rmagick || minimagick

Picking the right Magick library for your app.

ImageMagick (& GraphicsMagick)

- open-source image processing
- identification, drawing, typography, transformation & conversion
- image formats are modular (challenging to install)
- used via C API or shell commands
- many hosts provide xMagick

Differing Semantics

rmagick

conventional
methods & params

RMagick::Image instance
contains image data

returns image object

minimagick

interface to
shell command

MiniMagick::Image instance
references a temp file

returns shell command output,
not an image object

Usage Comparison

rmagick

```
def resize_and_crop(image, square_size)
  geometry = to_geometry(
    square_size, square_size)
  if image.columns < image.rows
    image.crop!(
      Magick::CenterGravity,
      image.columns,
      image.columns,
      true)
  elsif image.columns > image.rows
    image.crop!(
      Magick::CenterGravity,
      image.rows,
      image.rows,
      true)
  end
  image.change_geometry(geometry) do
    |cols,rows,img|
    img.resize!(cols,rows)
  end
end

new_image =
  resize_and_crop(
    Magick::Image.from_blob(
      File.read("1.jpg")).first )
```

minimagick

```
def resize_and_crop(image, square_size)
  geometry = to_geometry(
    square_size, square_size)
  if image[:width] < image[:height]
    shave_off = ((
      image[:height]-
      image[:width])/2).round
    image.shave("0x#{shave_off}")

  elsif image[:width] > image[:height]
    shave_off = ((
      image[:width]-
      image[:height])/2).round
    image.shave("#{shave_off}x0")

  end
  image.resize(geometry)


  return image
end

new_image =
  resize_and_crop(
    MiniMagick::Image.from_file("1.jpg"))
```

Method Call Differences

rmagick

image processing methods are bound to the Magick API

```
image.unsharp_mask(
  UnsharpMaskRadius,
  UnsharpMaskSigma,
  UnsharpMaskAmount,
  UnsharpMaskThreshold)
```

minimagick

image processing methods are undefined

#method_missing triggers a call to the system command `mogrify`:

- method name as the first option
- a single string parameter containing all the remaining mogrify arguments

```
image.unsharp(
  "#{UnsharpMaskRadius}x"+
  "#{UnsharpMaskSigma}+"+
  "#{UnsharpMaskAmount}+"+
  "#{UnsharpMaskThreshold}")
```

minimagick Tempfile gotcha

- Ruby's `Tempfile` naming scheme uses a meaningless numerical extension
- mini trips on erroneous file extensions
- *solution:* redefine `Tempfile#make_tmpname` to avoid the extension
- see: detailed article
  http://marsorange.com/archives/of-mogrify-ruby-tempfile-dynamic-class-definitions

image processing != cheap

host-imposed limits
processor utilization
memory footprint

Runtime Characteristics

rmagick

instance contains image data;
up to 3x pixel map size

memory shared with Rails;
explicit garbage collection helps

processor usage **accumulates**
on the Rails dispatcher

minimagick

instance references temp file;
a tiny stub

separate memory allocation
for process `mogrify`

`mogrify` is a **short-lived**,
per-method system call

Select a Library

purpose	rmagick	minimagick
create images: drawings, graphs, typography…	**yes**	**no** (unsupported)
resize, sharpen, tranform & convert existing images	**yes** (limited on shared servers)	**yes**
compositing images	**yes** (limited on shared servers)	**no** (unsupported)

Ruby-Magick Resources

rmagick

project site
http://rmagick.rubyforge.org/

full documentation:
rmagick
http://studio.imagemagick.org/RMagick/doc/

minimagick

project site
http://rubyforge.org/projects/mini-magick/

basic usage & examples:
see the included README

command reference:
mogrify
http://www.imagemagick.org/script/command-line-options.php

presented by Mars

http://marsorange.com/