Name

groff_char – GNU *roff* special character and glyph repertoire

Description

The GNU *roff* typesetting system has a large glyph repertoire suitable for production of varied literary, professional, technical, and mathematical documents. *groff* works with *characters;* an output device renders *glyphs.* *groff*'s input character set is restricted to that defined by the standards ISO Latin-1 (ISO 8859-1) and CCSID "code page" 1047 (an EBCDIC arrangement of Latin-1). For ease of document maintenance in UTF-8 environments, it is advisable to use only the Unicode basic Latin code points, a subset of all of the foregoing historically referred to as US-ASCII, which has only 94 visible, printable code points. In *groff,* these are termed *ordinary characters.* Often, many more are desired in output.

AT&T *troff* in the 1970s faced a similar problem: the available typesetter's glyph repertoire differed from that of the computers that controlled it. *troff*'s solution was a form of escape sequence known as a *special character* to access several dozen additional glyphs available in the fonts prepared for mounting in the phototypesetter. These glyphs were mapped onto a two-character name space for a degree of mnemonic convenience; for example, the escape sequence **\(aa** encoded an acute accent and **\(sc** a section sign.

groff has lifted historical *roff* limitations on special character name lengths, but recognizes and retains compatibility with the historical names. *groff* expands the lexicon of glyphs available by name and permits users to define their own special character escape sequences with the **char** request. Special character names are *groff* identifiers; see section "Identifiers" in *groff*(7). Our discussion uses the terms "glyph name" and "special character name" interchangeably; we assume no character translations or redefinitions.

This document lists all of the glyph names predefined by *groff*'s font description files and presents the systematic notation by which it enables access to arbitrary Unicode code points and construction of composite glyphs. Glyphs listed may be unavailable, or may vary in appearance, depending on the output device and font chosen when the page was formatted. This page was rendered for device **pdf** using font **TR**.

A few escape sequences that are not *groff* special characters also produce glyphs; these exist for syntactical or historical reasons. **\'**, **\`**, **\–**, and **_** are translated on input to the special character escape sequences **\[aa]**, **\[ga]**, **\[–]**, and **\[ul]**, respectively. Others include ****, **\.** (backslash-dot), and **\e**; see *groff*(7). A small number of special characters represent glyphs that are not encoded in Unicode; examples include the baseline rule **\[ru]** and the Bell System logo **\[bs].**

In *groff*, you can test output device support for any character (ordinary or special) with the conditional expression operator "**c**".

```
.ie c \[bs] \{Welcome to the \[bs] Bell System;
did you get the Wehrmacht helmet or the Death Star?\}
.el No Bell System logo.
```

For brevity in the remainder of this document, we shall refer to systems conforming to the ISO 646:1991 IRV, ISO 8859, or ISO 10646 ("Unicode") character encoding standards as "ISO" systems, and those employing IBM code page 1047 as "EBCDIC" systems. That said, EBCDIC systems that support *groff* are known to also support UTF-8.

While *groff* accepts eight-bit encoded input, not all such code points are valid as input. On ISO platforms, character codes 0, 11, 13–31, and 128–159 are invalid. (This is all C0 and C1 controls except for SOH through LF [Control+A to Control+J], and FF [Control+L].) On EBCDIC platforms, 0, 8–9, 11, 13–20, 23–31, and 48–63 are invalid. Some of these code points are used by *groff* for internal purposes, which is one reason it does not support UTF-8 natively.

Fundamental character set

The ordinary characters catalogued above, plus the space, tab, newline, and leader (Control+A), form the fundamental character set for *groff* input; anything in the language, even over one million code points in Unicode, can be expressed using it. On ISO systems, code points in the range 33–126 comprise a common set of printable glyphs in all of the aforementioned ISO character encoding standards. It is this character set and (with some noteworthy exceptions) the corresponding glyph repertoire for which AT&T *troff* was implemented. On EBCDIC systems, printable characters are in the range 66–201 and 203–254; those without counterparts in the ISO range 33–126 are discussed in the next subsection.

All of the following characters map to glyphs as you would expect.

```
! # $ % & ( ) * + , . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [ ] _
a b c d e f g h i j k l m n o p q r s t u v w x y z { | }
```

The remaining ordinary characters surprise computing professionals and others intimately familiar with the ISO character encodings. The developers of AT&T *troff* chose mappings for them that would be useful for typesetting technical literature in a broad range of scientific disciplines: Bell Labs used the system for preparation of AT&T's patent filings with the U.S. government. Further, the prevailing character encoding standard in the 1970s, USAS X3.4-1968 ("ASCII"), deliberately supported semantic ambiguity at some code points, and outright substitution at several others, to suit the localization demands of various national standards bodies.

The table below presents the seven exceptional code points with their typical keycap engravings, their glyph mappings and semantics in *roff* systems, and the escape sequences producing the Unicode basic Latin character they replace. The first, the neutral double quote, is a partial exception because it does represent itself, but since the *roff* language also uses it to quote macro arguments, *groff* supports a special character escape sequence as an alternative form so that the glyph can be easily included in macro arguments without requiring the user to master the quoting rules that AT&T *troff* required in that context. (Some requests, like **ds**, also treat **"** non-literally.) Furthermore, not all of the special character escape sequences are portable to AT&T *troff* and all of its descendants; these *groff* extensions are presented using its special character form **\[]**, whereas portable special character escape sequences are shown in the traditional **\(** form. **\–** and **\e** are portable to all known *troff*s. **\e** means "the glyph of the current escape character"; it therefore can produce unexpected output if the **ec** request is used. On devices with a limited glyph repertoire, glyphs in the "keycap" and "appearance" columns on the same row of the table may look identical; except for the neutral double quote, this will *not* be the case on more-capable devices. Review your document using as many different output devices as possible.

Keycap	Appearance and meaning	Special character and meaning
"	" neutral double quote	`\[dq]` neutral double quote
'	’ closing single quote	`\[aq]` neutral apostrophe
–	- hyphen	`\-` or `\[-]` minus sign/Unix dash
\	(escape character)	`\e` or `\[rs]` reverse solidus
^	ˆ modifier circumflex	`\(ha` circumflex/caret/"hat"
`	‘ opening single quote	`\(ga` grave accent
~	˜ modifier tilde	`\(ti` tilde

The hyphen-minus is a particularly unfortunate case of overloading. Its awkward name in ISO 8859 and later standards reflects the many distinguishable purposes to which it had already been put by the 1980s, including a hyphen, a minus sign, and (alone or in repetition) dashes of varying widths. For best results in *roff* systems, use the "**–**" character in input outside an escape sequence *only* to mean a hyphen, as in the phrase "long-term". For a minus sign in running text or a Unix command-line option dash, use **\–** (or **\[–]** in *groff* if you find it helps the clarity of the source document). (Another minus sign, for use in mathematical equations, is available as **\[mi]**). AT&T *troff* supported em-dashes as **\(em**, as does *groff*.

The special character escape sequence for the apostrophe as a neutral single quote is typically needed only in technical content; typing words like "can't" and "Anne's" in a natural way will render correctly, because in ordinary prose an apostrophe is typeset either as a closing single quotation mark or as a neutral single quote, depending on the capabilities of the output device. By contrast, special character escape sequences should be used for quotation marks unless portability to limited or historical *troff* implementations is necessary; on those systems, the input convention is to pair the grave accent with the apostrophe for single quotes, and to double both characters for double quotes. AT&T *troff* defined no special characters for quotation marks or the apostrophe. Repeated single quotes (''thus'') will be visually distinguishable from double quotes ("thus") on terminal devices, and perhaps on others (depending on the font selected).

AT&T *troff* input	recommended *groff* input
`A Winter's Tale`	`A Winter's Tale`
`` `U.K. outer quotes' ``	`\[oq]U.K. outer quotes\[cq]`
`` `U.K. ``inner'' quotes' ``	`\[oq]U.K. \[lq]inner\[rq] quotes\[cq]`
`` ``U.S. outer quotes'' ``	`\[lq]U.S. outer quotes\[rq]`
`` ``U.S. `inner' quotes'' ``	`\[lq]U.S. \[oq]inner\[cq] quotes\[rq]`

If you frequently require quotation marks in your document, see if the macro package you're using supplies strings or macros to facilitate quotation, or define them yourself (except in man pages).

Using Unicode basic Latin characters to compose boxes and lines is ill-advised. *roff* systems have special characters for drawing horizontal and vertical lines; see subsection "Rules and lines" below. Preprocessors like *tbl*(1) and *pic*(1) draw boxes and will produce the best possible output for the device, falling back to basic Latin glyphs only when necessary.

Eight-bit encodings and Latin-1 supplement

ISO 646 is a seven-bit code encoding 128 code points; eight-bit codes are twice the size. ISO 8859-1 and code page 1047 allocated the additional space to what Unicode calls "C1 controls" (control characters) and the "Latin-1 supplement". The C1 controls are neither printable nor usable as *groff* input.

Two Latin-1 supplement characters are handled specially on input. *troff* never produces them as output.

NBSP encodes a no-break space; it is mapped to **\~**, the adjustable non-breaking space escape sequence.

SHY encodes a soft hyphen; it is mapped to **\%**, the hyphenation control escape sequence.

The remaining characters in the Latin-1 supplement represent themselves. Although they can be specified directly with the keyboard on systems configured to use Latin-1 as the character encoding, it is more portable, both to other *roff* systems and to UTF-8 environments, to use their special character escape sequences, shown below. The glyph descriptions we use are non-standard in some cases, for brevity.

¡	`\[r!]`	inverted exclamation mark	Ñ	`\[~N]`	N tilde
¢	`\[ct]`	cent sign	Ò	`\[`O]`	O grave
£	`\[Po]`	pound sign	Ó	`\['O]`	O acute
¤	`\[Cs]`	currency sign	Ô	`\[^O]`	O circumflex
¥	`\[Ye]`	yen sign	Õ	`\[~O]`	O tilde
¦	`\[bb]`	broken bar	Ö	`\[:O]`	O dieresis
§	`\[sc]`	section sign	×	`\[mu]`	multiplication sign
¨	`\[ad]`	dieresis accent	Ø	`\[/O]`	O slash
©	`\[co]`	copyright sign	Ù	`\[`U]`	U grave
ª	`\[Of]`	feminine ordinal indicator	Ú	`\['U]`	U acute
«	`\[Fo]`	left double chevron	Û	`\[^U]`	U circumflex
¬	`\[no]`	logical not	Ü	`\[:U]`	U dieresis
®	`\[rg]`	registered sign	Ý	`\['Y]`	Y acute
¯	`\[a-]`	macron accent	Þ	`\[TP]`	uppercase thorn
°	`\[de]`	degree sign	ß	`\[ss]`	lowercase sharp s
±	`\[+-]`	plus-minus	à	`\[`a]`	a grave
²	`\[S2]`	superscript two	á	`\['a]`	a acute
³	`\[S3]`	superscript three	â	`\[^a]`	a circumflex
´	`\[aa]`	acute accent	ã	`\[~a]`	a tilde
µ	`\[mc]`	micro sign	ä	`\[:a]`	a dieresis
¶	`\[ps]`	pilcrow sign	å	`\[oa]`	a ring
·	`\[pc]`	centered period	æ	`\[ae]`	ae ligature
¸	`\[ac]`	cedilla accent	ç	`\[,c]`	c cedilla
¹	`\[S1]`	superscript one	è	`\[`e]`	e grave
º	`\[Om]`	masculine ordinal indicator	é	`\['e]`	e acute
»	`\[Fc]`	right double chevron	ê	`\[^e]`	e circumflex
¼	`\[14]`	one quarter symbol	ë	`\[:e]`	e dieresis

½	`\[12]`	one half symbol	ì	`\[`i]`	i grave
¾	`\[34]`	three quarters symbol	í	`\['i]`	e acute
¿	`\[r?]`	inverted question mark	î	`\[^i]`	i circumflex
À	`\[`A]`	A grave	ï	`\[:i]`	i dieresis
Á	`\['A]`	A acute	ð	`\[Sd]`	lowercase eth
Â	`\[^A]`	A circumflex	ñ	`\[~n]`	n tilde
Ã	`\[~A]`	A tilde	ò	`\[`o]`	o grave
Ä	`\[:A]`	A dieresis	ó	`\['o]`	o acute
Å	`\[oA]`	A ring	ô	`\[^o]`	o circumflex
Æ	`\[AE]`	AE ligature	õ	`\[~o]`	o tilde
Ç	`\[,C]`	C cedilla	ö	`\[:o]`	o dieresis
È	`\[`E]`	E grave	÷	`\[di]`	division sign
É	`\['E]`	E acute	ø	`\[/o]`	o slash
Ê	`\[^E]`	E circumflex	ù	`\[`u]`	u grave
Ë	`\[:E]`	E dieresis	ú	`\['u]`	u acute
Ì	`\[`I]`	I grave	û	`\[^u]`	u circumflex
Í	`\['I]`	I acute	ü	`\[:u]`	u dieresis
Î	`\[^I]`	I circumflex	ý	`\['y]`	y acute
Ï	`\[:I]`	I dieresis	þ	`\[Tp]`	lowercase thorn
Ð	`\[-D]`	uppercase eth	ÿ	`\[:y]`	y dieresis

Special character escape forms

Glyphs that lack a character code in the basic Latin repertoire to directly represent them are entered by one of several special character escape forms. Such glyphs can be simple or composite, and accessed either by name or numerically by code point. Code points and combining properties are determined by character encoding standards, whereas glyph names as used here originated in AT&T *troff* special character escape sequences. Predefined glyph names use only characters in the basic Latin repertoire.

\(*gl* is a special character escape sequence for the glyph with the two-character name *gl*. This is the original syntax form supported by AT&T *troff*. The acute accent, **\(aa**, is an example.

\C'*glyph-name***'**
is a special character escape sequence for *glyph-name*, which can be of arbitrary length. The delimiter, shown here as a neutral apostrophe, can be any character not occurring in *glyph-name*. This syntax form was introduced in later versions of AT&T device-independent *troff*. The foregoing acute accent example can be expressed as **\C'aa'**.

\[*glyph-name***]**
is a special character escape sequence for *glyph-name*, which can be of arbitrary length but must not contain a closing square bracket "**]**". (No glyph names predefined by *groff* employ "**]**".) The foregoing acute accent example can be expressed in *groff* as **\[aa]**.

\C'*c***'** and **\[***c***]** are not synonyms for the ordinary character "*c*", but request the special character named "*c*". For example, "**\[a]**" is not "a", but rather a special character with the internal glyph name (used in font description files and diagnostic messages) **\a**, which is typically undefined. The only such glyph name *groff* predefines is the minus sign, which can therefore be accessed as **\C'−'** or **\[−]**.

\[*base-char composite-1 composite-2 . . . composite-n***]**
is a composite glyph. Glyphs like a lowercase "e" with an acute accent, as in the word "café", can be expressed as **\[e aa]**. See subsection "Accents" below for a table of combining glyph names.

Unicode encodes far more characters than *groff* has glyph names for; special character escape forms based on numerical code points enable access to any of them. Frequently used glyphs or glyph combinations can be stored in strings, and new glyph names can be created *ad hoc* with the **char** request; see *groff*(7).

\[u*nnnn*[*n*[*n*]]**]**
is a Unicode numeric special character escape sequence. Any Unicode code point can be accessed with four to six hexadecimal digits, with hexadecimal letters accepted in uppercase form only. Thus, **\[u02DA]** accesses the (spacing) ring accent, producing "˚".

Unicode code points can be composed as well; when they are, GNU *troff* requires NFD (Normalization Form D), where all Unicode glyphs are maximally decomposed. (Exception: precomposed characters in the Latin-1 supplement described above are also accepted. Do not count on this exception remaining in a future GNU *troff* that accepts UTF-8 input directly.) Thus, GNU *troff* accepts "caf**\['e]**", "caf**\[e aa]**", and "caf**\[u0065_0301]**", as ways to input "café". (Due to its legacy 8-bit encoding compatibility, at present it also accepts "caf**\[u00E9]**" on ISO Latin-1 systems.)

\[u*base-char*[*_combining-component*]...]
constructs a composite glyph from Unicode numeric special character escape sequences. The code points of the base glyph and the combining components are each expressed in hexadecimal, with an underscore (_) separating each component. Thus, **\[u006E_0303]** produces "ñ".

\[char*nnn***]**
expresses an eight-bit code point where *nnn* is the code point of the character, a decimal number between 0 and 255 without leading zeroes. This legacy numeric special character escape sequence is used to map characters onto glyphs via the **trin** request in macro files loaded by *grotty*(1).

Glyph tables

In this section, *groff*'s glyph name repertoire is presented in tabular form. The meanings of the columns are as follows.

Output shows the glyph as it appears on the device used to render this document; although it can have a notably different shape on other devices (and is subject to user-directed translation and replacement), *groff* attempts reasonable equivalency on all output devices.

Input shows the *groff* character (ordinary or special) that normally produces the glyph. Some code points have multiple glyph names.

Unicode is the code point notation for the glyph or combining glyph sequence as described in subsection "Special character escape forms" above. It corresponds to the standard notation for Unicode short identifiers such that *groff*'s **u***nnnn* is equivalent to Unicode's U+*nnnn*.

Notes describes the glyph, elucidating the mnemonic value of the glyph name where possible.

A plus sign "+" indicates that the glyph name appears in the AT&T *troff* user's manual, CSTR #54 (1992 revision). When using the AT&T special character syntax **\(***xx*, widespread portability can be expected from such names.

Entries marked with "***" denote glyphs used for mathematical purposes. On typesetting devices, such glyphs are typically drawn from a *special* font (see *groff_font*(5)). Often, such glyphs lack bold or italic style forms or have metrics that look incongruous in ordinary prose. A few which are not uncommon in running text have "text variants", which should work better in that context. Conversely, a handful of glyphs that are normally drawn from a text font may be required in mathematical equations. Both sets of exceptions are noted in the tables where they appear ("Logical symbols" and "Mathematical symbols").

Basic Latin

Apart from basic Latin characters with special mappings, described in subsection "Fundamental character set" above, a few others in that range have special character glyph names. These were defined for ease of input on non-U.S. keyboards lacking keycaps for them, or for symmetry with other special character glyph names serving a similar purpose.

The vertical bar is overloaded; the **\[ba]** and **\[or]** escape sequences may render differently. See subsection "Mathematical symbols" below for special variants of the plus, minus, and equals signs normally drawn from this range.

Output	Input	Unicode	Notes
"	`\[dq]`	u0022	neutral double quote
#	`\[sh]`	u0023	number sign
$	`\[Do]`	u0024	dollar sign
'	`\[aq]`	u0027	apostrophe, neutral single quote

/	\[sl]	u002F	slash, solidus +
@	\[at]	u0040	at sign
[	\[lB]	u005B	left square bracket
\	\[rs]	u005C	reverse solidus
]	\[rB]	u005D	right square bracket
^	\[ha]	u005E	circumflex, caret, “hat”
{	\[lC]	u007B	left brace
|	\[bv]	u007C	bar
|	\[ba]	u007C	bar
|	\[or]	u007C	bitwise or +
}	\[rC]	u007D	right brace
~	\[ti]	u007E	tilde

Supplementary Latin letters

Historically, **\[ss]** could be considered a ligature of “sz”. An uppercase form is available as **\[u1E9E]**, but in the German language it is of specialized use; ß does *not* normally uppercase-transform to it, but rather to “SS”. “Lowercase f with hook” is also used as a function symbol; see subsection “Mathematical symbols” below.

Output	Input	Unicode	Notes
Ð	\[-D]	u00D0	uppercase eth
ð	\[Sd]	u00F0	lowercase eth
Þ	\[TP]	u00DE	uppercase thorn
þ	\[Tp]	u00FE	lowercase thorn
ß	\[ss]	u00DF	lowercase sharp s
ı	\[.i]	u0131	i without tittle
	\[.j]	u0237	j without tittle
f	\[Fn]	u0192	lowercase f with hook, function
Ł	\[/L]	u0141	L with stroke
ł	\[/l]	u0142	l with stroke
Ø	\[/O]	u00D8	O with stroke
ø	\[/o]	u00F8	o with stroke

Ligatures and digraphs

Output	Input	Unicode	Notes
ff	\[ff]	u0066_0066	ff ligature +
fi	\[fi]	u0066_0069	fi ligature +
fl	\[fl]	u0066_006C	fl ligature +
ffi	\[Fi]	u0066_0066_0069	ffi ligature +
ffl	\[Fl]	u0066_0066_006C	ffl ligature +
Æ	\[AE]	u00C6	AE ligature
æ	\[ae]	u00E6	ae ligature
Œ	\[OE]	u0152	OE ligature
œ	\[oe]	u0153	oe ligature
IJ	\[IJ]	u0132	IJ digraph
ij	\[ij]	u0133	ij digraph

Accents

Normally, the formatting of a special character advances the drawing position as an ordinary character does. *groff*’s **composite** request designates a special character as combining. The *composite.tmac* macro file, loaded automatically by the default *troffrc*, maps the following special characters to the combining characters shown below. The non-combining code point in parentheses is used when the special character occurs in isolation (compare “**caf\[e aa]**” and “**caf\[aa]e**”).

Output	Input	Unicode	Notes
˝	\[a"]	u030B (u02DD)	double acute accent
¯	\[a-]	u0304 (u00AF)	macron accent
˙	\[a.]	u0307 (u02D9)	dot accent
^	\[a^]	u0302 (u005E)	circumflex accent
´	\[aa]	u0301 (u00B4)	acute accent +
`	\[ga]	u0300 (u0060)	grave accent +
˘	\[ab]	u0306 (u02D8)	breve accent
¸	\[ac]	u0327 (u00B8)	cedilla accent
¨	\[ad]	u0308 (u00A8)	dieresis accent
ˇ	\[ah]	u030C (u02C7)	caron accent
°	\[ao]	u030A (u02DA)	ring accent
~	\[a~]	u0303 (u007E)	tilde accent
˛	\[ho]	u0328 (u02DB)	hook accent

Accented characters

All of these glyphs can be composed using combining glyph names as described in subsection "Special character escape forms" above; the names below are short aliases for convenience.

Output	Input	Unicode	Notes
Á	\['A]	u0041_0301	A acute
Ć	\['C]	u0043_0301	C acute
É	\['E]	u0045_0301	E acute
Í	\['I]	u0049_0301	I acute
Ó	\['O]	u004F_0301	O acute
Ú	\['U]	u0055_0301	U acute
Ý	\['Y]	u0059_0301	Y acute
á	\['a]	u0061_0301	a acute
ć	\['c]	u0063_0301	c acute
é	\['e]	u0065_0301	e acute
í	\['i]	u0069_0301	i acute
ó	\['o]	u006F_0301	o acute
ú	\['u]	u0075_0301	u acute
ý	\['y]	u0079_0301	y acute
Ä	\[:A]	u0041_0308	A dieresis
Ë	\[:E]	u0045_0308	E dieresis
Ï	\[:I]	u0049_0308	I dieresis
Ö	\[:O]	u004F_0308	O dieresis
Ü	\[:U]	u0055_0308	U dieresis
Ÿ	\[:Y]	u0059_0308	Y dieresis
ä	\[:a]	u0061_0308	a dieresis
ë	\[:e]	u0065_0308	e dieresis
ï	\[:i]	u0069_0308	i dieresis
ö	\[:o]	u006F_0308	o dieresis
ü	\[:u]	u0075_0308	u dieresis
ÿ	\[:y]	u0079_0308	y dieresis
Â	\[^A]	u0041_0302	A circumflex
Ê	\[^E]	u0045_0302	E circumflex
Î	\[^I]	u0049_0302	I circumflex
Ô	\[^O]	u004F_0302	O circumflex
Û	\[^U]	u0055_0302	U circumflex
â	\[^a]	u0061_0302	a circumflex
ê	\[^e]	u0065_0302	e circumflex

î	\[^i]	u0069_0302	i circumflex
ô	\[^o]	u006F_0302	o circumflex
û	\[^u]	u0075_0302	u circumflex
À	\[`A]	u0041_0300	A grave
È	\[`E]	u0045_0300	E grave
Ì	\[`I]	u0049_0300	I grave
Ò	\[`O]	u004F_0300	O grave
Ù	\[`U]	u0055_0300	U grave
à	\[`a]	u0061_0300	a grave
è	\[`e]	u0065_0300	e grave
ì	\[`i]	u0069_0300	i grave
ò	\[`o]	u006F_0300	o grave
ù	\[`u]	u0075_0300	u grave
Ã	\[~A]	u0041_0303	A tilde
Ñ	\[~N]	u004E_0303	N tilde
Õ	\[~O]	u004F_0303	O tilde
ã	\[~a]	u0061_0303	a tilde
ñ	\[~n]	u006E_0303	n tilde
õ	\[~o]	u006F_0303	o tilde
Š	\[vS]	u0053_030C	S caron
š	\[vs]	u0073_030C	s caron
Ž	\[vZ]	u005A_030C	Z caron
ž	\[vz]	u007A_030C	z caron
Ç	\[,C]	u0043_0327	C cedilla
ç	\[,c]	u0063_0327	c cedilla
Å	\[oA]	u0041_030A	A ring
å	\[oa]	u0061_030A	a ring

Quotation marks

The neutral double quote, often useful when documenting programming languages, is also available as a special character for convenient embedding in macro arguments; see subsection "Fundamental character set" above.

Output	Input	Unicode	Notes
„	\[Bq]	u201E	low double comma quote
‚	\[bq]	u201A	low single comma quote
“	\[lq]	u201C	left double quote
”	\[rq]	u201D	right double quote
‘	\[oq]	u2018	single opening (left) quote
’	\[cq]	u2019	single closing (right) quote
'	\[aq]	u0027	apostrophe, neutral single quote
"	\"	u0022	neutral double quote
"	\[dq]	u0022	neutral double quote
«	\[Fo]	u00AB	left double chevron
»	\[Fc]	u00BB	right double chevron
‹	\[fo]	u2039	left single chevron
›	\[fc]	u203A	right single chevron

Punctuation

The Unicode name for U+00B7 is "middle dot", which is unfortunately confusable with the *groff* mnemonic for the visually similar but semantically distinct multiplication dot; see subsection "Mathematical symbols" below.

Output	Input	Unicode	Notes
¡	\[r!]	u00A1	inverted exclamation mark
¿	\[r?]	u00BF	inverted question mark
·	\[pc]	u00B7	centered period
—	\[em]	u2014	em-dash +
–	\[en]	u2013	en-dash
-	\[hy]	u2010	hyphen +

Brackets

On typesetting devices, the bracket extensions are font-invariant glyphs; that is, they are rendered the same way regardless of font (with a drawing escape sequence). On terminals, they are *not* font-invariant; *groff* maps them rather arbitrarily to U+23AA ("curly bracket extension"). In AT&T *troff*, only one glyph was available to vertically extend brackets, braces, and parentheses: **\(bv**.

Not all devices supply bracket pieces that can be piled up with **\b** due to the restrictions of the escape's piling algorithm. A general solution to build brackets out of pieces is the following macro:

```
.\" Make a pile centered vertically 0.5em above the baseline.
.\" The first argument is placed at the top.
.\" The pile is returned in string 'pile'.
.eo
.de pile-make
.  nr pile-wd 0
.  nr pile-ht 0
.  ds pile-args
.
.  nr pile-# \n[.$]
.  while \n[pile-#] \{\
.    nr pile-wd (\n[pile-wd] >? \w'\$[\n[pile-#]]')
.    nr pile-ht +(\n[rst] - \n[rsb])
.    as pile-args \v'\n[rsb]u'\"
.    as pile-args \Z'\$[\n[pile-#]]'\"
.    as pile-args \v'-\n[rst]u'\"
.    nr pile-# -1
.  \}
.
.  ds pile \v'(-0.5m + (\n[pile-ht]u / 2u))'\"
.  as pile \*[pile-args]\"
.  as pile \v'((\n[pile-ht]u / 2u) + 0.5m)'\"
.  as pile \h'\n[pile-wd]u'\"
..
.ec
```

Another complication is the fact that some glyphs which represent bracket pieces in AT&T *troff* can be used for other mathematical symbols as well, for example **\(lf** and **\(rf**, which provide the floor operator. Some output devices, such as **dvi**, don't unify such glyphs. For this reason, the glyphs **\[lf]**, **\[rf]**, **\[lc]**, and **\[rc]** are not unified with similar-looking bracket pieces. In *groff*, only glyphs with long names are guaranteed to pile up correctly for all devices—provided those glyphs are available.

Output	Input	Unicode	Notes
[	[	u005B	left square bracket
[	\[lB]	u005B	left square bracket
]	]	u005D	right square bracket
]	\[rB]	u005D	right square bracket
{	{	u007B	left brace
{	\[lC]	u007B	left brace
}	}	u007D	right brace

}	\[rC]	u007D	right brace
⟨	\[la]	u27E8	left angle bracket
⟩	\[ra]	u27E9	right angle bracket
⎪	\[bv]	u23AA	brace vertical extension + ***
⎪	\[braceex]	u23AA	brace vertical extension
⎡	\[bracketlefttp]	u23A1	left square bracket top
⎢	\[bracketleftex]	u23A2	left square bracket extension
⎣	\[bracketleftbt]	u23A3	left square bracket bottom
⎤	\[bracketrighttp]	u23A4	right square bracket top
⎥	\[bracketrightex]	u23A5	right square bracket extension
⎦	\[bracketrightbt]	u23A6	right square bracket bottom
⎧	\[lt]	u23A7	left brace top +
⎨	\[lk]	u23A8	left brace middle +
⎩	\[lb]	u23A9	left brace bottom +
⎧	\[bracelefttp]	u23A7	left brace top
⎨	\[braceleftmid]	u23A8	left brace middle
⎩	\[braceleftbt]	u23A9	left brace bottom
⎪	\[braceleftex]	u23AA	left brace extension
⎫	\[rt]	u23AB	right brace top +
⎬	\[rk]	u23AC	right brace middle +
⎭	\[rb]	u23AD	right brace bottom +
⎫	\[bracerighttp]	u23AB	right brace top
⎬	\[bracerightmid]	u23AC	right brace middle
⎭	\[bracerightbt]	u23AD	right brace bottom
⎪	\[bracerightex]	u23AA	right brace extension
⎛	\[parenlefttp]	u239B	left parenthesis top
⎜	\[parenleftex]	u239C	left parenthesis extension
⎝	\[parenleftbt]	u239D	left parenthesis bottom
⎞	\[parenrighttp]	u239E	right parenthesis top
⎟	\[parenrightex]	u239F	right parenthesis extension
⎠	\[parenrightbt]	u23A0	right parenthesis bottom

Arrows

Output	Input	Unicode	Notes
←	\[<-]	u2190	horizontal arrow left +
→	\[->]	u2192	horizontal arrow right +
↔	\[<>]	u2194	bidirectional horizontal arrow
↓	\[da]	u2193	vertical arrow down +
↑	\[ua]	u2191	vertical arrow up +
↕	\[va]	u2195	bidirectional vertical arrow
⇐	\[lA]	u21D0	horizontal double arrow left
⇒	\[rA]	u21D2	horizontal double arrow right
⇔	\[hA]	u21D4	bidirectional horizontal double arrow
⇓	\[dA]	u21D3	vertical double arrow down
⇑	\[uA]	u21D1	vertical double arrow up
	\[vA]	u21D5	bidirectional vertical double arrow
—	\[an]	u23AF	horizontal arrow extension

Rules and lines

On typesetting devices, the font-invariant glyphs (see subsection "Brackets" above) **\[br]**, **\[ul]**, and **\[rn]** form corners when adjacent; they can be used to build boxes. On terminal devices, they are mapped as shown in the table. The Unicode-derived names of these three glyphs are approximations.

The input character **_** always accesses the underscore glyph in a font; **\[ul]**, by contrast, may be font-

invariant on typesetting devices.

The baseline rule **\[ru]** is a font-invariant glyph, namely a rule of one-half em.

In AT&T *troff*, **\[rn]** also served as a one en extension of the square root symbol. *groff* favors **\[radicalex]** for this purpose; see subsection "Mathematical symbols" below.

Output	Input	Unicode	Notes
|	|	u007C	bar
|	\[ba]	u007C	bar
⎸	\[br]	u2502	box rule +
_	_	u005F	underscore, low line +
=	\[ul]	---	underrule +
	\[rn]	u203E	overline +
_	\[ru]	---	baseline rule +
¦	\[bb]	u00A6	broken bar
/	/	u002F	slash, solidus +
/	\[sl]	u002F	slash, solidus +
\	\[rs]	u005C	reverse solidus

Text markers

Output	Input	Unicode	Notes
○	\[ci]	u25CB	circle +
•	\[bu]	u2022	bullet +
†	\[dg]	u2020	dagger +
‡	\[dd]	u2021	double dagger +
◊	\[lz]	u25CA	lozenge, diamond
□	\[sq]	u25A1	square +
¶	\[ps]	u00B6	pilcrow sign
§	\[sc]	u00A7	section sign +
☜	\[lh]	u261C	hand pointing left +
☞	\[rh]	u261E	hand pointing right +
@	@	u0040	at sign
@	\[at]	u0040	at sign
#	#	u0023	number sign
#	\[sh]	u0023	number sign
↵	\[CR]	u21B5	carriage return
✓	\[OK]	u2713	check mark

Legal symbols

The Bell System logo is not supported in *groff*.

Output	Input	Unicode	Notes
©	\[co]	u00A9	copyright sign +
®	\[rg]	u00AE	registered sign +
™	\[tm]	u2122	trade mark sign
	\[bs]	---	Bell System logo +

Currency symbols

Output	Input	Unicode	Notes
$	$	u0024	dollar sign
$	\[Do]	u0024	dollar sign
¢	\[ct]	u00A2	cent sign +
€	\[eu]	u20AC	Euro sign
€	\[Eu]	u20AC	variant Euro sign
¥	\[Ye]	u00A5	yen sign
£	\[Po]	u00A3	pound sign

¤	\[Cs]	u00A4	currency sign

Units

Output	Input	Unicode	Notes
°	\[de]	u00B0	degree sign +
‰	\[%0]	u2030	per thousand, per mille sign
′	\[fm]	u2032	arc minute sign, foot mark +
″	\[sd]	u2033	arc second sign
µ	\[mc]	u00B5	micro sign
ª	\[Of]	u00AA	feminine ordinal indicator
º	\[Om]	u00BA	masculine ordinal indicator

Logical symbols

The variants of the not sign may differ in appearance or spacing depending on the device and font selected. Unicode does not encode a discrete "bitwise or" sign: on typesetting devices, it is drawn shorter than the bar, about the same height as a capital letter. Terminal devices unify **\[ba]** and **\[or]**.

Output	Input	Unicode	Notes
∧	\[AN]	u2227	logical and
∨	\[OR]	u2228	logical or
¬	\[no]	u00AC	logical not + ***
¬	\[tno]	u00AC	text variant of **\[no]**
∃	\[te]	u2203	there exists
∀	\[fa]	u2200	for all
∋	\[st]	u220B	such that
∴	\[3d]	u2234	therefore
∴	\[tf]	u2234	therefore
|	|	u007C	bar
|	\[or]	u007C	bitwise or +

Mathematical symbols

\[Fn] also appears in subsection "Supplementary Latin letters" above. Observe the two varieties of the plus-minus, multiplication, and division signs; **\[+−]**, **\[mu]**, and **\[di]** are normally drawn from the special font, but have text font variants. Also be aware of three glyphs available in special font variants that are normally drawn from text fonts: the plus, minus, and equals signs. These variants may differ in appearance or spacing depending on the device and font selected.

In AT&T *troff*, **\(rn** ("root en extender") served as the horizontal extension of the radical (square root) sign, **\(sr**, and was drawn at the maximum height of the typeface's bounding box; this enabled the special character to double as an overline (see subsection "Rules and lines" above). A contemporary font's radical sign might not ascend to such an extreme. In *groff*, you can instead use **\[radicalex]** to continue the radical sign **\[sr]**; these special characters are intended for use with text fonts. **\[sqrt]** and **\[sqrtex]** are their counterparts with mathematical spacing.

Output	Input	Unicode	Notes
½	\[12]	u00BD	one half symbol +
¼	\[14]	u00BC	one quarter symbol +
¾	\[34]	u00BE	three quarters symbol +
⅛	\[18]	u215B	one eighth symbol
⅜	\[38]	u215C	three eighths symbol
⅝	\[58]	u215D	five eighths symbol
⅞	\[78]	u215E	seven eighths symbol
¹	\[S1]	u00B9	superscript one
²	\[S2]	u00B2	superscript two
³	\[S3]	u00B3	superscript three
+	+	u002B	plus

+	\[pl]	u002B	special variant of plus + ***
−	\[-]	u002D	minus
−	\[mi]	u2212	special variant of minus + ***
	\[-+]	u2213	minus-plus
±	\[+-]	u00B1	plus-minus + ***
±	\[t+-]	u00B1	text variant of **\[+-]**
·	\[md]	u22C5	multiplication dot
×	\[mu]	u00D7	multiplication sign + ***
×	\[tmu]	u00D7	text variant of **\[mu]**
⊗	\[c*]	u2297	circled times
⊕	\[c+]	u2295	circled plus
÷	\[di]	u00F7	division sign + ***
÷	\[tdi]	u00F7	text variant of **\[di]**
⁄	\[f/]	u2044	fraction slash
*	*	u002A	asterisk
∗	\[**]	u2217	mathematical asterisk +
≤	\[<=]	u2264	less than or equal to +
≥	\[>=]	u2265	greater than or equal to +
≪	\[<<]	u226A	much less than
≫	\[>>]	u226B	much greater than
=	=	u003D	equals
=	\[eq]	u003D	special variant of equals + ***
≠	\[!=]	u003D_0338	not equals +
≡	\[==]	u2261	equivalent +
≢	\[ne]	u2261_0338	not equivalent
≅	\[=~]	u2245	approximately equal to
≃	\[|=]	u2243	asymptotically equal to +
~	\[ti]	u007E	tilde +
~	\[ap]	u223C	similar to, tilde operator +
≈	\[~~]	u2248	almost equal to
≈	\[~=]	u2248	almost equal to
∝	\[pt]	u221D	proportional to +
∅	\[es]	u2205	empty set +
∈	\[mo]	u2208	element of a set +
∉	\[nm]	u2208_0338	not element of set
⊂	\[sb]	u2282	proper subset +
⊄	\[nb]	u2282_0338	not subset
⊃	\[sp]	u2283	proper superset +
⊅	\[nc]	u2283_0338	not superset
⊆	\[ib]	u2286	subset or equal +
⊇	\[ip]	u2287	superset or equal +
∩	\[ca]	u2229	intersection, cap +
∪	\[cu]	u222A	union, cup +
∠	\[/_]	u2220	angle
⊥	\[pp]	u22A5	perpendicular
∫	\[is]	u222B	integral +
∫	\[integral]	u222B	integral ***
∑	\[sum]	u2211	summation ***
∏	\[product]	u220F	product ***
	\[coproduct]	u2210	coproduct ***
∇	\[gr]	u2207	gradient +
√	\[sr]	u221A	radical sign, square root +

‾	`\[rn]`	u203E	overline +
‾	`\[radicalex]`	---	radical extension
√	`\[sqrt]`	u221A	radical sign, square root ***
	`\[sqrtex]`	---	radical extension ***
⌈	`\[lc]`	u2308	left ceiling +
⌉	`\[rc]`	u2309	right ceiling +
⌊	`\[lf]`	u230A	left floor +
⌋	`\[rf]`	u230B	right floor +
∞	`\[if]`	u221E	infinity +
ℵ	`\[Ah]`	u2135	aleph symbol
f	`\[Fn]`	u0192	lowercase f with hook, function
ℑ	`\[Im]`	u2111	blackletter I, imaginary part
ℜ	`\[Re]`	u211C	blackletter R, real part
℘	`\[wp]`	u2118	Weierstrass p
∂	`\[pd]`	u2202	partial differential
ℏ	`\[-h]`	u210F	h bar
ℏ	`\[hbar]`	u210F	h bar

Greek glyphs

These glyphs are intended for technical use, not for typesetting Greek language text; normally, the uppercase letters have upright shape, and the lowercase ones are slanted.

Output	Input	Unicode	Notes
Α	`\[*A]`	u0391	uppercase alpha +
Β	`\[*B]`	u0392	uppercase beta +
Γ	`\[*G]`	u0393	uppercase gamma +
Δ	`\[*D]`	u0394	uppercase delta +
Ε	`\[*E]`	u0395	uppercase epsilon +
Ζ	`\[*Z]`	u0396	uppercase zeta +
Η	`\[*Y]`	u0397	uppercase eta +
Θ	`\[*H]`	u0398	uppercase theta +
Ι	`\[*I]`	u0399	uppercase iota +
Κ	`\[*K]`	u039A	uppercase kappa +
Λ	`\[*L]`	u039B	uppercase lambda +
Μ	`\[*M]`	u039C	uppercase mu +
Ν	`\[*N]`	u039D	uppercase nu +
Ξ	`\[*C]`	u039E	uppercase xi +
Ο	`\[*O]`	u039F	uppercase omicron +
Π	`\[*P]`	u03A0	uppercase pi +
Ρ	`\[*R]`	u03A1	uppercase rho +
Σ	`\[*S]`	u03A3	uppercase sigma +
Τ	`\[*T]`	u03A4	uppercase tau +
Υ	`\[*U]`	u03A5	uppercase upsilon +
Φ	`\[*F]`	u03A6	uppercase phi +
Χ	`\[*X]`	u03A7	uppercase chi +
Ψ	`\[*Q]`	u03A8	uppercase psi +
Ω	`\[*W]`	u03A9	uppercase omega +
α	`\[*a]`	u03B1	lowercase alpha +
β	`\[*b]`	u03B2	lowercase beta +
γ	`\[*g]`	u03B3	lowercase gamma +
δ	`\[*d]`	u03B4	lowercase delta +
ε	`\[*e]`	u03B5	lowercase epsilon +
ζ	`\[*z]`	u03B6	lowercase zeta +

η	\[*y]	u03B7	lowercase eta +
θ	\[*h]	u03B8	lowercase theta +
ι	\[*i]	u03B9	lowercase iota +
κ	\[*k]	u03BA	lowercase kappa +
λ	\[*l]	u03BB	lowercase lambda +
μ	\[*m]	u03BC	lowercase mu +
ν	\[*n]	u03BD	lowercase nu +
ξ	\[*c]	u03BE	lowercase xi +
ο	\[*o]	u03BF	lowercase omicron +
π	\[*p]	u03C0	lowercase pi +
ρ	\[*r]	u03C1	lowercase rho +
σ	\[*s]	u03C3	lowercase sigma +
τ	\[*t]	u03C4	lowercase tau +
υ	\[*u]	u03C5	lowercase upsilon +
φ	\[*f]	u03D5	lowercase phi +
χ	\[*x]	u03C7	lowercase chi +
ψ	\[*q]	u03C8	lowercase psi +
ω	\[*w]	u03C9	lowercase omega +
	\[+e]	u03F5	variant epsilon (lunate)
ϑ	\[+h]	u03D1	variant theta (cursive form)
ϖ	\[+p]	u03D6	variant pi (similar to omega)
ϕ	\[+f]	u03C6	variant phi (curly shape)
ς	\[ts]	u03C2	terminal lowercase sigma +

Playing card symbols

Output	Input	Unicode	Notes
♣	\[CL]	u2663	solid club suit
♠	\[SP]	u2660	solid spade suit
♥	\[HE]	u2665	solid heart suit
♦	\[DI]	u2666	solid diamond suit

History

A consideration of the typefaces originally available to AT&T *nroff* and *troff* illuminates many conventions that one might regard as idiosyncratic fifty years afterward. (See section "History" of *roff*(7) for more context.) The face used by the Teletype Model 37 terminals of the Murray Hill Unix Room was based on ASCII, but assigned multiple meanings to several code points, as suggested by that standard. Decimal 34 (") served as a dieresis accent and neutral double quotation mark; decimal 39 (') as an acute accent, apostrophe, and closing (right) single quotation mark; decimal 45 (-) as a hyphen and a minus sign; decimal 94 (^) as a circumflex accent and caret; decimal 96 (`) as a grave accent and opening (left) single quotation mark; and decimal 126 (~) as a tilde accent and (with a half-line motion) swung dash. The Model 37 bore an optional extended character set offering upright Greek letters and several mathematical symbols; these were documented as early as the *kbd*(VII) man page of the (First Edition) *Unix Programmer's Manual.*

At the time Graphic Systems delivered the C/A/T phototypesetter to AT&T, the ASCII character set was not considered a standard basis for a glyph repertoire by traditional typographers. In the stock Times roman, italic, and bold styles available, several ASCII characters were not present at all, nor was most of the Teletype's extended character set. AT&T commissioned a "special" font to ensure no loss of repertoire.

A representation of the coverage of the C/A/T's text fonts follows. The glyph resembling an underscore is a baseline rule, and that resembling a vertical line is a box rule. In italics, the box rule was not slanted. We also observe that the hyphen and minus sign were already "de-unified" by the fonts provided; a decision whither to map an input "–" therefore had to be taken.

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z
0 1 2 3 4 5 6 7 8 9 fi fl ffi ffl
! $ % & ( ) ‘ ’ * + – . , / : ; = ? [ ] |
• □ — - _ ¼ ½ ¾ ° † ′ ¢ ® ©

The special font supplied the missing ASCII and Teletype extended glyphs, among several others. The plus, minus, and equals signs appeared in the special font despite availability in text fonts “to insulate the appearance of equations from the choice of standard [read: text] fonts”—a priority since *troff* was turned to the task of mathematical typesetting as soon as it was developed.

We note that AT&T took the opportunity to de-unify the apostrophe/right single quotation mark from the acute accent (a choice ISO later duplicated in its 8859 series of standards). A slash intended to be mirror-symmetric with the backslash was also included, as was the Bell System logo; we do not attempt to depict the latter.

α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π ρ σ ς τ υ φ χ ψ ω
Γ Δ Θ Λ Ξ Π Σ Υ Φ Ψ Ω
" ´ \ ^ _ ` ~ / < > { } # @ + – = *
≥ ≤ ≡ ≈ ~ ≠ ↑ ↓ ← → × ÷ ± ∞ ∂ ∇ ¬ ∫ ∝ √ ¯ ∪ ∩ ⊂ ⊃ ⊆ ⊇ ∅ ∈
§ ‡ ☜ ☞ | ○ ⌈ ⌊ ⌉ ⌋ ⌈ ⌉ | ⌊ ⌋ ⌈ ⌉

One ASCII character as rendered by the Model 37 was apparently abandoned. That device printed decimal 124 (|) as a broken vertical line, like Unicode U+00A6 (¦). No equivalent was available on the C/A/T; the box rule **\[br]**, brace vertical extension **\[bv]**, and “or” operator **\[or]** were used as contextually appropriate.

Devices supported by AT&T device-independent *troff* exhibited some differences in glyph detail. For example, on the Autologic APS-5 phototypesetter, the square **\(sq** became filled in the Times bold face.

Files

The files below are loaded automatically by the default *troffrc*.

/usr/share/groff/1.23.0/tmac/composite.tmac
: assigns alternate mappings for identifiers after the first in a composite special character escape sequence. See subsection “Accents” above.

/usr/share/groff/1.23.0/tmac/fallbacks.tmac
: defines fallback mappings for Unicode code points such as the increment sign (U+2206) and upper- and lowercase Roman numerals.

Authors

This document was written by James Clark, with additions by Werner Lemberg and Bernd Warken, revised to use *tbl*(1) by Eric SRaymond, and largely rewritten by GBranden Robinson.

See also

Groff: The GNU Implementation of troff, by Trent A. Fisher and Werner Lemberg, is the primary *groff* manual. Section “Using Symbols” may be of particular note. You can browse it interactively with “info '(groff) Using Symbols'”.

“An extension to the *troff* character set for Europe”, E.G. Keizer, K.J. Simonsen, J. Akkerhuis; EUUG Newsletter, Volume 9, No. 2, Summer 1989

The Unicode Standard

“7-bit Character Sets” by Tuomas Salste documents the inherent ambiguity and configurable code points of the ASCII encoding standard.

“Nroff/Troff User’s Manual” by Joseph F. Ossanna, 1976, AT&T Bell Laboratories Computing Science Technical Report No. 54, features two tables that throw light on the glyph repertoire available to “typesetter *roff*” when it was first written. Be careful of re-typeset versions of this document that can be found on the Internet. Some do not accurately represent the original document: several glyphs are obviously missing. More subtly, lowercase Greek letters are rendered upright, not slanted as they appeared in the C/A/T’s special font and as expected by *troff* users.

groff_rfc1345(7) describes an alternative set of special character glyph names, which extends and in some cases overrides the definitions listed above.

groff (1), *troff* (1), *groff* (7)