Like The Title, This Book Is Pure Distilled Information

This reference and text treats the Smalltalk programming system and the web of object-oriented ideas within and around it. Thus it is more than a guide to the language; it also examines Smalltalk in its technical and historical setting, and along the way addresses the questions that every Smalltalk developer sooner or later naturally wonders about. Assuming nothing more than general programming experience, it begins with what objects and classes are, and eventually progresses to subtle matters such as the distinction between types and classes. Going beyond typical programming language books, it also covers the most important design patterns and how to write them in Smalltalk. The thrust then is not merely programming in Smalltalk with objects, but thinking and designing effectively with objects. This edition is a reprint of the original 1996 edition. Although the intervening years have brought the accustomed rapid changes in the computing industry, the principles presented here remain as relevant now as then. Three of my favorite topics are Smalltalk, objects, and design. Chamond Liu's book is the perfect blend of these topics. I heartily recommend this book to practitioners who don't want to read a dry treatment on design methodology or yet another programming book. You will be treated to elements of good design, a historical perspective, design patterns demystified and coded in Smalltalk, and just the right mix of Smalltalk programming and object concepts.—Rebecca Wirfs-Brock, author of Designing Object-Oriented Software

Well-written and well-thought-out. The chapter on design patterns especially is first-rate.—Doug Lea, author of Concurrent Programming in Java, and Object-Oriented System Development

One of those rare books that rewards both beginners and experts with insights appropriate to their levels. In addition, the writing style—combining incisiveness and grace—makes it a real pleasure to read.—Dave Collins, author of Designing Object-Oriented User Interfaces

Best book on Smalltalk I've seen!—Sherman Alpert, author of The Design Patterns Smalltalk Companion

... approachable, literate, fun, accurate and different ... the writing is of the highest calibre.—Ralph Johnson, author of Design Patterns: Elements of Reusable OO Software
am very impressed with it. . . . Unlike all of the other books, it covers all of
the important issues relevant to effective understanding of the language
under one cover.—Ed Klimas, author of Smalltalk with Style What I liked
most about this book was the confidence with which Liu presents a wide
range of topics: objects, architecture, gui, frameworks, design, patterns—
all the right stuff. If a bright developer from some other branch of our
discipline were to use his book as his only source of information, he would
come off as savvy and well connected, even at a Smalltalk conference.
And he would enjoy the experience too, since Liu never talks down to his
readers.—Ward Cunningham, CRC Card inventor A very solid and
entertaining book by an expert communicator . . . Beginners and old
hands alike will find useful ideas, entertaining writing and thought-
provoking allusions to broader technical, literary and philosophical
topics.—Martin Nally, chief architect of IBM Smalltalk and VisualAge . . .
the scope and level very interesting . . . a very useful collection of things
which are spread around and hard to find . . .—Tom Morgan, Manager of
Technology Development, Brooklyn Union Gas A very well written book; a
pleasure to read cover to cover. Good chapters on design patterns,
metaclasses and garbage collection, etc. You can tell by the writing style
that the author is also a teacher—conveys information gracefully and
effectively. Highly recommended.—Paul Jasek, Chubb & Son, Inc.

My Personal Review:
Smalltalk, Objects, and Design is such a simple and unexciting title, and
yet thats what this book is, in its entirety. This book will teach you about
Smalltalk, it will teach you about objects (using Smalltalk as the medium,
but you will learn much about Object Oriented programming that is
applicable in any language), and it will teach you about good design
principles.

It is written in clear, straightforward prose. In each chapter, the author
presents a new idea, discusses the reason for that idea and its
implementations (and often discusses several alternative solutions to a
given problem that have been adopted historically or in other languages,
as well), then gives some concrete examples, some exercises to reinforce
the point, and then summarizes. I learned a lot of things in this book that I
only vaguely understood before. How garbage collection works. How
Smalltalk really accomplishes everything is an object. What weak
references are and how to use them. How the Model-View-Controller
framework works. There are good discussions about inheritance and how
to avoid overusing it, implementation of polymorphism, reifying methods,
and so on. Good, meaty stuff especially for the beginning OO programmer.

This book talks about Smalltalk from a design perspective, so while there
is an introduction to Smalltalk and a lot of examples to help you learn, it
really isnt intended to teach the language. If you have no Smalltalk
experience at all, the book will teach you enough to follow along, but youll
want another book to go further in the language. It is fairly implementation-neutral; the author uses IBM’s VisualAge for his examples, but he provides tables and footnotes when it’s necessary to point out the differences with other Smalltalk implementations such as VisualWorks, and as a VisualWorks user myself, nothing in this book threw me off.

Objects are explained in great detail, starting with the hierarchy (in Smalltalk), what classes really are, how inheritance really works, buying vs. inheriting (i.e., has-a vs. is-a), abstract classes, Containers, polymorphism, and many other crucial OO concepts. Again, it’s focused on Smalltalk but if you do OO-programming in Java or C++ or some other language, there is a lot here that will be useful to learn (and the author even adds footnotes here and there commenting on how certain things are done in C++ or other languages).

The design chapters are good, though not as thorough, as software design is really a vast subject requiring many books in its own right. The author gives brief treatment of a lot of practices covered in much more detail in other books; for example, there is a chapter on design patterns, much of which is adapted from the famous Gang of Four book Design Patterns (Gamma, Helm, Johnson, and Vlissides). The author talks about CRC cards, UI design, software engineering practices, frameworks, and other topics that can’t really be done justice in a chapter each.

If I were teaching a class in Object-Oriented Programming, I would use Smalltalk as the language to teach basic OO concepts, and I would use this book as the text. Even if I had to use Java as the teaching language, I would find a good Java-based book on OO-programming, and supplement it with this one.

The final chapter, Why Developing Software is Still Hard, was the author’s opportunity to digress, and is worthwhile for all new programmers to read. Basically, there is a lot of Object-Oriented Kool-Aid out there, and if you take this chapter to heart, you’ll avoid drinking some of it.

For More 5 Star Customer Reviews and Lowest Price:
Smalltalk, Objects, and Design by Chmond Liu - 5 Star Customer Reviews and Lowest Price!