Java for PL/SQL Developers

Joe Greenwald
ODK Incorporated

Copyright 1999 - 2000
all rights reserved

Java and PL/SQL

How are
Java and PL/SQL
alike or different?

What can
I do with
Java?

Copyright 1999 - 2000
all rights reserved

Similarities and Differences

- Syntax
- Data Types
- Data Structures
- Control Structures
- Exception Handling
- Interfaces
- Database Access
- Components
- OO Paradigm
- Tools

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Syntax

- Both languages have common ancestors
  - Algol, C, Modula-2, Ada
- Syntax is similar
- Data types and scope are similar
- Structured programming is similar (use of modules)
- Java Programs (classes) more like PL/SQL packages
- No pointers
- No memory allocation
- Compiled and interpreted
- Write once – run anywhere

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Java vs. PL/SQL

```
package foo;
import java.Util.Vector;

public class Bar {

    private int age;
    private String name;

    public int getAge(){}
    public void setAge(int a){}
}
```

```
package Foo
    age number;
    name varchar2;
procedure setAge(age in number)
function getAge (empNbr in
    number) return number;

package body Foo
procedure getAge()
begin
end;
```

Copyright 1999 - 2000
all rights reserved

Data Types

- Java has two types of data types: object and non-objects (scalar types)
- PLSQL has scalar variables and record types
- Record types are similar to objects, without methods
- Java has more numeric types
- Java cannot create types like PL/SQL TYPE, but uses classes for this
- PL/SQL – implicit data type conversions

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Data Structures

- Java has complex data structures (called objects)
- PLSQL has Records
- Conceptually similar
- Records must use/be used in procedures/functions
- Java objects are tied to their functions/procedures

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Control Structures

- Blocks
- Scope
- Decision
- Loops
- Switch/case/goto
- break, continue, exit when

Copyright 1999 - 2000
all rights reserved

Exception Handling

- Exception handling is similar in both languages
- PL/SQL – exception occurs and branches
- Java – must define the block of code that is being tested

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Interfaces

- Java allows classes (types) to implement multiple "package headers"
- This is necessary for polymorphism

Copyright 1999 - 2000
all rights reserved

Database Access

- PL/SQL has embedded SQL
- Special constructs for resolving mapping to database
- Java uses JDBC to connect
- JDeveloper can make this easier
- SQLJ can duplicate PL/SQL-SQL interaction, with a more complex syntax

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Components

- Java Beans are reusable software components
- Graphical or non-graphical
- Can extend, add to or replace exiting components
- Components created can be reused anywhere
- Developer can use Java Beans (with current release) – not trivial!

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Tools

- Developer vs JDeveloper
- Developer is targeted, focused product
- JDeveloper is generic product
- Challenge for Oracle is to provide Developer functionality to JDeveloper and Java

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

OO Paradigm

- While some objects in Developer can be subclassed, PL/SQL does not really support OO paradigm:
  - Encapsulation
  - Inheritance
  - Polymorphism
- Major OO benefits are in reuse, extensibility, flexibility and adaptability

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Encapsulation

- All access to internal data, structures and implementation is hidden
- “Enforced” in Java by compiler
- Can do in PL/SQL, but is not enforced
  - use packages only to access variables: i.e. Designer API
- Changes to variable names, types, structures do not impact rest of program

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Encapsulation and Data Hiding

- If we hide the variables or the structure of a table in a function, then when changes are made to *implementation*, the user of the data or tables is protected
- The data is said to be *hidden* or *encapsulated*
- We write subroutines and functions to perform the manipulation of data for us
- Therefore, we only have to change it on one place – the subroutine. The main code is untouched.

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Encapsulation Example

- If we create a record or struct, we can have a more complex data structure:

```
TYPE StaffType IS RECORD (
  first VARCHAR2
  last  VARCHAR2
  age   SMALLINT
  pay   NUMBER)

  staff StaffType;
```

If we access this directly in a program, it can lead to side effects:

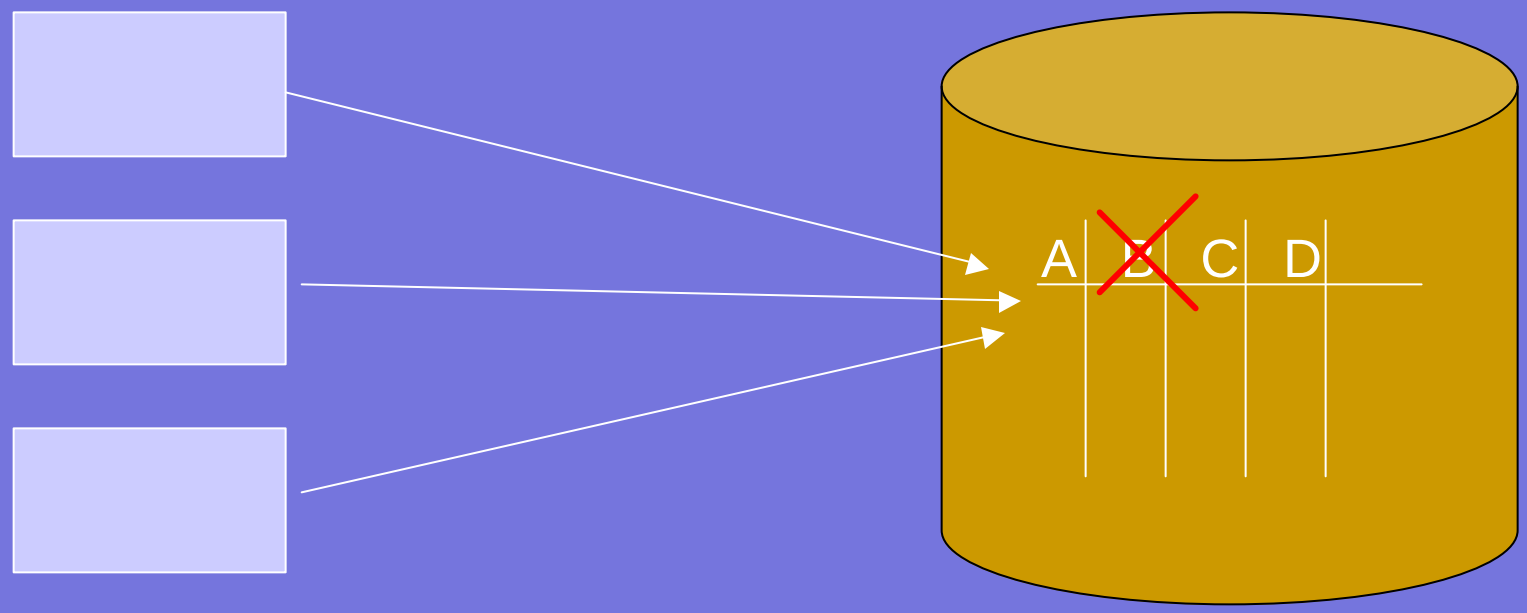
```
staff.first = “Tricia”
```

What happens if we change the variable name or how we store the name?

Copyright 1999 - 2000
all rights reserved

Access Databases

- The problem is more evident with databases
- If all programs directly access tables, what happens when the table structure changes ?



Copyright 1999 - 2000
all rights reserved

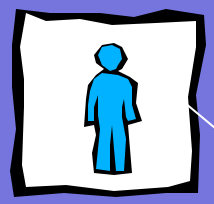
Encapsulation and Data Hiding

- If we hide the variables or the structure of a table in a function, then when changes are made to *implementation*, the user of the data or tables is protected
- The data is said to be *hidden*
- We write subroutines and functions to perform the manipulation of data for us
- Therefore, we only have to change it on one place – the subroutine. The main code is untouched.

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

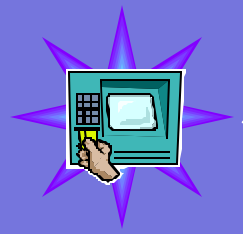
An Object's Public "Face"

- The functions and subroutines defined for an object are its public face ("interface")
- Many things can have the same interface, yet implement it differently



Person

Robot



ATM

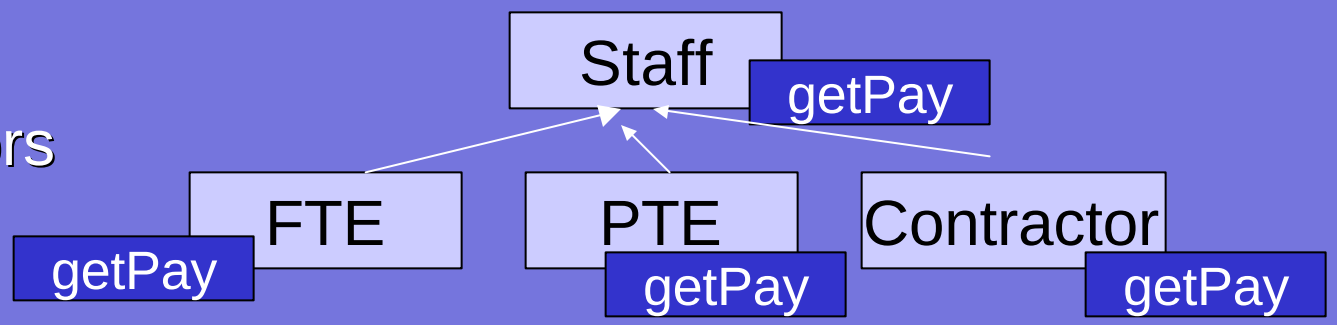
"How much money do you have?"

ODK
Objects • Data • Knowledge
training • consulting

Copyright 1999 - 2000
all rights reserved

Inheritance

- What if you had several types of Staff
  - FTE
  - PTE
  - Contractors



- Imagine you could share behavior from the more general Staff
  - getName
  - getPay
- Each type of Staff could implement getPay differently
- Why might this be a cool thing?

Copyright 1999 - 2000
all rights reserved

How to Work with Different Types

- In Procedural languages, you would need an IF..ELSE statement to determine the type and call the correct function:

```
IF (is FTE) getFTEPay (FTE)
  ELSE (is PTE) getPTEPay (PTE)
      ELSE (is Cont) getContPay(Cont)
```

OR even:

```
IF (is FTE) getPay(FTE)
  ELSE (is PTE) getPay(PTE)
      ELSE (is Cont) getPay(Cont)
```

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Polymorphism

- Object systems move the IF…ELSE to the runtime Virtual Machine:

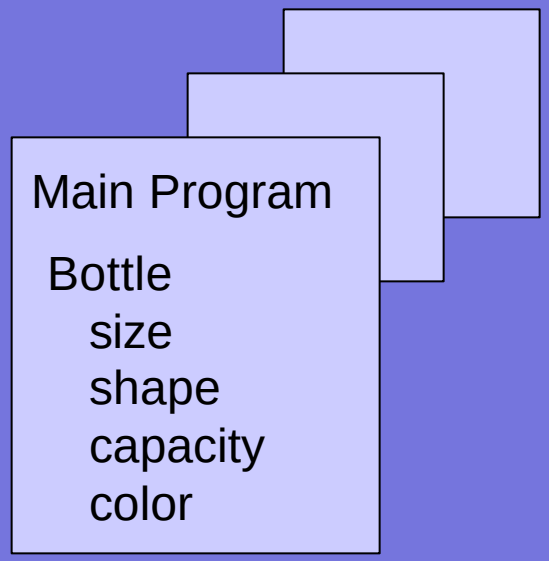
```
Staff s  = get_some_kind_of_Staff()
s.getPay()
```

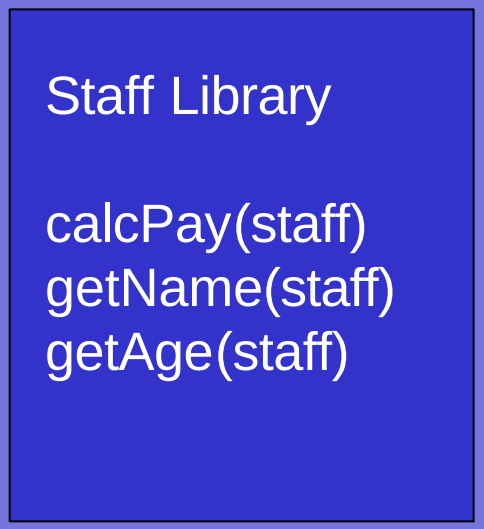
- At runtime, the system asks s “What type of object are you”
- Based on what s answers, the appropriate getPay() function is called
- This ability is called *polymorphism*

Copyright 1999 - 2000
all rights reserved

Procedural Programming

- PL/SQL programs put the data structure into the main program and call a function and pass the data structure
- What happens when the data structure changes?





Copyright 1999 - 2000
all rights reserved

Inheritance

- One objects inherits another objects data and methods
- Similar to subtype/supertype in ERDs
- Inherited code can be used, added to or overridden
- Supports the idea that a set of objects can be used in place of their parent
- Allows for higher code reusability and more flexible code
- Has its dangers

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Polymorphism

- Example: different employee type processing
- How to handle this in Java vs. PL/SQL???

Copyright 1999 - 2000
all rights reserved

OO vs. Structured

PLSQL

```
FTERecord, ContractorRecord, PTERecord
FTERec, PTERec, ContRec

Procedure UpdatePay(FTERecord)
Procedure UpdatePay(PTERecord)
Procedure UpdatePay(ContractorRecord)
If (fte) then UpdatePay(FTERec)
Elseif (pte)  then UpdatePay(PTERec)
Elseif (contractor) then UpdatePay(ContRec)
```

Java

```
Class Staff
FTE Extends Staff, PTE Extends Staff, Contractor Extends Staff

HR.ProcessPayrollChanges(Staff){
  staff.UpdatePay();}
```

Copyright 1999 - 2000
all rights reserved

Java Beans

- Java has the ability to create new, reusable objects
- These can be used within other program sand object to extend functionality
- Java Beans also broadcast events to listeners
- PL/SQL can use Java Beans in Developer Forms (not trivial!)

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

JDeveloper

- Oracle's tool for creating web applications:
  - Servlets
  - JSPs
  - Java Beans
  - XML
  - CORBA
  - EJB
  - HTML
- May eventually replace Developer as the IDE
- Must provide same functionality as Developer
- Transactions, data binding, LOVs, GUIs
- Can do this in Web (HTML) or Native Java GUI

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Connections

- JDeveloper can create connections to the server using JDBC
- Can view database structures and invoke SQL*Plus

Copyright 1999 - 2000
all rights reserved

Create Robust Applications with BC4J

- Maps Java objects to tables
- Similar to Developer
- Can create Native GUI or HTML
- Can deploy as libraries or remote, distributed objects
- Can run on any EJB server

Copyright 1999 - 2000
all rights reserved

Java or PL/SQL – How to Choose?

- Future directions
- Existing staff experience
- Need for flexibility
- Lock-in to PL/SQL and Developer
- Must ask “What problems are we facing in development” – OO/Java may not be the answer

ODK
Objects • Data • Knowledge
training • consulting
Copyright 1999 - 2000
all rights reserved

Questions?

- ODK Provides Training, advisory consulting and mentoring in Java, OO, and Oracle

Joe Greenwald

ODK Incorporated

Joe@odkinc

www.odkinc.com

Copyright 1999 - 2000
all rights reserved