Stimmen zur vorangegangenen Auflage:

„Es wurden alle Themen der Programmentwicklung im Zusammenhang mit Eclipse angesprochen."

Prof. Dr. Jobst Hoffmann, FH Aachen

„In einer übersichtlichen Art und Weise wird beschrieben was Eclipse ist, wie es arbeitet und was man alles damit tun kann. Die Tipps und Hinweise sind gut, aber nicht zu zahlreich für den Titel gestreut."

Amazon.de, 4/2005

„Dieses Buch beschreibt knapp, wie man mit der Java-Entwicklungsumgebung(!) Eclipse allgemein Java lernt. Wer genau das will: kaufen."

Amazon.de, 3/2005

Aus dem Bereich IT erfolgreich gestalten

Visual Basic .NET mit Methode
von Heinrich Rottmann

Warum ausgerechnet .NET?
von Heinrich Rottmann

Requirements Analysis realisieren
von Karl Scharber

Cobit kompakt und verständlich
von Wolfgang Goltsche

Management der Softwareentwicklung
von Carl Steinweg

Das neue PL/I
von Eberhard Sturm

Projektmanagement der SW-Entwicklung
von Werner Mellis

Profikurs ABAP®
von Patrick Theobald

SAP R/3® Kommunikation mit RFC und Visual Basic
von Patrick Theobald

Six Sigma in der SW-Entwicklung
von Thomas Michael Fehlmann

User interface-orientierte Softwarearchitektur
von Paul Chlebek

Erfolgreiche Datenbankanwendung mit SQL3
von Jörg Fritze und Jürgen Marsch

Web-basierte Systemintegration
von Harry Marsh Sneed und Stephan Henry Sneed

Terminalserver mit Citrix Metaframe XP
von Thomas Joos

Exchange Server 2000
von Thomas Joos

Profikus PHP-Nuke
von Jens Ferner

Unternehmensweites Datenmanagement
von Rolf Dippold, Andreas Meier, Walter Schnider und Klaus Schwinn

Netzarchitektur-Kompass für die Realisierung
von Thomas Spitz, Markus Blümle und Holger Wiedel

SIP – Die Technik
von Andreas Kanbach

IT-Sicherheit – Make or Buy
von Marco Kleiner, Lucas Müller und Mario Köhler

Mehr IT-Sicherheit durch Pen-Tests
von Enno Rey, Michael Thuman und Dominick Baier

IT-Risiko-Management mit System
von Hans-Peter Königs

IT-Sicherheit mit System
von Klaus-Rainer Müller

Der IT Security-Manager
von Heinrich Kersten und Gerhard Klett

ITIL Security Management realisieren
von Jochen Brunnstein

Profikurs Eclipse 3
von Gottfried Wolmeringer und Thorsten Klein

www.vieweg.de

Gottfried Wolmeringer
Thorsten Klein

Profikurs Eclipse 3

Mit Eclipse 3.2 und Plugins professionell Java-Anwendungen entwickeln – Von UML bis JUnit

2., verbesserte und erweiterte Auflage

Mit 81 Abbildungen

vieweg

Bibliografische Information Der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

Höchste inhaltliche und technische Qualität unserer Produkte ist unser Ziel. Bei der Produktion und Auslieferung unserer Bücher wollen wir die Umwelt schonen: Dieses Buch ist auf säurefreiem und chlorfrei gebleichtem Papier gedruckt. Die Einschweißfolie besteht aus Polyäthylen und damit aus organischen Grundstoffen, die weder bei der Herstellung noch bei der Verbrennung Schadstoffe freisetzen.

1. Auflage 2004
2., verbesserte und erweiterte Auflage Dezember 2006

Lektorat: Sybille Thelen / Andrea Broßler

Der Vieweg Verlag ist ein Unternehmen von Springer Science+Business Media.
www.vieweg.de

Konzeption und Layout des Umschlags: Ulrike Weigel, www.CorporateDesignGroup.de
Umschlagbild: Nina Faber de.sign, Wiesbaden
Druck- und buchbinderische Verarbeitung: MercedesDruck, Berlin
Printed in Germany

ISBN 978-3-8348-0007-7

Vorwort

In diesem Buch geht es nicht darum, Java zu lernen, es wendet sich vielmehr an den erfahrenen Javaprogrammierer, und das Anliegen des Buches ist es zu zeigen, wie man Javaprojekte mit Eclipse erfolgreich bewältigt.

In diesem Zusammenhang werden, neben Eclipse, weitere Werkzeuge eingesetzt. Daher reicht es hin und wieder nur zu kurzen Einführungen z.B. wie zu Ant, CVS oder Jalopy. Schließlich handelt es sich dabei um mächtige Werkzeuge, für deren professionellen Einsatz man fundiertes Fachwissen benötigt, das selbst wiederum ganze Bücher füllt. Es werden Online-Dokumentationen genannt, zu denen Sie Links auf der Internet-Seite zu diesem Buch (`http://www.eclipse3.de`) finden und sie herunterladen können.

Eclipse hat im Laufe seines noch jungen Lebens eine beispiellose Karriere hinter sich gebracht. Das ist um so erstaunlicher, als man nicht gerade behaupten kann, es herrschten ideale Bedingungen für die Entstehung neuer Entwicklerwerkzeuge (IDEs) vor. Der Marktführer hat vor wenigen Jahren sein neues DotCom-Konzept (.com) platziert, und gute Entwicklerwerkzeuge für Java gibt es wie Sand am Meer.

Java hat ebenfalls Furore gemacht als Programmiersprache, allerdings bereits Jahre, bevor Eclipse aus der Taufe gehoben wurde. Die objektorientierte Programmierung wurde, nach langen Jahren des Nischendaseins, durch Java flächendeckend verbreitet. Dabei ist Java im Grunde nicht mehr als eine Weiterentwicklung der C/C++ Linie und wurde dadurch bei Einführung von den zahlreichen C-Veteranen auch schnell akzeptiert. Eine bekannte Syntax allein hilft nur bedingt, um ein neues Paradigma zu erlernen. Daher kam es zu allerlei Problemen. Es dauerte eine geraume Zeit, bis aus C-Hackern brauchbare Objektfachleute wurden, und die an sich kurze C-Syntax ist bei Java doch sehr geschwätzig: Wenn C++-Programme an Kurzgeschichten erinnerten, so wurden aus Javaprogrammen Romane. Hinzu kam, Sun's GUI ist nicht gerade schnell. Bereits kurz nach der Markteinführung von Java zeigte sich: Java war nichts für den Standardquelltexteditor, wie Emacs oder vi, es mussten dicke Entwicklerwerkzeuge her, die den Javacode mit seinen endlosen Objektqualifizierungen, Wrapperklassen und bildschirmfüllenden Castings handhabbar machten. Java allein ist auf Grund der kurzen C-Syntax eine relativ kleine Sprache. Die ungeheuer umfangreichen Packages lassen das Entwicklersystem Java unhandlich erscheinen. Damit stand fest, das perfekte Entwicklerwerkzeug für die Javaumgebung musste Java entschärfen, um erfolgreich zu sein. Eclipse ist dies mit sei-

nen zahlreichen Hilfefunktionen, der Codecomplemention, dem Plugin-Konzept und der Ergänzung von AWT und Swing durch JFace und SWT ausgesprochen gut gelungen. So dass man heute sagen kann: Wenn Java-Entwicklung, dann mit Eclipse. Genau dabei will dieses Buch seine Unterstützung anbieten.

Das Buch soll dem erfahrenen Javaentwickler das Wissen liefern, um mit Eclipse professionelle Javaprojekte durchzustehen. Deshalb liegt im Mittelpunkt der Betrachtung der Entwicklungsprozess selbst. Trotzdem spricht es nicht nur den Projektleiter an, sondern vor allem den Programmierer. Denn es zeigt sich mehr und mehr, dass die Projektverantwortung sowie der Test teilweise in die Hände des Entwicklers gelegt werden müssen, damit ein Javaprojekt erfolgreich werden soll. Eclipse bietet dafür eine solide Basis.

Zum Grundkonzept des Buches gehört, dass ein Projekt exemplarisch auf Eclipse3 durchgesprochen wird. Es ist ein größeres Projekt, das nicht vollständig gezeigt werden kann, weil es einfach zu umfangreich ist. Wer sich die Beispiele über die kurzen Quellkodesequenzen hinaus anschauen möchte, findet sie auf der zum Buch gestalteten Homepage `http://www.eclipse3.de`.

Im Buch wird sowohl Extreme Programming angesprochen als auch die übliche Arbeit im Team, mit den entsprechenden Entwicklungsphasen. Somit kann jeder Leser, ob Programmierer, Informatiker oder Teamleiter, seinen Nutzen aus der Lektüre des Buches ziehen. Am Ende werden Sie mir hoffentlich Recht geben und sagen, mit Eclipse ist für Java die Sonne nicht unter-, sondern aufgegangen.

Kusel, im Juni 2004, Gottfried Wolmeringer

Vorwort zur 2. Auflage

Die zweite Auflage des Buches behandelt Eclipse 3.1 und die Neuerungen, die damit verbunden sind. Trotz der kleinen Änderung in der Versionsnummer ist doch sehr vieles verbessert worden.

Das Buch ist daher vollständig überarbeitet und an einigen Stellen auch erweitert worden. Einige Plugins werden mittlerweile nicht mehr weiterentwickelt, und andere sind hinzugekommen. Auch Java ist inzwischen in der Version 5 erschienen. Diesen Aspekten habe ich versucht Rechnung zu tragen.

Dennoch schreitet die Entwicklung rasant voran, und die nächste Eclipseversion steht schon vor der Tür. Daher kann dieses Buch nicht immer die aktuellsten Versionen zeigen, und ich verweise Sie auf die Quellen im Internet, insbesondere die Homepage zum Buch (`http://www.eclipse3.de`). Dort werden wir neben Errata auch aktuelle Hinweise und Quellenangaben pflegen und eine Diskussionsplattform bieten.

Das Buch kann sowohl als Nachschlagewerk als auch als Lesebuch verwendet werden. Die ersten beiden Kapitel bieten dem Anfänger einige Grundlagen zu Java und Eclipse, insbesondere der Installation. Das dritte Kapitel richtet sich an die Ungeduldigen und ist als Schnelleinstieg gedacht. Wir greifen jedoch alle wichtigen Punkte in den späteren Kapiteln wieder auf und vertiefen diese gegebenenfalls. Sie verpassen daher nichts Wichtiges, sollten Sie das Kapitel zunächst überspringen.

Danksagungen

Ein Buch entsteht nicht aus dem Nichts, und so bin ich froh, auf die gute Arbeit von Gottfried Wolmeringer zurückgreifen zu dürfen. Er ist es, der die erste Auflage zu einem großen Erfolg gemacht hat, und ich hoffe, daran anknüpfen zu können. Ich möchte die Gelegenheit nutzen, mich an dieser exponierten Stelle bei denjenigen zu bedanken, die zum Gelingen des Buches beigetragen haben. Neben den netten Kollegen des Vieweg-Verlages, Dr. Reinald Klockenbusch, Andrea Broßler und all den anderen, die im Hintergrund arbeiten, natürlich auch denjenigen, die mir direkt oder indirekt geholfen haben. Besonders zu nennen wären da mein Bruder, der mir in turbulenten Zeiten Infrastruktur zur Verfügung gestellt hat, mein Freund Meik, welcher in vielen Stunden der Diskussion die Lesbarkeit des Buches verbessert

hat, und vor allem Kirsten, die mich stets ermutigt und motiviert hat, wenn mein Durchhaltevermögen zu schwinden drohte.

Minden, im August 2006, Thorsten Klein

Konventionen

Verwendete interne Abkürzungen

Um den Text nicht durch extrem lange Fachbegriffe zu belasten, wird eine Handvoll Abkürzungen eingesetzt, die häufig verwendete lange Wortmonster ersetzen (siehe Tabelle 1).

Tabelle 1: Die im Text verwendeten Abkürzungen

Abkürzung	Bedeutung
BO	Business-Objekt
IDE	Integrierte Entwicklungsumgebung /Entwicklerwerkzeug
LMT	Linke Maustaste
RMT	Rechte Maustaste

Menüs und Schaltflächen

Die Anweisungen zum Aufruf vom Menüs und Schaltflächen sind folgendermaßen aufgebaut: Der Aufrufpfad wird durch das Zeichen „>“ getrennt und durch kursive Schrift gekennzeichnet.

Menüpunkt1>Menüpunkt2>Auswahl l>Button1>Button2

Spezielle Tasten

Sind ganz bestimmte Tasten erforderlich, werden sie in Spitzklammer, entweder als Abkürzung oder im Klartext gesetzt.

<Strg>

<Leertaste>

Aufgaben im Text

Viele der praktischen Beispiele sind als Aufgaben eingefügt. Dem Leser bietet sich dadurch die Möglichkeit, das soeben erworbene Wissen durch Übungen zu untermauern. Wenn er sich die Aufgabenstellung anschaut, das Buch zur Seite legt und es selbst versucht, erhöht sich der Lerneffekt. Der Leser, der nicht ganz so experimentierfreudig ist, findet direkt unter der Aufgabe jeweils die Lösung. Er kann daher ungehindert weiterlesen. Ausführlichere Codebeispiele findet man zusätzlich auf der Webseite.

Inhaltsverzeichnis

1 Neuerungen in Eclipse 3.1 und 3.2 1

2 Vor der Installation 3

2.1 Warum Java? 3
2.2 Warum Eclipse? 4
2.3 Die Quellen 5
2.4 Systemanforderungen 5

3 Die Installation 9

3.1 Downloads 9
3.2 Carlipso 18

4 AdHoc zum ersten Ergebnis 19

4.1 Es geht los 19
4.2 Programme starten 25
4.3 Programmdaten übernehmen 27
4.3.1 JDK-Projekte und einzelne Dateien übernehmen 27
4.3.2 Ganze Projekte von fremden IDEs übernehmen 28
4.3.3 Andere Eclipse Projekte übernehmen 29
4.3.4 Änderungen an der Paketstruktur bestehender Projekte 30

5 Das Eclipse Prinzip 33

5.1 Die Bausteine 33
5.2 Geburt eines Projekts 36

5.2.1 Neues Projekt anlegen 37
5.2.2 Working-Sets 39
5.2.3 Projekt löschen 40
5.3 Der Aufbau 40
5.3.1 Die Perspektiven 42
5.3.2 Die Menüleiste (Menubar) 45
5.3.3 Die Navigatorview 58
5.3.4 Die Editoren 60
5.3.5 Die Outlineview 67
5.3.6 Die Taskview (Bookmarks, Errorview) 69
5.3.7 Die Perspektivbar (Shortcutleiste - Shortcut Area) 69
5.3.8 Die Toolbars 69
5.3.9 Die Registerleisten 69
5.3.10 Die Statusbar 70

6 Versionsverwaltung mit Eclipse 71

6.1 Grundlagen der Versionsverwaltung 72
6.1.1 Das Repository 72
6.1.2 Versionierung 74
6.2 Entwicklungszweige (Branches) 78
6.3 Eclipse und CVS 78
6.3.1 Installation von CVS 78
6.3.2 CVS bei der Arbeit 81
6.3.3 CVS mit Eclipse nutzen 83
6.4 Subversion 86
6.4.1 Subversion installieren und einrichten 88

7 Das Design 91

7.1 Ein Wort zum Softwareengineering 91
7.2 Vorgehensmodelle 93
7.3 Modelle des Anwendungssystems 95
7.3.1 Dynamisches und statisches Modell 95
7.3.2 Bestandteile der Modelle 96

7.4 Vom Konzept zum Vorgehen ... 98
7.5 Analyse ... 100
7.5.1 Diagramme erzeugen ... 104
7.5.2 Komponentendiagramm ... 121
7.5.3 EMF ... 124
7.5.4 Datenbankdiagramme ... 125
7.6 Arbeiten mit Datenbanken ... 128

8 Die grafische Anwenderschnittstelle ... 131

8.1 Rapid Prototyping ... 131
8.1.1 Den Visual Editor (VE) einsetzen ... 132

9 Die Kodierung ... 143

9.1 Extreme Programming ... 143
9.1.1 Sequentielles Arbeiten ... 149
9.1.2 Arbiträres Arbeiten ... 152
9.1.3 Codeformating ... 156
9.1.4 Jalopy, das Beautifier-Plugin ... 157

10 Die Testphase ... 161

10.1 Warum testen? ... 161
10.1.1 ScrapBook Page ... 162
10.1.2 Main-Tests ... 165
10.1.3 Zusicherungen ... 165
10.1.4 Sysout ... 167
10.1.5 Debugger ... 167
10.1.6 JUnit (JDT) ... 170
10.1.7 Log4J ... 172
10.1.8 Test & Performance Tools Platform (TPTP) ... 172

11 Entwicklung von eigenen Plugins 177

11.1 Eine neue Art Javaprogramm....177
11.1.1 Die Buildphase mit ant....182

12 Anhang..... 185

12.1 TODO..... 185
12.1.1 API-Dokumentation verfügbar machen....185
12.1.2 Formatieren des Quellcodes..... 186
12.1.3 Darstellung auf das aktuelle Projekt beschränken....186
12.1.4 Kommentare mit verschiedenen Task-Prioritäten versehen....186
12.2 Die Eclipse Start-Parameter....189
12.3 Tastaturbefehle..... 192
12.4 Subversion für CVS-Nutzer..... 193
12.4.1 Subversionkonfiguration....193
12.4.2 Andere Revisionsnummern..... 194
12.4.3 Versionierung von Verzeichnissen....194
12.4.4 Branches und Tags..... 195
12.5 Plugins..... 195
12.5.1 Installation von Plugins..... 195
12.5.2 Automatisches Update von Plugins..... 196
12.5.3 Callisto..... 196
12.5.4 Die wichtigsten Plugins..... 197
12.6 Literatur..... 198
12.6.1 URL- und Literaturnachweis..... 198
12.6.2 Literaturverzeichnis..... 198
12.6.3 Webseiten zum Thema..... 199

Glossar..... 201

Sachwortverzeichnis..... 215

1 Neuerungen in Eclipse 3.1 und 3.2

Der Unterschied der Versionsnummern von Eclipse 3.0 zu 3.2 mag dazu verleiten keine großen Neuerungen zu vermuten, doch ist dies weit gefehlt.

Würden viele Nutzer wahrscheinlich die Unterstützung von Java 5.0 als die große Neuerung in Eclipse 3.1 und die von Java 6.0 als in Eclipse 3.2 nennen, so war den Entwicklern eine gesteigerte Performance besonders wichtig. Dies ist auch nicht weiter verwunderlich, da doch alle Anwender davon profitieren. Die Eclipse-Entwickler haben dabei eine Reihe von Optimierungen durchgeführt, die sich durch fast alle Bereiche der Workbench ziehen. Das Tuning betrifft neben der deutlich verbesserten Startgeschwindigkeit auch den Hauptspeicherbedarf, genauso wie die Geschwindigkeit beim Wechsel von Workspaces und auch eine verbesserte Skalierbarkeit. Es soll dem Anwender auch unter extremen Anforderungen eine ausreichende Verarbeitungsgeschwindigkeit zur Verfügung stehen, so der Wunsch der Eclipse-Entwickler. Dies ist sicher nicht generalisierbar, da nicht zuletzt eingesetzte Hardware und persönliches Nutzungsverhalten eine entscheidende Rolle spielen, doch ist die Tendenz wohl eindeutig zu spüren.

Eine wichtige Grundlage für diese Tuning-Maßnahmen haben die Entwickler durch Einführung der Klasse *Performance-Stats* in Eclipse 3.1 erreicht, die eine einfache Schnittstelle für Messungen innerhalb der Eclipse-Anwendungen zur Verfügung stellt. Die neuen Performance-Views oder entsprechende Plugins können beispielsweise den Speicherverbrauch ohne externe Profiler visualisieren. So konnte der Start der Workbench durch Optimierungen am JDT-Core um etwa 50% beschleunigt werden. Auch Einlesen und Wechsel zwischen verschiedenen Workspaces lag den Entwicklern am Herzen, so dass dies in der aktuellen Version bis zu zehnmal schneller geworden ist. Auch in der Version 3.2 hat man diesen Weg konsequent weiter verfolgt und die Performance für Enterprise-Anwendungen weiter verbessert.

Eine der großen Performancebremsen in Eclipse 3.0 war der modulare Aufbau und die damit einhergehende einfache Erweiterbarkeit. Eine Vielzahl von installierten Plugins mit zahlreichen Views und Perspektiven zwang Eclipse in die Knie. Es war daher von besonderer Bedeu-

tung, Eclipse auch unter diesen Bedingungen zu einer guten Arbeitsgeschwindigkeit zu verhelfen.

Doch auch die Spracherweiterung zu Java 5.0 hinterlässt einen guten Eindruck, auch wenn Eclipsenutzer hier länger als andere darauf warten mussten. Dies ist jedoch nicht verwunderlich, wenn man die gravierenden Änderungen bedenkt, die an der gesamten Eclipse-Basis dazu notwendig waren. Dies betrifft vor allem das Build-Management, den inkrementellen Compiler und die Refactoring-Funktionen, aber auch den Editor mit Formatierungs- und Programmierhilfen. Daher hat mit mit der Version 3.2 schon jetzt die Unterstützung vieler Java 6.0 Features eingebaut, obwohl SUN das Release erst für Oktober geplant hat.

Eine Java 5.0/6.0 Laufzeitumgebung ist jedoch nicht zwingend notwendig, um die neuen Möglichkeiten zu nutzen. Es reicht aus, die Projekteinstellungen auf Java 5.0 oder 6.0 zu ändern, und die entsprechenden Spracherweiterungen stehen zur Verfügung. Soll ein solches Projekt jedoch ausgeführt werden, so ist eine aktuelle Javaversion hilfreich, doch mit einigen Tricks kann das auch mit JRE 1.4 realisiert werden. Der Compiler und auch der Editor unterstützen nun generische Datentypen (Klassenschablonen oder Generics), so dass sowohl Codeformatierung als auch Syntaxeinfärbung korrekt funktionieren. Views und Dialoge wurden ebenso angepasst wie die Programmierhilfen *Code Assist, Code Select* und *QuickFix*.

Erwähnenswert ist noch die Unterstützung der erweiterten Schleifen. Im Fehlerfall erhält der Programmierer Verbesserungsvorschläge, die das Problem so weit eingrenzen lassen, dass der Code frei von Fehlern oder Warnungen ist. Auch „alte“ Schleifen lassen sich auf diese Weise mit Unterstützung zum neuen Typ konvertieren.

Daneben gibt es noch eine Reihe von Detailverbesserungen, die das Leben mit der Workbench erleichtern. Da ist zum einen die überarbeitete Import- und Export-Funktion zu nennen, die nun unter anderem auch mit *tar.gz*-Archiven umgehen kann, ebenso wie der überarbeitete CVS-Client, der eine Reihe sinnvoller Verbesserungen erfuhr.

2 Vor der Installation

Ich springe am liebsten, ohne zu zögern, ins kalte Wasser, Sie auch? Ganz so einfach ist es hier leider nicht. Ein paar Dinge müssen noch geklärt werden, bevor es los gehen kann. Warum wir gerade mit Java programmieren wollen und weshalb wir dazu Eclipse nehmen und welche Hardware man benötigt, bevor man „ins kalte Wasser springen" kann.

2.1 Warum Java?

Je nach Anwendungsfall gibt es Sprachfamilien, die sich für diesen besonders eignen. Funktionale, imperative oder objektorientierte Sprachen haben ihre Vor- und Nachteile. Häufig sind diese auch nicht so scharf voneinander abzugrenzen, da Sprachkonstrukte übernommen werden. Jede, zumindest mir bekannte, objektorientierte Sprache verfügt letztendlich auch über imperative Sprachkonstrukte. Es gibt innerhalb der Sprachfamilien auch noch Unterschiede zwischen kompilierten und interpretierten Sprachen. Erstere werden vor der Ausführung in Maschinencode übersetzt und letztere zur Ausführungszeit zeilenweise abgearbeitet, wobei es auch hier, natürlich, wieder Mischformen gibt.

Es sollte also eine Sprache sein, die nicht schwer zu erlernen ist, eine klare Syntax hat und sich für die meisten Anwendungsfälle gut eignet. Genau da kommt Java ins Spiel. Java ist eine objektorientierte Sprache, mit der kleine Programme für die Kommandozeile genausogut geschrieben werden können, wie große graphische Nutzeroberflächen und Anwendungen für das Internet. Die Syntax von Java ist zwar relativ einfach, als Sprache im Entwickleralltag ist der Einsatz von Java jedoch nicht trivial. Das hat dazu geführt, dass es zu kaum einer Sprache so viele Tools, professionelle Anleitungen und Schulungen wie für Java gibt. Softwareerstellung ist eben nicht das bloße Einhacken von Anweisungen in einer bestimmten Programmiersprache. Nun kann man sich fragen, warum wir Java verwenden, wenn es noch einfachere Sprachen gibt?

Zum einen sind gute Javaprogramme relativ leicht zu erstellen, wenn man die Sprache erst einmal leidlich beherrscht. Das kommt daher, weil man Zeigerarithmetik aus Java verbannt und Makrosprachen erst gar nicht hereingelassen hat. In Javaprogrammen gibt es daher nur eine

Syntax. Zum anderen hat der Hype um diese Sprache soviel freien Quellcode (im Sinne von Open Source) und freie Werkzeuge hervorgebracht, dass man mit keiner zweiten Sprache so kostengünstig professionelle Programme für ein weites Aufgabenspektrum erstellt.

2.2 Warum Eclipse?

Gute Lesbarkeit von Quellcode, Fehlerfreiheit und Stabilität, das alles verträgt sich nicht zwingend mit der Forderung Software schnell und individuell zu erstellen. Einen großen Teil zur Erfüllung dieser Ansprüche muss heute die Entwicklungsumgebung leisten. Dadurch besteht die Gefahr, dass sie selbst groß und unhandlich wird.

So ist Eclipse bisher eine der wenigen IDEs, die in Java geschrieben wurde und trotzdem auf einem Rechner mit dürftiger Hardwareausstattung verwendbar ist. Zum einen liegt es an dem hohen Grad der Modularität, zum anderen daran, dass man statt AWT oder Swing eigene GUIs (SWT und JFace) entwickelt hat. Dazu muss man wissen, dass AWT sowie Swing grafische Oberflächen sind, die einen größtmöglichen gemeinsamen Nenner der Oberflächen aller unterstützten Betriebssysteme darstellen. Hinzu kommt, dass sie eine Oberfläche integrieren, die mit sich verändernden Elementen arbeitet, wenn man die Größe einer Dialogbox oder eines Fensters ändert. Dadurch wurden Javaprogramme groß, unakzeptabel langsam und bekamen einen schlechten Ruf. Mit der Entwicklung von Eclipse ist man einen anderen Weg gegangen. Statt AWT oder Swing zu verwenden, hat man eigene GUIs entwickelt, die speziell für das jeweilige Betriebssystem zusammengestellt wurden. Dabei ruft SWT direkt die API-Funktionen der Windows-Oberfläche auf (näheres zu SWT auf der Homepage zum Buch). Damit bekommt man ein betriebssystemidentisches „Look and Feel“, wobei die Ausführungsgeschwindigkeit des Programms gegenüber den mit AWT oder Swing erstellten schneller ist. JFace ist seinerseits eine Erweiterung in Richtung eines neuen Java-Stylings auf Basis des SWT.

Durch kompatible Versionen für andere Betriebssysteme erreicht man auch für SWT und JFace Betriebssystemunabhängigkeit. Der Preis, den man zu zahlen bereit sein muss, ist die eingeschränkte Verfügbarkeit, da JFace/SWT-Pakete noch nicht für jedes Betriebssystem verfügbar sind.

Bei Eclipse selbst handelt es sich um ein OOP-Projekt der Firma IBM. Es wurde im Rahmen der „WebsphereStudio“-Workbench von der Tochterfirma OTI entwickelt. Diese Softwareschmiede hatte bereits die

Entwicklungsumgebung VisualAge programmiert, von dort wurden gute Ideen mit eingebracht.

Eclipse ist ein editorgestütztes Entwicklungstool zur Programmierung in verschiedenen Programmiersprachen. Die Akzeptanz von Eclipse war von Anfang an extrem hoch. Wegen der Vielzahl von verfügbaren Plugins für diverse Sprachen und der einfachen Erweiterbarkeit setzen nicht nur eingefleischte Javaprogrammierer dieses Werkzeug ein. Da Eclipse selbst in Java realisiert ist, steht es für eine ganze Reihe von Betriebssystemen zur Verfügung, sogar für Exoten wie QNX. Eclipse ist es auf ideale Weise gelungen, beliebig skalierbar zu sein. Eine Forderung, die sich bei Anwendersoftware nur schwer verwirklichen lässt. Nicht zuletzt ist Eclipse ein OpenSource-Produkt, was bedeutet, dass es kostenlos verwendet werden darf und der Quellcode für eigene Verbesserungen zur Verfügung steht. Insbesondere die gut dokumentierte API erleichtert es, eigene Plugins zu entwickeln, sei es für eine noch nicht unterstützte Programmiersprache oder eine interessante Implementierung des Damespiels.

2.3 Die Quellen

Die aktuellen Versionen der besprochenen Software findet man im Internet:

JDK: `http://www.javasoft.com`

Eclipse: `http://www.eclipse.org`

Plugins: `http://www.eclipse-plugins.2y.net`

Links zu diesen und anderen Adressen um Eclipse und Java finden Sie auf der Seite:
`http://www.eclipse3.de`.

2.4 Systemanforderungen

Sie sollten sich in jedem Fall eine Eclipse-Version auf Ihrem Rechner installieren, um die folgenden Ausführungen auch praktisch nachvollziehen zu können. Zur Einschätzung der notwendigen Rechnerkapazitäten hier die Systemanforderungen: Sie benötigen einen Pentium III, mit wenigstens 660 MHz und mindestens 128 MB Arbeitsspeicher. Auf der Festplatte werden etwa 120 MB für die Eclipse Basisinstallation benötigt. Für Plugins, aber vor allem für anstehende Projekte sollte ebenfalls noch genügend freier Speicherplatz vorhanden sein. Eine aktuelle JDK-Version benötigt etwa 70 MB Plattenplatz, ohne die

umfangreiche API-Dokumentation. Für die Dokumentation kann man noch einmal mit 180 MB rechnen.

Wie das im Bereich von Java üblich ist, werden fast alle gängigen Betriebssysteme unterstützt. Eclipse gibt es für Windows genauso wie für Linux, um die zwei wichtigsten zu nennen (vgl. Tabelle 2.1)

Eclipse-Plugins gibt es, außer für Java, für eine Reihe weiterer Programmiersprachen (vgl. Tabelle 2.2).

Tabelle 2.1: Eclipse gibt es für zahlreiche Betriebssysteme

System	Nähere Information
Windows 98/ME/2000/XP	http://www.microsoft.com/ http://www.eclipse.org/downloads/index.php
Linux (x86/Motif)/(x86/GTK 2)	http://www.linux.org/ http://www.eclipse.org/downloads/index.php
Solaris (SPARC/Motif)	http://www.sun.com/solaris/ http://www.eclipse.org/downloads/index.php
QNX (x86/Photon)	http://www.qnx.com/ http://www.eclipse.org/downloads/index.php
AIX (PPC/Motif)	http://www.ibm.com/servers/aix/ http://www.eclipse.org/downloads/index.php
HP-UX (HP9000/Motif)	http://www.hp.com/ http://www.eclipse.org/downloads/index.php
Mac OSX (Mac/Carbon)	http://www.apple.com/macosx/ http://www.eclipse.org/downloads/index.php

Tabelle 2.2. Eine Auswahl von Plugins für Programmiersprachen

Sprache	Nähere Informationen
C#	http://msdn.microsoft.com/vcsharp/ http://www.improve-technologies.com/alpha/esharp/

Sprache	Nähere Informationen
C++	http://www.research.att.com/~bs/C++.html http://msdn.microsoft.com/visualc/ http://www.eclipse.org/cdt/
C	http://www.lysator.liu.se/c/ http://www.eclipse.org/cdt/
Cobol	http://www.cobolportal.com/ http://www.eclipse.org/cobol
AspectJ	http://aspectj.org/ http://www.eclipse.org/ajdt/ http://www.eclipse.org/aspectj/
Goo	http://people.debian.org/~bfulgham/goo/goo_writeup/ http://people.debian.org/~bfulgham/goo/ org.googoogaga.ide.tar.gz
Pascal	http://www.freepascal.org/ http://sourceforge.net/projects/pasclipse
Eiffel	http://www.eiffel.com/ http://tom.loria.fr/
JavaScript	http://javascript.com/ http://www.mysunrise.ch/users/adinu/
Perl	http://www.perl.com/ http://e-p-i-c.sourceforge.net/
PHP	http://www.php.net/ http://sourceforge.net/projects/phpeclipse http://www.xored.com/download.php
Python	http://www.python.org/ http://sourceforge.net/projects/pyeclipse/ http://www.xored.com/products.php

Sprache	Nähere Informationen
Ruby	http://www.ruby-lang.org/en/ http://sourceforge.net/projects/rubyeclipse/

3 Die Installation

Bei Eclipse von Installation zu reden, ist maßlos übertrieben, zumindest, wenn man die üblichen Setuproutinen vor Augen hat. Irgend jemand, ich glaube es war dieser Herr mit Brille aus Redmont, meinte mal, dass die Software der Zukunft dadurch installiert wird, dass man sie auf die Festplatte kopiert. Eclipse wird so installiert und ihre Software, Herr G...?

3.1 Downloads

Um Eclipse einsetzen zu können, benötigen wir zuerst einmal die JRE (*J*ava *R*untime *E*nvironment), die bei `http://java.sun.com` erhältlich ist, und natürlich Eclipse selbst (`http://www.eclipse.org`). Beides kann man entweder direkt aus dem Internet herunterladen oder, in Ermangelung eines breitbandigen Internetzugangs, sich von einem netten Kollegen völlig legal auf einer CD-ROM geben lassen.

Wichtig sind auch einige Plugins, die im Rahmen des Buches genutzt werden. Das wären erst einmal folgende (s. Tabelle 3.1).

Tabelle 3.1: Plugins, die für die Beispiele im Buch eingesetzt werden

Plugin	Nähere Informationen
Omondo UML	`http://www.omondo.com`
Visual Editor	`http://www.eclipse.org/vep`
JUnit	Gehört zur Standardausstattung von Eclipse
Apache ant	Gehört zur Standardausstattung von Eclipse
Hinweise zu weiteren Plugins befinden sich im Anhang	

Man sollte noch wissen, dass einige Plugins mit Installer ausgeliefert werden, andere werden nur entpackt und ins Eclipseverzeichnis kopiert. Informationen zu den installierten Plugins findet man über die Menüpunkte: *Help>About Eclipse Platform>Plugin Details*. Sehr hilfreich für die Arbeit mit dem Buch ist zusätzlich die Dokumentation zu ant.

```
http://ant.apache.org/manual/index.html
```

Die Installation von Eclipse ist nicht schwierig. Die Eclipse-SDK-Zipdatei (eclipse-SDK-3.2-win32.zip) wird in das gewünschte Verzeichnis auf der Platte entpackt. Im Buch ist das *C:\Programme\Eclipse* unter Windows bzw. */opt/eclipse* unter Linux. Das ist schon alles. Unter Linux sollte man für das Verzeichnis noch die passenden Besitzer und Rechte setzen, um die Installation von Plugins zu erleichtern, aber das war es dann auch wirklich.

Abbildung 3.1: Fehlermeldung: Die virtuelle Maschine zum Starten fehlt

Alle Unterverzeichnisse aus der Zip-Datei sollten dabei erhalten bleiben. Ein Eclipse-Root-Verzeichnis namens „Eclipse“ ist in der Zip-Datei schon vorhanden. Damit zeigt sich bereits mit der Installation, dass Eclipse nicht nur neue Konzepte anbietet, sondern sie auch selbst konsequent einsetzt. Natürlich gibt es auch eine Art Plugin für eine deutsche Version von Eclipse, als sogenanntes LanguagePackFeature. Es wird wie ein Plugin installiert. Das Buch wird sich jedoch ausschließlich auf die englische Version beziehen. Wird Eclipse jetzt gestartet, gibt es eine Fehlermeldung (s. Abbildung 3.1).