The Fifth ATypI Working Seminar
The Computer and the Hand in Type Design:
The Aesthetics and Technology
of Digital Letterforms

During the week of July 31–August 7, at Stanford University, the Committee for Education and Research in Letterforms of the Association Typographique Internationale (ATypI) is sponsoring an International Working Seminar on electronic and traditional methods of letter design. The program includes workshops, seminars and illustrated lectures, and it will conclude with a typographical excursion to San Francisco.

The Seminar will begin on Monday morning, August 1, with the keynote address, "A Turning Point in Type Design", by John Dreyfus, Honorary President of ATypI. Other speakers include the type designers Hermann Zapf, Matthew Carter, Andre Guertler, Christian Mengelt, Gerard Unger and Bram de Does; typographers Jack Stauffacher and Charles Bigelow; lettering artists David Kindersley and John Benson; type punch-cutter Henk Drost; designer Veronika Elsher; and computer scientists Donald Knuth, Patrick Baudelaire and Neil Wiseman.

The theme and purpose of the Seminar are:

- To acquaint educators and designers with the new computerized methods of type production and to review certain traditional lettering crafts, including punch-cutting and stone-cutting.
- To provide practical experience with computer-aided design systems.
- To bring designers and engineers together for future cooperation and creation in type design.

Working installations of the IKARUS, METAFONT, ALTO, CAMEX, and other systems will be available for use during the Seminar. Demonstrations and assistance in using the computers will be provided. Morning seminar sessions will be devoted to working with computer systems, and the systems will be available at other times of the day and night for further exploration.

Prior to the Seminar, information materials on each computer-aided design system, with samples of selected design problems, will be sent to each registered participant. This is to acquaint participants with the systems before their arrival at the Seminar.

The Seminar is intended for design research, not scientific research. Participants do not need scientific training. The emphasis will be on the practice of design with the new computer technology, and with traditional hand technology.

The Seminar language is English, with translators for German and French.

Fees of $950 per person include Seminar, shared double room, meals, reception, and excursion. The rooms and meals are in a Stanford Residence Hall on the Stanford Campus. For a private, single room, the total Seminar fee is $1025 per person. [For partial bookings, e.g. seminar without room and meals, or a stay of less than the full week, send inquiries to the address below.]

To reserve a position at the Seminar, or for additional information, write to

Charles Bigelow
President, ATYPI Committee on
Education and Research in Letterforms
Department of Computer Science
Stanford University
Stanford, California 94305 USA

Reservations must be accompanied by the full fee.

* * * * * * * * * * *

```
MACRO
  O
  L
  U
  M
  N
```

Send Submissions to:
Lynne A. Price
TUG Macro Coordinator
Calma R&D
527 Lakeside Drive
Sunnyvale, CA 94086

TUGBOAT MACRO INDEX

The following list catalogues macros that have appeared in TUGboat. Entries are listed by volume, number, and page as well as author's name. Items that could not be categorized by an obvious headword have been listed under "miscellaneous". Many items refer to parts of large macro packages; users of other packages may find them valuable models for macros of their own.

Readers' comments on the format as well as the contents of this index are welcome.

ACM style	II:1	61, 82–83	A. Keller
Addresses	II:1	54	B. Beeton
	II:2	A-35	M. Díaz
Appendices	II:2	A-21	M. Díaz
Array operations	III:2	34–36	L. Lamport
Baseline, set to top of box	II:1	60, 77	A. Keller
Bibliography	II:2	A-25	M. Díaz
Boxes	II:1	59, 73	A. Keller
Box numbers, automatic allocation	III:1	33	M. Plass
Branching, see If			
Capital letters			
large ~ at beginning of paragraph	II:1	60, 78	A. Keller
	II:3	62	TEXarcana Class
	II:2	A-16	M. Díaz
Roman numerals	II:1	120–121	P. Milligan, L. Price
Centering a sequence of lines	II:2	A-13	M. Díaz
Chapters and Sections	II:1	60–61, 79–81	A. Keller
	II:1	111–118	L. Price
	II:2	A-8–9, 20–22	M. Díaz
Character width determination	IV:1	38	R. McClure
Characters, macros to produce special	II:1	57, 67–70	A. Keller
Chemical notation	II:3	57–58	M. Nichols, B. Beeton
Columns			
balanced	II:3	58–59	L. Price
multiple	II:2	A-38–40	M. Díaz
	II:3	24–25	B. Beeton
	III:2	33	B. Beeton
Comparison of integral values	II:1	119–120	P. Milligan, L. Price
Counters			
automatic allocation	III:1	33	M. Plass
pseudo	II:1	60, 77	A. Keller
	II:1	120	P. Milligan, L. Price
	III:2	30	B. Beeton
Cross references	II:3	24	B. Beeton
Deferred output	II:1	60, 86–86	A. Keller
Division	II:2	47	B. McKay
Equality of integral values	II:1	119–120	P. Milligan, L. Price
Figures	II:2	A-25–27	M. Díaz
Font			
character width determination	IV:1	38	R. McClure
declaring families of a particular point size	II:1	56–57, 65–66	A. Keller
	II:2	A-11	M. Díaz
definition	II:1	119	P. Milligan, L. Price
	II:2	44–45	P. Milligan
display in table form	III:1	35	R. Beeman
Fontcodes	III:2	26	C. Jackson
Footnotes	II:1	58, 71–72	A. Keller
	II:2	A-24–25	M. Díaz
French	II:2	A-12	M. Díaz
Graphics	II:2	48–49	B. McKay
	II:3	63	TEXarcana Class
Hashsize	IV:1	36	B. Beeton
Headings, page	II:2	A-23–24	M. Díaz
Hidden Text	II:3	61	TEXarcana Class
If			
comparison of integral values	II:1	119–120	P. Milligan, L. Price
groupless \if	II:2	46	B. McKay
null string, see Null string			
testing math-style (display, script or scriptscript)	II:2	46	B. McKay
Index production	I:1	Appendix A	T. Winograd, W. Paxton
	II:2	A-28	M. Díaz
Justification			
of reviewer's names	II:3	62	TEXarcana Class
right ~	II:3	63	TEXarcana Class
Layout macros	IV:1	37	A. Mohr
Letters	II:2	A-32–35	M. Díaz
Letterhead	II:2	A-33	M. Díaz
Line numbering	III:1	43	TEXarcana Class
Lists	II:1	59, 72–72	A. Keller
	II:1	98–110	L. Price
	II:2	A-15	M. Díaz
Margins	II:2	A-19	M. Díaz
Matrices	II:2	A-30	M. Díaz
Memos	II:2	A-32–35	M. Díaz
Miscellaneous			
automatic printing of macro names	II:3	60–61	L. Price
avoiding "Argument of ⟨control sequence⟩ has an extra }."	II:2	50	M. Spivak
conditional evaluation of macros	II:2	50	M. Spivak
input-dependent macro redefinition	II:3	59–60	L. Price
\input within \if	II:2	50	M. Spivak
single tokens, identifying	II:2	52	M. Spivak
Multiplication	II:2	47	B. McKay
Nofill			
macros	II:1	59–60, 74–76	A. Keller
	II:2	A-16–18, 36	M. Díaz
program (SAIL)	II:1	87–93	L. Price, P. Milligan
program (Pascal)	II:1	94–97	L. Price, P. Milligan
program errata (SAIL and Pascal)	II:2	43–44	
Notes			
output to the writer on a separate file	II:1	60, 76, 85	A. Keller
printed at end of document	II:2	A-25	M. Díaz
Null string, testing for	II:1	60, 77	A. Keller
	II:2	51–52	M. Spivak
Numbering, page	II:1	57, 70–71	A. Keller
line	III:1	43	TEXarcana Class
Output routines	II:1	57–58, 60–62, 71, 82–85	A. Keller
	II:2	A-18, 40	M. Díaz
	III:2	33	B. Beeton
Overlining	II:2	A-13	M. Díaz
Page layout	IV:1	37	A. Mohr
Page numbering	II:1	57, 70–71	A. Keller
	II:2	A-18, 23	M. Díaz
Paragraphs			
beginning with large capital letters	II:1	60, 78	A. Keller
	II:2	A-16	M. Díaz
in tables	III:2	38	Problems column
indented	II:1	58, 72	A. Keller
	II:2	A-13–15	M. Díaz
numbered, see Lists			
Parentheses, assorted sizes	II:2	A-11	M. Díaz
Pictures, plotting	II:2	48–49	B. McKay
Point, declaring font families of a particular ~ size	II:1	56–57, 65–66	A. Keller
	II:2	A-11	M. Díaz
Proofs	II:2	A-31–32	M. Díaz
Punctuation, 'hanging'	III:2	38	Problems column
	IV:1	39	Problems column
Push-down stacks	III:2	34–36	L. Lamport
Recursion	II:2	46–48	B. McKay
	II:2	53	M. Spivak

References	II:2	A-25	M. Díaz
Registration marks . . .	III:2	30	B. Beeton
Roman numerals, uppercase .	II:1	120–121	P. Milligan, L. Price
Seating charts	III:1	39	R. Beeman
Spanish	II:2	A-12	M. Díaz
Strings			
testing for ~ equivalence .	II:3	61	L. Price
testing for the null ~ . .	II:1	60, 77	A. Keller
. . . .	II:2	51–51	M. Spivak
Struts	IV:1	35–36	B. Beeton
Syntax charts	II:3	39–56	M. Plass
Table of Contents . . .	II:1	60, 62, 86	A. Keller
. . . .	II:1	111–118	L. Price
. . . .	II:2	A-27–28	M. Díaz
. . . .	II:3	24	B. Beeton
Tables	II:2	A-25–27	M. Díaz
paragraphs in ~ . . .	III:2	38	Problems column
Testing			
integral values	II:1	119–120	P. Milligan, L. Price
math-style (display, script or scriptscript) . . .	II:2	46	B. McKay
for string equivalence . .	II:3	61	L. Price
for the null string . . .	II:1	60,77	A. Keller
. . . .	II:2	51–52	M. Spivak
Theorems	II:2	A-31–32	M. Díaz
Top, baseline set to ~ of box .	II:1	60, 77	A. Keller
TUGboat submissions . .	II:1	53–54	B. Beeton
. . . .	II:3	25	B. Beeton
Underlining	II:1	59, 73	A. Keller
. . . .	II:2	A-13	M. Díaz
Uppercase letters			
large ~ at beginning of paragraph	II:1	60, 78	A. Keller
. . . .	II:2	A-16	M. Díaz
Roman numerals . . .	II:1	120–121	P. Milligan, L. Price
Verbatim			
mode	II:1	59–60, 74–76	A. Keller
. . . .	II:2	A-16–18, 36	M. Díaz
program (SAIL) . . .	II:1	87–93	L. Price, P. Milligan
program (Pascal) . . .	II:1	94–97	L. Price, P. Milligan
Vertical text	II:3	64	TEXarcana Class

* * * * * * * * * * *

HOW TO BUILD A \STRUT

Barbara N. Beeton
American Mathematical Society

Struts are things that keep objects a fixed distance apart, like the wings of a biplane. Because of the way that TEX puts boxes together vertically, struts are sometimes needed to maintain the desired distance. The concept was introduced in the definition and explanation of \| on pp. 108–109 of the TEX manual [*TEX and Metafont*]: "TEX doesn't use `\baselineskip` and `\lineskip` before and after horizontal rules." The failure of `\baselineskip` to apply the desired spacing also affects adjacent `\hbox` pars which contain more than one line, but in that case, a visible vertical rule would not be a satisfactory remedy. As it happens, an *invisible* vertical rule is just the thing, but first let us look at the problem in more detail.

In text, `\baselineskip` is typically set at 2 points greater than the text body size: 10-on-12, 9-on-11, 8-on-10. For some special work, though, it may be desirable to set material more densely, even "solid" — 10-on-10, etc. Only rarely are lines of text set any closer than that, and struts won't help with that problem in any case, so it will be ignored here. A strut for solid text should be the same height and depth as the tallest and deepest characters in the font; in METAFONT text fonts, a parenthesis ( ) or square bracket [ ] qualifies, so adjacent vertical boxes containing one of these on the last and first lines respectively will be separated by the desired distance. Consider the following example, which consists of three `\hbox` pars: the first junction lacks sufficient ascenders and descenders to force the baselines apart to the `\baselineskip` distance (this is `\tenpoint\rm \baselineskip 10pt`), but the second junction looks no different from two lines in the middle of a paragraph.

This paragraph has no
descenders in the last line.
one can scarce see an
answer to this (let's cheat).
(This example may be
contrived, but it works.)

Now, define a strut with the maximum height and depth of any character in the font, and insert it at the beginning of the first and end of the last line in each paragraph:

This paragraph has no
descenders in the last line.
one can scarce see an
answer to this (let's cheat).
(This example may be
contrived, but it works.)

Finally, reset `\baselineskip 12pt` and apply a strut that is 2 points longer:

This paragraph has no
descenders in the last line.
one can scarce see an
answer to this (let's cheat).
(This example may be
contrived, but it works.)

There are probably many ways actually to define struts, but only two will be shown here. In one approach, strut is defined within the range of each "size" definition (this is Knuth's approach):

```
\def \tenpoint{\baselineskip 12pt ...
    \def\strut{\lower 3.5pt\vbox to 12pt{}}
    ... }
```

The following is equivalent for `\tenpoint`, but more efficient (because rules take less memory space