

New England Software Symposium

Sheraton Framingham
September 11 - 13, 2009

<http://www.nofluffjuststuff.com/conference/boston/2009/09/home>

Fri, Sep. 11, 2009

	Ashland	Middlesex	Carlisle	Concord
12:00 - 1:00 PM	REGISTRATION			
1:00 - 1:15 PM	WELCOME			
1:15 - 2:45 PM	Effective Concurrent Java Brian Goetz	On Being a Software Architect Mark Richards	Building External DSLs Venkat Subramaniam	JSF 2.0: An Introduction David Geary
2:45 - 3:15 PM	BREAK			
3:15 - 4:45 PM	The Java Memory Model Brian Goetz	Common AntiPatterns and How To Avoid Them Mark Richards	Effective Java Venkat Subramaniam	JSF 2.0: Advanced Topics David Geary
4:45 - 5:00 PM	BREAK			
5:00 - 6:30 PM	Are All Web Applications Broken? Brian Goetz	Programming Scala Venkat Subramaniam	Transaction Pitfalls and Strategies Mark Richards	Flex for Java Developers David Geary
6:30 - 7:15 PM	DINNER			
7:15 - 8:00 PM	Keynote: by Ted Neward			

Sat, Sep. 12, 2009

	Ashland	Middlesex	Carlisle	Concord
8:00 - 9:00 AM	BREAKFAST			
9:00 - 10:30 AM	Garbage-collector-friendly programming Brian Goetz	The Busy Java Developer's Guide to Java Platform Security Ted Neward	Introduction to JMS Mark Richards	Cleaning up Code Smell Venkat Subramaniam
10:30 - 11:00 AM	BREAK			
11:00 - 12:30 PM	Inside the JVM Brian Goetz	Testing with dependencies Venkat Subramaniam	Advanced Topics in JMS Mark Richards	The Busy Java Developer's Guide to Advanced Platform Security Ted Neward
12:30 - 1:30 PM	LUNCH			
1:30 - 3:00 PM	Third time's a charm: What's new in Spring 3.0 Craig Walls	GWT: An Introduction David Geary	Tackling Concurrency on the JVM Venkat Subramaniam	The Busy Java Developer's Guide to Collections Ted Neward
3:00 - 3:15 PM	BREAK			
3:15 - 4:45 PM	Beyond JUnit: Powertools for Test-Driven Development Craig Walls	The Busy Java Developer's Guide to Java7 Ted Neward	Groovy XML Ninja Skills Scott Davis	GWT: Advanced Topics David Geary
4:45 - 5:45 PM	BIRDS OF A FEATHER SESSION			

Sun, Sep. 13, 2009

	Ashland	Middlesex	Carlisle	Concord
8:00 - 9:00 AM	BREAKFAST			
9:00 - 10:30 AM	JavaScript: the Good, the Bad, and the Ugly Nathaniel Schutta	WAR and Pieces: Breaking down monolithic web applications with Spring-DM and OSGi Craig Walls	Dim Sum Grails: A Sampler of Practical Non Database-Driven Grails Applications Scott Davis	Taking Agile From Tactics to Strategy Stuart Halloway
10:30 - 11:00 AM	MORNING BREAK			
11:00 - 12:30 PM	Making Web Apps Suck Less Nathaniel Schutta	That old Spring magic has me in its SpEL: DI Wizardy with the Spring Expression Language Craig Walls	RESTful Grails Scott Davis	Agile Retrospectives Stuart Halloway
12:30 - 1:15 PM	LUNCH			
1:15 - 2:15 PM	EXPERT PANEL DISCUSSION			
2:15 - 3:45 PM		Web 2.0 Checklist: Deconstructing Modern Websites Scott Davis	Java.next: Clojure, Groovy, JRuby, and Scala Stuart Halloway	Seven Habits of Highly Dysfunctional Teams Nathaniel Schutta
3:45 - 4:00 PM	BREAK			
4:00 - 5:30 PM	10 Things you should know about Software Risk Management Mark Johnson	Groovy Testing Scott Davis	Programming Clojure Stuart Halloway	Hacking Your Brain for Fun and Profit Nathaniel Schutta

New England Software Symposium

Sheraton Framingham
September 11 - 13, 2009

<http://www.nofluffjuststuff.com/conference/boston/2009/09/home>

Groovy XML Ninja Skills by Scott Davis

"XML is like violence: if it doesn't solve your problem, you aren't using enough of it." (Anonymous) XML is everywhere. Whether you are dealing with local configuration files (web.xml, struts-config.xml) or remote web services (SOAP, REST, RSS, Atom), the modern software developer needs to be able to request, slice, and dice XML with ease. That requires a set of razor-sharp tools that reduce the inherent complexity of the problem, not multiply it. Once you see XML tremble in fear at the awesome power of Groovy, you'll wonder what you ever did without it.

Dim Sum Grails: A Sampler of Practical Non Database-Driven Grails Applications by Scott Davis

"The proof of the pudding is in the eating. By a small sample we may judge of the whole piece." (Miguel de Cervantes Saavedra) Most Grails tutorials demonstrate how easy it is to build simple CRUD (Create/Retrieve/Update/Delete) applications. While skinning a database with a web front-end is undeniably one beneficial aspect of Grails, it isn't the only thing Grails is good for. As you'll see here, Grails can be used to build a wide variety of web applications. You won't see a single HTML table with "edit" and "delete" links, I promise.

RESTful Grails by Scott Davis

"Any intelligent fool can make things bigger, more complex, and more violent. It takes a touch of genius - and a lot of courage - to move in the opposite direction." (Albert Einstein) REST and Resource-Oriented Architecture (ROA) are popping up in technical discussions more and more frequently. Here, you'll see practical examples of adding RESTful web services to your Grails application.

Web 2.0 Checklist: Deconstructing Modern Websites by Scott Davis

"The challenge of modernity is to live without illusions and without becoming disillusioned." (Antonio Gramsci) There are plenty of sarcastic "Web 2.0" checklists out there -- be perpetually in BETA, when in doubt add rounded corners, etc. While we can all laugh at the superficial aspects of the Web 2.0 revolution, there are plenty of serious aspects to it as well. Is your website mash-up friendly or hostile? Do you tell your visitors when things change (via RSS or Atom syndication), or do you expect them to check in daily for updates? Is your website a silo or a part of a larger ecosystem?

Groovy Testing by Scott Davis

"Tests don't break things; they dispel the illusion that it works." (Anonymous) In this era of "Test-First" and "Test-Driven" development, the modern software engineer knows that testing is no longer an optional part of the process. You need to have the best tools at your fingertips: a set of utilities that maximize your results with a minimum of effort. Groovy offers Java developers an optimal set of testing tools.

JSF 2.0: An Introduction by David Geary

This session introduces JSF 2.0 fundamentals, with emphasis on new features in JSF 2.0. **Prerequisite:** *Familiarity with JSF, or other component-based frameworks*

JSF 2.0: Advanced Topics by David Geary

This session covers two of the most important features of JSF 2.0: composite components and built-in Ajax. **Prerequisite:** *Familiarity with JSF, or other component-based frameworks. Familiarity with Ajax. This session builds on the JSF 2.0 Introduction talk, so it is helpful, although not required, if you attend the intro talk before coming to this session.*

Flex for Java Developers by David Geary

An introduction to Flex for Java developers. **Prerequisite:** *Familiarity with Flex and at least one other web application framework*

GWT: An Introduction by David Geary

An introduction to Google Web Toolkit. **Prerequisite:** *Familiarity with a component-based framework, preferably a desktop application framework*

GWT: Advanced Topics by David Geary

Learn to do really cool stuff with GWT. **Prerequisite:** *The GWT: Introduction session is not a prerequisite for this session, but it will help if you have some familiarity with GWT.*

Effective Concurrent Java by Brian Goetz

The Java programming language has turned a generation of applications programmers into concurrent programmers through its direct support of multithreading. However, the Java concurrency primitives are just that: primitive. From them you can build many concurrency utilities, but doing so takes great care as concurrent programming poses many traps for the unwary.

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? Its probably worse than you think -- modern shared-memory processors can do some pretty weird things when left to their own devices.

Are All Web Applications Broken? by Brian Goetz

Many developers believe that web frameworks "take care of" the details of concurrency, but this is only because most web applications make limited use of state. Stateful web applications also need to be careful about hazards like races. This talk will use the Java Memory Model to analyze common patterns of state management in web applications. **Prerequisite:** *The Java Memory Model*

Garbage-collector-friendly programming by Brian Goetz

To many developers, garbage collection is black magic. Accordingly, there are is a lot of conflicting advice about what is good or bad for the garbage collector. In this talk, I look at how garbage collection is implemented in the HotSpot VM, and techniques for writing programs that exhibit good garbage collection behavior. Surprisingly, many of these techniques coincide with writing good, clean code.

Inside the JVM by Brian Goetz

Ever wondered what happens to your bytecodes when they're executed by a Java Virtual Machine? This talk provides a peek "under the hood" of modern JVMs, exploring dynamic compilation, speculative optimization, garbage collection, and some hardware-specific optimizations.

Taking Agile From Tactics to Strategy by Stuart Halloway

Teams adopting agile should begin at a tactical level, but they shouldn't end there. The Agile Manifesto operates at many different levels. Learn to apply the principles of agile at a strategic level. Otherwise you can have a great agile ground game and still lose.

Agile Retrospectives by Stuart Halloway

Agile teams manage change and risk by apapting. But to adapt, you must identify opportunities for change and take them. Retrospectives are a fun, cost-effective way for your team to learn and change.

Java.next: Clojure, Groovy, JRuby, and Scala by Stuart Halloway

In this talk, we will explore and compare four of the most interesting JVM languages: Clojure, Groovy, JRuby, and Scala. Each of these languages aims to greatly simplify writing code for the JVM, and all of them succeed in this mission. However, these languages have very different design goals. We will explore these differences, and help you decide when and where these languages might fit into your development toolkit. For more information see <http://blog.thinkrelevance.com/2008/9/24/java-next-overview>.

Programming Clojure by Stuart Halloway

Find out why Clojure is Java.next: * Clojure provides clean, fast access to all Java libraries. * Clojure provides all the low-ceremony goodness you know and love from dynamic languages such as Ruby and Python. * Clojure includes Lisp's signature feature: Treating code as data through macros. * Clojure's

emphasis on immutability and support for software transactional memory make it a viable option for taking advantage of massively parallel hardware.

Groovy Closures - The way to cleaner code by Mark Johnson

The factory patterns and callbacks have been around for a long time as a technique to provide flavor specific code variations. But they are awkward and hard to update. Enter Groovy closures. Imagine having the ability to inject different coding flavors using code closures. If you need a different flavor, then just pass a different code block. Now imagine that all of this works on the JVM!

10 Things you should know about Software Risk Management by Mark Johnson

Once you leave academic "hello world" projects, software development is full of unknowns which result in the high rate of project failure we see too often in industry. This presentation will cover 10 principles of software risk management necessary for project success.

The Busy Developer's Guide to Iconoclasm by Ted Neward

History is littered with the stories of iconoclasts--people who truly stood out as pioneers, lateral thinkers, and in some cases, outright heroes--and their successes and failures. From the baseball management vision of Branch Hickey to the glassblowing vision of Dale Chihuly to the engineering design vision of Steve Jobs, iconoclasts have changed our world in subtle and profound ways, sometimes loudly, sometimes quietly.

The Busy Java Developer's Guide to Java Platform Security by Ted Neward

Permissions, policy, SecurityExceptions, oh my! The Java platform is a rich and powerful platform, complete with a rich and powerful security mechanism, but sometimes understanding it and how it works can be daunting and intimidating, and leave developers with the basic impression that it's mysterious and dark and incomprehensible. Nothing could be further from the truth, and in this presentation, we'll take a pragmatic, code-first look at the Java security platform, including Permissions, the SecurityManager and its successor, AccessController, the Policy class and policy file syntax, JAAS, and more.

The Busy Java Developer's Guide to Advanced Platform Security by Ted Neward

So you know the platform security model, and now you want to use it in new and interesting ways, like creating a custom Policy implementation, a custom Permission, or create a custom security context in which code will execute. Perhaps you even wish to make certain objects accessible only to those with the right permissions, or cryptographic key. Nothing could be easier, despite Java security's reputation as a dark and arcane place. **Prerequisite:** *The Busy Java Developer's Guide to Platform Security*

The Busy Java Developer's Guide to Collections by Ted Neward

For so many Java developers, the java.util.* package consists of List, ArrayList, and maybe Map and HashMap. But the Collections classes are so much more powerful than many of us are led to believe, and all it requires is a small amount of digging and some simple exploration to begin to "get" the real power of the Collection classes.

The Busy Java Developer's Guide to Java7 by Ted Neward

Even though the Java 7 JSR has yet to be formed, some interesting things are beginning to emerge from Sun about what Java7 may include when its formal release contents are finally made public.

On Being a Software Architect by Mark Richards

One way to stop a conversation dead while at a party or gathering is to mention you are a software architect. Why? Because it takes about an hour (complete with Powerpoint slides) to explain what you do for a living. By then the person you are talking to is so bored they would rather sit in a corner licking nine-volt batteries. The problem is that no one inside or outside our industry really knows what a software architect is or what they do. In this highly interactive (and slightly humorous) session we will take a deep dive into the role a software architect plays in the IT industry. We will explore the characteristics an architect needs to have, and the elements that make a good architect and a bad architect. Through amusing antidotes and real-world examples, we will see how to become an effective software architect and help shape the industry in terms of the role and title of software architect. **Prerequisite:** *None*

Common AntiPatterns and How To Avoid Them by Mark Richards

In the book "97 Things Every Software Architect Should Know" (O'Reilly, 2009) I wrote about the importance of design patterns as a useful means of communication between architects and developers. Equally

important to patterns is an understanding of AntiPatterns - things that we repeatably do that produce negative results. AntiPatterns are used by developers, architects, and managers every day and are one of the main factors that prevent progress and success. In this session we will look at some of the more common and significant development and architecture antipatterns. Through coding and design examples, you will see how these antipatterns emerge, how to recognize when the antipattern is being used, and most importantly, how to avoid them. By attending this session, you will be part of a movement to reduce the AntiPattern catalog from hundreds of entries to only a few. **Prerequisite:** *None*

Transaction Pitfalls and Strategies by Mark Richards

In previous years I have given sessions related to my book "Java Transaction Design Strategies", where I have reviewed the basics of programmatic and declarative transactions and outlined the basic patterns described in the book. In this new session for 2009 I will focus on some of the pitfalls encountered while dealing with transactions and then how to develop an effective transaction strategy. I will start this session by describing and illustrating some of the common pitfalls I continue to see in both Spring and EJB. I will then describe four common transaction strategies you can use and implement, including a transaction strategy for high-speed transactions, a transaction strategy for client orchestration, a transaction strategy for use with API's, and finally a strategy for highly concurrent environments. Note: This session assumes you know a little bit about transactions and have been using them in either Spring or EJB. It is not intended to be an introductory session on how transactions work. You can obtain a free PDF download of my transaction book at <http://www.infoq.com/minibooks/JTDS> to quickly come up to speed with transactions. **Prerequisite:** *Java, Spring or EJB; some knowledge of transactions and JTA.*

Introduction to JMS by Mark Richards

There's no doubt about it - messaging is quickly becoming a standard part of most application architectures, particularly as more and more companies struggle to find ways to integrate heterogeneous environments due to mergers, acquisitions, or to streamline existing application portfolios. The Java Message Service (JMS) API allows Java applications to implement messaging using a standard API, therefore removing the dependency of any particular messaging provider. In this introductory session we will take a look at the basics of messaging and the JMS API. I will start by discussing the different messaging models, the structure of a basic JMS message, and the JMS API interfaces and how they interrelate. Then through interactive coding I will show the basics of sending and receiving messages using the point-to-point messaging model and how to do request/reply processing. NOTE: this session is meant to be an introduction to messaging and JMS - no prior JMS or messaging experience is needed for this session. **Prerequisite:** *None*

Advanced Topics in JMS by Mark Richards

This session covers some of the more advanced features of JMS messaging, and is intended for those who are familiar with JMS and messaging in general. Some of the topics I will be covering in this session include message grouping (where I will demonstrate sending a large JPG image using messaging), transacted sessions, client-based acknowledgement, and some various messaging design considerations and things to watch out for from a design and coding perspective. I will be doing live coding demonstrations to illustrate the techniques described in this session. Although this session is entirely JMS provider agnostic, I will be using ActiveMQ, a popular open source JMS provider, during the live coding demonstrations. **Prerequisite:** *Some knowledge of messaging and JMS would be helpful*

JavaScript: the Good, the Bad, and the Ugly by Nathaniel Schutta

Thanks to Ajax, JavaScript is cool again and developers are taking a second look at this much maligned language.

Making Web Apps Suck Less by Nathaniel Schutta

We've all used web applications that had us screaming at their creators - unfortunately sometimes we're the ones being cursed. Believe it or not, there are some simple steps we can take to ensure that our users have a great experience.

Seven Habits of Highly Dysfunctional Teams by Nathaniel Schutta

Being on a high performing team is a transcendent experience - unfortunately, many of us find more dysfunction than function. In this talk, we'll take a look at some of the common issues that face teams and discuss some ways of working towards a happy crew.

Hacking Your Brain for Fun and Profit by Nathaniel Schutta

The single most important tool in any developers toolbox isn't a fancy IDE or some spiffy new language - it's our brain. Despite ever faster processors with multiple cores and expanding amounts of RAM, we haven't yet created a computer to rival the ultra lightweight one we carry around in our skulls - in this session we'll learn how to make the most of it. We'll talk about why multitasking is a myth, the difference between the left and the right side of your brain, the importance of flow and why exercise is good for more than just your waist line.

Building External DSLs by Venkat Subramaniam

Domain Specific Languages (DSLs) are languages targeted at a particular problem and domain. They have context and are fluent. They help users of applications at various levels to easily communicate with your application. Developing DSLs, however, are not easy. You could easily get dragged into using parsers and tools with steep learning curve.

Effective Java by Venkat Subramaniam

Java is a well established language, that has been around for more than a decade. Yet, programming on it has its challenges. There are concepts and features that are tricky. When you run into those, the compiler is not there to help you.

Programming Scala by Venkat Subramaniam

Scala is a static fully object-oriented, functional language on the JVM. While taking advantage of the functional aspects, you can continue to make full use of the powerful JVM and Java libraries.

Cleaning up Code Smell by Venkat Subramaniam

Projects often start out simple, but soon become complex and turn into a lose cannon. Organizations are struggling to maintain and evolve software. Poor code quality is a significant part of that problem. Improving the quality of code is critical to success of enterprise projects.

Testing with dependencies by Venkat Subramaniam

Testing is a key ingredient to the success of a project. However, testing becomes awfully hard when your application deals with dependencies and that is often the reality.

Tackling Concurrency on the JVM by Venkat Subramaniam

In this presentation we will take a quick walk through the issues with concurrency and how the solutions provided in Scala and Clojure help address those.

Third time's a charm: What's new in Spring 3.0 by Craig Walls

In this session, I'll lead a guided tour through the latest that Spring has to offer. Whether you're a Spring veteran or a Spring newbie, there will be something new for nearly everyone.

Beyond JUnit: Powertools for Test-Driven Development by Craig Walls

Writing tests is more than just writing JUnit test cases and hoping that they'll pass when your project is built. If you want assurance that your code is sound and provides the desired functionality, then you'll want to test it from every angle and run those tests as frequently as possible.

WAR and Pieces: Breaking down monolithic web applications with Spring-DM and OSGi by Craig Walls

In this session, we'll explore modular web application development using Spring-DM and OSGi. I'll dispel the myth that OSGi is hard and show you tips and tricks that make Spring-DM and OSGi development easy.

That old Spring magic has me in its SpEL: DI Wizardy with the Spring Expression Language by Craig Walls

Spring 3.0 introduced the Spring Expression Language (SpEL), an extremely powerful yet succinct way to wire non-trivial values into Spring beans. In this presentation, we'll explore SpEL in great detail and see how SpEL opens up a whole new realm of bean wiring possibilities.