Unit testing in PL/SQL

Catherine Devlin
Intellitech
catherine_devlin@urscorp.com
Of course I test my code!

SQL> r
1  SELECT dates.workdays_between(
2       '20-OCT-2003','22-OCT-2003')
3       "Result"
4* FROM   dual

           Result
----------
          2

SQL>
And my users test it even better!

('20-OCT-2003','22-OCT-2003') Monday to Wednesday
('22-OCT-2003','20-OCT-2003') order is reversed
('23-OCT-2003','26-OCT-2003') today to Sunday
('20-OCT-2003', NULL) umm... NULL?
('Oct. 20', 'Oct. 22') unexpected format
('Oct. 20', 'Oct. 32') invalid date

DELETE FROM office_holidays; Hey!
Testing should be...

thorough
  *Everything that can go wrong...*

repeated
  *Write “A” and test it*
  *Write “B”… “B” breaks “A”*

fine-scale
  *“The database doesn't work.”*

Exactly *what doesn’t work,*
*in exactly what circumstances?*
OK, it’s on the list

- Floss daily
- Call Aunt
- Eat 5 fruits and vegetables
- Rehearse recovering database after media failure
- 20 minutes/day aerobic exercise
- Keep tires fully inflated
- Test PL/SQL code properly
- Stop eating from vending machine
- Document code well
- Vacuum under furniture
Unit testing – the methodology

1. Separate test for each identifiable task
2. Use code to test code

exec test_bread();
   Water quantity OK? pass
   Water temperature OK? pass
   Flour quantity OK? pass
   Yeast active? fail
   Sugar quantity OK? pass
   Salt quantity OK? pass

Thorough, repeated, fine-scale
3. Use a “test framework”

To the rescue!
utPLSQL

Free, open-source, PL/SQL packages

1. Download from sourceforge.net
2. SQL> @utplsql_install

3. Write test package ut_my_function
4. exec utplsql.test('my_function')

Requirements: UTL_FILE_DIR assigned, DBMS_PIPE granted
CREATE OR REPLACE PACKAGE ut_DATES
IS
  PROCEDURE ut_setup;
  PROCEDURE ut_teardown;
  PROCEDURE ut_IS_WEEKEND;
END ut_DATES;
/

CREATE OR REPLACE PACKAGE BODY ut_DATES IS
PROCEDURE ut_setup IS
  BEGIN
    NULL;
  END;
PROCEDURE ut_teardown IS
  BEGIN
    NULL;
  END;
PROCEDURE ut_IS_WEEKEND IS
  BEGIN
    utAssert.this('Saturday is weekend',
                  dates.is_weekend('25-OCT-03'));
    utAssert.this('Monday is not weekend', NOT(dates.is_weekend('27-OCT-03')));
  END ut_IS_WEEKEND;
END ut_DATES;
/
demo running exec utplsql.test('dates');
utAssert: “Test this condition and report”

utAssert.this('Force success', TRUE);
    SUCCESS - demo.UT_DEMO: Force success

utAssert.this('Force failure', FALSE);
    FAILURE - demo.UT_DEMO: Force failure

utAssert.eq('Subtraction', mysubtract(4,2), 2);
    SUCCESS - demo.UT_DEMO: EQ "Subtraction"
    Expected "2" and got "2"

utAssert.eq('Addition', myadd(2,2), 5);
    FAILURE - demo.UT_DEMO: EQ "Addition" Expected "5" and got "4"
utAssert.eqquery:
“Are these queries equal?”

utAssert.eqquery('Q-check',
'SELECT table_name FROM tabs',
'SELECT object_name FROM user_objects WHERE object_type = 'TABLE'');

SUCCESS - demo.UT_DEMO: EQQUERY
"check with eqquery"
Result: Result set for "SELECT table_name FROM tabs”
does match that of "SELECT object_name FROM user_objects WHERE object_type = 'TABLE'"
More utAssert procedures

```sql
utAssert.eqqueryvalue('No non-zero returned', 'SELECT MIN(my_func(n)) FROM tab1', 0);
```

```sql
utAssert.isnull('Null input -> null output', my_func(NULL));
```

```sql
utAssert.throws('Exception if input < 0', 'my_func(-1);', 'NO_DATA_FOUND');
```

... and many more ...
OUnit

GUI for executing utPLSQL tests

Free download

Easy install (and will install utPLSQL)

Windows only
Exercises for the reader

utGen: Automatically generate package skeleton

Testing DML code: ut_setup, ut_teardown

Testing triggers: remove “guts” to package or procedure

Test-first development
  http://www.extremeprogramming.org/rules/testfirst.html

Code generation with OUnit 1.5 (“Fall 2003”)

Use utPLSQL from TOAD
Web resources

http://utplsql.sourceforge.net/
  code and documentation

http://utplsql.oracledeveloper.nl/
  utPLSQL newsgroup

http://www.ounit.com/
  OUnit

http://www.stevenfeuerstein.com/

catherine_devlin@urscorp.com