

adfsymbols

Clea F. Rees*

3rd August, 2010

Abstract

Hirwen Harendal, Arkandis Digital Foundry (ADF) has produced Symbols ADF. This guide outlines the T_EX/L^AT_EX support provided with version 1.001 of the fonts in postscript type 1 format.

1 Introduction

This document explains how to use the T_EX/L^AT_EX support included with version 1.001 of the Symbols ADF font collection in postscript type 1 format. The fonts were developed by Hirwen Harendal of the Arkandis Digital Foundry (ADF), and information about the fonts themselves, together with copies of the fonts in opentype format, can be found at <http://pagesperso-orange.fr/arkandis/ADF/tugfonts.htm>. The fonts are released under the GPL. For details, see README, NOTICE and COPYING.

The T_EX/L^AT_EX support package consists of all files listed in `manifest.txt` and these files are released under the L^AT_EX Project Public Licence as explained in the included licensing notices and README. Please let me know of any problems so that I can solve them if I can. If you can correct the problems and send me the fix, that would be even better. Unlike the fonts themselves, the T_EX/L^AT_EX support is somewhat experimental.

`adfsymbols` includes a copy of the fonts in type 1 format, documentation and support files for T_EX/L^AT_EX including two L^AT_EX package files, `adfarrows.sty` and `adfbullets.sty`.

2 The support package

`adfsymbols` provides access to the symbols in `ArrowsADF` and `BulletsADF` in L^AT_EX through two packages, `adfarrows` and `adfbullets`. Both packages require `pifont` and `adfarrows` also requires `fp`.

2.1 adfarrows

`adfarrows` provides the command `\adfarrows{}` which takes a single numerical argument. There are 52 arrows in `ArrowsADF` which can be produced by feeding the relevant number between 1

*cfrees <at> imapmail <dot> org

and 52 to `\adfarrow{}`¹:

1: ↗	14: ✓	27: ↘	40: ↙
2: ↖	15: ←	28: ↗	41: ←
3: ➤	16: ↖	29: ➤	42: ↖
4: ↘	17: ↑	30: ↘	43: ↑
5: ↓	18: ↗	31: ↓	44: ↗
6: ↗	19: ➡	32: ↗	45: ➡
7: ←	20: ↘	33: ←	46: ↘
8: ↘	21: ↓	34: ↘	47: ↓
9: ↑	22: ↗	35: ↑	48: ↗
10: ↗	23: ←	36: ↗	49: ←
11: →	24: ↘	37: →	50: ↘
12: ↘	25: ↑	38: ↘	51: ↑
13: ↓	26: ↗	39: ↓	52: ↗

For example, `\adfarrow{5}\adfarrow{9}` produces: \Downarrow .

2.1.1 Alternative commands

To make things a little more convenient, additional commands are provided to access the various arrows. The effect is to typeset one of the arrows show above but it is not necessary to look up or remember the correct numerical argument.

First, four commands are provided to access the four half arrows. In each case, the number of the arrow is given first. This may be used directly with the `\adfarrow{}` command as explained above. The alternative command is given next. This command may be used to typeset the same ornament. For example both `\adfarrow{1}` and `\adfhalfarrowright` produce \Rightarrow . Finally, the arrow produced by the two commands is typeset to their right.

1	<code>\adfhalfarrowright</code>	\Rightarrow	2	<code>\adfhalfarrowleft</code>	\Leftarrow
27	<code>\adfhalfarrowrightsolid</code>	\blacktriangleright	28	<code>\adfhalfarrowleftsolid</code>	\blacktriangleleft

The remaining arrows consist of six families each containing eight arrows — one for each of the eight directions of the compass. These may be accessed in two ways, in addition to using `\adfarrow{}`.

First, eight commands are provided. Each command takes a single numerical argument in the range 1–6. The argument corresponds to one of the six families of arrows. So using the same number with the different commands will typeset arrows from the same family pointing in different directions.

direction	command	example usage
north	<code>\adfarrown</code>	<code>\adfarrown1</code> \Uparrow
northeast	<code>\adfarrowne</code>	<code>\adfarrowne2</code> \nearrow

¹The argument 0 will simply typeset a space and should be avoided as using it may interfere with \TeX 's spacing algorithms. The problem is that \TeX will not recognise it as a space and so will treat it instead as a character.

direction	command	example usage	
east	<code>\adfarrowe</code>	<code>\adfarrowe3</code>	➤
southeast	<code>\adfarrowse</code>	<code>\adfarrowse4</code>	↘
south	<code>\adfarrows</code>	<code>\adfarrows5</code>	↓
southwest	<code>\adfarrowsw</code>	<code>\adfarrowsw6</code>	↙
west	<code>\adfarroww</code>	<code>\adfarroww1</code>	←
northwest	<code>\adfarrownw</code>	<code>\adfarrownw3</code>	↖

Second, a further command is provided which allows you to specify both the family and direction as separate arguments. This is in fact the base command `\adfarrow` again. Above, we used the command with just one argument: `\adfarrow{}`. In effect, we left the optional argument empty: `\adfarrow[]{}.` The optional argument may, however, take one of six family names or their corresponding numbers. In this case, the second argument specifies the arrow's direction. *Note that you must specify a family if you specify a direction.* If the optional argument is omitted, the command expects the numerical argument corresponding to the arrow you wish to typeset.

family names for first argument	
family number	family name
1	opentail
2	plain
3	comic
4	solidtail
5	thick
6	tail

The arrow's direction may be specified in either a long or an abbreviated form:

direction names for the second argument		
direction	long form	short form
north	north	n
northeast	northeast	ne
east	east	e
southeast	southeast	se
south	south	s
southwest	southwest	sw
west	west	w
northwest	northwest	nw

The different possibilities may be illustrated by the examples in the following table where each row consists of a selection of equivalent commands which may be used to produce identical output in different ways. In each case, the number of the arrow is given first. This may be used directly with the `\adfarrow{}` command as explained above. One of the eight commands from the previous section follows. Two additional uses of `\adfarrow` are given next using the `\adfarrow[family]{direction}` form described in this section. Finally, the arrow each of these

commands typesets is displayed to their right.

4	<code>\adfarrowse1</code>	<code>\adfarrow[1]{southeast}</code>	<code>\adfarrow[opentail]{se}</code>	↘
51	<code>\adfarrown6</code>	<code>\adfarrow[tail]{north}</code>	<code>\adfarrow[6]{n}</code>	↑
42	<code>\adfarrownw5</code>	<code>\adfarrow[thick]{nw}</code>	<code>\adfarrow[5]{northwest}</code>	↖
15	<code>\adfarroww2</code>	<code>\adfarrow[2]{w}</code>	<code>\adfarrow[plain]{west}</code>	←
31	<code>\adfarrows4</code>	<code>\adfarrow[solidtail]{south}</code>	<code>\adfarrow[4]{s}</code>	↓
22	<code>\adfarrowsw3</code>	<code>\adfarrow[comic]{sw}</code>	<code>\adfarrow[3]{southwest}</code>	↙

2.2 adfbullets

adfbullets provides the command `\adfbullet{}` which takes a single numerical argument. There are 52 bullets in BulletsADF which can be produced by feeding the relevant number between 1 and 52 to `\adfbullet{}`²:

1: ♠	14: ❖	27: •	40: ▶
2: ⊕	15: ⊗	28: •	41: •
3: ♣	16: ⊕	29: ■	42: •
4: ✖	17: ⊗	30: ◆	43: •
5: ♠	18: ⊗	31: ◀	44: ·
6: ✖	19: ⊕	32: ▶	45: ◦
7: ♠	20: ◦	33: ▲	46: ■
8: ⊕	21: ♠	34: ▼	47: ■
9: ♠	22: ⊕	35: ◀	48: ♠
10: ⊕	23: ❖	36: ▶	49: ◆
11: ✖	24: ❖	37: ◀	50: ◆
12: ♣	25: ♠	38: ▶	51: ◦
13: ♣	26: ⊕	39: ◀	52: ◦

For example, `\adfbullet{17}\adfbullet{19}\adfbullet{23}` produces: ⊗⊕❖.

3 Usage

Arrows and bullets may be especially useful when designing:

- * diagrams;
- * presentations;
- * handouts;
- * brochures, leaflets and pamphlets;
- * forms, including fill-in [pdf] forms.

²Again, a 0 will simply typeset a space and should be avoided as using it may interfere with T_EX's spacing algorithms.

Many of these uses are made easier by the use of various packages. For example, `enumitem` allows you to easily change the format of lists and the previous list was typeset as follows:

```
\begin{itemize}[label=\adfbullet{25}]
\item diagrams;
\item presentations;
\item handouts;
\item brochures, leaflets and pamphlets;
\item forms, including fill-in \textsc[pdf] forms.
\end{itemize}
```

Refer to the package documentation for further details.

`adfarrows` and `adfbullets` can be used in `beamer` presentations to produce lists with custom bullet markers; as icons and markers in `pgf` diagrams; with `sectsty`, `titlesec` and/or `fancyhdr` to typeset custom headings, headers and footers. For example,

```
\pagestyle{fancy}
\fancyhf[ch]{}
\fancyhf[lf]{}
\fancyhf[rf]{}
\fancyhf[lh]{}
\fancyhf[rh]{}
\fancyhf[ch]{%
\itshape adfsymbols\hspace*{1.5em}{\Large\adfbullet{14}}\hspace*{1.5em}\dyddiad}
\fancyhf[cf]{%
\itshape {\large\adfbullet{39}} \thepage~\ofname~\pageref{LastPage} %
{\large\adfbullet{40}}}
\renewcommand{\headrulewidth}{0pt}
```

was used to customise this document's headers and footers with `fancyhdr`. (`\dyddiad` is an uninteresting command which simply holds the date used for the document.)