

# DYNAMIC DEBUGGING

Pain-free Troubleshooting of  
Production PL/SQL

# AGENDA

- ⦿ Fluff (10 min)
- ⦿ Meat (20 min)
- ⦿ Q&A and Collaboration (15 min)

# PRODUCTION PROBLEMS

- User calls or submits ticket
  - Getting error, nothing, or taking too long
- Proactive monitoring
  - Determines errors occurring
  - Determines performance is not typical
- No monitoring or history
  - aka “Taking your lumps”
  - User, customer, manager, CEO calls...
  - Backend process doesn't complete, doesn't run, spitting errors
- Find and Fix. Then rebuild trust.

# DEBUGGING METHODS

- ◉ DBMS\_OUTPUT
- ◉ Log to file
- ◉ Log to table
- ◉ GUI tool or DBMS\_DEBUG API
- ◉ Others?

# DEBUG HEAVEN

- ◉ **Dynamic:** Can be turned on and off
  - Debug instrumentation can remain in Prod code
  - Turn on for a PL/SQL unit
  - Turn on for an existing session
  - Turn on for a named process
  - Turn on for an identifiable end user
    - Other? (IP address, Domain, Client program)
- ◉ **Visible immediately.** No waiting.
- ◉ **Output choice:** file, table, screen, other?
- ◉ **Painless for developers to learn and use**
- ◉ **Auto-report time and origination**

# WHICH WAY TO HEAVEN?

- ◉ Simple debug API
  - `dbg('Calling X with `||i_parm);`
- ◉ Table-driven parameters with defaults
  - Debug toggle
  - Debug context of interest
  - Output destinations
  - Defaults overrideable
- ◉ Robust file API for file-based
- ◉ Anonymous transaction API for table-based
- ◉ Method to identify end user

# PL/SQL STARTER FRAMEWORK

Oracle 8i - 11g Enterprise Database

Application Schema #1

Application Schema #2

Application Schema #3

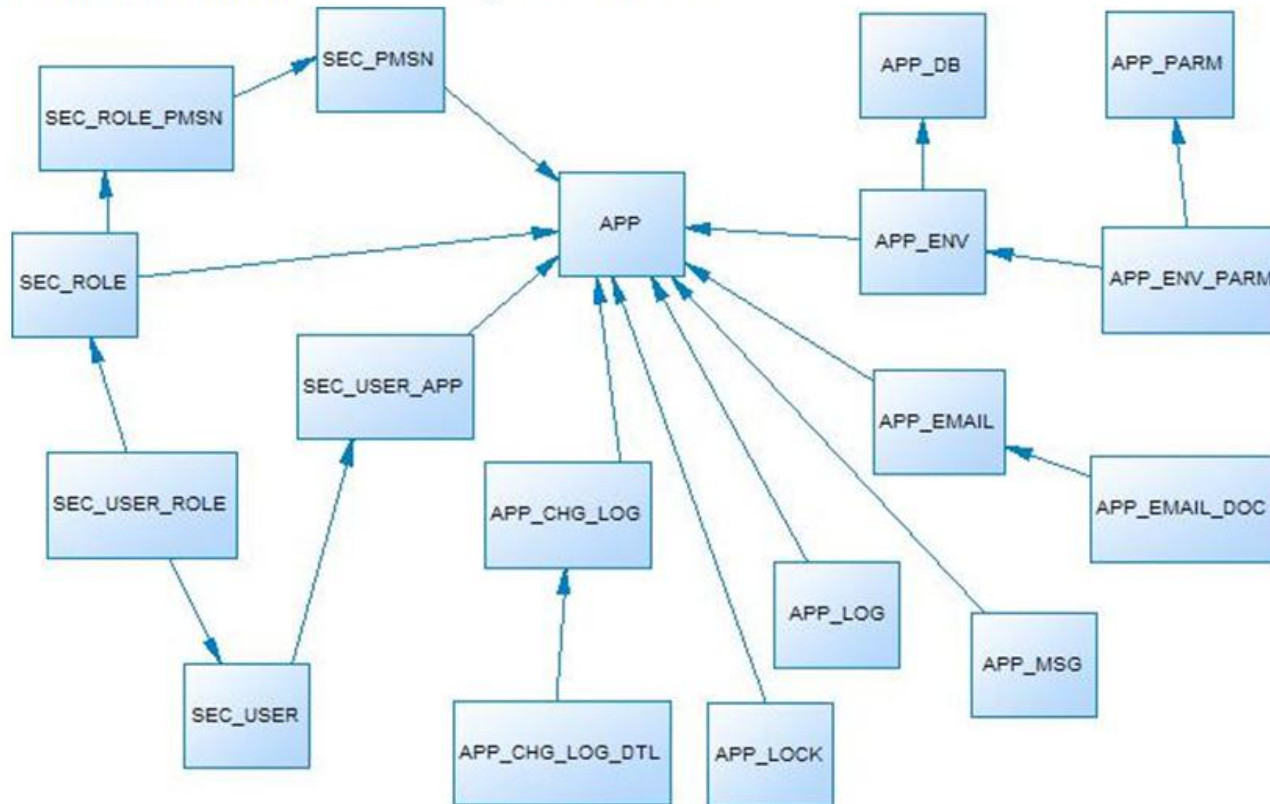
Core Schema (where PL/SQL Framework and shared business entities reside)

Higher-level Packaged Libraries: ENV, MSGS, LOGS, TIMER, MAIL, LOCKS

Lower-level Packaged Libraries: STR, NUM, DT, PARM, IO, EXCP, CNST, TYP and UDTs.

Independent DBA package for agile continuous builds: DDL\_UTILS

Generated or Customized Table API Packages: API\_APP\_LOG



SYS Schema: STANDARD, DBMS\*, UTL\*, and other built-ins

# RELEVANT PIECES

- **Debug API: LOGS**
  - `logs.dbg()`
    - Critical private routines: `check_debug_toggle()`, `env.caller_meta()`
- **Toggles: APP\_ENV\_PARM**
  - Debug (on/off, session, unit, user)
  - Debug Toggle Check Interval (in minutes)
  - Default Log Targets (Screen=N,Table=Y,File=N)
- **File API: IO**
- **Log Table API: APP\_LOG\_API**
- **End-to-End User Identification**
  - Modified Java connection classes
  - Application Context and `DBMS_SESSION.set_identifier` and `DBMS_APPLICATION_INFO` “tagging” (ENV pkg)



# IT'S GO TIME!

- CIO just called. After last night's release, she is not getting her daily email about the critical problem/solution repository.

- ⦿ Questions?

- ⦿ Ideas?

Contact: [bcoulam@yahoo.com](mailto:bcoulam@yahoo.com)

Website: [www.dbartisans.com](http://www.dbartisans.com)

Framework:

[sourceforge.net/projects/plsqlframestart/](http://sourceforge.net/projects/plsqlframestart/)