

Greater Michigan Software Symposium 2005

Westin Southfield Detroit - Detroit, MI

September 09 - 11, 2005

(session listing as of 9/7/2005)

The No Fluff Just Stuff Software Symposium 2005 tour is pleased to announce the Greater Michigan Software Symposium coming to Detroit on September 09 - 11, 2005. You will have the opportunity to attend the best technically focused Java / Open Source event anywhere. We make this claim based on the following:

- 1) **Excellent Speakers with unparalleled access**
- 2) **Limited Attendance = 250 Registrants Max**
- 3) **No Vendors, No Sales Pitches, No Marketecture**
- 4) **Unmatched Value - less than 1/3 of the cost of a national conference**
- 5) **Since 2002, we have delivered over fifty(50) conferences throughout North America**

The No Fluff Just Stuff Software Symposium Series caters to individual developers, development teams, project managers, architects and independent consultants. The Greater Michigan Software Symposium will offer 3 concurrent sessions over three days with over 33 sessions to choose from. The following topics will be featured:

- 1) Client Side Java
- 2) ServerSide Java
- 3) XML / Web Services
- 4) Architecture
- 5) Core Java
- 6) .Net

Registration Fees

Attendees	Before 8/22/2005	After 8/22/2005
1-4	\$675	\$775
5-9	\$600	\$675
10-14	\$575	\$650
15-24	\$550	\$625
25+	\$525	\$600

The Registration Fee includes the following:

- 1) All Access Pass to the three day symposium
- 2) Handouts from all sessions attended w/binder
- 3) CD with all presentational content @ registration
- 4) Custom NFJS Laptop/Backpack
- 5) Opportunity to win an iPod everyday during the symposium raffle

Go to <http://www.nofluffjuststuff.com> for more details.

Questions/Comments: Contact Jay Zimmerman: zimmerman@nofluffjuststuff.com or (303)469-0486.

Greater Michigan Software Symposium 2005

Westin Southfield Detroit - Detroit, MI

September 09 - 11, 2005

(session listing as of 9/7/2005)

Applied Object-Oriented Metrics by Brian Sletten

Object-oriented code metrics are a little like Artificial Intelligence: those who did it twenty years ago roll their eyes at the thought and prophesy the same ultimate failure at applicability now. Those who grew up with Java are approaching the topic with new eyes and are finding useful ways of incorporating metrics into their projects. Come hear about tools and ways to measure properties of software, how they might be beneficial and where you are likely to go astray with this approach.

Advanced Object-Relational Mapping with Hibernate by Brian Sam-Bodden

Hibernate is rapidly becoming the tool of choice when it comes to Object-Relational Mapping in Java. For simple applications with fairly simple object models and database schemas, using Hibernate is fairly straight forward. Unfortunately for most of us real applications have complex object-models that need to be wired to sometimes ancient and convoluted database schemas.

Applied AOP by Brian Sletten

Most people new to Aspect-Oriented Programming (AOP) are fed up with separation of concerns zealots explaining how great their techniques are at dealing with... logging. Ok, you get it. Logging is a cross-cutting concern that can be appropriately modularized. What else does AOP have to offer? A lot, it turns out. This talk will give an introduction to the motivations of AOP as well as a series of concrete examples drawn from enterprise and client side Java. Come learn how AspectJ-flavored AOP can begin to benefit you immediately either in development or production environments. Learn how to enforce architectural policies, find Swing threading issues, reduce the invasiveness of the Observer design pattern or even improve the reusability of your domain models.

XML made easy with XOM by Brian Sam-Bodden

XML is quickly becoming the common ground for disparate systems to exchange information and most Java developers deal with XML almost on a daily basis, whether is in deployment descriptors and configuration files or as the data format at the center of their applications.

Complex Builds with Ant by Brian Sam-Bodden

Ant has revolutionized the way we build applications in Java and it has become a de facto standard in the Java world. As applications grow in complexity some developers are finding themselves dealing with ever growing and complex builds. Complex builds have to deal with Multiple Operating System, multiple Application Servers, multiple APIs and multiple stages of development.

Business Rules Engines in Java and J2EE- An Introduction to the Drools Rules Engine by Brian Sam-Bodden

Software development is expensive, when business rules are hard-coded in your application's source code, changes and additions to those rules translate to wasted time and money. Good object-oriented, component-based approaches can alleviate the burden of keeping up with changes in the business world but they still require that expert knowledge of the changes be passed from the decision makers to the business analysts and finally to programmers that need to implement these changes. Business Rule Engines and Business Rule Languages are based on the basic premise of separation of concerns by empowering business domain experts to express the rules of business in a way that it is directly usable by applications.

Applied Design Patterns by Brian Sletten

Just about every modern software developer has a copy of the Gang of Four "Design Patterns" book sitting on a shelf; many of them have actually read it. The dark secret of the patterns community is that there is often a large gulf between whiteboard simplicity and real-world complexity. Language choice plays a part in the design (and even importance) of patterns. The situation is made even more confusing by the fact that many of the core patterns have now been "voted off the island" for one reason or another. This talk will give a pragmatic overview of the motivations behind design patterns and will focus on applying a handful of the GOF patterns to example scenarios in Java and C#. A quick introduction to the role AOP plays in changing the patterns landscape will also be covered.

Felix: A bag of Tricks for Java Server Faces by David Geary

Okay, so you know a little about JSF. You understand managed beans, action outcomes and how to attach standard JSF validators to components in a JSP page. But there is a great deal of functionality that the

average web application supports that JSF doesn't provide out of the box. For example, wouldn't you like to have JSF automatically place asterisks in front of labels for required fields? You are going to implement client-side validation, which JSF does not support out of the box, aren't you? Of course, you're going to test your application, right? And don't forget to trap unauthorized use of the back button.

Killer Web UIs by David Geary

User interfaces are usually the most turbulent aspect of an application during development. Constant tinkering with the UI means constant changes to your code, so as a UI developer, you want to minimize the scope and effects of those code changes. Open-source Java provides two powerful software packages that help you manage UI complexity: Tiles and Sitemesh. Tiles composes webpages from discrete regions of your user interface known as tiles. A tile contains a JSP page for layout and one or more JSP pages for content. Sitemesh decorates webpages with decorators that can be associated with URL patterns. Once you set up your decorators, you can decorate pages that match a decorator's URL pattern.

Software Metrics and the Great Pyramid of Giza by David Bock

Most software engineers hate metrics... Why? Because we know the work we do is hard to quantify # any measurement of 'software engineering' is like trying to tell how tall someone is by how much they weigh... There may be some correlation, but there is so much deviation as to make the answer practically meaningless. As a result, we often see metrics used to justify improper conclusions. There are plenty of good metrics though, and plenty of ways to use them effectively.

Designing and Developing Pluggable Application Architectures by David Bock

Pluggable application architectures are everywhere. Applications like Photoshop, Eclipse and other IDEs, and even application servers are all examples of applications that allow other developers to 'install' new functionality. There are plenty of reasons for wanting to install new functionality into an application that is already developed and deployed... from dynamic upgrading to the creation of a 'component marketplace', where end users can purchase components with 'extra' capability.

An Introduction to JavaServer Faces by David Geary

There are a lot of Java-based web application frameworks, but how many of them are: 1. Based on the most popular open-source framework (Struts), and 2. The standard that must be supported by every J2EE 1.5 container? JavaServer Faces (JSF) debuted in the Spring of 2004. Throughout the rest of 2004, JSF gained momentum with a handful of books and a growing user community that includes the popular MyFaces open-source JSF implementation which has recently moved from SourceForge to Apache. Perhaps the most telling sign of the times is Craig McClanahan's proposal for Struts 2.0, code named Shale, which reinvents Struts as a set of services for JSF applications.

Shale: The Next Struts? by David Geary

Struts is the most popular Java-based Web application framework today, but that's rapidly changing. There's a newcomer on the block, a leaner, meaner, better-designed framework loosely based on Struts that's poised to dethrone Struts as the reigning king of Java-based web application frameworks. That framework, of course, is JavaServer Faces. Craig McClanahan, the father of Struts and the co-spec lead for JSF 1.0, has proposed reinventing Struts for Struts 2.0 as a set of services for JSF applications. That new framework, which has no direct ties to Struts as we know it, is called Shale.

Ruby on Rails by David Geary

At about the same time Java was brewing, another language from the far east entered the landscape with hardly any notice. Carefully crafted by Japanese devotees, Ruby, a potent mix of SmallTalk, Python, and Perl, toiled in relative obscurity as the marketplace moved in droves to Java. Today we have J2EE, the 800-pound gorilla of enterprise development. That 800 pounds cuts both ways: J2EE is a complicated beast with a long and steep learning curve that sports a dissing array of peripheral open-source software. And like any 800-pound gorilla, J2EE has many design compromises and idiosyncrasies that reflect it's growth and evolution. Some J2EE developers have begun to wonder if there's a better way...

Writing Secure Web Services (with Java and Axis) by Justin Gehmland

Web Services are message-oriented. This means that any application intention (the need for security, for transactionality, for reliability, etc.) must be included in the message and not just assumed as external context. The WS-Security specifications are very advanced and currently being used in the wild to create robust, secure web services.

Advanced Hibernate by Justin Gehmland

Hibernate is easy to get started with, but can sometimes be hard to make efficient or secure. In fact, the

default settings for Hibernate create applications that will run slowly, cause unwanted round trips to the database, and may be more restrictive and/or permissive from a security standpoint than you would otherwise want.

Spring Security with ACEGI by Justin Gehtland

Spring offers developers a simpler, more robust method for configuring applications. These benefits extend to security through the ACEGI framework. ACEGI makes the otherwise daunting task of securing your application logical and straightforward. More importantly, through its support for single sign-on provision through Yale's CAS system and its ability to provide instance-level authorization, Spring extends the common security model of most J2EE apps beyond what they are traditionally capable of.

Introduction to Hibernate by Justin Gehtland

O/RM (Object/Relational Mapping) seeks to eliminate repetitive or tedious work enabling the CRUD (create, read, update, delete) that underlies most applications. Hibernate is a popular, open-source O/RM tool that uses reflection (instead of code generation, like EJB, or bytecode injection, like JDO) to manage your persistence layer. This session will introduce you to Hibernate. After an overview of common usage scenarios, including web and enterprise applications, we'll examine the basics of getting Hibernate running. We'll cover the mapping file format and syntax, including common relational mapping structures. Then, we'll examine the Hibernate API for interacting with the framework. Finally, we'll cover the common architectural decisions you'll have to make as you include this (or any other) O/RM framework.

Spring Intro by Justin Gehtland

The Spring framework is one of the fastest growing open source frameworks. New job postings are gaining rapidly, and many customers are adopting Spring instead of heavier alternatives. In this session, we'll introduce Spring. You'll see how Spring can give you much of the power of EJB, without the complexity or pain. Spring uses concepts like dependency injection and aspect oriented programming to ease standard enterprise development. Spring developers write plain, ordinary Java objects (POJOs), instead of sophisticated components. In this session, you'll see a basic Spring application. You'll also see some details about some of the enterprise integration strategies, including: # Spring AOP # Transactions # Persistence # Model/view/controller When the session is over, you won't be an expert, but you should have a much clearer understanding of what Spring does, what it doesn't do, and why it's growing so rapidly.

Creating Polished Swing Applications by Scott Delap

Too often, Swing applications are slow, ugly, and hard-to-maintain. It turns out that it doesn't have to be this way. Swing can be used to create highly-responsive, beautiful applications that are very maintainable. If this isn't consistent with your own experience, don't feel bad; it's not very obvious how to make Swing sing.

5 Minutes Forms with JGoodies Binding and Validation by Scott Delap

Application developers often spend hours on the simple tasks of laying out a form, wiring components to objects, and validating the data entered. This is time that could be much better spent on the business problems your application is trying to solve. This session will show how to leverage open source libraries to take the work out of the form building process.

SWT Fundamentals by Scott Delap

The Eclipse project's SWT GUI toolkit provides one of the only viable alternatives to Swing for creating so-called rich client applications in Java. Whereas Swing paints its own widgets and has distinguished itself with a complex (and often obtuse) API, SWT relies on the host operating system for widget rendering and sports a simple, clean API. If your goal is to create a Java application that "looks" like a normal Windows application (or OS X, or Linux), SWT will revolutionize your world. In this session, I introduce SWT from the ground up.

Design Patterns Revisited: Taking advantage of dynamic, reflective languages by Stuart Halloway

(3 Hour Session) Attendees should attend the Introduction to Reflection talk, or have some experience using reflection or metaprogramming in a reflective language such as Java, Objective-C, Smalltalk, Python, or Ruby. Familiarity with the GOF book is helpful but not required. Design patterns are recurring solutions to problems that consistently appear in software development. However, this does not mean that design patterns cannot be "solved", i.e. converted into language or library features. In fact, most of the original design patterns can be solved using dynamic language features such as reflection.

Introduction to Java Reflection by Stuart Halloway

Reflection is writing code that manipulates itself. Well-written reflective code automates a broad class of repetitive, error-prone programming tasks. Poorly-written reflective code obfuscates programs and destroys

the benefits of the type system. We'll focus on the former.

Programming Java Concurrency by Stuart Halloway

Java has always provided a model for concurrency and threads. With Java 1.5, this model received a major facelift. Learn how to use the new concurrency utilities to build responsive, scalable, and correct concurrent applications.

Unit Testing Java with Jython by Stuart Halloway

JUnit is great. Jython is even better. Unit testing libraries look the same everywhere, so why not use the one that lets you get your job done faster?

Java Platform Security and JAAS by Stuart Halloway

The Java platform is built from the ground up with security in mind. This talk will introduce the security features of the J2SE, building quickly from the basic classes to realistic examples.

Introduction to Web services, 2005 edition by Ted Neward

WSDL, and Schema and SOAP, oh my! It's 2005, and the Web services landscape looks even more confusing than it did two years ago, despite all sorts of promises to the contrary. What's it all mean, and how the heck did we get here when the original goal was to try and keep it all simple?

Working with Java Metadata by Ted Neward

As part of JDK 1.5, Java has introduced a facility for developers to create and use custom metadata annotations, as developed by the JSR 175 committee. This represents a radical new shift for the Java programming language, quite possibly larger and farther-reaching than generics or any other language feature.

The Fallacies of Enterprise Systems by Ted Neward

There's a set of fallacies that every enterprise developer has fallen for at some point in their enterprise development lives, and unless they've come to realize it early enough, all cause big trouble and painful learning experiences in the long run.

Effective Enterprise Java: Security by Ted Neward

Security's become a hot topic among enterprise developers in recent years, but to many developers, security is still the white elephant in the middle of the room. Discussions about security usually begin with, "Uh, we'll worry about that later", or, "Start with two really large prime numbers.....". Security isn't as hard as developers make it out to be, but it is something that developers need to face and recognize.

Effective Enterprise Architecture by Ted Neward

Bring all of your enterprise Java questions to this open forum discussion hosted by the author of #Effective Enterprise Java#, Ted Neward.