Jakarta Commons Cookbook

Keith Alcock, TCS Member
tcs@keithalcock.com

Jakarta Commons is a collection of small, reusable, popular, and free (both as in free beer and in freedom) Java software components developed under the auspices of the Apache Software Foundation (ASF). Approximately 30 projects, including Jakarta, are organized under the foundation. Jakarta's emphasis is on server side software and it encompasses nearly 20 subprojects itself, including Tomcat, Cactus, and Commons. Jakarta Commons consists of over 30 sub-subprojects, of which only about half are discussed in this book (according to the author, those which can be covered without a detailed 30-page introduction apiece). The ASF project hierarchy is under constant reorganization so that some projects described in the book like Slide and Velocity are presently siblings of Commons, and Lucene is now on par with Jakarta. There is one stranger included, Freemaker, which seems unrelated to ASF. All in all, approximately 1/1200th of ASF software is described in the 400 pages of this book, which attests to the great amount of open source Java code available and the number of recipes needed to describe it. Object-oriented programming and open source projects seem to be a successful combination.

The book is divided into twelve chapters in which closely related projects are described in O'Reilly's cookbook format. For example, XML tools including Commons Digester (utilities for XML-to-Java-object mapping), Commons Betwixt (services for mapping JavaBeans to XML documents), and Commons JXPath (tools for manipulating Java classes using XPath syntax) are collected in chapter six. Approximately fifteen recipes are included per chapter and each recipe has problem, solution, discussion, and sometimes "see also" sections. The table of contents available at the book's web site lists the topics that are addressed. By far the weakest sections contain the problem statement which is all too often dictated by the solution: "You need to model a conditional statement with functors" (problem), so (solution) download our handy IfClosure (model), Predicate (condition), and Closure (functor) classes. This redundancy makes one wonder whether the cookbook format is appropriate. The discussion sections are the most interesting and contain some very original examples: comparing vehicle efficiency, evaluating space shuttle launch conditions, modeling logical circuits, and many more. Why so much space is wasted there explaining the trivial Maven project settings when Maven is discussed nowhere else in the book is a riddle, however. The "see also" sections contain helpful pointers to other recipes, books, projects (e.g., Clover, JCoverage, JUnit, Colt), web sites, and documents on specific web pages.

The author includes compelling justification for adding high quality, open source Jakarta Commons code to software projects and discourages stubborn adherence to the not-invented-here mindset or unknowing duplication of freely available code. Other advice is substantially more disputable, however. One is advised to write finicky methods that will potentially duplicate checks for null values, invalid arguments, or out of bounds indexes rather than to define by contract what code is responsible for checking or to rely on the built-in, optimized, and maintenance-free bounds checking. The author advocates 100% test coverage and that is especially important for languages and programs that perform type checking or casting at runtime, but no distinction is made between statement coverage and condition or path coverage, for which completeness is not feasible. The author compares writing good code to writing a well-written essay, and I concur, but wish that more of the included code lived up to the comparison.

That criticism may seem unduly harsh, but it comes on the heels of the book's major disappointment: the solution sections in particular contain numerous technical and typographical errors, at least several dozen that I could find. After the first twenty pages I began to question every piece of code and tidbit of advice. Many examples won't compile (due to syntax errors, misnamed variables, nonexistent library methods), won't run to completion (infinite loops), or won't produce the output described. Writing a sentence about 100% test coverage does not make sense when portions of the code never saw a compiler or were never executed. No matter how well an essay is written, it still requires proofreading. So although I recommend that Java programmers learn all about Jakarta Commons, this particular first edition would not be my recommended source of examples. Insist on an updated, corrected version and in the meantime tour the web site (http://jakarta.apache.org/commons/), download the jar files, and skim the API documentation. You will probably be an expert before you know it.

Title: Jakarta Commons Cookbook
Author: Timothy M. O'Brien
Publisher: O'Reilly
Pages: 400
Price: US $44.95
ISBN: 0-596-00706-X
Website: http://www.oreilly.com/catalog/jakartackbk/