

Class 14: Documenting Programs and Procedures

Held: Monday, 21 September 2009

Summary: We consider documentation for your programs - Why to write it, when to write it, how to write it.

Related Pages:

- EBoard.
- Reading: Documenting Your Procedures.

Notes:

- We potentially have prospectives in class today.
- Are there questions on Exam 1?
- There was a mixup on Writeup 3. If you can't get it in until later this week, there will be no penalty.
- We will go over efficient checkerboard drawing today.
- There's a new Mac Version of GIMP+MediaScript (0.1.1.15). You will probably need it in order to work on the next few labs.
- Reading for tomorrow: Homogeneous Lists.
- Writeup 2 returned. Check means "fine". All checks will earn you an A in writeups. If you did not get a grade on writeup 2, please talk to me asap.
- HW 3 returned. Check means "fine". All checks will earn you a B in homework. (How's that for consistency?)

Overview:

- The need for documentation.
- The Six P's - a strategy for documenting procedures.
- Practice.

Documentation

- Why document? Text is often easier to read than code.
- Who is your audience? You have many audiences.
 - People who must *maintain* and *update* your code.
 - People who must *use* your code.
 - People who *incorporate* your code into a bigger package (arguably, these are people who do both of the previous things).
- You write comments *within* code for those who must maintain and update your code. You often also write a *big picture* statement for such folks.
- Most of the comments you write in this class will be for people who are likely to use your code. We call these your *client programmers*.

- For such folks, we focus on *what* your procedures do.

The Six P's

- I prefer to document procedures for client programmers using a careful system, in which we always describe six aspects of the procedure: The name, the parameters, the purpose, the value the procedure produces, the preconditions, and the postconditions.
- The *Purpose* is intended to give a short, informal, summary of what the procedure does.
- The other three of the first four parameters are used to name things, so that we can refer to them elsewhere.
- The *Preconditions* formalize requirements that must be met in order for the procedure to function correctly.
- The *Postconditions* formalize the results of the procedure. They are often expressed in a combination of mathematics, Scheme, and English.

Practice

- We'll try writing documentation collaboratively for a variety of procedures.
- `(sqrt val)`
- `(square val)`
- `(rbg-redder val)`
- `(max val1 val2 ... valn)`
- ...

Copyright © 2007-9 Janet Davis, Matthew Kluber, Samuel A. Rebelsky, and Jerod Weinman. (Selected materials copyright by John David Stone and Henry Walker and used by permission.) This material is based upon work partially supported by the National Science Foundation under Grant No. CCLI-0633090. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. 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.