Xinet Asset Management:
Your Work. Your Workflow.

XINET

GUIDE TO DEVELOPMENT APIs

Version 16.04

Contents

Chapter 1

Introduction

Contents of this Guide

In this *Xinet Guide to Development APIs*, Xinet collected the various resources which are available for custom programming of FullPress® and the WebNative® family of software products.

As Xinet products are engineered to make development and customization as easy as possible, the software contains a built-in set of tools which allow programmers to extend the products' functionality to suit the special needs of individual sites or integrate Xinet products within other proprietary systems.

The *Xinet Guide to Development APIs* includes information about the following resources:

- FullPress file format
- FullPress file system access via the Xinet *ks-utilities*
- Customizing the FullPress GUI
- WebNative customization options
  - *Basket* plug-ins (CGIs)
  - *Button* APIs
  - *Styles* interface customizations
  - *Uploader* customization
- WebNative Portal APIs

Though this guide is quite useful on its own, Xinet also recommends that developers briefly review the features available in each product and its accompanying *Administration Guide* to gain familiarity with the software's functionality.

Other reference sources

Each Xinet *Administration Guide* explains the products' functionality and configuration in great detail. The *FullPress Administration Guide* and the *WebNative and WebNative Venture Administration Guide* provide details about administrative settings and how to use them effectively.

Sites experimenting with the "look and feel" of the Xinet web interfaces will find instructions in Chapter 5 in the *WebNative and WebNative Venture Administration Guide*, which discusses

ways to uniquely change how WebNative displays information in web-browsers via HTML and Javascript programming. WebNative Portal provides even more opportunities for customizing product functionality and the user experience, as its PHP infrastructure offers programmers a much faster development and integration path. In the *WebNative Portal Administration Guide*, current features are explained as well as tools which can be used to extend the data-display options within templates.

To help developers and programmers understand the core functionality of Xinet products, Xinet offers regularly-scheduled training classes in product administration at basic and advanced levels. Xinet also offers custom programming seminars for WebNative *Styles* interfaces as well as instruction in template creation and adding scripting to WebNative Portal. For more information on course content and scheduling, please refer to the *Training a*rea of the Xinet web site, *www.xinet.com*.

The Xinet Developer Program

While all Xinet customers have access to product development tools as part of their standard software licenses, if customers do not wish to use their production server as a testing and development platform, Xinet offers non-production *Software Development Kit* licenses to members of its Developer Program.

The Xinet Developer Program is designed to assist the following people:

- Customers who are designing and testing interfaces, plug-ins, or other in-house programming customizations and wish to do this on a separate machine rather than on their main “live” Xinet production server.
- Software developers and consultants who are writing plug-ins or other application tools which will interoperate with Xinet servers.
- Vendors and manufacturers who wish to use Xinet software for integration or testing with their own products and solutions.

For more details about the Xinet Developer Program, please refer to the Xinet Web site, *www.xinet.com*. You'll find an on-line application to join the program, as well as a description of our *Software Development Kit* license and other membership benefits.

Conventions used in this manual

We use several typographical conventions in this manual to help readers distinguish what must be typed as is and what is simply a parameter to be substituted. In particular, readers should type text set in a typewriter-like font, `like this`, *literally.* For example:

```
# more README or c:\> type README
```

Readers should enter this command exactly as shown above (minus the shell prompt on systems, #, or the `c:\>` prompt on Windows systems). Readers should substitute their own applicable text for text set in slanted typewriter type, *`like this`*. For example, when one sees

```
% ls directory_name or c:\> dir directory_name
```

one might type:

```
% ls doc or c:\> dir doc
```

On UNIX systems

The two examples given above illustrate another convention: when a command must be issued by the superuser (root), you will see the prompt `#`; when a command may be issued by a normal user, you will see the prompt `%`. As in most documentation for UNIX software, text references to UNIX functions have the form *more*(1). The word in italic type is the name of the function; the parenthesized number is the section of the UNIX Programmer's Manual containing the function's manual page.

On Windows systems

When the manual discusses locations for files and programs, it uses default installation pathnames. If you install files and programs in other locations, the pathnames will not be the same as in this manual.

Where to look for technical information

Xinet technical information spans several volumes, with each software distribution having an appropriate guide. The guides are available for downloading in the *Maintenance* section of the Xinet Web site, *www.xinet.com.* The Web site also provides links for ordering printed copies.

To help you easily find what you need, the guides are extensively indexed and cross-referenced, and whenever possible, the index entries and cross references provide live links to the appropriate pages within the PDF.

Xinet technical documentation includes:

- The *WebNative Suite Administration Guide*

  This guide provides basic information about installation, licensing and WebNative Suite administration. It also provides information to consider before installation, such as optimal file-system organization and security issues. Chapters include details about administrative settings and basic ways to customize the look and feel of Web sites.

  The guide includes information about setting up everyday management of the WebNative Suite database, including ways to customize data fields and control their use by users and groups. It also contains information about automating production activities based on changes in the database and information about importing data.

  Within this guide, you'll also find information about file-sharing and print spooling optimizations for production workflows and details about the many options available when running WebNative Suite software in a production environment.

- The *Xinet Client Guide*

  This guide provides information to help end-users who are WebNative Suite server clients—with either direct access to the server or access through a Web browser. It explains the growing list of Xinet utilities that help users get the most out of a WebNative Suite server. It also includes tips for working within that environment when using Macintosh or Windows applications from Quark, Adobe, etc., as well as features available when sites make use of Video for WebNative Suite, WebNative Archive, and WebNative Portal.

- *Xinet PC Connectivity Guide*

  This guide provides information about installing, setting up and maintaining the Xinet modified Samba software which allows Microsoft Windows clients to interact with WebNative Suite.

- The *Xinet Guide to Development APIs*

  Not all sites where Xinet software is installed will need this volume. It presents information about WebNative Suite that will interest programmers customizing the product beyond ways made available through the GUIs which ship with it. Turn here for command-line customizing and diagnostics and more significant changes to WebNative Suite interfaces.

- The *WebNative Portal Guide*

  This guide explains installation, licensing and administration for WebNative Portal sites. It also provides an overview of using WebNative Portal to customize the look and feel of user sites and gives examples of building new functionality to even further extend the product.

- The *Xinet Guide to WebNative Portal Data Interchange*

  This guide, intended for developers who want to customize and extend WebNative Portal beyond those options which ship with the product, provides information about how to communicate with a WebNative Suite server using the *portalDI* CGI. The *portalDI* CGI is a query-based application responsible for such things as constraining information displays, presenting browse and search results, and managing shopping basket and metadata interactions.

- The *Video for Xinet WebNative Suite Administration Guide*

  This guide provides information about installing and configuring the WebNative Suite Video module, including information about its various options, settings, metadata capabilities and interactions with the WebNative Suite database.

- The *Xinet WebNative Archive Administration Guide*

  This guide provides details about using WebNative Archive to allow WebNative users to communicate over the Web with Symantic Backup Exec™ or SGL FlashNet archiving software. The guide explains installation and administration and provides guidelines for user interaction over the Web.

- On-line manual pages

  UNIX-style on-line *man*(1) pages are included with UNIX distributions. While not intended for the lay reader, they provide handy reference for more technically-advanced administrators and for programmers who want to work with Xinet programs from the command line. The *man*(1) pages are also included as part of each PDF manual.

- WebNative Suite on-line help

  Context-sensitive Help buttons in the WebNative Suite administration interfaces provide information about options in GUIs. They open appropriate pages within the *WebNative Suite Administration Guide*.

- *Xinet TechNotes*

  Xinet TechNotes provide information about technical issues not included in manuals. Like the *Xinet Guide to Development APIs* and *man*(1) pages, the notes are often aimed at those extending Xinet products beyond functionality offered in Xinet GUIs. They also provide information on issues that change quickly or that are of a transient nature.

Chapter 2

WebNative Suite development tools

The WebNative Suite file format

Summary

This section describes the file format that WebNative® Suite uses to store Macintosh files on a UNIX file system. It does not apply to the Microsoft Windows platform.

This information provided here as accurate as possible; however the design and the code it describes are subject to change at any time.

Introduction

The file format that Xinet uses for UNIX file systems was chosen with four main goals:

- A clean interface from the UNIX side
- Portability and extensibility
- Performance
- Backwards compatibility

The desire to allow UNIX users and Macintosh users to easily share data files mandated a double file format with the data fork easily accessible. To keep the UNIX interface clean, we choose to hide the resource forks in a hidden directory, *.HSResource*. This allows easy access for UNIX utilities to the files data fork, and allows quick and easy access to the resource fork from the AFP server. Since there were no standard file formats at the time (AppleSingle and AppleDouble did not exist yet), and since it was clearly a performance loss to store file attribute information in each file, Xinet chose to put all of the attribute information for a single directory in one Desktop file, called *.HSancillary.* This format also makes it easy to determine if the directory contains WebNative-Suite-format files. (Simply check if *.HSancillary* exists.) *resfilename()*will map a data file name to the corresponding resource fork file name, hiding the details from the applications programmer.

Files created on the server (rather than by Macintosh applications) do not normally carry the type of information found in DeskTop files. Whenever the server encounters a file which did not originate on one of the clients, it tries to determine the type of the file (from a list of rules in the *filetype* database), assigns it a file type and creator.

A side effect of the way WebNative Suite creates *.HSancillary* files is that if users actually remove an *.HSancillary* file, they remove all "file type" information about the files. The server will then rebuild the*.HSancillary* file, doing its best to re-create type and creator information.With the introduction of AFP protocol version 3, Xinet software began supporting longer file names encoded as UTF-8. Previous versions of Xinet software had strict limitations on file names stored on the server.

Contents of *.HSancillary*: struct ai

The *.HSancillary* file contains one or more structures of the type *ai* (*ai* stands for "ancillary info"). The following shows the definition of *struct ai.* The current version number, which is contained in field *ai_version*, is *8139.*

```
                                                /* Decimal Offset - Description */
struct ai {
      struct accrts    ai_inhAR;                /* 0 - rights inhibit bits */
      struct fInfo     ai_finfo;                /* 8 - finder info */
      long             ai_createDate;           /* 40 - creation date */
      long             ai_backupDate;           /* 44 - backup date */
      unsigned int     ai_version;              /* 48 - for future reference *
      /unsigned short  ai_attr;                 /* 52 - attributes */
      unsigned char    ai_comment[MAXCMTSIZ + 1]; /* 54 - comment in Pascal format */
      char             ai_lname[CNNBUFLEN];     /* 254 - Mac AFP Long Name (8-bit) */
      char             ai_sname[SNBUFLEN];      /* 286 - AFP Short Name */
      u_char           ai_xmaphigh; /* 299 - .HSxmap index (upper 8 of 24 bits) */
};
```

The following code shows predefined constants:

```
#define CNNBUFLEN         32                    /* length of AFP Long Name buffer */
#define SNBUFLEN          13                    /* length of "short name" buffer */
#define AI_VERSION1       8139                  /* version of struct ai */
#define MAXCMTSIZ         199                   /* size of a comment string */
#define AI_SIZE           300                   /* size of struct ai for sun 4 */
#define FINFOSIZE         32                    /* size of Finder Information */
```

Code for *struct accrts* follows:

```
struct accrts {
unsigned long ar1;         /* Access rights (Mac format) */
u_char ar_ancest;          /* Ancestor rights, from parent dirs */
u_char textstatus;         /* whether TEXT files are CR/LF-flipped *
/u_int16 xmaplow;          /* low 16 bits of .HSxmap index */
};
```

With the advent of distributed file systems it is possible that the *.HSancillary* file is actually on an NFS file system shared between different processor architectures. Therefore, one should make sure that all programs dealing with the file use *network byte ordering* and SUNSparc architecture structure-padding conventions (see *htons*(3)). Correct field offsets are noted in Figure 2-1. The current size of the structure is *AI_SIZE*. Application programmers can use `opendt()` to open the DeskTop file, and the various `*ai()` routines to read and write DeskTop entries.

The server reuses entries in this file rather than deleting them when files are deleted. An entry describes an actual file when `ai_lname` contains the Macintosh file name, *i.e*, `ai_lname[0] !=0`. Entries for which `ai_lname[0] == 0` should be reused before new entries are appended to the DeskTop file. (This is done automatically if you use `findaientry()`).

The structure *accrts* contains the access information. Its size is 8 bytes. Figure 2-3 gives a simplified description.

The structure `fInfo`

The structure `fInfo` shown below contains the Finder information. The structure also contains the type and creator fields `fi_type` and `fi_creator`, respectively.

```
struct fInfo {
           unsigned long    fi_type;/* 0 - four chars (OSType) */
           unsigned long    fi_creator;/* 4 - four chars (OSType) */
           short            fi_flags;/* 8 */
           short            fi_location[2];/* 10 */
           short            fi_fldr;/* 14 - file's window */
           unsigned char    fi_dummy[16];/* 16 - used in new Mac OS's to store FXInfo. */
};
```

The `fi_type` and `fi_creator` are the structures one changes to set the Macintosh type and creator information.

File naming conventions

Macintosh files may contain characters not normally acceptable for UNIX file names. WebNative Suite software handles this on the UNIX side by substituting a three-character hexadecimal representation of the Macintosh character; for example, the Macintosh file name *jam/jelly* becomes *jam:2Fjelly* when moved to the UNIX server. This name mapping will be transparent to WebNative Suite Macintosh clients — in other words, the file will have its original Macintosh name when opened by a Macintosh application.

WebNative Suite support for AFP protocol version 3 includes longer file names encoded as UTF-8. Previous versions of Xinet software always converted non-ASCII characters in the *AFP Long Name* (limited to 31characters) to their colon-escaped equivalents. (Mainly because early UNIX file systems only allowed 7-bit ASCII in file names.) Since modern UNIX file systems support a full 8-bit file name space, new file names are stored as UTF-8 in the UNIX file system. They are supported without changing the layout of the *struct ai* data (described

above), but by adding a new file, *.HSxmap,* that maps these non-AFP-Long-Name-conforming names to their *AI* entries.

If an *AI* entry's `ai_xmaphigh` and/or `ai_inhAR.xmaplow` fields are non-zero, the entry is for an AFP 3 file name (UTF-8, 255 characters maximum), and the actual file name is stored as a Pascal string in *.HSxmap*, at byte offset `((ai_xmaphigh << 16) + ai_inhAR.xmaplow) * 64`. Otherwise, the `ai_lname` can be converted to the actual file name by replacing any byte values less than 32, greater than 126, or equal to 47 (the "slash") with their upper-case HEX-value preceded by a colon (illegal in the AFP file name space). Note that even UTF-8 file names must have control-characters and "slash" converted to their colon-escaped HEX equivalents.

If you use `getdtent()` and `setdtent()` calls, `oropendt()` and its companions (`readai`, `writeai`, `makeai`, `findaientry`, `closedt`), all AFP name mapping is handled inside those routines. They all take file system paths as arguments and convert names as necessary. Note that it is perfectly legal to mix old-style ASCII-HEX-encoded names and new-style UTF-8 names in the same directory.

WebNative Suite file manipulation utilities

Xinet provides a number of command-line tools for working with WebNative-Suite-formatted files in the file system. Developers who wish to manipulate these files should do so using the utilities described in this section.

When a user modifies files using a mounted volume on Mac OS 9 or Mac OS X, or a Xinet Samba volume mounted on a PC client, the daemon *ksd*(1M) makes changes on both forks of the file. If you wish to move, copy or delete files in a WebNative Suite volume without using a mount, Xinet file manipulation utilities allow you to preserve the resource/data fork relationship. The utilities are located in:

UNIX:	*/usr/etc/appletalk*
Windows:	*C: \Program Files\Xinet\FullPress*

They include:

- *kats*(1) Read ancillary information about files and directories. Use the *-v* flag for verbose output; for example `kats -v file.tif`.
- *ksmv* Move files or directories. Use the *-p* flag to preserve permissions; for example, `ksmv -p /path/1 /path/2`.
- *kscp* Copy files or directories. Use the *-p* flag to preserve permissions; for example `kscp -p file1.tif file2.tif`.
- *ksrm* Remove files or directories. Use the *-r* flag to allow recursive deletion; for example `ksrm -r /path/1`.
- *kunarc*(1) Convert between various Macintosh archival formats and the WebNative Suite file format. The supported archive formats include AppleSingle, AppleDouble, BinHex, CAP-AUFS 3.0, Helios Ethershare, MacBinary, TOPS (version 2 or later) and IPT uShare 4.1
- *dtrebuild*(1) Maintain databases used by the AppleShare server, a la "Rebuild DeskTop" on the Macintosh. Also use it to (re)create FPO/Web images.

In addition to preserving file resources, the WebNative Suite utilities also ensure that events are sent to WebNative Venture to update the state of each file in the database.

Customizing print queue options in the WebNative Suite GUI

Xinet's FullPress 11.0 introduced print queues that generated PDF, TIFF, and TIFF/IT output. GUI-configureable PDF queues followed in version 12. This section explains simpler ways to change the parameters for these and other WebNative Suite queues that use the *PostScript Interpreter* to output files in other formats. Complete details of supported parameters and known problems can be found in *Xinet TechNote 122*.

The program responsible for making PDFs and other image format files is called *rip.* The *rip* program is called by the *runrip*(1) program, which is in turn called by the *lp*(1) system when a job is sent to a Xinet queue. Ultimately, the *rip* process needs to have configuration parameters set. You can do this in several ways:

- Edit the *runrip*(1) command used by the queue.
- Make a *getrip_conf.local* file that will affect all new queues
- Make an *OPTIONS.PS* file for a queue or spooler.
- Copy and edit the *XINET.PPD* to allow parameters to be selected at print time.
- Finally, a few parameters are set by the application when you print from the Mac.

Background

The *runrip*(1) program accepts arguments that can be passed on to the *rip* program. See the man page of *runrip*(1)and *Xinet TechNote 122* for more details. Almost all arguments to *runrip*(1) (save three specific ones) are simply passed on to the *rip* program, so changing the options for *runrip*(1) really changes the options for the *rip*.

When a print queue is created, an entry is made in the */etc/printcap* file on UNIX systems and *C:\Program Files\Xinet\FullPress\Admin\printcap.wri* on Windows. The *ex* field for a PDF, TIFF, or TIFF/IT queue will have the command `runrip` in it, which is executed by *lp*(1) and sets up the execution environment for *rip*.

Please note that /etc/printcap *entries should be typed in a single line, without "returns."*

A sample line from */etc/printcap* for a PDF (print) queue:

```
:ex=/usr/etc/appletalk/runrip -q -dNOPAUSE -dBATCH -sDEVICE=pdfwrite
-dCompatibilityLevel=1.3 -c .setpdfwrite -f
/var/adm/appletalk/psfiles/pdfscreen -_:
```

A sample line from */etc/printcap* for a TIFF/IT queue:

```
:ex=/usr/etc/appletalk/runrip -t /usr/etc/riptmp -s -q -dNOPAUSE
-dBATCH -sDEVICE=tiffitp1 -dMaxBandStorageMB=1 -dMaxBandMB=40
-dCTResolution=304.8 -r18 28.8 -_:
```

The arguments to *runrip*(1) influence the creation of the output files. The meaning of each parameter is described in detail in *Xinet TechNote 122* under the topic "General Operation and Interaction." There is a separate subsection for TIFF/IT file parameters.

The default settings for *runrip*(1) for each type of queue is defined in the file */var/adm/appletalk/getrip_conf* (UNIX) and *C:\Program Files\Xinet\FullPress\Admin\getrip_conf* (Windows). This file is not meant to be edited, since it will be updated in future releases of WebNative Suite.

The other way to influence how output files are produced is to add parameters to the PostScript stream, either by a "slug" or at the source. One "slug" method is a file on the server added to the *runrip*(1) command for a specific queue, which gets read by the *rip* before the main body of the PostScript. Parameters for distilling must be set via the *setdistillerparams* operator. Parameters for making other format output must be set via the *setpagedevice* operator.

Xinet adds its own slug to PDF queues by default. Every PDF queue made uses the default set of parameters for PDF preview queues listed in a file: */var/adm/appletalk/psfiles/pdfscreen* (UNIX) and *C:\Program Files\Xinet\FullPress\Admin\pfscreen* (Windows). For PDF print queues it uses */var/adm/appletalk/psfiles/pdfprint* (UNIX) and *C:\Program Files\Xinet\FullPress\Admin\pdfscreen* (Windows) These files are not meant to be edited. They use standard Adobe-defined parameters for distilling. There is no default file for parameters for TIFF and TIFF/IT queues. All the options are in the runrip command line.

Editing arguments to *runrip*(1)

The easiest way to change the parameters for a queue is by altering the *runrip*(1) command line in the WebNative Suite GUI. The *runrip*(1) command is listed in *Output Print Queues > Configuration > File/Custom* and can be edited there. That changes the *ex* entry in the *printcap* file.

Use the above method if you want to change parameters for a single, existing queue.

Editing the *getrip_conf.local* file

The default settings for *runrip*(1) for each type of queue is defined in the file */var/adm/appletalk/getrip_conf*. This file is not meant to be edited.

You can make your own entries for *getrip_conf* in a file called *getrip_conf.local* and add your own default settings or even your own output format for new queues, as well as override existing queue types in *getrip_conf*. This will alter the *runrip*(1) options on all future print queues, not just a particular queue. Existing queues will not be changed.

Use the above method if you know all future queues will need configuration options other than the defaults selected by the *getrip_conf* file.

Making an *OPTIONS.PS* file

An *OPTIONS.PS* file can be used to pass on a PostScript slug to a queue or for a particular spooler to a queue. Look at the section "Adding PostScript slugs to Print Jobs at the Server" inside the *WebNative Suite Administration Guide* for details about using an *OPTIONS.PS* file.

Parameters can be placed in an *OPTIONS.PS* file and they will be inserted at the start of the PostScript stream.

Here is an example of setting the resolution for the CT and LW of a TIFF/IT queue. The entire contents of the *OPTIONS.PS* include:

```
<< /CTResolution 304.8 /HWResolution [1828.8 1828.8] >> setpagedevice
```

All TIFF/IT options listed in *Xinet TechNote 122* can be part of the *OPTIONS.PS* file. If you add an option here, it's a good idea to remove the command line equivalent from the *runrip*(1) entry for the queue. If the same parameter is specified in both the *OPTIONS.PS* file and the *runrip*(1) command line, the slug in the file will override the *runrip*(1) options. The only exceptions are the *-r* (resolution) and *-g* (height and width) options, where the opposite is true.

PDFs parameters are set like this:

```
<< /GrayImageResolution 300 >> setdistillerparams
```

The settings within an Acrobat Distiller can be "imported" to a Xinet PDF queue by copying them to an *OPTIONS.PS* file for a print queue and/or spooler.

Not every parameter will work with the Xinet interpreter; exceptions are detailed in *Xinet TechNote 122* in the section "Making PDFs: not your mother's Distiller."

The *OPTIONS.PS* file is read after the *pdfscreen* and *pdfprint* files So if a setting is defined in both the *pdfscreen/print* files and the *OPTIONS.PS* file, the parameter in the *OPTIONS.PS* file is the one that will take precedence.

Use the above method if:

- You want to "import" the parameters used in Adobe's Distiller. Just put them all in the file as part of *setdistillerparams*
- You want different options per spooler, but you also want just one print queue.

For more details on importing Acrobat Distiller options, see *Xinet TechNote 130*

Editing the *XINET.PPD*

To make this work, the *XINET.PPD* needs to have more *OpenUI* information added to it. This is a standard feature of PPDs and part of the PostScript language. Essentially, different selections made in the client's print dialog box will add different PostScript slugs to the PostScript stream. It's the same concept as putting slugs in the *OPTIONS.PS* file.

Here is an example. It adds a menu with options for making TIFF/ITs at different resolutions.

```
*% set the resolution of TIFF/IT files:
*OpenUI
*Resolution: PickOne
*OrderDependency: 30.0 AnySetup
*Resolution
*DefaultResolution: CTResolution304.8
```

```
*Resolution CTResolution304.8/CT Resolution 304.8: " <</CTResolution
304.8 /HWResolution [1828.8 1828.8]>> setpagedevice"
*Resolution CTResolution152.4/CT Resolution 152.4: " <</CTResolution
152.4 /HWResolution [914.4 914.4]>> setpagedevice"
*CloseUI: *Resolution
```

This example uses a menu bar, but you could also use a radio button. Most PPDs use *OpenUI*, so it's easy to find examples to follow.

The *XINET.PPD* will be overwritten with each upgrade of WebNative Suite, so you want to make a copy of it and edit that copy. To add that PPD to a specific already-existing queue, move the new PPD to */var/spool/*PRINTERNAME*/PRINTER.PPD* (UNIX) or *C:\Program Files\Xinet\FullPress\Admin\spool\PRINTERNAME\PRINTER.PPD* (Windows) replacing the current *PRINTER.PPD* in that directory. To make the new PPD the default selection when making new PPDs, make a *getrip_conf.local* file (see section above for details) and add a reference to the new PPD. The last field of the line for each queue refers to the default PPD.

Here are the basic steps to follow for editing your PPD:

1. Make a copy of the *XINET.PPD*, saving the old one.
2. Edit the PPD to add the *OpenUI* functions you want.
3. Make a new print queue and spooler.
4. Edit the *runrip* line for the queue so that any options you've added in the PPD are *not* in the *runrip* line. So if you've added a way to set the resolution of TIFF/ITs in the PPD, remove the *-dCTResolution* and *-r* flags to *runrip* for this queue.
5. Distribute the new PPD to the Macs and make a desktop spooler.
6. When you print from QuarkXPress, the new menu bars will be the *Printer* set up in *Printer Specific Options*.

If you *don't* want to make a new queue, you can keep the old one and just change the PPD it uses. The PPD used is in */var/spool/PRINTERNAME/PRINTER.PPD* (UNIX) and *C:\Program Files\Xinet\FullPress\Admin\spool\PRINTERNAME\PRINTER.PPD* (Windows). Edit the PPD in */var/adm/appletalk/ppds* (UNIX) or *C:\Program Files\Xinet\FullPress\Admin\ppds* (Windows), then copy it (don't move it) to the spool directory of your print queue. Also distribute the new PPD to Macintosh users. Edit the *runrip* line, too, if necessary.

Use the above method if you want Macintosh users to be able to select parameters when they print from QuarkXPress or any other layout application. Using this method, you can have one print queue, one spooler, and let people decide what options they want when they print.

When parameters are set by the layout program

Finally, some parameters, such as the type of screening, are beyond your control; i.e., they are set by QuarkXPress or InDesign when the PostScript is made. In particular, this affects making MonoTIFFs.

Advanced customization of print queue options

This section describes advanced ways to customize print queue options found in the GUI, and is appropriate for system integrators or more advanced administrators of large operations.

"Adding PostScript slugs to Print Jobs at the Server" in the *WebNative Suite Administration Guide* describes how to use a PostScript slug file called *OPTIONS.PS* for custom print queues.Typically, an administrator creates this slug and installs it "by hand." More advanced administrators may want to employ an optional PPD-like file to maintain user-configureable options that are then written to the *OPTIONS.PS* file. The PDF/X-1a and Configureable PDF output queues already use this mechanism.

The *printcap* option

To enable this feature, add an *op* key to the *printcap* entry for a print queue, or include it in a *getrip_conf.local* entry. (See the comments in the *getrip_conf* administration file).þ The value of the *op* key should be the filename of a PPD file containing the options.þ If the filename is not absolute, the file will be looked up in the default Xinet PPD directory (*/var/adm/appletalk/ppd*s on UNIX and, by default, *C:\ProgramþFiles\Xinet\FullPress\Admin\ppds* on Windows systems).

To support internationalization, the filename is first looked up with the current locale as a suffix.þ For example, you can have English strings in an option file called, say, *myOpts* in the PPD directory, and german strings in an equivalent file called *myOpts.de*, and set the *op* key like this `:op=myOpts:`.

What's in the *option* file?

The file referenced by the *op* key should be in PPD format, though it need not contain all the usual PPD-ish information.þ The only things used from the file are the *NickName* and all *User Interface* sections (delimited by *OpenUI* and *CloseUI* lines).

The GUI shows a new configuration option button, named *Output* by default, to access the *UI* options.þ The button name will be set to the *NickName* in the option file, if the file has one.þ Clicking that button brings up a configuration window showing each *UI* option and allowing the user to select them and save the options.These options get written to an *OUTPUT.PPD* file in the print queue's administration directory.þ The "PostScript" from the *User Interface* options gets saved to *OPTIONS.PS* in the same directory.

See an example in the *PDFX.OPT* file in the PPD directory.þ It works, along with the *pdfx* file in the *psfiles directory*, to implement all the PDF/X-1a functionality in WebNative Suite.

Chapter 3

WebNative Development Tools

WebNative® Suite provides a Web interface to WebNative Suite volumes. WebNative accesses assets in the file system and provides display, search, ordering and download, conversion, and other features via a standard Web browser. Depending on each user's or group's permissions, WebNative allows users to upload files, modify images and download them (individually or in batches), create new subdirectories, rename, copy and delete files and directories.

The display mechanism for WebNative uses "Styles" which are written in Javascript. Xinet creates and distributes a set of Styles with WebNative, and the product is designed to allow users to customize their own look-and-feel by modifying these supplied Styles or writing their own. WebNative also includes plug-ins for QuarkXPress and Adobe InDesign on Macintosh platforms, which embed previews and linked image information inside the document in a Xinet-proprietary format. WebNative parses this information from the files on-demand and displays this information in a Web browser.

WebNative customization possibilities

Besides the use of WebNative Portal, there are five ways to customize WebNative Suite Web interactions:

1. Basket plugins

   The section, "The WebNative Basket API" on page 16 of this guide, describes the Basket API. Xinet supports the creation of CGIs accessed by users who are allowed to use WebNative shopping baskets. These CGIs generally perform a function such as a batch-conversion of all files in the basket, or email an administrator to alert them to a request made.

2. Buttons

   Rather than customizing the whole Style, you may wish to add or modify buttons that perform specific actions within each Style. The section, "Xinet Button API" on page 24 of this guide describes the addition of custom buttons, including ways to add them to the Javascript files that control interaction between the Styles and Xinet CGIs.

3. Styles

   WebNative uses Styles to customize a user's or group's online experience. The default Style for all actions is "House," Xinet's own Style. However, users often want their site tailored to their brand —colors, logos, metadata, etc. can be integrated to create a unique

face for WebNative. For more information on using and creating custom Styles, see "Customizing WebNative Styles" on page 28 in this guide.

4. WebNative CGIs

   WebNative is comprised of CGI programs that are tied together to generate the WebNative environment. In many instances, for example, when building custom interfaces or working with database integration, there may be reasons to call the CGIs individually. For more information about what the different CGIs do, how to call them, and what results they return, refer to"WebNative Suite CGIs" on page 33.

5. Employing WebNative *Uploader* in an AppleScript

The WebNative Basket API

Introduction

The *WebNative Basket API* has been designed to allow Xinet integrators to provide unique services while still preserving the easy-to-use interface of WebNative. The *Basket API* allows additional functionality to be added to WebNative in the form of simple "plugins." The API is designed to keep the plugins easy to write while still providing enough flexibility to allow most user demands to be met.

Basic programming steps

This section outlines the basic steps you'll need to take to program your own *WebNative Basket API*. After you've read the examples which follow it, you should be able to follow these general steps to develop your own interfaces.

In order to generate any WebNative plugin, you need three pieces:

- An icon which represents the operation to users, stored normally as a GIF image no higher than 30 pixels
- A name for the operation
- A CGI program that accepts a basket as an argument

Now, here are the steps to follow:

1. First, name the operation, develop your CGI, and design the icon you'll need.

   While generating the icon and coming up with a name are quite simple, generating the actual CGI is a little bit more work. The CGI can be any program executable on the server, including compiled C programs, shell-scripts, PERL scripts, etc. If the program wishes to invoke a user-interface, it must act as a CGI-form and arrange to have itself called again. "Examples of custom shopping basket operations" on page 18 presents examples of actual CGIs.

2. Place a copy of your program and icon in the WebNative *plugins* directory, set appropriate permissions, and put a copy of the new program in each user's configuration options.

On Windows systems, use the Windows interface to do these things, placing the icons in the folder *C:\ Program Files\Xinet\WebNative\Admin\images* and the plugin itself in the folder *C:\Program Files\Xinet\WebNative\Bin\plugins*.

On UNIX systems, do the following:

```
# cp myplugin /usr/etc/webnative/plugins
# chmod 4555 /usr/etc/webnative/plugins/myplugin
# chown username /usr/etc/webnative/plugins/myplugin
# cp myicon /var/adm/webnative/images/myicon.gif
# chmod 444 /var/adm/webnative/images/myicon.gif
```

3. Then *cd* to the */var/adm/webnative* directory on UNIX systems or open the *C:\Program Files\Xinet\WebNative\Admin* on Windows systems. Edit the *basket.config* file there, so that it will contain information about the components of the API you've written and how you'd like users to interact with the API.

   The format of each entry in the *basket.config* file (tab separated) is as follows:

   ```
   CGI-program     name     icon     username     flags
   ```

   where:

   - *CGI-program* represents the name of the CGI. The program should always be located in */usr/etc/webnative/plugins* on UNIX systems and *C:\Program Files\Xinet\WebNative\Bin\plugins* on Windows systems. It should be executable by the world.
   - *name* represents the name of the plugin which will appear in pop-up menus and in title bars.
   - *icon* represents the name of the icon file, which must be stored in */var/adm/webnative/images/* on UNIX systems and *C:\Program Files\Xinet\WebNative\Admin\images* on Windows systems.
   - *username* is not used at this time.
   - *flags* is a decimal number representing the options set for this plugin. The flags themselves are the bits set in the number's binary representation:

     1 in this field means that new users will see this plugin by default.
     2 says the plugin handles archive files.
     4 means the plugin needs the WebNative Suite database

     (More flags may be defined in the future.)

4. Test the plugin to make sure it functions correctly.

   The user *nativeadmin* can delete plugins.

   By default, plugins will run as the user *nobody* and have that user's permissions. This is standard Apache behavior. If the plugin needs to be run as a different user with more permissions (say, as *root*), you have the option to make the plugin a *setuid*(2) program, using the following command:

   ```
   # chmod 4755 plugin_name
   ```

   This allows the plugin to run as the current owner of the program. If the owner is *root*, then *root* permissions will apply when the program is run.

However, this approach will not work if the plugin is a script, as UNIX, as a security measure, does not allow scripts to be run with *setuid*(2). If you have written a script and *nobody* permissions are insufficient, you have three options:

- Change permissions on the server so all affected areas are writeable by *world*.
- Rewrite your script as a program and make it run as *setuid*(2).
- Write a one line C program that simply calls the script. Then make that program be *setuid*(2).The permissions will be inherited by the script.

Examples of custom shopping basket operations

This section describes some sample plugin operations in order to demonstrate the power and simplicity of the WebNative plugin interface. After describing what they do, we will show how they were implemented as examples of what can be done easily and quickly to customize WebNative.

Ordering images

For a variety of reasons, some prepress shops may not want customers to download images over the Web; for example, the available bandwidth may not make it reasonable, or they may be hesitant for legal considerations, or the shop may want to charge for the service of providing the images. A reasonable scenario might be for these shops to allow customers to *view* their images via WebNative but not be able to download them. The customers could select which images they wanted, and using the basket interface, request that the service provider, for example, place the images on removable media and send them to the customer. Using the *Request shipment* interface, the customer would provide the mailing address, and select the type of media. In our example, the plugin emails an operator instructions about what to

generate; but many other mechanisms are possible. We've programmed an interface which looks like this:

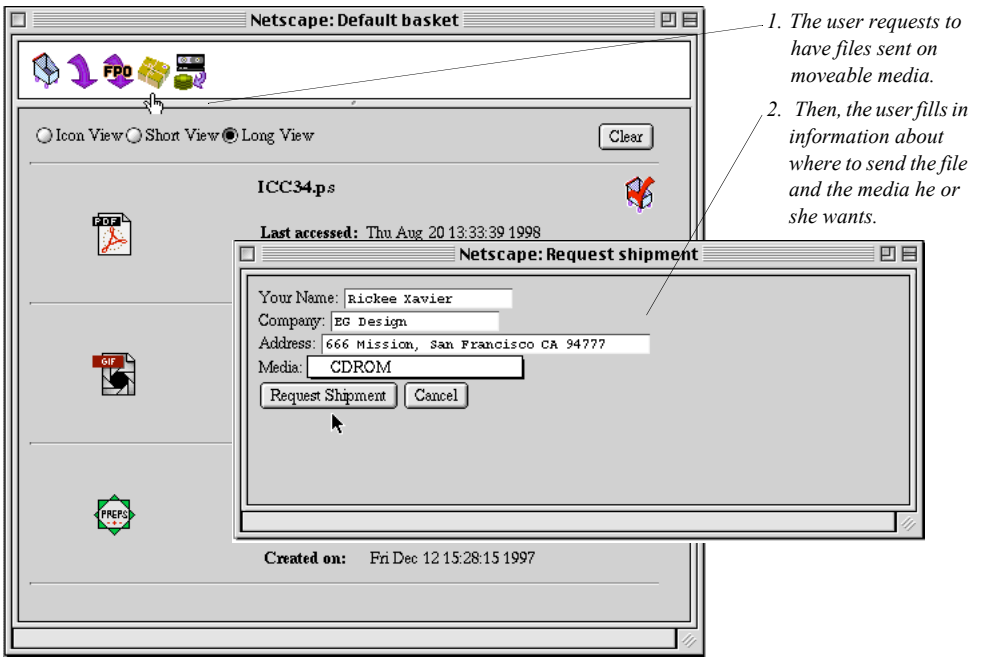


Figure 3-1 Ordering an image

Programming the "Request shipment" interface

After deciding on the name *Request shipment*, we created the wrapped package icon for the GUI. The next step was to write the CGI which was executable on the server.

In this case the program provides a way for the user to select the type of media which will be shipped and to supply an address. A complex user interface wasn't necessary. We implemented the program as a C-Shell script; it could, however, be easily implemented in any language. The source code, shown below, is also available online in */usr/etc/webnative/plugins/ship on UNIX systems and C:\Program Files\Xinet\WebNative\Bin\plugins\ship on Windows systems*.

```
root@xinet:/usr/etc/webnative/plugins 4 -> more ship
#!/bin/csh -f
#
# A very simple example of a webnative plug-in
# This shell script requests some data from the user, then when all
# the information is collected it sends mail to another user with the
# information collected and the names of the file in the users' collection.
# This sample is mainly here to show how to get input from the user and
# combine it with the collection information

# This is the header.  It tells the browser that we are producing html.
# Note the 2 carriage returns, which are essential to end the header
echo "Content-type: text/html"
            echo ""
            echo "<HTML>"
```

```
# Set the title bar of the window.
echo "<TITLE>Request shipment</TITLE>"


# Initialize some variables
set NAME=""
set COMPANY=""
set ADDR=""
set MAILUSER="root"
set MEDIA=""
set nonomatch

if (-f "/usr/adm/webnative/$REMOTE_USER/mailto") then
        set MAILUSER=`cat "/usr/adm/webnative/$REMOTE_USER/mailto"`
else if  (-f /usr/adm/webnative/mailto)  then
        set MAILUSER=`cat /usr/adm/webnative/mailto`
endif


# Extract our input variables (come through stdin) and place them
# in their equivalent csh variables. You are not expected to understand this
# Made extra complicated by Solaris sed bugs.
cat > /tmp/tmp.$$
echo " " >> /tmp/tmp.$$
foreach arg ( `cat /tmp/tmp.$$ | sed -e "s/&/ /g" -e "s:%2F:/:g"` )
        set var=`echo $arg | sed 's/=.*//'`
        set value=`echo $arg | sed -e 's/^.*=//' -e 's/+/ /g' -e
's/%[0-9ABCD][0-9ABCD]/ /g' -e 's/\\r//'`
        eval "set $var="\"$value\""
end
rm -f /tmp/tmp.$$

# Get rid of odd argument passing artifacts
set basketfile=`echo $1 | sed -e 's/\\//g'`

set MAILER=""
foreach file (/usr/ucb/Mail /usr/bsd/Mail /usr/sbin/Mail /usr/bin/mail)
        if (-x $file) then
                set MAILER=$file
                break
        endif
end
if ($MAILER == "") then
        echo "Sorry, I can't send mail"
endif

# Here we check and see if the information is filled out.  If so, it
# means we are being called from ourselves and the submit button

if ("$NAME" != "" && "$COMPANY" != "" && "$ADDR" != "") then

# generate a mail message in a temporary file and place the user
```

```
# information in it.  The $$ makes the file unique.
        touch /tmp/mailreq.$$
        echo "$NAME" >> /tmp/mailreq.$$
        echo "$COMPANY" >> /tmp/mailreq.$$
        echo "$ADDR" >> /tmp/mailreq.$$
        echo " " >> /tmp/mailreq.$$
        echo "Requested a $MEDIA be shipped containing: " >> /tmp/mailreq.$$
        echo " " >> /tmp/mailreq.$$
        # The basketfile variable contains the name of the collection file
        # In this case we just list out the contents of the file to
        # our temp file.
        cat /tmp/mailreq.$$ "$basketfile" | $MAILER -s "Web Order" $MAILUSER
        # Clean up the temporary file
        rm /tmp/mailreq.$$
        # Give the user feedback.  You can't have buttons
        # unless you have a FORM
        echo "<FORM>Thank you for your order $NAME<BR>"
        echo "Your $MEDIA will be sent as soon as possible."
        echo "<P>"
        echo "<INPUT TYPE=BUTTON VALUE='Close Window'
onClick='window.close()'>"
        echo "</FORM></HTML>"
        exit 0
endif


# Here the user filled out some of the information but not all
# A more robust program would complain about the specific
# entries that were not filled out.
if ("$NAME" != "" || "$COMPANY" != "" || "$ADDR" != "") then
        echo "Please fill out all of the requested fields"
endif

# Request the information if we do not have it yet
echo "<FORM METHOD=POST>"
echo "Your Name: <INPUT TYPE=TEXT NAME=NAME VALUE='$NAME'><BR>"
echo "Company: <INPUT TYPE=TEXT NAME=COMPANY VALUE='$COMPANY'><BR>"
echo "Address: <INPUT TYPE=TEXT NAME=ADDR VALUE='$ADDR' SIZE=40><BR>"
# Use a fancy pull-down for the media selection
echo "Media: <SELECT NAME=MEDIA>"
echo "<OPTION value=CDROM selected>CDROM"
echo "<OPTION value='ZIP Disk'>ZIP Disk"
echo "<OPTION value='Syquest Cartridge'>Syquest Cartridge"
echo "</SELECT>"
echo "<BR>"

# We have a button to submit the form (calls the same program)
# and one to cancel the whole operation.
echo "<INPUT Type=submit Value='Request Shipment'>"
echo "<INPUT TYPE=BUTTON VALUE=Cancel onClick='window.close()'>"
echo "</FORM>"
```

As you can see, it is not difficult to generate a WebNative plug-in. Some simple enhancements that could be made to this plug-in based on a site's individual needs include (but are not limited to):

- Loading the customer's address automatically from a database.
- Automatically generating a proof-sheet (with the contents of the media, and preview images) to go along with the media.
- Providing the customer with a summary of charges for the operation before they request it.
- Actually generating the media on the server (see third-party plugins for more info).

Requesting restoration of archived files

Most sites operate in an environment where each customer job is archived sometime after printing. Through integration with the archiving and back-up software, FlashNet from SGL, WebNative makes it possible to review what's been placed in FlashNet archives, even though the files are no longer online. We include a CGI with WebNative which allows end-users to request restoration of archived files. You can look at this CGI in */usr/etc/webnative/plugins/restore* on UNIX systems and *C:\Program Files\Xinet\WebNative\Bin\plugins\restore* on Windows systems.

Requirements of a CGI

Now that we have seen sample CGIs, it will make more sense to describe what a CGI must do. The CGI will be called initially with a single argument, which is a file containing the list of the files in the basket. If the CGI requires any user interaction, then it is responsible for generating an HTML form that calls the CGI again. It is also responsible for passing on the name of the basket file. Even without an API, the CGI should generate valid HTML telling the user whether the operation succeeded or failed.

The CGI will always run in its own window. Opening other windows is not recommended. The CGI itself has permission to do very little. If it needs to access the actual files referenced, it should do so by calling the built-in WebNative CGIs. The most common CGI to call is *getimage*, which will preview FPOs, or high-res data for files. This allows CGIs to be programmed without having to worry as much about their security implications. If necessary, CGIs can be specified to run as other users, but this is not recommended for novice programmers.

Basket files

WebNative maintains a user's basket state as a text file on the server. The file exists in the user's WebNative home directory *(/var/adm/webnative/*`user_name` on UNIX systems and *C:\Program Files\Xinet\WebNative\Admin*`user_name` on Windows systems), with the name of the file being controlled entirely by the *BAKSETNAME* cookie. Typically, WebNative maintains the *BASKETNAME* cookie which is composed of the user's IP address, followed by eight random hexadecimal digits, followed by the extension *basket*, e.g., *192.168.0.23.ca84e231.basket*. This format ensures that users sharing a login and IP address,

like those behind a NAT router, have unique baskets. However, a style author can take control of the *BASKETNAME* cookie and use any naming scheme he or she prefers.

The basket file contains a new-line-delimited list of file descriptions. Those descriptions follow two formats:

- The first, for regular files that are not archived, is simply the path to the file on the server.
- The second, for files that have been archived, is as follows:

  `voltime:volname:archiveno:path`

where `voltime` represents the time the file was archived, `volname` indicates the name of the volume from which the file was archived, `archiveno` indicates how many times the file has been archived previous to the current archive, and `path` is where the file lived in the file system at the time the archive was created.

Plug-in suggestions

The plug-in interface is designed to provide a powerful tool for customizing without sacrificing the simple WebNative interface. In order to make your plug-in effective, try to follow these guidelines:

- Keep it simple.
  Don’t give users more interface then they need. If you need more than one screen to enter your interface, consider implementing two different plugins. Anytime you can make a reasonable decision for the user, do so. Don’t get caught in the trap of thinking that more options mean a better program. If in doubt, leave it out.
- Try to make your icon and name meaningful.
  The first interaction a user has with your plug-in will be the icon and pop-up name. Make sure the icon indicates function, and does not require previous knowledge. For example, don' t use your corporate logo, as the general user probably has no idea what you make. Make the name descriptive enough to let the user decide if it is what they want to do. Good icons are hard to make. Don't be embarrassed about changing your icon if you find people don't get it. The Apple publication *Macintosh Human Interface Guidelines* gives some great tips for designing icons. You can download a copy of this publication from *ftp://ftp.apple.com/devworld/Technical_Documentation/Human_Interface/Human_Interface_Guidelines.sit.hqx*
- Allow the user to cancel.
  At any time before finishing the transaction, users should have some method of canceling the request, and if applicable, changing their requests.
- Check for errors.
  The most time-consuming part of writing a plug-in is checking and properly reporting error conditions (full disk, etc.). If you are writing a general-use plug-in, take the time to get these conditions right, and test every one of them.
- Use guinea pigs.
  Once you are done, have people run the plug-in while you watch. Don't tell them what to do, and see where they trip.

Xinet Button API

Introduction

The button API was established to give WebNative Style authors a way to extend the default behavior of WebNative Styles without having to actually modify the Styles themselves.

It allows a Style author to add a context-sensitive button to the list of existing, file-specific WebNative buttons, which reacts to user direction in a way completely defined by the button author. Moreover, it allows the button author to do so without modifying the host Styles. This enables button authors to extend Styles without having to sacrifice new features implemented in future Style revisions.

File organization

UNIX: */usr/etc/webnative/styles/filebuttons*
Windows: *C:\Program Files\Xinet\WebNative\Bin\styles\filebuttons*

Location of all global button definitions, as well as the master *button.js* file. Each button definition should be stored in a separate file, should only contain JavaScript, and must have the *.js* extension. Files that only contain JavaScript can be marked with the *WN_Include_Safe* tag. Button definitions stored in this directory are included for every WebNative action.

UNIX: */usr/etc/webnative/styles/filebuttons/buttons.js*
Windows: *C:\Program Files\Xinet\WebNative\Bin\styles\filebuttons\buttons.js*

The master buttons JavaScript file, which contains button-related object definitions, as well as the global button manager reference and the global action context reference. This file is included in every WebNative action.

UNIX: */usr/etc/webnative/styles/filebuttons/readme*
Windows: *C:\Program Files\Xinet\WebNative\Bin\styles\filebuttons\button.js*

This file is actually a copy of this chapter.

UNIX: */var/adm/webnative/*`username`*/filebuttons*
Windows: *C:\Program Files\Xinet\WebNative\Admin*`username`*\filebuttons*

Location of all user-specific button definitions. Files in these directories should follow all recommendations suggested for global button definitions, with the additional stipulation that none be named *buttons.js*. Button definitions are included with every WebNative action, but only for the user for whom they are defined.

API information

Context flags are used to describe appropriate situations for displaying a button. Flags can be bitwise-OR'd together to create an integer describing all appropriate situations.

- *CONTEXT_BROWSE*

  If enabled, the button should be displayed in *browse*.

- *CONTEXT_SEARCH*

  If enabled, the button should be displayed in *search.*

- C*ONTEXT_IMAGEINFO*

  If enabled, the button should be displayed in *imageinfo.*

- *CONTEXT_BASKET*

  If enabled, the button should be displayed in the *basket*.

- *`WNButton class`*

  The *WNButton* `class` describes a button's characteristics, as well as provides a template for Style interaction with that button. Interaction with *WNButton* objects typically never extends beyond creation, initialization, and registration with the button manager.

- *WNButton(img, alt, context, orderPreference)*

  img — A string describing the path for image used to represent the button. If the image path does not begin with a forward-slash, it is assumed that the image exists in */var/adm/webnative/images* (UNIX), *C:\Program Files\Xinet\WebNative\Admin\images* (Windows), or a Style subfolder within. Icon Style dimension constraints apply regardless of where the image exists.

  alt — A string describing the alt text for the HTML IMG tag generated for the button.

  context — An integer describing the appropriate context for displaying the button.

  orderPreference — A floating point number describing the button's order preference, relative to other buttons registered with the button manager. Buttons with a higher order preference will appear before those with lower order preference.

- *checkPermission(file, userPerms)*

  The *checkPermission* function is stubbed on object creation. It is up to button authors to define an implementation for it and register it with their button objects. Those implementations should examine the characteristics of the objects passed in and determine whether or not it is appropriate to display the button under those circumstances. If the user should view the button, `checkPermission()` should return true. If not, it should return false.

  file — An object representing a file, as defined in *buildobject(),* located in *global.js*.

  userPerms — An object representing the user's permission configuration, as defined in *userinfo(),* located in *global.js.*

- *getAnchorProperties(file, userPerms, args, context);*

  As with *checkPermission(), getAnchorProperties()* is stubbed on object creation. Button authors are required to create an implementation and register it with their button object.

The implementation should examine the arguments passed in and create a string describing the anchor tag properties for their button.

file An object representing a file, as defined in *buildobject(),* located in *global.js.*

userPerms An object representing the user's permission configuration, as defined in *userinfo(),* located in *global.js.*

args An associate array containing any extraneous arguments Style authors wish to pass to button authors. House Style for example, passes an `Fref` object (defined in *global.js*), which button authors can use to generate *mouseover* code. Can be *null*.

context An integer describing context the button is being considered in.

- *registerCheckPermission(cpf)*

  The *registerCheckPermission()* function is used to register a new *checkPermission()* definition with the button object.

  cpf A reference to a *checkPermission()* function implementation.

- *registerGetAnchorProperties(gapf)*

  The *registerGetAnchorProperties()* function is used to register a new *getAnchorProperties()* definition with the button object.

  cpf A reference to a *getAnchorProperties()* function implementation.

- *getHouseMouseOverProperties(args, msg)*

  The *getHouseMouseOverProperties()* function is a utility function which automatically generates house *mouseover* handlers, for use in `getAnchorProperties()` implementations.

  args Associate array containing arguments passed from house Styles. Typically, button authors will simply pass the args Array received in *getAnchorProperties()* directly into *getHouseMouseOverProperties()*.

  msg A string describing the message to be displayed by the *mouseover* code.

- *WNButtonManager* `class`

  The *WNButtonManager* `class` is responsible for maintaining a list of registered buttons and handling button display requests from WebNative Styles. Button authors need only concern themselves with the button registration functionality. Style authors need only concern themselves with the *writeButton()* function. A button manager can have a maximum of 1024 buttons registered at one time.

- *WNButtonManager()*

  Button and Style authors should never overwrite the global button manager with a new instantiation. Doing so could potentially disable other third party buttons registered with the button manager.

- *addButton(button)*

  The *addButton()* function registers a button with the button manager.

  button A reference to the button object to be added.

- *writeButtons(framedoc, file, userInfo, args, context);*

  The *writeButtons()* function loops through the list of registered buttons and writes each to an arbitrary document, given the user has permission and the button works in the current context.

framedoc	A reference to the document in which the *writeButtons()* function should write HTML. It is assumed the document object has already had `open()` called on it.
file	An object representing a file, as defined in *buildobject(),* located in *global.js.*
userPerms	An object representing the user's permission configuration, as defined in *userinfo()*, located in *global.js.*
args	An associative array containing arbitrary arguments defined by Style authors. Can be *null.*
context	An integer describing the current action context.

Usage and examples

Creating new buttons is a four step process:

1. Determine whether to make the button global or user-specific. Create a file in the appropriate location.

   Example:

   Global button definition:

   UNIX: */usr/etc/webnative/styles/filebuttons/uploadbutton.js*

   Windows: *C:\Program\Files\Xinet\WebNative\Bin\styles\filebuttons\uploadbutton.js*

   User-specific button definition:

   UNIX: */var/adm/webnative/*`username`*/filebuttons/uploadbutton.js*

   Windows: *C:\Program\Files\Xinet\WebNative\Admin*`username`*\filebuttons\uploadbutton.js*

2. In that new file, create a button object to represent your button.

   Example:

   ```
   var myButton = new WNButton("wn_upload.jpg",
            lang.upload,
            .9,
            CONTEXT_SEARCH|CONTEXT_BROWSE);
   ```

3. Define and attach *checkPermission()* and *getAnchorProperties()* functions to your button object.

   Example:

   ```
   function uploadCheckPermission(file, userInfo) {
      return (file.isadir && userInfo.upload);
   }

   function uploadGetAnchorProperties(file, userInfo, args, context)
   {
   ```

```
    return "HREF='/webnative/upload?" + file.path + "'
    TARGET='uploadWindow'";
}

myButton.registerCheckPermission( uploadCheckPermission );

myButton.registerGetAnchorProperties( uploadGetAnchorProperties );
```

4. Register your button with the button manager.

   Example:

   ```
   wnButtonManager.addButton(myButton);
   ```

Please see the file, */usr/etc/webnative/styles/filebuttons/xinetsample.js* (UNIX) or *C:\Program Files\Xinet\WebNative\Bin\styles\filebuttons\xinetsample.js* (Windows), for a complete example of a working button.

Customizing WebNative Styles

You certainly are not limited to the Styles and plugins which ship with WebNative. This chapter provides an introduction to customizing WebNative Styles. "The WebNative Basket API" on page 16 provides information about plug-in development.

With some knowledge about JavaScript, you can make copies of and modify any of the existing Styles or create entirely new ones of your own. JavaScript is a relatively simple language, and with a little reading you will probably understand what is going on. Even if you can't program, you can edit the strings for localization without a problem. You can also edit some (but not all) of the Style files with a GUI editor. Please be careful, however, to test that everything functions correctly afterwards.

Overview

Organization of Style folders

Inside the */usr/etc/webnative/styles* folder on Unix systems and the *C:\Program Files\Xinet\WebNative\Bin\styles* on Windows systems, you will find subfolders for the various WebNative actions. Inside each action folder, you will find various subfolders for available "styles" that may be applied to that action. While the Styles you see initially will be useful in their own rights, they are mainly intended as starting points for your own customizing endeavors.

How Styles work

After you have defined your Styles, you will use the *WebNative Administration Style Configuration* GUI to assign them to end users. Refer to "Configuring a user's styles" in the *WebNative Administration Guide* for more information about assigning existing Styles. After you select a Style for a user, and that user performs the action indicated (for example, *upload*), the *WebNative Style Manager* figures out how to present things to the user by looking at the files listed in Table 3-1 on page 29, in the order in which the table lists them.

The purpose of the hierarchy of files is to allow customizing of the whole interface with as little work as possible. Table 3-1 also shows where the files are located and their purposes. *Keep in mind that any time there are conflicts in settings, the settings in the last-read file will rule*.

Table 3-1 Style files (in order of inclusion)

File	Location	Contents
global.js	**UNIX:** */usr/etc/webnative/styles/* **Windows:** *C:\Program Files\Xinet\WebNative\Bin\styles*	Global JavaScript from Xinet useful in all actions and Styles
language/style.js	**UNIX:** */usr/etc/webnative/styles/* **Windows:** *C:\Program Files\Xinet\WebNative\Bin\styles*	Contains the English version of strings
`action`*/style.js*	**UNIX:** */usr/etc/webnative/styles/* **Windows:** *C:\Program Files\Xinet\WebNative\Bin\styles*	JavaScript from Xinet useful in the particular action
language/`style`*/language.js*	**UNIX:** */usr/etc/webnative/styles/* **Windows:** *C:\Program Files\Xinet\WebNative\Bin\styles*	Contains translations of the English strings into the selected language
local.js	**UNIX:** */var/adm/webnative* **Windows:** *C:\Program Files\Xinet\WebNative\Admin*	Local JavaScript useful in all Styles. Can override global.
user.js	**UNIX:** */usr/adm/webnative/*`user`*/* **Windows:** *C:\Program Files\Xinet\WebNative\Admin*`user`	JavaScript specific to the user
header.html	**UNIX:** */var/adm/webnative/*`user` **Windows:** *C:\Program Files\Xinet\WebNative\Admin\WebNative*`user`**	The user's header file
`actionstyle`*.html*	**UNIX:** */usr/etc/webnative/styles/*`action/style` **Windows:** *C:\Program Files\Xinet\WebNative\Bin\styles*`action\style`	Action and style-specific information
trailer.html	**UNIX:** */var/adm/webnative/*`user` **Windows:** *C:\Program Files\Xinet\WebNative\Admin*`user`	The user's trailer

Hierarchy for assigning Styles to users

It is easy to assign Styles for users using the *Style Configuration* window. If you set up a new user without specifying Styles, however, the *Style Manager* will check to see if you've defined "default Styles" for new users. (You do this by setting Styles for the user *Default* in the Style Configuration GUI.) If you have not set up default Styles, the *Style Manager* will use the *classic* Style for each action. Of course, any Styles you do assign to a user will over-write all defaults.

Where customized Styles belong

It is very important that when you want to create a custom Style that you work on a copy of an existing Style, not the original Style itself. If you modify an original Style provided by Xinet, your changes will be destroyed next time you install a new version of WebNative. If you want to change parameters inside the *global.js* file, you need to copy them into the *local.js* file and modify them there. Since the local files are read after the global files, definitions in them will override ones in the global file.

For example, if you wanted to localize the button names, you would find the code that displays the buttons in *global.js*, and copy it into *local.js*. To complete this example, you would edit *local.js* to change the button names. If on the other hand you want to modify a entire Style for an action, copy the entire directory for the Style into a new directory, for example:

```
cd uploads
cp -pr classic formsubmit
```

Then modify all of the files in *formsubmit* to suit your needs. Miraculously, you have created a brand new Style which you can assign to any user.

When copying an existing Style, be sure to copy all the "support" files used by the Style. Much like Styles, these files will also be overwritten by upgrades. For example, all of the *house* Styles use JavaScript from the file */var/adm/webnative/images/house/common.js*. When copying a *house* Style, be sure to make a copy of the
common.js file and direct your Style to read from this copy.

WebNative actions

WebNative ships with ten *actions*, each of which can be customized through Styles. The actions are described by the CGIs that implement them, including the calling syntax as described in "WebNative Suite CGIs" on page 33. The CGIs are responsible for gathering the correct JavaScript and HTML files, and defining a series of JavaScript variables for use by the Style. Descriptions of each action follow:

Table 3-2

action	Description
basket	The basket action controls the display of shopping baskets and is comprised of a collection of separate CGIs that control basket Styles. See page 77 and page 84 for more information.
browse	The *browse* action is responsible for displaying the contents of folders in WebNative, implemented by the *listdir* CGI. (See "Listing directories and volumes" on page 44 for details.) It receives an array of *files* from *0* to *(numfiles - 1)*. Each *files* element is an object as returned by *buildobject()*. The object may have image structures and/or archive information. Files and directories (both online and archived) will be returned together. The array is sorted in a Mac-style sort by object name.

Table 3-2

action	Description
css	The *css* action is used to isolate all Cascading Style Sheets (CSS) properties in a central location. It enables a WebNative Suite Style author to apply a single set of CSS definitions to all WebNative users' sessions, enabling greater uniformity and cohesion between actions. Like the language and icon actions, it is included in the result stream for nearly every WebNative CGI and can be applied to all users, including *nativeadmin*.
imageinfo	The *imageinfo* action provides the interface for viewing all information about a file, and optionally changing the *Desktop Comments* field (keywords). To change the comment, this page must "post" a form containing a *comment* `Form` variable. See "Getting detailed image info" on page 60 for details.

language	The language Style applied to this action is used when displaying any text elements generated by WebNative for end-users. The language Style is special, in that the Style Manager always reads the language Style before other Styles.
	To create a language *Style*:
	1. Make a new subdirectory in the *language* directory named for the new language.
	2. Copy the file *style.js* in the language directory to the newly created subdirectory, giving it the appropriate name for the language:
	`cp style.js` *`languagename`*`/language.js`
	3. Edit *`languagename`*`/language.js`, removing the line that says
	`var lang = new Object()`
	4. Translate all of the strings in the file.
	When you define strings, most special non-alphanumeric characters will confuse the browsers, especially various quote symbols. You'll have to use the ISO Latin equivalents for the special characters. Refer to *http://www.w3.org/TR/REC-html32.html* if you need help finding the equivalency, or use a visual editor that understands HTML strings.

mview	The *mview* action (known as *quarkview*[a] in earlier versions of WebNative) allows end-users to preview various kinds of files in their browsers.The users can also follow image links within the files. With the database engaged, *mview* will also write preview information into the database. (You can do this "by hand" from the command line by running *synchvoltodb*(1M) on a directory. The *fpod*(1M) daemon will then call upon *mview* to write preview information to the database.)

Table 3-2

action	Description
mview (contd.)	The *mview* program operates on the following file types: • QuarkXPress documents on WebNative servers that were saved with WebNative XT 3.0 or later • InDesign documents on WebNative servers that were saved with the WebNative ID plug-in (3 or later) • PDF files on WebNative Suite servers Users can follow image links in PDF files that satisfy one of the following: – They were created with the Xinet Annotator XT or Annotator ID plug-in. – They contain OPI comments. On PDF files, *mview* either reads OPI comments or Xinet Annotator marks within the files. On other layout documents, the *mview* program parses the documents and reads out special "slugs" left by the XTs or the plug-ins. The XT and plug-in make preview images for each spread. "Showing QuarkXPress, PDF and InDesign file previews" on page 80 provides details
order	An *order* action presents an order form with specific details about a single image file, and allows users to download images in a format and size customized for their particular needs. Refer to "Custom-ordering images" on page 64 for the calling sequence to do the actual order.
search	The *search* Style is split into two parts, controlled by the files *searchquery.js* and *searchdisplay.js*. The *searchquery.js* file controls the part of the display where the user selects search criteria; the *searchdisplay.js* file controls the part of the display where search results appear. In general, the Style described by *searchdisplay.js* should be very similar to the user's *browse* Style to present a consistent interface; but, this is not a requirement. See "Searching" on page 47 for details.
toplevel	The *toplevel* action presents what is essentially the "welcome" screen the, starting point for all the other pages. It provides a list of the user's configured volumes. See "Entry point for WebNative — toplevel" on page 58 for details.
upload	The *upload a*ction controls uploading files from a user's environment to the server. It is available on volumes that have the *Allow Uploads s*witch set to on. "Sending files to a WebNative server" on page 87 provides details

a. The *quarkview* action remains in the distribution solely for purposes of backwards compatibility.

WebNative Suite CGIs

Introduction

WebNative Suite is comprised of CGI programs that are tied together to generate the WebNative environment. In many instances, for example, database integration, there may be reasons to call the CGIs individually. This document explains what the different CGIs do, how to call them, and what results they return. You should not need to refer to the information in this chapter unless you plan on building custom WebNative interfaces.

Architecture

The WebNative CGIs do not actually create images on the WebNative Suite server. They rely on the engine comprised of *fpod*(1M) and *mkfpo*(1)) to keep the various previews of images up to date. If an image source is missing or out of date, they will signal *fpod*(1) by calling *dtrebuild*(1M). The *dtrebuild*(1M) program will then send a packet to *fpod*(1M), which will schedule an update of the file and call *mkfpo*(1). The *mkfpo*(1) program will actually read the high-res image and generate the necessary previews. Depending on the file and options, some or all of the following preview images will be stored in the resource fork of the file for use by WebNative Suite:

- Custom Icon in Mac icon format (32x32 pixels)
- Small PICT preview (112x112 pixels)
- Small Web GIF or JPEG preview (112x112 pixels)
- 72 DPI or less PICT preview
- 72 DPI or less Web GIF or JPEG preview
- Selectable DPI EPSF or TIFF FPO image

In the list above, any image noted to be "GIF or JPEG" will be a JPEG, unless its source happens to be:

- Black and White
- Masked
- Clipped

If the image has any one of the three attributes listed above, it will be a GIF, because there is no way to represent the image correctly in JPEG.

The phrase "72 DPI or less" means that the image will be 72 DPI unless the user has constrained the absolute size of FPOs. In that case, the resolution may be lower to meet that constraint. In these cases, all three images types will have the same resolution.

Permissions

For the CGIs to work, three conditions must be met:

1. They must be passed a `REMOTE_USER` (in the environment) that is a configured WebNative user. Usually, this is done automatically by the Web browser and server.
2. The file being accessed must be on one of that user's WebNative volumes.
3. In addition, that user's volume must be configured to allow that operation. For example, the user might be restricted from downloading high-res images, in which case the *-high* flag would not work. Failure to authenticate will return an HTTP authentication error header.

Some common JavaScript variables

The following tables describes JavaScript variables which are common to more than one WebNative CGI:

Table 3-3 Common JavaScript variables

Variable	Definition
`uinfo`	The `uinfo` variable points to an object containing information about the current WebNative user. The following fields are contained in the `uinfo` object: • `uinfo.user` — contains the user's WebNative username. • `uinfo.basket` — if *true,* WebNative will allow the user to have a basket • `uinfo.archives` — if *true*, WebNative will allow the user to view archived files/folders • `uinfo.iconview` — if *true,* the user's view should default to `icon` (or nearest equivalent) view • `uinfo.shortview` — if *true*, the user's view should default to `short` (or nearest equivalent) view • `uinfo.longview` — if *true,* the user's view should default to `long` (or nearest equivalent) view • `uinfo.showfpoinfo` — if *true*, then WebNative will allow the user to view FPO information • `uinfo.showdates` — if true, WebNative will allow the user to view file dates. • `uinfo.changecomment` — if *true*, WebNative will allow the user to change *Finder* comments for a file • `uinfo.highres` — if *true*, WebNative will allow the user to view/download high-res files • `uinfo.upload` — if *true*, then WebNative will allow the user to upload files • `uinfo.imgorder` — if *true*, WebNative will allow the user to custom order high-res images.

Table 3-3 Common JavaScript variables

Variable	Definition
	• `uinfo.fileadm` — if *true*, WebNative will allow the user to move/copy/rename/delete files • `uinfo.defaultview` — contains the name of the view to which the user's view should default. • `uinfo.changepassword` — if *true*, WebNative will allow the user to change their own passwords. • `uinfo.primarygroup` — contains the name of the user's primary group. (It will contain the empty string if the user doesn't have a primary group.) • `uinfo.allownonimages` — if *true*, the user can download non-image files when high-resolution image download is disabled. • `uinfo.changekeywords` — if *true*, the user can modify WebNative Suite metadata according to his or her permission set. • `uinfo.showcomments` — if *true*, the user has permission to view the Finder comments associated with files.
`files`	The `files` variable is an array of objects, each describing a file somewhere on the server. Each file object is described by the following characteristics: Note: In the following descriptions, i refers to any arbitrary object within the array. • `files[`*i*`].name` — describes the name of the file • `files[`*i*`].path` — describes the full path to the file on the server • `files[`*i*`].archive` — indicates whether this file is an archive. See the information about the `archive` field below for details. • `files[`*i*`].comment` — describes the *Finder* comments associated with the file • `files[`*i*`].fileid` — describes the unique file ID number associated with the file in the WebNative Suite database. Does not exist for files on non-database-enabled volumes. • `files[`*i*`].type` — describes the Macintosh 4-char file type for the file. • `files[`*i*`].creator` — describes the Macintosh 4-char creator for the file • `files[`*i*`].length` — describes the size of the file (in bytes) • `files[`*i*`].isadir` — if *true*, then the file represented is a directory • `files[`*i*`].inbasket` — if *true,* then the file represented is in the user's basket • `files[`*i*`].cdate` — a JavaScript `Date` object describing the Macintosh creation date for the file

Contd...

Table 3-3 Common JavaScript variables

Variable	Definition

`files` *(Contd.)*	• `files[`*i*`].mdate` — a JavaScript `Date` object describing the Macintosh modification date for the file
	• `files[`*i*`].adate` — a JavaScript `Date` object describing the Macintosh access date for the file.
	• `files[`*i*`].isimage` — if *true*, the file is classified by WebNative as an image.
	
	Example: Consider a file on the server with the path /RAID/ticket23/header.tif. The file object for this file would have
	`files[`*i*`]name = 'header.tif'`
	`files[`*i*`].path = '/RAID/ticket23/header.tif'`
	`files[`*i*`].type = 'TIFF'`
	All other fields would be set appropriately.
	
	If the `archive` field of the file object is not false, undefined, or null, then it contains the following fields:
	• `files[`*i*`].archive.name` — describes the name of the archive
	• `files[`*i*`].archive.number` — describes which back-up of the file this object represents
	• `files[`*i*`].archive.date` — a JavaScript `Date` object describing when this file was backed up
	`files[`*i*`].archive.offset` — an offset used by the *archivepict* CGI to retrieve the archived image.
	• `files[`*i*`].archive.pict` — if *true,* then a picture accessible to the *archivepict* CGI is available for viewing
	• `files[`*i*`].archive.near_line` — if *true*, the file is near-line and may be restored without operator intervention.
	
	Example: If the header.tif had been backed up for the second time on July 31, 2001, it would have an `archive object` with the following values:
	`files[`*i*`].archive.number = 2`
	`files[`*i*`].archive.name = 'header.tif'`
	`files[`*i*`].archive.date = new Date ('July 31, 2001')`
	
	Depending on the image type and the user's permissions, file objects which correspond to images potentially have additional information. This additional information describes all "supporting" images WebNative Suite has made on the server

Table 3-3 Common JavaScript variables

Variable	Definition

	Note: Even though a file is an image, there is no guarantee that any of the fields will be present. It is always a good idea to test them for existence before referencing them.
	• `files[`*i*`].highres` — describes the original high-res image
	• `files[`*i*`].fpo` — describes the FPO made of the high-res image
	• `files[`*i*`].largeweb`— describes the Web-ready preview of the image made by WebNative
	• `files[`*i*`].smallweb`— describes a scaled down version of the Web-ready preview made by WebNative
	• `files[`*i*`].custicon`— describes a custom icon for the image, similar to Photoshop's preview icons.
	
	Each of these support images is described by the following information. Note: In the following descriptions, support_image refers to any of the five support images described above.
	• `files[`*i*`].`*support_image*`.width` — describes the image's width
	• `files[`*i*`].`*support_image*`.height`— describes the image's height
	• `files[`*i*`].`*support_image*`.resolution` — describes the image's resolution (in DPI)
	• `files[`*i*`].`*support_image*`.imagetype` — describes the image's file type
	• `files[`*i*`].`*support_image*`.colorspace` — describes the image's colorspace
	• `files[`*i*`].`*support_image*`.length` — describes the image's size (in bytes).
	• `files[`*i*`].`*support_image*`.dbfield` — the WebNative Suite database file ID for the associated image file.
	
	Example: If header.tif was 6*40 pixels* by *480 pixels*, and had a resolution of *72 DPI*, it would have a `highres` *o*bject with the following values:
	`files[`*i*`].highres.width = 640`
	`files[`*i*`].highres.height = 480`
	`files[`*i*`].highres.resolution = 72`
	All other fields would be set appropriately.

`numfiles`	The `files` array is, typically, much bigger than the actual number of items within it. The `numfiles` variable describes how many files are actually represented in the `files` object. This variable should be used in place of `files.length`.

Table 3-3 Common JavaScript variables

Variable	Definition

`dir`	The `dir` *v*ariable represents the current directory being browsed, searched, etc. It is described by the following fields:
	• `dir.path` — describes the full path to the directory on the server
	• `dir.htmlpath` — an HTML-friendly, printable path in Macintosh path syntax
	• `dir.name` — the name of the directory
	• `dir.inbasket` — if *true,* then the directory is currently in the user's basket
	• `dir.archive` — has the same characteristics as `files[`*i*`].archive`
	Example: The directory */RAID/ticket23* would have a `dir` object with the following values:
	`dir.htmlpath = 'RAID:ticket23:'`
	`dir.name = 'ticket23'`
	`dir.path = '/RAID/ticket23'`
	Other values would be set accordingly.

`volumes`	The `volumes` variable points to an array of objects describing the volumes available to the user. Each object is described as follows. Note: In the following descriptions, *i* is merely a place holder, and refers to any arbitrary object within the array. `volume[`*i*`].path` — the full path to the volume on the server `volume[`*i*`].name` — an HTML-friendly name for the volume, as defined by the WebNative administrator `volume[`*i*`].uinfo` — a `uinfo` object, as described in "Common JavaScript variables" on page 34, detailing the user's permissions for the given volume.
`numvols`	The `numvols` variable contains the number of volumes available to the user for browsing and/or searching.
The following two variables only pertain to the WebNative Suite database.	
`keywords`	The `keywords` variable points to an array of objects representing WebNative Suite database fields. Each database field is described by the following characteristics. Note: In the following descriptions, `i` refers to any arbitrary object within the array. • `keywords[`*i*`].id` — a unique ID number for this field • `keywords[`*i*`].order` — the rank in which the field should be displayed • `keywords[`*i*`].editable` — if true, a user can modify this field if they have the permission • `keywords[`*i*`].name` — the printable name for this field`keywords[`*i*`].type` — the field type according to the database

Table 3-3 Common JavaScript variables

Variable	Definition

	• `keywords[`*i*`].size` — the size of the field in the database, specified in 1-byte characters.
	• `keywords[`*i*`].cols` — the number of columns the field would have if displayed
	• `keywords[`*i*`].rows` — The number of rows the field would have if displayed.
	• `keywords[`*i*`].precision` — for floating point fields, the number of decimal places to use
	• `keywords[`*i*`].flags` — general purpose information storage area
	• `keywords[`*i*`].display` — information regarding how to display to the keyword
	• `keywords[`*i*`].hidden` — if *true,* then this field should not be shown to the user
	• `keywords[`*i*`].limited` — if *true,* then only a limited number of values can be used in this field
	• `keywords[`*i*`].values` — if `limited` is *true*, then this points to an array of values which can be used for this field
	• `keywords[`*i*`].fixed` — if *true*, then the field can not be edited
	• `keywords[`*i*`].indexflag` — if *true*, the field is indexed.
	• `keywords[`*i*`].permflags` —if *non-zero*, the field is editable.
	• `keywords[`*i*`].htmlvalue` — contains the same data as `keywords[`*i*`]value`, albeit guaranteed to be in a displayable encoding.
	• `keywords[`*i*`].htmlvalues` — contains a displayable list of potential values for the metadata in question.
	• `keywords[`*i*`].dbvalues` — contains a list of potential values for the metadata in question, as they appear in the database.
	• `keywords[`*i*`].valueflag` — general purpose flag for keyword values
	• `keywords[`*i*`].value` — the current value of this field for a given file
	Note: This variable is only set when it is produced by the image-info CGI or any other CGI that deals with only one file.
	Example: Consider a limited database field with *ID = 17* and a value set of *"queried," "in process"* and *"done."* It would have a `keyword object` with the following values:
	`keyword[`*i*`].id = 17`
	`keyword[`*i*`].limited = true`
	`keyword[`*i*`].values = newArray(3)`
	`keyword[`*i*`].values[0] = 'Queued'`
	`keyword[`*i*`].values[1] = 'in process'`
	`keyword[`*i*`].values[2] = 'done'`
	Other values would be set appropriately.

Table 3-3 Common JavaScript variables

Variable	Definition
`numkeywords`	The `numkeywords` variable describes how many keywords are actually represented in the array pointed to by `keywords`. This variable should be used in place of `keywords.length`, as the array pointed to by `keywords` might be bigger than the actual number of keywords within it.

Displaying icons

With all WebNative CGIs, the *icons* action is performed every time a WebNative CGI is invoked. The action prints information about where WebNative Style files should look for WebNative icons, as well as providing information about how to display those icons. *Note*: Only icons drawn with the *putwnicon()* JavaScript function will respond to the icons action. Icons placed any other way may not respond as expected.

On UNIX systems, the icons rely on:

/usr/etc/webnative/styles/icons/`style`*/icons.js*

and reside in the directory

/var/adm/webnative/images/`style`*.*

The equivalent files on Windows systems reside in:

C:\Program Files\Xinet\WebNative\styles\icon`style`*\icons.js*
C:\Program Files\Xinet\WebNative\Admin\images`style`*.*

Arguments:
Not applicable

Corresponding HTML files:
(where `style` can be *classic*, *buttons, smallbuttons, or house)*

Corresponding HTML files:
(where `style` can be *classic*, *buttons, smallbuttons, or house)*

Table 3-4 Displaying icons

Name	Description	Classic	Buttons	Small buttons	House
wn_basket.gif	basket icon				

Table 3-4 Displaying icons

Name	Description	Classic	Buttons	Small buttons	House
wn_basketchecked.gif	checked basket icon				
batchapply	batch keyword apply icon				
wn_browse.gif	icon representing browse action				
changepassword.gif	user change password and sets preference for icons				
changeuser.gif	changeuser icon				
wn_filemgr.gif	file manager icon				
wn_fmcopy.gif	file manager copy icon				
wn_fmmove.gif	file manager move icon				
wn_fmtrash.gif	file manager trash icon				

Table 3-4 Displaying icons

Name	Description	Classic	Buttons	Small buttons	House
wn_fmcancel.gif	file manager cancel icon				
wn_folder.gif	folder icon				
wn_high.gif	download high-res icon				
wn_home.gif	toplevel action icon				
wn_info.gif	imageinfo action icon				
wn_linkedfiles.gif	view linked files icon				
wn_low.gif	download FPO icon				
wn_nearline.gif	nearline icon				 (green)
wn_nextpage.gif	next page of multipage preview				
wn_offline.gif	offline icon				

Table 3-4 Displaying icons

Name	Description	Classic	Buttons	Small buttons	House
wn_order.gif	custom image order icon				
wm_prevpage.gif	previous page of multipage preview				
wn_promote.gif	make work-ing-version icon				
wn_qmark.gif	question mark icon				
wn_search.gif	search action icon				
wn_stream.gif	view file icon				
wn_upload.gif	upload action icon				
wn_versions.gif	show versions icon				
wn_wrench.gif	configuration icon				

JavaScript variables:

NOTE: Unlike other WebNative Styles, the *icons* action does not automatically print the following variables. They must be *statically initialized* in the *icons.js* file. Additionally, each image shown above must be present in the new icon Style. If not all are present, incorrect behavior should be expected.

- *var wn_icondirpath*

  The *wn_icondirpath* is a string representing the path where WebNative should look for icon files. This path must be specified relative to the top-level WebNative image directory, and must be followed with a trailing forward-slash. For example, to specify the path */webnativedoc/images/mybuttons*, one should initialize *wn_icondirpath* with the value *mybuttons/*.

- *var wn_iconwidth*

  The *wn_iconwidth* variable should be initialized with a string value specifying the width of the icons contained within the directory specified by *wn_icondirpath*.

- *var wn_iconheight*

  The *wn_iconheight* variable is similar to the *wn_iconwidth* variable, only it specifies the height of the icons rather than their width.

Accepted Form variables:
Not applicable.

Listing directories and volumes

On UNIX systems, */usr/etc/webnative/listdir*
On Windows systems, *C:\Program Files\Xinet\WebNative\Bin\listdir.exe*

The *listdir* CGI browses a directory on a WebNative volume, gathers information about its contents, and prints HTML which displays the contents. Additionally, it will provide information about files which have been backed up from that directory with the FlashNet archiving software. Essentially, it is a front-end for browsing WebNative file systems.

The *listdir* program also implements the *toplevel* action. It will perform *toplevel* if no path is supplied. If a path is supplied, it will perform *browse*. For backwards compatibility, the *searchengine* CGI also performs the *toplevel* action when called with no arguments (or the optional `-t`)

Command summary:

```
listdir [-q] [-f] [-t] [-c pagecount] [-s start]
[-u username] [-b username] [-nohead] [-nofoot] [-nosrc]
[-style style] [-i pathid] [-D] [-dir depth] [-cust c] [-N]
filename
```

The table below provides details about optional *listdir* arguments:

Table 3-5 listdir operation arguments and variables

Argument	Definition
`-q`	The `-q` argument specifies that listdir should operate in "quick" mode. In quick mode, support image information, including previews, FPOs, custom icons, etc., is not gathered for image files.
`-f`	If the `-f` argument is specified, the listdir CGI will print its support HTML rather than its frameset, provided the frameset exists. If the frameset exists and the `-f` *a*rgument is omitted, the frameset will be printed. If the frameset does not exist, the `-f` argument will have no effect.
`-t`	If the `-t` argument is the only specified, then *listdir* will perform the *toplevel* action.
`-c pagecount`	If the `-c` argument is specified, then *listdir* will only display `pagecount` *number* of files.
`-s start`	If the `-s` *ar*gument is given, then listdir will start listing files only after it has skipped `start` *number* of files.
`-u username`	The `-u` *a*rgument allows a user to re-login as a different user. In order to work, the `username` provided must be the same as the current user. Any other `username` will have no effect.
`-b username`	The `-b` *a*rgument forces a user to log in as the `username` specified. Any other username will result in an *"Unauthorized Access"* error.
`-nohead`	If `-nohead` is given, then the user's *header.html,* stored in his or her WebNative user directory, will not get printed.
`-nofoot`	If `-nofoot` is given, then the user's *trailer.html,* stored in his or her WebNative user directory, will not get printed.
`-nosrc`	If `-nosrc` is given, then *browse.html* will not be printed.
`-style style`	The *listdir* CGI can dynamically change *Styles b*y specifying the `-style` argument and the name of the *Style* to use.
-i `pathid`	With the database enabled, if the unique *path ID* `pathid` for a path is known before invoking *listdir,* it can be passed in with the `-i` argument. This eliminates the *path ID* lookup that would otherwise occur, were the argument not supplied.
-D	Instruct *listdir* to send WebNative Suite debugging information to Apache's error log.

Table 3-5 listdir operation arguments and variables

Argument	Definition
-dir depth	This variable instructs listdir to browse subdirectories beginning at the directory *depth*.
-cust c	The -cust flag is used to initialize the "custom" javascript with the variable value *c*.
-N	If present sets the *nonav* javascript variable to *true.*
filename	The name of the directory about which to gather information.
The following are JavaScript variables:	
files numfiles uinfo dir	Please see the "Some common JavaScript variables" on page 34 for information regarding these variables.
filecount	The filecount *v*ariable is similar to the numfiles variable, except it does not account for directories or hidden files.
pagecount	The pagecount variable reflects the -c pagecount value passed in. It represents how many files listdir actually returned.
startindex	The startindex variable reflects th*e* -s start value passed in. It represents how many files were skipped before listdir actually started printing file information.
endindex	The endindex variable is equal to zero if startindex plus *pagecount* is greater than the number of files in the directory. If not, then it is equal to startindex *plus* endindex.
arrayIndex	If the -dir argument was given upon invocation, then arrayIndex will be set to *1*.
custom	The custom variable will be equal to the number passed in via the -cust argument.
files[i].hasversions	If *true,* then versions of the file exist on the server.
files[i].numpreviews	If defined, indicates the number of small Web previews available for the given file.

Table 3-5 listdir operation arguments and variables

Argument	Definition
`dirpagecount`	The `dirpagecount` *variable reflects the* `-dc dircount` value passed in. It represents how many directories *listdir* actually returned.
`dirstartindex`	The `dirstartindex` variable reflects the `-ds` *`dirstart`* value passed in. It represents how many directories were skipped before *listdir a*ctually started printing directory information.
`filehints`	Array of objects describing files not included in the files array. Each object has a *"begin"* and *"end"* member variable, which describe a *"chunk"* of files defined by `filecount` and `startindex`.
`dirhints`	Array of objects describing directories not included in the *files* array. Each object has a *"begin"* and *"end"* member variable, which describe a *"chunk"* of directories defined by `dircount` and `dirstartindex`.
`dircount`	Total number of paths passed to the CGI for multi-directory searching.
`nonav`	If *true,* the style should not display the typical WebNative Suite navigation buttons like *Search, Upload*, and so on.
Accepted Form variables:	
	None

Corresponding HTML files

On UNIX systems:

- */usr/etc/webnative/styles/browse/*`style`*/browse.html*
- */usr/etc/webnative/styles/browse/*`style`*/frameset.html* (optional)

On Windows systems:

- *C:\Program Files\Xinet\WebNative\Admin\styles\browse*`style`*\browse.html*
- *C:\Program Files\Xinet\WebNative\Admin\styles\browse\style\frameset.html*

Searching

On UNIX systems, */usr/etc/webnative/searchengine*
On Windows systems, *C:\Program Files\Xinet\WebNative\Bin\searchengine.exe*

The *search* action is implemented via the *searchengine* CGI. The *searchengine* CGI will recursively traverse a WebNative-accessible directory and search for files which match user-specified criteria. It can also search through offline files.

Command Summary:

```
searchengine [-t] [-f] [-u] [-b] [-d 1|2|3] [-nohead]
[-nofoot] [-nosrc] [-style style] [noarch] [nokey] [noimg] [-D]
[-nodesc] [dir path [-dir path ...]] [-max int]
[-full] [allvols] [-sort] filename
```

The table below provides details about *searchengine* arguments:

Table 3-6 searchengine operation arguments

Argument	Definition
`-t`	If the `-t` argument is present, then searchengine will ignore all other options and perform the toplevel action. "toplevel operation arguments" on page 59 for details.
`-f`	If the `-f` argument is specified, the searchengine CGI will print its support HTML rather than its frameset, provided the frameset exists. If the frameset exists and the `-f` argument is omitted, the frameset will be printed. If the frameset does not exist, the `-f` argument will have no effect.
`-u`	The `-u` argument allows a user to re-login as a different user. In order to work, the `username` provided must be the same as the current user. Any other `username` will have no effect.
`-b`	The `-b` argument forces a user to log in as the `username` specified. Any other `username` will result in an "Unauthorized Access" error.
`-d 1|2|3`	The `-d` option is used to specify whether or not searchengine searches in online files only (`1`), offline files only (`2`), or both (`3`). If the selectedvol form variable is submitted to the CGI, it will override the value specified by the `-d` flag.
`-nohead`	If `-nohead` is given, then the user's header.html, stored in his or her WebNative user directory, will not be printed.
`-nofoot`	If `-nofoot` is given, then the user's trailer.html, stored in his or her WebNative user directory, will not be printed.
`-nosrc`	If `-nosrc` is given, then searchquery.js and searchdisplay.js will not be printed.
`-style style`	The searchengine CGI can dynamically change Styles by specifying the -style argument and the name of the Style to use.
`-noarch`	Instructs the searchengine CGI to suppress printing of archive information.

Table 3-6 searchengine operation arguments

Argument	Definition
`-nokey`	Instructs the searchengine CGI to suppress printing of keyword information.
`-noimg`	Instructs the searchengine CGI to suppress printing of preview information.
`-D`	Instructs the searchengine CGI to log WebNative Suite debugging information to Apache's error log.
`-nodesc`	Forces the search query to only search the *toplevel* volume
`-dir path [-dir path ...]`	Instructs the searchengine CGI to search multiple volumes.
`-max int`	Limit the number of query matches to `int` in two instances: • If it is less than the last value for max matches POSTed to the searchengine CGI *or* • If no value has been POSTed to the CGI, it is less than the default value of 12
`-full`	If a user has access to all volumes in the database, `-full` can be used in place of directory args to search the entire database. Users who do not have full access will only see results from volumes to which they have access.
`allvols`	Substituting the `allvols` token for a path has the same effect at the `-full` switch

Table 3-6 searchengine operation arguments

Argument	Definition
`-sort`	This flag causes the search results to be sorted by the WebNative Suite database; i.e., not according to Apple's standard sorting scheme. (Essentially the difference is that the old sort will count numbers as separate letters whereas the Apple sort will actually assess the "value" of the number; e.g., the Apple sort would put *file_088* after *file_1* because the numeric value of *088* is greater than *1*.) The `-sort` flag will also sort based on the full result set (instead of sorting within the subset of files that the query finds first). This may cause queries to become very slow in returning results if you have a large database. In SQL terms `-sort` causes the use of the following condition, *ORDER BY FileName* to be appended to the end of the query, instead of a simple *LIMIT XX* in the regular case. In order to use the `-sort` flag, the Style must be edited to include the flag where it sets the submit target to *searchengine*. The `-sort` argument has no effect on systems which are not running WebNative Suite database. Xinet introduced this argument solely for use with custom Styles. Styles that ship with. WebNative Suite do not use this sorting mechanism.
`filename`	The full path to the directory at which searches are rooted.

JavaScript variables

`dir`
`uinfo`
`keywords`
`files`
`numfiles`
`Numkeywords`
	Please see the "Some common JavaScript variables" on page 34 for information regarding the above variable.
	
	Note: The `Numkeywords` variable, while capitalized, has the same meaning as `numkeywords` on page 34.

Table 3-6 searchengine operation arguments

Argument	Definition

`Pathtype`
`Filetype`
`Filetypelogic`
`Maxmatches`
`Selectedvol`
`Datetime`[a]
`Datetype`
`Datewhence`
`Datelogic`
`Checkdate`[b]
`Exactmatch`
`Nocasematch`
`Comment`
`Commentlogic`
`Skipcount`
`Filename`
`Filenamelogic`
`Filenamehasany`
`Commenthasany`
`Pathsearchtype`
	Each of these JavaScript variables corresponds to a form variable, described below in the Form Variables subsection of this table. Each JavaScript variable contains the last value submitted to the searchengine CGI for the corresponding form variable. For example, if the form variable *pathtype* (notice the lower case) was submitted with the value *1*, then the JavaScript variable `Pathtype` will have the value *1*. These "mirror" variables are provided to enable the search interface to maintain its state across form submissions.

a. *Note:* corresponds to *dateyear, datemonth, dateday*
b. *Note:* corresponds to *dateon*

Table 3-6 searchengine operation arguments

Argument	Definition

`searchfield`	The *searchfield* variable, if not *null, zero*, or *undefined,* points to an array of objects representing how the *keyword* fields were searched during the last form submission. Each object is defined by the following fields:
	• `searchfield[`*i*`].type` — the database field's type
	• `searchfield[`*i*`].string` — the search string (corresponds to `dbsearchkeywordN`)
	• `searchfield[`*i*`].flag` — the type of search that was done (corresponds to `dbsearchflagN`)
	• `searchfield[`*i*`].casesense` — whether or not the search was case sensitive (corresponds to `dbsearchcaseN`)
	• `searchfield[`*i*`].logic` — how the search is combined with other results (corresponds to `dbsearchlogic`)
	• `searchfield[`*i*`].not` — whether or not the search was inverted (corresponds to `dbsearchnotN`)
	• `searchfield[`*i*`].multi` — whether or not a multi-word search was done (corresponds to `dbsearchmultiwordN`)
	• `searchfield[`*i*`].any` — whether or not a "has any" search was done (corresponds to `dbsearchhasanyN`)
	
	Example: If `dbsearchkeyword11` were submitted with the value *"cover.art"* and `dbsearchmultiword11` were submitted with the value *"1,"* then the eleventh object of the *searchfield* object would have the following values:
	`searchfield[11].string = "Cover Art"`
	`searchfield[11].multi = 1`
	
	Other fields would be set appropriately.

`searchall`	If `searchall` is not *null, zero,* or *false, th*en it points to an object describing the `dbsearchall` set of form variables. It is described by the same variables as a `searchfield[`*i*`]` object.
`morematches`	If `morematches` is *true, t*hen more matches for the previously submitted search criteria exist. These matches can be retrieved by re-submitting the same search criteria with an appropriate value for `skipcount`. Example: If 100 matches were available, but `skipcount` were *23* and `morematches` *12,* then `morematches` would be *true.*
`didsearch`	If `didsearch` is *true,* then the *searchengine C*GI performed a search of some kind. A "search" is defined as performing a new search, getting more results for a set of search criteria, or getting previous results for a set of search criteria.

Table 3-6 searchengine operation arguments

Argument	Definition
totalmatches	If you are running WebNative database, the totalmatches *v*ariable contains the total number of matches for the search criteria specified. If the display of results is limited, the variable contains the number of files actually displayed plus one. If you are not running the database, totalmatches will always = *0*.
Nodescend	Has a value of *true* if the -nodesc flag was given.
UseDB	If *non-zero,* the WebNative Suite database is enabled and will be used to perform searches.
FilenameFullText	If *non-zero,* then file names are Full-Text indexed in the WebNative Suite database.
CommentFullText	If *non-zero,* then finder comments are Full-Text indexed in the WebNative Suite database.
Allkeysfulltext	If *non-zero,* then all metadata values are Full-Text indexed in the WebNative Suite database.
volumes	An array of buildvol objects, described in "Entry point for WebNative — toplevel" on page 58.
numvols	An integer indicating the number of objects in the volumes array.
Accepted Form variables:	
dir	The dir *fo*rm variable specifies the directory within which the *searchengine* CGI should root its searches.
pathtype	The pathtype form variable specifies whether to include files, directories, or both in the search results (aka. File/Folders). Legal values include: 1 *(*include only files), 2 (include only directories), or 3 (include both).When used with other logic options, pathtype and *s*electedvol will always be tested first, regardless of the order in which the criteria have been submitted.
filetype	The filetype form variable specifies a specific file type, which all files must be in order to be included in the search results. Legal values for filetype include: 1 (for all file types), -1 (for all images), -2 (for all non-images), -3 (for all PDF files), -4 (for all QuarkXPress files), or a four-character Macintosh filetype string (like JPEG or GIFf*)*.

Table 3-6 searchengine operation arguments

Argument	Definition
`filetypelogic`	The `filetypelogic` form variable specifies whether the filetype described above must be matched in order for a file to be included in the search results. If it is set to `0` *then filetype must be matched to be included in the search results. If not, then a file will still be included if it matches other criteria.* (For those familiar with boolean searching, `0` corresponds to *AND-ing* the filetype to the search criteria and `1` corresponds to *OR-ing t*he filetype to the search criteria). Note: This variable only works with WebNative database enabled. In WebNative, the presence of a value for `filetype` implies that it must be matched in all files.
`filenamelogic`	This form variable is similar to `filetypelogic`.It specifies whether the file name must be matched in order for a file to be included in the search results. If set to *`0`* (the default value) an *`AND`* value will be used; if set to *`1`,* an *`OR`* value will be used.
`maxmatches`	The `maxmatches` variable instructs the CGI to return a maximum number of matches for the given query. Although more may be available, only *`maxmatches`* *will be printed. The rest can be retrieved by resubmitting the form with an appropriate* `skipcount`. Legal values are integers.
`selectedvol`	The `selectedvol` variable indicates whether files/folders in offline or online volumes are included in the search results (aka. Search In). Legal values include: *`1`* (online files only), *`2`* (offline files only), *`3`* (online and offline files). When used with other logic options, *s*`electedvol` *and* `pathtype` will always be tested first, regardless of the order in which the criteria have been submitted.
`dateon`	If the `dateon` variable is present and has a non-null value, then the searchengine CGI will perform date matching, which is driven by criteria described below.
`datetype`	The `datetype` *v*ariable specifies which date to perform date searching on. Legal values include: *`1`* (creation date), *`2`* (last modification date), and *`3`* (last access date).
`datewhence`	The `datewhence` variable describes whether to search before, on, or after the date specified. Legal values include: *`1`* (newer), *`2`* (older), and *`3`* (same).
`dateyear`	Specifies the year to match against (use four digits, as in *2001*)
`datemonth`	Specifies the month to match against (any whole number *0–11*)
`dateday`	Specifies the day to match against (any whole number *1–31*)
`datelogic`	Similar to `filetypelogic`, only with respect to date matching.

Table 3-6 searchengine operation arguments

Argument	Definition
`exactmatch`	If `exactmatch` is specified, then all criteria must be matched exactly (i.e. the string *foo* will only match *foo* and not things like *foobar* nor *so much foo*).
`comment`	*The* `comment` variable specifies a string with which to search Finder comments.
`commentlogic`	The `commentlogic` variable is similar to `filetypelogic`, only with respect to comment matching. If set to `0` (the default value) an `AND` value will be used; if set to `1`, an `OR` value will be used.
`nocasematch`	If the `nocasematch` variable is submitted, then case will be ignored when matching strings.
`skipcount`	The `skipcount` variable specifies how many matches the *searchengine* CGI should skip before printing results.
`pathsearchtype`	Indicates how to search filenames. With a value of *1*, the filename argument may match anywhere in the searched filename. With a value of *2*, the filename argument must match the beginning of the searched filename.
`filenamehasany`	With a value of *1*, the filename argument will be split into tokens and a Full-Text search will be performed on the searched filenames. A filename Full-Text index must be present on the server for this field to have any effect.
`actionsearch`	Submitting the `actionsearch` variable will cause the *searchengine* CGI to search the filesystem rooted at the path stored in the `dir` form variable, using the specified criteria. It will start printing the first matches it finds after skipping `skipcount` number of files and will print `maxmatches` number of matches.
`actionmore`	Submitting the `actionmore` variable has a similar effect as `actionsearch`, only the CGI will start printing after it has skipped `skipcount` *plus* `maxmatches` number of files.
`actionless`	Submitting the `actionmore` variable has a similar effect as `actionsearch`, only the CGI will start printing after it has skipped `skipcount` *minus* `maxmatches` number of files.
`actionclear`	Submitting the `actionclear` variable will reset all JavaScript variables to default values.

Table 3-6 searchengine operation arguments

Argument	Definition
The following Form variables only work with the WebNative Suite database. The letter `N` *following each is merely a placeholder. It should be replaced with the numerical field ID for the keyword to be searched.*	
`dbsearchflagN`	The `dbsearchflagN` determines the type search that is performed with the string stored in `dbsearchkeywordN`. Legal values are `0` *(*the field contents must contain the search string), `1` (the field contents must be exactly equal to the search string), `2` (the field contents must start with the search string), `3` (the field contents must end with the search string), and `4` (the search string is treated as a regular expression).

`dbsearchdate-yearN`
`dbsearchdate-monthN`
`dbsearchdate-dayN`
`dbsearchdate-hourN`
`dbsearchdate-nuteN`
`dbsearchda-tesecsN`
	If the database field is a date type, it can be searched by inserting values in the above variables, rather than using the `dbsearchkeyword`*N variable.*
	`dbsearchdateyearN` can be any 4-digit number
	`dbsearchdatemonthN` can be any integer *0–11*
	`dbsearchdatedayN` can be any integer *1–31*
	`dbsearchdatehourN` can be any integer *0–23*
	`dbsearchdatenuteN` can be any integer *0–59*
	`dbsearchdatesecsN` can be any intege*r 0–59*

`dbsearchkey-wordN`	The `dbsearchkeywordN` form variable contains the search string with which to search database field `N`.
`dbsearchlogic`*N*	The `dbsearchlogicN` form variable is similar to the `filetypel-ogic` variable, only with respect to `database field N`.
`dbsearchnot`*N*	If the `dbsearchnotN` variable is set to `1`, then the CGI will return everything whose keyword *N value* doesn't match the search string in `dbsearchkeywordN`. If it is `0` or is omitted, then it behaves normally.
`dbsearchmulti-word`*N*	This argument has been superseded. Use `dbsearchhasanyN`, instead.
`dbsearch-hasany`*N*	If `dbsearchhasanyN` *h*as a value of `1`, then the CGI will consider the presence of any word within the search string as a match. This means, for example, that files could contain *keyword1* or *keyword2* or *keyword3* ....(If it is `0` or omitted, then the CGI matches normally.) Continuing the example, if `dbsearchhasanyN` has a value of `2`, the files must contain *keyword1* and *keyword2* and *keyword3* .... and, thus returns fewer results.

Table 3-6 searchengine operation arguments

Argument	Definition
dbsearchhase-achN	When set to 1 provides the same thing as dbsearchhasany 2
filename-hasanyN	*The values for* N are the same as for dbsearchhasanyN. When set to 2, it provides the same as filenamehaseach 1.
commenthasanyN	*The values for* N are the same as for dbsearchhasanyN. When set to 2, it provides the same as commenthaseach 1.
dbsearchregexpN	If dbsearchregexpN has a value of 1, then the search string within dbsearchkeywordN may contain a full regular expression instead of merely a search string.
dbsearchcaseN	If dbsearchcaseN has a value of 1, then searches will be case sensitive. If it is 0 or is omitted, then searches will be case insensitive.
dbsearchall*	If one wants to search all database fields with the same criteria, then one should use the dbsearchall* pattern instead of the dbsearch*N pattern. For example, to search all database fields for the word *foo, o*ne would search using the dbsearchallkeyword variable with a value of foo.
dbsearchtypeN	Should contain the metadata type found in the javascript keyword object. This field is used to optimize searches for each data type.
dbsearchhasanyN	With a value of *1*, this field will instruct *searchengine* to perform a Full-Text search using the dbsearchkeywordN value. The metadata field being searched must have a Full-Text index in the WebNative Suite database for this field to have any effect.
dbsearchall-withfile	If dbsearchallwithfile is *non-zero, searchengine* will include filenames in its "search all" query.
dbsearchall-withcomment	If dbsearchallwithcomment is *non-zero, searchengine* will include file comments in its "search all" query.
custquery_X custquerylogic_X	Provides a search cause type that allows users to enter a custom SQL search clause using columns in the *"file"* (e.g., *FileName, Creator,* etc.) and *keyword1* table (e.g., *Field123,* etc.). Refer to "About custom queries" on page 58 for more information.

Corresponding HTML files:

- On UNIX systems
- */usr/etc/webnative/styles/search/style/searchquery.js*
- */usr/etc/webnative/styles/search/style/searchdisplay.js*
- */usr/etc/webnative/styles/search/style/frameset.html* (optional)

On Windows systems

- *C:\Program Files\Xinet\WebNative\Admin\styles\search\style\searchquery.js*
- *C:\Program Files\Xinet\WebNative\Admin\styles\search\style\searchdisplay.js*
- *C:\Program Files\Xinet\WebNative\Admin\styles\search\style\frameset.html* (optional)

About custom queries

WebNative Suite supports form variables that allow users to enter a custom SQL search clause using columns in the "file" (e.g., *FileName*, *Creator*, etc.) and *keyword1* table. (e.g., *Field123*, etc.). The only constraint is that the access to the data fields is limited naturally to the those to which the particular user has access. Custom clauses with illegal fields will simply be ignored. However, it is well within the user's power to add syntax errors to the clause which can cause the entire search to fail, so this should be used with caution.

An example of a custom clause is:

```
Field123 = "bbb" AND LENGTH(FileName) > 5
```

The variable, `custquery_X` represents the feature. Multiple values of this variable may be submitted to pass custom SQL conditions that will be included with the other built-in *search* conditions. (Where `X` is the index to the particular custom query, starting from `0`. The `X` number is basically the count of custom queries involved. Normally there would just be one, so you would only use `custquery_0` which is associated with `custquerylogic_0` and `custquerynot_0`. If a second one is added they would all share the suffix `_1`, and so on...The values that are submitted follow the same rules as the *dbsearch* `logic` and `not` values.)

The variable `custquerylogic_X` is also supported, providing the logical operation for how the whole condition is joined to the query. The `custquerynot_X` variable negates the condition (*e.g.*, `custquery_0 = "Field44 = \"foo\"", custquerylogic_0 = 1`).

Entry point for WebNative — *toplevel*

On UNIX systems: */usr/etc/webnative/listdir*
/usr/etc/webnative/searchengine

On Windows systems: *C:\Program Files\Xinet\WebNative\Bin\listdir.exe*
C:\Program Files\Xinet\WebNative\Bin\searchengine.exe

The *toplevel* action, which is not associated with a single, named CGI, is the "starting point" for most WebNative activities. It provides the user with a convenient, single entry-point into WebNative file systems.

Command summary:

```
listdir [-t] [-nohead] [-nofoot] [-nosrc]
```
or
```
searchengine [-t] [-nohead] [-nofoot] [-nosrc]
```

The table below gives details about *toplevel* arguments:

Table 3-7 toplevel operation arguments

Argument	Definition
`-t`	The `-t` flag, if given, will force the two CGIs listed above to perform the *toplev*el action. If the `-t` is omitted, and no other argument is given to the CGIs, the *toplevel* action will also be performed. If any other flags are given, the CGIs will attempt to perform their regular actions.
`-nohead`	If `-nohead` is given, then the user's *header.html*, stored in his or her WebNative user directory, will not be printed.
`-nofoot`	If `-nofoot` is given, then the user's *trailer.html,* stored in his or her WebNative user directory, will not be printed.
`-nosrc`	If `-nosrc` is given, then *toplevel.js* will not be printed.
JavaScript variables	
`uinfo`	Please see the file "Common JavaScript variables" on page 34 for a description.
`numvols`	The `numvols` variable contains the number of volumes available to the user for browsing and/or searching.
`volumes`	The `volumes` variable points to an array of objects describing the volumes available to the user. Each object is described as follows. Note: In the following descriptions, `i` is merely a place holder, and refers to any arbitrary object within the array. `volume[i].path` — the full path to the volume on the server `volume[i].name` — an HTML-friendly name for the volume, as defined by the WebNative administrator `volume[i].uinfo` — a `uinfo` object, as described in "Common JavaScript variables" on page 34, detailing the user's permissions for the given volume.
`UserInfo`	An object containing information about the user's WebNative account. It consists of the following fields: • `UserInfo.name` — the user's name • `UserInfor.issubadmin` — if *true*, the user has *subadmin* privileges. • `UserInfo.id` — the user's WebNative Suite database numerical ID.
`HasAllVols`	if *true,* the user has access to every WebNative Suite database-enabled volume. This is useful for generating "allvols" searches, which are optimized searches of every database-enabled volume.
Accepted Form Variables	
	None

Corresponding HTML Files:

On UNIX,
/usr/etc/webnative/styles/toplevel/`style`*/toplevel.js*

On Windows,
C:\Program Files\Xinet\WebNative\Admin\styles\toplevel`style`*\toplevel.js*

Getting detailed image info

On UNIX systems — */usr/etc/webnative/imageinfo*
On Windows systems — *C:\Program Files\Xinet\WebNative\Bin\imageinfo.exe*

The *imageinfo* CGI provides detailed information about any file on a WebNative mounted volume. Additionally, it can be used to edit a file's meta-information, provided the user has permission to do so.

Command summary:

```
imageinfo [-a action] [-b] [-f] [-nohead] [-nofoot]
[-nosrc] [-style style] [-nokeys] [-I] [-v] [-D] [-p]
[-onlykeys] [-uploadkeys] filename
```

The table below gives details about *imageinfo* arguments:

Table 3-8 imageinfo operation arguments

Argument	Definition
`-a` *action*	The imageinfo CGI is not limited to using the default HTML associated with it. In fact, any WebNative action can have its own *imageinfo* HTML file. Any action's *imageinfo* HTML file can be used by giving *imageinfo* the `-a` flag, followed by the name of the action. If `-a` is omitted, then the default *imageinfo* action is assumed.
`-b`	The `-b` flag informs the CGI that the *nobuttons* JavaScript variable should be set to *true*, rather than the default *false* .
`-f`	The `-f` flag informs the *imageinfo* CGI to print the default *imageinfo* HTML file rather than the *frameset* HTML file. If the *frameset* file does not exist, then this flag has no effect.
`-nohead`	If `-nohead` is given, then the user's *header.html*, stored in his or her WebNative user directory, will not get printed.
`-nofoot`	If `-nofoot` is given, then the user's *trailer.html*, stored in his or her WebNative user directory, will not be printed.
`-nosrc`	If `-nosrc` is given, then *imageinfo.html* will not be printed.

Table 3-8 imageinfo operation arguments

Argument	Definition
`-style` *style*	The *imageinfo* CGI can dynamically change Styles by specifying the `-style` argument and the name of the Style to use.
`-nokeys`	Instructs the *imageinfo* CGI to suppress printing of keyword information.
`-l`	Activates the display of linked-file information.
`-v`	Activates the display of version information.
`-D`	Instructs the *imageinfo* CGI to log WebNative Suite database-debugging information to Apache's error log.
`-p`	Instructs the *imageinfo* CGI to set the *browseparent* variable to *false.*
`-onlykeys`	If supplied, the *imageinfo* CGI will only display metadata-related JavaScipt
`-uploadkeys`	Similar to `-onlykeys`, except it only displays metadata-related JavaScript which has been approved for Uploader usage.
filename	The name of the file which the *imageinfo* CGI should gather and print information about.
JavaScript variables	

`file`
`keywords`
`uinfo`
	Please see "Some common JavaScript variables" on page 34 for descriptions of these variables.
	Note: The `file` variable has the same elements as `files[`*i*`]`, described in "Some common JavaScript variables" on page 34.

`usedb`	If *true,* file information comes from the WebNative Suite database, rather than the file system.
`Syncing`	If *true,* the volume the file resides on is currently being synced with the WebNative Suite database.
`xmpwritable`	If *true,* the file has XMP data that can be written by the WebNative Suite database
`xmppad`	Indicates the amount of data that can be written to the file's XMP packet.
`dtcomment`	The `dtcomment` variable contains a string representing the *Finder* comments associated with the file.

Table 3-8 imageinfo operation arguments

Argument	Definition
`editkeywords`	If `editkeywords` is *true*, then the user has permission to edit a file's database keywords, given those keywords are editable.
`nobuttons`	If `nobuttons` is *true,* then the user interface should not display buttons.
`errcode`	If an error occurred, this variable will have a non-zero integer value. If all processing completed successfully, then this will be zero.
`browseparent`	Indicates whether it is appropriate for Styles to display a browse icon for the current directory
`fhistory`	An array of objects describing the WebNative Suite database transaction history for the given file.
`fhistory[i].event`	The type of event that occurred.
`fhistory[i].time`	The UNIX time at which the event occurred.
`fhistory[i].user`	The WebNative *username* of the user which initiated the event.
`fhistory[i].ipaddr`	The IP address which the event request originated from.
`hasversions`	If *true,* there are versions of the given file stored on the server.
`versions`	An array of file objects describing the versions. This variable is only populated if the `-v` switch was given.
`versionpwd`	The directory which contains the file's version repository.
`versioncount`	Number of file objects in the version array.
`haslinks`	If *true*, the file is linked in QuarkXPress and/or InDesign files on a WebNative Suite database-enabled volume.
`links`	An array of file objects describing the QuarkXPress and/or InDesign files the image is used in. This variable is only populated if the `-l` switch was given.
`numlinks`	Number of files in the links array.

Table 3-8 imageinfo operation arguments

Argument	Definition
`naa`	If `naa` is not *false* or *null*, then it points to an object describing the NAA comments associated with the file. Note: Even if the `naa` object exists, not all of the following fields may be defined or have a value. One should check each field for existence before referencing it: `naa.Vers` — the NAA version the comments are stored as: `naa.keywords` — an array of keywords `naa.objname` — object name `naa.category` — category `naa.categories` — an array of supplemental categories `naa.date` – date `naa.byline` — by-line `naa.title` — by-line title `naa.city` — city `naa.state` — province/state `naa.country` — country `naa.instr` — instructions `naa.xmit_ref` — transmission reference `naa.headline` — headline `naa.credit` — credit `naa.source` — source `naa.copyright` — copyright `naa.caption` — caption/abstract `naa.writer` — writer/editor
Accepted Form variables	
`dir`	Specifies the full path to the file whose comments/keywords are to be changed. This field is optional if the file path is passed in as an argument.
`comment`	Specifies a value for the file's finder comments.
`submitcomment`	If `submitcomment` is defined, then *imageinfo* will attempt to change the files' comments to the string specified in the `comment` form variable.
`keyword`*`N`*	Specifies a value for the file's database field with id `N`. *Note:* `N` is merely a placeholder, and refers to a unique database field id. For example, to set the value for database field 2, one would submit form variable named `keyword2` with the desired value.

Corresponding HTML files:

On UNIX systems

- */usr/etc/webnative/styles/*`action/style/`*imageinfo.html*
- */usr/etc/webnative/styles/*`action/style/`*frameset.html* (optional)

but, typically:

- */usr/etc/webnative/styles/imageinfo/*`style/`*imageinfo.html*
- */usr/etc/webnative/styles/imageinfo/*`style/`*frameset.html* (optional)

On Windows systems

- *C:\Program Files\Xinet\WebNative\Admin\styles*`action\style\`*imageinfo.html*
- *C:\Program Files\Xinet\WebNative\Admin\styles*`action\style\`*frameset.html*

but typically:

- *C:\Program Files\Xinet\WebNative\Admin\styles\imageinfo.html*`style\`*imageinfo.html*
- *C:\Program Files\Xinet\WebNative\Admin\styles\imageinfo.html*`style\`*frameset.html*

Custom-ordering images

On UNIX systems — */usr/etc/webnative/imageorder*
On Windows systems — C:\Program Files\Xinet\WebNative\Bin\imageorder.exe

The *imageorder* program handles both the initial full display of an image's properties and the actual download of the custom-sized image. Additionally, the program prints support HTML responsible for presenting a user with an interface.

Command summary:

```
imageorder [-f] [-g [dhesz]] [-imagetox args] filename
```

For example:

To create a 96 DPI greyscale TIFF from the file */path/to/file,* one would use the following URL:

```
/webnative/imageorder?-gd+-xtif+-cgrey+-r96+/path/to/file
```

The table below gives details about *imageorder* arguments:

Table 3-9 imageorder operation arguments

Argument	Definition
`-f`	If the `-f` argument is specified, the *imageorder* CGI will print its support HTML rather than its *frameset*, provided the *frameset* exists. If the *frameset* exists and the `-f` argument is omitted, the *frameset* will be printed. If the *frameset* does not exist, the `-f` argument will have no effect.
`-g[dhesz]`	Convert and download the image `d` — Convert and download a compressed archive containing the image. `h` — Suppress printing of the HTTP header. Useful only in a non-Web environment. `e` — Suppress addition of three character file type extension to filename. `s` — Force imageorder to use StuffIt archive format for downloads. `z` — Force imageorder to use ZIP archive format for downloads. zz — Force imageorder to use AppleDouble ZIP archive format for downloads
`filename`	The *`filename`* argument specifies the name of the file to convert, scale, and/or crop.
-imagetox args	See *imagetox*(1) man page.
JavaScript variables	
`sourcewidth`	The `sourcewidth` variable corresponds to the original image's pixel width.
`sourceheight`	The `sourceheight` variable corresponds to the original image's pixel height.
`sourcedpi`	The `sourcedpi` variable corresponds to the original image's resolution, in dots per inch.
`filename`	The `filename` variable contains the full path to the file specified at invocation.
`htmlname`	The `htmlname` variable contains a printable, HTML-friendly version of the filename contained in the `filename` variable.
`hasclip`	If `hasclip` is true, then the image has a clipping path.
`hasmask`	If `hasmask` is true, then the image has a mask.

Table 3-9 imageorder operation arguments

Argument	Definition
`imagetype`	The `imagetype` variable contains the 4-character Macintosh file type for the file.
`ispixels`	If `ispixels` is true, then the file is pixel-oriented (like JPEG), rather than vector-oriented (like PDF).
`colorspace`	The `colorspace` variable contains the colorspace the image is in.
`numspots`	The `numspots` variable contains an integer describing how many spot colors the image has.
`spots`	If `numspots` is greater than zero, then the `spots` variable will point to an array of strings. Each member of that array is an object described as follows: `spots[`*i*`].name` — the name of the spot color `spots[`*i*`].cyan` — the cyan value of the spot color `spots[`*i*`].megenta` — the magenta value of the spot color `spots[`*i*`].yellow` — the yellow value of the spot color `spots[`*i*`].black` — the black value of the spot color
`previews`	A generic JavaScript object containing fields for each preview image available for the file. The fields are set like the `support_image` variables on page 37 but may also include a "tiff" filed describing an EPS' TIFF preview.
`hasiccprofile`	If *true,* then the given file has an embedded ICC profile.
`hasiccdefault`	If *true,* the file type has an ICC profile associated with it. If `hasiccprofile` is *true,* `hasiccdefault` will always be *false.*
`iccprofile`	Describes either the embedded or default ICC profile for the image, given either exists.
`renderingintent`	Describes the default rendering intent.
`iccdirectory`	Directory on the server where ICC profiles exist.
`ICCGreyfiles`	Array of filenames for all the ICC profiles on the server for the Grey colorspace.
`ICCLabfiles`	Array of filenames for all the ICC profiles on the server for the Lab colorspace.
`ICCRGBfiles`	Array of filenames for all the ICC profiles on the server for the RGB colorspace.

Table 3-9 imageorder operation arguments

Argument	Definition
`ICCCMYKfiles`	Array of filenames for all the ICC profiles on the server for the CMYK colorspace.
`ICCGreynames`	Array of names for the above Grey ICC profiles.
`ICCLabnames`	Array of names for the above Lab ICC profiles.
`ICCRGBnames`	Array of names for the above RGB ICC profiles.
`ICCCMYKnames`	Array of names for the above CMYK ICC profiles.
`usmfiles`	Array of filenames describing preset unsharp masks on the server.
`usmnames`	Array of name describing the above unsharp masks.
`numpages`	Indicates how man pages the document has.
`setstyle`	If the default Style was overridden with the `-style` argument, the new Style name will be reflected in `setstyle`.
Accepted Form variables	
	None

Corresponding HTML files:

On UNIX systems

- */usr/etc/webnative/styles/order/`style`/imageorder.html*
- */usr/etc/webnative/styles/order/`style`/frameset.html* (optional)

On Windows systems

- *C:\Program Files\Xinet\WebNative\Admin\styles\order\`style`\imageorder.html*
- *C:\Program Files\Xinet\WebNative\Admin\styles\order\`style`\frameset.html (optional)*

Extracting Web-ready previews of images

On UNIX systems — */usr/etc/webnative/getimage*
On Windows systems — *C:\Program Files\Xinet\WebNative\Bin\getimage.exe*

The *getimage* program provides a centralized access point for retrieving a high-res image and all support images associated with that high-res image. For example, it can be used to extract a preview image into a Web-ready form. Additionally, *getimage* will compress and archive in *StuffIt* format large-sized high-res and FPO images before sending them to the user.

Command Summary:

```
getimage [-s pagenumber] [-f fmt] [-D] [-E] [-id dbid]
[-d date] [-n] -web|eweb|small|high|fpo filename
```

The table below gives details about *getimage* arguments:

Table 3-10 getimage operation arguments

Argument	Definition
`-s pagenumber`	Given multiple previews for a single file have been stored in the WebNative Suite database, this argument can be used to retrieve those previews individually. The `pagenumber` argument indicates which preview to retrieve.
`-f fmt`	When sending high-res or FPO files, *getimage* will compress and archive the image as a *StuffIt* file before sending it. The `-f` flag can be used to specify an alternate compression/archive format. Supported formats include: `zip` — Windows-style *zip* file `uzip` — Uncompressed Windows-style *zip* file `maczip` — Mac-style *zip* file, which includes resource forks `unmaczip` — Uncompressed Mac-style *zip* file `sit` — *StuffIt* file.
`-D`	Instructs the *getimage* CGI to log WebNative Suite debugging information to Apache's error log.
`-E`	If image retrieval fails, the `-E` argument will cause *getimage* to download a file describing the error in place of the requested image.
`-d date`	Instructs *getimage* to retrieve a large Web preview from the large Web preview history table. The preview selected will be as close to `date` as possible, but will never be older. If no preview is available in the large Web history table, the current large Web preview will be used. The `date` argument is encoded as the number of seconds since january 1, 1970. This argument is only supported with the WebNative Suite database enabled.
`-n`	Informs *getimage* that, if no download format is explicitly specified, it may choose *maczip* as a format. If not supplied, getimage will only choose between *sit* and *zip*.
`-id dbid`	Instructs the *getimage* CGI to retrieve image data from the database, rather than the file system. `dbid` must be a valid WebNative Suite file ID number

Table 3-10 getimage operation arguments

Argument	Definition
`-web|eweb|small|high|fpo` *filename*	The getimage CGI can return the following types of images. • `web` — Web-friendly preview option will return the larger GIF or JPEG of the image, with the appropriate MIME header (Content-type: *image/gif* or Content-type: *image/jpeg*). This image will be up to 72 DPI, and can be quite large in the case of large images on a server with no limit on FPO sizes. This image should not be scaled to a square size, as it will change the aspect ratio. • `eweb` — identical to `web`, but forces the GGI to not wrap images in an HTML IMG tag. • `small` — get a small, Web-friendly preview with the appropriate MIME header (Content-type: *image/gif* or Content-type: *image/jpeg*). This image will be 112 pixels in the larger direction. This size image is appropriate for inlining in Web pages. This image should not be scaled to a square size, as it will change the aspect ratio. • `high` — get the original high-res images (compressed and archived). The data and resource fork will be placed together in a *StuffIt* file (Content-type: *application/x-stuffit*). If a Zip file is requested, the resource fork is omitted. For images, extraneous resources will be discarded. • `fpo` — get the image's FPO (compressed and archived). The data and resource fork will be placed together in a *StuffIt* file (Content-type: *application/x-stuffit or just the data in a Zip*). Extraneous resources will be discarded. This is similar to usage of the *FPO Exporter* application on the Mac. Reasonable Web browsers will automatically decode the image to a Mac file on the other end with the correct file name.
JavaScript Variables	
	None
Accepted Form variables	
	None

Getting icons

On UNIX systems, */usr/etc/webnative/webicon*
On Windows systems, *C:\Program Files\Xinet\WebNative\Bin\webicon.exe*

The *webicon* CGI is used to search WebNative's icon database or a file's resource fork and get a Web-friendly icon for a specified file. By default, the *webicon* CGI returns a 32x32 GIF (with a Content-type: *image/gif header*) for any file the user has permission to access. The *webicon* CGI first checks if the file has a custom icon. If so, the custom icon is returned. If not, the icon database on the server will be consulted, and the icon for the application that created the image will be returned. If there is no application icon for the image, a document icon with a `?` in the middle of it will be returned. The default icon will always be square, and always be exactly 32x32 pixels.

Command Summary:

```
webicon [-16] [-off] [-gifonly] [-TC hextypeandcreator]
[-T hextype] [-C hexcreator]] path
```

The table below gives details about *webicon* arguments:

Table 3-11 webicon operation arguments

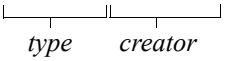
Argument	Definition
`-16`	A 16 pixel by 16 pixel icon is retrieved (as opposed to 32 by 32) if the -16 argument is given.
`-off`	If the `-off` flag is given, then the *webicon* CGI will draw the icon to specify "offline" status (i.e. greyed).
`-gifonly`	If `-gifonly` is specified, then the icon retrieved will always be a GIF.
`-TC hextypeandcreator`	The file's *type* and *creator* can be specified in two ways. The first is to give the `-TC` argument, followed by the hex representation of the file's *type* followed by the hex representation of the file's *creator*, with no space in between the *type* and *creator*. *Example:* A JPEG (*type* `JPEG`) created by Photoshop (*creator* `8BIM`) would have a hex *type/creator* string `4A5045473842494D`. 
`-T hextype` `-C hexcreator`	If `-TC` is not used to identify *type* and *creator*, then the `-T` argument can be given to specify the file's *type*. Again, the type is given as the hex representation. Additionally, the creator can be specified with the `-C` argument, which is followed by the hex representation of the file's *creator*.

Table 3-11 webicon operation arguments

Argument	Definition
path	If a file has a custom icon within its resource fork, the path to the file can be specified instead of the *type* and *creator*. The custom icon will be extracted from the resource fork instead of the icon database. *Note:* A file has a custom icon if the *.custicon* field of the file's corresponding file object (i.e. *file[i].custicon*) exists and has a value.
JavaScript variables	
	None
Accepted Form variables	
	None

Corresponding HTML files:

None

File management

On UNIX systems — */usr/etc/webnative/filemgr*
On Windows systems — *C:\Program Files\Xinet\WebNative\Bin\filemgr.exe*

The *filemgr* CGI is used to remotely administer WebNative file systems. Users with file management permissions can move, copy, rename, and delete any file or directory in a WebNative volume. Once the CGI is live and running in a browser window, however, interaction should proceed with form variables rather than command-line operations.

Command summary:

```
filemgr [-f] -action operator [path]
```

The table below shows details of *filemgr variables*:

Table 3-12 *filemgr* Form and operation variables

Variable	Definition
`-f`	If the `-f` flag is given, then *filemgr* will not print its *frameset* if it is present. Instead, it will print its HTML file. If no *frameset* file is specified, then the html file will always be displayed and `-f` will have no effect.
`-numvols`	The `numvols` variable contains the number of volumes available to the user for browsing and/or searching.

Table 3-12 *filemgr* Form and operation variables

Variable	Definition
`-volumes`	The `volumes` variable points to an array of objects describing the volumes available to the user. Each object is described as follows. Note: In the following descriptions, *i* is merely a place holder, and refers to any arbitrary object within the array. `volume[`*i*`].path` — the full path to the volume on the server `volume[`*i*`].name` — an HTML-friendly name for the volume, as defined by the WebNative administrator `volume[`*i*`].uinfo` — a `uinfo` object, as described in "Common JavaScript variables" on page 34, detailing the user's permissions for the given volume.
`-action`	Informs the *filemgr* CGI which action invoked it. The *browse* action, for example, would invoke it with `-browse`. Note: Any other action wishing to invoke the *filemgr* CGI must have a *filemgr.html* file within that action's current style directory, or it must invoke it as if it were the *browse* action.
`operator`	The *`operator`* argument may be any one of the following values: `nop`, `list`, `mkdir`, `rm`, `del`, or `premv`. The *`operator`* argument instructs the *filemgr* CGI to prepare itself for the operation associated with the argument. Each of these operators can be submitted to the CGI as values for the "`action` form variable described on page 74 • `nop` — no operation is performed, aside from printing the CGI's corresponding HTML files. • `list` — identical to `nop`. • `mkdir` — makes a directory. • `rm` — informs the CGI that the user would like to remove a file. • `del` — identical to `rm`, except the file is actually removed. • `premv` — informs the CGI that the user would like to move a file/directory. • `mv` — performs a move using form variables • `updir` — instructs the *filemgr* CGI to create a new diretory for files to be uploaded. Similar to *mkdir*, only the user does not need general file management permission to use it, only *upload* permission. • `promote` — promotes a version of a given file to be the current working version. The old current working version is saved as a new version beforehand. The *WebNative Administration Guide* provides more information about WebNative Version Management.

Table 3-12 *filemgr* Form and operation variables

Variable	Definition
`path`	The working directory for *filemgr*. This may be specified on the command line or may be passed as a form variable. The form variable takes precedence. • `precp` — similar to `premv`, except the file will be copied instead of moved. • `cp` — similar to `mv`, except the file will be copied instead of moved.
JavaScript Variables	

`files`
`numfiles`
`dir`
`uinfo`
	Please see "Some common JavaScript variables" on page 34 for information about these variables. All are initialized relative to the root directory passed in at invocation.

`volumes`	
`numvols`	
`action`	The `action` variable contains a string mirroring the *`operator`* specified upon invocation. The user interface should draw itself appropriately.
`filename`	The `filename` variable is only set after a `premv` or `precp` operation has been submitted via `POST`. It contains the full path to the file on the server upon which *filemgr* is to act. This path is specified in the `filename` form variable in the form submitted during the `precp` or `premv` operation.
`htmlfile`	This variable contains a printable, Macintosh-formatted version of the path stored in the `filename` variable.
`newname`	The `newname` variable is only set after a `mv` or `cp` operation has been submitted via `POST`, or to cause a directory to be made via *mkdir*. It contains the new name given to the moved or copied file.
`msg`	If an error occurred, a message explaining it will be stored here.
`errcode`	If an error occurred, the `errcode` variable will contain a positive integer. If `errcode` is negative, *filemgr* is just in the information-gathering stage.

Table 3-12 *filemgr* Form and operation variables

Variable	Definition
`rmcount`	If the `rm` operation is specified in a form submission, and the filename specified upon submission was a directory, this variable will contain the number of entries within the directory.
Accepted Form variables	
`filename`	The `filename` form variable should contain the full path of the file to be operated on.
`newname`	If a file is to be moved or copied, the `newname` form variable should contain the new name for the file. The same name as the `filename` can be used here if the file is being moved or copied to another directory.
`dir`	If a file is to be moved or copied, the `dir` form variable should contain the full path to the final directory where the moved or copied file is to reside. For any other operation, it should contain the full path to the directory when the file specified in `filename` exists.
`action`	The `action` form variable should contain one of the operators specified in "Some common JavaScript variables" on page 34.

`action_premv`
`action_precp`
`action_mv`
`action_cp`
`action_del`
	The `action_*` variables should have no value. Their presence is equivalent to setting the `action` form variable to each's respective suffix. While many `action_*` elements may be present in a form, only the last will be used in processing.

`overwrite`	If not defined *filemgr* or the value is *0*, *filemgr* does not overwrite files. If defined and value is *1 filemgr* overwrites files on copy and move.

Corresponding HTML files

On UNIX systems

/usr/etc/webnative/styles/`action`*/*`style`*/filemgr.html*

Typically, though...

/usr/etc/webnative/styles/browse/`style`*/filemgr.html*

On Windows systems

C:\Program Files\Xinet\WebNative\Admin\styles`action`**`style`*\filemgr.html*

Typically, though...

C:\Program Files\Xinet\WebNative\Admin\styles\browse`style`*\filemgr.html*

Previewing archived files

On UNIX systems — */usr/etc/webnative/archivepict*
On Windows systems — C:\Program Files\Xinet\WebNative\Bin\archivepict.exe

The *archivepict* CGI searches for a preview of an archived file. If it finds one, it will send back a Web-friendly version of it.

Command Summary:

```
archivepict [-o offset -l length] volumename number path
```

The table below shows details of *archivepict variables*:

Table 3-13 archivepict flags and variables

Variable	Definition
`-o offset -l length`	If a specific preview is desired, an `offset` and `length` into Flashnet's preview storage can be specified as the first two arguments. These flags should not be used unless the offset and length values were generated by a WebNative CGI.
`volumename`	The name of the volume the archived picture resides on.
`number`	More than one backup for a specific file may exist at one time. The `number` variable refers to the *n*th instance of the file's backup for which a preview is to be retrieved.
`path`	The `path` to the file for which a preview is requested
JavaScript variables	None
Accepted Form variables	None

Corresponding HTML files: None

Determining plug-in availability

On UNIX systems — */usr/etc/webnative/basketbuttons*
On Windows systems — *C:\Program Files\Xinet\WebNative\Bin\basketbuttons.exe*

The *basketbutton* CGI collects information about which plugins are available to a user. In most cases, *basketbutton* will never be invoked by anything other than *basketcontrol*.

The table below shows details of *basketbutton variables*:

Table 3-14 basketbuttons flags and variables

Variable	Definition
Arguments	
	None
JavaScript variables	

`basketbtns`	Contains an array of objects describing the plugins available to the user. Each object within this array corresponds to one plugin, which are described by the following characteristics.
	Note: In the following descriptions, `i` refers to any arbitrary object within the array.
	`basketbtns[`*i*`].path` — contains the path portion of the URL for the plugin
	`basketbtns[`*i*`].icon` — contains the path to the plugin's icon on the server
	`basketbtns[`*i*`].name` — contains a user-friendly name for the plugin
	`basketbtns[`*i*`].confirm` — if the plugin associated with this object does not handle archived images, this flag will be set to true.
	Example: The *batchorder* plug-in has the following entry in the `basketbtns` array:
	`basketbtns[i].path = '/webnative/plugins/batchorder.dir/batchorder'`
	`basketbtns[i].icon = 'wn_order.gif'`
	`basketbtns[i].name = 'Batch Image Order'`
	`basketbtns[i].confirm = 'true'`

`basketfile`	Contains the path to the user's basket file on the server. Most plugins require this to be passed in as an argument.
Accepted Form variables	
	None

Corresponding HTML Files:

On UNIX systems:
/usr/etc/webnative/styles/basket/`style`*/buttons.html*

On Windows systems:
C:\Program Files\Xinet\WebNative\Admin\styles\basket`style`*\buttons.html*

Gaining access to a user's basket

On Unix systems — /usr/etc/webnative/basketcontrol
On Windows systems — *C:\Program Files\Xinet\WebNative\Bin\basketcontrol.exe*

The *basketcontrol* CGI is responsible for displaying *control.html*, which invokes other basket CGIs with appropriate arguments. It is the primary entry point for accessing and displaying a user's WebNative basket.

Command summary:

```
basketcontrol [-n|-a|-r|-p] [path [...]]
```

The table below shows details of *basketcontrol arguments and variables*:

Table 3-15 basketcontrol arguments and variables

Variable	Definition
	Arguments
`-n|-a|-r|-p` *path*	An "operation" may be specified as an argument to *basketcontrol*. The `noop` operation, meaning do nothing but display text, is specified by the `-n` flag. Adding a file or group of files to the basket is specified by giving *basketcontrol* the `-a` flag. Removing a file or group of files is specified by the `-r` flag. If the `-p` flag is given, the *basketcontrol* CGI will proxy add/remove requests to the *showbasket* CGI rather than handling them itself. Typically, this is undesirable. Only one of the flags may be specified. If none of them is specified, then an operation of `noop` is assumed. Additionally, *basketcontrol* is not directly responsible for executing these operations. It merely reports them to other CGIs which do the real work associated with the flags. If the `-a` or `-r` flag is specified, a list of paths to files on the server must be specified.
JavaScript variables	
`controluser`	This variable corresponds to the current user's WebNative username.
`add`	If `add` is true, then the `-n` argument was given to *basketconrol.* All interested CGIs, particularly *showbasket*, should be notified via appropriate flags.
`remove`	If `remove` is *true*, then the `-r` argument was given to *basketconrol*. All interested CGIs should be notified via appropriate flags.

Table 3-15 basketcontrol arguments and variables

Variable	Definition
`noaction`	If `noaction` is true, then the `-n` argument was given to *basketcontrol*. Again, all interested CGIs should be notified.
`controlpath`	If `-r` or `-a` were specified upon invocation, the `controlpath` variable contains any path information also passed in. If the `-n` flag or no flag was specified, then `controlpath` is set to null.
`controlpath`	If the `-p` flag was given, this variable will contain the path list which should be passed to the *showbasket* CGI.
Accepted Form variables	
	None

Corresponding HTML files

On UNIX systems:
/usr/etc/webnative/styles/basket/`style`*/control.html*

On Windows systems:
C:\Program Files\Xinet\WebNative\Admin\styles\basket`style`*\control.html*

Generating a rectangular mask

On UNIX Systems — */usr/etc/webnative/cropimage*
On Windows Systems — *C:\Program Files\Xinet\WebNative\Bin\cropimage.exe*

The *cropimage* CGI is used in conjunction with *imageorder*. Its primary purpose is to run a separate cropping frame for image ordering. It is also used to generate a rectangular mask which illustrates how cropping will occur based on the user's input. It can be used, however, in any instance where a rectangular mask is needed. For example, the "preview" order Style uses *cropimage* to create a dynamic "bubble" for the slider.

Command Summary:

```
cropimage [-fname] [-c]
[height,width,crop-x,crop-y,crop-width,crop-height]
```

The table below shows details of *cropimage arguments and variables*:

Table 3-16 cropimage arguments and variables

Variable	Definition
Note: If no arguments are present, then cropimage will print the cropimage.html file.	
`-fname`	If `-f` is specified, then *cropimage* will print the file called *`name`* from the *imageorder style* directory
`-c height,width,crop-x, crop-y,crop-width, crop-height`	If the `-c` flag is specified, then cropimage will send the image mask specified. The image mask specification consists of a comma-delimited list of six integers. The first two integers correspond to the height and width of the entire mask. Then next two describe the position of the top-left corner of the non-masked area (coordinate numbering is done from top to bottom and from left to right). The next two describe the width and height of the non-masked area.
JavaScript variables Note: The following variables are output every time cropimage prints html, regardless which frame it is printing.	
`origwidth`	Corresponds to the *`width`* argument passed in upon invocation.
`origheight`	Corresponds to the *`height`* argument passed in upon invocation.
`dispwidth`	Corresponds to the *`width`* argument passed in upon invocation. Will never be less than 0, however, regardless of `origwidth`.
`dispheight`	Corresponds to the *`height`* argument passed in upon invocation. Will never be less than 0, however, regardless of `origheight`.
`icropx`	Corresponds to the *`crop-x`* argument passed in upon invocation.
`icropy`	Corresponds to the *`crop-y`* argument passed in upon invocation.
`icropw`	Corresponds to the *`crop-width`* argument passed in upon invocation.
`icroph`	Corresponds to the *`crop-height`* argument passed in upon invocation.
`filename`	Full path to the given file on the server.
`htmlname`	Displayable filename for the given file.
`origres`	Indicates the file's resolution, in dots-per-inch.

Table 3-16 cropimage arguments and variables

Variable	Definition
`setstyle`	If the default style was overridden with the `-style` argument, the new style name will be reflected here.
Accepted Form variables	
	None

Corresponding HTML files:

On UNIX systems

- */usr/etc/webnative/styles/order/`style`/cropimage.html* — the primary HTML file, responsible for opening/filling any other frames
- */usr/etc/webnative/styles/order/`style`/cbframe.html* — buffer frame (optional)
- */usr/etc/webnative/styles/order/`style`/ccframe.html* — control frame (optional)
- */usr/etc/webnative/styles/order/`style`/cdframe.html* — display frame (optional)
- */usr/etc/webnative/styles/order/`style`/cnframe.html* — blank frame (optional)

On Windows systems

- *C:\Program Files\Xinet\WebNative\Admin\styles\order\`style`\cropimage.html*— the primary HTML file, responsible for opening/filling any other frames
- *C:\Program Files\Xinet\WebNative\Admin\styles\order\`style`\cbframe.html* — buffer frame (optional)
- *C:\Program Files\Xinet\WebNative\Admin\styles\order\`style`\ccframe.html*— control frame (optional)
- *C:\Program Files\Xinet\WebNative\Admin\styles\order\`style`\cdframe.html*— display frame (optional)
- *C:\Program Files\Xinet\WebNative\Admin\styles\order\`style`\cnframe.html* — blank frame (optional)

Showing QuarkXPress, PDF and InDesign file previews

On UNIX systems — */usr/etc/webnative/quarkview*
On Windows systems — C:\Program Files\Xinet\WebNative\Bin\quarkview.exe

The *mview* CGI (formerly known as *quarkview*[1]) allows a user to view spreads and linked image information for any QuarkXPress file on the server which was saved with the

[1] The *quarkview* CGI remains in the distribution for backwards compatibility only. Use *mview* for all new development.

WebNative XT enabled or any InDesign document that was saved with the WebNative ID plug-in.

Command Summary:

```
mview [-f] [-D] [-d dlev] [-sync] [-fidfileid]
[-volflagsflags] [-sint] [-lint] [-p pagenumber] [-sspreadnumber |
-lspreadnumber] filename
```

The table below shows details of *mview* arguments and variables:

Table 3-17 mview arguments and variables

Variable	Definition
`-f`	If the `-f` argument is specified, the *mview* CGI will print its support HTML rather than its *frameset*, provided the *frameset* exists. If the *frameset* exists and the `-f` argument is omitted, the *frameset* will be printed. If the *frameset* does not exist, the `-f` argument will have no effect.
`-D`	Instruct *mview* to send WebNative Suite debug information to Apache's error log.
`-d dlev`	Instruct *mview* to send WebNative Suite debugging information to Apache's error log, using a custom debug level `dlev`.
`-sync`	Puts the *mview* CGI in database *sync* mode. When given, it will synchronize the given QuarkXPress, InDesign document, or PDF with the WebNative Suite database. Note: In database *sync* mode, the *mview* CGI does not output *JavaScript* or HTML.
`-fidfileid`	Used in *sync* mode to supply the CGI with the QuarkXPress, InDesign or PDF document's WebNative database ID number. Has performance benefits.
`-volflagsflags`	Used by mview to determine which previews are being stored for the volume in which the file exists.
`-sint`	Used to pull the small spread preview numbered `int` from the database.
-l`int`	Used to pull the large spread preview numbered `int` from the database.
`filename`	The full path on the server to the QuarkXPress or InDesign document to be examined.
`-p pagenumber`	Display the supplied page number on page load rather than the first page.

Table 3-17 mview arguments and variables

Variable	Definition
`-l`*`spreadnumber`*	The `-l` option requires a spread number. It will send the large Web preview for the file given with *`filename`* to standard out. If the preview is stored in the WebNative Suite database, that preview will be used. Otherwise, the previews will come from the file system, and therefore will not be provided if the file has been archived.
`-s`*`spreadnumber`*	The `-s` option requires a spread number. It will send the small Web preview for the file given with *`filename`* to standard out. If the preview is stored in the WebNative Suite database, that preview will be used. Otherwise, the previews will come from the file system, and therefore will not be provided if the file has been archived.
JavaScript variables	
`uinfo` `path`	Please see "Some common JavaScript variables" on page 34 for information regarding these variables.
`newvers`	If `newvers` is non-zero, then the QuarkXPress/InDesign document was saved with a new version of the WebNative XT/WebNative ID which is incompatible with this version of WebNative.
`image`	The `image` variable points to an array of objects describing all the images used within the document. Each image is described by the following characteristics. Note: In the following descriptions, `i` refers to any arbitrary object within the array. • `image[`*i*`].name` — the name of the image • `image[`*i*`].top` — the *y* value of the top of the image (with respect to the spread) • `image[`*i*`].bottom` — the *y* value of the bottom of the image (with respect to the spread) • `image[`*i*`].left` — the *x* value of the left side of the image (with respect to the spread) • `image[`*i*`].right` — the *x* value of the right side of the image (with respect to the spread) • `image[`*i*`].spread` — the spread on which the image appears.

Table 3-17 mview arguments and variables

Variable	Definition

	• `image[`*i*`].status` — the image status
	• Example: A 100 pixel by 500 pixel image in the extreme top-left corner of the 4th spread of a 6-spread document would have the following values:
	`image[`*i*`].top = 0`
	`image[`*i*`].bottom = 500`
	`image[`*i*`].left = 0`
	`image[`*i*`].right = 100`
	`image[`*i*`].spread = 4`
	All other fields would be set accordingly.
	• `image[`*i*`].volume` — the name of the volume the image exists on
	• `image[`*i*`].file` — if the *quarkview* CGI can find the file corresponding to the image, its information is printed here (as described in "Some common JavaScript variables" on page 34)

`numimage`	The `numimage` variable describes the number of images within the image array.
`smallspread` `largespread`	Both the `smallspread` and the `largespread` variables point to an array of objects. Each object in these two array corresponds to a spread preview within the QuarkXPress or InDesign document. Both arrays contain information for every spread within the document. The only difference between the previews is their size. The previews represented in `smallspread` are suitable for use as thumbnails, while those in `largespread` are the same size as the original document. Each spread is described by the following information. Note: Both `smallspread` and `largespread` contain the same fields. So only `smallspread` is detailed below. Again, `i` is used as a reference to any given object in the array. • `smallspread[`*i*`].top` — the top of the spread with respect to the document • `smallspread[`*i*`].bottom` — the bottom of the spread with respect to the document • `smallspread[`*i*`].left` — the left of the spread with respect to the document • `smallspread[`*i*`].right` — the right of the spread with respect to the document • `smallspread[`*i*`].size` — the size of the spread in bytes • `smallspread[`*i*`].offset` — the offset of the spread in the document

Table 3-17 mview arguments and variables

Variable	Definition
`numspread`	The `numspread` variable describes the number of spreads within both the `largespread` and `smallspread` arrays.
`docfileid`	The file's WebNative Suite file ID number.
`resolution`	The document's resolution, in dots-per-inch.
`offline`	If *true*, the file is offline
`isimage`	If *true,* the file is classified as an image by the WebNative Suite server. Certain non-image files can be viewed with *mview,* like movies and layout documents.
`docslugoutofdate`	If true, the document was saved without the Xinet plug-in enabled.
Accepted Form variables	
	None

Corresponding HTML files:

On UNIX systems

- */usr/etc/webnative/styles/quarkview/*`style`*/view.html*
- */usr/etc/webnative/styles/quarkview/*`style`*/frameset.html* (optional)

On Windows systems

- *C:\Program Files\Xinet\WebNative\Admin\styles\quarkview*`style`*\view.html*
- *C:\Program Files\Xinet\WebNative\Admin\styles\quarkview*`style`*\frameset.html*

Information about basket contents

On UNIX systems — */usr/etc/webnative/showbasket*
On Windows systems — C:\Program Files\Xinet\WebNative\Bin\showbasket.exe

The *showbasket* CGI is responsible for gathering information about and displaying the contents of the user's WebNative basket. Additionally, it is responsible for maintaining the contents of that basket.

Command summary:

```
showbasket [-n|-a|-r|-c] [-t string] [-D] [-s] [path [-f path]]
```

The table below shows details of *showbasket arguments* and *variables*:

Table 3-18 showbasket arguments and variables

Variable	Definition
`-n|-a|-r|-c`	An "operation" may be specified as an argument to *basketcontrol.* The "noop" operation, meaning "do nothing but print HTML," is specified by the `-n` flag. Adding a file or group of files to the basket is specified by giving *basketcontrol* the `-a` flag. Removing a file or group of files is specified by the `-r` flag. Clearing the basket of all files is specified by the `-c` flag. Only one of the flags may be specified. If none of them is specified, then an operation of "noop" is assumed.
`-t` *`string`*	In order to prevent a user's browser from caching the documents generated by *showbasket,* the `-t` flag can be used to create a unique URL each time the CGI is invoked. In order to work correctly, it must be followed by a unique string each time the CGI is invoked.
`-D`	Instruct *showbasket* to send WebNative Suite debugging information to Apache's error log.
`-s`	If supplied, basket contents will not be sorted alphabetically; rather, the contents will appear as they are ordered in the user's basket file.
`path` `[...]`	If the `-a` or `-r` flag is specified, a list of paths to files on the server must be specified, separated by `-f` arguments. These files will be added or removed.
JavaScript variables	

`uinfo`
`files`
`numfiles`
	Please see "Some common JavaScript variables" on page 34 for information about these variables.

`isIE`	If the `isIE` variable is set to true, then the user's browser is Internet Explorer. Some versions of Internet Explorer cache documents with the same URL, which causes problems with the CGI must invoke itself. If this variable is true, it is recommended to use the `-t` flag in any subsequent invocations.
`controlpath`	Contains the list of paths showbasket acted upon.

Table 3-18 showbasket arguments and variables

Variable	Definition
`files[`*i*`].uinfo`	Each element in the `files` array has a `uinfo` object associated with it, as described in "Some common JavaScript variables" on page 34. This `uinfo` object describes the user's permissions for the volume on which that file resides.
Accepted Form variables	
	None

Corresponding HTML files:

UNIX systems
/usr/etc/webnative/styles/basket/`style`/showbasket.html

Windows systems
C:\Program Files\Xinet\WebNative\Admin\styles\basket\`style`\showbasket.html

Streaming files to a browser

On Unix systems, */usr/etc/webnative/streamfile*
On Windows systems, C:\Program Files\Xinet\WebNative\Bin\streamfile.exe

The *streamfile* CGI is used to stream a file of arbitrary type to the user's Web-browser.

Command summary:

```
streamfile [-a][-cd] mimetype filename
```

The table below shows details of *streamfile arguments and variables*:

Table 3-19 streamfile arguments and variables

Variable	Definition
`-a`	If the `-a` argument is given, then the *streamfile* CGI will attach the file specified as a MIME attachment, rather than have it comprise the body of the message.
`-cd`	Causes streamfile to use the Content-Disposition HTTP header to name the file, instead of Content-Type. This increases compatibility with Internet Explorer on Windows.
`mimetype`	The MIME-type of file to be streamed. This can also be the value `-pdf` for PDF files, or `-html` for HTML files.
`filename`	The name of the file to be streamed.

Table 3-19 streamfile arguments and variables

Variable	Definition
JavaScript variables	
	None
Accepted Form variables	
	None

Corresponding HTML files:

None.

Sending files to a WebNative server

On UNIX systems, */usr/etc/webnative/upload*
On Windows systems, C:\Program Files\Xinet\WebNative\Bin\upload.exe

The *upload* CGI is used to send files to the WebNative server. A file may be encoded before sending, using a number of Macintosh "file-flattening" encodings, or it may be transmitted as merely the data segment. Typically, this is all handled with the `<INPUT TYPE=file>` HTML form widget. Hence, casual developers need not concern themselves with the actual data transmission. Additionally, the upload CGI can perform complex, script-based post-upload operations as well as generate detailed reports describing the upload.

Command summary:

```
upload [-f] [-r] [-c] [-b] [-nohead] [-nofoot]
[-style style] path
```

The table below shows details of *upload arguments and variables*:

Table 3-20 upload arguments and variables

Variable	Definition
`-f`	If the `-f` argument is specified, the *upload* CGI will print its support HTML rather than its *frameset*, provided the *frameset* exists. If the *frameset* exists and the `-f` argument is omitted, the *frameset* will be printed. If the *frameset* does not exist, the `-f` argument will have no effect.
`-r`	If the `-r` flag is given, then the *upload* CGI will ignore all Finder comments associated with the file, regardless of whether they were transmitted.
`-c`	If the `-c` argument is given, then the file specified in *path* will only have its *Desktop* and *Finder* information updated.

Table 3-20 upload arguments and variables

Variable	Definition
`-b`	If the `-b` argument is given, then the file specified in *`path`* will be decoded on the server using *kunarc*.
`-nohead`	If `-nohead` is given, then the user's header.html, stored in his or her WebNative user directory, will not be printed.
`-nofoot`	If `-nofoot` is given, then the user's trailer.html, stored in his or her WebNative user directory, will not be printed.
`-style` *`style`*	The upload CGI can dynamically change Styles by specifying the `-style` argument and the name of the Style to use.
`path`	The directory in which to upload files or, if the `-c` or `-b` argument is given, the file to act upon.
JavaScript variables	
`uinfo`	Please see "Some common JavaScript variables" on page 34 for a description of the `uinfo` variable.
`dir`	The `dir` variable contains the full path passed into the CGI upon invocation.
`htmldir`	The `htmldir` variable contains a Macintosh-formatted version of the path specified in the `dir` variable.
`dirobj`	identical to `dir` object in "Some common JavaScript variables" on page 34.
Note: If an upload report was generated, it will contain all of the variables described above, as well as the following variables.	
`numfiles`	The number of files uploaded to the server.
`uploads`	The `uploads` variable points to an array of objects representing each file uploaded to the server. Each object is described below. Note: In the following descriptions, `i` refers to any arbitrary object within the array. • `uploads[`*i*`].pathname` — the path to the file on the server • `uploads[`*i*`].href` — a pre-prepared URL for the file • `uploads[`*i*`].filename` — an HTML-friendly name for the file • `uploads[`*i*`].bytes` — the size of the file, measured in bytes • `uploads[`*i*`].comments` — the comments specified for the file, if any
`REMOTE_USER`	The username of the user who uploaded the files
`REMOTE_ADDR`	The IP address of the user who uploaded the files

Table 3-20 upload arguments and variables

Variable	Definition
DOWNLOADTIME	The time the upload occurred.
Accepted Form variables	
comments	If desired, the comments form variable can contain new Finder comments for the file being uploaded. Any comments in the comment form variable will override those already in the encoded file.
creation_date	Can be used to overwrite a file's creation date or to give a creation date to a created folder. Date must be in AppleShare format.
dbkeyword*XX*	If supplied, the upload CGI will attempt to apply the value of this field to the newly-uploaded file. For the purposes of this documentation, *XX* represent the keyword ID for the metadata field to be modified.
dir	The dir form variable should contain the name of the directory in which the uploaded files will be stored.
dir_end	Indicates that an application has finished uploading a directory and that trigger processing should be engaged.
dir_start	Indicates that an application is beginning to upload a directory and that trigger processing should be suspended until notified.
filedata	The filedata form variable should contain an encoded version (BinHex or AppleSingle encoding, for example) of the file or the file's data segment. Note: The filedata form variable may be enumerated (filedata1, filedata2, etc.), allowing multiple files to be uploaded.
modification_date	Can be used to overwrite a file's modification date or to give a modification date to a created folder. Date must be in AppleShare format.
newdir	If present, will create a new directory with a name equal to its value.
overwrite	With a value of 1 or true, will cause the upload CGI to overwrite existing files when file-name conflicts arise. Omitting it or giving it a value other than 1 or true will engage default behavior, which is to append a numerical suffix to the uploaded file's name.

Table 3-20 upload arguments and variables

Variable	Definition
`reportname`	If `reportname` is submitted and has a value, then the upload CGI will generate a report describing the upload and place it in the same directory as the uploaded file. This report will have a name containing the value of this form variable, followed by a unique id and the *.html* extension.

Corresponding HTML files:

On UNIX systems
/usr/etc/webnative/styles/upload/`style`*/upload.html*
/usr/etc/webnative/styles/upload/`style`*/frameset.html* (optional)

On Windows systems
C:\Program Files\Xinet\WebNative\Admin\styles\upload`style`*\upload.html*
C:\Program Files\Xinet\WebNative\Admin\styles\upload`style`*\frameset.html* (optional)

The *upload.script* file

If present the *upload* CGI will execute the *upload.script* file. User-defined scripts must implement the following arguments.

Command summary:

```
upload.script [-r report] user dirname filename1 filename2 ...
```

The table below shows details of *upload.script arguments* and *variables*:

Table 3-21 upload.script arguments and variables

Variable	Definition
`-r report`	If a report was generated by the upload CGI, then it will execute the script with the `-r` argument, which is immediately followed by the path to the report.
`user`	The user's username is the first non-optional argument passed to the script.
`dirname`	The directory to which to files were uploaded
`filename1 filename2 ...`	Any other arguments to the CGI will be the names of the files uploaded.

Corresponding HTML/script files

On UNIX systems

- */usr/etc/webnative/styles/upload/*`style`*/upload.html*
- */usr/etc/webnative/styles/upload/*`style`*/frameset.html* (optional)
- */usr/etc/webnative/styles/upload/*`style`*/upload.script* (optional)
- */usr/etc/webnative/styles/upload/*`style`*/postupload.html* (optional)
- */usr/etc/webnative/styles/upload/*`style`*/uploadreport.html* (optional)

On Windows systems

- *C:\Program Files\Xinet\WebNative\Admin\styles\upload*`style`*\upload.html*
- *C:\Program Files\Xinet\WebNative\Admin\styles\upload*`style`*\frameset.html* (optional)
- *C:\Program Files\Xinet\WebNative\Admin\styles\upload*`style`*\upload.script* (optional)
- *C:\Program Files\Xinet\WebNative\Admin\styles\upload*`style`*\postupload.html* (optional)
- *C:\Program Files\Xinet\WebNative\Admin\styles\upload*`style`*\uploadreport.html* (optional)

Employing WebNative Uploader from an AppleScript

Xinet Uploaders provide an efficient drag-and drop mechanism for clients to upload files to a WebNative Suite or WebNative Portal server. It also allows users to associate metadata with uploaded files. WebNative Uploader now supports AppleScript, which allows you to automate its use in your workflow. For more information on what is possible using AppleScript, open the dictionary from within *Script Editor*. Figure 3-2 presents this part of the *Uploader* suite:

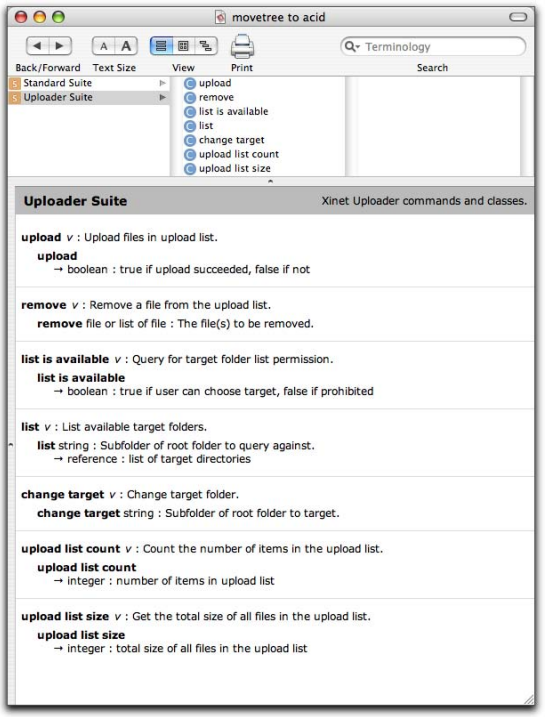


Figure 3-2 *Uploader* library

WebNative Portal APIs

BROWSE— listdir equivalent

Receives files/folders lists from WebNative Venture servers for display in a web browser. Based on input parameters, can also update/change/add metadata.

Table 4-1*BROWSE* input parameters

Input parameter	What it does
`_REQUEST['path']`	Full path to folder on WebNative Venture server from server root
`_REQUEST['view']`	Icon/short/long view. Sets the user's current view to the selected value. `_REQUEST['view']` linked to `_SESSION['VIEW'].`
`_REQUEST['show_columns']`	Sets the number of columns to use for files in icon, short or custom templates. `_REQUEST['show_columns']` linked to `_SESSION['SHOW_COLUMNS']`.
`_REQUEST['show_rows']`	Sets the number of rows to use for files in all BROWSE templates. `_REQUEST['show_rows']` linked to `_SESSION['SHOW_ROWS']`
`_REQUEST['show_index']`	Sets the file display index point from pagination. `_REQUEST['show_index']` linked to `_SESSION['SHOW_INDEX']`
`_REQUEST['show_rows_dirs']`	Sets the number of rows to use for folders in all BROWSE templates. `_REQUEST['show_rows_dirs']` linked to `_SESSION['SHOW_ROWS_DIRS']`
`_REQUEST['show_index_dirs']`	Sets the folder display index point from pagination. `_REQUEST['show_index_dirs']` linked to `_SESSION['SHOW_INDEX_DIRS']`

Table 4-1*BROWSE* input parameters

Input parameter	What it does
`_REQUEST['server']`	*Base64_encoded* string of the WebNative server's IP address or domain name. For multi-server setup, `_REQUEST['server']` linked to `_SESSION['SERVER']`
`_REQUEST['siteurl']`	*Base64_encoded* string of the current template set base folder. Example: *http://127.0.0.1/House* yields *base64_encode('House')* Example: *http://127.0.0.1/* yields *NULL*
`_REQUEST['sort']`	Sorts the files by any TAG defined in the *$files_info* array. Example: *File Type*; `sort=FILE_TYPE.` Example: *Keyword*; `sort=KEYWORD_VALUE`
`_REQUEST['tmpl']`	Template specified. if `_REQUEST['releasetmpl']` is not specified, the template remains for the user session or until a change is specified by setting `_REQUEST['tmpl']` again. `REQUEST['tmpl']` linked to `_SESSION['TMPL']['BROWSE']`
`_REQUEST['releasetmpl']`	When used with `_REQUEST['tmpl']`, prevents `$_SESSION['TMPL']['IMAGEINFO']` from linking.
`_REQUEST['win']`	The *width:height* of the user's browser window. Used to calculate columns when set to *auto*.
`_REQUEST['submitcomment']`	Process metadata input
`_REQUEST['AIComment'][{FILEID}]`	IPTC comments field
`_REQUEST['keyword{KWID}'][{FILEID}]`	Sets a metadata field value other then formatted date values. (See below.) Example: when keyword ID is *278* and file ID is *1623*, `_REQUEST['keyword278'][1623]`
`_REQUEST['date{KWID}'][{FILEID}][0]`	*Year* value
`_REQUEST['date{KWID}'][{FILEID}][1]`	*Month* value
`_REQUEST['date{KWID}'][{FILEID}][2]`	*Day* value
`_REQUEST['date{KWID}'][{FILEID}][3]`	*Hour* value
`_REQUEST['date{KWID}'][{FILEID}][4]`	*Minute* value

Table 4-1*BROWSE* input parameters

Input parameter	What it does
`_REQUEST['date{KWID}'][{FILEID}][5]`	*Seconds* value
`_REQUEST['date{KWID}'][{FILEID}][6]`	*AM/PM* value for 12-hour formats
`_REQUEST['version']`	Display the current Xinet WebNative Portal version in the user's browser window

IMAGEINFO—imageinfo equivalent

Receives file/folder detailed information from WebNative Venture servers for display in a web browser. Based on input parameters, can also update/change/add metadata or process an image order.

Table 4-2*IMAGEINFO* input parameters

Input parameter	What it does
`_REQUEST['file']`	Full path to file/folder on WebNative Venture server from server root.
`_REQUEST['server']`	Base64_encoded string of the WebNative Venture server's IP address or domain name. For multi-server setup, `_REQUEST['server']` linked to `_SESSION['SERVER']`
`_REQUEST['siteurl']`	Base64_encoded string of the current template sets base folder. Example: http://127.0.0.1/House yields base64_encode('House') Example: http://127.0.0.1/ yields NULL
`_REQUEST['tmpl']`	Template specified. If `_REQUEST['releasetmpl']` is not specified, the template remains for the user session or until a change is specified by setting `_REQUEST['tmpl']` again. `_REQUEST['tmpl']` linked to `_SESSION['TMPL']['IMAGEINFO'].`
`_REQUEST['releasetmpl']`	When used with `_REQUEST['tmpl']`, prevents `$_SESSION['TMPL']['IMAGE-INFO']` from linking.
`_REQUEST['ipdf']`	WebNative-Venture-style path to file. Sets `_REQUEST['nonav']` = `true`. `_REQUEST['ipdf']` links to `_REQUEST['file']`

Table 4-2*IMAGEINFO* input parameters

Input parameter	What it does
`_REQUEST['action']`	When value = del, runs file management delete file process. Usually only used with versions. Requires both: `_REQUEST['filepath']`, the full path to file on WebNative Venture server from server root, and `_REQUEST['filename']`, the file name.
`_REQUEST['versions']`	True/false. Get versions data from WebNative Venture server for `_REQUEST['file'].`
`_REQUEST['links']`	True/false. Get linked files data from the WebNative Venture server for `_REQUEST['file'].`
`_REQUEST['nonav']`	Turns off navigable tags: `{SBB}, {FM}, {PB}, {NB}`, etc.
`_REQUEST['process']`	Process image order input. See IMAGEORDER for input parameters.
`_REQUEST['submitcomment']`	Process metadata input.
`_REQUEST['AIComment'][{FILEID}]`	IPTC comments field
`_REQUEST['keyword{KWID}'][{FILEID}]`	Sets a metadata field value, other than formatted date values. (See below.) Example when keyword ID is 278 and file ID is 1623, `_REQUEST['keyword278'][1623]`
`_REQUEST['date{KWID}'][{FILEID}][0]`	Year value
`_REQUEST['date{KWID}'][{FILEID}][1]`	Month value
`_REQUEST['date{KWID}'][{FILEID}][2]`	Day value
`_REQUEST['date{KWID}'][{FILEID}][3]`	Hour value
`_REQUEST['date{KWID}'][{FILEID}][4]`	Minute value
`_REQUEST['date{KWID}'][{FILEID}][5]`	Second value
`_REQUEST['date{KWID}'][{FILEID}][6]`	AM/PM value for 12 hour formats
`_REQUEST['version']`	Display the current Xinet WebNative Portal version in the user's browser window

SEARCHOPTIONS—searchengine equivalent

Provides searching options from WebNative Venture servers for display in a web browser.

Table 4-3*SEARCHOPTIONS* input parameters

Input parameter	What it does
`_REQUEST['path']`	Full path to folder on WebNative Venture server from server root.
`_REQUEST['searchallservers']`	If true, all WebNative-Venture-enabled volumes from all servers configured for access within config.inc.php will be searched.
`_REQUEST['searchallvols']`	If true, all WebNative-Venture-enabled volumes are searched.
`_REQUEST['server']`	Base64_encoded string of the WebNative server's IP address or domain name. Required for multi-server setup. `_REQUEST['server']` linked to `_SESSION['SERVER'].`
`_REQUEST['siteurl']`	Base64_encoded string of the current template set base folder. Example: http://127.0.0.1/House yields base64_encode('House') Example: http://127.0.0.1/ yields NULL
`_REQUEST['tmpl']`	Template specified. If `_REQUEST['releasetmpl']` is not specified, the template remains for the user session or until a change is specified by setting `_REQUEST['tmpl']` again. `_REQUEST['tmpl']` linked to `_SESSION['TMPL']['BROWSE'].`
`_REQUEST['releasetmpl']`	When used with `_REQUEST['tmpl']`, prevents `$_SESSION['TMPL']['IMAGE-INFO']` from linking.
`_REQUEST['addfield']`	Add searchoptions field to view.
`_REQUEST['removefield_n']`	Remove searchoptions field from view.
`_REQUEST['Clear']`	Clear all selected search options.
`_REQUEST['actionsearch']`	True = perform search
`_REQUEST['SEARCHFIELD']['FIELD-NAME']` and `_REQUEST['FIELDNAME']`	Both are allowable for all search criteria submissions listed below:
`_REQUEST['SEARCHFIELD']['FILEDNAME']`	Used for documentation purposes only

Table 4-3*SEARCHOPTIONS* input parameters

Input parameter	What it does
`_REQUEST['SEARCHFIELD']['DBSEAR-CHALL']`	Value to search against all fields
`_REQUEST['SEARCHFIELD']['DBSEARCH-LOGIC']`	0/1 for And/Or
`_REQUEST['SEARCHFIELD']['FILENAME']`	Value to search against file and/or folders
`_REQUEST['SEARCHFIELD']['FILENAMELOGIC']`	0
`_REQUEST['SEARCHFIELD']['PATHSEARCH-TYPE']`	`1` = Contains `2` = Starts With `-1` = Has Any (Must be Full-Text enabled)
`_REQUEST['SEARCHFIELD']['PATHTYPE']`	`1` = Files `2` = Folders `3` = Files and Folders
`_REQUEST['SEARCHFIELD']['PATHTYPE-LOGIC']`	`0`
`_REQUEST['SEARCHFIELD']['FILETYPE']`	`-1` = Images `-2` = Non Images `-3` = PDF `-4` = QUARK `-7` = InDesign `1` = All Types `PIX` = Alias PIX `ArtP` = Art Pro CT `CT ` = Barco CT `CT01` = Contex CT `Cros` = Crosfield `DA{M` = Dalim `TILE` = Eclipse Tile `EPSF` = EPSF `GIFf ` = GIF `PEG` = JPEG `BMP` = MSWin BMP `PCDI` = Photo CD `8BPS` = Photoshop Native `..CT` = Scitex CT `LINE` = Scitex LW `SGI` = SGI Imagelib `SUNR` = Sun Raster `TIFF` = TIFF `XWND` = X Windows Dump
`_REQUEST['SEARCHFIELD']['FILETYPELOGIC']`	`0/1` for *And/Or*

Table 4-3*SEARCHOPTIONS* input parameters

Input parameter	What it does
`_REQUEST['SEARCHFIELD']['SELECTEDVOL']`	`1` = Online Files `2` = Archive Files `3` = Online and Archived files
`_REQUEST['SEARCHFIELD']['SELECTEDVOLLOGIC']`	`0`
`_REQUEST['SEARCHFIELD']['DATE']['MONTH']`	Month value
`_REQUEST['SEARCHFIELD']['DATE']['DAY']`	Day value
`_REQUEST['SEARCHFIELD']['DATE']['YEAR']`	Year value
`_REQUEST['SEARCHFIELD']['DATELOGIC']`	`0/1` for *And/Or*
`_REQUEST['SEARCHFIELD']['DATETYPE']`	`1` = Created `2` = Modified `3` = Last Accessed
`_REQUEST['SEARCHFIELD']['DATEON']`	`on` = search on `DATE` input values
`_REQUEST['SEARCHFIELD']['DATEWHENCE']`	`1` = Since `2` = Before `3` = On `8` = Is Null
`_REQUEST['SEARCHFIELD']['COMMENT']`	Comment field value
`_REQUEST['SEARCHFIELD']['COMMENTLOGIC']`	`0/1` for *And/Or*
`_REQUEST['SEARCHFIELD']['DBSEARCHALLKEYWORD']`	Value to search against all keyword fields
`_REQUEST['SEARCHFIELD']['DBSEARCHALLLOGIC']`	`0/1` for *And/Or*
`_REQUEST['SEARCHFIELD']['DBSEARCHKEYWORD{KWID}']`	Metadata field search Value, other than formatted date values. (See below.)
`_REQUEST['SEARCHFIELD']['DBSEARCHDATE{KWID}'][YEAR]`	*Year* value
`_REQUEST['SEARCHFIELD']['DBSEARCHDATE{KWID}'][MONTH]`	*Month* value
`_REQUEST['SEARCHFIELD']['DBSEARCHDATE{KWID}'][DAY]`	*Day* value
`_REQUEST['SEARCHFIELD']['DBSEARCHDATE{KWID}'][HOUR]`	*Hour* value

Table 4-3*SEARCHOPTIONS* input parameters

Input parameter	What it does
`_REQUEST['SEARCHFIELD']['DBSEARCHDATE{KWID}'][NUTE]`	*Minute* value
`_REQUEST['SEARCHFIELD']['DBSEARCHDATE{KWID}'][SECS]`	*Seconds* value
`_REQUEST['SEARCHFIELD']['DBSEARCHLOGIC{KWID}']`	*0/1 for And/Or*
`_REQUEST['SEARCHFIELD']['DBSEARCHFLAG{KWID}']`	*For Text Fields:* `-1` = Has Any (Full-Text fields only) `0` = Contains `1` = Equals `2` = Starts With `3` = Ends With `4` = Regular Expression `5` = Full Text (Full-Text fields only) `8` = Is Null *For Date Fields:* `402` = Before `404` = Since `405` = On *For Integer Fields:* `201` = Less Then `203` = Greater Then `205` = Equals `8` = Is Null
`_REQUEST['version']`	Display the current Xinet WebNative Portal version in the user's browser window.

SEARCHRESULTS— searchengine equivalent

Search results, files/folders lists, from WebNative Venture servers for display in a web browser. Based on input parameters, can also update/change/add metadata.

Table 4-4*SEARCHRESULTS* input parameters

Input parameter	What it does
`_REQUEST['view']`	Icon/short/long. Sets the users current view to the selected value. `_REQUEST['view']` linked to `_SESSION['VIEW']`

Table 4-4*SEARCHRESULTS* input parameters

Input parameter	What it does
`_REQUEST['show_columns']`	Sets the number of columns to use for files in icon, short or custom templates. `_REQUEST['show_columns']` linked to `_SESSION['SHOW_COLUMNS']`
`_REQUEST['show_rows']`	Sets the number of rows to use for files in all BROWSE templates. `_REQUEST['show_rows']` linked to `_SESSION['SHOW_ROWS']`
`_REQUEST['show_index']`	Sets the file-display index point from pagination. `_REQUEST['show_index']` linked to `_SESSION['SHOW_INDEX']`
`_REQUEST['show_rows_dirs']`	Sets the number of rows to use for folders in all BROWSE templates. `_REQUEST['show_rows_dirs']` linked to `_SESSION['SHOW_ROWS_DIRS']`
`_REQUEST['show_index_dirs']`	Sets the folder-display index point from pagination. `_REQUEST['show_index_dirs']` linked to `_SESSION['SHOW_INDEX_DIRS']`
`_REQUEST['skipcount']`	Actionmore/actionless display the next or previous page of results-to-display. Used when `Allow_SearchAll` is disabled.
`_REQUEST['sort']`	Sorts the files by any TAG defined in the $files_info array. Example: File Type; `sort=FILE_TYPE`. Example: Keyword; `sort=KEYWORD_VALUE`
`_REQUEST['tmpl']`	Template specified. If `_REQUEST['releasetmpl']` is not specified, the template remains for the user session or until a change has been specified by setting `_REQUEST['tmpl']` again. `_REQUEST['tmpl']` linked to `_SESSION['TMPL']['BROWSE']`
`_REQUEST['releasetmpl']`	When used with `_REQUEST['tmpl']`, prevents `$_SESSION['TMPL']['IMAGE-INFO']` from linking.
`_REQUEST['win']`	The width:height of the user's browser window. Used to calculate columns when set to auto.

Table 4-4*SEARCHRESULTS* input parameters

Input parameter	What it does
`_REQUEST['submitcomment']`	Process metadata input as explained below.
`_REQUEST['AIComment'][{FILEID}]`	IPTC comments field
`_REQUEST['keyword{KWID}'][{FILEID}]`	Sets a metadata field value, other than formatted date values. (See below.) Example: when keyword ID is 278 and file id is 1623: `_REQUEST['keyword278'][1623]`
`_REQUEST['date{KWID}'][{FILEID}][0]`	Year value
`_REQUEST['date{KWID}'][{FILEID}][1]`	Month value
`_REQUEST['date{KWID}'][{FILEID}][2]`	Day value
`_REQUEST['date{KWID}'][{FILEID}][3]`	Hour value
`_REQUEST['date{KWID}'][{FILEID}][4]`	Minute value
`_REQUEST['date{KWID}'][{FILEID}][5]`	Seconds value
`_REQUEST['date{KWID}'][{FILEID}][6]`	AM/PM value for 12-hour formats
`_REQUEST['version']`	Display the current Xinet WebNative Portal version in the user's browser window

IMAGEORDER— imageorder equivalent

Based on input parameters, process a *Custom Image Order*.

Table 4-5*IMAGEORDER* input parameters

Input parameter	What it does
`_REQUEST['file']`	Full path to file/folder on the WebNative Venture server from server root
`_REQUEST['server']`	Base64_encoded string of the WebNative server's IP address or domain name. For multi-server setup, `_REQUEST['server']` linked to `_SESSION['SERVER']`.

Table 4-5*IMAGEORDER* input parameters

Input parameter	What it does
`_REQUEST['siteurl']`	Base64_encoded string of the current template sets base folder. Example: *http://127.0.0.1/House* yields `base64_encode('House')` Example: *http://127.0.0.1/* yields `NULL`
`_REQUEST['tmpl']`	Template specified. If `_REQUEST['releasetmpl']` is not specified, the template remains for the user session or until a change is specified through resetting `_REQUEST['tmpl']`. `_REQUEST['tmpl']` linked to `_SESSION['TMPL']['IMAGEINFO']`
`_REQUEST['releasetmpl']`	When used with `_REQUEST['tmpl']`, prevents `$_SESSION['TMPL']['IMAGE-INFO']` from linking.
`_REQUEST['webready']`	Output is ready for display in a web page from an `<img>` HTML tag

Table 4-5*IMAGEORDER* input parameters

Input parameter	What it does
`_REQUEST['format']`	File format for output: • *eps** Encapsulated PostScript • *tif* Tag Image File Format (TIFF) • *gif* Composure's Graphics Interchange Format • *jpg** Native JPEG, "lossy" compression format • *web** Either GIF (if image is masked or clipped) or JPEG • *bmp* MS Windows BitMaP (aka DIB) format • *png* Portable Network Graphic format Those formats marked with * all take an optional number following the format (e.g., -x eps83). This number sets a JPEG compression factor (normally off for EPS format). The JPEG compression factor is a number with a useful range between 5 and 95, where 5 creates a highly-compressed image that doesn't look very much like the original, and 95 produces a less-compressed image, which will look very much like (but not exactly like) the original. When JPEG output format is requested, the default compression factor is 75. Requesting EPS format with a compression value of 0 will cause the output to be HEX encoded (instead of the default BINARY); but, not JPEG compressed.
`_REQUEST['colorspace']`	Process the output to a colorspace: Grey, RGB, LAB or CMYK.
`_REQUEST['pict']`	Extract the PICT image if available.
`_REQUEST['inputicc']`	Sets an input ICC profile pathname, which overrides any other profile included within the image.

Table 4-5*IMAGEORDER* input parameters

Input parameter	What it does
`_REQUEST['outputicc']`	Sets an output ICC profile pathname. This option enables ICC color correction if one of the following conditions is met (precedence in the order listed): 1. An input profile has been supplied with the `inputticc` option 2. The image has an embedded profile 3. A default profile has been specified on the WebNative server for the image format and colorspace in the file */var/adm/appletalk/coloropts* (UNIX) or *C:\Program Name\Xinet\FullPress \Admin\coloropts* (Windows). (FullPress creates/updates the *coloropts* file whenever the administrator saves default ICC profiles in a *Print Queue* for various formats.) 4. The input image is in LAB colorspace.
`_REQUEST['renderingintent']`	Sets the Rendering Intent for ICC color conversion.Normally, the default Rendering Intent supplied in the profile is used. However, it can be overridden with this flag, as long as the profile that is being used supports the substituted Rendering Intent. If the requested Rendering Intent isn't supported, the profiles' default will be used. Current valid Rendering Intents include: `0` = Perceptual `1` = Relative Colorimetric `2` = Saturation `3` = Absolute Colorimetric

Table 4-5*IMAGEORDER* input parameters

Input parameter	What it does
`_REQUEST['usm']`	Adds an Unsharp Mask (sharpening) filter over the image. This option takes one argument, which can be one of the following: • The number *0* causes the sharpening parameters to be read from the default system table (a file named *default* in the */var/adm/appletalk/usm* directory (UNIX) or *C:\Program Files\Xinet\FullPress\Admin\usm* directory (Windows). • You can supply a file name and the table will be read from that file. • The argument can alternatively be *1*–to–*4* comma-separated numbers giving the parameters for the filter. The sharpening parameters include: The *radius,* in pixels, of the blurring Aperture, which must be greater than *0.5* (and has no default) A *percentage strength of the sharpening values* to apply (default is *100*), where anything *0* or less disables the Unsharp Mask filter A *lower threshold, in pixel intensity* (*0-255*), below which no sharpening will occur (default is *0*). An *upper limit on the sharpening* that will be applied to the image (default is *255*, and any number out of the *1–255* range will be ignored). A value of *255* means no limit, a value of *10* means limit the maximum amount of change to a pixel to *10* (in component intensity value units, which are *0–255*). Note that the Unsharp Mask parameters differ from PhotoShop in the radius only: add one to a PhotoShop Unsharp Mask Radius to get the equivalent sharpening.

Table 4-5*IMAGEORDER* input parameters

Input parameter	What it does
`_REQUEST['crop']`	The argument specifies a rectangle on the source image, in pixels, numbered from (0,0), which is the pixel in the upper-left corner of the image. The coordinates cropx (horizontal) and cropy (vertical) provide the coordinates of the first pixel that will be placed in the upper-left corner of the output image. The pair, cropw (width) and croph (height) give the number of pixels of the original image to include in the cropped image.
`_REQUEST['watermark']`	Turns on watermarking for the output image. This turns on automatically if the FullPress installation has not been licensed. By default, a watermark is tiled, pixel-for-pixel across and down the image as many times as it will fit.
`_REQUEST['dpi']`	Output resolution to a DPI (dots per inch)
`_REQUEST['scale']`	Images may be scaled (up or down) by supplying the -s option with the (floating point) percentage of the image size (in pixels) to output.
`_REQUEST['width']`	Scale to a specific width
`_REQUEST['height']`	Scale to a specific height
`_REQUEST['backcolor']`	Sets the background color, when a Masked or Clipped image is output in a format that does not support masks or Clipping paths to the given color.
`_REQUEST['spotoff']`	Spot colors will be merged with the color-space channels and output in the image. This option excludes spot channels.
`_REQUEST['clipping']`	Causes all Masks and Clipping paths to be ignored.
`_REQUEST['preview']`	This option causes imageorder to search for a “preview” image before reading in the “full-scale” image.
`_REQUEST['pctiff']`	If an EPS image has a PC-style TIFF preview, this option causes the TIFF to be read instead of the EPS.

Table 4-5*IMAGEORDER* input parameters

Input parameter	What it does
`_REQUEST['nonav']`	Turns off navigable tags: `{SBB}`, `{FM}`, `{PB}`, `{NB}`, etc.
`_REQUEST['process']`	Required to start process
`_REQUEST['version']`	Display the current Xinet WebNative Portal version in the user's browser window

Appendix A

Xinet End User Software License Agreement

THIS XINET END USER SOFTWARE LICENSE AGREEMENT ("AGREEMENT") IS THE LEGAL AGREEMENT THAT GOVERNS YOUR USE OF THE ASSET MANAGEMENT SOFTWARE MADE AVAILABLE BY XINET INC. (TOGETHER WITH ITS ACCOMPANYING DOCUMENTATION, THE "SOFTWARE"). THIS AGREEMENT IS BETWEEN YOU, THE CUSTOMER WHO HAS ACQUIRED THE SOFTWARE ("YOU"), AND XINET INC. ("XINET"). PLEASE READ THIS AGREEMENT CAREFULLY.

XINET IS ONLY WILLING TO PROVIDE THE SOFTWARE TO YOU ON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS AGREEMENT. YOU ACCEPT THIS AGREEMENT BY INSTALLING OR USING THE SOFTWARE OR CLICKING ON THE BUTTON INDICATING ACCEPTANCE OF THE AGREEMENT WHICH IS PRESENTED UPON SOFTWARE INSTALLATION. BY CLICKING ON THE BUTTON TO ACCEPT THIS AGREEMENT OR BY INSTALLING THE SOFTWARE, YOU REPRESENT AND WARRANT THAT YOU HAVE THE AUTHORITY TO ENTER INTO THIS AGREEMENT, PERSONALLY OR IF YOU HAVE NAMED A COMPANY AS CUSTOMER, ON BEHALF OF THE COMPANY NAMED AS CUSTOMER, AND TO BIND EITHER YOURSELF OR SUCH COMPANY TO THE TERMS OF THIS AGREEMENT.

IF YOU DID NOT ACQUIRE THE SOFTWARE FROM XINET OR FROM AN AUTHORIZED XINET INTEGRATOR OR A XINET AFFILIATE (EACH, AN "AUTHORIZED INTEGRATOR"), THEN YOU MAY NOT ENTER INTO THIS AGREEMENT OR USE THE SOFTWARE. NO OTHER PARTY HAS THE RIGHT TO TRANSFER A COPY OF THE SOFTWARE TO YOU.

IF YOU ARE UNWILLING TO ACCEPT THIS AGREEMENT, DO NOT USE THE SOFTWARE. IF YOU HAVE ALREADY PAID FOR THE SOFTWARE WITHOUT HAVING A PRIOR OPPORTUNITY TO REVIEW THIS AGREEMENT AND ARE NOW UNWILLING TO AGREE TO THESE TERMS, YOU MAY, WITHIN TEN (10) DAYS AFTER THE DATE ON WHICH YOU ACQUIRED THE SOFTWARE, RETURN IT TO XINET OR THE AUTHORIZED INTEGRATOR FROM WHOM YOU ACQUIRED IT, ALONG WITH ITS ORIGINAL PACKAGING AND PROOF-OF-PURCHASE, FOR A FULL REFUND.

NOTWITHSTANDING ANYTHING HEREIN TO THE CONTRARY, NO AUTHORIZED XINET INTEGRATOR ACTS AS AN AGENT OF XINET, AND NO SUCH PARTY MAY ENTER INTO ANY CONTRACTS ON BEHALF OF XINET. NO AUTHORIZED INTEGRATOR HAS THE AUTHORITY TO MODIFY THE TERMS OF THIS AGREEMENT.

1. Grant of License. The Software consists of (a) certain components that are designated by Xinet as server software ("Server Software") that permit you to manage creative asset files stored on the one or more internal Xinet servers for which you have purchased a license (each, a "Xinet Server"), (b) certain client-side plug-in and extension utility components that permit the number of your internal employees for whom you have purchased a use license concurrently to access and use the Server Software and the files stored on the Xinet Server ("Client Software"), and (c) certain client-side utility components that provide desktop computer users, which may be external to your organization, with limited access to, and use of, your Xinet Server via the internet, solely to enable such users to upload, download, and search the creative asset files stored on your Xinet Server ("Utility Software"). The Server Software, Client Software and the Utility Software are collectively referred to herein as the "Software". Depending on the specific set of Xinet Software components for which you have paid the corresponding license fees, certain components may not be available to you in your Software installation and are not covered by the licenses granted in this Agreement.

1.1 Server Software. Subject to the terms and conditions of this Agreement, Xinet grants you a non-exclusive, non-transferable, worldwide, revocable license, during the term of this Agreement, to (a) install and execute, in each case on the Xinet Server in your possession, one (1) copy of the Server Software for which you have paid the corresponding license fees, in executable code form solely in accordance with the Documentation and solely to establish and use, on an internal basis and by the number of concurrent users for whom you have paid the corresponding license fees, one or more volumes of creative asset files to be stored on your Xinet Server and (b) to make one (1) copy of the Server Software for which you have paid the corresponding license fees, solely for backup or archival purposes.

1.2 Client Software. Subject to the terms and conditions of this Agreement, Xinet grants you a non-exclusive, non-transferable, worldwide, revocable license, during the term of this Agreement, to make a reasonable number of copies of the Client Software and to install and execute such copies of the Client Software on end-user computers in your possession, in each case solely in executable code form and in accordance with the Documentation, solely for your internal business purposes, and to enable the number of concurrent users for whom you have paid the corresponding license fees to access and use the Xinet Server and upload and download files from the Xinet Server.

1.3 Utility Software. Subject to the terms and conditions of this Agreement, Xinet grants you a non-exclusive, non-transferable, worldwide, revocable license, during the term of this Agreement, to (a) permit each third party to whom you are providing consulting services related to the management of creative asset files (each, a "Utility User") to install and execute the Utility Software, solely in executable code and solely (i) in accordance with the Documentation and (ii) to enable the Utility User to access, via the internet, the Xinet Server and upload and download files from the Xinet Server, in each case in connection with the consulting services provided by you. You will be solely responsible for each Utility User's use of the Utility Software and any services provided by you to any Utility User. Xinet and its suppliers make no warranty, and have no obligation to provide support or other services to any Utility User. You will defend and indemnify Xinet and its suppliers from any claims or liabilities arising from or related to any Utility User to whom you provide access to the Software. You must enter into transactions with each Utility User on your own account and not on behalf of Xinet.

2. Restrictions on Use. You acknowledge, and will notify each Utility User, that the Software and its structure, organization, and source code constitute valuable trade secrets of Xinet and its suppliers. Accordingly, you agree not to (a) modify, adapt, alter, translate, or create derivative works from the Software or any copy, in whole or in part; (b) merge the Software with other software; (c) except as expressly permitted in Section 1.3, sublicense, lease, rent, loan, or otherwise transfer the Software to any third party, (d) use the Software in any service bureau or time-sharing arrangement, (e) **REVERSE ENGINEER, DECOMPILE, DISASSEMBLE, OR OTHERWISE ATTEMPT TO DERIVE THE SOURCE CODE FOR THE SOFTWARE**, or (f) otherwise use or copy the Software except as expressly allowed under Section 1. You will not remove, alter, or obscure any proprietary rights notices (including copyright notices) of Xinet or its suppliers on the Software, and you shall reproduce the same proprietary rights notices (including copyright notices) on any copy of the Software as appears on the original. You will ensure that each copy of the Utility Software that is distributed or made available by you is conspicuously marked with Xinet's proprietary rights notices.

3. Delivery and Acceptance. The Software is licensed and not sold, and the nonexclusive license set forth in this Agreement is not a sale of the Software or any copy. Title to the media on which the Software is recorded will transfer to you upon the delivery to you of such media, and the Software shall be deemed irrevocably and unconditionally accepted upon such delivery. Title to the Software remains with Xinet.

4. Proprietary Rights. **THE SOFTWARE, AND ALL WORLDWIDE INTELLECTUAL PROPERTY RIGHTS THEREIN, ARE THE EXCLUSIVE PROPERTY OF XINET AND ITS SUPPLIERS.** Except as stated above, this Agreement does not grant you any rights to any patents, copyrights, trade secrets, trade names, trademarks (whether registered or unregistered), or any other rights, franchises, or license in respect of the Software. All rights in and to the Software not expressly granted to you in this Agreement are reserved by Xinet and its suppliers.

5. Term and Termination; Effects of Termination. The term of this Agreement will begin on the date you accept this Agreement as provided above and will continue indefinitely. You may terminate your license at any time by destroying all copies of the Software in your possession or control. Xinet will have the right to terminate the license(s) granted to you by this Agreement immediately upon notice to you if you fail to comply with any term or condition of this Agreement. Upon any termination of this Agreement and/or the licenses for any reason, any amounts owed to Xinet under this Agreement before such termination or expiration will be immediately due and payable, all license rights granted in this Agreement will immediately cease to exist, and you must promptly discontinue all use of the Software, erase all copies of the Software from your computers, and return to Xinet, at your expense, all copies of the Software on tangible media in your possession or control.

6. Limited Warranty.

6.1 Xinet warrants to you that, for a period of thirty (30) days after delivery to you ("Warranty Period") the Software, when used as permitted under this Agreement and in accordance with the instructions in the Documentation (including use on a computer hardware and operating system platform supported by Xinet), will be free from Errors. As used herein, "Error" means a reproducible failure of the Software to perform substantially in accordance with the Documentation provided to you together with the Software. Xinet does not warrant that your use of the Software will be completely error-free or uninterrupted. If you received the Software from an Authorized

Integrator, you may be required to obtain services pursuant to the warranty in this Section 6.1 directly from your Authorized Integrator.

6.2 If you notify Xinet or the Authorized Integrator, as applicable, during the Warranty Period that the Software fails to comply with the limited warranty in Section 6.1 above, Xinet will, at its option and expense and as its sole obligation and your exclusive remedy for any breach of such warranty, use commercially reasonable efforts to correct any reproducible error in the Software reported by you in writing during the Warranty Period, or, if Xinet determines that it is unable to correct the error, Xinet may, at its option, refund to you or the Authorized Integrator, on your behalf, the license fees actually paid by you to Xinet or the Authorized Integrator, as applicable, in which case the License and your right to use the Software will be terminated. Any Error correction provided to you will not extend the original Warranty Period. Notwithstanding anything herein to the contrary, if you received the Software from an integrator, distributor, reseller, or other third party you shall only be entitled to make a warranty claim to, and receive warranty remedies from, such third party.

6.3 **XINET AND ITS SUPPLIERS DO NOT AND CANNOT WARRANT THE PERFORMANCE OR RESULTS YOU MAY OBTAIN BY USING THE SOFTWARE. YOU UNDERSTAND THAT, EXCEPT FOR THE EXPRESS WARRANTY SET FORTH IN SECTION 6.1, XINET AND ITS SUPPLIERS MAKE NO WARRANTIES OF ANY KIND, WHETHER EXPRESS, IMPLIED, OR STATUTORY, WITH RESPECT TO THE SOFTWARE, INCLUDING ANY WARRANTIES AS TO PERFORMANCE, NON-INFRINGEMENT OF THIRD PARTY RIGHTS, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. EXCEPT FOR THE EXPRESS WARRANTY STATED IN SECTION 6.1, THE SOFTWARE IS PROVIDED "AS IS" WITH ALL FAULTS, AND THE ENTIRE RISK AS TO SATISFACTORY QUALITY, ACCURACY, AND EFFORT IS WITH YOU. YOU ACKNOWLEDGE AND AGREE THAT YOU HAVE NOT RELIED ON ANY ORAL OR WRITTEN INFORMATION OR ADVICE, WHETHER GIVEN BY XINET OR ANY AUTHORIZED INTEGRATOR, AGENTS OR EMPLOYEES.**

6.4 Some states do not allow the exclusion of implied warranties or limitations on how long an implied warranty may last, so the above limitations may not apply to you. This warranty gives you specific legal rights and you may also have other rights which vary from state to state.

7. Limit of Liability.

7.1 **IN NO EVENT WILL XINET OR ITS SUPPLIERS BE LIABLE TO YOU FOR ANY CONSEQUENTIAL, INDIRECT, EXEMPLARY, SPECIAL, OR INCIDENTAL DAMAGES, OR DAMAGES FOR ANY LOST DATA OR LOST PROFITS, ARISING FROM OR RELATING TO THIS AGREEMENT, EVEN IF XINET HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. XINET'S TOTAL CUMULATIVE LIABILITY IN CONNECTION WITH THIS AGREEMENT AND THE SOFTWARE, WHETHER IN CONTRACT OR TORT OR OTHERWISE, WILL NOT EXCEED THE AMOUNT OF LICENSE FEES PAID TO XINET OR YOUR AUTHORIZED INTEGRATOR, AS APPLICABLE, HEREUNDER. YOU ACKNOWLEDGE THAT THE LICENSE FEES REFLECT THE ALLOCATION OF RISK SET FORTH IN THIS AGREEMENT AND THAT XINET WOULD NOT ENTER INTO THIS AGREEMENT WITHOUT THESE LIMITATIONS ON ITS LIABILITY. IN ADDITION, XINET DISCLAIMS ALL LIABILITY OF ANY KIND OF XINET'S SUPPLIERS.**

7.2 Some states do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.

8. Export. You acknowledge that the laws and regulations of the United States restrict the export and re-export of the Software. You agree that you will not export or re-export the Software or media in any form without the appropriate United States and foreign government approval.

9. Choice of Law. This Agreement will be governed by the laws of the State of California without regard to any conflict of laws provisions that would require the applications of the laws of a different jurisdiction. The United Nations Convention on Contracts for the International Sale of Goods does not apply to this Agreement. Any action or proceeding arising from or relating to this Agreement may be brought in a federal court in the Northern District of California or in state court in Alameda County, California, and each party irrevocably submits to the jurisdiction and venue of any such court in any such action or proceeding.

10. U.S. Government Restricted Rights Legend. The Software is comprised of "commercial computer software" and "commercial computer software documentation" as such terms are used in 48 C.F.R. 12.212 and are provided to the Government (i) for acquisition by or on behalf of civilian agencies, consistent with the policy set forth in 48 C.F.R. 12.212 or (ii) for acquisition by or on behalf of units of the Department of Defense, consistent with the policies set forth in 48 C.F.R. 227.7202-1 and 227.7202-3.

11. Notices. All notices, consents, and approvals under this Agreement must be delivered in writing by courier, by electronic facsimile (fax), or by certified or registered mail, (postage prepaid and return receipt requested) to the other party, for Xinet at the address set out on the Xinet website or for you, at the address provided to Xinet when

the Software was licensed by you, and such notice will be effective upon receipt or three (3) business days after being deposited in the mail as required above. You may change your address by giving notice of the new address to Xinet.

12. Assignment. You may not assign or transfer, by operation of law or otherwise, any of your rights under this Agreement (including your licenses with respect to the Software) to any third party without Xinet's prior written consent. Any attempted assignment or transfer in violation of the foregoing will be void.

13. Remedies. Except as provided in Sections 6 and 7, the parties' rights and remedies under this Agreement are cumulative. You acknowledge that the Software contains valuable trade secrets and proprietary information of Xinet, that any actual or threatened breach of Section 2 will constitute immediate, irreparable harm to Xinet for which monetary damages would be an inadequate remedy, and that injunctive relief is an appropriate remedy for such breach.

14. Third Party Software. Certain items of software that may be provided to Xinet together with the Software are owned by third parties and are subject to "open source" or "free software" licenses ("Open Source Software"). The Open Source Software is not subject to the terms and conditions in this Agreement other than Section 6.3 and Section 7. Instead, each item of Open Source Software is licensed under the terms of the end-user license that accompanies such Open Source Software. Nothing in this Agreement limits your rights under, or grants you rights that supersede, the terms and conditions of any applicable end-user license for the Open Source Software. If required by any license for particular Open Source Software, Xinet provides the information to access such Open Source Software and its accompanying end-user license in the Documentation.

15. Third Party Beneficiary. Artifex Software Inc. is an express third-party beneficiary of this Agreement. Except as expressly set forth in this section, there are no intended third-party beneficiaries of this Agreement.

16. Waivers. All waivers must be in writing. Any waiver or failure to enforce any provision of this Agreement on one occasion will not be deemed a waiver of any other provision or of such provision on any other occasion.

17. Severability. If any provision of this Agreement is unenforceable, such provision will be changed and interpreted to accomplish the objectives of such provision to the greatest extent possible under applicable law and the remaining provisions will continue in full force and effect. Without limiting the generality of the foregoing, you agree that Section 7 will remain in effect notwithstanding the unenforceability of any provision in Section 6.

18. Construction. The headings of Sections of this Agreement are for convenience and are not to be used in interpreting this Agreement. As used in this Agreement, the word "including" means "including but not limited to".

19. Integration. You acknowledge that you have read this Agreement, understand it, and that by installing the Software you agree to be bound by its terms and conditions. You further agree that it is the complete and exclusive statement of the agreement between Xinet and you, which supersedes any proposal or prior agreement, oral or written, and any other communications between Xinet and you relating to the subject matter of this Agreement. No variation of the terms of the Agreement or any different terms will be enforceable against Xinet unless Xinet gives its express consent, including an express waiver of the terms of this Agreement, in writing signed by an officer of Xinet.

Xinet Trademark Acknowledgements, Attributions, and Credit Notices

Xinet, the Xinet logo, WebNative, Picture Wrangler, FullPress, WebVolume, Queue Master, Color Verite, Server to the Macs, FPO Exporter, K-Talk, K-Time, K-FS, K-Spool, and K-AShare are either trademarks or registered trademarks of Xinet, registered in the United States and other countries.

• • • • •

Xinet acknowledges the following trademarks for company names, brands, and product names mentioned within its published materials:

Adobe, Adobe Acrobat, Adobe PDF and the Adobe PDF logo design, Adobe Premiere, Adobe Type Manager, Creative Suite, Distiller, Flash and the Flash logo design, FrameMaker, Freehand, GoLive, InDesign, Illustrator, Macromedia, PageMaker, Photoshop and the Photoshop logo design, PostScript and the PostScript logo design, XMP and the XMP logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Apache and the Apache feather logo are trademarks of The Apache Software Foundation.

Apple, AppleScript, AppleShare, AppleTalk, Bonjour, Ethertalk, Extensions Manager, Final Cut, Finder, ImageWriter, LaserWriter, LocalTalk, Macintosh, Mac OS, QuickTime, Safari, TeachText, TrueType, Xsan, Xserve and other Apple related marks are either trademarks or registered trademarks of Apple Inc., registered in the United States and/or other countries.

Artifex is a registered trademark of Artifex Software Inc. in the United States and/or other countries.

Intel is a registered trademark of Intel Corporation in the United States and other countries.

Linux is the registered trademark of Linus Torvalds in the United States and other countries.

Microsoft, Active Directory, Excel and the Microsoft Excel logo design, Internet Explorer and the Microsoft Internet Explorer logo design, Microsoft Office and the Office logo design, PowerPoint, Windows, Windows NT, Microsoft Windows Server, Windows XP, Windows 2000, Windows 2003 and the Windows logo design are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Mozilla, Firefox and the Firefox logo design, Camino, and the Camino logo design are registered trademarks of the Mozilla Foundation in the United States and other countries.

MultiAd and MultiAd Creator are registered trademarks of Multi-Ad Services, Inc., in the United States and other countries.

Quark, the Quark logo, QuarkXPress, QuarkXPress Passport, Quark Publishing System, XTensions, the XTensions logo, and other Quark related marks are trademarks or registered trademarks of Quark, Inc. and its affiliates in the U.S. and/or other countries.

Red Hat, Red Hat Enterprise Linux, and the Red Hat "Shadowman" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

Symantec, NetBackup, Backup Exec, and Veritas are trademarks or registered trademarks of Symantec Corporation or its affiliates in the United States and other countries.

Silicon Graphics, SGI and the SGI logo design, Altix, IRIX, Octane, Origin, Rackable Systems and the Rackable logo design, and XFS are either registered trademarks or trademarks of Silicon Graphics International in the United States and/or other countries worldwide.

Sun, Sun Microsystems, the Sun logo, Java and all Java-based trademarks and logo designs, MySQL and the MySQL logo, NFS, Solaris and the Solaris logo are trademarks or registered trademarks of Sun Microsystems, Inc. or its subsidiaries in the United States and/or other countries.

SPARC is a registered trademark of SPARC International, Inc. in the United States and other countries.

Telestream and FlipFactory are registered trademarks of Telestream, Inc. in the United States and/or other countries.

UNIX, X/Open, Motif, OSF/1, and The Open Group are registered trademarks of The Open Group in the United States and other countries.

Zend and Zend Technologies, and/or the logos and icons associated with each of these names are trademarks of Zend Technologies Ltd. and may be registered in certain jurisdictions.

All other product names, brands, company names, registered trademarks or unregistered trademarks are the property of their respective owners.

• • • • •

Depending on the Xinet software products licensed to you under the *Xinet End User Software License Agreement*, some or all of the copyright and warranty notices which follow may pertain to your installation.

— Portions of the software © Copyright 1999–2003 Apple Computer, Inc.

APPLE COMPUTER, INC. ("APPLE") MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THAT THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE REGARDING THE APPLE SOFTWARE. APPLE DOES NOT WARRANT, GUARANTEE OR MAKE ANY REPRESENTATIONS REGARDING THE USE OR THE RESULTS OF THE USE OF THE APPLE SOFTWARE IN TERMS OF ITS CORRECTNESS, ACCURACY, RELIABILITY, CURRENTNESS OR OTHERWISE. THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE APPLE SOFTWARE IS ASSUMED BY YOU. THE EXCLUSION OF IMPLIED WARRANTIES IS NOT PERMITTED BY SOME STATES. THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. IN NO EVENT WILL APPLE, ITS DIRECTORS, OFFICERS, EMPLOYEES OR AGENTS BE LIABLE TO YOU FOR ANY CONSEQUENTIAL, INCIDENTAL OR INDIRECT DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, AND THE LIKE) ARISING OUT OF THE USE OR INABILITY TO USE THE APPLE SOFTWARE EVEN IF APPLE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU. APPLE'S LIABILITY TO YOU FOR ACTUAL DAMAGES FROM ANY CAUSE WHATSOEVER, AND REGARDLESS OF THE FORM OF THE ACTION (WHETHER IN CONTRACT, TORT (INCLUDING NEGLIGENCE), PRODUCT LIABILITY OR OTHERWISE), WILL BE LIMITED TO $50.

— Portions Copyright © 2000 Artifex Software Inc. All rights reserved. Portions Copyright © 1998, 2000 Aladdin Enterprises. All rights reserved. Portions Copyright © Soft Horizons. All rights reserved.

— Portions of this software include software whose copyright is owned by, or licensed from, MySQL AB and Sun Microsystems. © Copyright 1995-2008 MySQL AB, © Copyright 2000, MySQL Finland AB, © Copyright 2008-2009 Sun Microsystems, Inc

— QUARK IS NOT THE MANUFACTURER OF THIRD-PARTY SOFTWARE OR OTHER THIRD-PARTY HARDWARE (HEREINAFTER "THIRD-PARTY-PRODUCTS") AND SUCH THIRD-PARTY PRODUCTS HAVE NOT BEEN CREATED, REVIEWED, OR TESTED BY QUARK, THE QUARK AFFILIATED COMPANIES OR THEIR LICENSORS. (QUARK AFFILIATED COMPANIES SHALL MEAN ANY PERSON, BRANCH, OR ENTITY CONTROLLING, CONTROLLED BY OR UNDER COMMON CONTROL

WITH QUARK OR ITS PARENT OR A MAJORITY OF THE QUARK SHAREHOLDERS, WHETHER NOW EXISTING OR FORMED IN THE FUTURE, TOGETHER WITH ANY PERSON, BRANCH, OR ENTITY WHICH MAY ACQUIRE SUCH STATUS IN THE FUTURE.) QUARK, THE QUARK AFFILIATED COMPANIES AND/OR THEIR LICENSORS MAKE NO WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING THE QUARK PRODUCTS/SERVICES AND/OR THIRD-PARTY PRODUCTS/SERVICES, THEIR MERCHANTABILITY, OR THEIR FITNESS FOR A PARTICULAR PURPOSE. QUARK, THE QUARK AFFILIATED COMPANIES AND THEIR LICENSORS DISCLAIM ALL WARRANTIES RELATING TO THE QUARK PRODUCTS/SERVICES AND ANY THIRD-PARTY PRODUCTS/SERVICES. ALL OTHER WARRANTIES AND CONDITIONS, WHETHER EXPRESS, IMPLIED, OR COLLATERAL, AND WHETHER OR NOT, MADE BY DISTRIBUTORS, RETAILERS, XTENSIONS DEVELOPERS OR OTHER THIRD-PARTIES ARE DISCLAIMED BY QUARK, THE QUARK AFFILIATED COMPANIES AND THEIR LICENSORS, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF NON-INFRINGEMENT, COMPATIBILITY, OR THAT THE SOFTWARE IS ERROR-FREE OR THAT ERRORS CAN OR WILL BE CORRECTED. THIRD-PARTIES MAY PROVIDE LIMITED WARRANTIES AS TO THEIR OWN PRODUCTS AND/OR SERVICES, AND USERS MUST LOOK TO SAID THIRD-PARTIES FOR SUCH WARRANTIES, IF ANY. SOME JURISDICTIONS, STATES OR PROVINCES DO NOT ALLOW LIMITATIONS ON IMPLIED WARRANTIES, SO THE ABOVE LIMITATION MAY NOT APPLY TO PARTICULAR USERS. IN NO EVENT SHALL QUARK, THE QUARK AFFILIATED COMPANIES, AND/OR THEIR LICENSORS BE LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL, CONSEQUENTIAL, OR PUNITIVE DAMAGES, INCLUDING, BUT NOT LIMITED TO, ANY LOST PROFITS, LOST TIME, LOST SAVINGS, LOST DATA, LOST FEES, OR EXPENSES OF ANY KIND ARISING FROM INSTALLATION OR USE OF THE QUARK PRODUCTS/SERVICES, IN ANY MATTER, HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY. IF, NOTWITHSTANDING THE FOREGOING, QUARK, THE QUARK AFFILIATED COMPANIES AND/OR THEIR LICENSORS ARE FOUND TO HAVE LIABILITY RELATING TO THE QUARK PRODUCTS/SERVICES OR THIRD-PARTY PRODUCTS/SERVICES, SUCH LIABILITY SHALL BE LIMITED TO THE AMOUNT PAID BY THE USER TO QUARK FOR THE SOFTWARE/SERVICES AT ISSUE (EXCLUDING THIRD-PARTY PRODUCTS/SERVICES), IF ANY, OR THE LOWEST AMOUNT UNDER APPLICABLE LAW, WHICHEVER IS LESS. THESE LIMITATIONS WILL APPLY EVEN IF QUARK, THE QUARK AFFILIATED COMPANIES, THEIR LICENSORS AND/OR THEIR AGENTS HAVE BEEN ADVISED OF SUCH POSSIBLE DAMAGES. SOME JURISDICTIONS, STATES OR PROVINCES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS LIMITATION OR EXCLUSION MAY NOT APPLY. ALL OTHER LIMITATIONS PROVIDED UNDER APPLICABLE LAW, INCLUDING STATUTES OF LIMITATION, SHALL CONTINUE TO APPLY. IN THE EVENT ANY OF THESE PROVISIONS ARE OR BECOME UNENFORCEABLE UNDER APPLICABLE LAW, SUCH PROVISION SHALL BE MODIFIED OR LIMITED IN ITS EFFECT TO THE EXTENT NECESSARY TO CAUSE IT TO BE ENFORCEABLE. USE OF THE QUARK PRODUCTS IS SUBJECT TO THE TERMS OF THE END USER LICENSE AGREEMENT OR OTHER APPLICABLE AGREEMENTS FOR SUCH PRODUCT/SERVICE. IN THE EVENT OF A CONFLICT BETWEEN SUCH AGREEMENTS AND THESE PROVISIONS THE RELEVANT AGREEMENTS SHALL CONTROL.

— Tag Image File Format (TIFF) Library © Copyright 1988-1997 Sam Leffler. © Copyright 1991-1997 Silicon Graphics, Inc. Permission to use, copy, modify, distribute and sell the Tag Image File Format Library software and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notices and this permission notice appear in all copies of the software and related documentation, and (ii) the names of Sam Leffler and Silicon Graphics may not be used in any advertising or publicity relating to the software without the specific, prior written permission of Sam Leffler and Silicon Graphics. THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

— XMP portions Copyright © 1999-2008, Adobe Systems Incorporated. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of Adobe Systems Incorporated, nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

— This software is based in part on the work of the Independent JPEG Group.

— Portions of the software, including *ndbm(3)*, Copyright © 1983, Regents of the University of California. All rights reserved. The Berkeley software License Agreement specifies the terms and conditions for redistribution.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of the Regents of the University of California nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS OF THE UNIVERSITY OF CALIFORNIA "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OF THE UNIVERSITY OF CALIFORNIA BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

— This software includes portions of the Darwin open source project.

— The *curl* and *libcurl* portions of this software © Copyright 2000, Daniel Stenberg, daniel@haxx.se. All rights reserved. Permission is hereby granted, free of charge, to any person obtaining a copy of *curl* and *libcurl* and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

— Portions of this software © Copyright 2003 Joe Penn.

— PHP and the PHP logo designs are copyright © 2001-2009 The PHP Group. All rights reserved. Portions of this software © Copyright 2001–2003 PHP Application Tools. Portions of this software © Copyright 1999–2009 The PHP Group. This product includes PHP software, freely available from http://www.php.net/software/. Use of the PHP software provided in the Xinet WebNative Portal software distribution is governed by *The PHP License, version 3.01*, below:

The PHP License, version 3.01

Copyright (c) 1999 - 2009 The PHP Group. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name “PHP” must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact group@php.net.

4. Products derived from this software may not be called “PHP”, nor may “PHP” appear in their name, without prior written permission from group@php.net. You may indicate that your software works in conjunction with PHP by saying “Foo for PHP” instead of calling it “PHP Foo” or “phpfoo”

5. The PHP Group may publish revised and/or new versions of the license from time to time. Each version will be given a distinguishing version number. Once covered code has been published under a particular version of the license, you may always continue to use it under the terms of that version. You may also choose to use such covered code under the terms of any subsequent version of the license published by the PHP Group. No one other than the PHP Group has the right to modify the terms applicable to covered code created under this License.

6. Redistributions of any form whatsoever must retain the following acknowledgment: “This product includes PHP software, freely available from <http://www.php.net/software/>”.

THIS SOFTWARE IS PROVIDED BY THE PHP DEVELOPMENT TEAM ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE PHP DEVELOPMENT TEAM OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software consists of voluntary contributions made by many individuals on behalf of the PHP Group. The PHP Group can be contacted via Email at group@php.net. For more information on the PHP Group and the PHP project, please see <http://www.php.net>. PHP includes the Zend Engine, freely available at <http://www.zend.com>.

— Portions of this software make use of the Zend Optimizer, Copyright © 1998-2009, by Zend Technologies Ltd. Use of the Zend software provided in the Xinet WebNative Portal software distribution is governed by the license agreement below:

ZEND OPTIMIZER - ZEND LICENSE AGREEMENT

ZEND TECHNOLOGIES LTD. (“ZEND”) SOFTWARE LICENSE AGREEMENT (“AGREEMENT”)

IMPORTANT: READ THESE TERMS CAREFULLY BEFORE INSTALLING THE SOFTWARE KNOWN AS THE “ZEND OPTIMIZER,” AS INSTALLED BY THIS INSTALLATION PROCESS, IN MACHINE-EXECUTABLE FORM ONLY, AND ANY RELATED DOCUMENTATION (COLLECTIVELY, THE “SOFTWARE”) BY INSTALLING, OR OTHERWISE USING THIS SOFTWARE, YOU (THE “LICENSEE”) ACKNOWLEDGE THAT YOU HAVE READ THIS AGREEMENT, AND THAT YOU AGREE TO BE BOUND BY ITS TERMS AND CONDITIONS. IF YOU DO NOT AGREE TO ALL OF THE TERMS AND CONDITIONS OF THIS AGREEMENT, YOU ARE NOT AN AUTHORIZED USER OF THE SOFTWARE AND IT IS YOUR RESPONSIBILITY TO EXIT THIS INSTALLATION PROGRAM WITHOUT INSTALLING THE SOFTWARE, OR TO DELETE THE SOFTWARE FROM YOUR COMPUTER.

1. License. Subject to the terms and conditions of this Agreement, including, without limitation, Section 2 hereof, Zend hereby grants to Licensee, during the Term (as defined below), a limited, a non-exclusive license (the “License”) to: (i) install and operate the Software on a computer or a computer network owned or operated by Licensee; (ii) make copies of the Software; and (iii) sublicense and distribute a limited, non-exclusive sublicense to install, use and sublicense such copies of the Software, provided that any sub-license granted hereunder shall be subject to the limitations and restrictions set forth in this Agreement.

2. Restrictions. Except as otherwise expressly set forth herein, Licensee or any of its sub-licensees shall not: (a) translate or decompile, or create or attempt to create, by reverse engineering or otherwise, the source code form from the object code supplied hereunder; (b) modify, adapt, translate or create a derivative work from the Software; (c) remove any proprietary notices, labels, or marks on the Software.

3. Termination. This Agreement and the License hereunder shall be in effect from and after the date Licensee installs the Software on a computer in accordance with the terms and conditions hereof and shall continue perpetually unless terminated in accordance with this Section 3. This Agreement shall be automatically terminated upon any breach by Licensee of any term or condition of this Agreement. Such period shall be referred to herein as the “Term”. Within five (5) business days of any such termination, Licensee shall return the Software to Zend (or, at Zend's sole discretion and only at Zend's direction, destroy the Software and certify in writing to Zend that said Software has been destroyed). Upon return of the Software or upon receipt of notice of the destruction of the Software, as appropriate, this Agreement shall terminate and Zend shall have no further obligations to Licensee. Articles 2, 4, 5, 6, 7, 8 and 10 hereof shall survive the expiration or termination of this Agreement for any reason.

4. Intellectual Property Rights. Licensee hereby acknowledges and agrees that Zend or its licensors own and retain all rights, title, and interest in and to the Software, regardless of the form or media in or on which the original or other copies may subsequently exist including, without limitation, all copyrights, trademarks, patents and trade secret rights inherent therein or appurtenant thereto. This Agreement shall not constitute a sale of the Software and no title or proprietary rights to the Software are transferred to the Licensee hereby. Licensee acknowledges that the Software is a unique, confidential and valuable asset and trade secret of Zend or its licensors, and Zend or its licensors shall have the right to obtain all equitable and legal redress which may be available to it for the breach or threatened breach of this Agreement including, without limitation, injunctive relief.

5. Warranty; Disclaimer. THE SOFTWARE IS BEING LICENSED HEREUNDER WITH NO WARRANTY WHATSOEVER. LICENSEE ACKNOWLEDGES THAT ITS USE OF THE SOFTWARE IS AT ITS OWN RISK. THE SOFTWARE IS PROVIDED

SOLELY ON AN “AS-IS” BASIS. ZEND AND ITS LICENSORS MAKE, AND LICENSEE RECEIVES, NO WARRANTIES, EXPRESS, IMPLIED, OR OTHERWISE. ZEND EXPRESSLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, FITNESS FOR A PARTICULAR PURPOSE, AND/OR NONINFRINGEMENT. ZEND DOES NOT WARRANT THAT THE OPERATION OF THE SOFTWARE SHALL BE OPERABLE, UNINTERRUPTED OR ERROR FREE OR THAT IT WILL FUNCTION OR OPERATE IN CONJUNCTION WITH ANY OTHER PRODUCT.

6. No Liability. IN NO EVENT SHALL ZEND OR ITS LICENSORS BE LIABLE FOR ANY DIRECT, CONSEQUENTIAL, EXEMPLARY, SPECIAL, INDIRECT, INCIDENTAL OR PUNITIVE DAMAGES OR LOST PROFITS (EVEN IF THEY HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES) ARISING FROM OR RELATING TO THIS AGREEMENT OR THE SUBJECT MATTER HEREOF, INCLUDING WITHOUT LIMITATION, DAMAGES ARISING FROM LOSS OF DATA, OR ANY OTHER DAMAGES, HOWEVER CAUSED, AND UNDER ANY THEORY OF LIABILITY. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES OR LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, AND IN SUCH CASE AND ONLY TO THE EXTENT PROHIBITED BY APPLICABLE LAW, THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO LICENSEE. WITHOUT LIMITATION OF THE FOREGOING, UNDER NO CIRCUMSTANCES SHALL LICENSOR'S TOTAL AGGREGATE LIABILITY TO LICENSEE AND ANY OF ITS SUB-LICENSEES (REGARDLESS OF THE LENGTH OF THE TRAIL OF SUBLICENSES) FOR ANY AND ALL CLAIMS ARISING HEREUNDER, INCLUDING WITHOUT LIMITATION ARISING OUT OF YOUR USE OF THE SOFTWARE, EXCEED THE SUM OF $100.

7. Indemnity. Licensee will, at its own expense, defend any action brought by a third party against Zend to the extent that such action is based on a claim arising from or relating to: (a) Licensee's use of the Software, (ii) any distribution of the Software by Licensee or by any of or by any sub-licensee, regardless of privity of contract and regardless of the length of the trail of sublicenses, (iii) any claims based upon warranties, guarantees or representations made by Licensee or any of its employees, agents or sub-licensees; or (iv) any use of the Software by any of the foregoing sub-licensees. Zend shall have the exclusive right to control such defense. In no event shall Licensee settle any such claim, lawsuit or proceeding without Zend's prior written approval.

8. U.s. Government restricted rights. The Software provided hereunder is a “commercial item,” as that term is defined in 48 C.F.R. 2.101, consisting of “commercial computer software” and “commercial computer software documentation” as such terms are used in 48 C.F.R. 12.212. Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4, the Software made available to the United States of America, its agencies and/or instrumentalities, is provided with only those rights set forth in this Agreement. Use, duplication or disclosure of the Software by the government is subject to the restrictions as set forth in subparagraph (c)(1) and (2) of the Commercial Computer Software-Restricted Rights clause at 48 C.F.R. 52.227-19, as amended, or any successor regulations thereto.

9. Export and Import Restrictions. Licensee shall be solely responsible for obtaining any required regulatory approvals for exportation, importation and/or distribution of the Software.

10. Miscellaneous. This Agreement is made in and shall be governed by the laws of the State of Israel, excluding choice of law principles. Venue for all proceedings shall be Tel Aviv, Israel. Notwithstanding the foregoing, Zend shall have the right to apply to any court of competent jurisdiction for injunctive or other relief. The United Nations Convention for the International Sale of Goods shall not apply. There is no relationship of agency, partnership, joint venture, employment or franchise between the parties. Licensee or its sub-licensees do not have the authority to bind Zend or to incur any obligation its behalf or to represent itself as Zend's agent or in any way that might result in confusion as to the fact that the parties are separate and distinct entities. The section headings herein are provided for convenience only and have no substantive effect on the construction of this Agreement. If any provision of this Agreement is held to be unenforceable, this Agreement shall be construed without such provision. The failure by Zend to exercise any right hereunder shall not operate as a waiver of Zend's right to exercise such right or any other right in the future. This Agreement may be amended only by a written document executed by a duly authorized representative of each of the parties. This Agreement constitutes the complete and exclusive statement of the terms and agreement between Zend and Licensee and supersedes all prior representations, understandings and communications, oral or written, between the parties with respect thereto, including memoranda of agreement. In the event legal action is taken by Zend or its licensors to enforce any provision of this Agreement, all costs and expenses, including reasonable attorneys' fees and expenses, incurred by Zend or its licensors shall be paid by Licensee, in addition to other damages to which Zend or its licensors may be lawfully entitled. Licensee agrees that it will not refer to the Software or to the existence of this Agreement nor will it use Zend's name in any press releases, advertising, marketing or other materials without Zend's advanced written consent in each instance. Any notice provided by Licensee to Zend pursuant to this Agreement shall be in writing to Zend Technologies Ltd. at P.O. Box 3619, Ramat Gan 52136, Israel, and shall be deemed given (i) if by hand delivery, upon receipt thereof; (ii) if mailed, three (3) days after deposit in the U.S. mails, postage prepaid, certified mail, return receipt requested; or (iii) if by next day delivery service, upon such delivery. Any notice provided by Zend to Licensee pursuant to this Agreement shall be sent to the e-mail or mailing address provided by Licensee upon registering on Zend's Web site, and shall be deemed given (i) if by hand delivery, upon receipt thereof; (ii) if mailed, three (3) days after deposit in the U.S. mails, postage prepaid, certified mail, return

receipt requested; or (iii) if by next day delivery service, upon such delivery. Either party may change its address by giving written notice to the other party.

— Portions of this software include software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/). Copyright © 1998–2008 The OpenSSL Project. All rights reserved. The OpenSSL Toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL please contact openssl-core@openssl.org.

OpenSSL License

Copyright (c) 1998-2008 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment: “This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).”

4. The names “OpenSSL Toolkit” and “OpenSSL Project” must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.

5. Products derived from this software may not be called “OpenSSL” nor may “OpenSSL” appear in their names without prior written permission of the OpenSSL Project.

6. Redistributions of any form whatsoever must retain the following acknowledgment: “This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).”

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Original SSLeay License

Copyright © 1995–1998 Eric Young (eay@cryptsoft.com). All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com). The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and non-commercial use as long as the following conditions are aheared to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young’s, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement: “This product includes cryptographic software written by Eric Young

(eay@cryptsoft.com).” The word “cryptographic” can be left out if the rouines from the library being used are not cryptographic related :-).

4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: “This product includes software written by Tim Hudson (tjh@cryptsoft.com)”

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The licence and distribution terms for any publicly available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]

— Portions of this software governed by the GNU *Lesser General Public License* available from http://www.gnu.org/licenses/lgpl.html.

— Use of a modified '*ffmpeg*' library with the Xinet Video software is governed by the GNU *General Public License version 2 or later*. Running */usr/etc/venture/bin/ffmpeg -L* from the command prompt on UNIX systems and *C:\Program Files\Xinet\Venture\ffmpeg.exe -L* on Windows systems provides *FFmpeg* copyright information. As Xinet's PC Connectivity software package contains modified software from *samba.org*, Xinet Samba modifications and VFS Module are released and distributed as open-source under the GNU *GPL* terms as well. Xinet will make the full source code of its modifications available to any interested parties under the terms of the GNU *GPL*; the complete GPL License appears below.

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation, Inc. <http://fsf.org/>. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains,

we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the

unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

* a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

* b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

* c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

* d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

* a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

* b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

* c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

* d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

* e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in

certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

* a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

* b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

* c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

* d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

* e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

* f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Down Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a

right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's “contributor version”.

A contributor's “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to down recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is

the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

Xinet Trademark Acknowledgements, Attributions, and Credit Notices

Revision Date: February 25, 2010

#####

ENDBENUTZER-LIZENZVERTRAG FÜR XINET-SOFTWARE

Gültig nur für Produkte, die in Deutschland lizensiert werden.

This applies only to products licensed in Germany.

Wichtig bitte sorgfältig lesen: Dieser XINET-Endbenutzer-Lizenzvertrag ist ein rechtsgültiger Vertrag zwischen Ihnen (entweder als natürliche oder juristischer Person) und der XINET Inc. für ein XINET-Software-Produkt. Das Software-Produkt umfaßt Computer-Software sowie möglicherweise dazugehörige Medien, gedruckte Materialien und „Online“ oder elektronische Dokumentation („Software-Produkt“). Indem Sie das Software-Produkt installieren, kopieren oder anderweitig verwenden, erklären Sie sich einverstanden, durch die Bestimmungen dieses Lizenzvertrages gebunden zu sein. Falls Sie diesen Bestimmungen nicht zustimmen, sind Sie nicht berechtigt, das Software-Produkt zu installieren oder zu verwenden; Sie können es jedoch gegen Rückerstattung des Kaufpreises an den Händler zurückgeben, von dem Sie es erworben haben.

Das Software-Produkt wird sowohl durch Urheberrechtsgesetze und internationale Urheberrechtsverträge geschützt, als auch durch andere Gesetze und Vereinbarungen über geistiges Eigentum. Das Software-Produkt wird lizenziert, nicht verkauft.

1. Lizenzeinräumung. XINET räumt Ihnen unter der Einschränkung der Regelung in Ziff. 2.f) dieses Vertrages das zeitlich unbegrenzte, nicht ausschließliche und nicht übertragbare Recht ein, das Software-Produkt im vertragsgemäßen Umfang zu nutzen. Sie sind berechtigt, eine Kopie des Software-Produkts auf einem einzelnen Computer zu installieren und zu verwenden. Ferner dürfen Sie eine Sicherungskopie anfertigen.

2. Weitere Rechte und Einschränkungen

a) Nicht zum Weiterverkauf bestimmte Software. Falls Ihnen das Software-Produkt als Demonstrationsmaterial überlassen wird, sind Sie ungeachtet anderer Abschnitte dieses Lizenzvertrages, nicht berechtigt, das Software-Produkt weiterzuverkaufen oder auf andere Weise gegen einen Gegenwert oder unentgeltlich zu übertragen. Sollten Sie die Hardware auf welcher die Software installiert ist veräußern, sind Sie verpflichtet, sämtliche Kopien des Softwareprodukts zu zerstören bzw. an XINET zurückzugeben.

b) Beschränkung im Hinblick auf Zurückentwicklung (Reverse Engineering), Dekompilierung und Disassemblierung. Sie sind nicht berechtigt, das Software-Produkt zurückzuentwickeln (Reverse Engineering), zu dekompilieren oder zu disassemblieren, es sei denn und nur insoweit wie das anwendbare Recht, ungeachtet dieser Beschränkung, dieses ausdrücklich gestattet.

c) Trennung von Komponenten. Das Software-Produkt wird als einzelnes Produkt lizenziert. Sie sind nicht berechtigt, dessen Komponenten für die Verwertung auf mehr als einem Computer zu trennen.

d) Vermietung. Sie sind nicht berechtigt, das Software-Produkt zu vermieten, zu verleasen oder zu verleihen.

e) Serviceleistungen. XINET bietet Ihnen möglicherweise Serviceleistungen in Verbindung mit dem Software-Produkt („Serviceleistungen“). Die Serviceleistungen können entsprechend den XINET-Bestimmungen und -Programmen, die im Benutzerhandbuch, der „Online“-Dokumentation und/oder anderen von XINET zur Verfügung gestellten Materialien beschrieben sind, genutzt werden. Jeder ergänzende Software-Code, der Ihnen als Teil der Serviceleistungen zur Verfügung gestellt wird, wird als Bestandteil des Software-Produkts betrachtet und unterliegt den Bestimmungen und Bedingungen dieses Lizenzvertrages. XINET ist berechtigt, die technischen

Daten, die Sie XINET als Teil der Serviceleistungen zur Verfügung stellen, für geschäftliche Zwecke, einschließlich Produktunterstützung und -entwicklung, zu verwenden. XINET verpflichtet sich, solche technische Daten ausschließlich anonym im Sinne des Datenschutzes zu verwenden.

f) Kündigung. Unbeschadet sonstiger Rechte ist XINET berechtigt, diesen Lizenzvertrag zu kündigen, sofern Sie gegen die Bestimmungen und Bedingungen dieses Lizenzvertrages verstoßen. In einem solchen Fall sind Sie verpflichtet, sämtliche Kopien des Software-Produkts und alle seine Komponenten zu vernichten. Durch Zerstörung der Software einschließlich seiner Kopie und damit verbundenen Dokumentation kündigen Sie selbst den Lizenzvertrag.

3. Urheberrecht. Eigentum und Urheberrecht an dem Software-Produkt (einschließlich, aber nicht beschränkt auf Bilder, Fotografien, Animationen, Video, Audio, Musik, Text und „Applets“, die in dem Software-Produkt enthalten sind), den gedruckten Begleitmaterialien und jeder Kopie des Software-Produkts liegen bei XINET. Das Software-Produkt ist durch Urheberrechtsgesetze und internationale Urheberrechtsbestimmungen geschützt. Aus diesem Grund sind Sie verpflichtet, das Software-Produkt wie jedes andere durch das Urheberrecht geschützte Material zu behandeln, mit der Ausnahme, daß Sie berechtigt sind, das Software-Produkt auf einem einzelnen Computer zu installieren, vorausgesetzt, Sie bewahren das Original ausschließlich für Sicherungs- und Archivierungszwecke auf und schützen es vor unerlaubter Offenbarung. Sie sind nicht berechtigt, dass das Software-Produkt begleitende gedruckte Material zu vervielfältigen.

4. Gewährleistungen

a) XINET und ihre Vertragshändler weisen darauf hin, daß es nicht möglich ist, Software so zu entwickeln, dass sie für alle Anwendungsbedingungen fehlerfrei ist. Dafür übernimmt XINET für einen Zeitraum von 30 Tagen ab Auslieferung an Sie die Garantie, dass das Software-Produkt im wesentlichen gem. den beiliegenden bedruckten Materialien arbeitet.

b) Darüber hinaus stehen XINET und ihre Vertragshändler für Mängel, die bei der Übergabe des Software-Produkts vorhanden sind, während einer Gewährleistungsfrist von 6 Monaten gem. folgenden Regeln ein.

c) Als Mängel im Sinn der Ziffer 4.b) gelten Abweichungen des Software-Produkts von den beiliegenden bedruckten Materialien, soweit diese Abweichungen die Tauglichkeit des Software-Produkts beeinträchtigen. Eine unerhebliche Minderung der Brauchbarkeit bleibt außer Betracht.

d) Erweist sich das Software-Produkt als nicht brauchbar oder fehlerhaft, werden XINET und ihre Vertragshändler diese Mängel binnen angemessener Frist beheben. Gelingt ein derartiger Nachbesserungsversuch nicht innerhalb angemessener Frist und schlägt er auch innerhalb einer weiteren, von Ihnen angemessen gesetzten Frist fehl, und stellen XINET und ihre Vertragshändler keine Umgehungslösung gem. Ziff. 4.e) zur Verfügung, so stehen Ihnen die gesetzlichen Gewährleistungsrechte zu. Sie können dann den Vergütungsanspruch herabsetzen (mindern) oder den Vertrag rückgängig machen (wandeln).

e) XINET und ihre Vertragshändler sind berechtigt, einen eventuell auftretenden Fehler zu umgehen, wenn der Fehler selbst nur durch unverhältnissemäßigen Aufwand zu beseitigen ist und dadurch der Einsatz des Software-Produkts nicht erheblich leidet.

f) XINET und ihre Vertragshändler haften nicht, wenn der Fehler unseres Produkts auf einem Unfall, Mißbrauch oder fehlerhaften Anwendung beruht.

g) XINET und ihre Vertragshändler haften nicht für Folgeschäden und im übrigen nur nach den Bestimmungen der Ziff. 5. Weitergehende Gewährleistungsansprüche sind ausgeschlossen. Darüber hinausgehende Leistungen bieten XINET und ihre Vertragshändler im Zusammenhang mit Support-Leistungen an, soweit nicht ein zusätzlicher Vertrag über Support-Leistungen abgeschlossen wurde.

5. Haftung

a) XINET und ihre Vertragshändler haften für anfängliches Unvermögen, Verzug und Unmöglichkeit auf solche Schäden begrenzt, die aufgrund der vertraglichen Verwendung typisch und vorhersehbar sind. In einem derartigen Fall ist die Haftung auf den Betrag, den Sie für das Software-Produkt bezahlt haben, begrenzt.

b) Im übrigen haften XINET und ihre Vertragshändler unbeschränkt nur für Schäden, die durch Vorsatz oder grobe Fahrlässigkeit ihrer gesetzlichen Vertreter und/oder ihrer leitenden Angestellten verursacht wurden. Für Schäden,

die durch ein Verschulden sonstiger Mitarbeiter und Erfüllungsgehilfen verursacht worden sind, haften XINET und ihre Vertragshändler nur im Umfang der Haftung nach Ziff. 5.a).

c) Für leichte Fahrlässigkeit haften XINET und ihre Vertragshändler nur, sofern eine Pflicht verletzt wird, deren Einhaltung für die Erreichung des Vertragszwecks von besonderer Bedeutung ist (Kardinalpflicht). Bei Verletzung einer Kardinalpflicht gilt Ziff. 5.a) entsprechend.

d) Den besonderen Risiken der Überlassung des Software-Produkts entsprechend wird die Haftung für Folgeschäden und entgangenen Gewinn ausgeschlossen, soweit ein derartiger Schaden nicht auf Rechtsmängel, das Fehlen zugesicherter Eigenschaften, Vorsatz oder grobe Fahrlässigkeit der gesetzlichen Vertreter oder leitenden Angestellten zurückzuführen ist. XINET und ihre Vertragshändler haften nur für die Wiederherstellung von Daten, soweit Sie regelmäßig und gefahr-entsprechend Sicherungskopien angefertigt und sichergestellt haben, daß die Daten aus diesen Sicherungskopien mit vertretbaren Aufwand rekonstruiert werden können. Eine darüber hinausgehende Haftung für Datenverluste ist ausgeschlossen.

e) Die vorstehenden Regelungen gelten auch zugunsten der Mitarbeiter und Erfüllungsgehilfen von XINET und ihren Vertragshändlern. Die Haftungsbeschränkungen gelten nicht für Ansprüche nach dem Produkthaftungsgesetz.

6. Anwendbares Recht. Wenn Sie unser Produkt in Deutschland gekauft haben, unterliegt der Lizenzvertrag deutschem Recht.

Appendix B

Xinet software development license agreement

This license agreement is applicable to members of the Xinet Developer Program who are running non-production Software Development Kit licenses.

This software license agreement (this "Agreement") is a legal agreement between you (either as an individual or a business entity) and Xinet, Inc. a California corporation ("Xinet") for the use of the Xinet software issued to you under this Agreement, including certain beta and other pre-release versions thereof, certain updates and upgrades thereof, and related documentation (collectively, the "Software").

DO NOT use the Software until you have carefully read this Agreement. BY USING THIS SOFTWARE YOU ACCEPT ALL THE TERMS AND CONDITIONS OF THIS AGREEMENT. If you do not agree with the terms and conditions of this Agreement, promptly return the unused and unkeyed Software to the place from which you obtained it for a full refund.

1. GRANT OF LICENSE. Xinet grants to you a nonexclusive, nontransferable, time-limited license (without the right to sublicense) to use the Software only for the purpose of developing or testing application programs that run with Xinet's software products. You may use the Software only on one computer system, and you may make one copy of the Software solely for backup purposes. You must ensure that the same copyright and other proprietary rights notices appearing in the original copy you receive from Xinet also appears in the backup copy. You may not use the Software in production environments or for production purposes. You may not transfer, sell, assign, rent, sublicense or distribute any copies of the Software to others. Xinet reserves all rights not specifically granted to you.
2. PROPRIETARY RIGHTS. As a licensee, you own the media on which the Software is originally recorded. The Software is copyrighted by and proprietary to Xinet and its suppliers. Xinet and its suppliers retain title and ownership of all copies of the Software. Should Xinet provide you with any additional code, programs, or libraries (collectively, the "Materials") in connection with your use of the Software or any support services, you may use these Materials only in accordance with this Agreement. The Materials are copyrighted by and proprietary to Xinet and its suppliers and Xinet and its suppliers retain title and ownership of all copies of the Materials. Should Xinet authorize your distribution of any such Materials, you may distribute the Materials only as incorporated into your own software application programs in machine-readable code and only for use on or in conjunction with servers that are running Xinet's commercial software products in accordance with Xinet's End User Software License Agreement. The nonexclusive license set forth in this Agreement is not a sale of the Software or any copy. You agree to hold the Software in confidence and to take all reasonable steps to prevent unauthorized disclosure. Subject to the foregoing, you own all right, title, and interest in and to the software that you develop with the assistance of the Software to the extent that the software you develop is a copyrightable work.
3. NO OTHER RIGHTS. Except as stated above, this Agreement does not grant you any rights to patents, copyrights, trade secrets, trade names, trademarks (whether registered or unregistered), or any other rights, franchises, or licenses in respect of the Software. You MAY NOT MODIFY, TRANSLATE, REVERSE ENGINEER, DISASSEMBLE, OR DECOMPILE THE SOFTWARE OR ANY COPY, IN WHOLE OR IN PART. Xinet reserves the right to alter Software or any other Materials supplied to you at any time, under its sole discretion, and you acknowledge that Xinet is under no obligation to maintain existing Software or other Materials that any applications, programs, or software developed by you may rely on. Unless otherwise agreed to in writing by Xinet, you may not make any statements that any application or software developed by you is "certified" or that its performance is guaranteed by Xinet. You acknowledge that Xinet is free to enter into any area of development, including areas which may compete with applications, software, or products developed by you, and Xinet has no obligation to inform you of Xinet's development plans at any time.
4. TERM. This Agreement is effective until (i) the expiration of all software license keys provided by Xinet to you for the purpose of allowing you to use the Software, or (ii) as earlier terminated as set forth in this Section 4. Xinet may terminate this Agreement immediately for any reason; provided, however, if Xinet terminates this Agreement, and you have not breached any term or condition of this Agreement, then Xinet will pay to you a pro-rata refund of fees paid by you based on the number of days remaining in the term for such payment. Upon any termination or expiration of this Agreement, all license rights shall immediately terminate.
5. WARRANTY DISCLAIMER. THE SOFTWARE AND ANY RELATED SUPPORT, SERVICES, AND MATERIALS PROVIDED TO YOU BY XINET ARE PROVIDED "AS IS" WITH NO WARRANTIES OF ANY KIND. XINET DOES NOT MAKE ANY WARRANTIES OF ANY KIND WITH RESPECT TO THE SOFTWARE OR ANY RELATED SUPPORT, SERVICES, OR MATERIALS, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE.
6. LIMITATION OF LIABILITY. IN NO EVENT SHALL XINET'S AGGREGATE LIABILITY UNDER THIS AGREEMENT, OR FOR BREACH HEREOF, UNDER ANY CONTRACT, TORT (INCLUDING NEGLIGENCE), STRICT LIABILITY OR OTHER LEGAL OR EQUITABLE THEORY, (i) EXCEED AMOUNTS PAID OR PAYABLE BY YOU TO XINET IN THE PAST TWELVE (12) MONTHS; OR (ii) INCLUDE ANY SPECIAL, INDIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES OR LOSS OF PROFITS OR PROSPECTIVE BUSINESS INCOME; OR (iii) INCLUDE ANY COST OF PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY OR SERVICES. THIS LIMITATION WILL APPLY EVEN IF XINET HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES AND NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY PROVIDED HEREIN.
7. CONFIDENTIALITY. If you are using any beta or other pre-release version of the Software (the "Beta Software"), you may not disseminate or in any way disclose the Beta Software, or any part thereof, to any person, firm or business; provided, however, that you may disclose the Beta Software to your employees and consultants who need to access such Beta Software and who have previously agreed to be bound by a confidentiality obligation consistent with the obligation set forth in this Section 7. You will treat the Beta Software with the same degree of care as you accord to your own confidential information, but in any event with no less than reasonable care.
8. EXPORT. You acknowledge that the laws and regulations of the United States restrict the export and re-export of the Software. You agree that you will not export or re-export the Software or media in any form without the appropriate United States and foreign government approval.
9. CHOICE OF LAW. This Agreement will be governed by the laws of the State of California, without regard to conflicts of law provisions thereof. Any action to enforce any of the rights herein or interpret any provision hereof shall be brought and adjudicated in a state or federal court located in Alameda County, California, and each party irrevocably submits to the jurisdiction of any such court.
10. U.S. GOVERNMENT RESTRICTED RIGHTS LEGEND. The Software (including the documentation) is provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(l)(ii) of the Rights in Technical Data and Computer Software clause at 252.2277013.
11. INTEGRATION. You acknowledge that you have read this Agreement, understand it, and that by installing the Software you agree to be bound by its terms and conditions. You further agree that it is the complete and exclusive statement of the agreement between Xinet and you which supersedes any proposal or prior agreement, oral or written, and any other communications between Xinet and you relating to the subject matter of this Agreement. No variation of the terms of the Agreement or any different terms will be enforceable against Xinet unless Xinet gives its express consent, including an express waiver of the terms of this Agreement, in writing signed by an officer of Xinet.

Index