

Zpravodaj Československého sdružení uživatelů TeXu

Oldřich Ulrych

Jak jsem se sázel svisle (viz podpis)

Zpravodaj Československého sdružení uživatelů TeXu, Vol. 2 (1992), No. 4, 201–205

Persistent URL: <http://dml.cz/dmlcz/149647>

Terms of use:

© Československé sdružení uživatelů TeXu, 1992

Institute of Mathematics of the Czech Academy of Sciences provides access to digitized documents strictly for personal use. Each copy of any part of this document must contain these *Terms of use*.



This document has been digitized, optimized for electronic delivery and stamped with digital signature within the project *DML-CZ: The Czech Digital Mathematics Library* <http://dml.cz>

Jak jsem se sázel svisle (viz podpis)

Oldřich Ulrych

Před několika týdny se ke mně obrátil kolega s ukázkami tabulek, které by bylo možné snadno vytvářet v $\text{\TeX}$ u, které však musí sázet pomocí jiného software pro DTP, neboť potřebuje mít v záhlaví tabulky svisle vysázené texty. Příklad tabulky, na které je demonstrován uvedený problém, je vysázen

	Poled- ník	Rovno- běžka	Sva- čina	Oběd
Praha	15	25	10.30	12.20
Brno	18	24	10.45	13.00

vpravo. Pokusy tisknout tabulku tak, že nejdříve vytiskneme svislé texty a pak vodorovné, selhávaly kvůli mechanickým vřlím papíru ve vodící dráze tiskárny.

Protože šlo o to, jak sázet v záhlaví tabulek pouze takový hladký text, ve kterém každý znak ze vstupního souboru je přímo znakem fontu (tj. není možné používat žádné složené objekty $\text{\TeX}$ u a tudíž ani ligatury), navrhl jsem celkem bezpracné řešení, které svislou sazbu umožňovalo. Toto řešení je založeno na následujících podmínkách a krocích:

1. Aby bylo možné sázet české texty, musíme použít osmibitové fonty, např. fonty DC Norberta Schwarze nebo CS fonty upravené Ladislavem Lhotkou a Karlem Horákem. Tato potřeba plyne z toho, že každý znak zdrojového textu musí být přímo znakem ve fontu.
2. Upravíme zdrojový text fontu tak, aby při generování METAFONtem se bitová reprezentace písmen vytvářela pootočená o 90° a generujeme font v požadovaném zvětšení.
3. Napsat makro pro $\text{\TeX}$, které bude umisťovat písmena do sloupečku nad sebe tak, že zpracováním `dvi` souboru vznikne v požadovaných místech (s použitím fontu s pootočenými znaky) svislá sazba (přirozeně bez možnosti automatického dělení slov a zalamování do odstavců).

Podotkněme, že se nejedná o komplexní řešení vertikální sazby, ale je zde řešena pouze jedna velmi speciální úloha $\text{\TeX}$ em. Omezení, která z navrženého postupu vyplývají jsou natolik vážná, že je nutno tento příspěvek chápat spíše jako hru s METAFONtem a $\text{\TeX}$ em. Možných

postupů, jak sázet svisle existuje mnoho, tento se mi zdál pro daný účel nejjednodušší.

Jestliže jsme zhruba načrtli postup, věnujme se nyní všem detailům. Protože popis všech změn a úprav není dlouhý (včetně makra pro $\text{T}_{\text{E}}\text{X}$) uveřejníme zde i všechny zdrojové texty. V dalším předpokládáme, že chceme sázet svislé texty fontem `dcr10` (desetibodová antikva z rodiny písem DC).

1. Zkopírujme soubor `dcr10.mf` např. do souboru `dcr10-v.mf` a na jeho začátek napíšeme příkaz

```
input vbase
```

2. V adresáři zdrojových textů pro METAFONT vytvoříme zmíněný soubor `vbase.mf` s následujícím obsahem:

```
% The changes in this file rotate characters in CM and DC fonts
% 90 degree counterclockwise.
%
extra_beginchar:="currenttransform:=identity slanted slant rotated 90;"&
      "def t_ = transformed currenttransform enddef;";
```

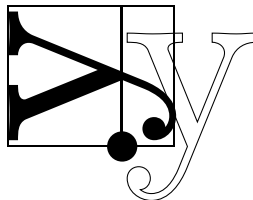
Tyto řádky způsobí, že bitová mapa každého znaku, generovaného METAFONTEM bude otočena o 90° proti směru hodinových ručiček. Toto je vidět na obrázku níže. Obrys písmene *y* znázorňuje výstup bez otočení, plné písmeno *y* je výsledkem výše uvedeného příkazu. Puntíkem je znázorněn referenční bod znaku (tento zůstává nezměněn, stejně jako výška, šířka a hloubka znaku i po otočení — otáčí se právě kolem referenčního bodu).

3. Generujeme font `dcr10-v.mf` pro požadované výstupní zařízení a požadované zvětšení.

Tímto máme připraven font `dcr10-v`, kterým můžeme sázet svisle (zdola nahoru).

Celá idea vertikální sazby je založena na umisťování referenčních bodů jednotlivých znaků sázeného textu nad sebe s případným vložením mezery (kerningu). Navrhl jsem makro `downup.tex`, které obsahuje tyto definice:

`\downup{...}` je řídicí slovo s jedním argumentem, jímž je text, který se bude sázet zdola nahoru. Uvnitř tohoto textu je možné použít řídicí



znak `\\` k naznačení zlomu řádku. V takovém případě jsou sázeny jednotlivé svislé řádky vedle sebe podle pravidel obvyklých při vkládání mezer mezi horizontální řádky ve vertikální seznamu (tj. využívají se hodnoty `\baselineskip`, `\lineskiplimit` a `\lineskip`). `\fordownup{...}` je řídicí slovo s jedním argumentem, který se vkládá před začátek textu zpracovávaného řídicím slovem `\downup`.

Zdrojový text pro tento odstavec, který je sázen částečně vodorovně a částečně svislé, je uveden níž, aby bylo čtenáři zřejmé, jak jsme jej vysázeli:

```
\font\fsvisle=dcr10-v
\fordownup{\fsvisle}
Zdrojový text pro tento odstavec,
\downup{který je sázen\\ částečně \\vodorovně a\\ částečně svislé,}%
\downup{je uveden níž,\\ aby bylo \\ čtenáři zřejmé,} jak jsme jej
vysázeli:
```

Pro úplnost ještě uvádím celé makro `downup.tex` v podobě, v jaké zde bylo použito:

```
% Simple macro for vertical typesetting of pure text.
% For detail description, see the end of this file.
%
% Oldrich Ulrych                                October 9, 1992
% Mathematical Institute of Charles University
% e-mail: oulrych@cspguk11.bitnet
%
\edef\catcodeat{\the\catcode'\@ }      \catcode'\@=11
%
% definitions of new registers
%
\newbox\vertb@x          % box for vertical text
\newbox\auxb@x           % auxiliary box
\newdimen\m@xht          % "height" of vertical "row"
\newdimen\m@xdp          % "depth" of vertical "row"
\newdimen\@uxdim         % auxiliary dimension
\newdimen\pr@vdp@pth \pr@vdp@pth=0pt % "depth" of previous "row"
%
{\def\l{\global\let\spa@ce= } \1 }% \spa@ce is space
\def\\{ }%                % \\ is defined
```

```

\def\downup{\begingroup\fd@downup % user's setting
  \ifvmode\leavevmode\fi % to vboxes are set in line
  \m@xht=0pt % "left height" is zero
  \m@xdp=0pt % "right depth" is zero
  \setbox\vertb@x=\vbox{}% % the box is empty initially
  \def\pr@vletter{}% % previous letter is none
  \afterassignment\sp@cetreat % (for kerning)
  \let\next= } % eat the opening bracket
\def\sp@cetreat{\futurelet\next\t@stclosing} % next character
\def\t@stclosing{%
  \ifx\next\egroup
    \let\next\lastputvb@x % all argument is scanned over
  \else
    \expandafter\ifx\next\spa@ce
      \let\next\@naspace % next character is space
    \else
      \ifx\next\\%
        \let\next\separateb@x% next token is "line break"
      \else
        \let\next\@@tone % other cases
      \fi\fi\fi\next}
\def\lastputvb@x{\putvb@x\endgroup % make last vbox
  \let\next= } % and eat the closing bracket
\def\putvb@x{% % make standard "height and depth" of box
  \@uxdim=-\pr@vd@pth
  \advance\@uxdim by -\m@xht
  \advance\@uxdim by \baselineskip
  \ifdim\@uxdim<\lineskiplimit
    \kern\lineskip
  \else
    \kern\@uxdim % interline skip
  \fi
  \global\pr@vd@pth=\m@xdp
  \hbox{\kern\m@xht\box\vertb@x\kern\m@xdp}}% and form box properly
\def\@naspace{\setbox\vertb@x
  =\vbox{\kern1ex\unvbox\vertb@x}% % leave vert space
  \def\pr@vletter{}% % space doesn't influence the kerning
  \afterassignment\sp@cetreat\let\next= } % eat the space
\def\@@tone#1{\setbox\vertb@x=\vbox{\r@tletter{#1}\unvbox\vertb@x}%
  \def\pr@vletter{#1}\sp@cetreat} % add the token to the vert. list
\def\r@tletter#1{\setbox\auxb@x=\hbox{\pr@vletter{#1}}%letters with kerning
  \@uxdim=-\wd\auxb@x % remember their width
  \setbox\auxb@x=\hbox{\pr@vletter}% % take the previous letter
  \advance \@uxdim by \wd\auxb@x % and subtract its width
  \setbox\auxb@x=\hbox{#1}% % take the letter

```

```

\advance \@uxdim by \wd\auxb@x      % and subtract its width
\ifdim\m@xht<\ht\auxb@x \global\m@xht=\ht\auxb@x \fi % maximal height
\ifdim\m@xdp<\dp\auxb@x \global\m@xdp=\dp\auxb@x \fi % maximal depth
\ht\auxb@x=\wd\auxb@x \dp\auxb@x=0pt \wd\auxb@x=0pt% change its dim's
\box\auxb@x                          % print letter
\kern-\@uxdim}                       % make vertical kerning
\def\separateb@x#1{\putvb@x          % make vbox
\def\pr@vletter{}%                   % prevletter is none
\futurelet\next\tryp@ce}            % look ahead
\def\tryp@ce{\ifx\next\spa@ce
\let\next\eat@neitem\else           % ignore following spaces
\let\next\sp@cetreat\fi\next}
\def\eat@neitem{\afterassignment\sp@cetreat\let\next=} % eat space
\def\fordownup#1{\def\frdownup{#1}}
\fordownup{\tenrm}
%
\catcode'\@=\catcodeat \let\catcodeat=\undefined
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% C o m m e n t s %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% \fordownup{...} should contain the font for vertical printing
% \downup{...}    is the simple text printed from bottom to the top
%                (ligatures are printed as separated letters)
% \\\             can be used to separate lines in the argument
%                of \downup
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

Oldřich Ulrych