WebMagick

Image Web Generator
December 1998 (Version 1.45)

by Bob Friesenhahn

1 Introduction

WebMagick creates HTML index files and imagemaps corresponding to a directory of image files. WebMagick is ideal for making large image collections available on the net or for just using your Web browser for browsing images.

Configuration options are derived from either the command line (options prefixed with '-', '–', or '+'), configuration file ($opt_ options), or internal defaults ($opt_ options). Some arguments (like X11 font specifications) seem to confuse PERL's getopt function. When this happens, try using the syntax `--option=arg` to avoid any confusion caused by a space.

The default configuration file name is '`.webmagickrc`'. Configuration files are written in PERL syntax and are sourced via PERL's eval statement. When WebMagick is executed, it first reads files listed in the @global_option_files array ('`/etc/webmagickrc`' and '`$HOME/.webmagickrc`' by default). When a directory is processed, WebMagick will process any '`.webmagickrc`' files found in the path from the server's root directory (specified by `$opt_rootpath` (see Section 3.1.1 [Server Root Path], page 6)) to the directory being processed. All configuration options are additive.

Evaluation order (inverse of precedence) is as follows :

1. Files listed in @global_option_files

   (default '`/etc/webmagickrc`' and '`$HOME/.webmagickrc`')
2. Command line options
3. Per directory .webmagickrc files.

   Webmagickrc files are evaluated in path order from server root to current directory. This allows sub-directories to "inherit" the configuration of parent directories.

Each directory is processed with items 1 & 2 already complete.

In general, WebMagick's defaults produce nice output. However, the options `$opt_prefixpath` (see Section 3.1.2 [Root Prefix], page 6), `$opt_rootpath` (see Section 3.1.1 [Server Root Path], page 6), `$opt_iconpath` (see Section 3.1.3 [Icon Location], page 6), `$opt_htimage` (see Section 5.1 [Imagemap Program], page 10), and `$opt_maptype` (see Section 5.2 [Imagemap Type], page 10) must be configured on a per-site basis so please read the documentation on these options and perform the required configuration before executing the program.

A key configuration/installation issue is that WebMagick depends on the image files being on the same filesystem as the icons. The reason for this limitation is that this allows WebMagick to determine the mapping between the physical disk and the server's URLs. If someone knows a better algorithm than this one, please let the author know.

By default the output files of WebMagick for the first page are as follows (second page replace '1' with '2', etc.):

`index.html`
: Main (default server index) HTML index file.

`indexjs.html`
: Javascript master index file (name currently hard-coded).

`.cache` Thumbnail cache directory

`.indexdir.html`
: HTML page that displays directory frame

`.indexjs.html`
: HTML file to load JavaScript source into browser.

`.index.pl`
: WebMagick status file (in PERL format). Re-generated per run. May be used by other programs to produce additional HTML output.

`.index.js`
: JavaScript variables file (similar to status file but in JavaScript).

`.index1.html`
: HTML page that displays thumbnail frame

`.index1.gif`
: Montaged images (thumbnails) in GIF format

`.index1.jpg`
: Montaged images (thumbnails) in JPEG format

`.index1.map`
: Server-side imagemap (clickable map) file

2 General

2.1 Display Help

`Variable:`
`$opt_help=1`

`Argument:`
`--help`

Display usage message.

2.2 Specify Source Directory

`Variable:`
$opt_srcdir=path

`Argument:`
–srcdir path

Image directory to process or start recursion at (default current directory). Specify this option (usually on the command line) if the directory you want to process is not the current directory. This simply causes WebMagick to change directories before starting processing.

2.3 Enable Verbose Output

`Variable:`
$opt_verbose=1

`Argument:`
–verbose

Normally WebMagick prints only the directory names for directories it processes. Selecting the verbose option displays the number of images and sub-directories in the directory as well as the number of pages to be generated, and the current page number. It will also show the current image index WebMagick is currently creating as a progress report.

2.4 Enable Debug Output

`Variable:`
$opt_debug=1

`Argument:`
–debug

Print debugging messages.

2.5 Sub-Directory Processing

2.5.1 Recurse Sub-Directories

`Variable:`
    $opt_recurse=1

`Argument:`
    –recurse

Recurse directory tree (default off). The default is to only process the current directory, generating appropriate links for sub-directories and the parent directory. Selecting this option causes WebMagick to recurse through sub-directories as well.

2.5.2 Prune Directory

`Variable:`
    $opt_prune=1

If `$opt_prune` is set in a directory's '`.webmagickrc`' file (default off), then sub-directories below that directory will not be processed by WebMagick when operating in the recursive mode. See Section 2.5.3 [Ignore Directory], page 4, for details on how to skip a directory rather than prune it.

2.5.3 Ignore Directory

`Variable:`
    $opt_ignore=1

If `$opt_ignore` is set in a directory's '`.webmagickrc`' file (default off), then that directory will not be processed (but will still be linked to). If recursion is enabled, then recursion will continue into any sub-directories. This provides support for subdirectories with HTML files which are generated by other means. also `$opt_indexname`.

2.6 Forcing Output To Be Generated

WebMagick uses algorithms similar in concept to those used by `make` to try to only do required processing. Occasionally due to modification of a parameter that WebMagick doesn't track or due to external modification of file modification times, WebMagick will skip doing an update that it should have done. In these cases, options are provided to force WebMagick to perform a task.

2.6.1 Force Generation Of Thumbnail Cache

`Variable:`
    $opt_forcecache=1

`Argument:`
    –forcecache

Force cache files to be generated (default off). Without this option, WebMagick will only re-generate cache files if the source file is newer. In order for this option to take effect, a montage must be generated and `$opt_cache` must be set to a non-zero value. The –forcemontage option be used to force montages.

2.6.2 Force Generation Of HTML Files

`Variable:`

$opt_forcehtml=1

`Argument:`

–forcehtml

Force HTML files to be generated (default off). Without this option, WebMagick will only re-generate HTML files as required.

2.6.3 Force Generation Of Montage

`Variable:`

$opt_forcemontage=1

`Argument:`

–forcemontage

Force montage (default false). Without this option, WebMagick will only re-generate montages as required.

3 Paths

3.1 URL Computation

3.1.1 Specify Server Root Path

`Variable:`

$opt_rootpath=path

`Argument:`

–rootpath path

Absolute path to server root directory (NCSA/Apache DocumentRoot) or the users Home page directory. This must be an absolute path (starting with '`/`') to the top of the WWW directory tree WebMagick will be processing in. If this value is specified incorrectly, WebMagick will generate incorrect URLs.

3.1.2 Specifying Root Prefix

`Variable:`

$opt_prefixpath=path

`Argument:`

–prefixpath path

The prefix to prepend to all URLs to specify the WWW location of `$opt_rootpath` (see Section 3.1.1 [Server Root Path], page 6). For directories under the WWW server's root directory ('`/`') this is a empty string (`""`). For a user's personal pages this is the URL path to their home pages (usually '`/~username`' equating to '`~/public_html`', but may be configured differently on your server).

If a relative URL can be computed then this option is not actually used. It however must be used if the WebMagick icon images are not in the same logical WWW directory tree as the directories WebMagick is processing. (See the next option).

3.1.3 Specifying WebMagick Icon Location

`Variable:`

$opt_iconpath=path

`Argument:`

–iconpath path

This is the relative path from the top of the WWW directory tree specified in `$opt_rootpath` (see Section 3.1.1 [Server Root Path], page 6). If this value is specified incorrectly then WebMagick will generate incorrect URLs for its icons.

4 File Names

4.1 Specifying Icon File Names

`Variable:`

%opt_icons=('id', 'filename', ... ;

This is a hash table of all the images used directly by WebMagick and located in the `$opt_iconpath` (see Section 3.1.3 [Icon Location], page 6) directory. The elments are pairs of icon '`id`'s and the filename of the image. The '`id`'s currently used internally by '`WebMagick`' are :- '`background`', '`dirbackground`', '`prev`', '`next`', '`next_grey`', '`up`', '`help`', and '`ball`'. These IDs are used as follows:

`background`
: thumbnail frame background image

`dirbackground`
: directory frame background image

`prev`
: link to previous page

`next`
: link to next page

`next_gray`
: link to next page (when there isn't one)

`up`
: link to parent directory up

`help`
: link to help Readme File (if it exists)

`ball`
: a decorative ball. If no definition exists (array element is left undefined) then the ball is not displayed in preference to unadorned text.

By default the `background` and `dirbackground` options are undefined which results in no background images being defined.

Other images can also be defined in the hash table and WebMagick will automatically process those images along with the images it uses, to generate the icon URL and size infomation. This information can then be used in the various special formating options to add extra, or special purpose links.

An alternative way to define entries (particularly useful in directory .webmagickrc files) is to define (or undefine) individual hash table entries.

Use syntax similar to

```
$opt_icons{'background'} = 'background.gif';
```

to add a background image to the thumbnail page or override an existing definition.

Use syntax similar to

```
undef $opt_icons{'background'};
```

to undefine an existing definition for the image corresponding to '`background`'. This can be useful if the default is to display a background image but you want to turn off the background image for a particular sub-directory tree.

4.2 Index File Base Name

`Variable:`

$opt_indexname=filename

`Argument:`

–indexname filename

Base name of the master index files (default server index), usually set to '`index.html`'. This is also the filename WebMagick will look for in sub-directories and parent directory '`.webmagickrc`' files as the document to make links to.

EG: If a directory is not processed by WebMagick, by either never running WebMagick on that directory, or the option `$opt_ignore` has been set, then setting `$opt_indexname` will specify the document WebMagick created index files of OTHER directories should link to.

If you do not want WebMagick to ever link to this directory then setting the option, `$opt_indexname`, to "NOLINK" will will prevent WebMagick index files in other directories from linking to this directory.

If you want WebMagick to entirely ignore a particular directory tree, even during recursion of the surrounding directories, then you can use the following options in its "'`.webmagickrc`'" file:

```
#
# .webmagickrc -- WebMagick is not to process or even link
# to this directory tree
#
$opt_indexname = 'NOLINK'; # Do not link to this directory
$opt_ignore    = 1;        # Do not generate WebMagick image indexes
$opt_prune     = 1;        # Don't recurse into sub-directories
```

4.3 Specifying README File Name

`Variable:`

$opt_readme=filename

`Argument:`

–readme filename

Name of directory information file (default '`README.html`'). If this file exists in the directory, then it will be displayed in preference to the first page of images and a link will appear in the directory frame which links to the image index. Typically the contents of this file describe the images to be viewed.

4.4 Page Index File Name

`Variable:`

$opt_pageindexname=string

`Argument:`

–pageindexname string

Base name of page-related index files (default '`.index`'). This name is used as the prefix for all generated files except for the main index file. The default naming creates these as hidden files (UNIX convention).

4.5 Specifying Sub-Directory Titles

`Variable:`

$opt_dirindexname=filename

`Argument:`

–dirindexname filename

Specify the file name to contain the directory-name to title cross-reference for naming sub-directories (default file name '`.dirindex`'). If this file exists, then *dirtext* will replace '`dirname`' if '`dirname`' is found in the file. This provides the ability to use user-specified text rather than actual sub-directory names. The order specified by this file is used to sort directory names. If an entry for a directory is not in this file, then that directory name is sorted alphbetically with relation to the other directory names.

The format of the file is:

```
dirname  dirtext
```

With *dirname* representing the directory name and *dirtext* representing the replacement link text.

It is wise to limit the length of *dirtext* to fit within the limitations of the directory frame.

5 Server-Side Imagemaps

5.1 Imagemap Program

`Variable:`

$opt_htimage=path

`Argument:`

–htimage path

Specify imagemap CGI program URL (set to '' for none). Older servers (e.g. CERN and old NCSA) require an external program to handle server-side imagemaps. These programs are usually supplied with a full URL to the referenced file. Newer servers have server-side imagemap support built-in and can accept URLs relative to the current directory. If you have a modern server, then specify this value as a null string (just hit ⟨RETURN⟩ when running the configure script).

5.2 Imagemap Type

`Variable:`

$opt_maptype=maptype

`Argument:`

–maptype maptype

Specify server-side map type (set to `"ncsa"` or `"cern"`). NCSA and Apache servers require `"ncsa"`. CERN requires `"cern"`. Phttpd can use either format.

NCSA format looks similar to:

```
default .index1.html
rect scooby1.gif 440,0 549,128
```

while CERN format looks similar to:

```
default .index1.html
rect (440,0) (549,128) scooby1.gif
```

6 Thumbnail Caching

6.1 Enable Caching

`Variable:`

$opt_cache

`Argument:`

–cache

Enable thumbnail caching. This is good for a 5X reduction in processing time when a single image is added or removed from a directory since thumbnails are cached in reduced form.

As an added benefit, WebMagick can share its cache with John Bradley's `xv` image manipulation program (Visual Schnauzer feature) if configuration values are set as follows:

```
$opt_cacheformat= 'P7';
$opt_cachegeom  = 80x60;
$opt_cachedir   = '.xvpics';
```

Sharing the cache with `xv` is only feasable if all of the images being cached are compatable with `xv` (ImageMagick supports many more formats than `xv`). A further drawback of sharing the cache with `xv` is that `xv` is limited to thumbnails with a geometry of 80x60 which may be small for your tastes (I prefer larger).

6.2 Cache Directory

`Variable:`

$opt_cachedir=dirspec

`Argument:`

–cachedir dirspec

Specify subdirectory name to cache thumbnails in (default '`.cache`').

6.3 Cache Thumbnail Format

`Variable:`

$opt_cacheformat=imageformat

`Argument:`

–cacheformat imageformat

Specify the format that cached thumbnails are saved in. Can be any format that ImageMagick knows how to read and write. The default is 'JPEG' which works well for photorealistic images.

6.4 Cache Thumbnail Geometry

`Variable:`

$opt_cachegeom=geometry

`Argument:`

–cachegeom geometry

Specifies the geometry specification for cached thumbnails. The default is to use the value specified by `$opt_thumbgeometry` (see Section 7.5.5 [Thumb Geometry], page 19). If a smaller value is used, there will be more unused space around the montaged thumbnails. Use of a larger value than the thubnail geometry is to be avoided because the cached thumbnail will still have to be reduced for the montage, costing both time and thumbnail quality.

6.5 Cache Minimum Image Size

`Variable:`

$opt_cachemin=minsize

`Argument:`

–cachemin minsize

Specifies the original Image size (in total pixels) over which caching will be used. For images smaller than this size, WebMagick will use the original image in the Montage and no thumbnail will be cached. The default value is 60,000 which is equivalent to a 300×200 image.

Selecting a value for this option is a judgement call based on space vs time. If a small value is selected, the initial run will be slower and more disk space will be consumed by thumbnails. However, subsequent updates should be faster since more thumbnails are already reduced. If a large value is selected, then less disk space will be consumed, the first pass will be faster, and subsequent passes will be slower.

7 Montage Options

7.1 The Anatomy Of A Montage

7.1.1 Montage Layout

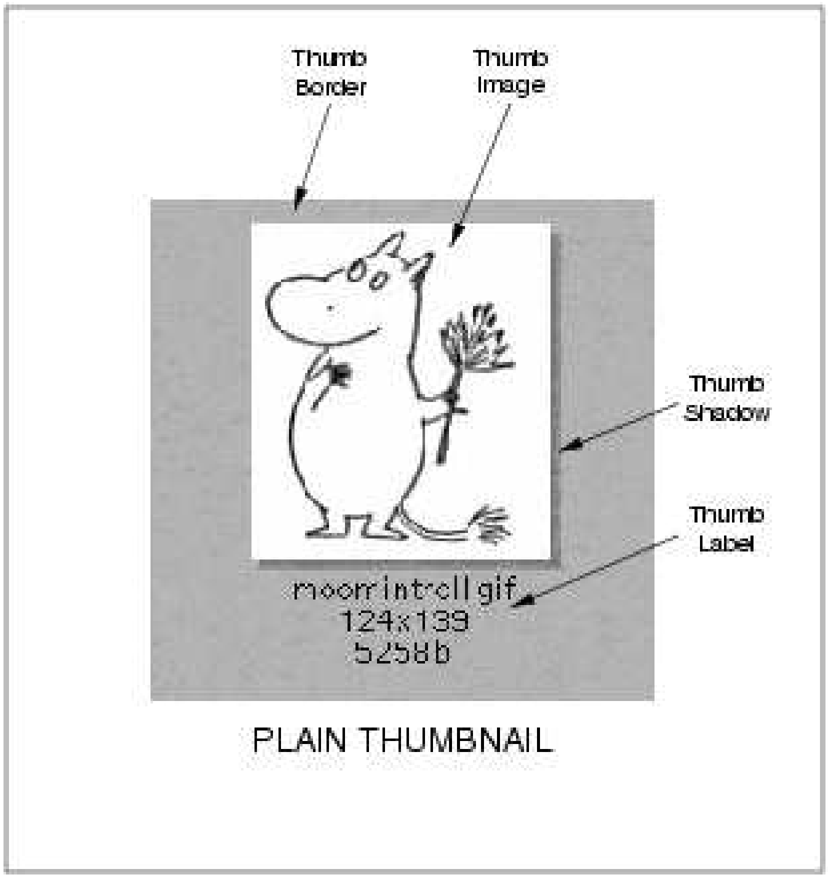
A montage is a single image which is composed of thumbnail images layed out in a grid. The size of the montage image is determined by the size of the individual thumbnails and the number of rows and columns in the grid.

The following figure shows a montage consisting of three columns and two rows of thumbnails:

7.1.2 The Anatomy Of A Thumbnail

7.1.2.1 Un-Framed Thumbnails

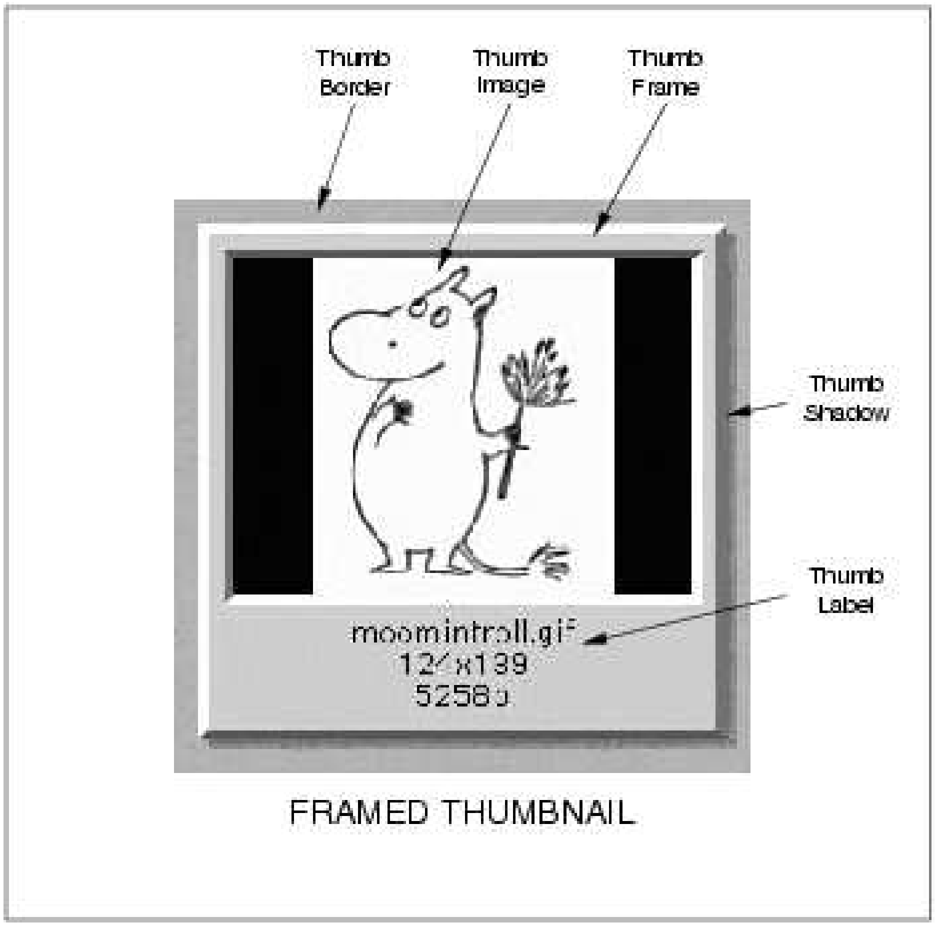
Un-framed thumbnails consist of four components: the thumbnail image, the thumbnail border, an optional thumbnail shadow, and an optional thumbnail label area.



PLAIN THUMBNAIL

7.1.2.2 Framed Thumbnails

Framed thumbnails consist of four components: the thumbnail image, the thumbnail frame, the thumbnail border, an optional thumbnail shadow, and an optional thumbnail label area.



7.2 Montage Geometry

The montage geometry specifies the size in rows and columns of the generated thumbnail grid. Since a single montage is displayed per HTML page, after the montage grid is full, a new page is started. The montage geometry combined with the thumbnail geometry determines the total size of the montage image and therefore the size that the user's web browser must be set to in order to view the entire montage.

Setting the rows to a very large value (large enough that rows x columns exceeds the number of images) results in a single HTML page being generated. The user can then use the browser's vertical scroll bar to peruse the thumbnails.

7.2.1 Columns

`Variable:`

$opt_columns=number

`Argument:`

–columns number

Specify the montage columns (max). This is the maximum number of thumbnail columns allowed per thumbnail row.

7.2.2 Rows

`Variable:`

$opt_rows=number

`Argument:`

–rows number

Specify the montage rows (max). This is the maximum number of thumbnail rows allowed per page.

7.3 Montage Image Format

7.3.1 GIF/JPEG Transition Point

`Variable:`

$opt_maxgif=number

`Argument:`

–maxgif number

Specify the maximum size of GIF imagemap before trying JPEG (default 30000). This is an optimization which tries to avoid the JPEG conversion step. When indexing true or deep-color images, JPEG usually leads to a smaller montage imagemap file.

Even if a JPEG image is generated, the smaller of the two images is selected for use and the larger one is deleted to save disk space.

7.3.2 Force GIF Format

`Variable:`

$opt_forcegif=1

`Argument:`

–forcegif

Force GIF imagemaps to be generated (default off). This avoids the possible generation of JPEG format imagemaps when the GIF file is large (over 30K). The primary reason for using this option is because GIF supports image transparancy while JPEG does not, leading to a possibly more attractive display when using GIF format imagmaps. The drawbacks of forcing GIF format are larger imagemap files (often two or three times the size of JPEG when montaging true or deep-color images) and the limited color depth (256 colors) of the GIF format.

7.4 Montage Colors

7.4.1 Thumbnail Foreground Color

`Variable:`
    $opt_thumbforeground=colorspec

`Argument:`
    –thumbforeground colorspec

Specify the montage foreground color. This color is used as the color for thumbnail label text.

7.4.2 Thumbnail Background Color

`Variable:`
    $opt_thumbbackground=colorspec

`Argument:`
    –thumbbackground colorspec

Specifies the background color that thumbnails are imaged upon.

7.4.3 Thumbnail Transparent Color

`Variable:`
    $opt_thumbtransparent=colorspec

`Argument:`
    –thumbtransparent colorspec

Specify a montage color to set transparent. Usually should be set the same as the background color `$opt_colorback` (see Section 7.4.2 [Thumb Background Color], page 17) of the imagemap page. GIF format imagemaps use the transparent color directly while JPEG format imagemaps may approximate "transparency" by setting the imagemap background color to the same color used as the background for HTML pages. This approximation becomes less useful for clients with shallow colormaps.

7.4.4 Thumbnail Frame Color

`Variable:`
    $opt_thumbframecolor=colorspec

`Argument:`
    –thumbframecolor colorspec

Specify the thumbnail frame color (only applicable if frames are enabled).

7.4.5 Thumbnail Frame Border Color

`Variable:`
: $opt_thumbframebgcolor=colorspec

`Argument:`
: –thumbframebgcolor colorspec

Specifies the background color within the thumbnail frame. Ignored if there is no frame.

7.4.6 Quantize To Netscape Color Cube

`Variable:`
: $opt_mapnetscape

`Argument:`
: –mapnetscape

Re-map the montage colors to the Netscape 216-color cube. Netscape maps colors to a fixed 216-color cube on PseudoColor color displays. Using this option allows you to ensure that Netscape doesn't dither or re-map your colors when used with common PseudoColor displays and the images may be smaller. The drawback to using this option is that the color quality will be significantly decreased on True and DirectColor displays or when using browsers that don't use the 216-color cube technique.

When using image transparency, make sure that the transparent color is selected from one of the 216 `"standard"` colors or transparency won't work.

7.5 Thumbnail Imaging

7.5.1 Enable Thumbnail Frame

`Variable:`
: $opt_thumbframe=geometry

`Argument:`
: –thumbframe geometry

Specify the geometry specification for frame to place around thumbnail. If no frame is desired then the string `"false"` should be specified. The specified geometry values are in *addition* to the thumbnail geometry. For example, a frame geometry of '8x8' places an 8 pixel frame around the image.

7.5.2 Enable Thumbnail Shadows

`Variable:`
: $opt_thumbshadow=boolean

`Argument:`
: –thumbshadow boolean

Set to string `"true"` or `"false"` to control the use of decorative shadows around image thumbnails (or frames if enabled). The default is `"false"`.

7.5.3 Thumbnail Frame Border Width

`Variable:`

$opt_thumbborderwidth=borderwidth

Specify the border (in pixels) to place between a thumbnail and its surrounding frame. This option only takes effect if thumbnail frames are enabled and the thumbnail geometry specification doesn't also specify the thumbnail border width.

7.5.4 Thumbnail Background Texture

`Variable:`

$opt_thumbtexture=texturespec

`Argument:`

–thumbtexture texturespec

Specify a texture to use as montage background. The built-in textures "granite:" and "plasma:" are available. A texture is the same aas a background image. For example:

Granite texture:

```
$opt_thumbtexture='granite:';
```

Plasma texture:

```
$opt_thumbtexture='plasma:blue-yellow';
```

Image file texture:

```
$opt_thumbtexture='vibrant.gif';
```

7.5.5 Thumbnail Geometry

`Variable:`

$opt_thumbgeometry=geometry

`Argument:`

–thumbgeometry geometry

Specify image thumbnail geometry. Thumbnail geometry is specifed in the form:

```
<width>x<height>+<borderwidth>+<borderheight>{!}{<}{>}
```

Specify the size and scaling options for thumbnail images (default '106x80+2+2>').

The following description is extracted verbatim from the montage manual page regarding the treatment of this option:

By default, the width and height are maximum values. That is, the image is expanded or contracted to fit the width and height value while maintaining the aspect ratio of the image. Append an exclamation point to the geometry to force the image size to exactly the size you specify. For example, if you specify 640x480! the image width is set to 640 pixels and height to 480. If only one factor is specified, both the width and height assume the value.

Use `>` to change the dimensions of the image only if its size exceeds the geometry specification. Use `<` to resize the image only if its dimensions is less than the geometry specification.

For example, if you specify 640x480> and the image size is 512x512, the image size does not change. However, if the image is 1024x1024, it is resized to 640x480.

Each image is surrounded by a border whose size in pixels is specified as <border width> and <border height> and whose color is the background color.

7.5.6 Thumbnail Gravity

`Variable:`
$opt_thumbgravity=direction

`Argument:`
–thumbgravity direction

Specify the thumbnail positioning within the specified geometry (`$opt_thumbgeometry` area. If the thumbnail is smaller in any dimension than the geometry, then it will be placed according to this specification. Available options are `NorthWest`, `North`, `NorthEast`, `West`, `Center`, `East` `SouthWest`, `South`, and `SouthEast`. Just as with a map (at least those in the northern hemisphere) `North` is up and `West` is left. The default is `Center` so that thumbnails are centered in their grid.

7.5.7 Thumbnail Composition Algorithm

`Variable:`
$opt_thumbcompose=operation

`Argument:`
–thumbcompose operation

Specify the image composition algorithm for thumbnails. This controls the algorithm by which the thumbnail image is placed on the background. Available options are (default `Replace`): `Over`, `In`, `Out`, `Atop`, `Xor`, `Plus`, `Minus`, `Add`, `Subtract`, `Difference`, `Bumpmap`, `Replace`, `MatteReplace`, `Mask`, `Blend`, and `Displace`. Use of `Over` is recommended for use with images that have transparency. This option may have negative side-effects for images without transparency.

7.5.8 Thumbnail Zoom Filter

`Variable:`
$opt_zoomfilter=filtertype

`Argument:`
–zoomfilter filtertype

Select the filter algorithm to use for image zooms (reductions & enlargements). The available values are `Box`, `Triangle`, and `Mitchell`. The algorithms have a significant effect on processing time and may have a perceptible effect on image quality. Images zoomed via the `Box` algorithm will have the lowest quality (unsuitable for further processing) and will zoom the fastest. Images zoomed via the `Triangle` algorithm usually have sufficient quality that it takes careful inspection to detect differences from the best algorithm.

The `Triangle` algorithm is almost as fast as the `Box` algorithm so there is little need to use the `Box` algorithm. The `Mitchell` algorithm produces the best possible results (preserving fine details) but with a 40% (or so) speed penalty overall as compared with the `Triangle` algorithm. If speed is not important for your application, and end-users are likely to be using deep-color displays, then by all means choose the `Mitchell` algorithm because if you choose a lesser agorithm, you will never know what you are missing.

7.6 Thumbnail Label Options

7.6.1 Thumbnail Labels

`Variable:`
$opt_imgindexname=filename

`Argument:`
–imgindexname filename

Specify the filename for the image name to thumbnail-title cross-reference for images (default '`.imgindex`'). If this file exists, then *labelformat* will replace *imagename* if *imagename* is found in the file. This provides the ability to use user-specified label text rather than the default provided by `$opt_thumblabel`. See the description of `$opt_thumblabel` or the montage(1) manual page for information on the label format. The order specified by this file is used to sort image names. If an entry for an image is not in this file, then that image name is sorted alphbetically with relation to the other image names.

The format of the image index file consists of an image name followed by an image label. The image name and the image label are delimited by white space which may be one or more space or tab characters as shown in the following example:

```
mr2-91ev.jpg '91 Front
mr2-91re.jpg '91 Back
mr2-91si.jpg '91 Side
```

7.6.2 Thumbnail Label Format

`Variable:`
$opt_thumblabel=format

`Argument:`
–thumblabel format

Specifies the default format of thumbnail labels. The image filename, type, width, height, or scene number in the label by embedding special format characters. Embed %f for filename, %d for directory, %e for filename extention, %t for top of filename, %m for magick, %w for width, %h for height, %s for scene number, %b for file size, or \n for newline. For example,

```
-label "%m:%f %wx%h"
```

produces an image label of MIFF:bird.miff 512x480 for an image titled bird.miff and whose width is 512 and height is 480. (This explanation borrowed from montage manual page). If a label is not desired, then use the argument "*label*" to turn off this feature.

The default format is '%f\n%wx%h %b' which displays the filename over the image geometry followed by image size.

See the `$opt_imgindexname` option for a description of a more powerful per-image labeling mechanism.

When `$opt_cache` is enabled, a number of labeling features will not work correctly since they may refer to characteristics of the cached thumbnail rather than the original image.

The labels which are supported when caching is enabled are:

`%b` (file size)

`%f` (full filename)

`%h` (height)

`%m` (magick)

`%n` (filename minus extension)

`%w` (width)

7.6.3 Thumbnail Label Width

`Variable:`
$opt_thumblabelwidth=labelwidth

`Argument:`
–thumblabelwidth labelwidth

Specify the maximum column width (in characters) of label text. Label text longer than this is truncated to the specified width. The purpose of this is to ensure that the label is not munged due to excessive length.

7.6.4 Thumbnail Label Font

`Variable:`
$opt_thumbfont=fontspec

`Argument:`
–thumbfont fontspec

Specify the thumbnail title font. This is the X11 font used to title thumbnails (default 5x8). PERL's newgetopt module seems to have difficulties with the dashes in most X11 font specifications. If 'getopt' prints the usage message rather than doing what you want to, then try using the command line syntax `--thumbfont=fontspec` and you should have better luck.

7.7 Thumbnail Processing Hooks

WebMagick provides EMACS-like hooks that can be used to insert additional processing code fragments into the thumbnail generation code.

7.7.1 Read Failure Hook

`Variable:`

$opt_readfailhook=string

PERL code fragment to evaluate if an original image fails to read during the process of building a thumbnail. Usually this is due to the original image being corrupted. Your mileage may vary.

When the code fragment is executed, the name of the current image is in the variable '$imagename'. For example:

```
$opt_thumbreadfailhook='unlink($imagename)';
```

removes the image, assuming that it is defective. In the case of images retrieved from the binaries newsgroups this is not a bad assumption.

7.7.2 Thumbnail Post-processing Hook

`Variable:`

$opt_thumbposthook=string

PERL code fragment to evaluate on the reduced thumbnail before saving it to cache or using it in the montage. The thumbnail image is available via the variable $image. For example:

```
$opt_thumbposthook='$status=$image->Sharpen(factor=>40);
  warn $status if "$status"';
```

applies the PerlMagick Sharpen operation on the thumbnail,

```
$opt_thumbposthook='$image->Set(colorspace=>"Gray");
  $status=$image->Quantize();
  warn $status if "$status"';
```

displays thumbnails in grayscale, and

```
$opt_thumbposthook='$image->Set(colorspace=>"Gray");
  $status=$image->Quantize();
  warn $status if "$status";
  $status=$image->Emboss();
  warn $status if "$status"';
```

displays embossed thumbnails.

See the PerlMagick documentation (`http://www.wizards.dupont.com/cristy/www/perl.html`) for the many other operations which may be applied to an image.

7.7.3 Thumbnail Pre-Processing Hook

`Variable:`

$opt_thumbprehook=string

PERL code fragment to evaluate on the original image in memory prior to reducing it into a thumbnail. The image is available via the variable $image. For example:

```
$opt_thumbprehook='$status=$image->Blur(factor=>80);
warn $status if "$status"';
```

applies the PerlMagick Blur operation on the image. See the PerlMagick documentation (http://www.wizards.dupont.com/cristy/www/perl.html) for the many operations which may be applied to an image.

8 HTML Options

8.1 HTML Content

8.1.1 JavaScript Mode

`Variable:`

$opt_javascript=1

`Argument:`

–javascript

When JavaScript mode is enabled, JavaScript code is added to the main index file (default index.html) to re-set the location to the JavaScript index file (currently hard-coded as "indexjs.html"). This results in JavaScript being loaded into the browser automatically if JavaScript is supported by the browser and the browser is one that WebMagick supports (currently Netscape 3.0X, Netscape 4.0X, and Internet Explorer 4.0X). Browsers that do not support JavaScript, have JavaScript disabled, or are not recognized by WebMagick safely ignore the JavaScript and operate directly from WebMagick's usual HTML files.

The advantage of JavaScript mode is that other than retrieving a few intial files and the images to be viewed, all HTML is generated directly within the user's browser, minimizing server accesses. In addition, navigation and presentation are dramatically improved since there is no need to pre-compute every possible viewing option and JavaScript has better control and knowledge of the browser than HTML does.

In order for JavaScript mode to operate correctly, the file 'webmagick.js' must be installed in the location specified by `$opt_iconpath` (see Section 3.1.3 [Icon Location], page 6) and the server must be updated to associate MIME type "application/x-javascript" with the extension ".js". Read your web server's documentation to see what is required in order to add new MIME types (if the type is not already supported).

Apache server: Update conf/mime.types and add the line

```
application/x-javascript        js
```

CERN or W3 server: Update config/httpd.conf and add the line

```
AddType .js application/x-javascript 8bit
```

Phttpd: Update modules/file.conf and add the following to the content-types list

```
        .js             application/x-javascript
```

8.1.2 README File Treatment

`Variable:`

$opt_readmevisible=1

`Argument:`

–readmevisible

Specify the handling of a README file. If the file designated by `$opt_readme` exists (default '`README.html`') then make it the first page seen when the user enters a directory. Regardless of this option, the help icon will appear if the README file exists which is a link to the file.

8.1.3 Address Information

`Variable:`
: $opt_address=string

`Argument:`
: –address string

Specify additional information to place in <ADDRESS></ADDRESS> tags in page frame. WebMagick copyright and support information is placed on all generated pages.

8.1.4 Anonymize

`Variable:`
: $opt_anonymous

`Argument:`
: –anonymous

Turn off the WebMagick copyright info and author's address on all generated pages. If you are operating a site that my mother wouldn't approve of, please use this option.

8.1.5 Page Header

`Variable:`
: $opt_header=string

`Argument:`
: –header string

Specify text to add to the page header in the thumbnail frame (a short term hack). This option is subject to change and will be eliminated if the thumbnail frame is templatized.

8.1.6 Page Title

`Variable:`
: $opt_title=string

`Argument:`
: –title string

Specify the page title. If this option is not specified, then WebMagick will generate its own title in the form 'Index of directory "subdirectory name"'.

8.1.7 Directory Page Template (No Frames)

`Variable:`
: $opt_dirfmt=string

Specify a template (written in PERL) that represents the format used to generate the frame-less directory navigation page. This format will be seen by browsers that do not support frames. This is a guru-level option that may require reading WebMagick code.

8.1.8 Directory Page Template (Framed)

`Variable:`

$opt_frameddirfmt=string

Specify a template (written in PERL) that represents the format used to generate the framed (left frame) directory navigation page. This format will be seen by browsers that support frames. This is a guru-level option that may require reading WebMagick code.

8.1.9 Per-Image HTML

When per-image HTML mode is enabled, an HTML file is generated corresponding to each image. The HTML file is named based on the image file name with an HTML extension.

8.1.9.1 Enable Per-image HTML

`Variable:`

$opt_pichtml=1

`Argument:`

–pichtml

Enable per-image HTML file generation.

8.1.9.2 Per-image HTML extension

`Variable:`

$opt_pichtmlext='.ext'.

`Argument:`

–pichtmlext extention

Set the file extension to use for per-image HTML files. The final name is the image file name appended with this file extension. The default extension is '.html'.

8.1.9.3 Default Frame Target

`Variable:`

$opt_pichtmltarget='targetname';

`Argument:`

–pichtmltarget target_name

Set the default URL target name for links in per-image HTML files. There is normally no need to set this target because HTML is normally displayed in the current frame and there are no links by default, however, sometimes it is desirable for links to display in a different (or new) window.

8.1.9.4 Extra Top HTML

`Variable:`
    $opt_pichtmltop='html_source';

`Argument:`
    –pichtmltop 'html_source'

Extra HTML to insert above the image.

8.1.9.5 Extra Bottom HTML

`Variable:`
    $opt_pichtmlbottom='html_source';

`Argument:`
    –pichtmlbottom 'html_source'

Extra HTML to insert below the image.

8.2 Text Colors

8.2.1 Thumbnail Frame Background Color

`Variable:`
    $opt_colorback=colorspec

`Argument:`
    –colorback colorspec

Specify the background color for thumbnail frame.

8.2.2 Thumbnail Frame Text Foreground Color

`Variable:`
    $opt_colorfore=colorspec

`Argument:`
    –colorfore colorspec

Specify the toreground color for text in thumbnail frame.

8.2.3 Directory Frame Text Color

`Variable:`
    $opt_dircolorfore=colorspec

`Argument:`
    –dircolorfore colorspec

Specify the foreground text color of the directory frame. Defaults to the value of `$opt_colorfore` (see Section 8.2.2 [Thumb Foreground], page 28) if not explicitly set.

8.2.4 Directory Frame Background Color

`Variable:`
$opt_dircolorback=colorspec

`Argument:`
–dircolorback colorspec

Specify the background color of the directory frame. Defaults to the value of `$opt_colorback` (see Section 8.2.1 [Thumb Background], page 28) if not explicitly set.

8.3 HTML Link Colors

8.3.1 Thumbnail Frame Active Link Color

`Variable:`
$opt_coloralink=colorspec

`Argument:`
–coloralink colorspec

Specify the link (active) color for the thumbnail frame.

8.3.2 Thumbnail Frame Un-visited Link Color

`Variable:`
$opt_colorlink=colorspec

`Argument:`
–colorlink colorspec

Specify link (unvisited) color for the thumbnail frame.

8.3.3 Thumbnail Frame Visited Link Color

`Variable:`
$opt_colorvlink=colorspec

`Argument:`
–colorvlink colorspec

Specify link (visited) color for the thumbnail frame.

8.3.4 Directory Frame Active Link Color

`Variable:`
$opt_dircoloralink=colorspec

`Argument:`
–dircoloralink colorspec

Specify link (active) color for the directory frame. Defaults to value of `$opt_coloralink` (see Section 8.3.1 [Thumb Active Link Color], page 29)if not explicitly set.

8.3.5 Directory Frame Un-Visited Color

`Variable:`
: $opt_dircolorlink=colorspec

`Argument:`
: –dircolorlink colorspec

Specify link (unvisited) color for the directory frame. Defaults to the value of `$opt_colorlink` (see Section 8.3.2 [Thumb Un-visited Color], page 29) if not explicitly set.

8.3.6 Directory Frame Visited Color

`Variable:`
: $opt_dircolorvlink=colorspec

`Argument:`
: –dircolorvlink colorspec

Specify link (visited) color for the directory frame. Defaults to value of `$opt_colorvlink` (see Section 8.3.3 [Thumb Visited Color], page 29) if not explicitly set.

8.4 Frame Options

8.4.1 Enable Frame Borders

`Variable:`
: $opt_frameborder=borderenable

`Argument:`
: –frameborder borderenable

Enable decorative frame borders with the value 'YES' or disable decorative frame borders by specifying the value 'NO'. These options are passed directly to the HTML's FRAMESET 'FRAMEBORDER' option in the generated pages.

8.4.2 Frame Border Thickness

`Variable:`
: $opt_framebordersize=bordersize

`Argument:`
: –framebordersize bordersize

Specifies the number of pixels allocated to the frame border.

8.4.3 Frame Margin Height

`Variable:`
$opt_framemarginheight=marginheight

`Argument:`
–framemarginheight marginheight

Specifies the number of pixels allocated to the frame margin in the vertical direction.

8.4.4 Frame Margin Width

`Variable:`
$opt_framemarginwidth=marginwidth

`Argument:`
–framemarginwidth marginwidth

Specifies the number of pixels allocated to the frame margin in the horizontal direction.

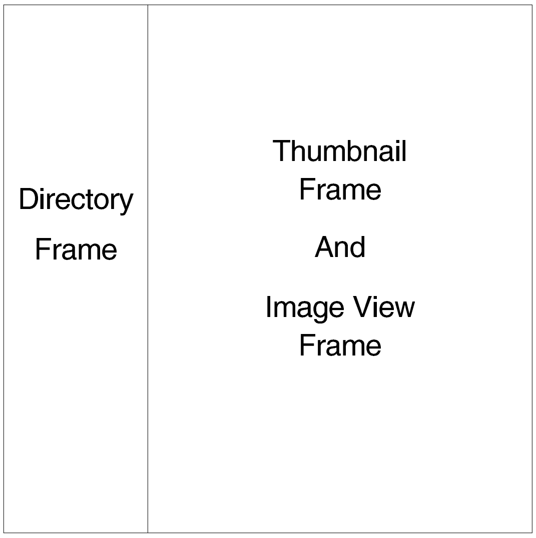
8.4.5 Frame Style

`Variable:`
$opt_framestyle=framestyle

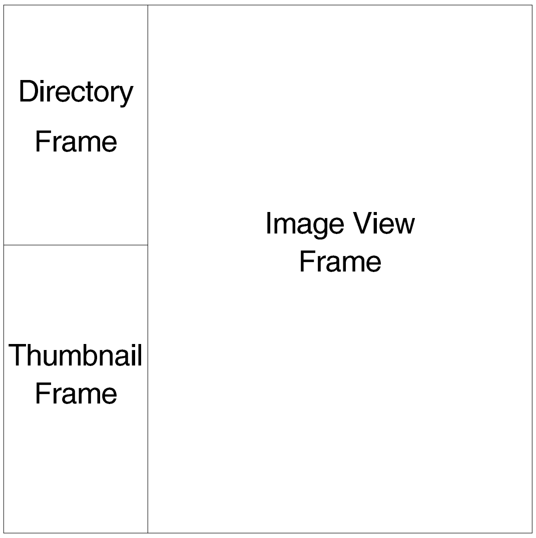
`Argument:`
–framestyle framestyle

Specifies the frame template to use. Templates are currently specified for the range of 1-4 with the following effect:

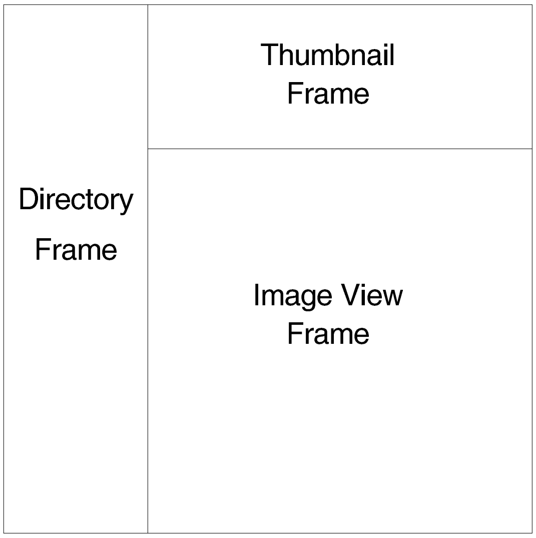
1. Simple two frame screen with directories listed in the left frame and imagemap or README displayed in the right frame.



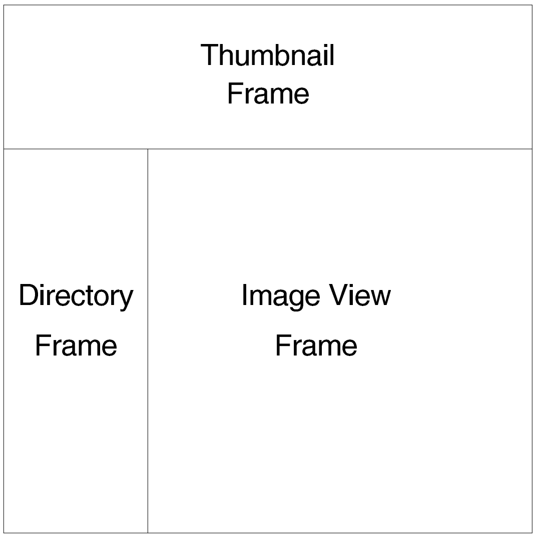
2. Three frame screen with directories listed in top-left frame, imagemap displayed in bottom-left frame, and README/Images displayed in full-height right-hand frame.



3. Three frame screen with directories listed in left frame, imagemap displayed in top-right frame, and README/Images displayed in lower-right frame.



4.
   Three frame screen with directories listed in lower-left frame, imagemap displayed in top frame, and README/Images displayed in lower-right frame.



8.5 HTML Meta Tags

8.5.1 Meta-Tag Author

`Variable:`
    $opt_metaauthor=string

Specify author name to add to the HTML meta tags in generated HTML files.

8.5.2 Meta-Tag Classification

`Variable:`
    $opt_metaclassification=string

Specify any classification information to add to the HTML meta tags in generated HTML files. This may be used by indexing robots.

8.5.3 Meta-Tag Description

`Variable:`
    $opt_metadescription=string

Specify page description text to add to the HTML meta tags in generated HTML files. This information may used by indexing robots.

8.5.4 Meta-Tag Expires

`Variable:`

$opt_metaexpires=string

Page expiration date to add to the HTML meta tags in generated HTML files. Should be specified in the form:

```
"Tue, 20 Aug 1996 14:25:27 GMT"
```

Since the specified expiration time is absolute, use of this option implies a commitment to execute WebMagick with option `--forcehtml` as least as often as the specified expiration period.

This information is used by browsers and caching servers to determine when cached data becomes stale.

8.5.5 Meta-Tag Key-Words

`Variable:`

$opt_metakeywords=string

Specify keywords to add to the HTML meta tags in generated HTML files. Should be specified as a string with keywords delimited by a comma (e.g. "key1,key2,key3"). This information may used by indexing robots.

9 Authors

WebMagick is written and maintained by Bob Friesenhahn (`bfriesen@simple.dallas.tx.us`) with copious input from Anthony Thyssen (`anthony@cit.gu.edu.au`). Anthony's diligent assistance as the first alpha tester is very much appreciated. Patches to support per-image HTML files and other usability enhancements were contributed by Andrey A. Chernov (`ache@nagual.pp.ru`).

ImageMagick and PerlMagick are written by John Cristy (`cristy@dupont.com`). WebMagick would not be possible without his wonderful software. The author greatly appreciates Cristy's assistance with ironing out PerlMagick's (or WebMagick's) bugs and feature set during the development of WebMagick.

10 Obtaining WebMagick

The following packages are required in order to install WebMagick. Retrieve each package and install according to the following order:

1. PERL version 5 (`http://www.perl.com/perl/index.html`)
2. ImageMagick (`http://www.wizards.dupont.com/cristy/ImageMagick.html`)
3. PerlMagick (`http://www.wizards.dupont.com/cristy/www/perl.html`)
4. WebMagick (`http://www.cyberramp.net/~bfriesen/webmagick/index.html`)

Each package provides its own installation instructions. Please follow them carefully.

11 Installing WebMagick

11.1 Basic Installation

Execute the provided configure script specifying a –prefix option for a GNU-style directory heriarchy you would like to install in (e.g. "./configure –prefix=/opt/tools"). This will generate a base copy of WebMagick that has been edited to reflect the location of your PERL interpreter and the X11 RGB database.

The configure script will ask you questions to provide definitions of `$opt_prefixpath`, `$opt_rootpath`, `$opt_iconpath`, `$opt_htimage`, and `$opt_maptype` in the '`webmagick`' script and the sample '`webmagickrc`' file. Read the Introduction section of this manual to learn the details. A few examples are provided below. The script provides a running narrative with examples so it should not be necessary to read much documentation before running it.

Then "make install". This will install WebMagick, sample icons, the WebMagick JavaScript interface, and TexInfo documentation. The WebMagick icons must be installed in a directory somewhere under the same server "root" as the images you want to catalogue. This root does not neccessarily have to be the actual server root. The "root" may be established via a server path mapping (rather than symbolic link) that offsets onto another filesystem.

In order for JavaScript mode to operate correctly, the file '`webmagick.js`' must be installed in the location specified by `$opt_iconpath` and the server must be updated to associate MIME type "application/x-javascript" with the extension ".js". This file contains the various JavaScript routines that define WebMagick's JavaScript interface. See the formal documentation for more on JavaScript.

11.2 Installation Names

By default, '`make install`' will install the package's files in '`/usr/local/bin`', '`/usr/local/man`', etc. You can specify an installation prefix other than '`/usr/local`' by giving `configure` the option '`--prefix=`*path*'.

In addition, if you use an unusual directory layout you can give options like '`--bindir=`*path*' to specify different values for particular kinds of files. Run '`configure --help`' for a list of the directories you can set and what kinds of files go in them.

Option Index

Summary index of available WebMagick option variables. Many of these options have command-line equivalents.

$

%

Argument Index

Summary index of available WebMagick command-line arguments. All of these arguments have equivalent option variables.

Concept Index

T

U

V

W

Z

Table of Contents