

# New England Software Symposium 2005

Sheraton Framingham - Framingham, MA

September 23 - 25, 2005

(session listing as of 9/18/2005)

The No Fluff Just Stuff Software Symposium 2005 tour is pleased to announce the New England Software Symposium coming to Framingham on September 23 - 25, 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 New England Software Symposium will offer 6 concurrent sessions over three days with over 65 sessions to choose from. The following topics will be featured:

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

## Registration Fees

Attendees	Before 9/12/2005	After 9/12/2005
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](mailto:zimmerman@nofluffjuststuff.com) or (303)469-0486.

# New England Software Symposium 2005

Sheraton Framingham - Framingham, MA

September 23 - 25, 2005

(session listing as of 9/18/2005)

## **Ajaxian JavaScript Frameworks by Ben Galbraith**

In the "Introduction to Ajax" session, we discuss what Ajax is, how it works, and how others are using it. This session goes deeper into Ajax by reviewing the existing JavaScript frameworks that aim to make it easier.

## **Making the Most of XML by Ben Galbraith**

For many of us, XML has become a ubiquitous presence in application development, whether parsing, validating, or manipulating it. For many of us, all that XML is coupled with pain, in the form of tedious APIs (like, say, the W3C DOM API) and confusing technologies (oh, I don't know, W3C XML Schema?).

## **Being Productive with Java in the Enterprise by Ben Galbraith**

It sounded like such a good idea back in the mid-nineties: based the Java platform on a standards-based, open community, and let anyone participate. There is no question that Sun's strategy for Java's stewardship via the JCP and sponsored open-source has yielded some enormous benefits. However, these have not been enjoyed without tremendous cost.

## **Creating Killer Graphics and Professional PDFs with XML by Ben Galbraith**

You can do some pretty cool things with XML these days (despite what some curmudgeons in the technology world may claim). In the past few years, XML has solidified its place as the lingua franca of data sharing and data manipulation. But XML as a data transfer language is only marginally interesting. Things get really exciting when XML is dynamically transformed into other formats. In this session, I focus on two XML formats which can be readily transformed into high-quality presentation-centric output formats. XSL-FO is a typesetting format for XML that can be readily converted into PDF (or Postscript and some other formats). SVG is a vector graphics language in XML -- a sort of open-source version of the popular Macromedia Flash format. SVG files can be converted into beautiful, completely scalable -- and interactive -- images.

## **Creating Polished Swing Applications by Ben Galbraith**

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.

## **Advanced Swing: Architecture and Frameworks by Ben Galbraith**

Are you spending more time plumbing your Swing applications than solving business problems? Has your Swing application grown out of control? This session is for you.

## **Introduction to Ajax by Ben Galbraith**

Ajax -- called DHTML just a few months ago -- has revolutionized (or "radically iterated", if you like) web application development in the short few months since the term was coined. What is it all about? Why are we excited about a set of capabilities that have been sitting in our browser for years? What can you do with it? And, how can you do it?

## **Introduction to concurrency utilities in JDK 5.0 by Brian Goetz**

JDK 5.0 is a huge step forward in developing concurrent Java classes and applications, providing a rich set of high-level concurrency building blocks.

## **Garbage Collection in the HotSpot JVM by Brian Goetz**

Pop quiz: which is faster, Java or C++? If you are talking about allocation performance, the answer is Java, hands-down.

## **The Java Memory Model by Brian Goetz**

What's the worst thing that can happen when you fail to synchronize in a concurrent Java program? It's probably worse than you think -- modern shared-memory processors can do some pretty weird things when left to their own devices.

## **Squashing bugs with FindBugs by Brian Goetz**

Does your program have bugs, despite unit tests, integration tests, and code reviews? You bet. Fortunately,

there are some new code auditing tools that can help spot some bugs missed by other approaches.

### **Stretching Java by Bruce Tate**

In Stretching Java, we'll look at some of Java's limitations, see how other programming languages solve those problems, and look at how Java developers can implement those ideas in Java using open source frameworks, design strategies, and tools.

### **Beyond Java by Bruce Tate**

All programming languages have a limited life span, and Java is no different. This is a philosophical session rather than a programming session. Sooner or later, Java will lose its leadership position. This session will explore Java's strengths and weaknesses. We'll try to understand whether conditions are ripe for alternatives to emerge, and what those alternatives may be.

### **Where Agile meets Argyle: New processes in established companies by Bruce Tate**

Agile programming is a collection of core principles and techniques that allow software developers to create lighter, more responsive applications, and to have fun doing it. Many established organizations are either openly or sub-consciously hostile to many of the principles of Agile development.

### **Introduction to Spring by Bruce Tate**

This session, for the Spring beginner, helps you: # Understand dependency injection and inversion of control # Know the meaning of lightweight containers and Spring # Understand the basic pieces of Spring # See core Spring modules in action, including Persistence, AOP, transactions. Attendees need not know anything about Spring. This session does talk about integration with core J2EE frameworks like JDBC and transactions.

### **Herding Racehorses and Racing Sheep by Dave Thomas**

Are you frustrated by experts who can't tell you what to do, or by junior team members who refuse to see the big picture? How can you best develop careers: both yours and those of your teammates and managers? How can we learn to apply experience more effectively, and why do the many approaches designed to tame complexity actually end up increasing it?

### **Ruby for Java Programmers by Dave Thomas**

Ruby recently enjoyed its tenth birthday. Instead of cake and candles, the community celebrated by releasing a wave of new libraries and frameworks that make Ruby programming even easier. This talk features some of the best of these, as we explore Ruby.

### **OpenSource Ecosystems by Dave Thomas**

Open Source communities produce high quality software with little management and (typically) no pay. Most people looking at open source focus on using this software in their projects.

### **Ruby on Rails by Dave Thomas**

The Ruby on Rails framework has exploded onto the scene over the last few months. Propelled by some genuine benefits, and fueled by a whole lot of controversy, Rails seems here to stay. So, is it a Java killer? (No.) Is it a great way to develop certain classes of web application? (Yes.) Does it really deliver the 10-fold increase in developer productivity that some have claimed? (It depends...)

### **Writing and Telling User Stories by David Hussman**

The technique of expressing requirements as user stories is one of the most broadly applicable techniques introduced by Extreme Programming. User stories are an effective approach on all time constrained projects, not just those using XP.

### **Transitioning to Agile # Keys to Success by David Hussman**

Transitioning to an agile process from a traditional process is fraught with potential dangers. Attend this class and learn the key things you absolutely must do in order to succeed.

### **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 its growth and evolution. Some J2EE developers have begun to wonder if there's a better way...

### **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.

### **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.

### **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.

### **Choosing An Agile Path: Agile Case Studies by David Hussman**

With more and more companies choosing to take an agile path, the bounds of agile ways and means are expanding. Agile implementations differ a great deal depending on the company, the project or product, and the players.

### **Overview of Agile Estimating & Planning by David Hussman**

Planning is important even for projects using agile processes such as XP, Scrum, or one of the many other agile processes. Unfortunately, we've all seen so many worthless plans that we'd like to throw them away altogether. The good news is that it is possible to create a project plan that looks forward six to nine months that can be accurate and useful.

### **Code Coverage: A Guardian of Quality by Ian Roughley**

Code coverage is generally viewed as a metrics that managers use to chart progress, a number that has to be blindly attained. In this talk we discuss everything that you, the developer, need to know to make it more than a number and part of a process that will improve code quality.

### **The Open Source Bazaar by Ian Roughley**

Many companies and most, if not all, software today utilizes open source. Whether it is databases, application servers, frameworks or libraries, these projects are fast becoming a standard commodity for building business-related functionality upon and speeding up development time. Sometimes technology evaluations are done, but frequently the library is simply slipped into the code base to address an urgent requirement - often without evaluating the technology beyond the immediate need.

### **Writing Secure Web Services (with Java and Axis) by Justin Gehtland**

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.

### **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 Gehrtland**

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.

### **Advanced Hibernate by Justin Gehrtland**

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 MVC by Justin Gehrtland**

The Spring team, as in all things they do, have learned the valuable lessons of the past when introducing a Spring solution. Spring MVC is everything Struts should be, and more besides.

### **A Pragmatic Look at Agile Architecture by Mark Richards**

Designing application and enterprise architectures is a complex process. We follow defined processes, create lots of attractive architecture diagrams, kill lots of trees producing hundreds of pages of architecture documentation, and yet we find that in many cases the architectures we design are not followed by developers or simply not understood by the stakeholders or development community. As a result, either the delivered software does not match the original architecture, or the architect works 25 hours a day to ensure that the software is in compliance with the original architecture. This is a case where Agile Architecture can help. Agile architecture is an architecture process that leverages Agile principles and applies them to the design of application and enterprise architectures. During this session you will learn the shortcomings of standard architecture processes and why they sometimes go wrong. We will then look at some Agile principles and see how we can apply these methods to simplify the architecture process and produce better architectures. We will also look at some pragmatic techniques for architecture design and diagramming that support Agile Architecture, and learn how ATAM (Architecture Trade-off Analysis Method) can be applied to the concept of Agile Architecture. Of course, no pragmatic session would be complete without a discussion of the issues and shortcomings of the Agile Architecture process itself, so we will explore where the Agile Architecture process sometimes fails to live up to its promise. There will be plenty of room for discussion in this session, so please bring your architecture pains (and successes) with you so we can engage in a lively discussion.

### **Understanding the Role of an ESB by Mark Richards**

The Enterprise Service Bus is an integral part of any Service-Oriented Architecture. It is the glue that binds the business services to the client applications. There are many ESB third-party products and solutions in the marketplace, but in most cases these products only serve to further confuse us in terms of what an ESB is, particularly when you consider that an ESB is really an architectural component that has many different implementations. In this session we will take a detailed, product-agnostic look at the role of an ESB and the capabilities an ESB must provide. Through this session you will learn what an ESB is, the role of an ESB, what capabilities it provides, and the various ways an ESB can be implemented. We will also take a close look at the Java Business Integration (JBI) specification (JSR-208) and see what impact it will have with the ESB world. With the information from this session you will learn how to determine your own specific requirements for an ESB and then match those requirements to the product space rather than having the tail wag the dog!

### **Using the J2EE Command Pattern to Simplify Client-Server Communications by Mark Richards**

The latest buzz in the industry relating to EJBs is to avoid them at all costs. However, there are many times application architectures require the use of remote services using EJBs. Using EJBs (specifically Stateless Session Beans) can sometimes lead to configuration complexity, transactional complexity, performance issues, and testing complexity. In this session we will see how the use of the J2EE Command Pattern can

address these issues. The J2EE Command Pattern is an important design pattern that every developer and architect should be familiar with. We will first take a look at how the Command Pattern can help resolve some of the complexity and performance issues, and then look at two different implementations of the Command Pattern; Dynamic Commands and Static Mapped Commands. Through this session you will learn when the J2EE Command Pattern is appropriate, the different ways of implementing the pattern, and the implications of each approach.

### **EJB 3.0 and Java Persistence API Review by Mark Richards**

The new EJB 3.0 spec (JSR-220) offers some great improvements over the prior EJB specs in terms of development simplicity and new features. In this session we will take a look at the new EJB 3.0 spec and the new Java Persistence API. Included in this session will be a discussion about Java metadata annotations, simplification of enterprise beans (session and message-driven beans), interceptors, changes in transaction processing, and how the new Java Persistence API works. During the session I will be demonstrating how the EJB 3.0 spec differs from the EJB 2.1 spec through code example comparisons. I will also be discussing how the new Java Persistence API compares to related Java persistence options and whether we should be excited about the new persistence API or (yawn) sticking with what we have.

### **Development Infrastructure Patterns by Paul Duvall**

Design Patterns became part of the software development industry mainstream in the mid-1990s with the release of the Go4 Design Patterns book. Since then, architecture, design, and more recently, organizational patterns have become a part of our nomenclature. But, what about the software that helps us develop and deliver the software to our users: the software development infrastructure?

### **Continuous Integration using CruiseControl and CVS by Paul Duvall**

Continuous Integration (CI) is the process of continually building and testing your software under development. It is identified as a core XP practice, although it works with many software development processes.

### **What's new in AOP by Ramnivas Laddad**

A lot is happening in the field of Aspect-oriented programming (AOP). AspectJ and AspectWerkz, the two leading AOP implementations, have merged bringing in their respective strengths. The merged version (AspectJ 5, currently in a milestone release) adds many new features aimed at simplifying writing and deploying aspects. The new features include an annotation-based and XML-based syntax to define aspects, support for new Java 5 concepts, and load-time weaving. The tools support for AOP continues to improve, as well. Further, the most popular IOC framework # Spring # enables integrating aspects written in AspectJ. There is also serious discussion and preliminary work going on to support AOP right into the VM itself. All in all, there is a lot to learn about the changes in the exciting field of AOP. This session is designed to help you get up to date with all these changes.

### **Introduction to Aspect-oriented programming with AspectJ by Ramnivas Laddad**

Aspect Oriented Programming (AOP) enables modularizing implementation of crosscutting concerns that abound in practice: logging, tracing, dynamic profiling, error handling, service-level agreement, policy enforcement, pooling, caching, concurrency control, security, transaction management, business rules, and so forth. Traditional implementation of these concerns requires you to fuse their implementation with the core concern of a module. With AOP, you can implement each of the concerns in a separate module called aspect. The result of such modular implementation is simplified design, improved understandability, improved quality, reduced time to market, and expedited response to system requirement changes. Come to this session and learn all about how AOP can help you simplify developing complex systems.

### **Performance monitoring in J2EE applications by Ramnivas Laddad**

J2EE has become the main new platform for enterprise application deployment. Good performance is an important requirement from the business viewpoint. Supporting this requirement needs application profiling during the development phases and performance monitoring after deploying the application. Come to this session to understand challenges and choice in monitoring J2EE applications.

### **Cryptography for Programmers by Stuart Halloway**

For centuries people have used crypto to build (and break) secure systems. Computers have only raised the pitch of conflict, providing enormous cryptographic power at commodity prices. Most programmers do not write their own crypto libraries, instead relying on the services of an operating system or virtual machine. But even with all this support, building secure systems is a daunting task.

### **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.

### **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.

### **Good, Bad and Ugly of Java Generics by Venkat Subramaniam**

Java introduced Generics in the 1.5 version (Java 5). What are the capabilities of Generics? How do you use it? Are there some gotchas in using it? In this example driven presentation, we will start at the basics of generics and look at its capabilities. We will then look at some of the under the hood details on generics implementation. We will then delve into the details of some of the changes to Java libraries to accommodate generics. Finally we will take a look at some restrictions and pitfalls that we need to be familiar with when it comes to practical and prudent use of generics.

### **Prudent OO Design by Venkat Subramaniam**

Is your code object-oriented? Developing with objects involves more than using languages like Java, C#, C++ or Smalltalk for that matter. From time to time, the OO paradigm stumps even expert developers. Agile programming becomes a mere act of hack if we code without knowing the OO principles. What are these principles # the ones that influence your design? In this presentation the speaker will present some of the challenges that are fundamental in nature. Then he will present OO Design principles and good practices for prudent development of OO code.

### **Programming with Mock objects by Venkat Subramaniam**

You are convinced that Test Driven Development is good for you and your project. You realize the benefits it has to offer. What's holding you back? All the code and components that your code so heavily depends on is most likely making you wonder if TDD is really for you. We will start out by looking at dependency and dependency inversion. Then we will discuss how mock objects can help separate our code from its dependencies.

### **Agile Methodologies by Venkat Subramaniam**

Agile development is picking up steam. You have heard about eXtreme Programming(XP). What other Agile methodologies are you familiar with and what do they bring of interest or significant to the table of Agility? More important, why should you learn about these different methodologies instead of simply focusing on one? There is no one shoe that fits all. Any methodology that requires you to follow it in totality and not let you adapt is rather dogmatic, not pragmatic. To be effective we have to take the best of different approaches and apply to our projects base on our specific needs.

### **Java 5 Features, What's in it for you? by Venkat Subramaniam**

A number of new features have been introduced in Java. What benefit do these features offer you. Are there issues with using these features. For instance, when should you use annotation? The objective of this presentation is not simply to introduce you to the features, but to the effective use of these as well.