

You can start your TCP stack yourself before you start AWeb, or while AWeb is running. To make life easier, you can even let AWeb start your TCP stack when it is needed for the first time. For example, if you are browsing through local documents, and then follow a hyperlink to some document out there on the World Wide Web, then AWeb could start the TCP stack automatically.

To use this feature, you have to configure (see section 3.9) the start script (or program) for your TCP package.

If you have configured the stop script (or program) for your TCP stack, AWeb can terminate the TCP stack automatically after you quit AWeb. Note that it does so only if AWeb has started the TCP stack before, and only after asking for your permission to do so. If you started the TCP stack yourself, AWeb will not terminate it.

AWeb will try to start the TCP stack only *once*. If AWeb has started the TCP stack, and it appears to have failed for some reason, then AWeb will never try to start it again. You have to start it manually in that case.

3.6 The network status window

3.6.1 Purpose

The Network Status Window shows all pending network and local file accesses. There are possibilities to cancel selected transfers, or all transfers.

3.7.2 Using the list

Press the Hotlist button, or use the `Hotlist / Show hotlist` menu item to load a page with all links to all remembered pages.

With the *Hotlist button gives requester* setting, you can make the button pop up a requester. This is the same requester as for the `Hotlist / Maintenance...` menu item. The functions in this requester are described in the next section.

3.7.3 Maintenance

When you choose the `Hotlist / Maintenance...` menu item, a requester will open. In this requester you can change entries in the hotlist, move them around, group entries or delete them.

Use the window close gadget, or the **Esc** key to close the requester.

Change an entry

Select the entry you want to change, either with the mouse or with the cursor up and down keys. Now you can change the name of the entry and the URL that it points to.

Move an entry

Select the entry you want to move, either with the mouse or with the cursor up and down keys. Now you can move the entry up and down in the list with either the arrow up and arrow down buttons, or with the **Ctrl** + cursor up/down keys. The entry will step into open groups, but will skip closed groups.

If you move a group title, the entire group will move.

Add an entry

Use the *Add a link* button in case you want to add an entry to the list and type the name and URL yourself.

Group entries

Use the *Add a group* button to create a group, then change its name. Now you can add entries to the group, or move existing entries into it. Entries within a group will be displayed indented in the list. You can include other groups within a group, to any level.

Click on the little arrow next to the group title in the list to open or close the group. If the group is closed, its members are not visible, making the list clearer. Also, you can use the **Enter** key to open and close a group if the title is selected.

If the history window is already open, it will pop up to front, and will be activated. Also, the selected window will change to the window for which you opened the history.

3.8.3 The history list

In the list, all visited pages are shown.

The **first column** contains the *window number*. This is the same number as appears in the title bar of the window.

The **second column** contains the *mainline indicator*. This is a little arrow. Suppose you retrieve pages **A** - **B** - **C** - **D**, then go back to **C**, back to **B**, and then retrieve page **E**. In this case the mainline is **A** - **B** - **E**. These are the pages that the back and forward buttons will walk along.

The **third column** contains the *title* of the page, or the URL if no title is known.

The **fourth column** contains the *cache indicator*. If the 'in-cache' symbol is present, the page is still in AWeb's cache. Going back to this page won't need any network access.

If you selected *natural* or *mainline* order, the current document will be highlighted. The two fields below the list show the title and URL for the highlighted page.

3.8.4 Redisplay a page

Select the page you want to see again from the list. You can click on the entry, or use the arrow keys to move the highlight. Then click the *display* button, or hit the **Enter** key.

Another way to redisplay a page is to doubleclick on the entry.

If you have the History window auto close setting selected, the window will close automatically.

3.8.5 Filtering

If the *Filter* checkbox is selected, the list will only show entries for the window number selected in the *window* field.

If you deselect the *Filter* checkbox, the list will show entries for all windows.

3.8.6 Ordering

With the *Order* chooser, you can select how the entries in the list should be ordered.

Natural
Natural ordering shows all displayed pages in the order you have displayed

In the *command* string gadget, you can type in a command. It is advised that you specify the full path, or else AWeb might not be able to execute the command. You can click on the button to pop up a file requester, so you can easily pick the command.

In the *arguments* string you specify the arguments for the command. Any text you type here will be passed to the external program, but the first two or three **%s** instances will be replaced by parameters from AWeb. These parameters differ for each command, but you can click the 'arrow' gadget to pop up a short descriptive list.

Instead of a program, you can specify a DOS script. If you do, make sure that the *script* bit for this file is set. You can set this bit either by the DOS command protect *script_name* +s
or by the Workbench Information program (select the *Script* checkbox).

Note that an external program setting will *only* be recognized if *both* fields, command *and* arguments, are supplied.

3.9.5 Closing the requester

The bottom region of the requester contains three buttons:

Save
Apply all changes and save the settings. The next time you start AWeb the same settings will be used.

Use
Apply all changes for this session only. Unless you save the settings later using the `Settings / Save Settings` menu item, they will be forgotten the next time you start AWeb.

Cancel
Don't apply changes.

3.9.6 Browser 1: Fonts

On this settings page, you can change the font and style AWeb should use for different types of text.

The first column in the list contains the HTML tag for the type of text. The second column contains the current font and style selected. Note that *Normal* is not a HTML tag, but merely denotes normal text that is not subject to text style tags.

A brief description of the meaning of the HTML tag in the selected row is displayed below the list.

The Ff button pops up a standard font requester, where you can change the font, the size and the style.

Change the underlining

The checkbox determines whether links should be displayed underlined or not. If underlined is selected, *new links* are underlined with a solid line, and *visited links* will have a dashed line.

If this checkbox is selected, images that are links have a border in the appropriate colour.

Change the cycle field

Many people use a commodity that turns cycle gadgets into popup menus. In section 1.1.1 you can read why this is not a good idea. Because a cycle gadget certainly has its drawbacks, AWeb offers the possibility to display a cycle field in a form as a list. Note that selection fields with more than 5 selections, or with multiple selections, are always turned into a list.

If this checkbox is selected, all selection fields are displayed as lists, even those with less than 5 selections.

Use fancy backgrounds

Many pages contain background images or background colours. AWeb will display these only if this checkbox is checked. If it is unchecked, AWeb will use only the colours defined in the Browser 2: Colours settings page.

3.9.9 Screen 1: Screen

On this settings page, you can specify on which screen AWeb should open its windows.

Using the chooser, you can make AWeb to open on the default public screen, a named public screen, or let AWeb open its own public screen.

Default public screen

If this is selected, AWeb will open its windows on the default public screen. Usually this is the Workbench screen.

Named public screen

AWeb will open its windows on a public screen that is not necessarily the default. You can enter the screen name you want AWeb to open on.

If the screen doesn't exist when AWeb starts, the default public screen is used instead.

3.9.11 Network 1: General

On this settings page, you can change some general settings regarding the network access.

Image loading

This chooser lets you select if you want AWeb to load images:

All images
: AWeb will start loading every image as soon as it is encountered in a page. This could consume a lot of bandwidth, especially if you are using a slow connection.

Maps only
: This option doesn't load all images, but only clickable maps. This can save a lot of traffic, and still lets you use clickable maps, as these are often essential navigation tools.

Off
: AWeb will never load images automatically. If you want to see an image, you have to click on its icon.

Max. network connections

AWeb is capable of handling an unlimited number of parallel network connections. You might want to limit this number to a reasonable maximum, to avoid overloading your network line.

Note that if the maximum is reached, *image* loading will be queued, but not *document*. A document will always be loaded immediately, so the actual number of connections could be somewhat higher if you are retrieving a page.

The status of all connections can be viewed in the network status window.

Home page

This is the URL of your *home page* as far as AWeb is concerned. It doesn't need to be the same as your home page on the WWW. It is merely the page that will be shown if you select `Navigate / Home document` from the menu, or click the Home button.

You can type in the address, or click the = button to copy the current URL from your first (or only) browser window.

Local index

If the name of a local file requested ends in a slash (/), this name is appended to the file name. This allows setting up your own WWW pages locally without having to change the names.

Configuring a proxy

First, choose the protocol you want to define the proxy for. Use the chooser for this, it has 4 options:

- HTTP
- FTP
- Gopher
- Telnet

Then type in the address of the proxy in the *proxy* field. Make sure it is in one of these two forms: **http://proxy.foo.bar** or **http://proxy.foo.bar:8080**, where the name and port number may differ, of course.

If the address doesn't start with "http://", AWeb will prepend this to the address.

Limited proxy usage

Some proxies can't handle submitting forms with METHOD=POST, or authorized pages correctly. If you have problems with such pages, try selecting this checkbox. It makes AWeb handle these pages directly, without going through the proxy.

3.9.13 Network 3: External programs

On this settings page, you can configure some external programs for network accesses that AWeb doesn't support directly yet. Also, you can configure the commands that AWeb should use to start and stop your *TCP stack*.

Protocol "plug-ins"

Of all possible internet protocols, AWeb understands only HTTP: and Gopher: by itself. There are browsers available (mostly for the PC, but some for the Amiga) that handle other protocols internally, like FTP:, mailto: and news:. However, handling these protocols internally would make the executable bigger, and will never offer the same ease of use as dedicated software. Therefore AWeb offers the possibility to configure external programs that should be started when you follow a hyperlink using one of these protocols.

Use the chooser to select from one of the protocols listed below. Then use the Command and Arguments fields to specify your plug-in command and arguments.

You can configure plug-ins for these protocols:

mailto:
A *mailto:* address is for sending e-mail. If your mail reader supports

Start TCP

This command is used to start your TCP stack automatically.

For example, if you use the AmiTCP/IP package, you would set *command* to: *AmiTCP:bin/startnet* and leave the *arguments* field blank.

Argument parameters are:
first %s = screen name that AWeb is running on, in case your TCP script or program supports opening windows on a public screen.

End TCP

This command is used to stop your TCP connection after your confirmation. There are no argument parameter substitutions.

3.9.14 Program 1: General

On this settings page, you can change some general settings.

Save path

AWeb will always ask where to save a downloaded file, or the HTML source when using the `Project / Save As` menu function. The default save path will be the initial drawer used in the save file requester.

Scroll overlap

If you scroll up or down by a page, there is some overlap. Bacause you might want to have a larger overlapping area when you are using a larger font, you can change the overlap size.

Set this gadget to the desired overlap size, measured in pixels.

Allow Shell commands in links

AWeb offers a powerful facility to excute Shell commands just by clicking a hyperlink or submitting a form.

Although this feature can be very useful, it could also cause severe damage if an undesired command like FORMAT would be executed. Therefore this feature is disabled by default. Select this checkbox to enable it.

Hotlist requester auto close

If this checkbox is selected, the hotlist maintenance window will close automatically if you follow a link in that window. If this checkbox is not selected, the window remains open until you close it yourself.

Temp path

This is the directory where AWeb will create its temporary files. Temporary files are needed for several reasons: retrieved files for an external viewer, downloaded files, source files for inlined images, and swapped documents.

Type the full path name, or click the button located to the right of the gagdet to pop up a standard drawer requester.

Caches

Use these four gadgets to fine-tune the amount of cache AWeb should use.

All sizes should be given in kB.

Minimum free memory

These two gadgets determine the minimum amount of chip and fast memory that AWeb should leave free. Read the low memory description in section 4.6 to understand what these settings mean.

If you specify more memory than can possibly be freed (e.g. a non-zero fast memory limit on a machine with no fast memory), AWeb will continuously try to swap out all documents and images. So be careful what you enter here.

3.9.17 MIME types and external viewers

On this settings page, you can configure the MIME types AWeb should recognize, and the external viewers to use for each MIME type.

About MIME types

MIME (Multipurpose Internet Mail Extensions) is a mechanism for specifying and describing the format of Internet message bodies. It was primarily designed for e-mail, but MIME types are used also to identify the type of data in the HTTP protocol, the most widely used protocol on the World Wide Web.

For a browser like AWeb, the MIME type of a document determines whether the file should be displayed in the browser window, or be processed by some other program.

A MIME type consists of a *type* and a *subtype*. The *type* describes the major class of data, like text or image. The *subtype* is used for a subdivision of the major type into different formats, like GIF or JPEG images.

According to RFC 1521, the following official MIME types are defined:

TEXT/HTML
This is a document in the HTML hypertext format. Virtually all pages on the Web are in this format.

Processing

Files of types TEXT/HTML and TEXT/PLAIN will be shown in the browser window. Files of other types are processed by an external viewer. In spite of the name *viewer*, this is not limited to graphical files. The external "viewer" for an audio file, for example, will play the audio file.

Use the Command and Arguments fields to specify the viewer command to execute for this MIME type. Argument parameters are:**first %s** = file name to "view"**second %s** = screen name that AWeb is running on, in case your external viewer supports opening on a public screen. Use this only if you want it to open on the same screen as AWeb.

If AWeb can't determine the MIME type, or if the MIME type is known but not in the list, or if the MIME type is in the list but there is no external viewer defined, AWeb will pop up a save requester. You can than save the file, and try to process it later.

Example

Suppose you want to see JPEG images using the VT program, and other images using the MultiView program on its own screen. You know that JPEG files can have extensions **jpeg**, **jpg**, or **jfif**, and that GIF files have an extension **gif**. IFF images can be recognized by **iff**, **ilbm**, **ham** or **ham8**. You want AWeb to recognize other image formats you don't know of.

Then you would configure the following MIME types:

IMAGE/GIF gif
: This row specifies that GIF files can be recognized from their .gif extension. You specify no viewer because you want to use the default image viewer, defined in the IMAGE/* row.

IMAGE/JPEG jpeg jpg jfif SYS:Utilities/VT %s
: This row defines the possible extensions .jpeg, .jpg and .jfif for JPEG images. It also specifies that JPEG images should be displayed using the VT program.

IMAGE/X-IFF iff ilbm ham ham8
: This row defines an extension MIME type for IFF images. Note that the subtype starts with **X-** because it is not an official MIME type. This line is important when looking at IFF files on your local computer, as AWeb has no way to identify them as IFF images other than the extensions given here.

IMAGE/* SYS:Utilities/MultiView %s screen
: This row defines what viewer (MultiView) to use for all other images but JPEG. Even files with different subtypes than GIF or JPEG (but main type IMAGE) will be shown using this viewer. There are no extensions defined here, because all extensions are given in the different subtype rows. As an alternative, you could remove the IMAGE/GIF and IMAGE/X-IFF rows, and specify all extensions (gif iff ilbm ham ham8) here.

Chapter 4

Advanced topics

4.1 HTML modes

4.1.1 About HTML

Most of the documents (”pages”) found on the World Wide Web are written in *HTML* (HyperText Markup Language). HTML was originally designed as a standard hard- and software independent way of formatting documents. It is an application of *SGML* (Standard Generalized Markup Language).

The only official standard is HTML 2.0, which contains very limited possibilities. The W3 consortium was developing a new, extended standard, HTML 3.0.

Meanwhile, in the recent boom of Internet and the World Wide Web, some browser manufacturers have introduced several ad-hoc extensions to HTML, of which many didn't fit in the new HTML 3.0 standard. Probably because this would make HTML 3.0 an academic standard without any practical use, the development of HTML 3.0 was abandoned. AWeb supports some of the extensions found in HTML 3.0.

Recently the W3 group proposed a new HTML 3.2 standard, which contains many of the widely used NetScape and Microsoft Internet Explorer specific extensions. AWeb will fully support HTML 3.2 in the near future.

The large browser manufacturers have introduced other tags, that aren't included in the HTML 3.2 standard. AWeb will try to support most of these non-standard extensions.

To make things even more inconvenient, some earlier versions of popular PC browsers didn't stick to the SGML rules. And even recent versions of those browsers still have problems with SGML comments. Because many people design their pages using these browsers, there are many documents on the web that just are bad HTML.

- **<OL START=***n* **TYPE=***otype***>**
  <UL TYPE=*utype***>**
  <LI TYPE=*otype*—*utype* **VALUE=***n***>**
  Enhanced lists. *otype* can be **A**, **a**, **I**, **i** or **1**. *utype* can be **DISC**, **CIRCLE** or **SQUARE**. (strict, tolerant, compatible)

- **<OL CONTINUE SEQNUM=***n***>**
  <UL PLAIN SRC=*url* **DINGBAT=***name***>**
  <LI SKIP=*n* **SRC=***url* **DINGBAT=***name***>**
  List extensions from the HTML 3.0 specification. (tolerant, compatible)

- &*icon.name*
  All proposed WWW icon entities (dingbats). Look at the on-disk overview for all supported icons. (tolerant, compatible)

- **<FRAME SRC=***url* **NAME=***name***>**
  Very limited frame support. AWeb does not yet support frames, but it recognizes the <FRAME> tag and shows a hyperlink for each document. You will still see those annoying "your browser doesn't support frames, download NetScape here" messages, but at least you can access the information. (tolerant, compatible)

4.1.4 Compatible mode

As mentioned above, some pages contain bad HTML. When you view such a page, it can look distorted. You can expect large parts of the page missing, links to URLs that seem to contain HTML tags, and other strange things.

If you encounter such problems, try using the *compatible* HTML mode of AWeb. **Warning:** Using compatible HTML mode, documents containing *valid* HTML might look distorted in turn.

In compatible mode, AWeb exposes the following deviations from the SGML standard:

- quoted attribute values are terminated by any occurrence of ">"
- quoted attributes that contain URLs, like HREF, SRC and ACTION, are terminated by whitespace
- comments are terminated by any occurrence of "–>"

4.2 ARexx interface

4.2.1 ARexx port names

Every AWeb window has its own ARexx port. This port is named **AWeb.#**, where # is a unique number.

The About requester shows the actual name of the ARexx port for the window from which you selected the About menu item.

SCREEN - Retrieve the name of the screen that AWeb uses to open its windows on.

ACTIVEPORT - Retrieve the name of the ARexx port associated with the most recently activated AWeb window. Useful in ARexx macros started via a shell script or shell command to find the active window.

The information is returned in the reserved variable RESULT, unless the *VAR* argument is used to specify the variable name.

CACHE ITEM/A,FROM/A

Get information about AWeb's cache. The *ITEM* argument determines the information to return, and the *FROM* argument determines the object to get information for. *ITEM* can be one of the following:

URL - Get the URL for a file in cache. *FROM* must be the name of a cache temporary file, with or without path.

NAME - Get the file name for a file in cache. *FROM* must be the name of a cache temporary file, with or without path. The file name returned is the last part of the URL, after the last slash.

TEMP - Get the temporary file name for a URL. *FROM* must be a URL of a document or image in cache. Note that no URL parsing is done, so the URLs must match exactly. Also note that many documents won't have a corresponding cache file.

The information is returned in the reserved variable RESULT.

SAVEAS NAME,APPEND/S

Save the HTML source of the document.

If *NAME* is given, the source is saved under this name. If the *APPEND* switch is set, the source is appended to the file, otherwise the file will be overwritten.

If no *NAME* is given, a save requester will pop up.

ACTIVATEWINDOW

Make this window the active window.

WINDOWTOFRONT

Move this window in front of all other windows on the screen.

WINDOWTOBACK

Move this window to the back of all other windows on the screen.

SCREENTOFRONT

Move the screen that AWeb is using in front of all other screens.

SCREENTOBACK

Move the screen that AWeb is using to the back of all other screens.

CLOSE FORCE/S

Close this window. The FORCE switch suppresses the "Are you sure" requester if this was the last window.

QUIT FORCE/S

Quit AWeb. The FORCE switch suppresses the "Are you sure" requester.

4.3.2 ARexx macros

Starting ARexx macros from your page works in a similar way. Just add a normal hyperlink that points to a URL of the form x-aweb:rexx/*your_ARexx_macro*. If the user clicks on the hyperlink, *your_ARexx_macro* is started with the ARexx port for this window as the default command port.

4.3.3 Parameters

You can use a HTML *form* or a *clickable map* to pass parameters to your DOS command or ARexx macro.

Forms

Supply a ACTION="x-aweb:command/*your_command*" attribute in your <FORM> tag to execute the command if the user submits the form. Similarly, you can include a ACTION="x-aweb:rexx/*your_macro*" attribute to start the ARexx macro.

Form parameters are converted to Amiga DOS style parameters: the field name will be used as the argument name, and the field value will be used as argment value. The value will be quoted, with the *escape*, *newline* and *quote* characters in the value escaped as required by Amiga DOS.

Note: *switch arguments* (/S) cannot be passed in this way. You could use a script instead, like the example below.

Clickable maps

When using a clickable map, the x and y coordinates of the mouse pointer within the image are passed to the command as parameters without keyword.

ARexx arguments

Parameters for ARexx macros are passed in the same format as for DOS scripts. The argument string will contain the name, an equal sign, and a quoted value for each form parameter. Have a look at the second example below for one possible way of parsing this.

Load the result back into AWeb

If your script or macro has created a HTML document (or just a plain text file), you can automatically load this file back into AWeb. Use the ARexxOPEN command for this purpose. If you re-use the name of your file for different responses, be sure to add the RELOAD switch to prevent AWeb from showing the previous (cached) document again.

in pages that will only be viewed by AWeb, because other browsers will not recognize these URLs.

A good place for extension URLs would be your hotlist. You can, for instance, access the AMosaic or IBrowse hotlist from within your hotlist, or even configure one of these as your home page within AWeb.

4.5.1 Hotlists

AWeb uses extension URLs to identify its hotlist, and other browser's hotlists.

x-aweb:hotlist
: Identifies AWeb's own hotlist.

x-aweb:amhotlist.rexx
: Identifies the ARexx based hotlist of AMosaic version 1.2. It uses the file ENV:mosaic/hotlist.html.

x-aweb:amhotlist.20
: Identifies the hierarchical horlist of AMosaic 2.0 prerelease. It uses the file ENV:mosaic/.mosaic-hotlist-default.

x-aweb:ibhotlist/*path*
: Identifies the hotlist of IBrowse. Because this hotlist doesn't have a fixed location, you must specify the full path and file name in the URL.

4.5.2 Shell commands and ARexx macros

Two extension URLs exist to start shell commands or ARexx macros.

x-aweb:command/*shell_command*
: Forms the interface to start Shell commands.

x-aweb:rexx/*ARexx_macro*
: Forms the interface to start ARexx macros.

4.6 How the cache works

AWeb uses a special caching system, partially in memory and partially on disk, to reduce the number of network accesses.

Note that the AWeb cache is a *cache*, not a *proxy* system. This means that the cache is cleared when AWeb finishes. Also, the cache doesn't take the HTTP expiration date into account. This might change in later versions of AWeb.

4.6.1 Documents

Documents that can be displayed by AWeb are initially loaded in memory. When the document cache is full, the least recently displayed document is swapped

- a large *image disk cache*.

Of course, you can use other settings if you like. You can fine-tune the memory and disk usage by changing these settings.

If you use AWeb on a native screen, you will want to keep some chip memory free as a work space for the datatypes. On CyberGraphics systems images will go in fast memory, so you will want to leave some fast memory free on those systems. The exact amount depends on the number of colours on the CyberGfx screen, and of course on the total amount of memory.

- *The colour requester from the Screen 2: palette settings page messes things up on a CyberGraphics screen.*
  This is caused by the palette-orientated design of the Amiga OS. This works fine on Amiga chipset screens, but isn't really suited for graphics cards.
- *Closing the AWeb public screen while there is a visitor window open crashes if the screen is a CyberGraphics screen.*
  This is a bug in CyberGraphics V 2.15 and lower. The same thing happens on other CyberGraphics screens.
- *If the window is not active, a click in the scroller container (outside the knob) moves the scroller but doesn't scroll the window contents.*
  This is a bug in Intuition. The same thing happens sometimes with MultiView.
- *I use MagicMenu, and every now and then my system hangs if I open a menu.*
  This is a known problem in MagicMenu, caused by the fact that MagicMenu is not 100% compatible with the way Intuition handles menus.
- *I don't use MagicMenu, but my system hangs if I click on the wide part of the chooser gadget in the settings window.* Be sure to use the version of chooser.gadget included in this archive. Also, some animated mouse pointer commodities are known to cause this hang.

5.2 Known bugs

5.2.1 AWeb bugs

Although AWeb is thoroughly tested, it is very likely that there are some bugs left. At the time of release there were no known bugs. For the latest information, check the AWeb Support Page at
http://www.networkx.com/amitrix/index.html

5.3 Things to do

This page lists all suggested enhancements at the time of release. The online to-do list lists additional suggestions, received after the release. Please check both this page and the online page before sending me any suggestions.

5.3.1 Future releases

AWeb is still being developed. Future releases of AWeb will support:

- More HTML-3.2 commands like tables and image alignment
- Client pull

Browsing

- Implement global history. Also: possibility to choose an URL from the global history.
- Global window history, editing, rearrange, copy to hotlist.
- Configurable 'direct buttons'.
- Document headers feature. Either: 1) append <hr> and <dl> with headers to every document; 2) open window on user request with headers in it; 3) create button that links to x-aweb doc that has them, and maybe other interesting things as well.
- Let ctrl-click load the page in a new window.
- Use external viewer with pipe
- Option: View inline image with external viewer
- Direct support for .z and .gz files: uncompress before processing or starting external viewer.
- Different font settings for <address>, <blockquote> and <cite>
- Let dragging with the mouse scroll the page.
- Option: render images vertically compressed on 1:2 pixel screens (like 640x200).

Program

- Do 'Save as formatted text'
- Include the Print function
- Include the Find in current function
- View HTML using textview.gadget (editable?)
- Iconify option (transfers continue while iconified).
- Load settings, Save settings as...
- Check if saved/downloaded file fits on disk
- Screen preferences: alternative screen name if first one doesn't exist.
- Screen preferences: wildcards in screen name like DOPUS.*
- Ask save filename before download, instead of saving to temp file first.
- Support for HAM Workbench
- Support environment variables EDITOR and HTTP_PROXY etc.

- Option: new/visited links underline type. Or a user selectable image for underlining.
- Flag: proxy status on/off in window titlebar.
- Special colour for links to pages in cache.
- Enhanced list item bullets (3d)

Program

- Revert to previous screen mode if selected screen mode doesn't open, instead of falling back to Workbench.
- Save positions for 2nd and later windows.
- Take display clip into account when opening full-sized window.
- If settings changed but not saved, show warning requester before quit.
- If named screen does not exist, show error requester (and open on default public screen).
- On a snapshot, remember if the network window was open and re-open it after a restart.
- Save the zoomed window dimensions too.
- Preload grayscale palette for 16 or less colours.
- Screen preferences: requesters with present public screens to choose from.
- Open information window if external editor starts.
- Window with cache or memory indicator.

Miscellaneous

- Don't refresh border of listview and textarea form fields with every character typed
- Do proper URL encoding; decode on file://localhost
- Proxy prefs: separate string gadget for port number.

I also wish to thank my beta testers, without them AWeb would never have become the great program it is.

- Osma Ahvenlampi ("Tau")
- Jeroen Oudejans
- Thomas Tavoly ("aTmosh")
- Vincent Groenewold ("supernov")
- Donovan Janus
- Mike Meyer
- Paul Kolenbrander
- Donald Voogd
- Kristian Phillips
- Dale Currie
- Christopher Aldi
- Josef Faulkner

AWeb was written by Yvon Rozijn.