Class 08: Writing Your Own Procedures

Held: Friday, February 5, 2010

Summary: We begin to consider how you can write your own procedures and why you might do so.

Related Pages:

- EBoard.
- Reading: Writing Your Own Procedures & How Scheme Evaluates Expressions (version 2).

Notes:

- There are no additional readings for Monday. You should, however, review the readings you did for today.
- Quiz 2 distributed. Due at midnight tonight. See the quiz policies.
- Assignment 3 distributed. Due next Wednesday.
- EC for today's CS table.
- EC for tonight's support event for the Booth family.
- Continue to use the same lab partners today.
- Finish Wednesday's lab before starting today's lab. Expect to finish today's lab on Monday.

Overview:

- Why define your own procedures?
- How to define your own procedures.

User Defined Procedures

- It's clear that programmers often want to (and need to) define their own procedures.
- By "their own procedures", we mean collections of Scheme commands that are parameterized and referred to be a single name, just like the built-in procedures, such as square and +, or the DrFu procedures, such as drawing-hshift.
- Procedures take inputs (which we call parameters) and may produce a result.
- Some procedures modify their parameters:
  - An "open jar" procedure changes the state of the jar
  - `image-fill!` adds virtual paint to an image.
- Some procedures create new values, without modifying their parameters:
- User-defined procedures can add clarity to a program.
  - Rather than looking at *how* code does something, the user of a procedure can focus on *what* the code does.
  - A reader of the program is much more likely to understand a procedure call than the body of the procedures.

- Programmers can avoid repetitive (and, therefore, error prone) code.
  - Rather than retyping the same code again and again, just changing a few values, a programmer can give a name to the same code.
- How do you define your own procedures? Using the following template:

```
(define your-procedure
  (lambda (param1 ... paramn )
    expression1
    ...
    expressionm ))
```

- For example,

```
(define square
  (lambda (val)
    (* val val)))
```

- You can (and should) document your procedures so that others can understand what they are supposed to do. We'll come back to this issue in about a week.
- When the body of a procedure has multiple expressions (as in the template), only the value of the last procedure is returned.

Lab on User-Defined Procedures

- Do the lab.
- Be prepared to reflect (e.g., to describe the most important or most confusing thing you dealt with today). (And no, you can't say "Sam is the most confusing thing I dealt with today.")
