HANSER

Vorwort

Rainer Oechsle

Java-Komponenten

Grundlagen, prototypische Realisierung und Beispiele für Komponentensysteme

ISBN (Buch): 978-3-446-43176-8

ISBN (E-Book): 978-3-446-43591-9

Weitere Informationen oder Bestellungen unter

http://www.hanser-fachbuch.de/978-3-446-43176-8

sowie im Buchhandel.

Vorwort und Einleitung

Der Begriff Komponente kommt vom lateinischen Wort „componere“, was so viel wie „zusammensetzen“ heißt. Eine Komponente ist also ein Teil eines zusammengesetzten Ganzen, wobei die Einzelteile nicht nur irgendwie zusammengestellt, sondern in funktional sinnvoller Weise miteinander verbunden worden sind. Wenn beispielsweise die Einzelteile eines Fahrrads wie Rahmen, Bremse, Kette usw. einfach auf einen Haufen geworfen werden, kann man bekanntlich wenig mit dem Fahrrad anfangen – es wird in diesem Fall von den Menschen noch nicht einmal als Fahrrad wahrgenommen.

Die Vorstellung von Komponenten im Software-Bereich entstand bereits Ende der 60er Jahre im Zusammenhang mit der sogenannten Software-Krise. Man hoffte damals, dass eine Software-Industrie für die Massenproduktion von Komponenten entstehen würde und dass diese Komponenten in den unterschiedlichsten Anwendungen wiederverwendbar wären. So wie Kinder mit denselben Bausteinen Häuser, Schiffe oder Flugzeuge bauen, sollte eine neue Anwendung lediglich durch das „Zusammenstecken“ vorproduzierter Software-Komponenten entwickelt werden, wobei man dazu im Idealfall nicht einmal programmieren muss. Diese Hoffnung hat sich zumindest bis jetzt nicht erfüllt. Dennoch ist Komponenten-Software immer noch ein aktuelles Thema. So kann man beispielsweise die momentan weit verbreiteten Software-Plattformen für Enterprise Java Beans oder auch Android als Komponenten-Software sehen. Die Tatsache, dass sich das Thema Komponenten-Software schon so lange hält und kein vorübergehender „Hype“ ist, wie manche andere Themen in der Informatik, zeigt, dass es wohl eine gewisse Substanz haben muss.

Analog zum Begriff objektorientierte Programmierung gibt es den Ausdruck komponentenorientierte (manchmal auch komponentenbasierte) Programmierung. Der Komponentenbegriff ist allerdings nicht ganz unproblematisch: Während es einen breiten Konsens gibt, was ein Objekt ist (bei Unklarheiten empfehle ich ganz pragmatisch, einfach mal ein Java-Objekt mit Hilfe der Serialisierung abzuspeichern und sich die abgespeicherten Daten anzusehen), ist dies für eine Komponente weit weniger klar. Eine Ursache dafür dürfte u. a. sein, dass das, was eine Komponente ist, vom verwendeten Komponenten-Framework abhängt, und davon gibt es allein im Java-Umfeld eine ganze Reihe, wie dieses Buch zeigt.

In vielen Veröffentlichungen werden Komponenten dadurch charakterisiert, dass sie ausschließlich über Schnittstellen genutzt werden können (häufig werden auch „klar definierte“ Schnittstellen angeführt, was immer das auch heißen mag) und dass sie wiederverwendbar sind. Ich will nicht bestreiten, dass es Komponentensysteme gibt, auf die diese Beschreibung

zutrifft. Allerdings halte ich sie nicht für typisch. Für manche Komponentensysteme müssen keine Java-Schnittstellen implementiert werden, sondern Klassen aus vorgegebenen Klassen abgeleitet oder alternativ Methoden mit gewissen Annotationen versehen werden. Auch die Wiederverwendbarkeit ist oftmals keine bedeutsame Eigenschaft. Eine webbasierte Anwendung mit Servlets oder eine EJB-Anwendung, welche jeweils eine Komponente für das dazugehörige Komponenten-Framework darstellen, sind häufig nicht auf Wiederverwendbarkeit ausgelegt. Es geht in den genannten Fällen viel mehr um die Nutzung der Funktionen, die vom Komponenten-Framework bereitgestellt werden und die Anwendungsentwicklung dadurch erleichtern.

In diesem Buch soll eine klarere Vorstellung davon vermittelt werden, was Komponenten-Software im Java-Umfeld bedeutet. Im Einzelnen hoffe ich, dass Sie nach dem Durcharbeiten dieses Buches die folgenden Lernziele erreicht haben:

- Sie sollen die charakteristischen Merkmale von Java-Komponentensystemen kennen. Sie sollen verstanden haben, in welchen Ausprägungen diese Merkmale in den Beispielsystemen, die in diesem Buch behandelt werden, vorkommen. Ihnen sollen damit die Ähnlichkeiten, aber auch die Unterschiede der Frameworks deutlich geworden sein.
- Sie sollen ein Grundverständnis für den Umgang mit den Frameworks dieses Buches haben und einfache Komponenten dafür entwickeln können.
- Sie sollen sich mit diesem Grundverständnis leichter in eines der hier behandelten Frameworks in größerer Tiefe einarbeiten können. Auch sollen Sie sich mit einem anderen hier nicht vorgestellten Java-Framework schneller zurecht finden, indem Sie die Prinzipien dieses neuen Frameworks identifizieren, Ähnlichkeiten mit anderen Frameworks erkennen und davon profitieren können.
- Sie sollen aber nicht nur als Anwendungsentwickler und -entwicklerin die behandelten Frameworks nutzen können, sondern Sie sollen darüber hinaus auch verstehen, wie diese Frameworks ihre Funktionen realisieren. Aus diesem Verständnis heraus sollen Sie dann auch in der Lage sein, eigene Frameworks in Java zu entwickeln.

Nachdem Sie nun die Ziele kennen, die Sie erreichen sollen, wird die Gliederung des Buches vorgestellt. Das Buch besteht aus drei Teilen:

- Teil 1: Java-Grundlagen: Im ersten Teil werden die Grundlagen der Sprache Java behandelt, deren Kenntnis für das Thema Komponenten-Software unverzichtbar ist. Zwar wendet sich das Buch an Personen, die schon einigermaßen gute Java-Kenntnisse haben sollten. Aus Erfahrung weiß ich aber, dass manchmal auch Personen, die sich schon Jahre mit Java beschäftigen, die im Teil 1 behandelten Grundlagen nicht oder nur ungenügend beherrschen. Im Einzelnen geht es um Reflection in Kapitel 2, Annotationen in Kapitel 3, dynamische Proxies in Kapitel 4 sowie um das Klassenladen und das sogenannte Hot Deployment in Kapitel 5. Da Reflection einen guten Teil komplizierter geworden ist durch Generics und ich Reflection doch einigermaßen umfassend behandeln will, wird dem Kapitel über Reflection ein Kapitel über Generics vorangestellt (Kapitel 1).
- Teil 2: Java-Komponenten: Der zweite Teil ist zwar der kleinste der drei Teile. Er bildet aber den Kern des Buches, weil darin in Kapitel 7 erläutert wird, was in diesem Buch unter einer Java-Komponente und einem Java-Komponenten-Framework verstanden werden soll. Damit sich die Leserinnen und Leser bei dieser allgemeinen Charakterisierung etwas Konkretes dazu vorstellen können, wird im Kapitel davor (Kapitel 6) ein einfaches,

selbst entwickeltes Komponenten-Framework mit einigen einfachen Beispielkomponenten vorgestellt.

- Teil 3: Beispiele für Java-Komponentensysteme: Im dritten und letzten Teil des Buches wird eine Reihe von konkreten Java-Komponentensystemen präsentiert. Es handelt sich dabei um Java Beans (Kapitel 8), OSGi (Kapitel 9), Eclipse (Kapitel 10), Applets (Kapitel 11), Servlets (Kapitel 12), Enterprise Java Beans (Kapitel 13), Spring (Kapitel 14), Ereignisbusse (Kapitel 15) und Android (Kapitel 16). Für jedes Framework werden neben einigen allgemeinen Bemerkungen konkrete Beispielkomponenten entwickelt. Dabei geht es nicht nur um das Zusammenwirken jeder einzelnen Komponente mit dem Framework, sondern auch und vor allem um das Zusammenspiel der Komponenten untereinander. Jedes Kapitel endet mit einer Bewertung, inwiefern das gerade besprochene Komponentensystem der allgemeinen Charakterisierung aus Kapitel 7 entspricht. Selbstverständlich werden Komponentensysteme in Kapitel 7 so charakterisiert, dass die Beispiele im dritten Teil des Buches einigermaßen gut dazu passen.

Nachdem Sie jetzt wissen, was der Inhalt dieses Buches ist, möchte ich auch darauf eingehen, was in diesem Buch nicht vorkommt. Wie der Titel des Buches angibt, geht es in diesem Buch ausschließlich um Komponentensysteme, die auf Java basieren. Somit wird das Beispiel, das in anderen Veröffentlichungen über Komponenten-Software immer erwähnt wird, nämlich COM/DCOM aus der Windows-Welt, hier nicht behandelt. Auch die zu einem gewissen Teil sprachunabhängigen Frameworks CORBA und Web Services, welche üblicherweise als Beispiel-Frameworks angeführt werden, sind in diesem Buch nicht zu finden. Ich habe mir die Freiheit genommen, nur Frameworks vorzustellen, die mir gefallen. Außerdem sollten Sie nicht erwarten, dass Sie nach dem Durcharbeiten des Buches für alle vorgestellten Frameworks Experte bzw. Expertin sind. Bitte machen Sie sich klar, dass es zu den meisten Frameworks jeweils separate Bücher gibt, deren Seitenumfang doppelt so groß ist wie der Umfang dieses Buches. Somit finden Sie in diesem Buch beispielsweise nichts zu der Installation der einzelnen Frameworks oder zum Vorgehen bei der Entwicklung und Installation (Deployment) eigener Komponenten für diese Frameworks mit Hilfe von Entwicklungsumgebungen wie Eclipse. Auch werden viele Funktionen, die diese Frameworks haben, verschwiegen. Das Ziel dieses Buches ist, nur das „Komponentenhafte“ jedes behandelten Frameworks herauszustellen. Insofern ist dieses Buch ein klassisches Lehrbuch: Es geht nicht um das tiefe Eindringen in ein spezielles Framework, sondern es geht darum, den Leserinnen und Lesern Grundprinzipien zu vermitteln und sie in die Lage zu versetzen, Ähnlichkeiten zwischen den Frameworks zu erkennen. Wenn Sie irgendwann beim Lesen denken: „das ist ja genauso wie bei …“, dann ist das ein durchaus beabsichtigter Effekt. Durch die Betonung der Prinzipien erhoffe ich mir eine Vermittlung von Kenntnissen, die über einen etwas längeren Zeitraum bedeutsam sind und nicht mit dem Erscheinen der nächsten Version eines bestimmten Frameworks nutzlos werden.

Das Buch wendet sich sowohl an bereits im Berufsleben stehende Java-Software-Entwicklerinnen und -Entwickler als auch an Studierende der Informatik und verwandter Studiengänge. Die Studierenden sollten nach Möglichkeit schon mindestens zwei Jahre Erfahrungen mit Java gesammelt haben. Insofern ist das Buch für Bachelor-Studierende ab dem dritten Studienjahr oder für Master-Studierende geeignet. Neben einem guten Grundverständnis der Sprache Java, das man in der Regel im ersten Studienjahr eines Informatikstudiums erwirbt, sollten insbesondere Kenntnisse im Bereich der parallelen Programmierung

(Stichwort: Java-Threads), der Programmierung grafischer Benutzeroberflächen mit Java (Stichwort: Java-Swing) und der Programmierung verteilter Anwendungen mit Hilfe des Fernmethodenaufrufs (Stichwort: Java-RMI) vorhanden sein. Es ist mir zwar etwas peinlich, aber dennoch möchte ich erwähnen, dass diejenigen Leserinnen und Leser, die mein anderes Buch „Parallele und verteilte Anwendungen in Java“ (ebenfalls im Hanser-Verlag erschienen) durchgearbeitet haben, das für dieses Buch nötige Vorwissen mitbringen sollten, das man aber selbstverständlich auch auf andere Weise erworben haben kann.

Damit sich Personen beiderlei Geschlechts angesprochen fühlen, verwende ich an einigen Stellen im Buch sowohl die männliche als auch die weibliche Form. An anderen Stellen verwende ich dagegen nur die männliche oder nur die weibliche Form. Ich habe nicht gezählt, wie oft welche Form vorkommt. Ich hoffe aber sehr, dass die Verwendung in etwa ausgewogen ist.

Auf der Web-Seite http://jk.hochschule-trier.de finden Sie weitere Informationen zu diesem Buch. Insbesondere können Sie von dieser Seite den Beispielcode des Buches in Form einer Zip-Datei herunterladen. Für nachträglich entdeckte Fehler, die leider nicht vermeidbar sind, gebe ich auf dieser Web-Seite sowohl die entdeckende Person als auch die Fehlerkorrektur an. Wenn auch Sie Fehler entdecken, so teilen Sie mir diese bitte in Form einer E-Mail mit, die Sie an oechsle@hochschule-trier.de adressieren. Ich bin an allen Arten von Fehlern interessiert. Dies schließt einfache Fehler wie ausgelassene oder vertauschte Buchstaben und Kommafehler genauso ein wie inhaltliche Fehler. Gerne können Sie auch Ihre positiven oder negativen Kommentare an die angegebene Adresse senden und mit mir dadurch ins Gespräch kommen.

Zum Abschluss dieses Vorworts bleibt mir die angenehme Pflicht, denjenigen Personen zu danken, die mich in irgendeiner Weise bei der Entstehung dieses Buches unterstützt haben: Ich danke den Studierenden des Master-Studiengangs Informatik an der Hochschule Trier, die meine Komponenten-Vorlesung gehört und kritisch begleitet haben, der Kollegin Gaby Elenz und den Kollegen Patrick Fries und Prof. Dr. Andreas Künkler vom Fachbereich Informatik der Hochschule Trier für ihre Unterstützung in vielfältiger Weise sowie vom Hanser-Verlag Herrn Dr. Martin Feuchte, Frau Mirja Werner und Frau Franziska Kaufmann für ihre Hilfe. Ganz besonders danke ich auch meiner Familie. Zur Fertigstellung des Buches musste nicht nur der Text geschrieben werden, sondern ich musste auch die Beispielprogramme entwickeln und ausprobieren, was ebenfalls beträchtliche Zeit in Anspruch genommen hat. Ich danke euch für eure Geduld.

Konz-Oberemmel im Januar 2013
Rainer Oechsle