

No. 1 *i*-Technology Magazine in the World

# JDJ

JDJ.SYS-CON.COM VOL.11 ISSUE:9

**AJAXWORLD**  
CONFERENCE & EXPO

SEE PAGES 58-59 FOR DETAILS

**SANTA CLARA CONVENTION CENTER** | OCT 3-4  
WWW.AJAXWORLDEXPO.COM | 2006

# SWINGING

ITS PAST,  
PRESENT,  
& FUTURE



## PLUS...

- ▶ Apache Trinidad
- ▶ Mailets and Matchers

RETAILERS PLEASE DISPLAY UNTIL NOVEMBER 30, 2006

\$5.99US \$6.99CAN 10>

0 09281 01751 6



# Fireproof Your Code

## Prevent Java code fires with JProbe®

Constantly fighting Java code fires? Prevent infernos — before code moves into production — with award-winning JProbe from Quest Software.

Quickly pinpoint Java code hot spots with line-level analysis. Discover and debug memory leaks to dramatically improve performance. Automate the task of performing code analysis during off-peak hours. And release applications with confidence, knowing they have been fully tested. JProbe is the proactive solution that gives you higher levels of productivity and end user satisfaction.

Stop Java code performance flare ups — before they start. Improve code quality and increase application efficiency with JProbe.

---

Watch JProbe in action. View the new product demo at:  
[www.quest.com/JavaCode](http://www.quest.com/JavaCode)

---

**Editorial Board**

Java EE Editor: **Yakov Fain**  
 Desktop Java Editor: **Joe Winchester**  
 Eclipse Editor: **Bill Dudney**  
 Enterprise Editor: **Ajit Sagar**  
 Java ME Editor: **Michael Yuan**  
 Back Page Editor: **Jason Bell**  
 Contributing Editor: **Calvin Austin**  
 Contributing Editor: **Rick Hightower**  
 Contributing Editor: **Tilak Mitra**  
 Founding Editor: **Sean Rhody**

**Production**

Lead Designer: **Louis F. Cuffari**  
 Executive Editor: **Nancy Valentine**  
 Research Editor: **Bahadir Karuv, Ph.D**

To submit a proposal for an article, go to  
<http://jdi.sys-con.com/main/proposal.htm>

**Subscriptions**

For subscriptions and requests for bulk orders, please send your letters to Subscription Department:

888 303-5282  
 201 802-3012  
[subscribe@sys-con.com](mailto:subscribe@sys-con.com)

Cover Price: \$5.99/issue. Domestic: \$69.99/yr. (12 Issues)  
 Canada/Mexico: \$99.99/yr. Overseas: \$99.99/yr. (U.S. Banks or Money Orders) Back Issues: \$10/ea. International \$15/ea.

**Editorial Offices**

SYS-CON Media, 577 Chestnut Ridge Rd., Woodcliff Lake, NJ 07677  
 Telephone: 201 802-3000 Fax: 201 782-9638

Java Developer's Journal (ISSN#1087-6944) is published monthly (12 times a year) for \$69.99 by SYS-CON Publications, Inc., 577 Chestnut Ridge Road, Woodcliff Lake, NJ 07677. Periodicals postage rates are paid at Woodcliff Lake, NJ 07677 and additional mailing offices. Postmaster: Send address changes to: Java Developer's Journal, SYS-CON Publications, Inc., 577 Chestnut Ridge Road, Woodcliff Lake, NJ 07677.

**©Copyright**

Copyright © 2006 by SYS-CON Publications, Inc. All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy or any information storage and retrieval system, without written permission. For promotional reprints, contact reprint coordinator Megan Mussa, [megan@sys-con.com](mailto:megan@sys-con.com). SYS-CON Media and SYS-CON Publications, Inc., reserve the right to revise, republish and authorize its readers to use the articles submitted for publication.

**Worldwide Newsstand Distribution**

Curtis Circulation Company, New Milford, NJ  
 For List Rental Information:  
 Kevin Collopy: 845 731-2684, [kevin.collopy@edithroman.com](mailto:kevin.collopy@edithroman.com)  
 Frank Cipolla: 845 731-3832, [frank.cipolla@epostdirect.com](mailto:frank.cipolla@epostdirect.com)

**Newsstand Distribution Consultant**

Brian J. Gregory/Gregory Associates/W.R.D.S.  
 732 607-9941, [BJGAssociates@cs.com](mailto:BJGAssociates@cs.com)

Java and Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc., in the United States and other countries. SYS-CON Publications, Inc., is independent of Sun Microsystems, Inc. All brand and product names used on these pages are trade names, service marks or trademarks of their respective companies.



# Social Computing Will Turn the Web World Upside Down

By Jeremy Geelan

Since most any two words can and will be put together in this world, what with us being Homo Loquens and all, it's easy just to shrug when you hear new colloquies like "social software," "social networking," or "social computing" and dismiss them as just three more inevitable permutations in a world of whirling words and phrases.

But this time, trust me, things are different. I would go so far as to say that "social computing," far from being just a random word-combo along the lines of wannabe duos like "air walking," "base jumping," "text messaging" and suchlike, is that fabled "New New Thing" (a reference to Michael Lewis's invaluable book of that name, documenting Netscape's Jim Clark's serial Webpreneurship in the heady days of the Internet Boom 1.0).

In other words, Social Computing is about to turn the Web world upside down.

Before I explain how and why, let us just lay to rest one other ghost. There will be those who, out of nothing but the sheerest prejudice against computer geeks and geekdom, suggest that "social computing" is a blatant oxymoron, right up there with "benevolent despotism." I have no truck with such bigots. On the contrary, computing - it turns out - is one of the most social technological innovations in the last thousand years.

Think I'm exaggerating? Read on.

Social Computing has been defined as centered on "software that contributes to compelling and effective social interactions" (<http://research.microsoft.com/scg/>).

At IBM Research, where the premise of the Social Computing Group is that it is possible to design "digital systems that provide a social context for our activities," the group characterizes social computing thus:

The central hallmark of social computing is that it relies on the notion of social identity: that is, it is not just the data that matters, but who that data "belongs to," and how the identity of the "owner" of that data is related to other identities in the system. More generally, social computing systems are likely to contain components that support and represent social constructs such as identity, reputation, trust, accountability, presence, social roles, and ownership.

What's the big deal? Why am claiming that Social Computing is right up there with Quantum Mechanics in terms of its likely impact on our modern world?

The answer to that question has already been hinted at by Forrester, which has published a slim, 24-page report on Social Computing subtitled "How Networks Erode Institutional Power, And What to Do About It." And it has been succinctly explicated by Dion Hinchcliffe.

Published in February of this year, the Forrester report notes:

To thrive in an era of Social Computing, companies must abandon top-down management and communication tactics, weave communities into their products and services, use employees and partners as marketers, and become part of a living fabric of brand loyalists.

Then, linking it directly with "Web 2.0," Forrester nails its colors to the mast by drawing a very telling analogy to help people wrap their minds around the raw disruptiveness of Social Computing: "Web 2.0 is the building of the Interstate Highway System in the 1950s; Social Computing is everything that resulted next (for better or worse): suburban sprawl, energy dependency, efficient commerce, Americans' lust for cheap and easy travel."

Hinchcliffe reiterates this point, noting that one thing is clear, namely that the technologies of the modern Web are indeed reshaping our society, particularly of the younger generations that spend so much of their time there.

"The consequences could be dramatic," Hinchcliffe avers, "in the same way that the highway systems fundamentally disrupted the railroad industry."

Anyone wishing to explore further can click through on any of the below.

**Further Reading on Social Computing**

- Feedster: <http://www.feedster.com/search.php?q=%2522social+computing%2522>
- Digg: <http://www.digg.com/?s=social+computing&a=30>
- Technorati: <http://www.technorati.com/search/%22social%20computing%22v>

**Jeremy Geelan** is Conference Chair of the AJAXWorld Conference & Expo series and of the "Real-World Flex" One-Day Seminar series. From 2000-6, as first editorial director and then group publisher of SYS-CON Media, he was responsible for the development of all new titles and i-Technology portals for the firm, and regularly represented SYS-CON at conferences and trade shows, speaking to technology audiences both in North America and overseas. He remains executive producer and presenter of "Power Panels with Jeremy Geelan" on SYS-CON.TV, and is now developing new Conferences and One-Day Seminars for SYS-CON Media & Events.





## Bring your development plans to light

Sneak a peek at XMLSpy® 2006,  
and see how vital it is to master XML.

### Revealed in XMLSpy 2006 Release 3:

- Superior error messaging with dynamic hyperlinking
- New XSLT 2.0 and XQuery profilers
- Trace points for enhanced XSLT debugging
- Innovative restriction handling in XML Schema design

Altova® XMLSpy, the industry standard XML development environment, is indispensable for modeling, editing, transforming, and debugging XML-related technologies.

Illuminate your strategy with the world's leading XML editor, the original graphical schema designer, a code generator, file converters, debuggers, profilers, support for XSLT, XQuery, WSDL, SOAP, and a wealth of brilliant XML utilities and enlightened usability aides.

Become a markup mastermind!

Download XMLSpy® 2006  
today: [www.altova.com](http://www.altova.com)



# JDJ contents

## JDJ Cover Story

# FPO

by Anil Hemrajani

# 38

## Features

# 10

### Jakarta Struts & JavaServer Faces

by Herman Robinson

## EDITORIAL

### Unofficial History of Programming: '94-'06

by Yakov

..... 3

## VIEWPOINT

### Where Are the High-Level Open Source Design Tools?

by Ed Merks

..... 6

## MOBILE JAVA

### Integral Java: A Single Solution for Bypassing the Pitfalls of Split Stacks

*The future of mobile java*

..... 8

## DMA

### Detecting J2EE Problems Before They Happen

*Derived Model Analysis*

by Alan West & Gordon Cruickshank

..... 12

## SDO

### Data Access Services

*How to access relational data in terms of*

*Service Data Objects*

by Kevin Williams & Brent Daniel

..... 32

## DESKTOP JAVA VIEWPOINT

### The Death of Mediocrity

by Joe Winchester

..... 48

## IDE

### Managing a Standardized Build Process Outside of the Eclipse IDE

*Point-and-click solution won't cut it*

by Steve Taylor

..... 50

## AWARDS

### JDJ Editors' Choice Awards

..... 54

## JSR WATCH

### The JCP Program: Beyond the 300 Mark

by Onno Kluyt

..... 58

## FEEDBACK

### Letters to the Editor

..... 60

# 28

### A Generic JMS Listener for Apache Axis 1.x

by Parameswaran Seshan

JDJ (ISSN#1087-6944) is published monthly (12 times a year) for \$69.99 by SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645. Periodicals postage rates are paid at Montvale, NJ 07645 and additional mailing offices. Postmaster: Send address changes to: JDJ, SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645.

# The Time Is Now for Adobe Flash Player 9

by Emmy Huang

If past Flash Player download numbers are any indication, Flash Player 9 could have the biggest, early adopter audience of all.

In June 2006, Adobe released Flash Player 9, the most powerful Flash runtime to date, which introduced ActionScript 3.0 (an important ECMAScript-compliant update to the ActionScript language), faster performance through an optimized virtual machine, and interconnectivity with Adobe Flex 2. The small download size of Flash Player 9 and strong features may be at the core of why developers choose to deploy Flash applications. Equally important, and not to be overlooked, are the statistics about how many computer users already have Flash Player installed, and how quickly users can download and install or upgrade to the newest version. As developers, you want to know that your content and applications can be immediately experienced by the widest audience.

Independent research company Millward Brown conducted a study in June 2006 concluding Flash Player is on 97.3% of desktops in mature markets, and Adobe's own download statistics indicate the current demand for Flash Player is strong, as it's installed over 5 million times a day (Source: Akamai download statistics). A lesser-known fact is that each version of Flash Player has historically taken about 12 months to reach 80% penetration (Source: NPD Group Research), demonstrating the consistency of the adoption curve for each release. The latest quarterly study takes into account broader and deeper worldwide statistics and provides a more in-depth account of worldwide Flash Player penetration than previous studies. The study surveys users in six countries on a quarterly basis: United States, Canada, United Kingdom, Germany, France, and Japan. It also includes mainland China and South Korea every other quarter. (The margin of error in the Millward Brown survey is +3% with a 95% confidence level.)

In a previous wave of this study conducted in April 2006 by independent research company NPD Group Research, Flash Player 8 was at 69% penetration six months after its release, a considerable jump in the numbers from Flash Player 5 and 6, which were at 53% penetration during the same point in their cycles. The June 2006 study indicates that Flash Player 8 reached 86% penetration, just nine months after release—further indication that the demand for the latest versions of Flash Player is growing year over year.

In short, these studies highlight that Flash Player is one of the most pervasive software platforms on the Web and they suggest that users are adopting the latest Flash Players more quickly than their predecessors.

The demand for Flash Player, as demonstrated by the download numbers, is an indicator of the widespread use of Flash for creating compelling

experiences on the Web. Increasingly, Websites like Google Video, YouTube, MySpace, and MTV are turning to Flash Player and Flash video as their deployment solution for presenting rich content to their users. One of the primary reasons behind the quick adoption of Flash Player across desktops is that millions of users are consuming content across popular sites like these.

Another reason for the ubiquity of Flash Player is the simplicity of the installation and upgrade process, along with its small download size. As a developer, you can continue to rely on the default browser install experience for ActiveX in Microsoft Internet Explorer and the Firefox plug-in finder service, which we started supporting with the Flash Player 8 release. Alternately, you can now seamlessly upgrade your Website visitors to the latest Player using Express Install, a Flash-based experience introduced in Flash Player 8. Use Express Install to design an in-context upgrade experience for your content so users never have to leave your site to get the latest Player. You can also gracefully handle user cancellation and avoid system restarts. Learn more about implementing Express Install in your Flash applications in this article: Experiencing Flash Player Express Install ([http://www.adobe.com/devnet/flash-player/articles/express\\_install.html](http://www.adobe.com/devnet/flash-player/articles/express_install.html)). I also highly recommend watching the short demo of the Express Install user experience in the piece. To find out more about Player detection, Player installation and Express Install, see the Flash Player Detection Kit ([http://www.adobe.com/products/flashplayer/download/detection\\_kit/](http://www.adobe.com/products/flashplayer/download/detection_kit/)). You can also check out SWFObject, another popular solution for detection and Express Install developed by a member of the community.

The strong penetration of Flash Player, the ease of install and upgrade experience for end users, and the new features in Flash Player 9 make it a top deployment choice for your Rich Internet Applications. To learn more about new features in Flash Player 9 and how to use them, see my article, Introducing Flash Player 9 ([http://www.adobe.com/devnet/logged\\_in/ehuang\\_flashplayer9.html](http://www.adobe.com/devnet/logged_in/ehuang_flashplayer9.html)), and visit the Flash Player Developer Center.

To learn more about the Millward Brown and NPD studies, as well as methodologies used, see the following resources:

- Results for the Millward Brown June 2006 study: [http://www.adobe.com/products/player\\_census/flashplayer/](http://www.adobe.com/products/player_census/flashplayer/)
- Methodology for the Millward Brown June 2006 study: [http://www.adobe.com/products/player\\_census/methodology/](http://www.adobe.com/products/player_census/methodology/)
- Flash Player penetration by version: [http://www.adobe.com/products/player\\_census/flashplayer/version\\_penetration.html](http://www.adobe.com/products/player_census/flashplayer/version_penetration.html) ☺



President and CEO:

Fuat Kircaali [fuat@sys-con.com](mailto:fuat@sys-con.com)

Group Publisher:

Jeremy Geelan [jeremy@sys-con.com](mailto:jeremy@sys-con.com)

## Advertising

Senior Vice President, Sales and Marketing:

Carmen Gonzalez [carmen@sys-con.com](mailto:carmen@sys-con.com)

Vice President, Sales and Marketing:

Miles Silverman [miles@sys-con.com](mailto:miles@sys-con.com)

Robyn Forma [robyn@sys-con.com](mailto:robyn@sys-con.com)

Advertising Sales Manager:

Megan Mussa [megan@sys-con.com](mailto:megan@sys-con.com)

Associate Sales Managers:

Kerry Mealia [kerry@sys-con.com](mailto:kerry@sys-con.com)

Lauren Orsi [lauren@sys-con.com](mailto:lauren@sys-con.com)

## Editorial

Executive Editor:

Nancy Valentine [nancy@sys-con.com](mailto:nancy@sys-con.com)

## Production

Lead Designer:

Louis F. Cuffari [louis@sys-con.com](mailto:louis@sys-con.com)

Art Director:

Alex Botero [alex@sys-con.com](mailto:alex@sys-con.com)

Associate Art Directors:

Abraham Addo [abraham@sys-con.com](mailto:abraham@sys-con.com)

Tami Lima [tami@sys-con.com](mailto:tami@sys-con.com)

## Web Services

Information Systems Consultant:

Robert Diamond [robert@sys-con.com](mailto:robert@sys-con.com)

Web Designers:

Stephen Kilmurray [stephen@sys-con.com](mailto:stephen@sys-con.com)

## Accounting

Financial Analyst:

Joan LaRose [joan@sys-con.com](mailto:joan@sys-con.com)

Accounts Payable:

Betty White [betty@sys-con.com](mailto:betty@sys-con.com)

Accounts Receivable:

Gail Naples [gailn@sys-con.com](mailto:gailn@sys-con.com)

## Customer Relations

Circulation Service Coordinator:

Edna Earle Russell [edna@sys-con.com](mailto:edna@sys-con.com)

JDJ Store Manager:

Brunilda Staropoli [bruni@sys-con.com](mailto:bruni@sys-con.com)



DESKTOP



CORE



ENTERPRISE



HOME

**Emmy Huang** is senior product manager for Flash Player. Her experience includes working in engineering and product management on a range of digital entertainment technologies at Sony Pictures Digital, Liberate Technologies, and Intel. Emmy earned her MBA from the UCLA Anderson School of Management and has a BS in Computer Science from the University of Illinois.



\_INFRASTRUCTURE LOG

\_DAY 15: This project is out of control. The development team's trying to write apps supporting a service oriented architecture...but it's taking FOREVER!

\_DAY 16: Gil has resorted to giving the team coffee IVs. Now they're on java while using JAVA. Oh, the irony.

\_DAY 18: I've found a better way: IBM Rational. It's a modular software development platform based on Eclipse that helps the team model, assemble, deploy and manage SOA projects. The whole process is simpler, faster and all our apps are flexible and reusable. :)

\_The team says it's nice to taste coffee again, but drinking it is sooo inefficient!



**Rational**

Download the IBM Software Architect Kit at:  
[IBM.COM/TAKEBACKCONTROL/FLEXIBLE](http://IBM.COM/TAKEBACKCONTROL/FLEXIBLE)



# Java Bookshelf

by Yakov Fain

**F**inding and buying Java books online is great...as long as you know what to look for. The thing is in many cases it's not obvious from the book title what the book about (I'll give you some examples below). So visiting a real bookstore can be a much better experience. I'm lucky to work right by a large bookstore, so I have the luxury of visiting this store a couple of times a week. This is how it goes. You slowly move your eyes along the bookshelf...Stop, let me open this one. No hurry. I believe in the chemistry between books and readers: either you like it or not. This very moment. Without even reading it. No rush. Do you know that books are not put on the shelves randomly? Books on hot topics and books by well-established publishers like O'Reilly are always put on the shelves at eye level. When some languages or tools are hot, books start their "career" movement up the store ladder, or rather up the shelf. Three-four years ago, Java books dominated, and then .NET started fighting for space. All of a sudden books on Ruby and AJAX popped up at the level of my belly. Two weeks ago, all Flash 8 books suddenly moved from the floor level to the top shelf...

But let's get back to Java. I'd like to tell you about four recently published Java books. I'll start with excellent books whose titles don't exactly reflect their contents.

***Beginning POJOS*** from APRESS.

This book is actually about server-side development with Java. It starts by defining the application to be built, and over the course of the book the author presents the tools and frameworks required for the sample application. After defining the task, he walks you through the architecture of this application, domain model, and use cases.

The next stop is development and build tools (Eclipse and Ant). Simple examples and best practice advice make the learning process enjoyable. Data persistence with Hibernate comes next. Only now do POJOs come on stage, and their mapping to the database tables are well explained. EJB3, the Spring framework, testing, everything is here. Finally, there's a good discussion about the role of continuous integration in a project's lifecycle. This is a great book for junior and intermediate Java programmers who care about programming in style and are looking for a good tutorial on Java application development with popular Open Source tools. I own many quality books from APRESS, which has become one of the best publishers of the computer books.

***Agile Java Development with Spring, Hibernate, and Eclipse*** from Sams. This is yet another great book ... if you don't pay too much attention to the title. This book is not a Spring or Hibernate tutorial. It presents an excellent overview of the development process in an enterprise Java shop. The author is an experienced practitioner and this book is a jewel for any Java architect or development manager. You'll learn how to set up the environment, gather business requirements, and build the project deliverables an agile way. The author explains where Spring, Hibernate, build and test tools fit by going through the process of developing a sample Java application. Here and there he sprinkles concise Java or XML code samples that are short enough to not get you carried away with details, but at the same time they help you put all pieces of a Java Enterprise Application puzzle in the right places.

Your best bet is to buy these two books together. What a duo!

***Swing Hacks*** from O'Reilly. O'Reilly occupies the premium bookshelf space for a good reason. It's the best publisher of software books. If you're a Swing developer, stop reading this article and immediately order this book. Amazon sells it for under \$19 and at that price it's a steal. It contains 100 hacks, working code samples on various subjects of GUI programming with Swing. You'll find examples of working with tables and trees, file choosers, frames, animated windows, drag-and-drop, various tricks with text and fonts, and work with audio. This book isn't a tutorial, just keep it on your desk and use it as needed.

***Java Concurrency in Practice*** from Addison-Wesley. Let me just give you the names of some apprentices that the author of this book has invited: Joshua Bloch and Doug Lea. Should I even continue? Brian Goetz is the lead author. I bought this book online just because of the names. When it arrived, I was just amazed. The amount of the information per square inch of paper is extremely high. Talk about advanced stuff! Java is a great programming language, if you know how to use it. People who want to squeeze the best possible performance out of a JVM should buy and study this book. All new features of the `java.util.concurrent` package introduced in Java 5 are presented with reasonably simple examples. If you spent some time swimming in the Java Ocean, this manuscript will take you to the next level.

In my opinion, these books stand out from the crowd. You also have a chance to state your preferences by including your favorite books in JDJ's annual readers choice awards at <http://java.sys-con.com/general/2006rc.htm>. ☺

**Yakov Fain** is a senior IT architect consulting Wall Street companies. He's authored several Java books, dozens of technical articles and his blog is hugely popular. SYS-CON Books will be releasing his latest book, "Rich Internet Applications with Adobe Flex and Java: Secrets of the Masters" this Fall. Sun Microsystems has nominated and awarded Yakov with the title Java Champion. He leads the Princeton Java Users Group. Yakov teaches Java and Flex 2 at New York University. [yfain@sys-con.com](mailto:yfain@sys-con.com)

Innovations by InterSystems

# Java Developers Have Caché



The Object Database  
With Jalapeño.  
Give Your POJOs  
More Mojo.

InterSystems  
**CACHÉ**<sup>®</sup>

The object database that runs SQL faster than relational databases now comes with InterSystems Jalapeño™ technology that eliminates mapping. Download a free, fully functional, non-expiring copy at:  
[www.InterSystems.com/Jalapeno1P](http://www.InterSystems.com/Jalapeno1P)

# Barbarians at the Gate

*A report from the frontier where Open Source and Java meet*

by Andrew Cowie

**O**pen Source means different things to different people. For some it's a business model. For others it's a way of collaborating. Some see it as a way of reducing costs. And some are out to change the world.

That may sound a touch dramatic, but it's a pretty accurate description of the landscape. So let's look at each of these and see if we can't point out something interesting and perhaps unexpected about how Java and Open Source are interacting.

## Open Source as Collaboration

Although people gripe that Sun's Java implementation isn't Open Source (not yet, anyway), huge amounts of work have been done on software that supports the construction of sophisticated applications and platforms that is Open Source, especially under the auspices of the Apache Foundation. Having started as a simple Java servlet runner called Tomcat, the Jakarta project has blossomed into a diverse group of frameworks for common tasks, Web development, and enterprise applications.

Apache isn't the only one. One of the amazing things that even most Linux and Open Source advocates don't realize is just how much Open Source activity there is around Java. A huge percentage of the active projects listed at FreshMeat or hosted on SourceForge are written to run in Java Virtual Machines. You see the same level of collaboration in other dedicated communities including Codehaus and ObjectWeb, and, of course, the freight train known as the Eclipse Foundation.

When most people think about the products coming out of the Eclipse community, they think about the Eclipse Integrated Development Environment used to write and debug Java code. That's not wrong per se, but it overlooks the fact that Eclipse was designed as a platform to build applications on. Several major vendors have

recently decided to base their products on top of the Eclipse Rich Client Platform. The next version of Lotus Notes will be built on this technology. Of course this will be transparent to corporate users, but great news for developers: as ever with Open Source, the work that Notes engineers put into improving the Eclipse platform for their own needs will benefit everyone who uses Eclipse technologies.

All of these are examples of the essence of the Open Source movement: collaboration. But why do people do it?

## Open Source Reduces Costs

One of the common arguments advanced when explaining why organizations contribute to Open Source is to reduce development costs. Part of what any company does adds value – be this in terms of proprietary processes, specialized knowledge, or other unique abilities that give them a competitive advantage. At the same time, much of what has to be done to reach a point where they can provide that value is ground they share with others in the industry, and that is where collaboration, even with competitors, makes sense. We frequently see this in infrastructure – do you really need to create your own Java logging framework? No, of course not. There are already several excellent ones available (even over and above the Open Source-originated log4j that Java itself absorbed). On the other hand, if you've gone through the trouble of creating a library to do some complex yet common tasks that others have likely grunt through as well, does it make sense to maintain that library yourself, or to work with others so you can take advantage of improvements and fixes that they might make, quite frequently enabling new functionality you wouldn't have been able to afford? Really, each of these examples is nothing more than a joint venture, which is certainly a common enough structure

in the business world. The next time you're arguing in favor of using an Open Source solution, you might bring this up.

And it's not just development cost. Not to be sneezed at is the fundamental premise that access to the code makes maintenance easier – and in many cases is the only thing that makes it possible. Being able to see what's going on in an application or platform is often absolutely necessary to diagnose and troubleshoot problems. This is applicable up and down the IT stack, but is particularly relevant for Java developers trying to squeeze every last drop of performance out of a heavily loaded platform. If you're running an enterprise application server, the combination of JBoss and Hibernate on top of one of the Open Source databases is hard to beat. Being able to deploy on commodity hardware and leverage non-traditional architectures has led to a price/performance ratio that's hard to beat.

Just as important are infrastructure projects. It's easy to forget that tools that are fundamental to our daily work like JUnit and Ant are Open Source – and this is a case where the price is right indeed. Further up the stack are more comprehensive tools for a range of tasks – projects like Maven (a comprehensive build and deployment tool), continuous integration systems like CruiseControl and BeetleJuice, which help automate the task of building and testing large software projects, and systems that bring knowledge together like Cargo.

That last one bears more detailed mention. One of my clients has me working on revamping the infrastructure they use to build their products and run functional tests across them. They're a Java shop, and so it's no surprise that their product, a rather large Web application, is built on Java servlets and JSP; and since they target a wide range of enterprise customers they need to test their app on as many

**Andrew Cowie** is a long-time Unix and Linux user and advocate, but somewhat unusually was an infantry officer in the Canadian army, having graduated from the Royal Military College with a degree in engineering physics. He saw service across North America and did a peacekeeping tour in Bosnia. Based in Sydney, Andrew runs Operational Dynamics, a consultancy helping clients worldwide with crisis management. On the technical side, Andrew has extensive experience as a Unix/Linux sys admin, Java developer, and has long been an Open Source advocate.  
andrew@operationaldynamics.com



application server containers as possible.

Not terribly unusual, but when you're trying to run automated tests, it gets tricky. Although in theory one should be able to interchangeably use different app servers, we all know that the different vendors (Open Source and commercial) who have implemented the servlet, JSP, and J2EE specs have all done it differently. Even assuming the application you're testing doesn't use vendor-specific extensions, you still have to deal with the problem of setting up, starting, and stopping the app server containers themselves. And as you'd expect, each app server has a significantly different way of being configured and run.

That's where Cargo, a project hosted by Codehaus, comes in. Working together, groups from different environments have concentrated knowledge of how to configure, start, and stop a wide range of different containers into a simple API that you can use from within a Java program or add to your Ant or Maven scripts. And this dramatically accelerates your testing.

Infrastructure, tools, automation, and testing. All of these taken together have reduced barriers to cross-platform development enabling Windows, Unix, and Linux and Mac developers to work together, helping broaden audience and use while reducing cost and complexity.

## Open Source as Business Model

While some companies have leveraged Open Source as a way to reduce the cost of doing business, others have chosen Open Source as the basis of their business.

Object relational mappers are a necessary evil if you're forced to deal with a legacy SQL database system, but many developers, especially those working on small mobile or disconnected systems, have neither the resources nor the inclination to deal with such enormous complexity and in any case can't afford the footprint of such a combination on their devices. Enter db4o.

Its Java native embedded persistence solution db4o is remarkably easy to use. Developers can be up and running in less than 10 minutes. And that's not 10 minutes of figuring out how to install the thing. That's a few brief lines of code and you've got a complete database that stores your data in the

object-oriented form in which you work with it.

```
ObjectContainer db;
db = Db4o.openFile("warehouse.db");
// your code here. Perhaps...
Inventory a = new FastMovingItem();
q.setQuantity(400);

db.set(q);
db.commit();
```

And persisted! Queries are just as easy,

```
Inventory proto = new Inventory();
proto.setName("fruit");

List results = db.query(proto);
```

One of the great things about db4o is that it's native Java. You just feed it Java objects and get Java objects back out again – no need to translate your data model to some third-party pseudo-representation or any requirement to write endless metadata in XML. It does a really straightforward job of just getting on with persisting Plain Old Java Objects. No mucking around with bytecode enhancers (like JDO) and certainly none of the self-mutilation that goes with EJB.

db4o has proven really easy to use. You do have to wrap your head around the notion that you don't have to worry about foreign keys anymore – when you want a reference to another bit of data you just use a plain old Java variable because variables and instance fields are references in Java. Neat. The most remarkable part, though, has been the degree to which it enables object-oriented data models. For so long we've been forced to hamstring our domain object design in ways that are largely flat so we can shoe-horn them into the legacy rows and columns of relational databases. With db4o, you can develop elaborate data models with complex object graphs and deep inheritance hierarchies and then persist that information without any fuss. Quite the productivity boost.

There have been some remarkable success stories with db4o. Intel is using it in its chip fabrication plants; it's embedded in the robots Bosch makes for food preparation and consumer goods packaging; Boeing is using it for a new multi-mission aircraft and BMW has built it into its latest cars. Customers can't say enough about the short start-up times, zero administration,

and low memory footprint as they enthuse about why they chose it for their products. Not bad for a database that comes in a 400KB jar file.

db4objects has recently released a replication tool called dRS that lets you exchange data not just with other db4o instances but with relational database systems as well. This makes it easy to design systems where a large enterprise data store contains information that workers need to access from disconnected devices. This is a common problem – a salesperson on the road needs customer data; someone working in a warehouse needs to collect and propagate inventory information; medical devices need to gather and process vast amounts of data from a patient. In all these cases, mobile devices have limited resources and have to exchange information with larger systems. db4o excels at this.

Christof Wittig, CEO of db4objects, explains his company's decision to release its software this way: "We live in a post-materialist world. People don't have long lead times to do everything themselves. It's only through collaboration that we can empower each other to succeed." Asked why they support Open Source, "That's easy," he said. "We owe everything to the community. From the outset, the user and developer community have helped shape and refine the product design and direction. We wouldn't have had that if we weren't Open Source. Even more important, our product is already easy to use; Open Source means that db4o is easy try and that in turn wins us customers."

db4objects, Inc. releases db4o under a dual-license model. Manufacturers can choose a commercial license if doing proprietary embedded work for a surprisingly modest price per unit. And if you want to evaluate its use, are doing in-house work, or want to use db4o in your own Open Source projects then you can use db4o freely under the GPL.

Obviously we've been talking Java here, but developers who have to do cross-platform development in heterogeneous environments will be interested to know that there's a .NET version of db4o available as well. The two are entirely binary-compatible; a number of its customers are running db4o as the storage back-end on Java application servers while writing their clients in little .NET programs.



Yet-another example of Open Source bringing communities together.

### Open Source to Change the World

For most people, Open Source means cost free. But a growing number of people have come to believe in the freedom that the Free Software movement preaches. Freedom to innovate. Freedom to modify and to help others with your work. It's the way the information age began.

Over the last few years an increasing amount of effort has been invested in what I'll call Free Java. As ever, the word free is a tad inadequate; the Open Source movement encourages you to realize that their essential message is freedom as in liberty (to modify a program, to enhance or reuse it as you see fit to redistribute) in addition to the notably useful property of being free as in price (though many of us gladly pay our distributor or vendor to package it all up and provide us support).

Sun's Java VM is, of course, not free or Open Source and until recently its license forbid redistributing Java binaries. So, not unexpectedly, the free software community has been creating an implementation of Java that does give you those freedoms.

The GNU Classpath project is working towards providing a fully compatible set of class libraries and supports a dizzying array of Java Virtual Machines – from research VMs like Kaffe and SableVM to the quaint little JamVM to the rocketing CACAO and JikesRVM and the truly radical GCJ project.

GCJ deserves special mention. Five or six years ago, hackers at Cygnus (now part of Red Hat) realized that one way they could look at Java would be to consider it a specifically defined subset of C++ (in the same sense that XML is a specifically defined subset of SGML). They reasoned that if they wrote a first-stage compiler that would take Java in the front-end and spit out the tree representation used by the gcc compiler internally, they could take advantage of the tremendous power of the whole GCC suite behind it to optimize that code and link it to binary executables that would run on any of the many platforms that GCC already supports.

And, ta-da, one of the consequences of the Free Java effort is that you can run Java programs on virtually every architecture out there. You are no longer limited to the few platforms that the major vendors create their run-times for – a significant factor for embedded developers. Concurrent efforts to create a tight but highly performing garbage collector have meant that you can now create efficient and optimized programs with small footprints that – gasp! – are written in Java.

Ahead-of-time compilation (AOT) is somewhat unusual for us to think about – we're quite used to the just-in-time compilation (JIT) that has been part of commercial VMs since Java 1.1 days. JITs have a particular challenge though – they have to turn the intermediate representation (Java bytecode from a .class file) into native machine code fast enough that the user doesn't notice. That's a tall order, yet it's amazing how well modern VMs do it. There is, nevertheless, a limit to how much optimization a JIT compiler can achieve because of the limited time it has available. There's also a question of how much computing power is taken to do JIT compilation – horsepower that may not be available, especially on resource-limited machines like small embedded devices. By doing the compilation of native machine code ahead of time, GCJ isn't constrained by having to provide near-instantaneous compilation and has the luxury of being able to try more in-depth optimizations. The result is fast code and even better, there's no start-up penalty when a Java program is launched. Instead of the overhead of instantiating a massive VM and struggling through the JIT compilation of an enormous number of core classes, the code is already native and can immediately start executing – brilliant for small processes and desktop applications.

Red Hat, in particular, has invested significant effort into improving GCJ and the Classpath libraries to achieve this outcome on both small systems and enterprise servers. This has had impressive results: major projects like Eclipse and JONAS can be built native

resulting in a considerable performance boost. And if you're running the Fedora Core Linux distribution then any time you install a Java library it will be compiled native and transparently used to speed up your programs.

And there's even more to report from Free Java land. Recently a new project being incubated by the Apache Foundation has emerged on the scene. The effort, called Harmony, also aims to develop a fully compatible Open Source class library and VM implementation. It doesn't have much of a community behind it yet but it's been getting impressive donations of code from several major players in the Java space.

So across-the-board activity is up. Amazingly, some lament the bevy of choices maturing in the Java world, complaining that they have to invest effort into figuring out which solution is best for them. Ignoring for the moment that it has always taken hard work to figure out what course is best to take on an IT project, Java developers should be glad of all the competition. As all these projects jostle together, collaborating where they have common interests and competing where they think they can do better than the others, better platforms emerge, and that benefits all of us who write Java programs.

### Conclusion

All is not quiet on the Open Source front. From simple collaboration to unexpected innovation on platforms tiny or huge, Java stakes out a vibrant place in the Open Source pantheon.

Of course, we can't end a discussion about Open Source and Java without briefly touching on the strong signals coming out of Sun Microsystems that it will in the not-too-distant future open its Java implementation. It remains to be seen whether Sun will choose a license that will actually qualify under the Open Source Definition, but early indications are that it's listened to people from the FOSS community and know it has to go all the way.

And then watch in amazement at the hordes who'll be using Open Source. Keep watch at the gates! ☛

top **MISCONCEPTIONS** that drive  
Meet the most misunderstood developer team in the world.  
our Crystal Reports dev team crazy



**Crystal Reports® is too expensive.** Actually, the developer edition is just \$595<sup>1</sup> USD (or upgrade for only \$315<sup>1</sup>). Complimentary Crystal Assist support<sup>2</sup> provided with purchase.

**Crystal Reports doesn't include a free runtime license.** Not true, the developer edition includes a free runtime license<sup>3</sup> for each component engine.

**Getting reports on the web is complex.** False, the developer edition includes crystalreports.com<sup>4</sup> and Crystal Reports Server<sup>5</sup> to speed and simplify web reporting deployments.

**Crystal Reports only works in Windows®.** Not quite, whether you need to create or deploy reports on Windows, Linux or Unix, we have a Crystal Reports technology for you.

Find out more at: [www.businessobjects.com/devxi/misunderstood](http://www.businessobjects.com/devxi/misunderstood)

**Business Objects™**

1 Suggested retail price. 2 Complimentary access to support engineers and self-help. 3 Includes an unlimited runtime license for internal use of .NET, Java, and COM engines. 4 Includes ten named user licenses. 5 Includes five named user licenses. The Business Objects logo and Crystal Reports are trademarks or registered trademarks of Business Objects in the United States and/or other countries. All other names or products referenced herein may be the trademarks of their respective owners. © 2006 Business Objects. All rights reserved.



# Hello Dali!

*An introduction to the Eclipse Dali Java Persistence API tools project*

by Shaun Smith

**O**n June 26, 2006 the Eclipse Foundation announced the availability of new releases of 10 Open Source projects. This simultaneous release event, named Callisto, garnered a lot of attention for the 10 projects involved. But, meanwhile, on the same day and without much fanfare, not even a press release, the Dali JPA Tools project shipped its first formal release numbered 0.5. With the release of Dali 0.5, developers now have a solid set of tools for developing applications for the new Java Persistence API (JPA) in Eclipse.

## The Java Persistence API

The Java Persistence API is part of the new Java EE 5 EJB 3.0 specification and defines a vendor-neutral standard for object-relational mapping. But don't be fooled by the term "EJB." The JPA specification was certainly developed under the umbrella EJB 3.0 specification, but that doesn't mean it's just for Java EE. JPA is designed to work in Java SE as well as EE, and will likely be split off into its own specification in the future.

JPA defines a way to map plain old Java objects (POJOs), not Entity Beans, to relational databases. This means you can use JPA to store the Java objects you write without having to subclass a JPA-provided class or implement any JPA interfaces. One of the driving goals of the JPA specification was ease of use and it shows.

## JPA in Eclipse

One of the most striking features of JPA is the use of Java 5 annotations to define object-relational mappings. By adding annotations to your classes you can make instances persistent. JPA uses the term "Entity" for persistent objects and uses the @Entity annotation to identify them. This means that

you can use a simple text editor or the Eclipse Java editor to work with JPA.

Unfortunately the Java editor doesn't understand what the annotations mean. As far as it's concerned annotations are just metadata markup. It can validate the syntax but not the semantics. For example, in Figure 2 the Phone Entity's number field is mapped to a column named "NUM." That column may or may not exist in the database but without JPA-aware validation you won't find out until runtime — a very bad time to find out. This is essentially what Dali provides: JPA-aware tooling and validation to ensure that what developers build at design time will run at deployment time.

## Dali Overview

Dali provides tools to develop JPA applications targeted at either Java SE or Java EE and supports top-down, bottom-up, and meet-in-the-middle development approaches. Regardless of whether you want to persist an existing Java object model, manipulate data in an existing database, or connect your existing Java classes with an existing database, Dali can improve your productivity and help ensure that you don't spend your time in an endless edit, deploy, run, debug cycle.

For example, Figure 3 shows the same Phone Entity as Figure 2. But when using Dali, a problem is found in the JPA mapping for the number field. Dali has validated the column name specified in the @Column annotation against the Phone table and found that there's no such column.

## JPA Defaults

One of the most useful features of the JPA is its defaulting rules. For example, if an Entity is not explicitly mapped to a table then the table name defaults to that of the Entity. Default-

ing rules let developers "program by exception." That is, they only need to add annotations for things that don't match the defaults. In the case of our Phone example, Dali has confirmed that a table exists in the database with the name "Phone" — the same name as the Entity. Since there's no problem, no errors are displayed.

## Dali Views

Dali contributes two views to the Eclipse user interface along with a perspective that defines a layout suitable for performing object-relational mapping. Those two views are the Persistence Outline and Persistence Properties.

## Persistence Outline

The Persistence Outline view is similar to the Eclipse Java Outline but offers a JPA view of your object. In Figure 4 the Persistence Outline shows the Phone Entity and its mapped attributes. In JPA you can either put your mapping annotations on a Class's fields or properties (JavaBean style getters). The Persistence Outline displays the mappings the same way regardless of which of the two approaches you choose. Using the outline you can get a quick thumbnail sketch of the mappings for an Entity, even if those mappings are spread throughout the Java source file. For Phone you can see the id holds the primary key of the Entity and is a Basic mapping — mapped directly to the database column. The number attribute is also a Basic mapping while the custs attribute is a collection of objects mapped as a ManyToMany.

By default, the Persistence Outline selection is linked with the Java editor so you can navigate quickly around a Class to individual mappings. The linking is reciprocal — selection of attributes in the Java editor will also

Shaun Smith is co-lead of the Dali JPA Tools project and a principal product manager at Oracle for TopLink, the basis for the open source TopLink Essentials JPA reference implementation.

update the selection in the Persistence Outline. This quick navigation to mappings is useful if you want to jump to them in the Java source editor, but is more useful when paired with the Persistence Properties view.

### Persistence Properties

The Persistence Outline gives you a brief summary of your mappings and lets you navigate between them, but doesn't offer any help in editing your mappings. That's the function of the second view contributed by Dali: the Persistence Properties view. In Figure 3, we saw how Dali validated mappings and put error markers in the Java editor and errors into the problems view. But as the saying goes, acknowledging you have a problem is just the first step. The Persistence Properties provides tools for understanding and resolving mapping problems.

The Persistence Properties view performs a couple of very useful functions. It shows how a mapping is configured and, perhaps even more importantly, it shows the defaults that will be applied by a JPA runtime when the Entity is deployed.

For example, in Figure 5 the column mapping for the number attribute is defaulting to True for insertable and True for updatable. Defaulted values are clearly visible through the use of the word "Default." Notice that the column name isn't marked as a default value because the developer has explicitly specified it in an annotation.

But let's return to the problem Dali identified with the number attribute — there's no such column as NUM in the Phone table. A valid column name has to be selected, and the Persistence Properties view can help. It offers valid options for all mapping settings including settings that require access to the data model.

In Figure 6 the column name dropdown contains all the Phone table columns. It also displays what the default column name would be if nothing were specified. Since the default is correct, the default may as well be used. With the entire mapping using default values Dali removes the mapping annotation from the Java source to keep it uncluttered by unnecessary annotations.

Figure 7 and Figure 8 show that with no column specified, in fact no mapping specified at all, the defaults

```
@Entity
public class Customer {
    @Id
    private int id;

    @ManyToMany
    private Collection<Phone> phones;
```

Figure 1 When using Java 5, the Eclipse Java editor is aware of annotations and will perform code completion on annotation and property names

```
@Entity
public class Phone {
    @Id
    private int id;

    @Column(name="NUM")
    private String number;
```

Figure 2 The @Column overrides the default mapping of the 'number' field to the 'NUMBER' column

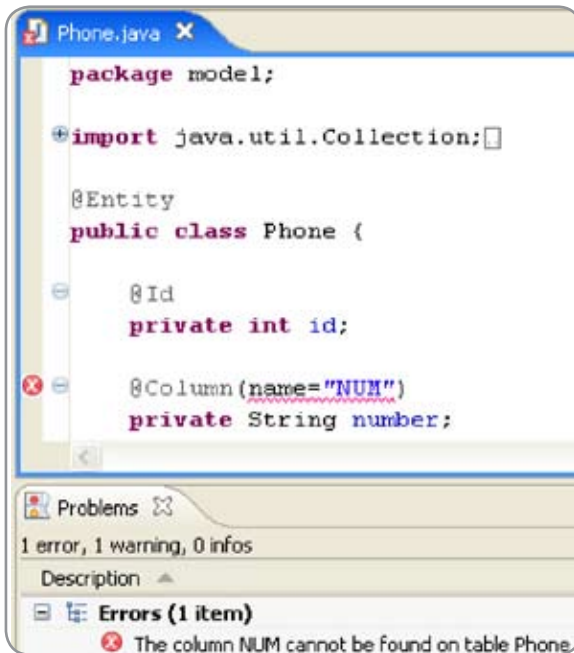


Figure 3 Dali validates JPA mappings against the data model. In this case there's no column named NUM on the Phone table

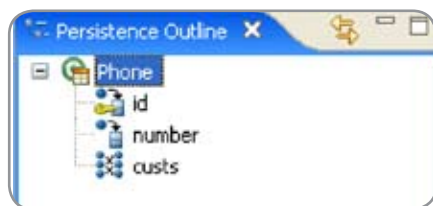


Figure 4 The Dali Persistence Outline View



Figure 5 The Persistence Properties View displays the settings for mappings explicitly defined by the developer and the defaults that are implicitly applied by JPA runtimes

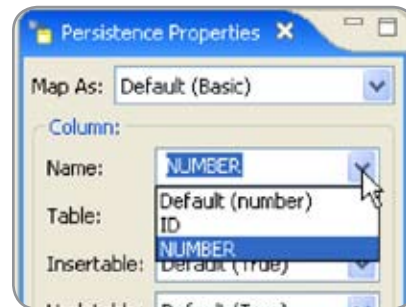


Figure 6 The Persistence Properties View provides access to the data model for mapping

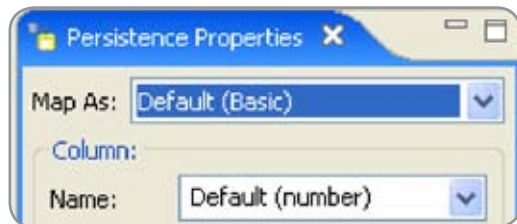


Figure 7 The Persistence Properties View showing the default Basic mapping of the number attribute and the default column name

validate against the data model and there are no problems.

### Top Down and Bottom Up

We've seen how Dali can help with the "meet-in-the-middle" approach of mapping an object model to an existing database but two other approaches are supported. Using Dali, it's possible to start with a set of Java classes annotated as Entities and generate the database tables they map to. Generation of Entities from tables is also supported. The al-

gorithm Dali uses for both Entity and table generation is defined by the JPA default mapping rules with a few extra heuristics to deal with differences in Java/database naming styles like underscores versus camel case.

Dali's support for generation offers a quick way to bootstrap a new JPA application. You can generate Entities or tables to get a starter configuration and then refactor either knowing that Dali will flag any breakage in your mappings with problem markers.

### Deployment

Deploying a JPA application is straightforward whether you're using Java SE or EE. Dali doesn't offer any specific packaging and deployment support beyond some assistance with maintaining the persistence.xml file (more on this below), however, deploying JPA Entities is just like deploying POJO applications. You can jar them up using the standard Eclipse support for exporting jars or include them in an Enterprise Archive (EAR) as a utility jar using the Web Tools Platform (WTP).

### Persistence.xml

The one XML configuration file required in the JPA specification is the persistence.xml file. This file defines important runtime settings including database connection information and transaction type. When you add persistence support to a Java project, Dali creates a basic persistence.xml file and places in the src\META-INF folder. Typically you'll hand-edit this XML file to reflect your deployment configuration.

As mentioned, JPA applications can be deployed to both Java SE and EE environments. However, when running outside an EJB 3.0 container, JPA requires an additional piece of information in the persistence.xml: a list of all the persistent Entities. In the 0.5 release Dali provides support for keeping the persistence.xml in sync with your defined Entities.

Right-clicking on the persistence.xml file in the Package Explorer and selecting Java Persistence>Synchronize Classes will update its list of classes (Figure 11).

### Future Directions

The focus of the Dali 0.5 release was annotation-based mapping and support for the core JPA mapping types. Dali 1.0 will offer editing and validation support for both annotation and XML-based mapping as well as the use of XML mappings to override annotations as defined in the JPA specification.

Smother integration of the Dali tools with WTP is also a high priority for 1.0. The Dali project is now incubating inside WTP as one of the new Java EE 5 technologies that will be incorporated into WTP 2.0.

In 1.0, Dali will also leverage the enhanced database support provided by the Data Tools Project (DTP). The combination of WTP with Dali and the DTP will provide a comprehensive toolset for the development of Java applications that rely on relational data.

### Getting Started

The best place to begin with Dali is to visit the project home page, check out the online demos, download the plug-ins, and go through the tutorial. The Dali newsgroup is monitored by the development team and is a great place to ask a question or get help.

And finally, like every Open Source Eclipse project, contributors are welcome. Contributors meet on the dali-dev@eclipse.org mailing list to discuss technical issues and make decisions.

### Resources

- Dali home page with links to downloads, documentation, project roadmap, and tutorials: <http://www.eclipse.org/dali>
- TopLink Essentials JPA Reference Implementation: <http://otn.oracle.com/jpa>
- JSR 220: Enterprise JavaBeans 3.0 specification: <http://www.jcp.org/en/jsr/detail?id=220>

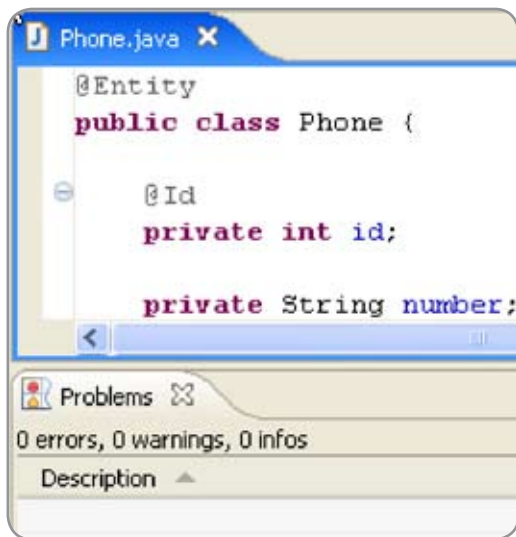


Figure 8 With no mapping annotation on the number attribute Dali validates the defaults as problem-free

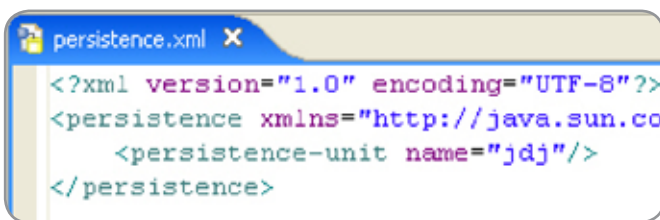


Figure 9 Basic persistence.xml created by Dali

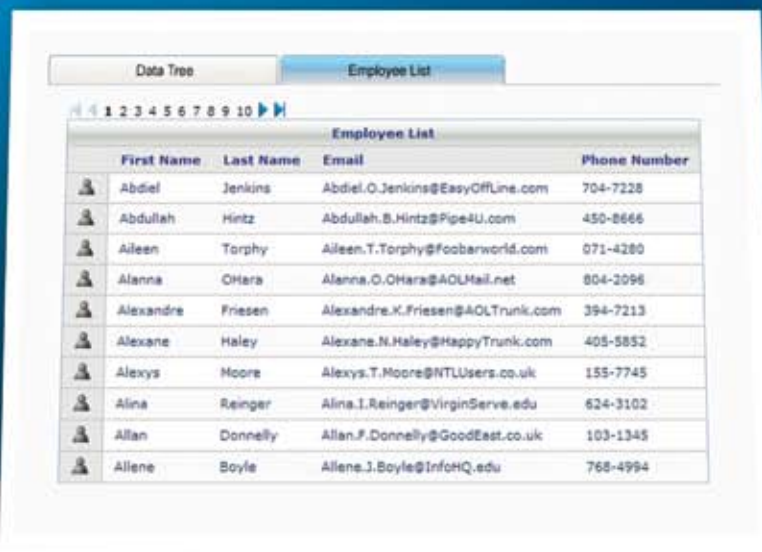


Figure 10 The list of Entities can be updated in the persistence.xml using the 'Synchronize Classes' menu option



# Speed. Simplicity. Style.

## Build a better User Interface with NetAdvantage



## NetAdvantage® for JSF

### 2006 Volume 1

AJAX-enabled JavaServer™ Faces components



**Speed** – Built to support large, data-driven applications

**Simplicity** – Intuitive features for end users; simple tag interface for developers

**Style** – Fully customizable look & feel

learn more: [infragistics.com/jsf](http://infragistics.com/jsf)

Infragistics Sales - 800 231 8588

Infragistics Europe Sales - +44 (0) 800 298 9055



WINDOWS FORMS

ASP.NET

WPF

JSF

grids

navigation

menus

listbars

trees

tabs

explorer bars

editors

Copyright © 2006 Infragistics, Inc. All Rights Reserved. Infragistics and the Infragistics logo are registered trademarks of Infragistics, Inc. All other trademarks are the property of their respective owners.

# Apache Trinidad – A World Cup Skinning Experience?

*A global style sheet that only has to be set in one place for the entire application*

by Jonas Jacobi & John Fallows

One of the 2006 Soccer World Cup highlights must surely be the Trinidad and Tobago versus Sweden game. The underdogs Trinidad and Tobago managed to push off the onslaught from the Swedish team. The game ended 0-0, which was for the people of Trinidad and Tobago a divine experience - their teams very first World Cup point!

So, you are, of course, asking yourself: What are these guys talking about? The question you should ask yourself is: Is Trinidad and Tobago going to be a success in the Java EE world as well? With the addition of project Trinidad to the Apache MyFaces community, the MyFaces project can now offer a rich and powerful component solution Trinidad and Tobago (and not to forget the original Tomahawk library). Project Tobago is a donation from Atanion GmbH (<http://www.atanion.com>) and, as a project, it recently graduated from the Apache Incubator.

Project Trinidad, the largest component library of the three, is a donation from Oracle. This donation is a subset of Oracle's well-known ADF Faces component library to the Apache Software Foundation (<http://incubator.apache.org/adffaces/>).

What is Apache Trinidad? Apache Trinidad is the most comprehensive free JSF component library on the planet. This library contains 12 helper objects such as converters and validators, and a staggering 93 components ranging from simple input components to complete page components with built-in menu model support. In addition, Apache Trinidad provides a set of extended services such as dialog framework and skinning framework.

This is the second article in a series of articles where we will be deep diving into various aspects of Oracle's

donation - Apache Trinidad - and give developers insight into its functionality and how to use and extend Apache Trinidad. In our first article we covered the HTML Ajax RenderKit provided with project Apache Trinidad (<http://jdj.sys-con.com/read/232061.htm>), and in this second article we are going to cover the skinning feature provided by this JSF component library.

## Project Trinidad's Skinning Framework

Project Trinidad lets you change the appearance, or look and feel, of an application without having to rewrite the code that implements the application's user interface. Trinidad currently provides two parent skins, Simple and Minimal, which you can extend to provide custom skins for your applications. The Minimal skin provides some formatting and the Simple skin contains almost no special formatting. By default, a custom skin inherits the appearance of its parent look and feel. When you wish to modify the appearance of a component, you simply provide custom style definitions and custom icons.

## About the Trinidad Skins

A skin in project Trinidad is a global style sheet that only needs to be set in one place for the entire application. Instead of having to style each component, or having to insert a style sheet on each page, you can create one skin for the entire application. Every component will automatically use the styles as described by the skin. Any changes to the skin will be picked up at runtime; no change to the code is needed. Skins are based on the Cascading Style Sheet specification, and are using CSS 3.0 syntax.

In addition to using a CSS file to determine the styles, skins also use a

resource bundle to determine the text. For example, the words "Previous" and "Next" in the navigation bar of the Project Trinidad's table component are determined using the skin's resource bundle. All the included skins use the same resource bundle. In this article, we are not going to cover the use of resource bundles.

## Creating a Custom Skin

You create a custom skin by extending the Simple skin and overriding the provided selectors. There are three different levels of selectors: Global, Button, and Component.

- Global Style selectors affect more than one component. If the selector name ends in the :alias pseudo-class, then the selector is most likely included in other component-specific selectors. Defining properties for a selector that ends in :alias will most likely affect the skin for more than one component
- Component-level selectors can be used to skin a particular project Trinidad component. The selectors defined below are specified by the component they affect. Let's say you want to skin the tr:chooseDate component. One of the selectors is af|chooseDate::title. The ::title pseudo-element indicates that this is the title portion of the tr:chooseDate component. If you want to skin the title of the tr:chooseDate component, you would set css properties on the af|chooseDate::title selector in your Project Trinidad skin .css file.

Note: Apache Trinidad is still undergoing incubation, so naming conventions such as af| will likely change in future builds of Apache Trinidad.

You may see selector names that end in :alias in the component-level section. These are meant to provide

**Jonas Jacobi** is a J2EE technology evangelist at Oracle. A native of Sweden, he has worked in the software industry for more than fifteen years. Prior to joining Oracle, Jonas worked at several major Swedish software companies in management, consulting, development, and project management roles. For the past three years, he has been responsible for the product management of JavaServer Faces, Oracle ADF Faces, and Oracle ADF Faces Rich Client in the Oracle JDeveloper team. [jonas.jacobi@oracle.com](mailto:jonas.jacobi@oracle.com)

**John Fallows**, former lead developer for Oracle ADF Faces Rich Client, has been working in distributed systems for over a decade. After five years spent focused on designing, developing the JavaServer Faces standard to provide AJAX functionality, playing a leading role in the Oracle ADF Faces team, he recently joined a start-up. Originally from Northern Ireland, John graduated from Cambridge University in the United Kingdom and has worked in the software industry for more than ten years. Prior to joining Oracle, he worked as a research scientist for British Telecommunications Plc. [john.r.fallows@gmail.com](mailto:john.r.fallows@gmail.com)

short-cuts to skin more than one component that shares a certain style or icon, or to skin more than one piece of a component. For example, the `.AFDefaultFont:alias` style defines skin properties that are shared by all `tr:outputXX` items. Therefore, if you change the `.AFDefaultFont:alias` style, you will affect all components sharing this style selector.

- Project Trinidad does not currently support component-level selectors for buttons. For example, you cannot customize a `goButton` differently from a `commandButton`. Skinning supports two very different button implementations. By default, standard browser buttons are used. However, the skinning also supports dynamic generation of image-based buttons. In order to enable image-based buttons, the following four button icons must be specified:

1. `.AFButtonStartIcon:alias`
2. `.AFButtonEndIcon:alias`
3. `.AFButtonTopBackgroundIcon:alias`
4. `.AFButtonBottomBackgroundIcon:alias`

When these four icons are specified, Project Trinidad combines the images specified by these icons into a single button image. (Note: These icons must be specified using either context-image or resource-image icons. Text-based icons are not allowed.)

### To Create a Custom Skin

In most cases you will probably not customize the skin of every component available in project Trinidad's component library (there are after all 84 UI components). By reviewing your application using, for example, the Simple skin, you can determine what components to customize.

Note: At the moment the application developer can only extend, and inherit from, the Simple skin, but there is progress in the Apache Trinidad community to improve this, so that a skin can inherit from any custom skin. A skin consists of the following artifacts:

- A CSS file that defines the actual look of the components
- A configuration file - `trinidad-skins.xml` - that lists all skins available for this application (not including Minimal and Simple). This file has to be located in your applications

WEB-INF directory

- An entry in the Trinidad/ADF Faces configuration file - `trinidad-config.xml`. This file should also be located in the WEB-INF directory.
- Any other resources need to create the actual look of the components - additional CSS files, Images etc...

### Modifying the Trinidad Skin CSS

We are going to use an application that mimics the Apache MyFaces Website, so that we can illustrate how the skinning feature works and compare with the original (see Figure 1).

The hard part of providing a skin is not the actual creation of the Trinidad artifacts; it is creating the actual graphics and styles to be used by the

application that is tedious. In this case we already have the above page as a foundation for the look and feel, which contains the styles and images needed. In the skin CSS file you can add any selectors that you wish to override, and set the properties as needed. You can set any properties as supported by the CSS specification. You can also create your own alias classes.

The application we have built is a standard JSF application using the Apache Trinidad's components with the Minimal skin (see Figure 2).

### Changing the Color Scheme of a Skin

The easiest part of changing the look of your application is to change the base classes of a skin to use other



Figure 1 The Apache MyFaces home page

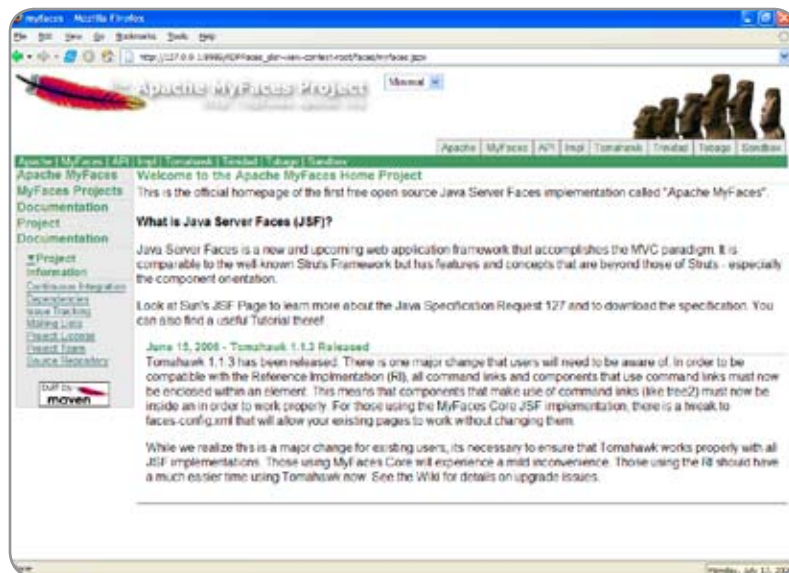


Figure 2 The sample application using Apache Trinidad's JSF components with the Minimal skin





Figure 3 The Simple skin applied to the sample application

colors. For example, the Simple skin looks Figure 3 without changes, and with some very minor changes to the following alias classes:

```
.AFDarkForeground:alias {color:#900000;};
.AFDarkBackground:alias {background-color:#333333;}
```

you can change the look and feel by changing the style classes controlling the base colors of the Simple skin (see Figure 4).

### Adding a Skin to a Apache Trinidad Component

The components that we are going to provide a skin for are - tr:panel-Header, tr:panelSideBar, tr:navigation-Pane, and tr:inputDate. We are going to start with the tr:inputDate component. At the moment the tr:inputDate component looks like this with the Minimal Look and Feel:



but we would like to change to look like this:



To able to do this we need to add the component selector for the tr:

inputDate component. By adding the following code to the skin CSS file (in this case we have named the file trinidadSkin.css), we can change the icon of the calendar launch button. Add the following af|selectInputDate style class code snippet to CSS file (note that in CSS the order of the style classes listed in the CSS file is important).

```
/** inputDate launch icon */
af|inputDate::launch-icon {
    content:url(/skins/trinidad/skin_images/timedate_ena.png);
    width: 16px;
    height: 16px;
}
af|inputDate::launch-icon:rtl {
    content:url(/skins/trinidad/skin_images/timedate_ena.png);
    width: 16px;
    height: 16px;
}
```

Notice that the Apache Trinidad skinning feature provides bi-directional support, causing the duplication of the style class name with an additional : rtl suffix to provide “right to left” style information in the second style class. It is also important that you add width and height to any style using images/ icons, since some browsers, such as Internet Explorer, might have problems displaying the actual icons. Next we are going to look at the Apache Trinidad’s

tr:navigationPane control. This component is slightly more complex than the tr:inputDate component with more controls and attributes that can be adjusted. The default look and feel for this control is shown in Figure 5, ...and we need to change it to Figure 6.

We are going to add Listing 1 just below the inputDate styles in our trinidadSkin.css file.

By using the alias class - AFTab-BarItem:alias - we can add a set of styles to the skin that are generic to all navigationPane controls. We have also added styles that are unique to all navigationPane link controls by using the component-specific selector.

We are not going to bore you with too much repetitive code, so to finish off the CSS bit of our new skin we are going to look at how you can apply a slightly different style to a nested component compared to how it would look outside its parent component. In this sample we have a panelHeader component that has the following skin setting:

```
/** panelHeader */
/** ----- */
af|panelHeader
{
    padding: 4px 4px 4px 6px;
    background-color: #DDDDDD;
    border: 1px solid #999999;
    vertical-align: middle;
}
af|panelHeader::level-one
{
    color: #900000;
    font-weight:bold;
    font-size: x-large;
}
af|panelHeader::level-two
{
    color: #333333;
    background-color: #EEEEEE;
    border: 1px solid #AAAAAA;
    font-weight:bold;
    font-size: large;
}
```

At runtime this will translate to H1, H2 equivalents, which will look like the Welcome note and the release announcement shown in Figure 7.

This is fine as long as you don’t use this panelHeader component anywhere

**RCP Developer™** is created by the experts who brought you the popular book *Eclipse: Building Commercial Quality Plugins* — Eric Clayberg & Dan Rubel



Java on the desktop is back!

# RCP Developer™



Instantiations RCP Developer harnesses the power of the Eclipse Rich Client Platform. Quickly create native desktop applications that are rich, cross-platform, and high-performance. The vast expertise of the Eclipse developer community and the experience of Instantiations lets you...

## STAND ON THE SHOULDERS OF GIANTS

### Create blazing fast rich Java clients.

Tap into the ultra high performance Eclipse graphical user interface libraries and ensure a native look and feel.

### Build rich Java client applications faster.

Focus on creating business specific functionality rather than reinventing the low-level logic required for applications to run.

### Deliver a more consistent user experience.

Tame the most demanding user interface requirements and reliably deliver a rich and consistent user experience across all applications.

### Fully exploit the Eclipse Rich Client Platform.

"The Eclipse Rich Client Platform is the leading framework for creating Java client applications. RCP Developer is the first product to bring comprehensive application construction, GUI testing and packaging of rich-client applications to Eclipse RCP."

—Mike Milinkovich, executive director of the Eclipse foundation



Download a risk-free trial copy:

[www.instantiations.com/rcpdeveloper](http://www.instantiations.com/rcpdeveloper)

### RCP Developer™

#### SWT Designer™



**Design.** Quickly create views, editors, perspectives, and preference pages for your applications. Intuitive visual design and access to the high performance SWT GUI framework of Eclipse RCP.

#### WindowTester™



**Test.** Automatically test the GUIs that comprise Eclipse RCP applications. Minimize the need to hand-code GUI test cases. Automated recording, test generation, assertion coding and playback facilities.

#### RCP Packager™



**Deploy.** Streamline build and deployment for Eclipse RCP applications. Assemble application elements into a single package. Quickly create and maintain high-quality Eclipse RCP installations.



[www.instantiations.com](http://www.instantiations.com) 1-800-808-3737

The leader in Eclipse RCP development tools



else, say nested within a panelSideBar component. In that case the styles will be too overwhelming, but there is a very simple standard CSS solution:

```
af|panelSideBar af|panelHeader
{
    font-size: smaller;
    border-left: 0px;
    border-right: 0px;
    border-top: 0px;
    border-bottom: 1px solid #AAAAAA;
    background-color: #F0F0F0;
    padding-top: 2px;
    padding-left: 9px;
}
```

```
color: #49635a;
}
```

In this code sample we have defined that any panelHeader within a panelSideBar should have a different style.

### Setup of a Custom Skin

Let's now have a look at how to set up your application to use an Apache Trinidad custom skin. First of all, we need the CSS file, stored somewhere at the root of our Web application. In our sample application, it's stored in the /skin/trinidad directory. We should also make sure that we have access to all resources needed for the skin, such

as images and other CSS files. For our application, these are stored in the /skin/trinidad/images directory.

Second, we need to make sure that our Apache Trinidad application is aware of the custom skin. This is done by adding a configuration file to the WEB-INF directory called trinidad-skins.xml (the name of the file is a leftover from the Oracle donation and will soon be renamed to comply with the Apache Trinidad's naming conventions). The content of our trinidad-skins.xml file looks like Listing 2.

### Register a Custom Skin

The <id> element in the trinidad-skins.xml can be used to reference a skin in an EL expression. For example, if you want to have different skins for different locals, you can create an EL expression that will select the correct skin based on its ID.

The <family> element configures an application to use a particular family of skins. This allows you to group skins together for an application, based on the render kit used.

The <render-kit-id> determines which render-kit to use for the skin. You can enter one of the following:

- org.apache.myfaces.trinidad.desktop: the skin will automatically be used when the application is rendered on a desktop.
- org.apache.myfaces.trinidad.pda: the skin will be used when rendered on a PDA.

The <style-sheet-name> element defines the path to the custom CSS file.

### Configuring an Application to Use a Custom Skin

When you have created a skin and are ready to use it, you need to make your application aware of it by defining which skin to use in the trinidad-config.xml file. You set an element in the trinidad-config.xml file that determines which skin to use, and, if necessary, under what conditions.

```
<?xml version="1.0" encoding="windows-1252"?>
< trinidad-config xmlns="http://myfaces.apache.org/trinidad">

<skin-family>#{sessionScope.skinFamily == null ? "minimal" : sessionScope.skinFamily
```



Figure 4 Sample application with some minor changes to the Simple skin



Figure 5 The new Trinidad skin applied to the sample application



```

}</skin-family>

<debug-output>true</debug-output>

</ trinidad-config>
If you only have one skin for your appli-
cation, you only need to replace the
<skin-family> value with the family name
for the skin(s) you wish to use.

<skin-family>trinidad</skin-family>

```

To conditionally set the skin-family value, you can enter an EL expression that can be evaluated to determine the skin to display. For example, if you want to use the German skin if the user's browser is set to the German locale, and use the English skin otherwise, you could have the following entry in the adf-faces-config.xml file.

```

<skin-family>#{facesContext.viewRoot.
locale.language=='de' ? 'german' : 'eng-
lish'}</skin-family>

```

In our sample application, we are going to use a selectOneChoice component to switch skins at runtime. For this we need to define the following in the trinidad-config.xml file with:

```

<skin-family>#{sessionScope.skinFamily ==
null ? "minimal" : sessionScope.skinFamily
}</skin-family>

```

The actual component that will perform the actual switching at runtime looks like this:

```

<tr:selectOneChoice label="Select Skin"
value="#{sessionScope.
skinFamily}"
onchange="form.sub-
mit();">
<tr:selectItem label="Simple"
value="simple"/>
<tr:selectItem label="Minimal"
value="minimal"/>
<tr:selectItem label="Trinidad"
value="trinidad"/>
<tr:selectItem label="MyCompany"
value="mycompany"/>
</tr:selectOneChoice>

```

The onchange event handler will perform a form POST whenever a skin is selected in the selectOneChoice component. Alternatively, you can add

a commandButton to the page that will re-submit the page. Every time there is a POST, the EL expression will be evaluated, and if there is a new value, redraw the page with the new skin. Figure 8 shows the completed page with the new Trinidad skin applied (at runtime.)

## Summary

Creating Apache Trinidad skins are easy and using them even easier. An application developer can set a skin

based on any criteria using EL expression and JSF backing beans (not shown in this article). This allows application developers to have different skins per user, page, application, and so on, without impacting the actual application logic! We should also take a note that Apache Trinidad is still in incubation as an Apache podling, thus skinning artifacts, such as style selectors discussed in this article, might change in future builds. ☺

### Listing 1

```

/** menuTabs **/
/** You can create borders, and have image-free tabs, or you can use the icon keys (e.g.,
af|menuTabs:selected-start-icon) to create tabs with more elaborate borders. An example
of this is provided at the end of the sample. */

/** This isn't needed when you use icons for the tabs */
.AFTabBarItem:alias
{
border-style: solid;
border-width: 1px 0px 0px 1px;
border-left-color: #FFFFFF;
border-top-color: #FFFFFF;
padding: 2px 6px;
background-color: #DDDDDD;
line-height:100%;
}

/* Remove the text decoration from all tabBar links */
.AFTabBarLink:alias
{
text-decoration:none;
}
/* Make the selected tab bold */
af|navigationPane::tabs-active
{
font-weight: bold;
font-size: 11px;
color: #900000;
}
af|navigationPane::tabs-inactive
{
font-weight: bold;
font-size: 11px;
color: #900000;
}

.MyLinkHoverColor:alias { color: #003300; }

af|navigationPane::tabs-inactive:hover
{
-ora-rule-ref:selector(".MyLinkHoverColor:alias");
}

/** This is not used in the Trinidad skin, but menuTabs that use icons are define like
the following */
/* af|navigationPaneTabs::tabs-start
{
content:url (/skins/trinidad/skin_images/menuTabsEnabledStart.png);
}
af|navigationPane::tabs-start:rtl
{
content:url (/skins/trinidad/skin_images/menuTabsEnabledEnd.png);
}

```

### Listing 2

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<skins xmlns="http://myfaces.apache.org/trinidad/skin ">
<skin>
<id>mycompany.desktop</id>
<family>mycompany</family>
<render-kit-id>
org.apache.myfaces.trinidad.desktop
</render-kit-id>
<style-sheet-name>
skins/mycompany/myCompanySkin.css
</style-sheet-name>
</skin>
<skin>
<id>trinidad.desktop</id>
<family>trinidad</family>
<render-kit-id>
org.apache.myfaces.trinidad.desktop
</render-kit-id>
<style-sheet-name>
skins/trinidad/trinidadSkin.css
</style-sheet-name>
</skin>
</skins>

```

# Open Source Software, Standards, and Java

by Stephen R. Walli

*What will it mean?*

**S**un Microsystems recently announced its intentions of finally publishing Java under an Open Source license. But what does that actually mean? We'll take a quick look at what it means to be "Open Source," how the Java language specification compares to other more formal language standards, and the importance of the brand and certification programs. We'll then look at what benefits Sun may get from distributing Java as Open Source and at some of the problems that will have to be addressed.

## Open Source Software

The Open Source Initiative defines Open Source software and a license must meet 10 criteria to be considered an "Open Source software" license. Essentially, it's a way of thinking about licensing software. It boils down to some very simple ideas about access to source code and the ability to modify the software and distribute those modifications. It encompasses the concept of Free Software as defined by the Free Software Foundation around a set of software "freedoms," such as the freedom to study how a program works and adapt it to your own needs.

Open Source software is typically developed in a collaborative community, either under a strong leader who coordinates the development community, or a meritocratic process where a developer earns the leadership role in the community like the process favored by the Apache community. Some companies build businesses based on Open Source software projects, generally ones they control. For example, MySQL (the company) maintains MySQL (the database engine). In these cases, the software has a copyright, is owned, and is therefore

licensable. Free and Open Source software is not "public domain" in any sense of the phrase.

Software developed in successful Open Source collaborative communities shows all the hallmarks of well-developed software from other processes. Essentially good software is developed by good software developers regardless of the licensing strategy. So Open Source software has just as much potential to be well-structured, have well-defined stable interfaces, and be delivered through a disciplined process that encompasses software inspection, mandatory version control, and automated building and testing as software developed in other ways. Where Open Source differs from other well-developed software is in the collaborative community. The best developers interested in the software can participate in its creation and evolution regardless of where they live or work. This provides a number of benefits:

- Many people see the source code. Software inspections regardless of how informal prove to be much more effective at finding bugs than testing.
- The code is used and tested in a broad base. This expansion of the "test" bed tends to shake out bugs faster and hardens the software.
- A diversity of expertise and experience can be leveraged. This applies both to improving the code base, as well as to innovating on the code base to take it in new and interesting directions.

Sun has been an active participant in the Open Source software world since its inception. The original SunOS operating system was a Berkeley Unix derivative that came out of the

collaborative efforts around Unix in the early days. Sun has contributed heavily to the accessibility features in the Gnome desktop. Most recently it has opened the Solaris source code base under its own OSI-approved Open Source license (the Common Development and Distribution License or CDDL) and has been developing the OpenSolaris community. So Sun definitely has experience with Open Source, both contributing to other and developing its own communities.

Let's shift gears for a moment and take a look at Java and the Java Community Process from a standards perspective, as that has been Java's history to date.

## Standards (Open and Otherwise)

A specification is simply a document describing some interface for interoperability. Lots of companies publish specifications to enable customers and partners to interoperate software with their products better. In such cases, where the specification is published by a single commercial entity, it typically benefits the company by encouraging add-ons to its product.

A standard is a specification that has been put through some form of consensus process by a collection of interested parties. It may be a formal government-supported de jure process with checks and balances to ensure that the consensus isn't anticompetitive collusion. It may be an industry or trade organization (CBEMA, ECMA, IEEE) with a broad interest in an area, e.g., computing standards. It may be an industry group with a narrower focus (e.g., OASIS, W3C, IETF). The consensus process has rules that define such things as participation, acceptance, interpretation, amendment, and withdrawal.

The economic purpose of a standard is to encourage and enable multiple implementations of the subject specified, i.e., it is the opposite of a company specification that directly benefits the company's own ecosystem. A standard is designed to increase choice and benefit consumers. A successful standard has to have multiple implementations that conform and interoperate. If there's only one implementation then it's just a vendor specification, regardless of the process it was put through to get a stamp of approval.

Sun created the Java Community Process (JCP) to manage and maintain the evolution of the Java language. While it's easy to claim that the JCP is "controlled" by Sun, the JCP actually has more in common with an industry group building standards than a vendor-controlled specification to enhance a vendor's own ecosystem. The JCP has members well beyond Sun. It has a well-defined, consensus-based process to manage the myriad Java-related specifications. Membership is open to all to participate. Its purpose is to encourage multiple implementations of Java and not simply add-ons to Sun's Java world.

The hard part of any standards organization is how best to measure and warrant that an implementation conforms to the specifications to protect the value of the standard's

brand in the marketplace. How does one best signal to the marketplace that a subject is what it claims to be? Conformance measurement and certification is an expensive process. Sun, through the JCP, has put in place an expensive process to certify that Java technologies delivered by anyone do indeed meet the specifications.

Certifications are always taken on by the group that stands to gain the most (or lose the most in some cases). Essentially the group that cares makes the investment and develops a program to certify things against the standard.

POSIX was a standards effort defined by the IEEE that undertook to define an operating system interface in the C language to support application portability. The IEEE didn't handle conformance measurement however. The U.S. National Institute of Standards and Technology developed the certification to support U.S. government procurements. The IETF skips expensive certification processes after the fact, and instead uses the reality that they define networking standards to require that an RFC has two independent interoperable implementations to gain standards status. The people who economically need measurable conformance take responsibility for putting the system in place.

So too is the case with Java. This is as true for proprietary product specifications and their certification programs as it is for industry and de jure standards. The vendor stands to lose the most with respect to its proprietary specification's brand in the ecosystem, as any industry standard has to gain from the value of demonstrating that multiple implementations conform.

### Standards and Open Source Software

Standards exist to enable multiple implementations of a technology. Open Source software to a certain extent represents the one true implementation of a technology. When there's one true implementation there's no need for a standard. For example, there'll never be a Perl language standard. But the interaction is actually more subtle.

Standards typically occur in mature spaces where there's a wealth of experience and expertise. When it comes time to create a technology standard, the vendors in the space will pick a shared technology base from which to create a standard. Every vendor would love to claim its technology is the standard, and often make claims to having the "de facto" standard, essentially such a ubiquitous technology that it's a "standard in fact." The real world doesn't work that way however, regardless of how it's marketed. When true

“Software developed in successful Open Source collaborative communities shows all the hallmarks of well-developed software from other processes”



standards are delivered, they come from a shared technology base so that none of the participants feels another has a market-dominant starting position.

The standard will differ from that core shared technology base, but not so much that the shared base doesn't quickly morph to conform to the new standard, and the collection of vendors can quickly bring products to market. In today's world, Open Source software projects represent that shared technology base out of which standards can be delivered to facilitate multiple implementations.

This is a somewhat odd position then for Sun with Java. As the keeper of a primary reference implementation, and the creator of the standards development organization, it would seem Sun is in an odd place. Indeed, it's almost as if the process is working backwards.

### Open Source Java

So what will Open Source Java mean? First remember that this effort hasn't been driven by Sun, but demanded by the community around Java. Whether the demands are valid or not, or indeed politically motivated or not, there are still good things to be had from the process.

Sun has driven the Java standardization process through the JCP for some time and has a strong collaborative community and process, supported by a strong certification and branding program. Delivering Java technologies as Open Source still makes sense, however, even if the standard has led the implementation so to speak.

As a primary reference implementation, it will provide the following benefits to the entire Java community, Sun included:

- It will harden the primary implementation for Sun's and the community's benefit. Allowing others to tinker and explore will uncover new and interesting problems, which can then be addressed.
- It will enable new innovation. Many claim Java's day is done. Allowing new implementors to explore the primary production base will invariably lead to new ideas and innovation on the platform.
- As new code enters the source base, it will likely come in at a very high level of quality. Even if Sun developers act as the primary committers for the foreseeable future a high level of inspection will be brought to bear on code coming in from the outside. For new work delivered from the inside, the inspection by the community will likely be equally vigorous.

That is not to say that there won't be challenges. As with any large code base that exists in a commercial product, all will have to be inspected carefully from a number of angles. Obviously Sun won't want any immature code released, but Sun also has to ensure that all code licensed in from the outside can be released and manage that process.

Sun already supports a strong development community around OpenSolaris and hopefully that experience can be leveraged by the "OpenJava" team. Likewise, Sun already supports

a strong collaborative community in the Java Community Process, so it has a great channel to begin its Open Source efforts when it figures out how it intends to publish what sources. It began the release of Java EE 5 with the GlassFish project, and now time will tell if it can harness all its collective experience in Open Source software, standards, and the JCP to bring about a complete Open Source Java world.

### References

- The Open Source Initiative and the Open Source definition can be found at <http://www.opensource.org/docs/definition.php>.
- Free software as defined by the Free Software Foundation can be found at <http://www.fsf.org/licensing/essays/free-sw.html>.
- Sun's CDDL license can be found at <http://www.opensource.org/licenses/cddl1.php>.
- The Shared Source CLI can be found at <http://msdn.microsoft.com/net/sscli/>.
- The Mono project can be found at <http://www.mono-project.com/>.

**Stephen Walli** is vice-president of Open Source development strategy for Optaros ([www.optaros.com](http://www.optaros.com)), responsible for architecting and managing Optaros' relationships with the Open Source community. Before joining Optaros, he was an Open Source advocate at Microsoft, where he focused on "shared source" business strategies and was responsible for the technical implementation of Open Source-related community projects. He was a long-time participant and officer of the IEEE and ISO POSIX standards groups, representing both USENIX and EurOpen (EUUG) and has been a regular speaker and writer on open systems standards since 1991.

“Open Source software to a certain extent represents the one true implementation of a technology”



# Crystal Reports Can't Hold Its Java

## Only JReport Delivers 100% Java Embedded Reporting

**What a mess!** Especially when you try to integrate Crystal into your Java environment.

Only JReport delivers seamless integration with your Java EE application architecture – which helps reduce development costs, increase developer productivity and provide scalable access to operational data across the enterprise. After all, isn't that the point of reporting? JReport is the most flexible, scalable 100% Java embedded reporting solution available today.

Avoid being burned by Crystal. Learn how to simplify your reporting solutions with JReport.

Download the JReport Embedded Reporting Kit or call 301.838.5560 now.



[www.jinfony.com/JavaReporting](http://www.jinfony.com/JavaReporting)



# Where Has My Data Gone?

## Accessing Data in a Service Oriented Architecture

by Doug Clarke & Clemens Utschig

**W**ith software architecture evolving toward SOA, many projects in this space have encountered challenges associated with accessing data. As has been said, “The way an organization thinks about applications and data must evolve — it must stop thinking about data as a second-class citizen that only supports specific applications and begin to recognize data as a standalone asset that has both value and utility.”

In today’s world, two different types of data usage can be found: traditional architecturally layered applications and SOA framework-based applications. Naturally, each one comes with different technological and behavioral characteristics.

On one hand, there are traditional applications — usually designed and written in one language with clear separation between layers, such as enterprise Java with JSP and EJB, or .NET. Communication between layers happens in memory without any intermediate protocol (such as XML). On the other hand, there are SOA frameworks, such as Business Process Execution Language (BPEL) and the concept of the Enterprise Service Bus (ESB), offering everything that looks like a Web Service (exposed through a WSDL) to be orchestrated and to represent itself as a service.

“Organizations should establish their data environments with ‘hubs of specific data families’ that expose data services that comply with industry standards and service contracts. The goal is to create a set of services that becomes the authoritative way to access enterprise data. In this target service-oriented environment, applications and data work together as peers. Thus, both an organization’s business functionality and data can be leveraged as enterprise assets that are reusable across multiple departments and lines of business.”

In most cases, modular applications already exist and therefore data services need not be built entirely from scratch. This article focuses on aspects of migration and on exposing application functionality for later use in a SOA. It also discusses the pros and cons of the technologies being used for accessing data.

### Technologies Used To Implement Data Services

Table 1 lists common technologies used by applications to obtain data. The question is which of these different implementations of data services should you use in a particular application? There are no hard and fast rules, but this article provides some guidance. Obviously, they can be differentiated through different data formats and different access methods. This article uses these data access technologies to explore the different strengths and weaknesses of data access with SOA enablement.

In a SOA, these technologies are usually composed together, because not all services are implemented in the same technology. This brings up several challenges involving transactional behavior across boundaries, including performance and mass data behavior.

More challenges? So why would you want to introduce a separate set of services in your architecture versus directly accessing a data store? The reasons to consider a separate data service include:

- **Defined interfaces:** Using data services forces you to define contracts that are used between the service and its clients. This is the first step towards abstracting the contract (the interface) from the implementation.
- **Loose coupling/decoupling:** Although a data access layer typically encourages good encapsulation and decoupling of data access functionality, it does not force the issue. Having a separate data service not only forces a well-defined contract but also minimizes implementation details creeping into the client. A client cannot bypass the interface contract because consumer and provider are not necessarily implemented with the same technology.
- **Reuse:** Using a common data service automatically ensures that all consumers can reuse the same implementation, which leads in the long term to reduced maintenance costs (bug fixes, changes), because code is not duplicated. On the other hand, it requires a well-defined process of change control — because more con-



sumers rely on a functioning piece of code and potential downtime affects more than one user.

- **Flexibility:** Having the implementation completely abstracted from the consumer through a well-defined contract offers greater flexibility. Over time, the implementation technology can change; additional performance enhancements can be introduced; or data stores can be upgraded, migrated, combined, or divided — all of this without distracting the applications using the data service.

Based on the reasons described to introduce data services into your architecture, the next section takes the challenges arising from the benefits into consideration and maps them against available technologies.

### Common Data Challenges in a SOA

In general, the challenges can be divided into four main groups, each defining a different part of an overall application: access, enrich, distribute, and persist. Usually you start by thinking about how to access a certain data store. Should it be done via handwritten Java code that embeds SQL with JDBC calls? Should an object-relational mapping (ORM) approach be taken? Should the Java Connector Architecture (J2CA) be used to connect to a foreign data source? Is the client Java or a .NET application — and therefore can a native protocol be used or not?

After considering the access options, the next step is to validate whether aggregation of data across boundaries is necessary, whether a high data load is expected, and where the data comes from. All of these options are considered in the discussion of Challenge 2 (enrich, cleanse, and aggregate data).

Another major challenge involves which and how many users plan to consume a service. In particular, the importance of interoperability should not be underestimated if data is being pulled from or pushed to a consumer (such as business-activity monitoring [BAM]). This is described in the discussion of Challenge 3 (distribute data).

Last but not least, there is the challenge of writing data back and guaranteeing consistency across multiple calls to a service as well as across boundaries. In SOA, use cases are generally implemented across technological and service boundaries as they are orchestrated into a process. These questions are covered in the discussion of Challenge 4 (persisting data).

#### Challenge 1: Access Data

Abstraction of data - A domain model is abstracted from the underlying data store. This model should be designed according to the requirements of the client applications. It could mirror the underlying data store or provide a rich abstraction. In either case, the consuming application is decoupled from the physical storage by this model.

In reality this means that a customer object used in the application usually differs from the underlying data stores and their view of data, such as object-oriented versus normalized.

Speed considerations - Allow for efficient retrieval of domain objects based on exposed operations. This is basically the exposure of queries the application needs. It also provides a natural boundary for testing and for the introduction of mock data in test scenarios. Because the next generation of applications and business processes are composites that leverage these services, data access abstraction must be considered a key aspect of a service-oriented testing strategy

#### Challenge 2: Enrich, Cleanse, Aggregate Data

In modern applications, data usually does not come just from one place and get presented directly to consumers. It often needs to be formatted (e.g., a date), aggregated or enriched, or cleaned.

For example, a big part of an employee record might come from the HR system but the employee's vacation time is stored in some other place requiring a merge of those two sources into one.

There are several ways aggregation or enrichment can be implemented. At the most basic level, it can be done natively in the application no matter what the technology. However, if the underlying data model changes (or just a column name changes), several places need to be updated, which can be time-consuming. This leads to the use of object-relational mapping solutions because the domain model is easily shared and the mapping can be changed just in one place. On the more sophisticated side the data service implementation could access multiple data sources using native results to compose the common domain model.

When multiple data sources return a variety of XML documents that comprise the common domain model then XML aggregation is the answer. Due to the evolution of orchestration and XML as a common data format, BPEL is the perfect tool for aggregating or enriching data that is already in XML and later serving it to consumers. Although for large payloads

**Clemens Utschig** works within Oracle's SOA Product Management Team responsible for security aspects and cross product integration. Aside from technology, Clemens' focus is on the project management and consulting aspects that come along with SOA implementations. As a native Austrian, Clemens' career with Oracle started in Europe at the local consulting services branch — working with customers on J2EE and SOA projects, and founded the local Java community. He is a frequent speaker at conferences evangelizing either technology or the human factor — two key aspects when introducing new concepts and shifts in corporate IT strategy. [clemens.utschig@oracle.com](mailto:clemens.utschig@oracle.com)

**Doug Clarke** is a principal product manager for the Oracle Application Server focused on persistence and Oracle TopLink. Prior to his current role Doug worked as a lead developer, trainer, and professional consultant. Over the past decade his primary focus has been on helping global Fortune 1000 customers integrate relational and non-relational data into their enterprise Java applications. Doug is a frequent speaker at conferences and user groups. [douglas.clarke@oracle.com](mailto:douglas.clarke@oracle.com)

Physical Characteristics	J2CA (Java 2 Connector Architecture)	Native JDBC	EJB/JPA	Web Service
Data Format	Record or custom structure	ResultSet	Serializable Java Objects	SOAP/XML
Access Method	J2CA is part of the Java EE specification and is used within a Java EE container	In both Java SE and Java EE applications	Can be accessed from Java SE and Java EE applications	From everywhere and even with different implementation languages (Java/C++/VB/.NET/ASP.NET, etc.)
Transactions	Native J2CA is transaction-aware and can participate in a two-phase commit	JDBC provides native access to the database (therefore the transactional behavior has to be controlled from within the application directly)	EJB is by definition transaction-aware. This means a set of EJB method calls can participate in one transaction (and therefore can be rolled back)	Support is only in parts, mostly by using WS-TX (Web Service transactions). Spanning multiple calls into one transaction can be achieved with WS-Coordination. However, in a case with many loosely coupled interactions, the concept of compensation transactions is more common. To use this concept every action has to provide a counterpart, e.g., <code>bookHotel()</code> and <code>cancelHotel()</code>

Table 1 Data Access approaches for building Data Services



Aggregating Data	Technology Choice			
Technology Characteristic	J2CA (Java 2 Connector Architecture)	Native JDBC	EJB/JPA	Web Services
Ease of Development	Requires high skill levels	Enriching and aggregation of data can be achieved via SQL, but if aggregation happens with different sources, it is more difficult and requires significant custom coding	EJBs can be grouped and "orchestrated" through Session Facades — which allows for data enrichment and other business logic, but this requires significant coding effort	Aggregating Web Services is really easy, because of orchestration languages, such as BPEL, offering the ability to put them together into a business process. Also true for ESB
Performance	In container, in memory—good	Good performance can be achieved through in-database operations. If aggregation happens outside of the database, performance declines	Dependent upon the underlying data access. Potentially hard and performance-killing. Aggregation has to happen in the model	Performance really depends on the implementation of the service. The amount of serialization is especially critical to performance. In this case every element and byte counts
Ease of Change	Changing the logic results in high maintenance and effort, because it affects the whole application	Same as native JDBC	If separation is done correctly, only one part (one EJB) is affected	Orchestration can be changed, so even if the change affects the whole service, it's never the whole application

Table 2 How the technology behaves for aggregated data

and results, using an Enterprise Service Bus might be a better choice.

**Challenge 3: Distribute Data**

When all components and layers reside within one application, access is native and in memory. In a SOA, however, services run in a decentralized manner and in different places, introducing the challenge of different protocols and even greater need for solid error handling. What happens if a service is down or not accessible — how will the application be affected? The use of BPEL offers a great way of composing these services into one process or composite services, which can be leveraged later. Using this approach brings more freedom to the service's consumers but also introduces another layer that needs to be maintained.

One purpose of this is data pulled into several UI technologies, such as a portal or BAM to monitor performance.

For example, a supervisor wants to see the performance of employees in real-time and wants to be able to take corrective actions (such as rerouting a request to other teams). In this case, the data from the business process (and its services) can be pulled out into a BAM instance to create a real-time dashboard.

Note: The technology mapping of this section has already

Accessing Data	Technology Choice			
Technology Characteristic	J2CA (Java 2 Connector Architecture)	Native JDBC	EJB/JPA	Web Services
Ease of Development	Hard. Developers have to understand both the source system and the target implementation language	Moderately easy to develop and run any SQL possible. Hard aspects are mapping types. Knowledge of SQL is required. Complex databases can involve complex JDBC/SQL coding	With today's generation tools and EJB 3.0's Java Persistence API (JPA), this is easy. Mapping to the DB tables doesn't cause much pain at all	Web Services can be exposed from almost any object or structure today. Very easy to develop
Reuse (Ease of)	Yes, but the adapter instance only — no business logic	Moderate reuse of SQL statements, other logic not written in SQL is tied to the implementation	Remote EJBs are made for remoting and can be reused very easily	Web Services can be accessed from various technologies, not just Java
Security	Yes, the connector is secured by means of Java EE	The database is secured; security considerations are entirely on the database side	Due to the Java EE security infrastructure this is possible but implementation is hard	Web Service standards offer sophisticated security models for both authorization and authentication. However, it is best not to embed security policies in the app code — it is better to capture and execute it in a Web Services management and security solution
Performance	Yes, due to native in-memory access (application to adapter communication)	If SQL is slow, the whole app is slow. Use if the aggregation and mapping needs to be fast — and can be done through SQL	EJBs can be accessed locally and remotely. The native protocol is RMI. Many querying and data retrieval optimizations available	Performance is affected by serialization to and from XML on top of the underlying implementation costs
Ease of Change	The adapter is portable, but the business logic resides in the app	Client code — therefore it is hard. If logic resides in the database, it is hard to port. Changes to schema or queries can be expensive across all embedded SQL code	Operations are within the Java EE container, such as administration and deployment. Mapping changes can be isolated from the client	Because these services are locked to the interface but implementation technology can change, change is easy. Nevertheless, it has to follow a process

Table 3 How the technology behaves when being used to access data

been covered in the Access table.

**Challenge 4: Persisting Data**

A data access solution must provide proper transactional support allowing the application to apply changes to the data stores through the exposed operations and the supplied changes to the domain model, ensuring that the proper transactional semantics are obeyed, based on the data store's requirements and implementing any necessary concurrency protection (locking).

Especially with SOA, more than one application will use a certain service at the same time, so transaction protection needs to be considered. One possibility is that each operation

# PUTTING THE \$0 IN SOFTWARE

- ✓ Open Standards, 100% Java
- ✓ Cross-Platform, Modular, Scalable, Extensible
- ✓ Application Prototyping Utilizing NetBeans GUI Builder Matisse
- ✓ Free & Open Source Application Development Framework



can be reverted. Another option is to sacrifice loose coupling to allow clients to use transactional behavior.

### Evolving a Multi tier Application into a Reusable Set of Services

The goal for the next generation of applications is to morph them smoothly into a SOA — and not force them to be completely rewritten. If a clean separation of layers was introduced earlier, this may sound harder than it actually is. Let's review two possible morphing approaches:

#### 1. Traditional application (architectural model)

Following all the design patterns, a clean separation of concerns is desirable — especially moving forward to a SOA. In the model shown in Figure 1, the application (representing the business logic) talks to the persistence via a defined set of interfaces and transports its data through Plain Old Java Objects (POJOs).

#### 2. Morphing into a hybrid model

Expose just the needed artifacts and functions into service interfaces (a.k.a. service enablement). These can be further used in BPEL to orchestrate complex business processes. The application, on the other hand, still uses the native, in-memory approach and is not affected at all. The model stays the same (see Figure 2), and just certain pieces are exposed (the same interfaces!).

### Integrating Data Through Services: A Use Case

Having discussed the pros and cons of various access technologies, such as JDBC, J2CA, Web Services, and Enterprise JavaBeans (EJB) as well as the challenges of accessing data, it's time to apply the information gained to a use case.

“The key to a successful transformation is a solid understanding of the purpose of the data service”

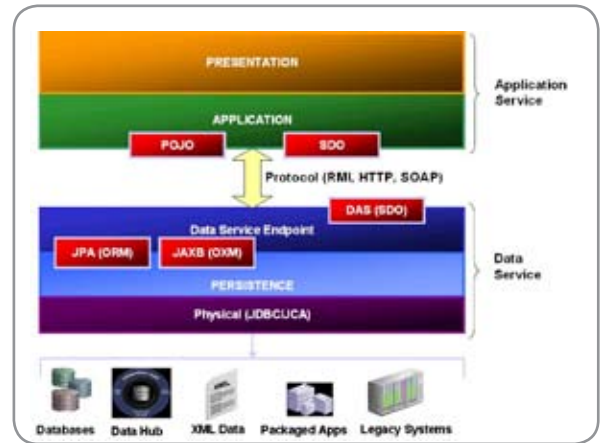


Figure 1 Complete separation of logic, with access only through remote protocols

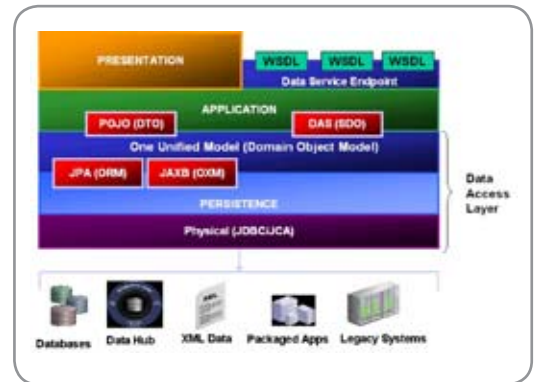


Figure 2 Layered model in which business logic is used locally as well as exposed for remote access

large company runs its inventory system as a mainframe application that is maintained and updated through daily batch jobs. Due to the rising demand of a new front-end system, the company decided to develop a custom Java EE application, with a Java Server Faces front-end, capable of serving multiple clients but holding the mission-critical information in the back-end. Between those two systems, new items should be exchanged and enriched with external data from a supplier. The service for retrieving this information is offered through a secured Web Service (see Figure 3).

Connecting these applications — all using different technologies — serves as a perfect example to illustrate choosing the right data access, aggregation, and persistence approach to build a best-of-breed solution.

Applying the technologies discussed about, J2CA connectors, which provide native access to the underlying technology, can be used for the mainframe. Most of these adapters also provide a WSDL interface to the outside world, describing their exposed services. Each time a new item is triggered, the adapter fires an event.

The Java EE application consists of a data access layer built on top of EJB 3.0 to ensure flexibility in the mapping between domain objects and the actual database schema. These EJBs can either be exposed as real Web Services or offer just a WSDL



interface but require native access.

In this case, an ESB can be used to provide the running infrastructure for orchestrating an overall flexible process. It not only contains rules for routing and aggregation but also offers error handling and ensures a high degree of data access and persistence performance by using native protocols.

### Outlined Process Flow (ESB System Diagram)

As this example shows (see Figure 4), an ESB can support the loosely coupled principles of a SOA but also address common data access challenges.

### Conclusion

A data service is a means of decoupling and encapsulating the access to one or more data stores. This concept offers an approach to sharing common functionality across multiple client applications or services. Its physical separation allows much greater flexibility in implementation and future independent evolution while guaranteeing that the consumer and the data service negotiate on a contract.

The benefits of this approach include the ability to transparently aggregate data and even completely change data stores without requiring changes to its consumers. Additionally, it allows for a data service that may have once been used only in a fixed set of applications to be used within a business process as a more dynamic service orchestrated with declarative processes.

Each technology that has been discussed has pros and cons (starting with performance and ending with transactionality). We believe that it is important to keep those in mind and that the more data-rich the application is (the more mass data it has) the more a native approach is appropriate. On the other hand, the more loose coupling is targeted, the more the approach should lean toward a Web Service-based architecture. In this sense, using BPEL seems like a great way to enable data sources into services and allow performance and, to some extent, transactionality.

The key to a successful transformation is a solid understanding of the purpose of the data service, including the pros and cons of each technology used. The use of proven data access frameworks makes exposing data as services simple while not sacrificing performance. Good performance is essential in multi-layered applications, because it is crucial to the user experience. The best SOA is worth nothing if the users are sick of using the consuming applications due to their poor performance.

Although having shared data services appears to be an obvious solution for shared access to the same enterprise data stores, it does not come without costs. As in all design and architectural abstractions, the benefits must be carefully weighed against the costs and challenges. A vision of the enterprise's SOA strategy must also be taken into consideration when deciding when and where to use data services. This article has attempted to alleviate the difficulty in addressing the challenges of making these decisions in a SOA by describing and assessing the characteristics of data accessibility.

### Reference

<http://webservices.sys-con.com/read/233667.htm>

Persisting Data	Technology Choice			
	J2CA (Java 2 Connector Architecture)	Native JDBC	EJB/JPA	Web Services
Ease of Change	Change is hard, because the record structure differs adapter to adapter, even if the interfaces are standard	Extremely hard, because the mapping of columns is coded somewhere in the application	While the bindings are embedded in the application the change is much simpler than a hand-coded JDBC-SQL layer	Easy, because with a defined contract, the implementation can change without affecting the consuming application
Transactions	As mentioned, J2CA adapters can participate in global transactions and provide the means for a two-phase commit	When the whole application uses only native JDBC access, it can control the commit cycles and the locking of resources around them. However, in a SOA, commits are atomic, therefore optimistic locking should be used	EJBs provide transactional capabilities and can participate in container transactions. However, when exposed as a Web Service, commits become atomic, meaning after each completed method commit is called implicitly therefore optimistic locking should be used. When you use them in a business process, you should foresee compensation logic	Web Services offer WS Transactions to control commit behavior. But when orchestrated in a BPEL process, each Web Service should provide compensation logic

Table 4 How the technology behaves in the context of persisting data

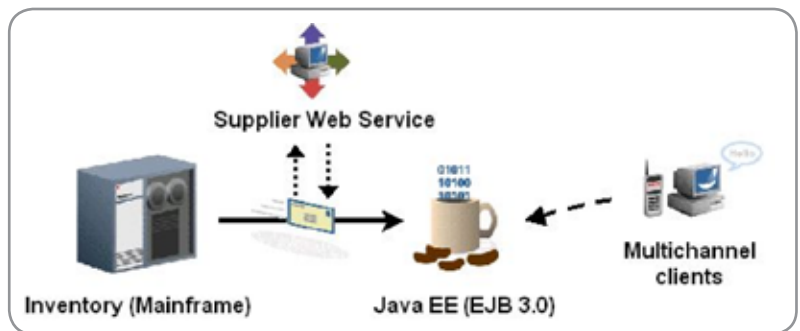


Figure 3 Data Services Sample Use Case

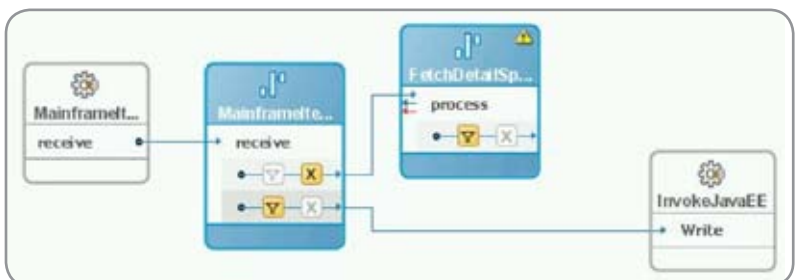


Figure 4

# Mailets and Matchers

*Using Apache James and JavaMail to implement variable envelope return paths*

by Aaron Johnson

**A**pache James is a full-featured SMTP, POP3, and NNTP server built using 100% Java and more importantly it's been designed from the ground up to be a mail application platform

Anyone who's spent any time working on an application that sends e-mails has come across more than his fair share of bounced e-mails. If you actually read the bounced e-mails, you probably noticed that many of them either came from the ISP's error handler (mailer-daemon@isp.com) or from an e-mail address that wasn't on your mailing list. The bounced message may not even have contained a copy of the original message. All of above scenarios make it very hard to figure out who the original message was sent to. Enter Daniel Bernstein, also known as djb, who in 1997, in response to this problem of matching bounced e-mail messages to subscription addresses, wrote a paper describing a technique he called Variable Envelope Return Paths or VERP for short. In the paper, he describes process as:

*...each recipient of the message sees a different envelope sender address. When a message to the djb-sos@silvertan.berkeley.edu mailing list is sent to God@heaven.af.mil, for example, it has the following envelope sender:*

```
djb-sos-owner-God=heaven.af.mil@silvertan.berkeley.edu
```

If the message bounces, the bounce message will be sent back to djb-sos-owner-God=heaven.af.mil@silvertan.berkeley.edu.

If God is forwarding His mail, the bounced message will still go to djb-sos-owner-God=heaven.af.mil@silvertan.berkeley.edu. No matter how uninformative the bounced message is, it will display God's subscription address in its envelope.

But you probably noticed that this article isn't only about VERP: Apache James is a full-featured SMTP, POP3, and NNTP server built using 100% Java and more importantly it's been designed from the ground up to be a mail application platform. The James mail application platform makes it a perfect candidate for handling bounced messages using VERP. Similarly, the JavaMail API is a framework for building mail and messaging applications using SMTP and POP3.

JavaMail makes it easy to customize the envelope sender address, which means Java developers can use JavaMail on the client and James on the server to build e-mail applications that enables VERP.

This article will describe an example VERP implementation, show how JavaMail can be used to modify the envelope sender address, and then illustrate how James can be used to recognize and process bounced e-mail messages. It's not intended to be an in-depth look at either the Apache James mail server or the JavaMail API. If you're interested in learning more about Apache James, a product review is available on the Sys-Con.com Web site (<http://java.sys-con.com/read/38667.htm>) and an extensive introduction to Apache James on the IBM developerWorks sit (<http://www-128.ibm.com/developerworks/library/j-james1.html>). The JavaMail API can also be reviewed on the Sys-Con.com site: <http://java.sys-con.com/read/36545.htm>.

## VERP and JavaMail

Let's start by looking at the e-mail newsletter that a fictional store called "Javazon" is sending to its customers. The developers at Javazon have been using the JavaMail API to successfully send the newsletter through their mail server using code similar to the example below.

```
String sendere-mail = "deals@javazon.com";
String toe-mail = "ajohnson@cephas.net";
Properties props = new Properties();

props.put("mail.smtp.host", mailserver);

Session session = Session.getInstance(props, null);

javax.mail.Message m = new MimeMessage(session);

m.setFrom(new InternetAddress(sendere-mail));

m.setSubject("New Deals at Javazon!");
    m.setRecipient(javax.mail.Message.RecipientType.TO,
new InternetAddress(toe-mail));
```

**Aaron Johnson** is a senior software engineer at Jive Software. He lives with his wonderful wife, young son and dog in a Portland, Oregon. You can find out more by reading his blog at <http://cephas.net/blog/>.

```
m.setContent(content, "text/plain");
```

```
Transport.send(m);
```

The code above will produce an e-mail message with headers that look like this:

```
Date: Wed, 26 Apr 2006 21:00:21 -0000
From: deals@javazon.com
To: ajohnson@cephas.net
Subject: New Deals at Javazon!
```

Because they want to be good e-mail citizens, the developers at Javazon use the POP3 functionality in JavaMail to retrieve the e-mails that bounce back to the address specified as 'sendere-mail' in the example above. Unfortunately, many of the bounced e-mails come from daemon accounts (instead of the recipient e-mail address), which makes it difficult to figure out what e-mail address the original message was sent to.

As mentioned at the beginning of this article, the only way to address the bounces that come from daemon accounts is to use VERP, which is a two-part process. The first is relatively simple. An e-mail message, according to the SMTP RFC-821 Section 2, is composed of two parts: an envelope that contains the SMTP source and destination addresses and the message, which consists of the headers and message body. To create a VERP-capable e-mail message, you only need to modify the envelope, which is easily done using the instance of `java.util.Properties` associated with the `javax.mail.Session`. Modifying the first example, the developers would end up with this:

```
String sendere-mail = "deals@javazon.com";
String toe-mail = "ajohnson@cephas.net";
String verpFrom = "deals-" + toe-mail.replaceAll("@", "=") + "@javazon.com";
Properties prop = new Properties();
prop.put("mail.smtp.from", verpFrom);
prop.put("mail.smtp.host", mailserver);
Session session = Session.getInstance(props, null);

javax.mail.Message m = new MimeMessage(session);

m.setFrom(new InternetAddress(sendere-mail));

m.setSubject("New Deals at Javazon!");
    m.setRecipient(javax.mail.Message.RecipientType.TO,
new InternetAddress(toe-mail));

m.setContent(content, "text/plain");

Transport.send(m);
```

When executed, this code would create an e-mail message with headers that look like this:

```
Return-Path: <deals-ajohnson=cephas.net@javazon.com>
Date: Wed, 26 Apr 2006 21:00:21 -0000
From: deals@javazon.com
To: ajohnson@cephas.net
Subject: New Deals at Javazon!
```

Notice the different "Return Path:" header from the first e-mail? If the e-mail message bounces back to Javazon, it will go

to the e-mail address associated with the 'Return Path' header: "deals-ajohnson=cephas.net@javazon.com" rather than "deals@javazon.com." This is where Apache James comes into the picture.

## VERP and James

James can be configured and used like any other e-mail server, but its real power comes from the ability it gives Java developers to plug right into the mail-processing pipeline. James enables you to process e-mail messages in the same way you might process HTTP requests that come into servlet containers like Tomcat, but in a more flexible manner. If you want to pre-process (or post-process) HTTP requests in Tomcat, you first create a class that implements the `javax.servlet.Filter` interface and then you create an entry in your `web.xml` that matches certain requests to that class. Your configuration might look something like this:

```
<filter>

<filter-name>myfilter</filter-name>

<filter-class>com.javazon.web.filters.GZipFilter</filter-class>

</filter>

<filter-mapping>

<filter-name>myfilter</filter-name>

<url-pattern>*.jsp</url-pattern>

</filter-mapping>
```

The servlet container limits how you match requests to a filter: you're limited to pattern matching on the URL. Instead of a `<filter>` and `<filter-mapping>`, James gives you a `<maillet>` made up of two parts: `Matchers` and `Maillets`. They're described on the James wiki:

“Apache James is a full-featured SMTP, POP3, and NNTP server built using 100% Java”



# “The only way to address the bounces that come from daemon accounts is to use VERP, which is a two-part process”

“Matchers are configurable filters which filter mail from a processor pipeline into Mailets based upon fixed or dynamic criteria.

“Mailets are classes that define an action to be performed. This can cover actions as diverse as local delivery, client-side mail filtering, switching mail to a different processor pipeline, aliasing, archiving, list serving, or gateways into external messaging systems.”

James ships with a number of Mailets and Matchers that you can use without writing a line of code, but the developers at Javazon will have to write their own Matcher and Maillet to handle the bounces generated from their e-mail campaigns.

So the first thing they’re going to need to do is create a class that intercepts the bounced e-mails. A matcher class can be created in one of two ways: a) create a class that implements the `org.apache.maillet.Matcher` interface, or b) create a class that extends the `org.apache.maillet.GenericMatcher` class. Because `GenericMatcher` already implements both `Matcher` and `MatcherConfig` and because it provides a simple version of the lifecycle methods, the path of least resistance is to extend the `GenericMatcher`. The `NewsletterMatcher` class is going to “match” only the recipients where the address of the recipient starts with the string “deals-”:

```
public class NewsletterMatcher extends GenericMatcher {
    public Collection match(Mail mail) throws MessagingException {
        Collection matches = new ArrayList();

        Collection recipients = mail.getRecipients();

        for (Iterator i=recipients.iterator(); i.hasNext(); ) {
            String recipient = (String)i.next();

            if (recipient.startsWith("deals-")) {
                recipients.add(recipient);
            }
        }

        return matches;
    }
}
```

The `NewsletterMatcher` class, as you can see, returns a `Collection` of `String` objects, each presumably an e-mail that has

bounced. To do something with these matches, developers will need to write a class that either implements the `org.apache.maillet.Maillet` interface or a class that extends the `org.apache.maillet.GenericMaillet` class. Again, it will be simpler to extend the `GenericMaillet` class:

```
public class NewsletterMaillet extends GenericMaillet {

    private static CustomerManager mgr = CustomerManager.getInstance();

    public void service(Mail mail) throws MessagingException {
        Collection recipients = mail.getRecipients();

        for (Iterator i=recipients.iterator(); i.hasNext(); ) {
            String recipient = (String)i.next();

            if (recipient.startsWith("deals-")) {

                int atIndex = recipient.indexOf("@");
                String rec = recipient.substring(0,atIndex)
                    .replaceAll("=", "@")
                    .replaceAll("deals-", "");

                mgr.recordBounce(rec);

                mail.setState(Mail.GHOST);
            }
        }
    }
}
```

In the example above, the `NewsletterMaillet` class overrides the `service()` method in the `GenericMaillet` class, loops over the list of recipients in the given e-mail message and then checks to see if the recipient e-mail address starts with the string “deals-”. If the recipient e-mail address starts with “deals-” then the class decodes the original recipient address by retrieving what is generally the username part of the e-mail address, replacing the equals sign (=) with an @ sign and then replacing the “deals-” prefix. Then the `Newsletter` maillet class uses `CustomerManager` (a class that the Javazon developers use to manage customer information) to record the bounced e-mail. If you were to step through the process, you’d see the recipient e-mail address start something like this:

```
deals-ajohnson-cephas.net@javazon.com
```



# IT'S IN THERE SOMEWHERE

**MAKE ANSWERS TO PERFORMANCE PROBLEMS COME TO YOU.**



**OPNET** **Panorama**  
Real-Time Application Analytics

**OPNET Panorama** offers powerful analytics for rapid troubleshooting of complex J2EE/.NET applications. Panorama quickly identifies how application, web, and database servers are impacting end-to-end performance. With Panorama, you can pinpoint the source of a problem, so time and money aren't spent in the wrong places.

*The most successful organizations in the world rely on OPNET's advanced analytics for networks, servers, and applications.*

[www.opnet.com/pinpoint](http://www.opnet.com/pinpoint)

**OPNET**<sup>®</sup>  
Making Networks and Applications Perform<sup>™</sup>

**OPNET Technologies, Inc.** 7255 Woodmont Avenue, Bethesda, Maryland 20814 phone: (240) 497-3000 • e-mail: [info@opnet.com](mailto:info@opnet.com) • NASDAQ: OPNT

© 2006 OPNET Technologies, Inc. All rights reserved. OPNET is a registered trademark of OPNET Technologies, Inc.

# “There are a number of other interesting ways you can improve your e-mail processing by extending James using mailets and matchers”

and then change to this:

```
deals-ajohnson=cephas.net
```

and finally to this:

```
ajohnson@cephas.net
```

The last step is to wire the mailet and matcher classes together in the Apache James configuration file, which is usually located here:

```
$JAMES/apps/james/SAR-INF/config.xml
```

You'll need to make a number of entries. First, you'll need to let James know where it should look for the mailet and matcher classes you've created by creating `<mailetpackage>` and `<matcherpackage>` entries inside the `<mailetpackages>` and `<matcherpackages>` elements:

```
<mailetpackages>
...
<mailetpackage>com.javazon.mailets</mailetpackage>
</mailetpackages>
<matcherpackages>
...
<matcherpackage>com.javazon.matchers</matcherpackage>
</matcherpackages>
```

Then add references to the matcher and the mailet using a `<mailet>` element like this:

```
<mailet match="NewsletterMatcher" class="NewsletterMailet">
  <processor>transport</processor>
</mailet>
```

The `match` attribute of the mailet element specifies the name of the matcher class that should be instantiated when the matcher is invoked by the spool processor and the `class` attribute specifies the name of the mailet that you want invoked should the matcher class return any hits.

After adding these configuration entries and adding the compiled classes to the `$JAMES/apps/james/SAR-INF/lib/` directory, restart the James process.

## Testing

To test the configuration/application, you'll have to have a James server configured and available via the Internet via port

25 with a valid DNS name and a corresponding MX record. As an example, the system administrator at Javazon would configure a machine with James, make it available to the Internet on port 25, and assign it a domain name like `bounces.javazon.com`. The developers could then send an invalid e-mail using JavaMail to:

```
bounceme@javazon.com
```

(an account that probably doesn't exist on the main `javazon.com` mail server) with a return path of:

```
deals-bounceme=javazon.com@bounces.javazon.com.
```

The bounced e-mail will be sent to the server associated with the MX record for the domain name `bounces.javazon.com`, which should be the server the system administrator set up above. The `NewsletterMatcher` class should 'match' on the "deals-" prefix and then pass it to the `NewsletterMailet`, which should record the bounce using the `CustomerManager` instance.

## Conclusion

After reading this article, you should hurry on over to the Apache James Web site, download the latest distribution, and read the documentation. There are a number of other interesting ways you can improve your e-mail processing by extending James using mailets and matchers.

## References


### JavaMail

- <http://java.sun.com/products/javamail/>
- <http://jdi.sys-con.com/read/36545.htm>
- <http://www.ibm.com/developerworks/java/edu/j-dw-javamail-i.html>

### VERP

- <http://cr.yip.to/proto/verp.txt>

### Apache James

- <http://james.apache.org/>
- <http://jdi.sys-con.com/read/38667.htm>
- <http://www.ibm.com/developerworks/java/library/j-james1.html>
- <http://www-128.ibm.com/developerworks/java/library/j-james2.html>
- [http://james.apache.org/spoolmanager\\_configuration\\_2\\_1.html](http://james.apache.org/spoolmanager_configuration_2_1.html) 



**ADOBE AD FPO**

# Money, Freedom and Open Source

by Joe Winchester

**T**he current polemic with Java and Open Source boils down to two important issues: money and power.

## Money

In 1996, Sun created Java and the terms under which it is distributed. Since then, the Java Community Process (JCP) has emerged, allowing companies to participate in shaping language changes, but the ownership of trademarks, licensing agreements, branding, and other fundamental product issues remains unchanged. One is reminded of this fact every time the Sun Microsystems™ trademark appears alongside the Java coffee cup logo, or when one is greeted with the message “brought to you by Sun Microsystems” at [www.java.com](http://www.java.com). For anyone to use the Java-compatible logo on a product requires verification against the test compatibility kit (TCK), for which one has to enter into negotiations with Sun. Java, the technology, the trademark, and the language, are owned by Sun.

The current licensing agreements for Java generate revenue for Sun in two ways; one is through direct fees to its licensees, and the other is through indirect revenue generated off the back end of Java's success.

When asked how much income is generated from Java, Jonathan Schwartz, CEO of Sun, replied, “about \$13 billion.” He went on to explain that this figure is calculated from many sources, highlighting the revenue generated by licensing products that sit on top of the Java runtime stack. [http://www.forbes.com/work/management/2006/05/04/sun-microsystems-schwartz-cz\\_ec\\_0504schwartz.html](http://www.forbes.com/work/management/2006/05/04/sun-microsystems-schwartz-cz_ec_0504schwartz.html) This demonstrates a mind shift on the part of Sun senior management regarding how Java income should be generated, with a move from direct to indirect revenue streams.

## Power

According to the Oxford English Dictionary, “power” is “the possession of control or command over others; authority; ascendancy.”

For Open Source to succeed power, must be relinquished and transferred.

One of my favorite essays on the subject was written by Simon Phipps, the chief Open Source officer at Sun. <http://www.webmink.net/free/Free-ix.htm> In it, he discusses how the word “free” in Open Source means much more than giving away something for nothing; that “‘free’ in this context is not about the price; it is about the liberty. ‘Free’ here is as used in the phrase ‘free speech.’”

One of the perceived problems of Open Source often focused on by its naysayers, is that, with disparate groups of individuals, each with separate agendas, paymasters, and self-interests, the effort will collapse under the weight of its own entropy and confusion. Simon's counter to this argument is that a “‘community of code’ maintains a code base of Open Source components or elements, using the behaviors and principles of the Open Source movement. These inherently lead to better code being created, debugged and documented faster, not least because of the scrutiny of the community.”

I am fortunate to be part of the Eclipse project which allows me to witness such dynamics on a daily basis, so I concur that Simon's vision of what defines a truly free Open Source project definitely works in practice. Within Eclipse, I work with companies that are fierce competitors in the marketplace with my daytime employer, however, together we shape and build the common codebase for the benefit of the greater good: our collective community of customers. Examples of the “freedom” that Simon talks about are that the Eclipse.org web site does not provide disproportionate links to any of its member companies' commercial products, the Eclipse codebase has large and diverse representation of code committers across its member companies, and EclipseCon conferences are not dominated by marketing speeches from CEOs of any of its member companies. It is the perfect implementation of Simon's vision for how Open Source flourishes when practiced well.

## Actions Speak Louder Than Words

Jonathan Schwartz understands how commercial offerings that sit on top

of the Open Source stack are key to Java's indirect revenue. Simon Phipps understands the dynamics of how successful projects operate, writing in his freedom essay, “Open Source is not just about the code; it is about the community. You don't make a project Open Source simply by publishing the source code.”

When we are told that Java has finally become Open Source, we can judge its success or failure by its meeting the following criteria:

- Use of the Java trademark is equal among all community members, so that no one community member can brand a product at “Java XXX” while dictating that another cannot.
- The image of Java in the marketplace is of a community of companies. The Java logo and Java branding are owned by the community, and not by any one of its member companies. Websites such as [java.com](http://java.com) or [java.net](http://java.net) cannot carry trademarks specific to member companies disproportionately. Links and marketing stories about commercial products do not favor one member company over another.
- Content and material for conferences like JavaOne are selected in a way that benefits the attendees, rather than benefiting any one community company's marketing agenda.
- The number and affiliation of committers to the core codebase is diverse and representative of the participation of the member companies.
- No rhetoric exists in the Java community, so for parts of the language that are outdated legacy, the community decides what to do for the greater good.
- “Java” certification for one's own implementation of the language, on any hardware or operating system platform, can be obtained by having access to the TCK at no cost

I just hope for the future of Java that behind all of the current discourse on the subject from Sun, it won't become another case of “do as I say and not as I do.” ☛



Solution  
PARTNER



# *I have the next great software idea.*

*I know how to change my industry.*

*I know people.*

*I know how to get investors on board.*

*I can lead.*

*I can inspire.*

*I need to do this.*

**YEAH, THAT'S ME.**

**Unlock your potential** with the help of industry leaders in Rich Internet Application development. Discover how everyday we help people just like you at [cynergysystems.com/thatsme](http://cynergysystems.com/thatsme).



[cynergysystems.com/thatsme](http://cynergysystems.com/thatsme)

# SWING

## *Its past, present, and future*

by Hans Muller and the Swing Team

**N**early a decade ago, when Java was still a fledgling portable software platform and the Tumbling Duke applet was considered cutting edge, the members of the newly minted Swing team, including yours truly, took in a packed JavaOne session given by Sun's JavaSoft president, Alan Baratz. He told the assembled multitude that our team would be delivering a new GUI toolkit in just 90 days. Although we'd been working on what was called a "lightweight toolkit" for some time, he hadn't bothered to mention the new project deadline to us. Until that moment. If there'd been enough room, we would have all fallen off our chairs.

A rather limited "0.1" Swing release did debut 90 days later. The fact that developers not only adopted this early

version of Swing, but actually built applications and even products with it, is testimony to the tidal wave of enthusiasm for all things Java that started in the mid 1990s. Sadly, the reverberations from the initial developer-enthusiasm spike, the one that occurred with Java's debut in 1995, began to fade as the years passed. By late 1998, when we released Swing 1.0 as part of the "Java 2," the Swing team was up to its ears in reality. Performance, native look-and-feel fidelity problems, and the usual bugs were dragging the project down.

Remarkably, the Swing team and the larger desktop client software group persevered. The implementation of the Java client software stack was extensively profiled and tuned, and by the next major Java release, performance was



NOG



respectable. In subsequent releases, performance ceased to be an issue at all. Native look-and-feel fidelity and robustness concerns have also fallen by the wayside. In JDK 5 and subsequent JDK releases, the task of rendering native components has been delegated to the native GUI toolkit, so Swing looks just as native as the natives on Linux, Macintosh, even Microsoft Vista.

New software fashions tend to have a half-life of about 18 months. After that the books start to move to the discount bin, the conferences go quiet, and developers move on to the next big thing. Swing has not suffered that fate. As the software has stabilized and improved over the years, it has attracted a loyal and growing community of developers who've built tens of thousands of applications deployed to millions of users. No less of an authority that Evan's Data Corporation has reported that "Java Swing with 47% use, has surpassed WinForms as the dominant GUI development toolkit." Not bad for a (nearly) 10 year old.

Joe Winchester, the Java Developer's Journal Desktop Java Editor, thought it would be interesting to take stock of Swing's future in an interview-style article and, naturally, we were happy to oblige. Joe provided most of the questions and various members of the Swing team, past and present, have provided answers.

*The Desktop Java track at JavaOne 2006 was one of the best attended and there was a lot of interest around Swing. This seems a little surprising, given all of the hoopla surrounding new software trends like AJAX, and the debut of Java EE 5.*

There are various ways to slice the data from JavaOne, like the number of attendees in our sessions, the relative popularity of our sessions, the ratings for individual talks, and the average ratings for our track overall. We were quite pleased with all of these metrics: the attendance numbers were great, the sessions were popular, the Desktop track had some of the top-rated talks in the conference, and the Desktop track overall was near the top of all tracks.

However, our favorite metric to wax poetic about this year is this: our participation from external speakers was higher this year than ever before. For

Contributors to this article include the following past and present members of the Swing team:

Richard Bair  
Chet Haase  
James Gosling  
Romain Guy  
Michael Knyazev  
Rick Levenson  
Josh Marinacci  
Phillip Milne  
Hans Muller  
Ethan Nicholas  
Alexander Potochkin  
Scott Violet  
Steve Wilson  
Arnaud Weber  
Jeff Dinkins  
Shannon Hickey  
Igor Kushnirskiy  
Tim Boudreau  
Thorsten Laux

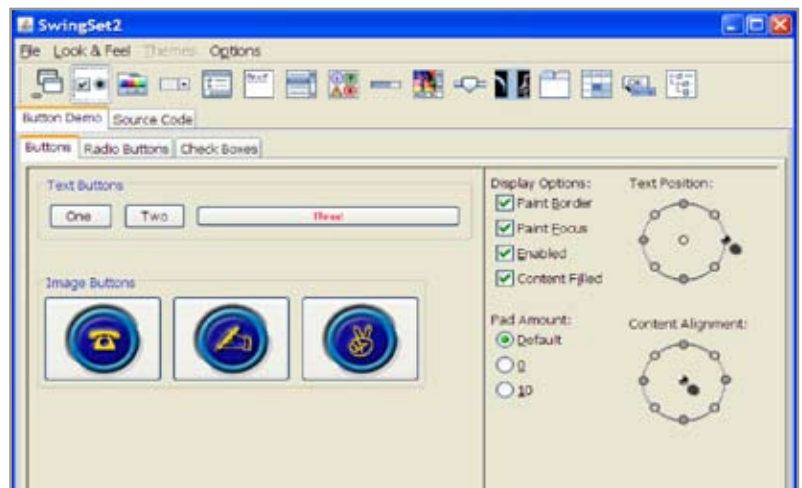


Figure 1 Swing Windows Look and Feel on XP

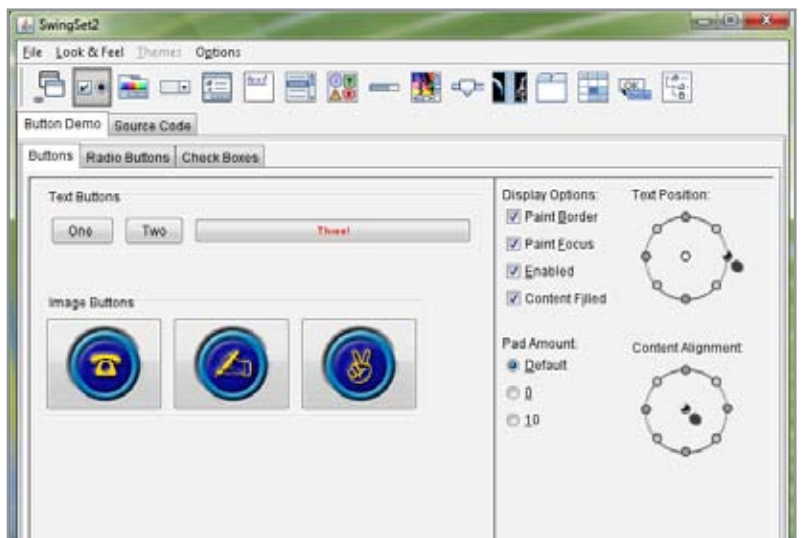


Figure 2 Swing Look and Feel on Vista beta (5384)

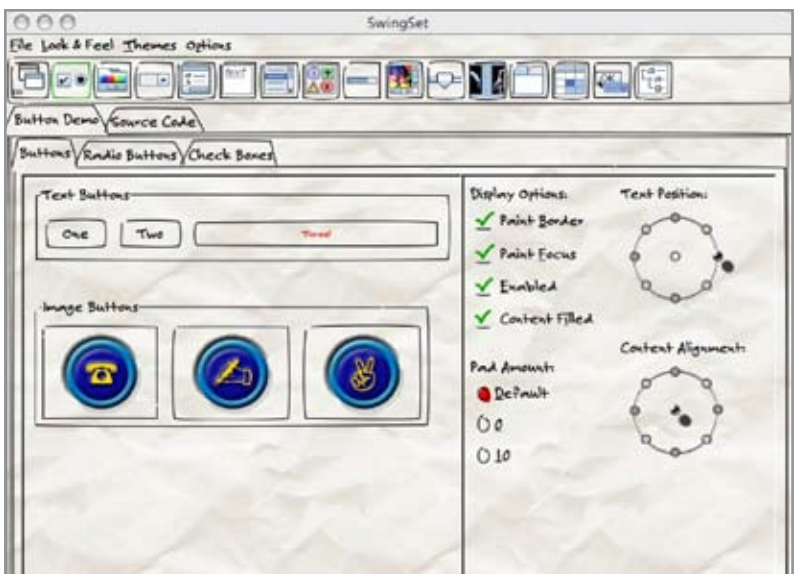


Figure 3 This is not a sketch; it's the Napkin Look & Feel. See <http://napkinlaf.sourceforge.net/>

# AJAX for Java



Backbase offers a comprehensive AJAX Development Framework for building Rich Internet Applications that have the same richness and productivity as desktop applications.

The Backbase AJAX Java Edition:

- is based on JavaServer Faces (JSF)
- runs in all major Application Servers
- supports development, debugging and deployment in Eclipse
- embraces web standards (HTML, CSS, XML, XSLT)

Download a 30-day Trial at [www.backbase.com/jsf](http://www.backbase.com/jsf)

example, in 2005 we had something like 11 Sun talks and eight external talks. This year, we had only six Sun talks and 14 external talks.

There are a couple of takeaways here, but the most interesting one in the context of this article is that we accepted a majority of external talks because people are doing some seriously great stuff with Desktop Java, and had some excellent and deep talks about their work. This included interesting case studies, as usual, but it also included some wonderful presentations on real-world development needs, such as Swing frameworks, data binding, or threading. The fact that there are known experts in the world like Ben Galbraith, Karsten Lentzsch, and Scott Delap who can give in-depth views into how to deal with this stuff is fantastic; it means that there is a growing community of people who know the platform well, and that JavaOne is not just about Sun evangelizing its platform.

We're hoping to see another excellent set of presentation submissions this year; let's keep that Desktop track momentum!

*What are the right ingredients for a renaissance in Swing, and do you think they're there at the moment?*

It's possible that we're in the middle of a Swing renaissance, although the evolution of the Desktop Java platform and the growth of its use has been pretty steady for a longer period of time than the typical technology era.

There clearly has been a renaissance of desktop user expectations recently, and Swing does fit rather nicely into that. Web users are no longer satisfied with glorified interactive brochures; they're demanding the same rich interactive experience they've always had with standalone desktop applications. Web-started Swing applications (even applets) provide that. Modern desktop applications also depend more and more on data pulled from various Web services. To provide dynamic, directly manipulatable visualizations of data drawn from the Net requires many things, not the least of which is support for highly threaded desktop applications. Swing and Java SE provide extensive support for this kind of work.

*Tell me about the SwingLabs project. What are its aims, and how do you see it working alongside mainstream Java SE development.*

SwingLabs is an open source project on Java.net that provides tools and libraries to help you make your Swing apps better. Swing Labs is entirely desktop focused and the name tries to express the idea that some of the projects may eventually make it back into Java SE. The SwingX subproject, for example, has a date picker, an error dialog, an Outlook style task pane, and a sortable table model. These are components people have been wanting for years. Some or all of these may make it into future versions of Java, but you can try them out now at [swinglabs.org](http://swinglabs.org).

SwingLabs has subprojects for other cool things like animations (the Timing Framework), working with threads (SwingWorker), using the native Web browser (the JDIC project), and working with mapping and Web services (the SwingX-WS project). You can find SwingLabs at [www.swinglabs.org](http://www.swinglabs.org) and [swingx.dev.java.net](http://swingx.dev.java.net).

*There is a lot of interest in projects like data binding and the application framework. What is the thinking behind these?*

Both of these JCP standard efforts, JSR-295 and JSR-296, respectively, have the same overall goal: simplify building Swing applications. Both projects aim to do so by eliminating much of the boilerplate code required by common application types.

JSR 295, "Java Beans Binding," trivializes keeping a pair of properties in sync, including automatic type conversion and validation. It also provides support for Java collections, database types like RowSets and (new in Java SE 6) DataSets, and Swing component models.

JSR 296, "Swing Application Framework," provides support for common application elements, notably resources and actions, as well as the application startup/run/shutdown lifecycle. Simplifying resource injection and writing multithreaded applications are significant goals.

We're hoping that the work on JSRs 295 and 296 will be completed in time for the Java SE 7 release. Prototype implementations of the APIs will be evolved out in the open, and we plan to make versions available for the current Java SE 5 and Java SE 6 releases.

The work on these JSRs reflects a trend. We're focusing more of our energy on providing higher-level support for application building.

*Does this mean that Swing is moving beyond just being a GUI toolkit toward a more complete solution?*

Yes. We look forward to a future where all applications will be built with Swing. From accounting software to computer vision systems, all software will be Swing, all the way down. In fact, we don't plan to rest until all things, from laundry soap to city governments, from fast food to artificial limbs have been created with Swing. The complete solution will make Swing literally the fabric of our lives. We're marching towards intergalactic Swing domination.

Ahem.

*[Note to editor: Please consider the previous statement off the record.]*

When you consider the overall desktop client stack, the extensive support for graphics and internationalization and deployment, and beyond that the overall Java SE platform, networking, and threads, and database access, and all the rest, what we're really up to is trying to simplify building complete solutions. All of the building blocks are there; what's important is making it easy to put them together.

Two important aspects of that are up-leveling, which was mentioned earlier, and "toolability." Historically, we have not been particularly strong on the tools front, supporting Swing in tools has required a great deal of custom work. There's no silver bullet for fixing that, and most GUI toolkits make similar demands of tools. However, we are mindful of the need to create APIs that take tools support into account. The GroupLayout layout manager, which was designed with direct manipulation GUI layout design in mind, is a good example of that.

*Some developers like to use Swing to create very high-fidelity desktop applications and shy away from its emulated*



# “SwingLabs is an open source project on Java.net that provides tools and libraries to help you make your Swing apps better”

*widgets, whereas others seem to enjoy its lightweight flexibility to produce very creative and rich user interfaces with dynamic content. Which set of users do you most see as your target user?*

With Swing, it's possible to have your cake and eat it too. If you want your buttons to look exactly like native buttons on Vista, we support it. If you want your buttons to look like squishy Jelly Bellies (tm), we support that too. The ability to support multiple look and feels is core to Swing's design.

In the early releases, Swing's rendering of platform-native components was off in a number of areas, and we got our collective ears filled with complaints about that. As of JDK 5 update 8, we've fundamentally changed the way native L&F components are rendered on Windows. Swing on Windows XP (and on the upcoming Windows Vista release) uses the native UXTheme API to render components. This means there's no guessing or mimicry. GUI components will look exactly like their native counterparts because they're rendered by the same code. This approach also works nicely for Windows desktop themes, even third-party visual styles. When Vista finally ships, Swing apps will look native. You'll see similar changes, based on the GTKStyle class, for Linux and Solaris in the JDK 6 release.

The SwingSet demo screenshots (see Figures 1 and 2) were made on Windows XP and an early beta of Windows Vista.

For those developers who want a unique look to their application, there are many options. Synth makes it much easier to create a custom look and feel from scratch. You can also buy or use a third-party look such as Plastic or Alloy. And, of course, you can write your own Swing look and feel from scratch. If those approaches sound like too much work, it's easy to customize individual component classes, to change their behavior, add animation, whatever you can think of.

The Napkin Look and Feel is an extreme example of what's possible with a custom Swing look and feel. It was written so that early versions of desktop applications with provisional GUIs wouldn't give anyone the mistaken impression that the application was actually finished. It's also pretty funny (see Figure 3).

*There is a lot of interest in technologies like AJAX and Web 2.0 that enhance the browser experience to be a richer user interface experience. Do you see this as a threat to Swing?*



Figure 4 SWT lead Steve Northover and sometime Swing engineer Tim Boudreau (facing camera) singing Kumbaya at this year's EclipseCon (Although Steve swears it was Led Zeppelin - Bd.)



Figure 5 The Aerith application lobby

It wouldn't be honest to say that we don't feel somewhat threatened by AJAX or any of the other desktop client technologies that are competing for developers' attention, tools support, and recognition in the finer trade and technical periodicals. On the other hand, competition is healthy and the growth in user's expectations of Web-based desktop application plays to our strengths.

Some of the advantages of Swing applications are:

- Can run online or offline.
- Native look and feel if you want it.
- High-performance 2D and 3D graphics.
- Support for visualization and direct manipulation of large data sets. No paging through tables of results or waiting for the server to compute the presentation of the next bit of data.
- Full access to the desktop (for signed apps).
- The Java software platform is deep and wide. Threads, collections, security, database, XML – there's an entire article here.
- Integration with the client OS desktop, like drag-and-drop with other applications.
- Hundreds of component libraries, lots of tools, large robust Java developer community.

*Some companies moved away from desktop apps toward Web-based frameworks simply for ease of deployment. What is going on in JNLP to help people who want rich desktop applications with a simple one-touch deployment that has the same ease of use as a browser URL?*

Easier deployment continues to be a focus for the desktop group.

One aspect of simplifying deployment is making sure that, in most cases, Java SE is already installed on users' PCs. To that end, we have bundling agreements with the top ten (and many more) PC OEMs, and we're getting new OEMs signed up all the time. At this point the majority of new PCs come with Java SE 5 pre-installed. We're also seeing phenomenal download rates: in the last 12 months we've served up about a quarter of a billion downloads. That's a lot.

For platforms that do not already have Java installed, we continue work on easy-to-use mechanisms for installing Java SE and for launching Java SE applications. Back in June, we published an article about the auto-install features that are part of JDK 5 (<http://java.sun.com/developer/technicalArticles/JavaLP/javawebstart/AutoInstall.html>). The article discusses an ActiveX component for Windows/IE that detects whether Java is installed, checks it against a desired version, installs Java if necessary, and launches a given Java Web Start application. There are also various JavaScript mechanisms that can be used to detect Java on browser platforms and help the user to install and launch Java as appropriate.

This is the model we'd like to work toward going forward: developers should be able to deploy their applications and applets in such a way that the browser can detect whether the user needs to install Java, and, if so, will perform that installation for them and launch the application.



Figure 7 Aerith trip report editor

We expect to develop and deliver modules and approaches to assist with this going forward; look for more articles and deployment components in the future.

*Both Microsoft and Adobe have implemented declarative GUI programming (XAML and MXML, respectively). Are there similar plans at Sun for the future Swing versions?*

There are some good open source languages that we've been keeping a close eye on. JAXX (<http://www.jaxxframework.org/>), in particular, is quite similar to Adobe's MXML - it is fully compiled, has nice data-binding features, and is styled using CSS. The biggest difference from MXML is that instead of using ActionScript and Flash, JAXX uses Java and Swing, so it's both faster and more powerful.

It's hard to say if Swing will ever directly incorporate a declarative UI language, but the open source solutions that exist today are definitely worth a look.

*How do you see the relationship between Swing and SWT both at the moment and moving forward?*

On a personal level the relationship between the two teams is friendly as can be plainly seen in the undoctored photo in Figure 4.

When SWT first debuted, we made some changes to ensure that Swing components would work within SWT applications; however, mixing two GUI toolkits in a single application is notoriously difficult. Over the years demand for a bridge has persisted within the Eclipse/RCP community, however, the technical problems haven't become



Figure 6 The Aerith 3D photo viewer

any simpler. Nevertheless the two development teams have settled into a peaceful coexistence.

*How important do you think IDE tools are for the adoption and usage of Swing?*

For the longest time we've held the line at, "We do the API, tool vendors write the tools." That's led to some embarrassments on our part in the form of untoolable APIs and

# Oracle Development Tools User Group

## IF YOU WORK WITH

- Oracle JDeveloper
- Oracle SQL Developer
- Oracle Designer
- PL/SQL
- Oracle Forms
- Business Intelligence, Data Warehousing, and ETL
- Oracle Reports
- XML
- Oracle Fusion Middleware
- J2EE and .NET
- Oracle Discoverer
- Toad
- Oracle Warehouse Builder
- Eclipse, NetBeans, and Visual Studio
- Oracle Application Express
- Other Development Tools

## YOU NEED ODTUG!

## MEMBER BENEFITS

- The quarterly ODTUG *Technical Journal*
- Exclusive access to our web site's "Members Only" area containing valuable utilities, code, tips and techniques, conference papers, presentations, and more
- Access to the most widely used e-mail lists for Oracle developers
- Discounts on our annual conferences featuring technical sessions by leading members of the user community, hands-on computer training, Oracle product managers, and panels of experts
- Discounts on several other Oracle-related events
- Online networking opportunities through participation in ODTUG SIGs
- A forum to provide feedback to Oracle Corporation

**The Only Oracle User Group  
Specifically for Developers  
by Developers**

Visit the NEW [www.odtug.com](http://www.odtug.com)





difficult or unusable tools; not good! XEmacs and vi are certainly enjoyable, but if you ever want more than rocket scientists to use the platform, you need compelling tools. Thankfully we've awakened to this fact.

For Java SE 6.0, we've worked closely with the NetBeans folks to address one of the hardest problems in Swing: layout. This partnership has resulted in a world-class tool, Matisse, that makes it trivial to create visually appealing layouts. With Matisse, you no longer need to know about layout managers, instead the user focuses on arranging the components in a way that is immediately familiar to nearly all developers.

We will continue to work closely with NetBeans and other tool vendors to make sure new APIs are toolable. The expert groups for 295 and 296 have numerous tool vendors on them, including companies such as Oracle, JetBrains, and Borland.

*There was a lot of interest at JavaOne in the timing framework and some of the very polished demos, especially the keynote flickr demo and the Extreme makeover e-mail client. Are these available for developers to download the code and see what was done under the covers?*

The code for the extreme demo e-mail clients will be made available later this year; look for announcements about that on javadesktop.org. Aerith, the Flickr/Google Maps demo shown during the keynote, was released online at aerith.dev.java.net. As you can see in Figures 5-7, Aerith is quite a looker.

We weren't able to release the Aerith demo quite as promptly as last year's SwingLabs JavaOne keynote demo. That one was actually released while the keynote presentation was underway. This year there was a bit of delay because the SwingLabs team wanted to clean up the code to make it a little easier for other developers to learn from it. Though it took more time, we think it was worth the wait. The source for many of the components that went into Aerith are available in their own projects including JOGL (Java bindings for Open GL at jogl.dev.java.net), the mapping component (swingx-ws.dev.java.net), and the animation framework (timingframework.dev.java.net).

**Chet Haase:** Did someone say animation? [Editors note: At this point the interview was interrupted by the appearance of a marching band led by (drum major) Chet Haase. It took a while to get the brass section to quiet down, and to make Chet stop leaping around and twirling his baton. Eventually we were able to resume the interview.]

The Timing Framework, in particular, is not specific to Aerith, but is instead a general framework that simplifies creating and running animations. The core facilities of java.util.Timer and javax.swing.Timer leave much to be desired in terms of the amount of work a developer must do to actually use animations in a Swing application. This probably helps explain why there are not many Swing applications that use animated effects. Hopefully, with the Timing Framework, we can help change that and make it easier for

Swing developers to write dynamic applications that use animated effects.

The framework is still in flux, but the current project site has working code that we encourage you to check out and use. In the meantime, we are refining the API and capabilities to make it more powerful as well as more easy to use.

*If you could re-do Swing all over again, what would you do differently and why?*

Rather than just provide one answer to this question, we decided to give every person who'd worked on the Swing project at Sun a shot at it. The first response rolled in less than five minutes:

**James Gosling:**

- Delete half the methods. The "747 cockpit control" thing got way out of control (i.e., just do the things that actually matter, not all the things that someone thinks could possibly matter in some obscure case)
- Rethink events and listeners; e.g., there are way too many addXxxListener() methods.
- Make JOGL part of SE and integrate it more cleanly with 2D.
- Get rid of 1.0 compatibility!
- Get rid of deprecations!
- Building new L&Fs is too hard. This is one of the areas where it feels like we overgeneralized
- Fold in a pile more standard Choosers (take all of the cool stuff in javadesktop.org and fold it into SE).

Ouch. Fortunately, not 10 minutes later, another response rolled in from left field:

**Chet Haase:**

I would have sold SUNW at \$63.

Over the next two weeks, many more responses rolled in. Some pithy, some painful, a few funny, and a rare gem that was a nice combination. Here's a sampling:

**Steve Wilson (original Metal L&F author, now a Sun Microsystems Vice President):**

We should have written (and shipped) our own GUI builder in Swing in tandem with building the toolkit - which we didn't do because we were too worried about hurting Symantec and Borland's Java tools businesses. Boy, in retrospect, it's good thing we didn't hurt those! Oh, wait...

**Phil Milne (original author of JTable among many other contributions):**

My first memory of life on the Swing team was the moment when I uncovered the 20 classes that performed the role of the Button class in NeXT's appkit. Soon the mystery was solved when kind colleagues patiently explained Swing's architecture and how it laughed in the face of common technical problems such as your Apple desktop spontaneously turning into a PC. Imagine my surprise when I checked my diary the other day to discover that the number

**Hans Muller** is the CTO for Sun's Desktop division. He's been at Sun for over 15 years and has been involved with desktop GUI work of one kind or another for nearly all of that time. He's been involved with the Java project since its earliest days and led the Swing team and later all of the client Java work at Sun.  
hans.muller@sun.com

of times this had actually happened to me over the last eight years was zero! A more AppKit-like API then: smaller, lighter, faster and designed to work well with a GUI builder.

*Romain Guy (Synth L&F developer, the mad genius behind countless demos, etc...):*

Real quick cause I'm leaving for China in a few hours. I would make Swing entirely vector-based (i.e., resolution independent) with as many built-in effects and animation support as possible.

*Rick Levenson (first engineering director for the Java SE desktop groups):*

If I could re-do Swing all over again, I would make sure this time to actually contribute something more meaningful than just my bouncing head...(see Figure 8).

*Arnaud Weber (all-around early Swing developer and performance tuner):*

If I could re-do Swing all over again, I would spend more than one week on ComboBox. Seriously, I believe I would focus a lot more on building a real application framework instead of simply delivering a graphical user interface toolkit. I would also greatly simplify all the type-safe listeners and replace them with more generic notifications. In my opinion, all these small inner classes in the client code not only have a performance impact but also make everything more complicated than necessary.

*Scott Violet (long time Swing tech lead and architect):*

I wish I knew how to spell; every time I see insertAtBoundry (HTMLEditorKit.InsertHTMLTextAction) and insureRowContinuity (DefaultTreeSelection) I feel ill.

This last response caused current Swing engineer Alexander Potochkin to bring up the following backwards-compatibility embarrassment that we'd all prefer didn't exist:

```
// from java.awt.event.KeyEvent
public static final int VK_SEPARATOR = 0x6C;
public static final int VK_SEPARATOR = VK_SEPARATOR;
```

Of course it didn't end there. There were followups and escalations; the e-mail thread is probably still sputtering away as you read this, but we'll spare you.

*What are the most important things that are going to occur in Swing over the next five to 10 years?*

In five to 10 years, in addition to tooling around in ethanol-powered flying cars, we expect users to be interacting with computers via datagloves and contact lenses with LCD displays. By then, the large glass panels Tom Cruise bossed around in "Minority Report" will look as old fashioned as teletypes and paper tape.

Seriously though, it's pretty clear we will need to continue to work on making Swing, and the Java platform as a whole, easier to use. We've begun work down that path with JSRs 295 and 296; there are many other things that could be done.

We've all been impressed by how popular AJAX has become in such a short time. At least part of its popularity stems from the relative ease with which you can reuse what others have done in novel ways. For example, we've all come across Web sites that make

Advertiser	URL	Phone	Page
Quest	www.quest.com/JavaCode		2
Altova	www.altova.com	978-816-1600	4
IBM	ibm.com/takebackcontrol/flexible		7
Intersystems	www.InterSystems.com/Cache21P	617-621-0600	13
OPNET	www.opnet.com/pinpoint	240-497-3000	15
SAP TECHED	www.sapteched.com		17
Fiorano	www.fiorano.com/downloadsoa	800-663-3621	23
Cynergy	www.cynergysystems/thatsme		27
Backbase	www.backbase.com/jsf	866-800-8996	31
Tibco	www.tibco.com/mk/gi	800-420-8450	35
Instantiations	www.instantiations.com/rcpdeveloper	800-808-3737	43
RogueWave Software	www.roguewave.com/developer/downloads		47
Roaring Penguin	www.roaringpenguin.com	613-213-6599	49
Real World Flex Seminar	www.flexseminar.com	201-802-3020	53
AJAXWorld Conference & Expo	www.AjaxWorldExpo.com	201-802-3020	54, 55
Northwoods	www.nwoods.com	800-434-9820	57
LinuxWorld Conference & Expo	www.LinuxWorldExpo.com		61
SoftwareFX	www.softwarefx.com		63
Parasoft	www.parasoft.com/JDImagazine	888-305-0041x3501	64

**General Conditions:** The Publisher reserves the right to refuse any advertising not meeting the standards that are set to protect the high editorial quality of *Java Developer's Journal*. All advertising is subject to approval by the Publisher. The Publisher assumes no liability for any costs or damages incurred if for any reason the Publisher fails to publish an advertisement. In no event shall the Publisher be liable for any costs or damages in excess of the cost of the advertisement as a result of a mistake in the advertisement or for any other reason. The Advertiser is fully responsible for all financial liability and terms of the contract executed by the agents or agencies who are acting on behalf of the Advertiser. Conditions set in this document (except the rates) are subject to change by the Publisher without notice. No conditions other than those set forth in this "General Conditions Document" shall be binding upon the Publisher. Advertisers (and their agencies) are fully responsible for the content of their advertisements printed in *Java Developer's Journal*. Advertisements are to be printed at the discretion of the Publisher. This discretion includes the positioning of the advertisement, except for "preferred positions" described in the rate table. Cancellations and changes to advertisements must be made in writing before the closing date. "Publisher" in this "General Conditions Document" refers to SYS-CON Publications, Inc.

This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions.

something new from Google maps. We've got to do the same for Java and Swing. We need to make it trivial for newbies to get apps up and running in hours, not days or weeks.

A key part of this change will be tools. No one wants to read thousand page tomes anymore! Developers want to start up a tool and in a matter of hours have something compelling that's just a button press away from being deployed on the Internet. That is what we have to strive for.

We recently saw a slide that showed how growth in GPUs is outpacing growth of other components of a computer. We'll continue to look for ways to make better use of GPUs in applications. You can see this in Vista and Leopard - rich animation and graphic effects in nearly every app. While we plan on doing work in this area for Java SE 7, I suspect we're just seeing the tip of the iceberg as far as what is possible.

Another desktop trend that's going to have a big impact on what desktop applications can do is the number of CPU cores in typical machines. Today, machines with two cores are common. Given the ruthless more/bigger competition in the PC business, we can probably expect desktops with dozens of cores in 5-10 years. This is already reality in the server world and Java's extensive support for threaded programming will serve us well. Making it possible, even easy, for Swing app developers to harness all of that parallel

power will be a challenge for us in the years ahead.

Our developer community will continue to be an important ingredient in whatever we do. You, as the end developer, know what is best for your problem domain. By working closely together we can make sure whatever we develop addresses both needs. The JDK development community has been tremendously successful in this area, and we continue to look for new ways to make it better.

*What's the best place to learn more about what's happening in Swing and to stay abreast of all the new features?*

For official documentation and articles, please see the Java Desktop Web pages on java.sun.com: <http://java.sun.com/javase/technologies/desktop>.

You can not only stay abreast of new features but can actually participate directly in the future of Swing by looking at and joining the SwingLabs project on java.net: <http://www.swinglabs.org>.

We'd also like to urge you to read javadesktop.org daily, where you'll find several Java Desktop luminaries (from both inside and outside Sun) blogging, as well as newsfeeds, discussion forums, and product announcements by developers using Java desktop technology: <http://www.javadesktop.org>.

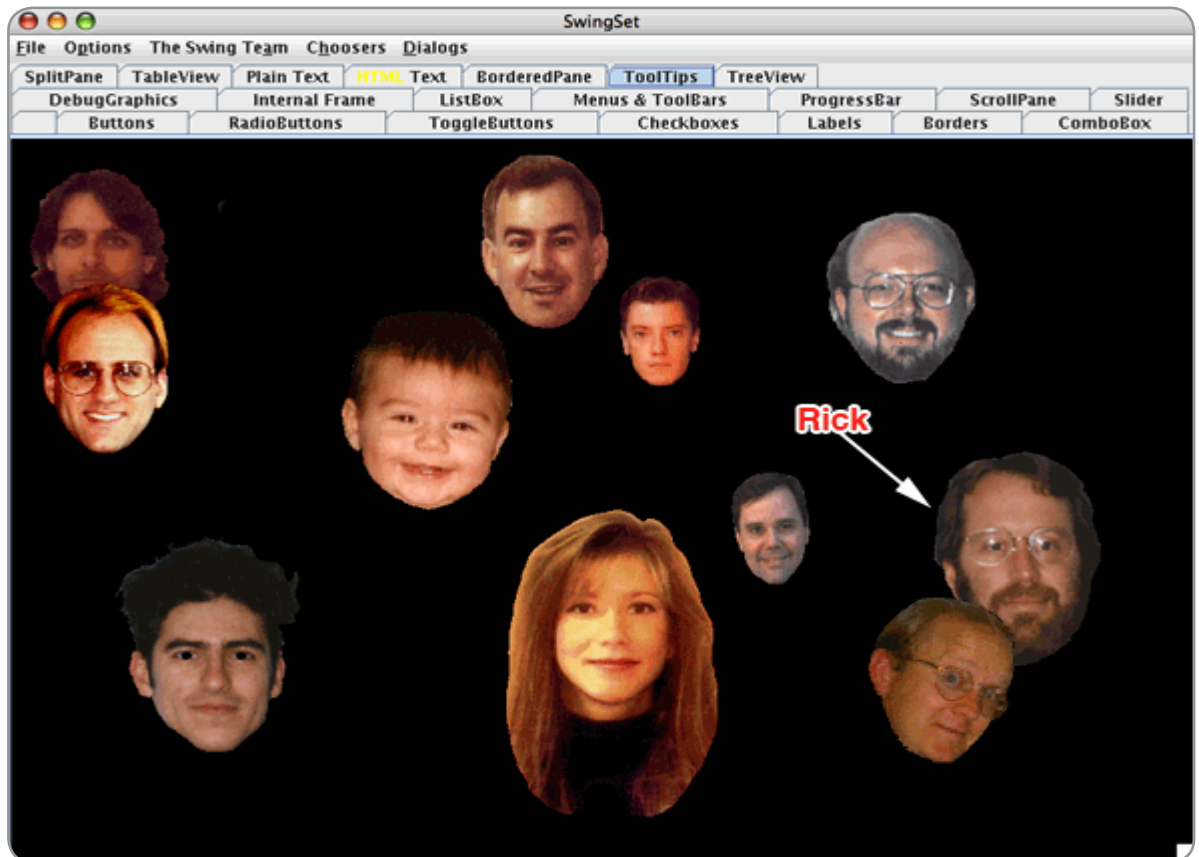


Figure 8 Bouncing heads of the Swing development team

# #1 Rated AJAX and Rich Internet Application toolkit\* TIBCO General Interface

Build 100% Pure Browser Rich Internet Apps with TIBCO General Interface™

“TIBCO General Interface is a friendly, capable toolkit for building sophisticated JavaScript Web applications that run in a browser.”



Download today  
at <http://www.tibco.com/mk/gi>

 **TIBCO**®  
The Power of Now®



# *Rich Internet Applications: AJAX,*

[www.AjaxWorldExpo.com](http://www.AjaxWorldExpo.com)

# AJAXWORLD™ CONFERENCE & EXPO

## *SANTA CLARA SILICON VALLEY*

**SYS-CON Events is proud to announce the first-ever  
AjaxWorld Conference & Expo 2006!**

**The world-beating Conference program will provide developers and IT managers alike  
with comprehensive information and insight into the biggest paradigm shift in website design,  
development, and deployment since the invention of the World Wide Web itself a decade ago.**

The terms on everyone's lips this year include "AJAX," "Web 2.0" and "Rich Internet Applications." All of these themes play an integral role at AjaxWorld. So, anyone involved with business-critical web applications that recognize the importance of the user experience needs to attend this uniquely timely conference – especially the web designers and developers building those experiences, and those who manage them.

### **CALL FOR PAPERS NOW OPEN!**

We are interested in receiving original speaking proposals for this event from i-Technology professionals. Speakers will be chosen from the co-existing worlds of both commercial software and open source. Delegates will be interested learn about a wide range of RIA topics that can help them achieve business value.

# Flash, Web 2.0 and Beyond...

REGISTER TODAY AND SAVE!

→ **October 3-4, 2006**

→ **Santa Clara Convention Center**  
Hyatt Regency Silicon Valley  
Santa Clara, CA

→ **To Register**

Call 201-802-3020 or  
Visit [www.AjaxWorldExpo.com](http://www.AjaxWorldExpo.com)

→ **May 7-8, 2007**

First International AjaxWorld Europe  
Amsterdam, Netherlands

**“Over the two information-packed days, delegates will receive four days’ worth of education!”**

**Early Bird\***

(Register Before August 31, 2006)  
..... **\$1,495\*\***

See website or call for group discounts

**Special Discounts\***

(Register a Second Person)  
..... **\$1,395\*\***

See website or call for group discounts

**(5 Delegates from same Company)**

..... **\$1,295/ea.\*\***

See website or call for group discounts

**On-Demand Online Access**

(Any Event)  
..... **\$695**

\* Golden Pass access includes Breakfast, Lunch and Coffee Breaks, Conference T-Shirt, Collectible Lap-Top Bag and Complete On-Demand Archives of sessions in 7 DVDs!

\*\*OFFER SUBJECT TO CHANGE WITHOUT NOTICE, PLEASE SEE WEBSITE FOR UP-TO-DATE PRICING

**“It Was The Best AJAX Education Opportunity Anywhere in the World!”** —John Hamilton

## Topics Include...

### Themes:

- > Improving Web-based Customer Interaction
- > AJAX for the Enterprise
- > RIA Best Practices
- > Web 2.0 – Why Does It Matter?
- > Emerging Standards
- > Open Source RIA Libraries
- > Leveraging Streaming Video

### Technologies:

- > AJAX
- > The Flash Platform
- > The Flex 2 Framework & Flex Builder 2
- > Microsoft’s approaches: ASPNET, Atlas, XAML with Avalon
- > JackBe, openLaszlo
- > JavaServer Faces and AJAX
- > Nexaweb
- > TIBCO General Interface

### Verticals:

- > Education
- > Transport
- > Retail
- > Entertainment
- > Financial Sector
- > Homeland Security

**GROUP DISCOUNTS AVAILABLE:**  
— 5 Delegates from Same Company —  
for only \$995 (each)  
— Register a Second Person —  
for only \$1195

**Hurry! Limited Seating  
This Conference Will Sell-Out!**



**LIVE SIMULCAST!**  
AROUND THE WORLD ON SYS-CON.TV

Receive **FREE** WebCast Archives of Entire Conference!

The best news for this year’s conference delegates is that your “Golden Pass” registration now gives you full access to all conference sessions. We will mail you the complete content from all the conference sessions in seven convenient DVDs after the live event takes place.



▶ This on-demand archives set is sold separately for \$995

HYATT  
HYATT REGENCY SILICON VALLEY



For more great events visit [www.EVENTS.SYS-CON.com](http://www.EVENTS.SYS-CON.com)

VISIT [WWW.AJAXWORLDEXPO.COM](http://WWW.AJAXWORLDEXPO.COM) FOR THE MOST COMPLETE UP-TO-DATE INFORMATION



# What to Do If Your Code Has No Tests

by Ted Husted

*Tools that practically write tests by themselves*

**W**hen MailReader – an example application bundled with the Struts Action framework – was created six years ago, most Java developers had yet to discover unit testing. Consequently neither the Struts framework nor the MailReader were created test-first. Since then, we’ve bolted a few unit tests onto the Action framework, but the MailReader for Struts 1.2 still has no developer tests at all.

Over the last 10 years, testing has become more popular with developers. Much of this popularity can be attributed to the JUnit framework: a simple effective tool for many teams. But JUnit isn’t enough. It makes writing tests easier, but we still have to write the tests. Many of us despair at the notion of writing even more code to test the application code we’re already writing. What we need are tools that can write most of the tests for us.

## Software Agitation

Software Agitation exercises and analyzes Java binary code and generates observations about how the code behaves. Developers can quickly create tests based on these observations, often without writing any new code. This article walks through using Agitator to create unit tests for the Struts MailReader application.

The flagship software agitation tool is Agitator by Agitar Software. The tool rapidly creates observations of code behavior, and helps the developer determine if the code is working as expected or see if Agitation has revealed unexpected behavior.

From within the Eclipse IDE, we can promote a valid observation to a unit test, or we can trace through the code to change the behavior. If the application code is valid, but the range of acceptable values needs to be adjusted to demonstrate correct be-

havior, we can assign custom factories to a parameter. Along with factories, Agitar provides for automatic “Domain Experts” that we can use to test code peculiar to our own API or to test code peculiar to frameworks like Struts.

## Getting Started

After launching Agitator’s version of Eclipse, we can create a new project using the “Java Project with Agitation” template. For this project, we point Eclipse at the root of the Struts 1.2 source tree. We have a library of JARs for the Struts 1.2.8 distribution, which are easy to add to the Eclipse Build Path as “External JARs” along with a reference to the servlet JAR for Java 2.3.

The one other thing we have to do is unselect irrelevant packages as source folders. In Struts 1.2, the MailReader source is mixed in with packages for the rest of the framework distribution. In the end, we have two source folders: `src/example` and `web/example`.

We can open the “Agitator” view from the menu bar, and “Agitate” the package containing the five MailReader business classes [`org.apache.struts.webapp.example.memory`].

A New York minute later, Agitator has done its thing. Three classes are in good shape, with 100% code coverage. Two of the five classes were flagged with warning symbols. The `MemoryDatabasePlugIn` class weighed in at 70% coverage. The key class, `MemoryUserDatabase`, had 38% coverage. The code coverage for each is shown in Table 1.

Let’s start with the low-hanging fruit and review the three classes with 100% coverage: `MemorySubscription`, `MemoryUser` and `TestUserDatabase`.

`MemorySubscription` and `MemoryUser` represent database entities. Being standard JavaBeans, these classes were easy for Agitator to test. Each of the JavaBean properties has a standard unit test to ensure that the field is set

by the parameter.

The only two methods lacking tests are the constructor and `toString`. The constructor is simple, but, still, Agitator has generated some essential tests like:

```

this.getHost() == host
this.getUser() == user
Table 2 - MemorySubscription Constructor Method
public MemorySubscription(MemoryUser
user, String host) {
    super();
    this.user = user;
    this.host = host;
}

```

We can mark these as unit tests to prevent simple silly mistakes like assigning a parameter back to itself.

The `toString` method creates a textual representation of the class. The method looks hard to test with a known set of input data. For now, we can change the method’s property to “Exclude from testing.”

```

public String toString() {
    StringBuffer sb = new StringBuffer
("<subscription host=\""");
    sb.append(host);
// ...
    if (username != null) {
        sb.append(" username=\""");
        sb.append(username);
        sb.append("\n");
    }
    sb.append(">");
    return (sb.toString());
}

```

Reviewing observations for the other classes and methods, we find several other assertion candidates. Each of these candidates corresponds to assertions that we might have made in a conventional JUnit test. For example, setting the pathname prop-

**Ted Husted** (<http://husted.com/ted/>) is a software engineer and an active member of several Open Source projects hosted by the Apache Software Foundation, including Struts, iBATIS, and MyFaces. His books include *JUnit in Action*, *Struts in Action*, and *Professional JSP Site Design*. He also is a consultant for Agitar Software. Check out his blog at <http://husted.com/ted/blog/>.

erty also sets the pathnameNew and pathnameOld fields. We didn't have to express that fact to Agitator. On its own, the software observed that

```
@EQUALS( this.pathnameOld, "database.xml.
old" )
```

All we have to do is confirm that the observation is an assertion that we should test. If we were writing a JUnit test, we'd have typed-out code like:

```
assertEquals( this.pathnameOld, "database.
xml.old" );
```

With Agitator in play, we just point and click.

Not bad. After only a few minutes of clicking around, we have almost 80 test points. Perhaps most important, these test points will automatically evolve with the code – something that hand-coded tests can never do.

Now, what's the problem with the other two classes that had less than 100% coverage?

Agitator displays a legend next to the lines in a class to show how often each line of code is being reached by Agitation, or if the line was even reached in the first place. In the case of MemoryUserDatabase, we can see that there are a lot of red lines after an input-output call, indicating that the code isn't being reached. Clicking through, it's easy to see why many of the Exceptions were being thrown: "File Not Found."

The "Memory" implementation of the MailReader data access object loads a list of Users and their e-mail Subscriptions from an XML document into an object graph stored in main memory. (Hence, the package name.) The file wasn't found because Agitator had no way of guessing the right file name. For now, I mark the problematic classes or methods "Exclude from testing" – at least until we can learn a bit more about Agitator.

Excluding the eight input-output members lowered the overall test coverage. But, even so, in only a few minutes, we were able crank up Agitator for the first time, create over 80 test points, and yield a test coverage of around 40%.

### Agitating Struts

The Agitator website hosts a 47-minute "webinar" on its Struts Expert. I watch this and skim the documenta-

Class	Initial Coverage	Description
MemorySubscription	100%	JavaBean representing a subscription to an e-mail list
MemoryUser	100%	JavaBean representing a user with zero or more subscriptions
TestUserDatabase	100%	A UserDatabase implementation designed to test runtime exceptions
MemoryDatabasePlugin	70%	An adapter that loads a MemoryDatabase from an XML document
MemoryDatabase	38%	An implementation of a UserDatabase interface that stores the list of users and subscriptions in memory

Table 1 Code Coverage

tion.

Now that we've had a taste of Domain Experts, let's put the Struts Expert through its paces.

### LogonAction

The webinar mentioned the standard Struts Expert, which can be found and enabled on Agitator/Plugin Experts menu. Eclipse is not displaying any red marks, so the code seems to be compiling. Let the Struts Agitation begin!

After Agitating the MailReader code base with the Struts Expert enabled, a number of red marks popped up in the Package Explorer. Drilling down, we find error icons next to the Action execute methods. The pop-up hint explains that the Struts configuration can't be found. Meanwhile, the Console view contains several warnings that a ServletContext can't be found either.

Returning to the Agitator menu, we find the likely item "Create J2EE Environment." A wizard leads us through creating a default environment, and even includes a "Test" button so we can check our work.

Now that we have a J2EE environment, for good measure, we pop back to the Plugin Experts menu and enable the J2EE expert too. After another Agitation, there are still red marks, but the messages are functional rather than systemic, with remarks like "Coverage failed" and "Outcome failed".

Opening up LogonAction, we find the Struts We already have 67% coverage, and the execute method has partitions for the various Struts outcomes: registration, logoff, logon, success, and welcome. Several class invariants were generated, but only "this.getServlet() != null" looks like a worthwhile assertion.




Focusing on the execute method, only two of the four

result partitions would ever happen, so we mark the global forwards "registration," "logoff," and "welcome" as unexpected. For a normal return, only the observation "@RETURN!=!null" seems like a worthy assertion.

However, all trials returned logon and none succeeded. Looking at the code, we can see that a User object isn't being found, and so all the trials are being returned to the logon outcome. This makes perfect sense; we just have to provide a valid User object to some of the trials as we saw done in the webinar. In the webinar, an omniscient architect had already provided a session factory that created a User object. This being real life, we'll have to roll our own.


The execute method calls a helper to fetch the User object. Looking at

## Build interactive diagrams easier than you ever imagined

**C**reate custom interactive diagrams, network editors, workflows, flowcharts and design tools. For web servers or local applications. It's easy-to-use and very flexible. We're the first developer of diagramming components and still the best. Find out for yourself: download our FREE fully functional evaluation kit, with full support at [www.nwoods.com](http://www.nwoods.com).

**FREE Download With Full Support!**



Interactive diagram components  
[www.nwoods.com](http://www.nwoods.com)



the code coverage panel for the `getUser` helper method, we can see that the database parameter is usually null, and when it's not, the User object is null. As a result, the code always returns a message, and the execute method always forwards to `logon` instead of success.

Hmmm. Why is the database parameter ever not null? We haven't set up a session factory for it and the methods that could read it from a file have been disabled. What gives?

True, right now, when we call `getUser` from the execute method, the database parameter would always be null, but Agitator also tests each method in isolation. When we ran a full Agitation against the code base, Agitator also called `getUser` itself, and used its own default factory to create the database parameter. When Agitator called `getUser`, it mixed up the calls null with default instantiations of `MemoryUserDase` and `TestUserDatabase`. We didn't tell Agitator to do that. This is the automatic default behavior we get out of the box.

We already have a `TestUserDatabase` object that exercises exception handling. To that we can add a static `getTestUserDatabase` method that will return an instance of `TestUserDatabase` populated with User and Subscription objects. Creating the method means a bit more coding, but not so much, and now we'll be able to test the product independently of the XML document parsing.

Once the method is coded, we can right-click on the Factory Assignment view, choose "Assign a Factory form a Constructor or Method," and specify our new method.

The code under test does validate that the database is not null, so we should pass some nulls to exercise that code. A simple solution is to leave the original factory running, but specify a weight of 1, and change the weight on our own factory to 9. This way, we will get some nulls and empty databases, but most often we will get a copy of our seed database.

After Agitating, we have more coverage, but we're still ending up with a null User on every trial. Checking the Snapshot, it's easy to see that the username and password parameters need factories too. When we right-click on the factory assignment for username, this time we're offered the chance to "Assign a list of values" to our string parameter. The values item leads directly to the Factory Settings dialog, where we can quickly add a couple of usernames. We repeat the process for password, so that the first items on each list are valid username and password credentials. The factories will mix these entries up, so that we get a spread of valid and invalid logon attempts. Since the code doesn't validate the username and password variables, we disable the default factories by setting the weight to zero.

With these changes, we now have 100% coverage on the `getUser` helper, but from the Outcomes view we can see that some trials are still throwing exceptions. Checking the Observations, we see that every time an Exception is thrown errors is null. The API contract for this sub-routine implies that errors should never be null.

We make that change to the default setting for the errors parameter and the

unexpected Exceptions disappear. To seal the contract, we promote the Observation "`@PRE(errors.messages) != null`" to an Assertion. Now if the contract is ever broken due to future code modifications, Agitator will mark the `getUser` method with a red mark and exclaim "errors == null."

If we Agitate the `LogonAction` execute method, we see that `getUser` is still returning errors each time. Of course this is because, in this case, the database, username, and password values are still coming from the default factories. As shown in the webinar, we promote our new factories to package scope so that we can use them with other methods.

Under the Factory Assignments for `LogonExecute`, we find that the Expert has correctly wired a `LogonForm` factory to the "form" parameter. Problem is we need the factory to include both correct and incorrect credentials. After some trial and error, we manage to replace the default factories with new Form Property Factories that return usernames and passwords from the lists we created for `getUser` and promoted to package scope.

Note: When creating factories be sure to scroll down and review all the available fields. For example, the last field on the Form Property Factory form is `classname`, which must be set to `org.apache.struts.config.FormPropertyConfig`.

One red mark remains. Agitator noticed that the `getUser` helper can throw an `ExpiredPasswordException`, but none of our trials threw that exception. As it happens, the `TestUserDatabase` class is designed to throw that exception if a certain username value is passed, so all we need to do is add that magic value to our factory. One last Agitation, and `LogonAction` is clear. The code coverage for the class is not 100% mainly because the logging statements aren't being reached, but all the domain code is being exercised.

Using the Agitator, we've gone from a standing start to having solid code coverage and a series of tests. We haven't written any JUnit code, but with Agitator's assertions we have a set of test assets that will automatically evolve as code is updated. Looking around in the manuals, it becomes clear that if somebody decided to write JUnit tests, those would work in tandem with Agitators and show up in the same summary statistics.

Agitator isn't a bad way to answer the question, "How are we going to create tests for our code base?"

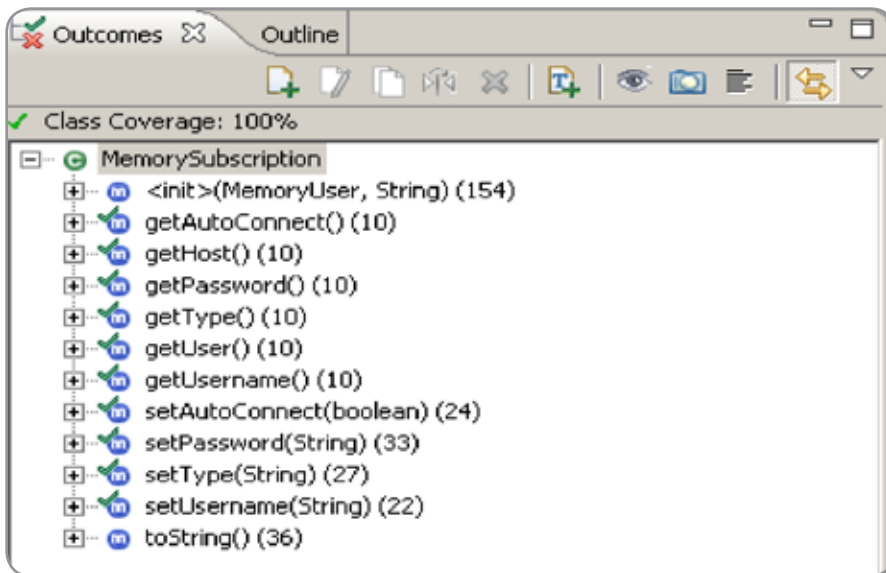


Figure 1

# Welcome to the Future

**REGISTER NOW!**  
[www.iTVcon.com](http://www.iTVcon.com)

CALL FOR PAPERS NOW OPEN!

**LIVE SIMULCAST!**  
AROUND THE WORLD ON [SYS-CON.TV](http://SYS-CON.TV)

# of Video on the Web!

**iTVCON.COM**  
INTERNET TV CONFERENCE & EXPO 2006

Coming in 2006 to New York City!

“Internet TV is wide open, it is global, and in true ‘Web 2.0’ spirit it is a direct-to-consumer opportunity!”



**For More Information, Call 201-802-3023  
or Email [itvcon@sys-con.com](mailto:itvcon@sys-con.com)**

## Welcome to the Future!

Did you already purchase your “.tv” domain name?  
You can’t afford not to add Internet TV to your Website in 2006!

2005 was the year of streaming video and the birth of **Internet TV**, the long-awaited convergence of television and the Internet. Now that broadband is available to more than 100 million households worldwide, every corporate Website and every media company must now provide video content to remain competitive, not to mention live and interactive video Webinars and on-demand Webcasts.

20 years ago the advent of desktop publishing tools opened the doors for the creation of some of today’s well-known traditional print media companies as well as revolutionized corporate print communications. Today, with maturing digital video production, the advent of fully featured PVRs, and significant advances in streaming video technologies, **Internet TV** is here to stay and grow and will be a critical part of every Website and every business in the years to come.

It will also very rapidly become a huge challenge to network and cable television stations: **Internet TV** is about to change forever the \$300BN television industry, too.

The Internet killed most of print media (even though many publishers don’t realize it yet), Google killed traditional advertising models, and **Internet TV** will revolutionize television the way we watch it today. You need to be part of this change!

**Jeremy Geelan**  
Conference Chair, [iTVCon.com](http://iTVCon.com)  
[jeremy@sys-con.com](mailto:jeremy@sys-con.com)

PRODUCED BY  
**SYS-CON**  
EVENTS

### List of Topics:

- > Advertising Models for Video-on-demand (VOD)
- > Internet TV Commercials
- > Mastering Adobe Flash Video
- > How to Harness Open Media Formats (DVB, etc)
- > Multicasting
- > Extending Internet TV to Windows CE-based Devices
- > Live Polling During Webcasts
- > Video Press Releases
- > Pay-Per-View
- > Screencasting
- > Video Search & Search Optimization
- > Syndication of Video Assets
- > V-Blogs & Videoblogging
- > Choosing Your PVR
- > Product Placement in Video Content
- > UK Perspective: BBC’s “Dirac Project”
- > Case Study: SuperSun, Hong Kong

- |                |  |
|----------------|--|
| <b>Track 1</b> | Corporate marketing, advertising, product and brand managers   |
| <b>Track 2</b> | Software programmers, developers, Website owners and operators   |
| <b>Track 3</b> | Advertising agencies, advertisers and video content producers  |
| <b>Track 4</b> | Print and online content providers, representatives from traditional media companies, print and online magazine and newspaper publishers, network and cable television business managers |

# Getting Serious About PDF Content Integration

*An introduction to enterprise-class PDF text and metadata extraction*

by Chas Emerick

The PDF file format has become the gold standard of document distribution and archiving. It's therefore virtually certain that data critical to your organization is sitting quietly in PDF documents somewhere. This situation means you have to get serious about integrating PDF content into your applications – taking shortcuts in this area and not finding or leveraging that mission-critical data can lead to millions of dollars in lost sales and/or similar levels of increased costs, compliance difficulties, or liability entanglements. It's time to use a high-performance PDF component that will yield accurate text and metadata extracts suitable for use with your existing search, content management, text analysis/mining, CRM, or other system(s).

However, the PDF file format is complex and wasn't designed for content extraction. So, any PDF library that doesn't specialize in content extraction is likely to exhibit various undesirable traits:

- Poor performance or performance degradation in high-volume environments
- Poor text extract accuracy
- Incomplete PDF file format support
- Lack of or limited support for extracting Unicode text, including Chinese, Japanese, and Korean text
- A complicated API that requires knowing the PDF file format
- Lack of any tools for identifying and converting unstructured data

If the PDF library you're using exhibits any of these problems then it's time to upgrade to an enterprise-class component that specializes in content extraction. And if your application doesn't leverage PDF content then it makes sense to skip the training wheels and use the best tool for the job from the start.

PDFTextStream fits the bill either way.

A pure Java library (also available for .NET and Python), PDFTextStream specializes in extracting text and metadata from PDF documents. Because of its focus, PDFTextStream has none of the downside all too

common when using a general-purpose PDF library for content extraction purposes.

This introduction will cover just a few of the use cases where PDFTextStream's focus on content extraction yields significant value.

## Simple/Powerful Text Extraction

PDF documents specify their text content a character at a time without any indication of each page's physical layout (such as lines, paragraphs, columns, tables, etc.). Thankfully, PDFTextStream automatically derives these structures for every page it extracts using state-of-the-art page segmentation and read-ordering processes – similar to how an OCR application derives the structure of a scanned document. And thankfully again, this accuracy doesn't come at the expense of speed or ease of use.

Now for some code. As you can see, extracting text using PDFTextStream is super-simple:

```
StringBuffer pdfText = new StringBuffer(1024);
com.snowtide.pdf.OutputTarget tgt = new
com.snowtide.pdf.OutputTarget(pdfText);
PDFTextStream stream = new
PDFTextStream(pdfFile);
stream.pipe(tgt);
stream.close();
```

The full text of the PDF file is now available in the pdfText StringBuffer.

OutputTarget is the default implementation of the com.snowtide.pdf.OutputHandler interface, which can be thought of as a SAX interface for PDFTextStream document model events. These events are generated any time an OutputHandler is passed into a pipe(OutputHandler) function, which is available on many document model objects as well (com.snowtide.pdf.Page, com.snowtide.pdf.layout.Block, and com.snowtide.pdf.layout.Line).

OutputTarget's primary purpose is to provide a straightforward way to direct

extracted text to a StringBuffer or a java.io.Writer. Further, OutputTarget passes through PDFTextStream's default text layout: content is in the proper semantic order, columns of text are separated, and rotated text is normalized and grouped in reasonable ways. This is really important if the PDF text you're extracting is going to be used as input to a semantically sensitive process, such as text mining or search engine indexing.

There are many OutputHandler implementations included with PDFTextStream, each of which interprets and processes PDF text events differently. If none of them meet your application's needs, you can easily write your own.

## Unicode Text Extraction

Today's global economy demands that your application be world-ready, in any major language. Thankfully, PDFTextStream always extracts text from PDF documents as Unicode (a perfect match for Java's consistent and thorough Unicode support). Further, PDFTextStream extracts Chinese, Japanese, and Korean (CJK) text from PDF documents without any performance penalties.

Nothing special needs to be done to enable these capabilities – they're always on, so you can use the simplest code and always get Unicode and CJK text out of your source PDF documents.

## Search Engine Integration

PDFTextStream was designed to be easily integrated into other applications, including content management systems, text mining processes, and, of course, search engines. A great example is its Lucene integration module, which produces Lucene documents using the content extracted from PDF files. Building a Lucene document that contains all of the text in a PDF file requires one line of code:

```
Document luceneDoc = com.snowtide.pdf.lucene.
PDFDocumentFactory.buildPDFDocument(pdfFile);
```

The contents of the Lucene document, including whether PDF document attributes (such as author's name, title, creation date, etc.) should be included, as well as the Lucene document's indexing, tokenizing, and storage parameters can all be customized (via `com.snowtide.pdf.lucene.DocumentFactoryConfig`).

Also of interest to those who work with search engines, `PDFTextStream` enables Web crawlers to source new URLs to retrieve from PDF documents – see [Enabling PDF Web Crawling](#) below.

## Metadata, Metadata Everywhere

Utilizing the metadata embedded in many PDF documents can add a great deal of value to your applications. `PDFTextStream` gives you easy access to the full world of PDF metadata:

- Document attributes (as a key/value Map or in Adobe XMP XML format)
- Document outline/bookmarks
- Acroform data – interactive form data
- PDF annotations (text notes, embedded URL links, etc.)

There's clearly a ton of metadata that you could work with; let's dig into a couple examples.

## The Bulk Metadata Import

Consider a scenario where you need to load PDF documents into a content management system. A common requirement would be for each document's author, title, and creation date to be imported as well. Let's retrieve those attributes:

```
PDFTextStream stream = new PDFTextStream(pdfFile);
Object author = stream.getAttribute(PDFTextStream.ATTR_AUTHOR);
Object title = stream.getAttribute(PDFTextStream.ATTR_TITLE);
Object createDtStr = stream.getAttribute(PDFTextStream.ATTR_CREATION_DATE);
Date createDt = null;
if (createDtStr != null && createDtStr instanceof String)
    createDt = PDFDateParser.parseDateString((String)createDtStr);
```

From here, you could easily add the metadata associated with each PDF document to the CMS. This code is straightforward, but there are some points worth noting:

- The `PDFTextStream` class provides a set of attribute name constants, making standard attribute lookups easy.
- The `getAttribute(String)` function returns an Object, not a String – this is because PDF files can technically specify attribute values of various types.
- PDF date strings have a standard format;

the `com.snowtide.pdf.PDFDateParser.parseDateString(String)` function can be used to convert PDF date Strings into java.util.Date objects.

## Enabling PDF Web Crawling

PDF documents can contain Internet URLs, but many Web crawlers don't look for and follow such links. Here, we'll retrieve the embedded PDF annotations that contain URL links, which could then be retrieved by a Web crawler.

```
PDFTextStream stream = new PDFTextStream(pdfFile);
List<Annotation> annots =
    stream.getAllAnnotations();
ArrayList<String> uriList = new
ArrayList<String>();
for (Annotation annot : annots) {
    if (annot instanceof com.snowtide.pdf.annot.
LinkAnnotation) {
        LinkAnnotation link =
(LinkAnnotation)annot;
        if (link.getLinkActionName().equals("URI"))
            uriList.add(link.getURI());
    }
}
```

This example will add all of the available URLs in the PDF document to the `uriList` `ArrayList`. The process is very simple: find all of the PDF annotations of type `com.snowtide.pdf.annot.LinkAnnotation`, and ignore any `LinkAnnotations` that do not have an "action name" of URI. There are a variety of link action names, each of which have different behaviors in a PDF viewer. Only URI `LinkAnnotations` contain a URL, which is retrieved using the `getURI()` function.

## Identifying and Converting Unstructured Data

Coping with "unstructured" data is a popular topic these days, mostly because:

- It's being recognized that unstructured data represents most of the data generated and received by most organizations
- Significant operational advantages can be achieved only if organizations can identify, convert, and harness the available unstructured data

Given that PDF documents are a primary vehicle for unstructured data, it's worth noting that `PDFTextStream` provides some tools to make extracting this data easier. The use of these tools is beyond the scope of this article, so please refer to the `PDFTextStream` documentation for details.

First, `PDFTextStream` provides a table API (`com.snowtide.pdf.layout.Table`) that represents the data of any table that `PDFTextStream` can detect while processing a PDF document.

This API can be used as the basis of a process that converts tabular data found in PDF documents into CSV or Excel files, or directly into database records.

Secondly, for broader unstructured data conversion purposes (or for tabular data that can't be detected automatically through its table API), `PDFTextStream` provides `VisualOutputTarget`, an `OutputHandler` implementation that renders PDF text to a `StringBuffer` or `java.io.Writer` while maintaining the visual layout of each page of text. This maintains the visual alignment of table columns and other textual elements, which makes text extracts retrieved using `VisualOutputTarget` ideal for input into downstream text analysis and mining tools.

## Conclusion: Enterprise-Class, Indeed

The term "enterprise-class" typically means that a component is robust – that it can take a beating and still keep going, while maintaining high performance levels.

That describes `PDFTextStream` quite well. It's feature-rich, it has a high degree of PDF file format support, and it's just plain fast: in extensive benchmarking (conducted by Snowtide Informatics and posted for review and verification on its Web site), `PDFTextStream` is shown to be 223% -1,141% faster than all other Java PDF libraries that are capable of text extraction. Even better, `PDFTextStream` clocks in as 13% faster than `pdftotext`, the popular native C/C++ PDF text extraction utility that's part of the Xpdf project.

There's a right tool for every job, and in general, it's better to use a tool that is designed for the specific job at hand. Accurately extracting text and metadata from PDF documents with high levels of performance is a surprisingly difficult job that presents a complex set of problems. Given the importance of finding and accessing critical data available only in PDF documents, it makes sense to use a PDF content extraction library designed from the ground up to solve these problems expertly and without compromises. Doing so will ensure that your application and your users receive the greatest benefits of enterprise-class PDF content integration.

## References

- Snowtide Informatics, publisher of `PDFTextStream`: <http://snowtide.com>
- `PDFTextStream` developer resources: <http://snowtide.com/Support>
- PDF text extraction benchmarks: <http://snowtide.com/Performance>.
- Adobe Extensible Metadata Platform (XMP): <http://www.adobe.com/products/xmp/main.html>
- Apache Lucene project: <http://lucene.apache.org>
- Xpdf project (home of `pdftotext`): <http://foolabs.com/xpdf/>



# JCP Bookmarks

## *Program training goes virtual*

by Onno Kluyt

**W**e'll be coming to the rescue and offering the training program virtually, yes, from the JCP.org site itself starting this September

Most JCP Program activity happens virtually except for a few times a year when the JCP takes to the road to deliver training sessions around the world. Such was the case this June and July. The training marathon started in Sweden. The JCP Program Management Office (PMO) took advantage of the face-to-face JCP Executive Committee (EC) meeting hosted by Sony Ericsson in Stockholm to train newer EC representatives and their alternates about the workings of the process. Then it traveled to Rome and Milan where sessions were delivered to Java developers and marketers attending the Italian Java Conference. The next stop was in Versailles where the training happened in conjunction with JavaDay 2006. Participants learned more about JCP membership, the process, and what it means to be a spec lead, expert, and EC member.

But you may be among those Java fans who didn't have a chance to attend the JCP training sessions in person back in February this year in Santa Clara, CA or at JavaOne in San Francisco in May or couldn't catch any of the JCP training events in Europe either. So here's a mental bookmark for you to make. We'll be coming to the rescue and offering the training program virtually, yes, from the JCP.org site itself starting this September.

So whether you have a broad interest in Java technology or Java standards development and maintenance or you're looking to learn the basics of the JCP, or becoming a spec lead or simply want a refresher about the program, you'll be able to tap into this training resource virtually. Most becoming for a global community like the JCP.

Here's a bit of what you should expect from the Web-based training. First off you'll become knowledgeable about the body that develops standards for the Java platform – the Java Community Process (JCP) Program, what it's all about, how you can leverage it to support your Java

endeavors, and how you can work with the PMO to get help in carrying out your Java projects.

Second, you'll learn about the rigorous JCP Java Specification Requests (JSR) review process and its mechanics from the members of the PMO who manage it on a daily basis.

Third, you'll get a better feeling what it means to be a spec lead and what the intricacies of this role are. Becoming a JSR spec lead and acting to expectations as a lead doesn't happen over night. Even experienced spec leads feel challenged when confronted with the rigors of developing a standard through the process. The JCP training sessions will give you the information you'll need to ride out the process effectively, make JSR deadlines, and develop a healthy relationship with expert groups – yours and others – as well as with other participants in the process.

Fourth, you'll acquire knowledge indispensable to developing a JSR proposal and taking the key JSRs deliverables to the finish line. Putting a standards proposal together has its specifics — that's why your experienced trainers will walk you through what's required to develop it for the JCP. You'll have the opportunity to ask them questions and discuss the details of the Technology Compatibility Kit (TCK).

And last you'll acquire community savvy that you can pass on to your colleagues be they project managers, marketers, sales gurus, or simply Java fans.

The wealth of information and knowledge you'll acquire in the JCP training will help transform you into a more effective player in your Java ecosystem. And the skills you learn are transferable — you'll be able to pass them along to help colleagues succeed with their Java projects. It will give you the training references and support that will meaningfully enhance your Java experience in a variety of roles you may currently have or take on in the future.

Another way of finding out more about the JCP Program, its workings, how you can get involved, and the latest changes is by attending the community panels, which we host at a variety of industry events. For instance, the one I moderated

one in early June at the TheServerSide.com Java Symposium in Barcelona. I was joined on the panel by Jon Bostrum, senior director of Java technology platforms for Nokia, co-leader of the Mobile Expert Group and co-spec lead for JSR 232; Cameron Purdy, founder and president of Tangosol and spec lead of JSR 107; Tom Baeyens, founder and lead developer of JBoss jBPM; Mike Keith from Oracle, co-spec lead of EJB 3.0, co-author of /Pro EJB 3: Java Persistence API/, and a Java EE 5 expert group member. We fielded mainly questions from the audience about the strengths and weaknesses of the process, what role individual developers can play in the process, and what it's like to be a spec lead. Developers made recommendations for more transparency and public scrutiny in the early specification stages, more active participation by individual members, less bureaucracy, and better communications among JSRs. They also commented on what they believe to be a strong benefit of the JCP – the diversity of opinions that goes into the specifications and the fact that the leading experts in a particular field can lead a project. We went on to compare Open Source communities with the JCP which helped the audience understand better the differences and similarities. The panel was also an excellent opportunity for me to express my views on how individual members sign up. It's one aspect of the process that I wish will be streamlined sooner rather than later to make it even easier for individual developers to get involved. Right now there's a lot of opportunity for them to participate but there's still plenty of room to grow.

It was an equally beneficial exercise for us panelists to get people's ideas on how to continue to increase transparency in the early stages of specifications, limit bureaucracy to the minimum needed, and help improve communication among JSRs especially among the Expert Groups of related specifications.

To find out more about the panel and hear the diversity of opinions expressed tune in to [http://www.theserverside.com/news/thread.tss?thread\\_id=41753](http://www.theserverside.com/news/thread.tss?thread_id=41753).

**Onno Kluyt** is the chairperson of the JCP Program Management Office, Sun Microsystems onno@jcp.org.

# Eclipse Data Visualization

*(No Silly Glasses Required)*



Chart FX for Java Eclipse plug-in.

## The Leading Charting Solution Now Provides Powerful Data Visualization for Eclipse

The Chart FX for Java 6.2 Eclipse plug-in brings enterprise-level data visualization features to the Eclipse IDE. The Designer is integrated into the IDE allowing quick customization of the charts and the required code generation. In addition to a myriad of traditional chart types, the Chart FX Maps extension is included to create dynamic, data-driven image maps, such as geographic maps, seating charts or network diagrams, among others. Chart FX for Java 6.2 is available as a Server-side Bean that runs on most popular Java Application Servers. The 100% Java component produces charts in PNG, JPEG, SVG and FLASH formats. The Chart FX Resource Center integrates into the Eclipse Help and includes a Programmer's Guide, the Javadoc API and hundreds of samples. This makes Chart FX for Java the most feature-rich, easy-to-use charting tool available for Java development. *Learn more about the seamless integration and powerful features at [www.softwarefx.com](http://www.softwarefx.com).*



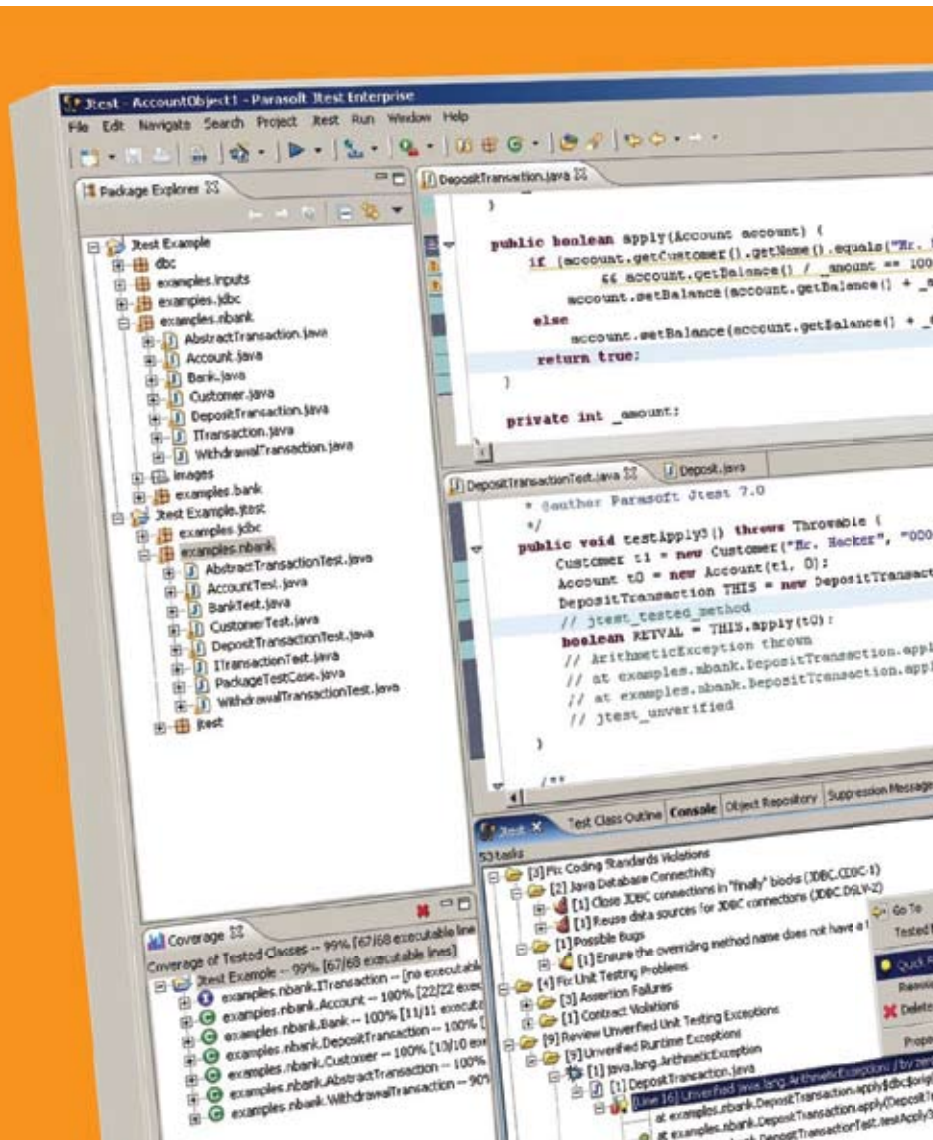
**Chart FX**

[www.softwarefx.com](http://www.softwarefx.com)

**New!**  
**version 6.2**  
**Now Includes Maps!**

# Code in quality. Test out bugs.

Take control of your code through Automated Unit Testing and Code Analysis...



**PARASOFT**  
*Jtest*<sup>®</sup>

- Automatically analyzes your code, applying 500+ Java coding best practices that surface poor code constructs affecting reliability, security and performance.
- Automatically unit tests your Java code evaluating code behavior and exposing unexpected exceptions, boundary condition errors and memory leaks.
- Generates extendable, high coverage test cases in JUnit format that can be quickly turned into libraries of reusable test assets.
- 100's of Quick Fix options for fast, accurate resolution of errors.
- Integrated code coverage analysis ensures that your code is thoroughly tested.
- Team collaboration support allows you to define, distribute and enforce code quality standards across entire development teams.

## Stop bugs before they start.

Jtest automates the valuable, yet often tedious task of code reviews and unit testing, allowing development teams to adopt a "test as you go" strategy that promotes testing code as it's developed, so that quality is built into the code from its inception and bugs are eliminated before they infect the rest of the code base.

The result? Cleaner, safer, more reliable and consistent code. Reduced code rework. Faster and more predictable release cycles, reliable applications, and happier, more productive end-users. [For more information on Parasoft Jtest, call 888-305-0031 ext. 3501 or go to www.parasoft.com/JDJmagazine](http://www.parasoft.com/JDJmagazine)

Parasoft Jtest clients include: Lehman Bros., Cisco, Disney, Raytheon, ADP, Sabre Holdings, GE and more.

**PARASOFT**  
We make software work.™

Automated Error Prevention™