

NAME

mandoc_char - mandoc special characters

DESCRIPTION

This page documents the special characters and predefined strings accepted by mandoc(1) to format mdoc(7) and man(7) documents.

Both mdoc(7) and man(7) encode special characters with ‘\X’ (for a one-character escape), ‘\XX’ (two-character), and ‘\N’ (N-character). One may generalise ‘\XX’ as ‘\XX’ and ‘\X’ as ‘\X’. Predefined strings are functionally similar to special characters, using ‘*X’ (for a one-character escape), ‘*(XX’ (two-character), and ‘*[N]’ (N-character). One may generalise ‘*(XX’ as ‘*[XX]’ and ‘*X’ as ‘*[X]’.

Note that each output mode will have a different rendering of the characters. It’s guaranteed that each input symbol will correspond to a (more or less) meaningful output rendering, regardless the mode.

SPECIAL CHARACTERS

These are the preferred input symbols for producing special characters.

Spacing:

<i>Input</i>	<i>Description</i>
~	non-breaking, non-collapsing space
\	breaking, non-collapsing n-width space
^	zero-width space
%	zero-width space
&	zero-width space
	zero-width space
0	breaking, non-collapsing digit-width space
c	removes any trailing space (if applicable)

Lines:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
\(ba		bar
\(br		box rule
\(ul	_	underscore
\(rl	-	overline
\(bb		broken bar
\(sl	/	forward slash
\(rs	\	backward slash

Text markers:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(ci</code>	o	circle
<code>\(bu</code>	o	bullet
<code>\(dd</code>	=	double dagger
<code>\(dg</code>	-	dagger
<code>\(lz</code>	<>	lozenge
<code>\(sq</code>	[]	white square
<code>\(ps</code>	¶	paragraph
<code>\(sc</code>	§	section
<code>\(lh</code>	<=	left hand
<code>\(rh</code>	=>	right hand
<code>\(at</code>	@	at
<code>\(sh</code>	#	hash (pound)
<code>\(CR</code>	␣	carriage return
<code>\(OK</code>	✓	check mark

Legal symbols:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(co</code>	(C)	copyright
<code>\(rg</code>	(R)	registered
<code>\(tm</code>	™	trademarked

Punctuation:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(em</code>	--	em-dash
<code>\(en</code>	-	en-dash
<code>\(hy</code>	-	hyphen
<code>\e</code>	\	back-slash
<code>\.</code>	.	period
<code>\(r!</code>	¡	upside-down exclamation
<code>\(r?</code>	¿	upside-down question

Quotes:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(Bq</code>	”	right low double-quote
<code>\(bq</code>	,	right low single-quote
<code>\(lq</code>	“	left double-quote
<code>\(rq</code>	”	right double-quote
<code>\(oq</code>	‘	left single-quote

<code>\(cq</code>	'	right single-quote
<code>\(aq</code>	'	apostrophe quote (text)
<code>\(dq</code>	"	double quote (text)
<code>\(Fo</code>	<<	left guillemet
<code>\(Fc</code>	>>	right guillemet
<code>\(fo</code>	<	left single guillemet
<code>\(fc</code>	>	right single guillemet

Brackets:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(lB</code>	[left bracket
<code>\(rB</code>]	right bracket
<code>\(lC</code>	{	left brace
<code>\(rC</code>	}	right brace
<code>\(l\></code>	<	left angle
<code>\(r\></code>	>	right angle
<code>\(bv</code>		brace extension
<code>\[braceex]</code>		brace extension
<code>\[bracketlefttp]</code>		top-left hooked bracket
<code>\[bracketleftbp]</code>		bottom-left hooked bracket
<code>\[bracketlefttex]</code>		left hooked bracket extension
<code>\[bracketrighttp]</code>		top-right hooked bracket
<code>\[bracketrightbp]</code>		bottom-right hooked bracket
<code>\[bracketrighttex]</code>		right hooked bracket extension
<code>\(lt</code>	,-	top-left hooked brace
<code>\[bracelefttp]</code>	,-	top-left hooked brace
<code>\(lk</code>	{	mid-left hooked brace
<code>\[braceleftmid]</code>	{	mid-left hooked brace
<code>\(lb</code>	,-	bottom-left hooked brace
<code>\[braceleftbp]</code>	'-	bottom-left hooked brace
<code>\[bracelefttex]</code>		left hooked brace extension
<code>\(rt</code>	-.	top-right hooked brace
<code>\[bracerighttp]</code>	-.	top-right hooked brace
<code>\(rk</code>	}	mid-right hooked brace
<code>\[bracerightmid]</code>	}	mid-right hooked brace
<code>\(rb</code>	-'	bottom-right hooked brace
<code>\[bracerightbp]</code>	-'	bottom-right hooked brace
<code>\[bracerighttex]</code>		right hooked brace extension
<code>\(parenlefttp]</code>	/	top-left hooked parenthesis
<code>\(parenleftbp]</code>	\	bottom-left hooked parenthesis

<code>\[parenleftex]</code>		left hooked parenthesis extension
<code>\[parenrighttp]</code>	\	top-right hooked parenthesis
<code>\[parenrightbp]</code>	/	bottom-right hooked parenthesis
<code>\[parenrightex]</code>		right hooked parenthesis extension

Arrows:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(<-</code>	<-	left arrow
<code>\(>-</code>	->	right arrow
<code>\(<></code>	<>	left-right arrow
<code>\(da</code>	v	down arrow
<code>\(ua</code>	^	up arrow
<code>\(va</code>	^v	up-down arrow
<code>\(lA</code>	<=	left double-arrow
<code>\(rA</code>	=>	right double-arrow
<code>\(hA</code>	<=>	left-right double-arrow
<code>\(uA</code>	^	up double-arrow
<code>\(dA</code>	v	down double-arrow
<code>\(vA</code>	^=v	up-down double-arrow

Logical:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(AN</code>	^	logical and
<code>\(OR</code>	v	logical or
<code>\(no</code>	~	logical not
<code>\([tno]</code>	~	logical not (text)
<code>\(te</code>	∃	existential quantifier
<code>\(fa</code>	∀	universal quantifier
<code>\(st</code>	-)	such that
<code>\(tf</code>	∴	therefore
<code>\(3d</code>	∴	therefore
<code>\(or</code>		bitwise or

Mathematical:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(pl</code>	+	plus
<code>\(mi</code>	-	minus
<code>\(-</code>	-	minus (text)
<code>\(-+</code>	-+	minus-plus
<code>\(+-</code>	+-	plus-minus

<code>\[t+-]</code>	<code>+-</code>	plus-minus (text)
<code>\(pc</code>	<code>.</code>	centre-dot
<code>\(mu</code>	<code>x</code>	multiply
<code>\[tmu]</code>	<code>x</code>	multiply (text)
<code>\(c*</code>	<code>x</code>	circle-multiply
<code>\(c+</code>	<code>+</code>	circle-plus
<code>\(di</code>	<code>:-:</code>	divide
<code>\[tdi]</code>	<code>:-:</code>	divide (text)
<code>\(f/</code>	<code>/</code>	fraction
<code>\(***</code>	<code>*</code>	asterisk
<code>\(<=</code>	<code><=</code>	less-than-equal
<code>\(>=</code>	<code>>=</code>	greater-than-equal
<code>\(<<</code>	<code><<</code>	much less
<code>\(>></code>	<code>>></code>	much greater
<code>\(eq</code>	<code>=</code>	equal
<code>\(!=</code>	<code>!=</code>	not equal
<code>\(==</code>	<code>==</code>	equivalent
<code>\(ne</code>	<code>!==</code>	not equivalent
<code>\(=~</code>	<code>=~</code>	congruent
<code>\(~~</code>	<code>~~</code>	asymptotically congruent
<code>\(ap</code>	<code>~</code>	asymptotically similar
<code>\(~~</code>	<code>~~</code>	approximately similar
<code>\(~=</code>	<code>~=</code>	approximately equal
<code>\(pt</code>	<code>oc</code>	proportionate
<code>\(es</code>	<code>{}</code>	empty set
<code>\(mo</code>	<code>E</code>	element
<code>\(nm</code>	<code>!E</code>	not element
<code>\(sb</code>	<code>(=</code>	proper subset
<code>\(nb</code>	<code>(!=</code>	not subset
<code>\(sp</code>	<code>=)</code>	proper superset
<code>\(nc</code>	<code>!)=</code>	not superset
<code>\(ib</code>	<code>(=</code>	reflexive subset
<code>\(ip</code>	<code>=)</code>	reflexive superset
<code>\(ca</code>	<code>(^)</code>	intersection
<code>\(cu</code>	<code>U</code>	union
<code>\(/_</code>	<code>/_</code>	angle
<code>\(pp</code>	<code>_ _</code>	perpendicular
<code>\(is</code>	<code>I</code>	integral
<code>\[integral]</code>	<code>I</code>	integral
<code>\[sum]</code>	<code>E</code>	summation

<code>\[product]</code>	TT	product
<code>\[coproduct]</code>	U	coproduct
<code>\(gr</code>	V	gradient
<code>\(sr</code>	√	square root
<code>\[sqrt]</code>	√	square root
<code>\(lc</code>	~	left-ceiling
<code>\(rc</code>	~	right-ceiling
<code>\(lf</code>	_	left-floor
<code>\(rf</code>	_	right-floor
<code>\(if</code>	oo	infinity
<code>\(Ah</code>	N	aleph
<code>\(Im</code>	I	imaginary
<code>\(Re</code>	R	real
<code>\(pd</code>	a	partial differential
<code>\(-h</code>	/h	Planck constant over 2n

Ligatures:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(ff</code>	ff	ff ligature
<code>\(fi</code>	fi	fi ligature
<code>\(fl</code>	fl	fl ligature
<code>\(Fi</code>	ffi	ffi ligature
<code>\(Fl</code>	ffl	ffl ligature
<code>\(AE</code>	AE	AE
<code>\(ae</code>	ae	ae
<code>\(OE</code>	OE	OE
<code>\(oe</code>	oe	oe
<code>\(ss</code>	ss	German eszett
<code>\(IJ</code>	IJ	IJ ligature
<code>\(ij</code>	ij	ij ligature

Accents:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(a"</code>	"	Hungarian umlaut
<code>\(a-</code>	-	macron
<code>\(a.</code>	.	dotted
<code>\(a^</code>	^	circumflex
<code>\(aa</code>	'	acute
<code>\(a'</code>	'	acute
<code>\(ga</code>	`	grave

\`	´	grave
\(ab	´	breve
\(ac	,	cedilla
\(ad	"	dieresis
\(ah	v	caron
\(ao	o	ring
\(a~	~	tilde
\(ho	,	ogonek
\(ha	^	hat (text)
\(ti	~	tilde (text)

Accented letters:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
\('A	A	acute A
\('E	E	acute E
\('I	I	acute I
\('O	O	acute O
\('U	U	acute U
\('a	a	acute a
\('e	e	acute e
\('i	i	acute i
\('o	o	acute o
\('u	u	acute u
\(´A	A	grave A
\(´E	E	grave E
\(´I	I	grave I
\(´O	O	grave O
\(´U	U	grave U
\(´a	a	grave a
\(´e	e	grave e
\(´i	i	grave i
\(´o	i	grave o
\(´u	u	grave u
\(~A	A	tilde A
\(~N	N	tilde N
\(~O	O	tilde O
\(~a	a	tilde a
\(~n	n	tilde n
\(~o	o	tilde o
\(:A	A	dieresis A

\(:E	E	dieresis E
\(:I	I	dieresis I
\(:O	O	dieresis O
\(:U	U	dieresis U
\(:a	a	dieresis a
\(:e	e	dieresis e
\(:i	i	dieresis i
\(:o	o	dieresis o
\(:u	u	dieresis u
\(:y	y	dieresis y
\(^A	A	circumflex A
\(^E	E	circumflex E
\(^I	I	circumflex I
\(^O	O	circumflex O
\(^U	U	circumflex U
\(^a	a	circumflex a
\(^e	e	circumflex e
\(^i	i	circumflex i
\(^o	o	circumflex o
\(^u	u	circumflex u
\(,C	C	cedilla C
\(,c	c	cedilla c
\(/L	L	stroke L
\(/l	l	stroke l
\(/O	O	stroke O
\(/o	o	stroke o
\(oA	A	ring A
\(oa	a	ring a

Special letters:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
\(-D	D	Eth
\(Sd	o	eth
\(TP	b	Thorn
\(Tp	b	thorn
\(.i	i	dotless i
\(.j	j	dotless j

Currency:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
--------------	-----------------	--------------------

<code>\(Do</code>	\$	dollar
<code>\(ct</code>	c	cent
<code>\(Eu</code>	EUR	Euro symbol
<code>\(eu</code>	EUR	Euro symbol
<code>\(Ye</code>	Y	yen
<code>\(Po</code>	L	pound
<code>\(Cs</code>	x	Scandinavian
<code>\(Fn</code>	f	florin

Units:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(de</code>	o	degree
<code>\(%0</code>	%o	per-thousand
<code>\(fm</code>	'	minute
<code>\(sd</code>	"	second
<code>\(mc</code>	mu	micro

Greek letters:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>\(*A</code>	A	Alpha
<code>\(*B</code>	B	Beta
<code>\(*G</code>		Gamma
<code>\(*D</code>	^	Delta
<code>\(*E</code>	E	Epsilon
<code>\(*Z</code>	Z	Zeta
<code>\(*Y</code>	H	Eta
<code>\(*H</code>	O	Theta
<code>\(*I</code>	I	Iota
<code>\(*K</code>	K	Kappa
<code>\(*L</code>	^	Lambda
<code>\(*M</code>	M	Mu
<code>\(*N</code>	N	Nu
<code>\(*C</code>	H	Xi
<code>\(*O</code>	O	Omicron
<code>\(*P</code>	TT	Pi
<code>\(*R</code>	P	Rho
<code>\(*S</code>	>	Sigma
<code>\(*T</code>	T	Tau
<code>\(*U</code>	Y	Upsilon
<code>\(*F</code>	O_	Phi

<code>\(*X</code>	X	Chi
<code>\(*Q</code>	Y	Psi
<code>\(*W</code>	O	Omega
<code>\(*a</code>	a	alpha
<code>\(*b</code>	B	beta
<code>\(*g</code>	y	gamma
<code>\(*d</code>	d	delta
<code>\(*e</code>	e	epsilon
<code>\(*z</code>	C	zeta
<code>\(*y</code>	n	eta
<code>\(*h</code>	0	theta
<code>\(*i</code>	i	iota
<code>\(*k</code>	k	kappa
<code>\(*l</code>	\	lambda
<code>\(*m</code>	u	mu
<code>\(*n</code>	v	nu
<code>\(*c</code>	E	xi
<code>\(*o</code>	o	omicron
<code>\(*p</code>	n	pi
<code>\(*r</code>	p	rho
<code>\(*s</code>	o	sigma
<code>\(*t</code>	t	tau
<code>\(*u</code>	u	upsilon
<code>\(*f</code>	o	phi
<code>\(*x</code>	x	chi
<code>\(*q</code>	u	psi
<code>\(*w</code>	w	omega
<code>\(+h</code>	0	theta variant
<code>\(+f</code>	o	phi variant
<code>\(+p</code>	w	pi variant
<code>\(+e</code>	e	epsilon variant
<code>\(ts</code>	s	sigma terminal

PREDEFINED STRINGS

These are not recommended for use, as they differ across implementations:

<i>Input</i>	<i>Rendered</i>	<i>Description</i>
<code>*(Ba</code>		vertical bar
<code>*(Ne</code>	!=	not equal
<code>*(Ge</code>	>=	greater-than-equal

<code>*(Le</code>	<code><=</code>	less-than-equal
<code>*(Gt</code>	<code>></code>	greater-than
<code>*(Lt</code>	<code><</code>	less-than
<code>*(Pm</code>	<code>+-</code>	plus-minus
<code>*(If</code>	<code>infinity</code>	infinity
<code>*(Pi</code>	<code>pi</code>	pi
<code>*(Na</code>	<code>NaN</code>	NaN
<code>*(Am</code>	<code>&</code>	ampersand
<code>*R</code>	<code>(R)</code>	restricted mark
<code>*(Tm</code>	<code>tm</code>	trade mark
<code>*q</code>	<code>"</code>	double-quote
<code>*(Rq</code>	<code>”</code>	right-double-quote
<code>*(Lq</code>	<code>“</code>	left-double-quote
<code>*(lp</code>	<code>(</code>	right-parenthesis
<code>*(rp</code>	<code>)</code>	left-parenthesis
<code>*(lq</code>	<code>“</code>	left double-quote
<code>*(rq</code>	<code>”</code>	right double-quote
<code>*(ua</code>	<code>^</code>	up arrow
<code>*(va</code>	<code>^v</code>	up-down arrow
<code>*(<=</code>	<code><=</code>	less-than-equal
<code>*(>=</code>	<code>>=</code>	greater-than-equal
<code>*(aa</code>	<code>’</code>	acute
<code>*(ga</code>	<code>‘</code>	grave

COMPATIBILITY

This section documents compatibility of **mandoc_char** with older or existing versions of groff(1).

The following render differently in **-Tascii** output mode:

```
\(ss, \(\nm, \(\nb, \(\nc, \(\ib, \(\ip, \(\pp, \([sum], \([product], \([coproduct], \(\gr, \(-h, \(\a.
```

The following render differently in **-Thtml** output mode:

```
\(≈, \(\nb, \(\nc
```

Finally, the following have been omitted by being poorly documented or having no known representation:

```
\([radicalex], \([sqrtex], \(\ru
```

SEE ALSO

mandoc(1)

AUTHORS

The **mandoc_char** manual page was written by Kristaps Dzonsons <kristaps@bsd.lv>.

CAVEATS

The `*(Ba` escape mimics the behaviour of the `|` character in `mdoc(7)`; thus, if you wish to render a vertical bar with no side effects, use the `\(ba` escape.