Indexing: Goals, strategies and tactics

Ronald J. Fehd

Abstract

In this article I review my index production notes from a previous project and make a checklist of goals, strategies and tactics for my next book. I compare the writing roles of author, editor and proofreader with the indexing team of author and indexer.

The purpose of this article is twofold: the first is to provide a workflow for documents with indexes that makes debugging, proofreading and polishing of index entries across multiple files easier. The second is to provide a template document that supports that workflow. The packages `chappg`, `fancyverb`, and `indextools` are used in the template.

The expected audience of this article is authors and technical writers who want to become proficient in the art and science of indexing.

1 Introduction

What is an index? A simple description is *an index assists the reader in finding information in a document.* The term *index* has two components, *what* and *where*. The *what* is the *entry*; the *where* is the *locator*, usually a page number. An index is an asset to the reader. Its presence is a selling point for the document. Consider the reader before sale, as a customer, and after sale, as a user.

A knowledgeable reader picks up a book and wants to access the quality of the index. A coarse metric of quality is quantity, the simple easy-to-calculate ratio of the number of pages of the index divided by the number of pages of the book. The range of this fraction, as a percentage, is 2–15%, where 2% indicates a trivial index and 10+% shows a concerted effort, as for a reference book or technical manual, that will be both useful and often used.

Mulvany [15, table 3.2, pg. 72], shows the correlation between this percentage and the professional indexer's estimated work effort; this metric is known as *entries per page* (E/pg) discussed below in the section 'Budget and quality'.

Team work In writing there are three people on the team: author, editor and proofreader. The editor's role has two tasks: micro, concerned with producing accurate and consistent syntax; and macro, concerned with focus, winnowing and elimination of redundancy. The proofreader's role is to detect inconsistency and do fact checking.

In indexing, there are apparently two people on the team: the author and the indexer. In project management we have the three choices of *fast, accurate, or cheap.* Professional indexers are paid to be the pair of choices of goodness: fast and accurate. Their job description encompasses both of the writers' team roles of editor and proofreader for the index.

The task of this article is to examine the work of editing and proofreading an index and present a workflow for writers which makes those tasks obvious, defined and easy to accomplish. Additional intermediate documents are needed for the index review; an example template which supports the review process is shown on page 35.

A professional definition Mulvany [15, pg. 8], provides this definition.

> An index is a structured sequence — resulting from a thorough and complete analysis of text — of synthesized access points to all the information contained in the text. The structured arrangement of the index enables users to locate information efficiently.

Concepts In mathematics we have the concept of an *array* which contains a set of values; an index is an integer, a pointer to an element in the array. The concepts of array and index are separate from the values in the array; the concepts are a method of accessing the values in an iterative loop.

In natural languages we can apply the concept of an *associative array.* An abbreviation or acronym is used as shorthand to refer to a longer word or concept; e.g., two-letter abbreviations for country names used in urls and two-letter state abbreviations used by the United States Postal Service.

In *database* theory, we have two types of tables, fact tables and dimension tables. A fact table is a record of a transaction, and contains two types of columns: foreign keys which are pointers to rows in dimension tables, and facts which are the measurements or quantities of the transaction. A row may be uniquely identified by a composite key, a set of the foreign keys. A dimension table contains two types of columns: a primary key (row identifier) which matches the values of the fact tables' foreign keys and columns of information about the key.

In the associative array examples, given an abbreviation, the primary key (country code, state abbreviation, etc.), we can retrieve the information about the entity — capital city, location, etc.

Index, my definition I propose that a document is a *knowledge base*, that the index is a database and that the words of an index entry are an item of an associative array, a composite key, of and for that document, that points to the information that the reader is seeking. The *locator* is the unique *row*

identifier, the page number, that the reader uses to access the information in the document. Just as the table of contents is a sequential exposition of the ideas in the document, the index is an alphabetical listing of the concepts, ideas, and tasks in the document. The index, in that sense, is a further extension and expansion of the table of contents.

Implications of this idea are discussed below.

1.1 How professionals work

For professional indexers the project of producing an index for a document consists of the tasks shown in Table 1.

Table 1: The professional indexer's process

0. read the complete document
1. identify, record ranges
2. for each page:
identify terms or tasks
recording: insert new or update old
3. polishing the database:
add redirects
contract single `item!sub-item`
to `item, sub-item`
expand items with too many references
by adding sub-items
contract series into ranges
4. transform database to output

Professional software offers support for the polishing processes during the item recording process, by saving the entries in a database so that each new entry is compared to its predecessors and a decision must be made to use an existing entry and add a locator, or create a new entry.

Compare this to the markup process shown in Table 2 where entries are written to an `.idx` file so we have a sequential list of entries, which are then sorted, duplicates removed, and formatted in one step.

1.2 Review of indexing markup commands

There are three types of indexing commands in LaTeX: *words(s)*, *ranges* and *redirects*.

▪ **word(s)** Words may be marked in three styles, *plain*, *special font*, or *special locator*.

plain: The indexer marks up plain entries in a document with this form:

```
\index{word(s)}
```

The `word(s)` can be any string. In support of later polishing, `word(s)` can be subdivided into as many as three levels with `!` characters:

```
\index{item!sub-item}
\index{item!sub-item!sub-sub-item}
```

special font: For the case where a word is to be set in another font the syntax is:

```
\index{item@\texttt{item}}
```

special locator: Special pages may be marked so that the *locator* appears in a different font by adding a vertical bar (`|`) plus the font shape (italic) or series (bold).

```
\index{item|textbf}
```

It is customary to mark a page containing a definition in bold. Pages in a glossary may be marked in italic. Any such special locator emphasis ought to be explained in a prologue to the index. For an example of an index prologue see *Guide to LaTeX* [12]. The `indextools` package [17] contains the command `\indexprologue` for this purpose.

▪ **ranges** To mark up a page range use a pair of commands, the first with `|(` and the second with `|)`:

```
\index{autoexec|(}%range begin
...
\index{autoexec|)}%range end
```

No markup is necessary for the consolidation of sequential locators. The `makeindex` processor consolidates sequential page numbers, e.g., `2, 3, 4` is changed to `2--4`.

▪ **redirects** A locator can be in one of three forms, either a single page number, or a page range such as 8–13, or a *redirect*, a reference to another index entry, without a page number. The syntax is

```
\index{canines|see{dogs}}
```

This produces an index entry without a page number, as in:

canines, *see* dogs

1.3 The `makeindex` process

Producing an index of a document using LaTeX and `makeindex` consists of the steps shown in Table 2. I highlight two intermediate steps in this process, sorting and removing duplicates, both of which are done in memory by `makeindex`. They can be split out into the creation of the two temporary files, *sorted.idx*, and *nodups.idx*. See Listing 2 for the system commands to produce the *sorted.idx* file.

Listing 1 shows a demonstration file. Processing the document produces output file `\jobname.idx` with the *locator* information (page numbers) added:

```
\indexentry{fox}{1}
\indexentry{dog}{1}
```

Processing with `makeindex` produces an output file named `\jobname.ind`, the formatted output:

Table 2: The `makeindex` process

Processor	Input, Action	Output
(pdf)latex	jobname.tex	jobname.idx
makeindex	jobname.idx	
	sort:	*sorted.idx**
	remove dups:	*nodups.idx**
	nodups.idx	jobname.ind
(pdf)latex	jobname.tex	jobname.pdf

* temporary file

Listing 1: Document with simple indexing

```
1  %this is the file demo-1-makeindex.tex
2  \documentclass{article}
3  \title{Example of Simple Indexing}
4  \author{R.J. Fehd}
5  \usepackage{makeidx}\makeindex
6  \begin{document}\maketitle
7  The quick brown fox\index{fox} %*
8  jumped over the lazy dog\index{dog}.
9  \printindex
10 \end{document}
```

* The index markup is placed after the word as recommended by the package documentation.

```
\begin{theindex}
  \item dog, 1
  \item fox, 1
\end{theindex}
```

The `\printindex` command (line 9) reads that `.ind` file if it exists:

```
\IfFileExists{\jobname.ind}
  {\input{\jobname.ind}}
  {}
```

2 Discussion of my experience

I will use a project of my own as an example. I worked on writing a book over a period of about a year. There were six chapters of 15–30 pages each. I spent about a month writing a chapter. Since I knew I would produce the index myself, I added the markup as I wrote.

In early chapters I used a form for noun in category `{noun category}` that I later changed to `{category!noun}`, e.g., from `{scan function}` to `{functions!scan}`.

In the middle chapters, I was echoing these entries, marking both

```
{noun category}
{category!noun}
```

In the last chapters I decided that I wanted `{category!noun}`, so I marked `{noun category}` as a redirect:

```
{noun category|see{category, noun}}
```

in those chapters. Since each redirect has a separate *locator* the duplicate removal process leaves multiple entries, only one of which is to be retained. At the end I changed all the `{noun category}` entries to *redirects* and placed them in a separate file.

2.1 Lessons learned from my mistakes

In this section I provide a summary of the categories of mistakes that I made.

These are the categories: *macros*, *multitasking*, *ranges*, *redirects*, *repair*, *testing*, and *vocabulary*.

macros I used macros for a while but gave up on them for two reasons. The first was that macros introduced extra spaces, which I was not a sophisticated enough TEXnician to fix; and second, complexity. Because I was marking up item, sub-item and sub-sub-item, with font changes, I realized that managing the set of macros might consume as much time as plain long-form markup. Swapping out the macros with long-form markup contributed to the rework budget.

multitasking Writing, editing and polishing are very different intellectual activities from indexing. Adding markup while I was polishing produced different entries e.g., *directories* and *folders*. Identifying synonyms, deciding which one to use, and then finding and fixing the other occurrences contributed to the rework budget.

ranges One of my most common problems in getting the index working for a chapter was incomplete ranges; these errors are noted in the `\jobname.ilg` file. Ranges may cross one or more sections and the pair of markup commands may be many pages apart.

redirects The markup syntax for *redirects* is the same as for *word(s)*. What is different is that the *locator* in the printed result is not a page number, but a reference to another word. My mistake was in thinking that the `makeindex` process removed duplicate *redirects*. Moving these entries to a central file contributed to the rework budget.

repair where? All the items listed here required that I return to one of the many chapter files in order to fix the problem entries. The index entry repair problem is discovering in which file the entry is located. The problem is that the *locator* is the page number of the complete document. Finding the name of the chapter file is a two-step process, first, note the page number, then return to the table of contents to find the page range of the chapter.

testing During the retrospective for this paper I realized that I spent a lot of time reviewing the

complete index. My solution for future work will be to do unit tests for each chapter and one final complete integration review.

vocabulary As noted above in the multitasking paragraph, choice of vocabulary is a key task to accomplish before beginning indexing. Indexing science addresses this idea with the concept of *controlled vocabulary*. Much to my surprise, words in the entries did not occur as phrases in the text.

Summary In this section I have described a number of mistakes. Time spent on this work can be reduced by discovering ways to implement these tasks. In short:

- change page number to include chapter number
- control vocabulary used in entries
- centralize *redirects*
- manage ranges early
- process each chapter independently, for both writing and indexing
- review temporary files: `\jobname.idx`, `\jobname.ilg`, `sorted.idx`
- polish the complete index only once

3 Budget and quality

Budget and daily job diary Table 3 is from Frederick Brooks' classic book, *The Mythical Man-Month* [3]. This table highlights the under-estimated items in the budget: unit and integration testing. I show this table in my presentations and point out the difference in the time spent coding, where much time is spent debugging — getting the code to work — and the two aspects of testing: testing the program against its expectations — unit testing — and how a program works as a caller or callee: integration testing.

Table 3: Time spent in program development (from [3])

Phase	Time	Action	Time
design	1/2	understand problem: education, research	1/3
		development coding	1/6
testing	1/2	unit test	1/4
		integration test	1/4

For the task of indexing I want to carry forward the ideas of planning and markup, typing the entries, unit test of entries in a chapter, and the integration test of all the chapters of a document. Editing and proofreading of index entries in a chapter are separate tasks from proofreading the index of the whole document.

Table 4: Estimates of percentage of index pages and entries per page (E/pg), by book type*

	index size as	
Book type	% of book	E/pg
light text, few details	2–5	3–5
reference	7–8	6–8
documentation, light	10	8–10
manuals, heavy	15+	10+

* adapted from [15, table 3.2]

The daily job diary (djd) is a necessary part of budgeting. If we keep track of time spent on a task then we can monitor and estimate whether we will be under or over budget.

Metrics for quality As explained in the introduction, an interested purchaser can calculate the percentage of pages of the index in a book. This is a rough estimate of the depth of the index: how much information is in the index and how easy is it to use the index to find information as compared to reading the table of contents.

There are two metrics of index quality: entries per page (E/pg) and locators per entry (L/E).

Entries per page (E/pg) This quantity is used as a forecasting aid; it is an indexing science term mentioned in [15] and is derived from the type of document. It is used to estimate the size of the index, the number of pages set aside in the document, as a percentage of the number of pages of text. In the section in which this table appears, studies are quoted as discovering that the average size of an index is three percent. Table 4 is my adaptation of the information.

Locators per Entry (L/E) This quantity is an integer, typically in the range 3–13, decided on before polishing the final draft of the index to subdivide entries and therefore increase the size of the index.

4 About goals, strategies and tactics

We have two goals. The first is to prepare an index for a reader with sufficient detail and depth for the type of document. The second is to be aware of the budget and enhance the production process by reducing time spent on indexing tasks.

We'll discuss some packages, other tools, and tactics to meet these goals.

4.1 Strategies: packages

Several of the problems noted earlier have solutions that are outside the set of indexing tasks. Three

packages can help reduce the time spent in administration of the document (all have many features besides what's mentioned here):

`chappg` changes *locator* output formatting from *page* to *chapter*.*page*.

`fancyvrb` provides `\VerbatimInput{file.ext}` to typeset external files verbatim.

`indextools` provides `\makeindex[intoc]` to put an index in the table of contents (instead of using `\addtocontents`), and `\indexprologue`, to typeset introductory text before the index, e.g., to describe the formatting conventions used.

More information about these packages is available through CTAN, at `http://ctan.org/pkg/`*pkgname*. They are all included in the usual TEX distributions.

4.2 Strategies: tools

This section discusses methods for using (LA)TEX commands that can help in index (and document) development.

■ **flags** A flag is a variable which takes only boolean values, true or false. These are the TEX commands to initialize a new boolean switch named `ReviewIndex`:

```
\newif\ifReviewIndex
  \ReviewIndexfalse % set to false
  \ReviewIndextrue  % set to true
```

Then, this is the template for conditionally executing statements with this flag:

```
\ifReviewIndex
  % statements if true
\else
  % statements if false
\fi %end:ReviewIndex
```

The `\else` part is optional.

As seen in the template document of Listing 3 (in the appendix), I use this flag to control formatting and display of various intermediate indexing files for review.

This set of statements may be implemented in LATEX with the `ifthen` package [12, pg. 440]. I use the plain TEX statements because I use it to choose the `\documentclass` options.

■ `\IfFileExists` This is a LATEX programming command [12, pg. 436]. This command can be used to ensure that the document compiles before temporary files are created.

```
\IfFileExists{filename.ext}
  {statements if true}
  {statements if false}
```

■ `\includeonly` This is a LATEX command used to process only one subpart of a document, such as a chapter. Its use requires two statements, the first in the preamble:

```
\includeonly{ch-A}%disable after unit test
```

and then in the body [12, pg. 206]:

```
% in text part, list all chapters
\include{ch-A}
...
\include{ch-Z}
```

4.3 Tactics during markup

Here are some minor habits I developed to help produce a cleaner document.

- Mark up ranges after outlining in a file, e.g., `ch-0-outline-index`
- One index command per line better supports later searching.
- End entries with percent sign (`%`) to avoid multiple spaces between words.
- Place all redirects (*see* and *seealso*) in one file to be input after `\begin{document}`, as in: `\input{ch-0-index-redirects}`

5 Suggestions for a workflow

Table 5 lists the major steps in index creation.

Table 5: Overview of workflow for indexing

Step	Idea	Task
1	setup	make files
2	for chapters	writing
3	for chapters	indexing, unit test
4	review index	integration test
5	finish	

We'll now discuss each of these in turn.

■ **setup** The first task is setting up files for the project and getting the document working with the various chapter and other files that support review. (The listings are in the appendix to this article.)

File	Task	Listing
`a-book.tex`	copy	3
`ch-0-outline-index.tex`	outline *draft index*, *mark ranges*	5
`ch-0-index-redirects.tex`	redirects	4
`a-book.bat`	doc working	2

The most important task in the setup, right after outlining the text, is to begin to draft the index. Make an entry for every task mentioned in the section commands. This is the first draft of the database — `sorted.idx` — that we will use as a reference when marking up each chapter.

■ **writing a chapter** Create a new chapter file and copy the draft outline and index entries from file

`ch-0-outline-index.tex` into it. Record amounts of time worked in a daily job diary.

File	Task
`a-book.tex`	`\includeonly{ch-X}`
	`\include{ch-X}`
`ch-X.tex`	write, type, edit, etc.

■ **indexing a chapter** The task of indexing requires a change of intellectual attitude. Remove your author's hat and put on your proofreader's hat; change mode from exposition to reader and reviewer; become objective. Part of indexing is comparing the draft index of the complete document to what has been written in the chapter under review. Each entry ought to be either *new* or *already existing* in the draft database, `sorted.idx`. This comparison is a step in the quality assurance of the index. Remember to record time-start and -end in the daily job diary.

File	Task
`a-book.tex`	disable this chapter:
	`%\includeonly{ch-X}`
`a-book.bat`	process whole document
`a-book.pdf`	print only `sorted.idx`
`a-book.tex`	`\includeonly{ch-X}`
`a-book.pdf`	print galley of ch-X
`ch-X.tex`	for each page:
	move draft entries to correct line
	compare markup to `sorted.idx`
	type index entry
`a-book.bat`	process chapter
`a-book.ilg`	confirm unit test correct
	add new entries to draft file
`a-book.tex`	`%\includeonly{ch-X}`

Refer to the `showidx` listing at the top of each page; count and enter the number of index entries per page in the Entries-per-Page (E/pg) log. Calculate the mean and standard deviation of E/pg. Examine pages with E/pg more than 2.5 standard deviations from mean. Review the daily job diary; compare time spent writing with time spent indexing.

■ **reviewing index** Choose the maximum locators per entry (L/E), e.g., from the Fibonacci series: (3, 5, 8, 13). This choice indicates the depth of the index. A large L/E indicates a shallow or small index. A reader will have to look up that many entries in the text to find what they want. Entries with more than that are to be subdivided.

File	Task
`a-book.tex`	disable all:
	`%\includeonly{ch-X}`
`a-book.bat`	process whole document
`a-book.pdf`	print `sorted.idx`
	find entries with large L/E:
`ch-?.tex`	expand entries
	find entries with single sub-items:
`ch-?.tex`	contract entries
`a-book.bat`	process whole document
`a-book.ilg`	confirm integration test complete
`a-book.pdf`	print index

■ **finish** With all unit tests of chapters completed and the integration test of the document complete, print the final document and review the budget.

File	Task
`a-book.tex`	disable flag:
	`%\ReviewIndextrue`
`a-book.bat`	process whole document
`a-book.pdf`	print

Review the daily job diary; compare time spent reviewing the complete index (integration test) with sum of chapter times (unit tests). Compare time spent on indexing with time spent writing.

6 Suggested reading

This section contains *inspiration*, *manuals*, *budgeting*, *macros*, *standards*, and *types of indexing*.

inspiration In 2001, Robert Horn received a Lifetime Achievement Award from the Association of Computing Machinery. His acceptance speech is titled *What Kinds of Writing Have a Future?* His discipline of presenting information is called *structured writing*. This article and [5] are examples of the author's work using structured writing.

manuals After a dictionary and thesaurus, an author perhaps most needs a style guide. For American English writers, the essential guide is *The Chicago Manual of Style* [7]. The University of Chicago Press also publishes the bible of professional indexers, *Indexing Books* [15]. Chapter 9, *Editing the Index*, is the basis for my recommendation of polishing the complete index — the integration test — as a separate activity.

As a technical writer I found Ament's *Indexing Guide* [1] to be just the reference book that I needed. While he does not use the term *task* his explanation of deconstructing index entries into verbs and nouns made sense to me. His basic discipline can be expressed as follows:

A task consists of a verb and a noun. Always mark verbs in gerund form and nouns as plurals,

i.e.: `\index{<verb>ing <noun>s}`
e.g.: `\index{copying files}`

He states that the task of indexing can be used as feedback to improve the quality of the writing, because text that is poorly written is difficult to easily and accurately index.

This book is small (97 pages) and concise, written in plain language (`centerforplainlanguage.org`). I recommend it.

The Indexing Companion [4] provides an overview of the ideas and sources of *controlled vocabulary*. That discussion led me to the concept of an index as a database of the document, the knowledge base.

The database concepts I used in the explanation of my definition of an index are from [8].

The American Society of Indexers publishes *Best Practices for Indexing* [13]; it has appendices which provide guidance for authors in various genres.

budgeting *How to Communicate Technical Information* [16] describes the issues of budgeting in large projects of software and hardware documentation.

macros Gregorio, in [6], provides guidance for finding extra spaces when writing macros.

standards *The Global English Style Guide* [11] lays out a discipline of writing standardized English in technical manuals, which supports machine translation.

types of indexing This article is a discussion of the issues of *closed-system indexing*, i.e., finding information in one document. *The Organization of Information* [18] is an exposition of the theory and practice of information science which is concerned with *open-system indexing*, the issues of retrieval from multiple documents.

closing quotation *The LaTeX Companion* [14] ends the discussion of index markup with this quote:

> While all of these tools help to get the correct page numbers in the index, the real difficulty persists: choosing useful index entries for your readers. This problem you still have to solve. . . . to produce a comprehensive index that helps you, the reader, find not only the names of things . . . but also the tasks, concepts, and ideas described in the book.

7 Future work

I see several ideas that can reduce or eliminate time spent on indexing tasks. These ideas are *external input*, *metrics* and *redirects*.

external input For the case when both the author and a professional indexer are employed to mark up the entries we need a description of a file format, such as `.csv`, for use by packages which write `*.idx` files. Vendors of professional indexing software must be able to supply this file format to the TEXnician.

metrics I have identified the metrics entries per page (E/pg) and locators per entry (L/E) as summaries that may be reviewed for quality assurance. The data for these sums are in file `\jobname.idx`. **E/pg:** This data can be seen when using the package `showidx`, but the count must be manually tallied and recorded. **L/E:** This data can be seen in the file `sorted.idx`, but the count must be manually tallied and recorded.

redirects Add an index command for redirects which uses the same syntax as `\index` but which writes its output with the *locator* set to one. This allows multiple redirects in many files, which eliminates the task of moving them to a central location.

8 Conclusion

Marking up index entries is simple. The hard part of index preparation is polishing, which is perhaps 80% of the effort in preparing the index. In this paper I have presented several ideas to make the polishing effort less complicated. The first is sorting and saving the entries in the `.idx` file. The second is adding a flag `ReviewIndex` to turn on additional features. Another is to change the definition of `\thepage` so that it contains more information — the chapter number — in the `sorted.idx` file so that problem entries can be more easily located in their respective files. The last is to use the `ReviewIndex` flag to display various index files during review. I believe this set of packages, tools and workflow can help significantly reduce the time spent polishing the index in your next project.

9 Appendix

We've seen Listing 1 (`demo-1-makeindex.tex`, a minimal document with indexing) earlier (pg. 30). This appendix contains the remaining program listings.

It is left as an exercise for the reader to create the file `ch-A.tex` in order to get the template `a-book.tex` (Listing 3) working.

Listing 2: `a-book.bat`, Windows batch file for processing a document

```
1 rem this is a Windows DOS batch file
2 rem this is file  a-book.bat
3 set       jobname=a-book
4 pdflatex %jobname%
5 sort     %jobname%.idx /o sorted.idx
6 pdflatex %jobname%
```

Listing 3: a-book.tex, template document with several index review features

```
% this is template document a-book.tex
\newif\ifReviewIndex\ReviewIndexfalse
  \ReviewIndextrue %disable for final
\ifReviewIndex
     \documentclass[12pt,oneside]{book}
\else\documentclass[10pt,twoside]{book}
\fi %end:ReviewIndex
%
%\includeonly{ch-A}%unit test
%
\title{Example of Complex Indexing with
  ChapterPage, IndexTools and ShowIdx}
\author{R.J. Fehd}
\date{2016-March}
\ifReviewIndex
  \usepackage{showidx}%in right margin
  \setlength\oddsidemargin{0pt}%oneside
  \usepackage{chappg}
\fi
\usepackage{fancyvrb}  %VerbatimInput
\usepackage{indextools}%after showidx
\makeindex[intoc]      %write \jobname.idx
%\usepackage{hyperref}
%\usepackage{glossaries}%after hyperref
%\input{glossary-entries}
%
\begin{document}
%\frontmatter         %pg: i--x
%\input{c-frontmatter}%title, toc, etc.
%\mainmatter          %Chapter 1--N, pg 1--N
\ifReviewIndex \input{ch-0-outline-index}\fi
\input{ch-0-index-redirects}
\include{ch-A}%{ch-B}...{ch-Z}
%\input{c-backmatter}%bib, gloss, etc.
\ifReviewIndex
  \newcommand\Echo[1]{%arg=filename.ext
    \chapter         {#1}
    \IfFileExists    {#1}
      {\VerbatimInput{#1}}  %fancyvrb
      {file missing:  #1 } }%end:Echo
  \Echo{\jobname.ilg} %index log
  \Echo{ch-0-index-redirects.tex}
  \Echo{sorted.idx}
\fi %end:ReviewIndex
\indexprologue %an indextools command
  {This text explains font conventions.}
\printindex
\end{document}
```

Listing 4: ch-0-index-redirects.tex, template for index redirects

```
% this is file ch-0-index-redirects.tex
\index{canines|see{dogs}}
\endinput
```

Listing 5: ch-0-outline-index.tex, template chapter file

```
% this is file ch-0-outline-index.tex
\chapter{Draft: Outline and Index}

\section{chapter: Canines}canine
\index{dogs|textbf}%definition in bold
\index{foxes|textbf}%definition in bold

\section{chapter: Dogs}dog
\index{dogs|(}%range begin
\index{dogs!breeding}%
\index{breeding dogs}%
\index{dogs|)}%range end

\section{chapter: Foxes}fox
\index{foxes!habitat}%
\index{foxes!hunting}%
\endinput
```

About the author R. J. Fehd is a member of the TEX Users Group, and has been writing programs in SAS®[1], a language for statistical analysis, since 1986. He first published in 1997 and began using LATEX in 2004; in 2006 he hacked other conference classes to produce sugconf. After retiring in 2012, he began work on a compilation of his papers; he continues to present papers and seminars at SAS user group conferences and is Senior Statistical Programmer at Stakana Analytics. He has been recognized as a peer for his contributions to the *SAS-L listserv*, and is a contributor to the *sasCommunity.org wikipedia*.

References

[1] Kurt Ament. *Indexing, A Nuts-and-Bolts Guide for Technical Writers*. William Andrew Publishing, Norwich, NY, June 2001. 97 pp., 3 chap., index: 11 pp. (11%). https://www.elsevier.com/books/indexing/ament/978-0-8155-1481-7.

[2] David Bausum. *TEX Reference Manual*. Springer, 2002. 388 pp., 2 chap., 3 app., index: 6 pp. (1%).

[3] Frederick P. Brooks Jr. *The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition*. Addison-Wesley, 2nd edition, 1995. 322 pp., 19 chap., index: 14 pp. (4%); theory and story based on development of the IBM System/360 operating system. https://en.wikipedia.org/wiki/The_Mythical_Man-Month.

[1] SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. In the USA and other countries ® indicates USA registration.

[4] Glenda Browne and Jon Jeremey. *The Indexing Companion*. Cambridge University Press, New York, NY, April 2007. 249 pp., 9 chap., index: 23 pp. (9%). http://www.cambridge.org/9780521689885.

[5] Ronald J. Fehd. An autoexec companion, allocating location names during startup. In *MidWest SAS Users Group Annual Conference Proceedings*, 2015. http://www.lexjansen.com/mwsug/2015/BB/MWSUG-2015-BB-10.pdf.

[6] Enrico Gregorio. Recollections of a spurious space catcher. *TUGboat*, 36(2):149–161, 2015. http://tug.org/TUGboat/tb36-2/tb113gregorio.pdf.

[7] John Grossman, editor. *The Chicago Manual of Style*. University of Chicago Press, 14th edition, 1993 (15th ed., 2006). 921 pp., 19 chap., gloss., bib., index: 25 pp. (6%); locators are chap.section, not page numbers. http://press.uchicago.edu/ucp/books/book/chicago/I/bo3625262.html.

[8] Ralph Kimball and Margy Ross. *The Data Warehouse Toolkit, The Complete Guide to Dimensional Modeling, Second Edition*. John Wiley & Sons, Inc., New York, 2002. 387 pp., 17 chap., gloss.: 29 pp., index: 18 pp. (4%). http://www.kimballgroup.com/html/booksDWT2.html.

[9] Donald E. Knuth. *The TEXbook*. Addison-Wesley, 1984. 483 pp., 17 chap., 8 app., index: 25 pp. (5%).

[10] Donald E. Knuth. *Literate Programming*. CSLI, Stanford, CA, 1992. 368 pp., 12 chap., index: 10 pp. (2%).

[11] John R. Kohl. *The Global English Style Guide: Writing Clear, Translatable Documentation for a Global Market*. SAS Press, 2008. 310 pp., 9 chap., 4 app., index: 14 pp. (4%). http://www.globalenglishstyle.com.

[12] Helmut Kopka and Patrick W. Daly. *Guide to LATEX*. Pearson Education, Inc., Boston, MA, 4th edition, February 2004. 597 pp., 18 chap., bib., index: 39 pp. (6%).

[13] Fred Leise and Devon Thomas. *Best Practices for Indexing*. American Society for Indexing, Tempe, AZ, 2015. 76 pp., 11 chap., index: 6 pp. (7%); 7 appendices for categories of books.

[14] Frank Mittelbach, Michel Goossens with Johannes Braams, David Carlisle, and Chris Rowley. *The LATEX Companion*. Addison-Wesley, 2nd edition, 2004. 961 pp., 14 chap., bib., index: 98 pp. (10%).

[15] Nancy C. Mulvany. *Indexing Books*. University of Chicago Press, 2nd edition, 2005. 315 pp., 10 chap., gloss., index: 25 pp. (8%). http://press.uchicago.edu/ucp/books/book/chicago/I/bo3625262.html.

[16] Jonathan Price and Henry Korman. *How to Communicate Technical Information: A handbook of software and hardware documentation*. Addison-Wesley, 1993. 402 pp., 21 chap., index: 22 pp. (5%).

[17] Maïeul Rouquette. indextools — producing multiple indices, 2015. http://ctan.org/pkg/indextools.

[18] Arlene G. Taylor. *The Organization of Information*. Libraries Unlimited, 1999. 280 pp., 10 chap., 3 app., gloss., 22 pp., index: 25 pp. (8%). http://www.abc-clio.com/product.aspx?isbn=9781591585862.

◇ Ronald J. Fehd
ronald dot j dot fehd (at) gmail dot com
https://www.linkedin.com/in/ronald-fehd-5125991

> The compiling of an index is interesting work, though some authors are apt to find it tedious and delegate the work to others. The proofreader who undertakes it will find that it is splendid mental exercise and brings out his latent editorial capability.
>
> Albert H. Highton,
> *Practical Proofreading*, (1926)
> quoted by Knuth, *The TEXbook*, pg. 481