IMAGE SERVICES
by
CLEAR IMAGE MEDIA
http://www.clearimageonline.com

INTRODUCTION

Image Services Add-on by Clear Image Media for Aestiva's HTML/OS and H2O

Congratulations on using Image Services from Clear Image Media. Image services are an indispensable tool for anyone using HTML/OS to design applications for the web.

Image Services removes the drudgery from managing images. Never again will you need to upload variations of the same image again. Web pages will display more quickly by the automatic addition of the width= and height= tags to the img tag. With image services broken img links are no longer a problem. It returns nothing if the image has not been added to the server. When logged in an upload icon allows you to add/change/delete your image. Images can be linked using a name/value pair. Managing multiple versions of the same image is transparent. Upload one image, not multiple images. Thumbnails and other variants are handled automatically. Image Services is optimized for speed. All image operations are cached for blazing performance. Image Services includes an amazing browser cache eliminator. Never have to refresh a web page again to force the image you just uploaded to show up. Image Services includes several free tools to help you debug your code. ci_stats() can be added near the bottom of your code to get timing and speed optimization help. You can also reference ci_tagresults for other helpful information.

Aestiva has brought new meaning to the web with its dynamic database driven design. Image Services extends that same idea to images.

Enough bragging. I am sure you will see that image services is one of the most valuable tools you will use next to Aestiva. Good luck building "advanced web applications".

INSTALLATION

Image Services can be run on a server that already has ImageMagick setup and running. It will also run just fine without ImageMagick, however many of the advanced features will not be present until ImageMagick is setup correctly.

ImageMagick Setup

First setup your system folder to allow external applications. This can be set by going to your Aestiva desktop >> Control Panel >> System >> System Folder. This should point to the folder on your server that has the external applications. Refer to the HTML/OS tag system() for more information. If you are using H2O you can set your system folder by adding or modifying a line in you htmlos.conf file. This is located in your CGI-BIN folder. Open it and change the line that starts with "SystemDirectory" to state "SystemDirectory /server/path/to/im". Of course you would put the path on your server to the Image Magick executables.

Image Services requires three programs installed in your system folder. They can be found by installing the ImageMagick programs at http://www.imagemagick.org. The three used by image services are identify, convert, and composite. If you are not sure how to add these to your system folder consult the knowledge base on Aestiva's web site. It is beyond the scope of Image Services to get ImageMagick running.

Image Services Installation

Next, use Aestiva's control panel to install the imageservices.bb file. Refer to the knowledge base on Aestiva's web site for instructions on installing applications. Upon installation, image services will install four files into a folder called clearimage at the root of your Aestiva system.

1. getimage.lib – All Image Services Functions are here
2. controlpanel.html – This is where you will set up Image Services. You can use it to determine how Image Services will run. It also included several coding examples. This is open source. Feel free to modify.
3. upload.html – This is an open source application for uploading your files. It is not required, but an upload application is required for Image Services to run.
4. edittext.html – This is a simple open source application for editing text files. test.html uses it. When image services is used for the first time it will create other files in the clearimage directory.
5. browse.html – This is a sophisticated file manager to replace the File Manager that is built into Aestiva. This is also open source. Feel free to modify the code and enhance it.

When image services is used for the first time it will create other files in the clearimage directory.

Image Services Configuration
Lastly, running controlpanel.html will configure the rest of image services. You may need the path to your servers document root and the names of the ImageMagick executables on your server. If things are improperly configured you will be asked to supply the correct information. Most of the time this information is automatically detected. If you are not asked for the information then Image Services was able to correctly identify things. If you are moving from one server to another you should remove the /clearimage/prefs.csv file. It will get recreated with your new server settings.

To use Image Service in your own applications simply place the following near the front of your html documents.

```
expand file="/clearimage/getimage.lib" /expand
```

With this in your page you can use any of the image services tags described in the following pages. Optionally you can add <<ci_stats()>> near the bottom of your code to get useful speed optimization tips.

Thanks for using Image Services. For the latest version go to
http://www.clearimageonline.com/imageservices.bb

Version History

```
1.0  Image services is introduced
1.5  Image Services is completely rewritten from the ground up
     It includes Web Browser Cache Killer, External URL's
     highly optimized speed improvements. Better cache
     management
     All image variants are now stored in a .TEMP directory
     found in the images directory
2.0  Added table management
     Added Complete Integrated Development Environment
     Added a reworked Text editor
     Added Traditional Image Map creator
2.01 Added CSS style image maps with rollovers
2.02 Fixed ci_geturl() to call with nosession
     Added a changelog

2.10 Completely Reworked Upload Icon. No Longer uses a graphic
     for upload, just uses CSS to place the icon
2.11 Reverted back to old upload system for stability reasons
2.20 Added a Timeout. If the page is out for more than 15000
     milliseconds then image will return "TIMEOUT"
     The end user will have to refresh the page to get
     additional images
2.21 Added to new tags for Beta testing
     ci_link()
     ci_download()
2.22 Reworked Imgmap2 completely. Much easier interface
2.3  Added ci_imgcrop() this will crop the images to fit
     instead of resize to the bounding box.
2.4  Added logging capabilities. All system calls are logged.
     Errors are reported through ci_tagresults
     Logs are XML. Logs can be turned on/off through the
     control panel.
     Events are logged to /clearimage/imagemagick.log
2.41 Fixed a few bugs that showed up on some older copies
     of Aestiva
3.0  Added a new code system upgrades are $48.00
3.01 Fixed ci_imgmap() bug that would lose map coordinates
     when editing
3.1  Added MultiUpload Capablities use
     ci_popup('file1.mov|file2.pdf','Upload Files')
3.11 Added MultiUpload capabilities to use clientname
     ci_popup('{{USECLIENTNAME}}|{{USECLIENTNAME}}','Upload 2')
3.12 Fixed bug that wouldn't allow mixed case file extensions
3.13 Removed email Clear Image Code from controlpanel.html The
     new logging facilities should report all the info needed.

3.2  Added ci_parsehref() to make creating popups easy syntax:
     href="go.html" popup="600,400" name="var" value="John Doe"
3.21 Cleaned up registration code. Automated registration with
     clearimageonline.com
3.22 Fixed bug where IM wasn't tiling the ci_imgmap2() images
     correctly
3.23 Added registration email address to registration process
```



```
3.3  Fixed long standing bug in IE browsers where clicking an
     item would pop the window to the top (file manager)
3.31 Added logging of all system calls, not just ci_img related
     calls. /clearimage/syscall.log
3.32 Removed all onclick events (left them commented out in the
     code as onclic k)
3.40 Added ci_makeimg() for compositing multiple images
     together (image,x,y,list) image is the image to make,
     x and y are the bounding coordinates, and list is the
     table of images to paste together
3.41 Reworked upload.html moved code from getimage.lib to
     upload.lib, added comments in upload.html
```


CI_IMG
CI_IMG2
CI_IMGNR
CI_IMGNR2
CI_IMGCROP
CI_LINK
CI_IMGLINK
CI_LINK2
CI_IMGLINK2

Returns an <img> tag formatted for display. It will also create the temporary image if it has not already been created.

Usage:

CI_IMG(path,x,y,login)
CI_IMG2(path,x,y,login,quality,command,cmdoptions)
CI_IMGNR(path,x,y,login)
CI_IMGNR2(path,x,y,login,quality,command,cmdoptions)
CI_IMGCROP(path,x,y,login,float,force)
CI_LINK(path,x,y,login) – Just like ci_img with user editable link
CI_IMGLINK(path,x,y,link,login) – Just like ci_img with coded link
CI_LINK(path,x,y,imgopts,login)
CI_IMGLINK(path,x,y,link,imgopts,login)

Params:

path- the path to the image
X – constrain width to this
Y – constrain height to this
Login – "TRUE" or "FALSE"
Quality – Override global quality setting leave empty for default
Command – "convert" or "composite"
Cmdoptions – [OVERWRITE]options for ImageMagick
Float – TOP, MIDDLE, BOTTOM, or LEFT, CENTER, RIGHT
Force – "TRUE" or "FALSE"

Example:

CI_IMG("logo.jpg",20,20) => "src=logoTEMP20x20.jpg width=20 height=12"

CI_IMGNR("logo.jpg",20,20) => "src=logo.jpg width=12 height=17"

If logo.jpg exists on the server then the code will be returned to display it constrained to a 20x20 box. If the original image is smaller than 20x20 pixels it will be scaled up if it is larger than 20x20 it will be scaled down.

CI_IMAGENR (no resize) would scale down images but leave smaller images untouched. If logo.jpg isn't found then nothing will be returned. Setting login to true would cause an upload icon to be returned.

CI_IMG2("photo.jpg",200,200,"FALSE","","composite","-dissolve 80 –gravity center "+ci_getpath("watermark.png"))

The above example will add the watermark.png file at 80% opacity to the center of photo.jpg and return the HTML markup to display it. The final image dimensions will be bound by a 200x200 box.

CI_IMGLINK2('/google.gif',100,100,^href="test.html" name="myvar" value="TRUE" onclick="alert('Login!');"^,'border="10"','TRUE')

The above example shows how you can add additional attributes to the img tag and the href tag.

CI_TAGRESULTS:
[1] – WIDTH
[2] – HEIGHT
[3] – IMAGE TYPE
[4] – FILE or DIR
[5] – PUBLIC, PRIVATE, or MIRROR
[6] – FILE SIZE
[7] – MODIFICATION DATE
[8] – TRUE or FALSE (if FALSE see [1])
[9] – 0-Not Defined 1-Ok 2-ImageNotIdentified 3-FileIsDirectory 4-FileNotFound 5-DirectoryNotFound 6-SystemCommandDisabled
[10] – path to resulting file
[11] – Filled in from other functions

CI_IMAGE
CI_IMAGE2
CI_IMAGENR
CI_IMAGENR2

NOTE: Most of these tags will be deprecated. Use ci_link() and ci_imglink() instead

Returns the HTML code for an <img> tag. It will also create the temporary image if has not already been created. These tags allow greater flexibility in the image tags returned. The most notable is it allows the <img> tags to link to other pages. Unless you are creating linkable images ci_img(…) tags would be preferable to the ci_image(…) tags.

Usage:
CI_IMAGE(path,x,y,link,name,value,javascript,login)
CI_IMAGE2(path,x,y,link,name,value,javascript,login,qt,cmd,opts)
CI_IMAGENR(path,x,y,link,name,value,javascript,login)
CI_IMAGENR2(path,x,y,link,name,value,javscript,login,qt,cmd,opts)

Params:
path- the path to the image
X – constrain width to this
Y – constrain height to this
Link – html document to link image to
Name – Variable to set for link
Value – Value for variable
Javascript – Javascript or extra html for the link
Login – TRUE or FALSE to allow uploads or not
Qt – Override default quality setting leave empty for default
Cmd – Either "CONVERT" or "COMPOSITE"
Opts – ImageMagick Commands for CONVERT or COMPOSITE

*note if opts starts with OVERWRITE then graphic will be created every time

Examples:

CI_IMAGE("logo.jpg",200,200,"aboutus.html","","","","FALSE")
CI_IMAGENR("logo.jpg",200,200,"aboutus.html","","","","FALSE")
CI_IMAGE("logo.jpg","","","","aboutus.html","","","","TRUE")
This example would not do any resizing of the image, and would also allow uploads.

CI_IMAGE("logo.jpg",32,32,"aboutus.html","var","test","","FALSE")
This example would return a 32x32 icon of the image and link it to aboutus.html and also set var="test" in the link.

CI_TAGRESULTS:
[1] – WIDTH
[2] – HEIGHT
[3] – IMAGE TYPE
[4] – FILE or DIR
[5] – PUBLIC, PRIVATE, or MIRROR
[6] – FILE SIZE
[7] – MODIFICATION DATE
[8] – TRUE or FALSE (if FALSE see [1])
[9] – 0-Not Defined 1-Ok 2-ImageNotIdentified 3-FileIsDirectory 4-FileNotFound 5-DirectoryNotFound 6-SystemCommandDisabled
[10] – path
[11] – Filled in from other functions

CI_CONFIG

Returns the HTML code for image services status and configuration.
Usage: CI_CONFIG()

Example: CI_CONFIG()

Returns the html snippet to change/modify Image Services settings.

CI_REGISTERED

Returns TRUE or FALSE.

Example: if ci_registered="TRUE" then a="great" else a="hmmm" /if

CI_RMIMAGE
CI_RMTEMP

House keeping routines for Image Services.
Usage: CI_RMIMAGE(path)
CI_RMTEMP(path)

Example: CI_RMIMAGE("logo.jpg")

Removes logo.jpg and all associated temp files. Use this instead of sysrm to clean up all associated TEMP files.

CI_RMTEMP("logo.jpg")

Removes all associated temp files. Leaves logo.jpg alone.

CI_IDENTIFY

Used to identify image properties.

Usage: CI_IDENTIFY(path)

Params: path-the path to the image

Example: x=CI_IDENTIFY("logo.jpg")

Returns a table as follows

CI_TAGRESULTS:
[1] – WIDTH
[2] – HEIGHT
[3] – IMAGE TYPE
[4] – FILE or DIR
[5] – PUBLIC, PRIVATE, or MIRROR
[6] – FILE SIZE
[7] – MODIFICATION DATE
[8] – TRUE or FALSE (if FALSE see [1])
[9] – 0-Not Defined 1-Ok 2-ImageNotIdentified 3-FileIsDirectory 4-FileNotFound 5-DirectoryNotFound 6-SystemCommandDisabled
[10] – path
[11] – Filled in from other functions
[12] – EXIF Data
[13] – Caption
[14] – Comments

CI_DONGLE
CI_VERSION

Usage: CI_DONGLE()
CI_VERSION()

CI_DONGLE() returns either “TRUE” or “FALSE ERROR MESSAGE” It runs several tests to make sure Image Magick is running.

CI_VERSION() Returns the version number

CI_RESIZE
CI_RESIZEQT

CI_CONVERT
CI_CONVERTQT

Usage: CI_RESIZE(path,x,y)
CI_RESIZEQT(path,x,y,quality)
CI_CONVERT(path,type)
CI_CONVERTQT(path,type,quality)

Params: path – path to the image
X – Constrain Width
Y – Constrain Height
Type – Image type
Quality – Override default quality setting

These functions are here for completion. They should only be needed for special cases, All of the caching, and web browser cache killer features will not be realized with these tags.

Used to resize and convert images.

CI_GETURL
CI_GETPATH

Usage: ci_geturl(path,x,y,login)
ci_getpath(path)

Params: path – the path to the image
X – Max x value
Y – Max y value
Login – Either TRUE or FALSE

CI_GETURL(path,x,y,login) returns a URL to the file on the server. This is necessary for sending image links in emails. It will always guarantee the right image will display in the email, even if the original image is changed. This also useful for e-Bay and other services that might need to reference Image Services Images.

CI_GETPATH(path) returns the system path to an image. If the file is private it returns the system path to Aestiva's private files. If the path is public or unknown then it will return the path to Aestiva's public files.

CI_PREFS

CI_PREFS are loaded from the /clearimage/prefs.csv file. They can be temporarily changed. Exercise caution when modifying these settings. Below is a list of their purpose. If you are ever moving files from one server to another make sure to delete /clearimage/prefs.csv on the new server. Image Service will then be forced to recreate it with all the settings from your new server.

ci_prefs[1,1]="…" – System path to Aestiva's public area
ci_prefs[1,2]="80" - Global image quality
ci_prefs[1,3]="70" – Upload icon opacity
ci_prefs[1,4]="TEMP" - Name used in resized/cached images
ci_prefs[1,5]="LOGIN" - Name used to identify LOGIN images
ci_prefs[1,6]="…" - Name of the identify application
ci_prefs[1,7]="…" - Name of the convert application
ci_prefs[1,8]="…" - Name of the composite application
ci_prefs[1,9]=".." – Path to Upload Application
(feel free to write your own upload application)
ci_prefs[1,10]="" – Currently not used
ci_prefs[1,11]=".TEMP/" – Name of temporary/cached files directory
ci_prefs[1,12]="TRUE" – Web Browser Cache Killer (only turn off if you are asked to).
ci_prefs[1,13]="" – Start Link
ci_prefs[1,14]="…" – System path to Aestiva's private area
ci_prefs[1,15]="…" – Registration Key
ci_prefs[1,16]="15000" – Timeout Delay in milliseconds when Image Services suspends processing images.
ci_prefs[1,17]="0" Convert non-web images to a web format 0 or 1 for yes or no
ci_prefs[1,18]=Preferred web format for non-web files
ci_prefs[1,19]="0" Resize All originals
ci_prefs[1,20]=Resize Width
ci_prefs[1,21]=Resize Height

CI_IMGMAP
CI_IMGMAP2

Usage: ci_imgmap(path,login)
ci_imgmap2(path,login)

Params: path – the path to the image
Login – Either TRUE or FALSE

CI_IMGMAP2("header.jpg","TRUE") These functions allow you to create Image Maps. Just use this one tag and the rest is handled online. The only difference between the 2 is ci_imgmap2(…) allows you to also use CSS style image maps that can include state data also like hover and active link.

CI_TEXT
CI_EDIT
CI_CSV
CI_POPUP, CI_UPLOAD

Usage: ci_text(path,x,y,linktext,search,login)
ci_edit(path,x,y,linktext,search,login)
ci_csv(path,returnpage,"",linktext,"",login)
ci_popup(path,linktext)
ci_upload(path,linktext)

Params: path – the path to the text, csv, or other file
X – Width of the popup window
Y – Height of the popup window
linktext – The text of the link
search – The text to be searched and highlighted, ignored by ci_csv
returnpage – The page to return to when finished editing.
Login – Either TRUE or FALSE

CI_TEXT("description.txt",600,400,"Edit","","TRUE") returns the contents of "description.txt" with a link called "Edit" on the end of it. If login is "FALSE" then the contents alone are returned. When Edit is clicked a text editor pops up that is 600px wide and 400px tall. If the file specified by path has a .txt extension then the popup editor will wrap text.

CI_EDIT("description.txt",600,400,"EDIT","TRUE") Does the same as ci_text(…) except it only returns the link, not the contents of the file.

CI_CSV("products.csv","order.html","","Edit Products","","TRUE") returns a link to the CSV table editor built into Image Services. The second argument is the HTML/OS page to return to when finished editing. The third and fifth argument are not currently used

CI_POPUP("brochure.pdf","Upload the PDF") This creates a popup link to the upload page for any file type. CI_POPUP and CI_UPLOAD are the same.

NOTES:

