Foreword

J2EE DEVELOPMENT IS HARD. I've been developing with Java for the past six years, and there are areas of the enormous API that I haven't touched. J2EE covers security, distributed transactions, persistence, messaging, web services, interoperability, and much, much more. Enterprise Java development doesn't mean coding only with this API, but rather involves the entire suite of Java editions. J2EE layers on top of the standard edition (J2SE). And to make your head spin even further, J2ME entered the rapidly growing mobile world.

To exacerbate matters, our industry still reels from the great .com shakeout. Like many of you, I experienced the joy of being dot bombed…*twice*. We rise from the ashes to contend with more complexity and fewer resources than ever. Our teams are smaller and budgets are tighter. Failure is not an option. Do not despair, however.

In fact, times have never been better! Smaller teams allow us to act and react more quickly. Attention to fiscal responsibility means that we focus on client satisfaction and quality workmanship. From our need to simplify, the agile methodologies speak to us at an instinctive level. "The hard and stiff will be broken. The soft and supple will prevail."[1] We create *soft*ware. As such, it is malleable. Refactoring keeps code clean as requirements change. As code evolves, keeping top-notch quality demands that we test. Testing addresses complexity, partly. We can focus on a single unit at a time, test it thoroughly, and build upon it with confidence.

Kent Beck said, "Any program feature without an automated test simply doesn't exist," which speaks to the importance of confidence and another facet, automation.[2] Repetitive tasks are vital, such as a heart beating. Such repetitive tasks, though, should be effortless. The failure of such tasks elicits grave concern. Effortless, yet unmistakably vital—otherwise the complexity would overwhelm us and suffocate our projects.

Open Source factors into both the testing and automation realms of Java development. JUnit and Ant are the de facto standards, and are built into all modern development environments. These projects, and the others discussed in this book such as Struts, Hibernate, and XDoclet represent more than just free software to use. Communities are thriving around these projects, and you benefit from the collective contributions of individuals around the world. Besides addressing the problems of a limited budget, Open Source addresses the complexity and quality

[1] *Tao Te Ching.*

[2] Beck, Kent. *Extreme Programming Explained: Embrace Change* (Addison-Wesley, 1999) p. 57.

issues from several angles. Common problems generally have open source alternatives; avoid the reinvention of the wheel and concentrate on adding business value, not plumbing. Many Open Source projects have comprehensive test suites, thereby relieving you of reliability concerns. And when the inevitable bug appears in an Open Source library, or a need for enhancement, robust test suites make opening the hood a real joy. Tune in to the mailing lists, and other avenues of collaboration including wikis and blogs, of the Open Source projects you leverage. It's commonplace that questions are answered within minutes, putting to shame most commercial support contracts.

All of this philosophizing leaves us wondering what to do next. We have real work to do, and real deadlines to meet. Deriving from a Greek word meaning "to do," *pragmatic* is what we must be. As pragmatic programmers, books are a fundamental part of our craft. The book you're holding contains many gems of wisdom gleaned from hard work. There's no substitute for experience, but it would be foolish not to learn from the experiences of others. I prefer practical examples and the voice of experience on technical topics, and this book provides both. Chris and Brian have made one of life's greatest sacrifices, the authoring of a book. I commend them for their effort, and applaud them for the quality.

—Erik Hatcher
Coauthor of *Java Development with Ant* (Manning 2003) and *Lucene in Action* (forthcoming from Manning 2004)
Member, Apache Software Foundation
Committer, Ant, XDoclet, Lucene, and Tapestry