Foreword

In the 1990s, when Java was in its infancy, learning the Java class libraries involved studying a handful of classes in four or five packages. The Java class libraries have grown in size and complexity, presenting a significant problem to developers wishing to learn Java today. Just like Java, the Eclipse platform has necessarily grown over the years, and therefore considerably more time and effort is required to learn Eclipse 3.1 than its predecessors. One of the principles of the Eclipse platform is that a plug-in should integrate seamlessly with the workbench and with other plug-ins. To achieve seamless integration, it is necessary for plug-in developers to understand the best practices, conventions, and strategies related to building software for Eclipse. *Eclipse: Building Commercial-Quality Plug-ins* covers everything you need to know to develop Eclipse plug-ins of which you will be proud.

Through the development of a **Favorites** plug-in, the Eclipse Standard Widget Toolkit (SWT) and JFace frameworks are thoroughly discussed, teaching you how to build professional-looking user interfaces such as views, editors, preferences pages, and dialogs. In addition to stock-in-trade subjects, such as user-interface design, lesser-understood Eclipse topics (for example, building features and product branding) are extensively covered, as well as the best discussion I have seen on using Ant to build a product from a single source that targets multiple versions of Eclipse.

Java developers new to Eclipse often have difficulty understanding the extension point mechanism and the critical link between a plug-in's declarative manifest and the Java code necessary to implement a plug-in's functional behavior. This book serves as a roadmap to using the Plug-in Development Environment (PDE) and the extension points defined by the Eclipse platform. It also provides the missing link that developers need to understand them aspects of a plug-in that should be described in the manifest, how to develop a plug-in using existing extension points, and how to contribute which other developers may further contribute.

When I first saw CodePro, I was both impressed with the productivity gains it brought to Eclipse and the extent to which its plug-ins integrated with the Eclipse platform. Having used CodePro for a while, it has become a part of my development toolkit that I cannot do without. By drawing on their extensive experience gained while developing CodePro, Eric and Dan have done an excellent job of capturing in this book those aspects of plug-in development necessary to create a high-quality and professional-looking Eclipse product.

—Simon Archer