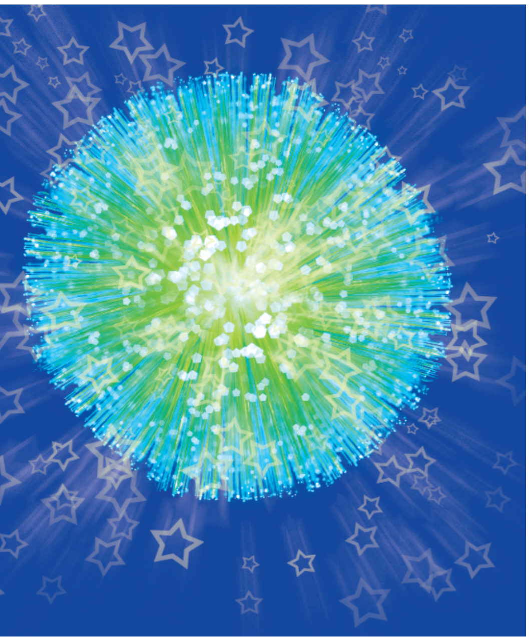
Editing images with ImageMagick

It's a Kind of Magic!



There's no need to start a complex program like The Gimp every time you want to resize, crop or convert an image. ImageMagick gives you the ability to handle a wide range of image manipulation jobs on the command line and helps you to automate picture processing by using clever shell scripts.

BY HEIKE JURZIK

The ImageMagick package contains various image manipulation tools that allow users to animate a sequence of images, create a composite image by overlaying a group of pictures, convert one or several images to different file formats or sizes, display and identify pictures, capture screenshots, and lots more.

These tools can save you a lot of time and typing or clicking – just put your favorite command calls into a shell script and automate the work. This article describes the more useful ImageMagick utilities, introducing you to the magic of creating, converting, and editing images on the command line.

Import your Screen

To capture a single window, a rectangular area or the whole desktop, you don't have to fire up tools like The Gimp or KDE's KSnapshot program. You can instead type the command, and specify your target by ID or name, or select it by clicking (and dragging) the mouse.

To capture a single window, type *import window.png*. ImageMagick automatically determines the file format to use from the extension you choose for the filename. Click the mouse in the window you want to capture – you should hear one beep when *import* begins to capture and another beep when the job is done. If you want to include the X window frame, you can add the option *-frame*. Alternatively, you can just press the left mouse button and then drag – *import* will now capture the selected area of the screen.

If you want to make a screenshot of the entire desktop and save it in the JPEG image format, you can use the option *-window*:

```
import -window root desktop.jpg
```

Need more time, or maybe you want to click on a pull-down menu to show certain aspects of a program? No problem – by combining *import* with the *sleep* command this is easily done:

```
sleep 10; import -window root desktop.jpg
```

You now have 10 seconds to arrange everything on the desktop before *import* will start its job and capture the desktop.

Informer

To quickly determine the dimensions of an image (and even check whether it is corrupt or incomplete), you can use *identify*. It gives you information about the width and height, the number of colors and bytes and the number of seconds it took to read and process the information. Together with the option *-verbose* you can learn more about the file and read additional image information:

```
Image: chickens.png
  Format: PNG (Portable Network Graphics)
  Geometry: 529x499
  Class: DirectClass
  Type: true color
  Depth: 8 bits-per-pixel component
```

It's even possible to print less information – *identify* also has an option called *-format*, which lets you specify the information you want to present. So, if you're only interested in the picture dimensions, you can use *identify* like this:

```
identify -format "%wx%h" chickens.png
529x499
```

%w and *%h* are only two of several special format characters. Additionally, you can ask for information on the file size (*%b*), the image comment (*%c*), the filename (*%f*), and many more. For a complete list, have a look at the ImageMagick manpage and search for *-format*. Note that you have to quote those format characters to prevent the shell from interpreting the %.

Convert your Pictures

It is not hard to guess what *convert* does: it allows you to convert between various graphics formats on the command line. But it can also create PostScript files, add captions to images, apply filters, and do much more.

To convert an image from JPEG to PNG format, type the following simple command: *convert image.jpg image.png*. The program evaluates the file extension to recognize the format, and thus knows the target format for the conversion. This command will not remove the original JPEG file but will simply give you an extra PNG file. The manpage has a list of the numerous image formats that *convert* supports. These include various PostScript formats – that can be really useful if you need to print out a picture. The *-page* parameter makes sure that the image is correctly positioned on the printed page. The following command:

```
convert -page A4 chickens.jpg ➥
document.ps
```

scales a file called *chickens.jpg* to fill an A4 PostScript document (see Figure 1).

If you need a quick summary of a large collection of image files, you can run *convert* to create a so-called “visual directory.” The following command:

```
convert 'vid:*.jpg' ➥
directory.jpg
```

first reduces all the JPEG files in the current directory, then adds them to a new image file called *directory.jpg* before adding captions that tell you the original filenames, the image size in pixels, and the file size.

Around the Clock

If you take many photos with a digital camera, you might want to rotate some picture portraits 90 degrees, if the camera model doesn’t do it automatically for you. Use *convert* together with the option *-rotate* and an additional argument, which specifies the number of degrees to the right to rotate the image:

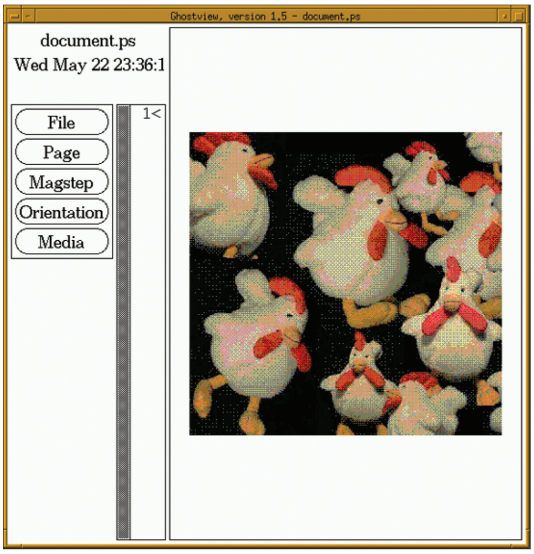


Figure 1: *convert* generates PostScript files from images.

```
convert -rotate 90 w➥
indow.jpg window_r.j➥
pg
```

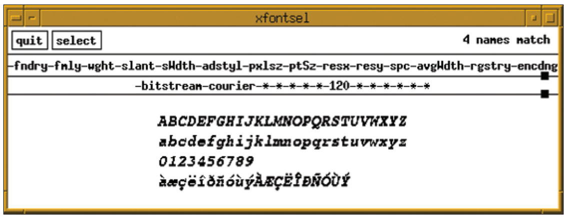
Alternatively, if you want to rotate to the left, use a negative number.

More Magic Tricks

convert helps you add captions to your images using *-font*, *-pointsize*, *-fill*, *-draw* options. You can choose between X server, and TrueType fonts. You need to add an @ prefix for the latter, as in *-font @font.ttf*. The *-pointsize* option allows you to specify the font size, as in *-pointsize 16* for a 16 point font.

To find installed TrueType fonts on your machine you could, for example, run the command *locate *.ttf*. If you opt for an X server font instead, you might like to run the *xfontsel* tool to tell you what fonts are available with what attributes, character sets, and sizes. Launch the program in an Xterm, and select the attributes you need. The number of fonts that match your criteria should decrease at every step; this is indicated by a display in the top right corner of the window (see Figure 2): “4 names match”). If you are happy with your selection, click *select*. You can now press the center mouse button to insert the string that you have created (for example *-font "-*-helvetica-medium-r-*-*-12-*-*-*-*-*-iso8859-*"*).

The *-fill* parameter specifies the font color. You have a choice of any colors supported by your X server (you can type *showrgb* for a full list of colors.) The *-draw* parameter specifies what to draw (circle, rectangle, text, and so on.) The best way to explain this is to look at an example with a combination of all these options:



4 names match

-fndry-fmly-wght-slant-sWdth-adstyl-pxlsz-ptSz-resx-resy-spc-avgWdth-rgstry-encdng

-bitstream-courier-*-*-*-*-*-120-*-*-*-*-*-*

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
0123456789
àæçëîðñóùýÀÆÇËÎÐÑÓÙÝ

Figure 2: *xfontsel* in action – helping you find the right font.

```
convert -font @/var/lib/➥
defoma/x-ttcidfont-conf.➥
d/dirs/TrueType/Verdana_➥
Bold.ttf -pointsize 50 -➥
fill white -draw "text 5➥
0,50 'Lots of chickens'" chicke➥
ns.jpg chickens_font.jpg
```

This writes “Lots of chickens” in a 50 point TrueType font (*Verdana_Bold.ttf*) in white on the image. The origin for the text position is the top left corner (0,0); so the text will be 50 pixels to the right and 50 down (*text 50,50*). The image is stored as *chickens_font.jpg*, and is shown in Figure 3.

To add a colored frame to the image, run *convert* with the *-mattecolor* and *-frame* options. The following command adds a blue frame that is six pixels wide vertically and horizontally:

```
convert -mattecolor blue -fram➥
e 6x6 chickens.jpg chickens_fr➥
amed.jpg
```

The Right Size

The *-geometry* option allows you to scale an image up or down and keeps the original proportions. An image that starts off at 320x200 is scaled to 640x400 if you type *-geometry 640x480*, for example. If you want to fill an area (and don’t mind distorting the image), you need instead to type *-geometry '640x480!'* (note the exclamation mark). The single quotes are not strictly necessary in this case, but since the shell uses ! for history expansion, it’s wise to escape it. Alternatively, you could put a backslash in front (*-geometry 640x480\!*). You can also specify *-geometry* with a percentage, as in *-geometry 200%*. As *convert* is part of the ImageMagick package, it can use multiple ImageMagick filters. The *-implode* filter gives you fun results, dragging the pixels towards the center of the image (see Figure 4).

convert is really useful if you need to automate the processing of a large number of files. For example, you might need to convert all the JPEG images in a directory into thumbnails of the same size. A script like the one shown in Listing 1 would do this for you.

Files with the *.jpg* extension are assigned to the *i* variable. Then, *convert -geometry '75x75!'* iterates against the variable. The first parameter is the input file (*$i*) and the second is the output file, which derives its name from the base filename without the extension, the *_small* string, and finally the extension *.jpg*.

A slightly easier way that requires less typing is to use *mogrify* – no scripting is needed if you choose to use this little program to create your thumbnails for all JPEG files in the current directory:

```
mogrify -sample '75x75!' *.jpg
```

The only disadvantage is that *mogrify* will overwrite the original images. One solution to solve this problem is that you specify a new output format by using the *-format* option:

```
mogrify -format png -sample '75x75!' *.jpg
```

The thumbnails are now being stored in the PNG format, while the original JPEGs remain untouched.

In the Eye of the Beholder

ImageMagick also includes a viewer program: *display* is the suite's image processing and displaying tool. To start the viewer, type *display* followed by the images you want to load (i.e. *display image.bmp*). If you want to look at all the JPEG files from one directory, you can load the whole lot by typing *display *.jpg*. Use the [Space] key to go to the next image, the [backspace] key to go back and [Q] to quit. Alternatively, you can open *display*'s menu by clicking with the right mouse button into the image window and find buttons for *Next*, *Former* and *Quit* in the *File* menu.

Most of the functions you've read about in this article can be found in the various menu entries. So, if you wanted to crop, chop or rotate a picture, instead of typing the command on the command line you could also go to the *Transform* menu and click on the appropriate button. *display* also understands most of ImageMagick's common command line options when you start the program. For example, to scale an image to 1024x768 pixels at program start, you can type:

```
display -geometry 1024x768 chicken.png
```

Similarly, you can define that a picture should be presented in black and white, that ImageMagick should add a frame, etc. One very nice feature of *display* is the ability to create HTML galleries with only a few clicks: Start the program by typing *display* on the command line. Right click the window to open the menu. Next go to *File / Visual Directory* and navigate to the directory which contains the gallery pictures. Click on the button *Directory* and *display* starts to create thumbnails for you. Depending on the number of pictures, this may take a while.

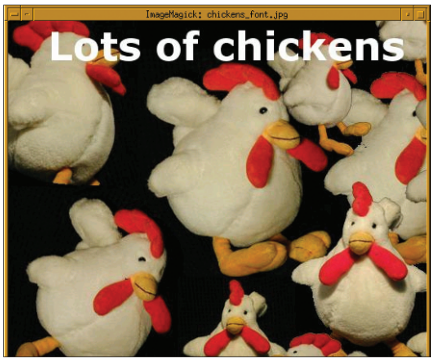


Figure 3: Scripted chickens.

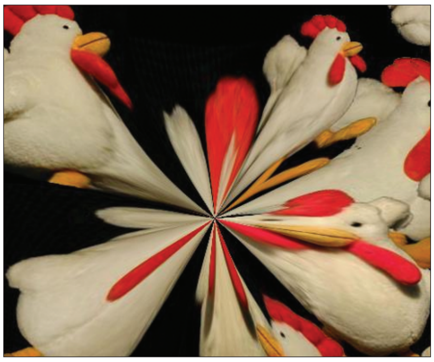


Figure 4: Imploding chickens.

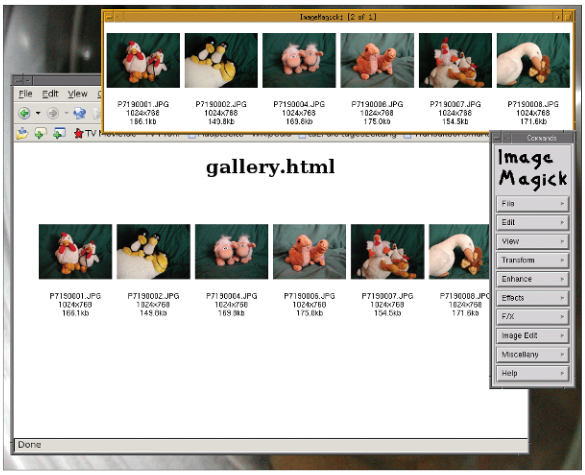

gallery.html
File
Edit
View
P7190001.JPG 1024x768 186.1kb
P7190002.JPG 1024x768 149.8kb
P7190004.JPG 1024x768 169.8kb
P7190006.JPG 1024x768 175.0kb
P7190007.JPG 1024x768 154.5kb
P7190008.JPG 1024x768 171.6kb
Image Magick
File
Edit
View
Transform
Enhance
Effects
F/X
Image Edit
Miscellany
Help
Done

Figure 5: *display* creates HTML galleries for you.

The new picture shows thumbnails of all the images, plus the filenames and sizes. To create an HTML gallery, just save the image with the thumbnails in HTML format by clicking on *File / Save* and choosing the HTML format. In your current directory you will find one *.html* file, one *.shtml* file with an image map and one transparent GIF with the thumbnails. You can now have a look at your new gallery by opening it in a browser – you should be able to click on each thumbnail to navigate through the pictures (see Figure 5).

The Gathering

Working with factors and positions in the command line may seem convoluted in comparison to image manipulation programs like The Gimp, but the ImageMagick tools are hard to beat when it comes to automated image processing. Web developers and administrators who frequently have to manipulate large collections of pictures can include their favorite program calls into shell scripts – an attractive alternative to clicking with the mouse. Even if you only need to apply one or two changes, typing on the command line can be much faster than a GUI and save you a lot of time. ■

Listing 1: Creating Thumbnails

```
#!/bin/bash

for i in *.jpg; do
  convert $i -geometry '75x75!' `basename $i .jpg`_small.jpg
done
```