The bag of tricks—Victor Eijkhout (p.260)
Here, two macros \storecat and \restorecat are defined, to be used as in file1. (In the article, \storecat is mistakenly also called \savecat.)

The purpose is to allow @ to be a letter within file1, but then to restore it to what it was before, when file1 is finished.

But now suppose we input file2 below, using the definitions supplied by Eijkhout.

\storecat @
\% file2.tex
\storecat @
\% macros
\% @ is a letter
\restorecat @
\input file1
\% macros
\restorecat @

When file2 is finished, @ will have \catcode 11, regardless of what it was before file2 was input. This is not what is wanted.

Babel, a multilingual etc.—Johannes Brahms (p.291-301) This article is inconsistent in its use of % to eliminate the space tokens that \TeX would otherwise place after { and } characters at the end of a line.

The use of \expandafter in Fig. 1, and its second use in Fig. 4, add nothing and are redundant. These defects are relatively slight, for they do not prevent the macros from working as advertised.

Letter from Victor Eijkhout (p.303) Here, he quotes a user as writing
\def\caption#1{\hbox{\hbox{ ... }\hbox{\top#1}}}

This line has 4 left braces but only 3 right braces.

Conclusion Errors lie in the dark side of programming. Usually, nobody likes making them, finding them, reporting them, or being told about them. We all wish that they did not exist. I hope that this letter has not made me any enemies, and has encouraged authors to submit more articles with fewer errors.

Jonathan Fine
203 Coldhams Lane
Cambridge
CB1 3HY
England
J.Fine@pmms.cam.ac.uk

One error less
Victor Eijkhout

Jonathan Fine correctly points out that the macros \storecat and \restorecat cannot be used nested. This problem cannot be remedied by using grouping, because these macros may enclose arbitrary pieces of code, and you may not want a group around them.

Here is a solution: the call \storecat\% defines the control sequence \restorecat\% to restore the category code and restore the previous definition of itself.

\def\storecat#1{%
\count255 =\escapechar
\escapechar=-1\relax
\csarg\if\csname\string#1\expandafter\string\endcsname\relax
\toks0\{\relax\else
\edef\act\{\toks0\%
\csname\string#1\endcsname\}\act
\toks0\xp\xp\xp\{\the\toks0\%
\fi
\csarg\edef\csname\string#1\endcsname{%
\catcode\string#1=\the\catcode
\expandafter\string#1%
\def\csname\string#1\endcsname{%
{\the\toks0}%
\catcode\expandafter\string#1=12\relax
\edef\act\{\toks0\%
\expandafter\string#1=\the\catcode
\expandafter\string#1%
\def\act\{\toks0\%
\def\csname\string#1\endcsname{%
\expandafter\string#1%
\def\csname#1\endcsname{%
\expandafter\string#1%
Here is some test input:
\storecat\%
\catcode'\%=3
\storecat\%
\catcode'\%=4
\storecat\%
\catcode'\%=5
\restorecat\%
\showthe\catcode'\%
\restorecat%
\showthe\catcode'\%
\restorecat\%
\showthe\catcode'\%

As a whimsical aside, the double assignment to \toks0 can also be done in one statement:
\toks0\xp\xp\xp\xp\xp\xp\xp
{\xp\xp\xp\xp\cxname\xp\cstring\xp
\restorecat\string#1\endcxname}

using 15 consecutive \expandafters.

ZzT\TeX{}: A Macro Package for Books

Paul C. Anagnostopoulos

Introduction

ZzT\TeX{} is a macro package for producing books, journals, and technical documentation. The primary advantage of ZzT\TeX{} is its design flexibility, which makes it well-suited to typesetting books according to the specifications of a professional book designer. During the past three years, I and my associates have used the package to produce approximately 25 books, ranging from the 100-page journal, System Dynamics Review, published by John Wiley & Sons, to the 1400-page book, VMS Internals and Data Structures, Version 5.2, published by Digital Press. In this article I hope to give you a taste of some of ZzT\TeX{}'s more interesting technical aspects. Future articles will delve deeper into the details of the macros themselves.

I was initiated into the composition and typesetting business when I agreed to compose my own book for Digital Press. I had written the book using L\TeX{}, and continued to use it for the composition. As a software engineer, I found it impossible not to fall in love with book production: finally, an endeavor that produces a concrete work of art as its end product, rather than some ethereal software. However, I needed more design and page-makeup flexibility than L\TeX{} had to offer, so I undertook to write my own macro package, which I subsequently named ZzT\TeX{}, after a rock group from Texas. Design flexibility is of paramount importance when producing books according to typographers' designs; neither they nor the publishers like to hear "I'm sorry, that element is too difficult to typeset."

The first book produced with ZzT\TeX{} was a real struggle. It took about two weeks to create the design file and produce sample pages. With time, I refined the package and added many new features. Each enhancement was a direct result of a design requirement in a book I had produced, so I believe ZzT\TeX{} is a practical, realistic macro package. Furthermore, my knowledge of the publishing business grew with each book. Now when I receive a design specification I can produce sample pages in less than a day. The package includes approximately 7,200 lines of \TeX{} code and various utilities written in AWK.

The ZzT\TeX{} macro package and manual are available from the author for a nominal fee. The macro package may be freely distributed, but the manual is copyrighted and must be ordered from the author. ZzT\TeX{} uses significant amounts of