

Laboratory: Writing Script-Fu Procedures

Summary: In today's laboratory, you will write and install new procedures for use within Script-Fu and the Gimp.

Contents:

- Exercises
 - Exercise 0: Preparation
 - Exercise 1: Grouping Commands
 - Exercise 2: Writing Your Own Procedure
 - Exercise 3: Adding Parameters
 - Exercise 4: Generalizing Your Procedure
 - Exercise 5: Telling DrScheme About Procedures
 - Exercise 6: Writing Helper Procedures

Exercises

Exercise 0: Preparation

a. Open a terminal window.

b. Type the following (substituting your user name)

```
ln -s /home/rebelsky/Web/Courses/CS151/2006F/Examples/gimp.scm /home/username/.gimp-2.2/scripts/gimp.scm
```

(If you are working outside the MathLAN, you'll need to save a copy of `gimp.scm` and put it in a similar folder. On a Macintosh running OS X, save the file on the desktop, open a terminal window (in the Applications/Utilities folder), and type `mv Desktop/gimp.scm .gimp-2.2/scripts/gimp.scm`. Windows users get to figure it out themselves.)

c. Start GIMP by typing `gimp` in the terminal window.

d. Open a new Script-Fu console.

e. If all has gone well, you should be able to use our GIMP utilities without loading the file. To ensure that you can, create an image and draw an oval.

```
(set-fgcolor GREEN)
(set-bgcolor BLACK)
(define img (create-image 100 100))
(select-ellipse img REPLACE 20 20 60 60)
(set-bgcolor WHITE)
(fill-bgcolor img)
(stroke img)
(show-image img)
```

Exercise 1: Grouping Commands

In the corresponding reading, I provided code for a procedure called `no-failure`. You can find that procedure in the file `no-failure.scm`.

- a. Make your own copy of that file.
- b. Load the file into the Script-Fu console by typing `(load filename)`. For example, if you've stored your stuff as `images.scm` on your desktop, you should use

```
(load "/home/username/Desktop/no-failure.scm")
```
- c. Type `(no-failure)` in the Script-Fu console and see what happens.
- d. Open `no-failure.scm` in DrScheme. Change the background color (say, to GREY) and the color for the letter (say, to BLACK). Save the file from DrScheme.
- e. In the Script-Fu console, reload `no-failure.scm` and type `(no-failure)`.

Exercise 2: Writing Your Own Procedure

Using `no-failure` as a template, write and test your own procedure to draw an interesting picture. Depending on your skill level, you might draw a smiley face, a stick figure, or something else.

Exercise 3: Adding Parameters

One disadvantage of `no-failure` is that it always draws an image of the same size. What if we want a bigger or smaller image? One possibility is to make the image size a parameter to `no-failure` and then make the various numbers depend on the size. Since our image is naturally square, the width and height should probably be the same. You can find the modified procedure in the file `new-no-failure.scm`.

- a. Make your own copy of this file.
- b. Load the file into the Script-Fu console.
- c. Test the procedure by typing each of the following commands in the Script-Fu console.

```
(new-no-failure 256 GREEN)
(new-no-failure 100 PURPLE)
```

Exercise 4: Generalizing Your Procedure

Generalize your procedure from exercise 2 so that it takes the image size as a parameter. You may choose to build square images (as I did) and take only the length of a side as a parameter or you may choose to build rectangular images and take both width and height as parameters.

Exercise 5: Telling DrScheme About Procedures

As the reading suggests, you can make your own procedures available in the GIMP menus.

- a. Make a copy of `no-failure-menu.scm` in the directory `/home/username/.gimp-2.2/scripts`
- b. Tell GIMP to refresh its Script-Fu list by selecting Refresh from the Script-Fu menu in the Xtns menu.
- c. You should now be able to select No Failure from a new Sample submenu of the Script-Fu submenu of the Xtns menu. Try it and see what happens.
- d. Using `no-failure-menu.scm` as a template, register your own script from the previous exercise. See if you can get it to run.

Exercise 6: Writing Helper Procedures

When doing more complex drawings, you will often find that you draw the same thing again and again at different places and in different sizes. Hence, it is sometimes useful to write helper procedures that can do the precise drawing for you. For example, I decided that I often make the "No" sign, and wrote a helper procedure for that sign. Here's my code.

```
;;; Procedure:
;;;   forbidden
;;; Parameters:
;;;   image, an image
;;;   x, the x coordinate of the center of the sign
;;;   y, the y coordinate of the center of the sign
;;;   radius, the radius of the sign
;;; Purpose:
;;;   Draws a "forbidden" sign (a circle with a slash) in
;;;   red centered at the specified location.
;;; Preconditions:
;;;   image and layer are initialized.
;;;   x, y, and radius are not negative.
;;; Postconditions:
;;;   The image now contains the specified sign.
;;;   The brush and foreground color may have changed.
(define forbidden
  (lambda (image x y radius)
    ; Choose a color and paintbrush that seem appropriate.
    (set-fgcolor RED)
    (set-brush "Circle (05)")
    ; Select the ellipse for the circle. Do the math to see why.
    ; the particular values were set.
    (select-ellipse image REPLACE
                     (- x radius) (- y radius)
                     (* 2 radius) (* 2 radius))
    ; Draw the nice red circle.
    (stroke image)
    ; Now we're ready to draw the slash. Some math tells us that
    ; it's offset by radius/(sqrt 2) from the center (in each
    ; direction).
```

```
(let ((offset (/ radius (sqrt 2))))  
  (line image  
    (- x offset) (- y offset)  
    (+ x offset) (+ y offset)))  
; Unselect all  
(select-nothing image)))
```

- a. Store that code in a new file.
 - b. Create a new image programmatically and use `forbid` to draw on that image in various places and sizes.
 - c. Write your own helper procedure to draw a rectangle.
 - d. Write your own helper procedure to draw a circle.
-

Copyright © 2006 Samuel A. Rebelsky. This work is licensed under a Creative Commons Attribution-NonCommercial 2.5 License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/2.5/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.