

Golden Rules for Developers

Steven Feuerstein
PL/SQL Evangelist, Quest Software

steven.feuerstein@quest.com

www.StevenFeuerstein.com

www.ToadWorld.com/SF

www.PLSQLChallenge.com

What is a "Golden Rule"?



- In the context of software, a golden rule is one that encourages you to write code that even ***you*** would want to ***maintain*** long after the original author left.

The Logic Behind My Golden Rules

- Tools help a lot, but for the most part writing "golden" code comes down to attitude, personal discipline and good habits.
 - There's an awful lot we can do all by ourselves.
- This presentation focuses on actions we can each take to make a big difference.

You have the power.

Golden Rules for Programmers

- Don't repeat anything.
- Hide *everything*.
- Don't take shortcuts.
- Embrace standards.
- Build on a foundation.
- Never lose information.
- Don't write code alone.

Don't repeat anything.

- This is Golden Rule Numero Uno.
- Repetition = hard coding = maintenance nightmare.
- Instead, aim for a Single Point of Definition (SPOD) for every aspect of your application.
 - Formulas and business rules
 - Magic values
 - SQL statements
- The hard part is recognizing all the sorts of hard-codings that can occur.

```
1  PROCEDURE process_employee (department_id_in IN NUMBER)
2  IS
3      l_id      INTEGER; l_salary NUMBER (9,2);
4      l_name    VARCHAR2 (100);
5
6      /* Full name: LAST COMMA FIRST (ReqDoc 123.A.47) */
7      CURSOR emps_in_dept_cur
8      IS
9          SELECT employee_id, salary, last_name || ',' || first_name lname
10         FROM employees
11        WHERE department_id = department_id_in;
12 BEGIN
13     OPEN emps_in_dept_cur;
14
15     LOOP
16         FETCH emps_in_dept_cur
17         INTO l_id, l_salary, l_name;
18
19         IF l_salary > 10000000 THEN adjust_comp_for_ceo (l_salary);
20         ELSE analyze_compensation (l_id, l_salary, 10000000); END IF;
21
22         EXIT WHEN emps_in_dept_cur%NOTFOUND;
23     END LOOP;
24     COMMIT;
25 EXCEPTION WHEN NO_DATA_FOUND THEN
26     RAISE_APPLICATION_ERROR (-20907, 'Invalid department ID');
27 END;
```

Hide *everything*.

- Information hiding is closely linked to SPOD.
- The more you *expose*, the less likely it is that you will be able to...
 - Workaround weaknesses in Oracle's solutions
 - Easily take advantage of new features from Oracle
 - Correct design and implementation mistakes in your own code.
- So put your own layer of code around just about everything....

What to Hide

- Hard-codings, as previously discussed
- Calls to most supplied package subprograms
 - UTL_FILE, DBMS_OUTPUT, etc.
- The *way* you do get things done
 - Writing to an error log is a great example.
 - Working with complex data structures, like multi-level collections

```
get_next_line.sqp  
awful_exception_section.sql  
string_tracker2.*
```

Don't take shortcuts.

- You know when you're doing it.
- You know it's wrong.
- You know you will regret it.
- Yet you, we, I do it anyway.
- Resist!
- Be disciplined.
- Say "no" to shortcuts.
- Take that little bit of extra time to do it right.

**Listen to that
little voice in
your head.**

**Especially
when it's
SHOUTING.**

Embrace standards.

- Developers often seem to resist standards.
 - "Don't tell me how to write my code!"
- Instead, we should *embrace* standards.
 - Standards only cover the least interesting and most tedious aspects of development.
 - Standards free up your time to focus on high level and more creative aspects.
- Visit the [PL/SQL Standards](#) page at PL/SQL Obsession for some starting points.

PL/SQL Obsession: www.ToadWorld.com/SF

Build on a foundation.

- What a strange language PL/SQL is!
 - Or at least: what strange programmers we are.
- Imagine – a mainstream programming language with *no third party libraries*.
- We build everything "from scratch."
- *Surely*, within each of our application development projects, we can put in a place a foundation of reusable code.
 - Error manager
 - Common utilities (string parser, Boolean converter, etc.)

Building Blocks of your Foundation

- PL/SQL Obsession
 - The demo.zip file
 - Quest Error Manager
 - Quest CodeGen Utility
- Oracle Technology Network
 - Growing body of shared code
- PL/Vision library – but quite old

Never lose information.

- We never do anything or need any piece of data *just once*.
 - Software development is (a little bit) like the great cycle of life.
- So make sure that anything you learn, anything you write, will not be lost.
- This is one of the weakest aspects of the Oracle data dictionary and even third-party IDE tools.
 - No modeling of *application*
 - Limited ability to search code intelligently

Never lose information (2)

- I plan to build an online code/script-sharing repository.
 - Should be available later in 2010
 - *Not* the same as a library, but it should be a big help.
- For now, I simply *always* save scripts I write, no matter how trivial, to a separate directory and at least can search there.

Don't write code alone.

- Even as members of a team, we tend to write our code in isolation.
 - And most of our code isn't even ever reviewed by another human being.
- In this mode, most of our ideas and questions and worries stay inside our head.
 - A very unreliable piece of real estate.
- It's key that you look for opportunities to get ideas out of your head and bounce them off of others.

Writing code with others

- Pair programming: no one codes alone!
 - "Extreme" but powerful
- Regular code review meetings
 - Build team as you share knowledge
- "Buddy system"
 - Private code review
- Ask for help
 - Sooner rather than later
 - Doesn't really matter too much who or how you ask.

Golden Rules for Programmers

- Don't repeat anything.
- Hide *everything*.
- Don't take shortcuts.
- Embrace standards.
- Build on a foundation.
- Never lose information.
- Don't write code alone.

Golden Rule Software

- If you are able to establish new programming habits built around my golden rules, the resulting code will be a joy to behold – and to maintain.
 - That helps *you* now and helps so many others in the future.
- So remember....

**Code unto others as you would
have them code unto you.**

Take the PL/SQL Challenge!

- A daily quiz that culminates in a quarterly championship tournament.
 - *Win cash* and other prizes just for your knowledge of PL/SQL.
- And sign up for my monthly newsletter at....



**steven
feuerstein.com**
Obsessed with PL/SQL...and that's ok with me.



home

about
steven

read
steven

hire
steven

learn
pl/sql

change
the world