Fundamentals of Computer Science I (CS151.01 2006F)

Generating GIMP Images with Script-Fu

Summary: In this laboratory, you will begin to investigate the ways in which you can use Scheme to script the GNU Image Manipulation Program. Because you will often be nesting procedures within other procedures, is important to keep in mind what your procedures are returning, and what you are giving procedures as arguments.

Contents:

Exercises

Exercise 0: Getting Started

a. Start the GIMP by typing `gimp` in a terminal window. If you haven’t run the GIMP before, you should allow the GIMP to do the normal installation procedure.

b. Select **Console** from the **Script-Fu** menu available from under the **Xtns** menu in the primary GIMP palette. You should see a window appear with some text at the top and an entry box at the bottom. This is the console for interacting with Script-Fu, which is based on a *variant* of Scheme called SIOD.

c. Type a few, simple, one-line Scheme expressions to see what happens. For example, you might ask for a few simple sums, extract parts of lists, or whatever.

d. To do most of this lab, you will need to load our library of helper procedures, which you do with the command Load them with

```
(load "/home/rebelsky/Web/Courses/CS151/2006F/Examples/gimp.scm")
```

If you’re working on the lab at home, make your own copy of `gimp.scm` on your desktop and follow the instructions for loading it from the reading.

Exercise 1: Drawing Your First Images

As the reading suggests, to create a new image you need to call `(create-image *width height*)`. You also need to name that image so that you can refer to it later.

a. Define the width and height of your new image with something like the following.

```
(define width 256)
(define height 384)
```

b. Create and name your new image with

```
(define image (create-image width height))
```

The image will not yet appear. Script-Fu assumes that you may want to do some things with the image before you show it on the screen. In this case, we have nothing to do first.

c. We’re now ready to show the image. You need only one line.

```
(show-image image)
```

d. Draw a bit in the image with the paint tool.

e. You can clear the image with the following sequence of commands:

```
(select-all image)
(clear-selection image)
(select-nothing image)
```

Try that sequence and ensure that it works as advertised.

Exercise 2: Simple Drawing

a. As you know, in order to draw you need a foreground color and a background color. Let’s try setting them to something simple.

```
(set-fgcolor RED)
(set-bgcolor YELLOW)
```

If you wish, you can also set the color using a list of the individual red, green, and blue components, each of which must be a number between 0 and 255, inclusive.

```
(set-fgcolor (list 255 127 127))
```

b. Using the current pen, manually draw to ensure that the colors have worked correctly.

c. What do you expect to happen if you clear your drawing, using the commands given in the previous section?

d. Verify your answer experimentally.

e. Although you can set colors numerically, many people find it more useful to refer to them with names. The name `colors` provides a list of all the colors we've named. Look at that variable and then pick an interesting foreground and background color.

Exercise 3: Paintbrushes

In order to draw, you need to be able to select a paintbrush. As you may recall from the reading, you set the brush with `set-brush`.

a. Set the current brush to the small circle with

```
(set-brush "Circle (03)")
```

b. Verify that the command worked as expected.

c. What do you expect to happen if you try to set the brush to an invalid name (e.g., `"Grinnell"`)?

d. Verify your answer experimentally.

e. Get a list of all brushes with `(list-brushes)`. Choose an interesting one.

Exercise 4: Selecting

As many of you discovered on the first day, the simplest drawing technique is to select something and then "stroke" it. If you don't know, "stroking" is telling the GIMP to apply the brush to a selection, cutting out the hassle of drawing freehand with a mouse.

a. Use that technique to draw a box from Script-fu.

```
(select-rectangle image REPLACE 10 10 100 50)
(stroke image)
(select-nothing image)
```

b. Use that technique to draw a circle in a different color.

```
(set-fgcolor (list 255 0 0))
(select-ellipse image REPLACE 50 50 80 80)
(stroke image)
(select-nothing image)
```

c. Experiment with selecting more complex areas using the `ADD`, `SUBTRACT`, and `INTERSECT` options.

Exercise 5: Line Drawing

Another way to draw in Scheme is to use the paintbrush to paint a straight line. (Any more complex line can be broken down into a series of small straight lines.) You draw a line with the `line` command.

a. Draw a line from the upper-left-hand corner to the lower-right-hand corner with

```
(line image 0 0 width height)
```

b. Draw an isoceles triangle with width 50 and height 30 by drawing three appropriate lines.

c. Change the brush and color using Script-Fu commands and verify that the `line` procedure now uses the current brush and foreground color.

Exercise 6: Designing Your Own Images

a. Using the commands that you learned in the previous exercise, tell the GIMP to create a new image in which you use Script-Fu to draw something interesting (perhaps a smiley face, perhaps a stick figure, perhaps something more complicated).

b. What advantages, if any, are there to using Script-Fu to create that image?

Exercise 7: Generalizing Your Image

a. Using the commands you have used in previous exercises, but not `set-fgcolor`, `set-bgcolor`, and `set-brush`, write a series of instructions to make the GIMP draw something. You should, of course, use the current brush, foreground color, and background color in your simple illustration.

b. Choose “random” foreground and background colors with

```
(set-fgcolor (list (random 256) (random 256) (random 256)))
(set-bgcolor (list (random 256) (random 256) (random 256)))
```

c. Choose a “random” brush using the following technique.

i. Make a list of potential brushes. For example,

```
(define brushes (list "Felt Pen" "Pencil Sketch" "Calligraphic Brush" "Circle Fuzzy (05)"))
```

ii. You can select a random element from that list using `list-ref`

```
(define some-brush (list-ref brushes (random (length brushes))))
```

iii. You can then set the brush with

```
(set-brush some-brush)
```

d. Repeat the commands to draw the figure using the new color and brush.
