名称

groff − GNU roff 言語の簡易リファレンス

解説

groff とは、*GNU roff* を表しており、roff 清書システムをフリーで実装したものです。groff システムの概観ならびに背景については **roff**(7) を参照してください。

このドキュメントでは、groff 内で使用される、あらかじめ定義された roff 言語要素を簡潔に説明しているだけです。古くからある機能と groff 拡張機能についての両方とも扱っています。

歴史的に、*roff 言語* は *troff* と呼ばれています。*groff* は、古くからあるシステムと互換性を持ちつつ、独自の拡張機能も 提供しています。そのため GNU 版では、*roff*, *troff*, *groff 言語* という術語は同義として使うこともできるでしょう。しかし、*troff* は、どちらかと言えば古くからの機能に対して適用されるという 傾向があり、それに対して、*groff* は GNU 拡張に重きを置いています。そして、*roff* は、この言語の一般的な術語になっています。

このファイルは、*groff* **info**(1) ファイルにある完全なドキュメントの簡易版に過ぎません。info ファイルの方がより詳細で実情に近く、正確な情報を含んでいます。

groff ドキュメントを書くために使う一般的な文法は比較的 やさしいのですが、roff 言語の拡張機能を書くのは少々骨が折れるかもしれません。

roff 言語は、行指向の言語です。行には、制御行とテキスト行の 2 種類しかありません。制御行は制御文字で始まります。制御文字は、デフォルトではピリオド“**.**”あるいはシングルクォート“**'**”です。そのほかの文字で始まる行はすべてテキスト行です。

制御行 は、コマンドを表し、オプションで引数を取ることもあります。 制御行は、次のような文法になります。 先頭の制御文字の後には、コマンド名を続けることができます。 引数は、もしあれば、コマンド名や他の引数と空白で分けられます。例えば、次のようにします。

```
.command_name arg1 arg2
```

インデント用に、先頭の制御文字とコマンド名との間にはスペースや　タブ文字をいくつ入れても良いですが、制御文字は行の先頭になくては なりません。

テキスト行 は、表示される部分のことです。テキスト行はエスケープシーケンスで変更することができます。エスケープシーケンスは先頭にバックスラッシュ‘****’を置くことで認識されます。これらは、行や単語の一部に含まれ、整形要素となったり 関数となったりします。エスケープシーケンスには、シングルクォート“**'**”で区切られた引数を取るものもありますし、開き括弧‘**(**’に続いて符合化され長さの一定なものや、角括弧‘**[**’と‘**]**’で括られているものもあります。

roff 言語は、マクロなど、言語拡張機能を書くための柔軟な手段を 提供しています。マクロ定義を解釈する際に、roff システムは **コピーモード** と呼ばれる特別なモードに入ります。

コピーモードの振る舞いはとても扱いにくいものでもありますが、 確実に安全に使えるようにするルールがいくつかあります。

1. 表示可能なバックスラッシュは **\e** と記述しなくてはなりません。 もっと正確に言えば、 **\e** は現在のエスケープ文字を表します。バックスラッシュのグリフを得るには **\(rs** を使用してください。
2. バックスラッシュはすべて 2 重にしてください。
3. テキスト行はすべて、スペースをとらない特殊文字 **\&** で始めてください。

このやり方は、一番効率の良いコードが生成できる　というわけではありませんが、最初の一歩としては使えるはずです。さらに良いやり方については groff info ファイルおよび **groff_tmac**(5) を参照してください。

roff ソースファイルを読むのはこれよりは簡単です。 すべてのマクロ定義部分で 2 重になっているバックスラッシュを 1 つに置き換えればよいだけです。

groff 要素

roff 言語の要素は、テキストファイルに整形用情報を 付加するものです。基本的な要素は、あらかじめ定義されたコマンド および変数であり、 この要素のおかげで roff は本格的なプログラミング言語に なっているのです。

roff コマンドには 2 種類あります。コマンドは引数を取ることもあります。**リクエスト** は、ドット‘**.**’あるいは“**'**”で始まる行に書かれるものです。これに対して **エスケープシーケンス** は、バックスラッシュ‘****’で始まる、行埋め込み式の関数や単語中の整形要素です。

ユーザは独自の整形コマンドを **.de** リクエストを使って定義することができます。これらのコマンドは **マクロ** と呼ばれていますが、実際にはリクエストとまったく同様に使われます。 マクロパッケージは groff 言語で書かれた定義済みのマクロセットです。ユーザが独自にエスケープシーケンスを作成できる場合というのは 非常に限られており、特殊文字のみマップすることができます。

groff 言語は、異なるインタフェースを持ついくつかの種類の変数を 提供しています。定義済みの変数もありますが、ユーザも同様に自分で変数を定義できます。

文字列 変数は文字列を格納します。この変数は **.ds** リクエストで設定され、値は ***** エスケープシーケンスを使って取り出します。

レジスタ 変数は数値、スケールファクタつき数字、 場合によっては文字列に似たオブジェクトを格納できます。**.nr** リクエストで設定され、値は **\n** エスケープシーケンスを使って取り出します。

環境 を使うことによって、行の長さやフォントサイズなどのような グローバルな整形パラメータを、後の再利用のために ユーザが一時的に格納することができます。これは **.ev** リクエストによって行われます。

フォント は、名前もしくは内部番号のどちらかで判別されます。現在のフォントは **.ft** リクエストもしくは **\f** エスケープシーケンスで選択されます。 デバイスごとに特別フォントがありますが、次のフォントはすべての デバイスで利用可能です。R は標準フォント、ローマン体です。B はその **ボールド体** です。 *イタリック体* フォントは I と呼ばれておりどこででも利用可能ですが、テキストデバイスではローマン体フォントに下線がついたものとして表示されます。 グラフィカルの出力デバイスに対しては、次のフォントの固定幅の 修飾文字が存在します。CR, CI, および CB です。テキストデバイスでは、いずれにしても文字はすべて固定幅です。

さらに、roff の拡張要素がいくつかあります。 **ディバージョン** は、後で利用できるように情報をマクロに格納します。**トラップ** は、ページの先頭から何行目とか、ディバージョンや入力において 何行目というような位置条件です。 アクションの中には、条件が揃ったときに自動的に起動するように 指示できるものがあります。

これより詳細な情報については、groff info ファイルに示されています。

制御文字

ある条件下で特別な制御タスクをもつ文字が いくつかあります。

. ドットは、行の先頭あるいは **.if**, **.ie**, **.el**, および **.while** リクエスト中の条件の後にある場合のみ特別です。その位置では、リクエスト (あるいはマクロ) を導入する制御文字になります。**\.** エスケープを使うと、この特別な動作を遅らせることができます。**.cc** リクエストを使うことで、この制御文字を別の文字に設定して、ドット‘.’を特殊文字でなくすることができます。

他の位置にドットがあるときは、ただのドット文字以上の意味はありません。 テキスト段落の中では、文はそれぞれ新しい行で始めるほうが有利です。

' シングルクォートには 2 つの制御機能があります。 行の先頭および条件リクエスト内では、シングルクォートは非ブレーク制御文字になります。これは、ドットのようにリクエストを導入しますが、リクエストが行を折り返さないような追加のプロパティをつけるものです。**.c2** リクエストを用いると、非ブレーク制御文字を別の文字に設定する ことができます。

2 つめのタスクとして、シングルクォートは、いくつかの関数エスケープシーケンスの中で 引数の区切り文字としてよく使用されます (引数に含まれない文字のペアならどれでも良いのですが)。 その他の場所にあれば、シングルクォート文字 あるいはアポストロフィ文字になります。groff は、表示用の表記として エスケープシーケンス **\(cq** を提供しています。

" ダブルクォートはリクエストおよびマクロ中の引数を括るときに使用されます。 **.ds** および **.as** リクエスト内では、引数の先頭にくるダブルクォートは取り除かれ、その後に続くダブルクォート以外の文字を定義文字列とします (先頭に空白が来るのを有効にします)。エスケープされたダブルクォート **\"** はコメントの開始になります。それ以外は、ダブルクォートには特別な機能はありません。groff は、表示用の表記として エスケープシーケンス **\(dq** を提供しています。

**** バックスラッシュは、通常エスケープシーケンスの開始を意味しています (この文字は、**ec** リクエストによって変更できます)。 表示用のエスケープ文字は、エスケープシーケンス **\e** で、バックスラッシュのグリフは **\(rs** で得られます。

(開き括弧は、エスケープシーケンス内で、ちょうど 2 文字でできた エスケープ名あるいは引数を導入したときのみ特別な意味を持ちます。groff では、この動作は **[]** の組で置き換えることができます。

[開き角括弧は、groff のエスケープシーケンス内でのみ特別な意味を持ちます。 これは、長いエスケープシーケンス名やエスケープシーケンスの引数を 導入するときに使用します。それ以外の場合は、例えばマクロ呼び出しなどでも特別な意味はありません。

] 閉じ角括弧は、groff のエスケープシーケンス内でのみ特別な意味を持ちます。これは、長いエスケープシーケンス名やエスケープシーケンスの引数を 終わらせます。 それ以外の場合は、特別な意味はありません。

space 空白文字は、機能的な働きしか持たない文字です。これは、 リクエストやマクロの引数の区切り文字、およびテキスト行における単語の区切り文字です。また、空白文字は groff の単語間の水平方向の空白計算に作用します。定義された空白幅を得たい場合は、'**\ **' (エスケープ文字とそれに続いた空白文字)、**\|**, **\^**, あるいは **\h** のようなエスケープシーケンスを使う必要があります。

newline
テキストの段落においては、改行は空白文字とほぼ同じような作用をします。 連結された行は、エスケープした改行で指定できます。つまり、行の最後の文字に'****'を指定するのです。

tab テキスト中にタブ文字があれば、インタプリタは次の定義済みのタブ位置へ 水平方向にインデントを作ります。タブ位置の調整には洗練されたインタフェースが存在します。

数式

数値 は、符号つき整数、符号なし整数、浮動小数点実数のいずれか、 およびそれに単位指定子を付加したものです。**単位指定子** は、計測単位を表す 1 文字の略語です。単位指定子が後についた数字は、サイズに関する値を意味しています。 デフォルトでは、数値は単位指定を持ちません。 つまり、単なる数字にすぎません。

roff 言語では、次の単位指定子を定義しています。

c	センチメートル
i	インチ
P	パイカ = 1/6 インチ
p	ポイント = 1/72 インチ
m	em = ポイント値でのフォントサイズ (文字'm'の幅)
M	em の 100 倍
n	en = em/2
u	実際の出力デバイスでの基本単位
v	基本単位での行送り幅
z	スケールされたポイント数 = 1 ポイントの 1/*sizescale* 倍 (フォントの *DESC* ファイルで定義)

数式 は、先で定義された数値と算術演算子'**+**', '**-**', '*****', '**/**', '**%**' (*剰余*), 比較演算子'**==**' ('**=**'と同じ), '**<=**', '**>=**', '**<**', '**>**', 論理演算子'**&**' (*論理積*), '**:**' (*論理和*), '**!**' (*否定*), および括弧'**(**'と'**)**'との組み合わせです。

さらに、*groff* は、数式に対して次の演算子を追加しました。

e1>?e2	*e1* と *e2* の最大値
e1<?e2	*e1* と *e2* の最小値
(c;e)	*c* をデフォルトの単位指定子として *e* を評価

詳細は groff info ファイルを参照してください。

条件式

条件式 は、リクエスト **.if**, **.ie**, および **.while** での判定文で出てきます。次の表にいろいろな種類の条件式を示します。

N	数式 *N* は値が 0 より大きければ真を返します。
!*N*	*N* の値が 0 以下であれば真です。
'*s1*'*s2*'	文字列 *s1* が文字列 *s2* と同一であれば真です。
!'*s1*'*s2*'	文字列 *s1* が文字列 *s2* と同一でなければ真です。
c*ch*	文字 *ch* が利用可能であれば真です。

d*name*	*name* という文字列、マクロ、ディバージョン、リクエストが存在すれば真です。
e	現在のページ番号が偶数なら真です。
o	現在のページ番号が奇数なら真です。
n	フォーマッタが nroff なら真です。
r*reg*	*reg* というレジスタがあれば真です。
t	フォーマッタが troff なら真です。

リクエスト

このセクションでは、定義済みのリクエストについて短いリファレンスを 与えます。groff では、リクエストとマクロの名前を任意の長さにすることができます。 長い名前に対して括弧で括ったりマークをつける必要はありません。

たいていのリクエストは 1 つ、ないしは複数の引数をとります。引数は空白文字で区切られます (タブではありません!)。 引数の長さや数について、固有の制限はありません。 引数はダブルクォートで括ることができます。 引数に空白文字が含まれる場合にとても便利です。 例えば、*"arg with space"* は 1 つの引数を表しています。

リクエストの中には、引数を与えると、引数なしの場合と異なった動きを するものがあります。 その詳細すべてについては、ここでは説明しません。詳細は、groff info ファイルを参照してください。

後に述べるリクエストの説明では、引数の名前の多くは その意味を表すように選ばれています。次にあげる表記についてだけは、意味を明らかにする必要があるでしょう。

c	1 文字を表します。
font	フォント名あるいはフォント番号で指定されたフォント
anything	行末までのすべての文字、あるいは **\{** と **\}** に囲われた文字
n	評価されると整数値を返す数式
N	符号つきまたは符号なしの任意の数式
±*N*	符号によって 3 つの意味があります。次に説明をします。

±*N* で定義された式が ‘**+**’ 符号で始まる場合、この式の結果の値は、関連するリクエストが すでに持っている固有値に加算されます。 例えば、数値レジスタに加算されます。 式が ‘**–**’ で始まる場合、この式の結果の値は、リクエストの値から減算されます。

符号がない場合は、既存の値を *N* で直接置き換えます。負の値を指定する場合は、0 を前に置くかあるいは負の値を括弧で括ってください。

リクエストの簡易リファレンス

.
空行 (無視されます)。ドキュメントの整形に便利です。

.\" *anything*
行全体がコメントとなります。

.ab *string*
string を標準エラー出力に出力し、プログラムを終了します。

.ad
現在の位置揃えモードで出力行の位置揃えを開始します。

.ad *c*
位置揃えモード *c* で行の位置揃えを開始します (*c* = l,r,b,n)。

.af *register c*
フォーマット *c* をレジスタ *register* に割り当てます (*c* = 1,i,I,a,A)。

.aln *alias register*
レジスタ *register* の別名 *alias* を作成します。

.als *alias object*
リクエスト、文字列、マクロ、ディバージョン *object* の別名 *alias* を作成します。

.am *macro*
.. が呼ばれるまでのものをマクロ *macro* に追加します。

.am *macro end*
.end が呼ばれるまでのものをマクロ *macro* に追加します。

.am1 *macro*
リクエスト **.am** と同じですが、マクロ展開の時に互換モードが無効になります。

.am1 *macro end*
リクエスト **.am** と同じですが、マクロ展開の時に互換モードが無効になります。

.as *stringvar anything*
文字列 *anything* を文字列変数 *stringvar* に追加します。

.asciify *diversion*
ディバージョン *diversion* に含まれる ASCII 文字、スペース、およびエスケープシーケンスの いくつかをアンフォーマットします。

.backtrace
入力のバックトレースを標準エラー出力に出力します。

.bd *font N*
フォント *font* を *N*−1 単位分強調します。

.bd *S font N*
現在のフォントが *font* のときに特別フォント *S* で強調します。

.blm 空行マクロを解除します。

.blm *macro*
空白行マクロをマクロ *macro* に設定します。

.box 現在のディバージョンを終了します。

.box *macro*
macro へ転換します。その際、部分的に行詰めされた行は取り除きます。

.boxa 現在のディバージョンを終了します。

.boxa *macro*
macro へ転換し、追加します。その際、部分的に行詰めされた行は 取り除きます。

.bp 現在のページを終了して新しいページを開始します。

.bp ±*N* 現在のページを終了します。次のページ番号を ±*N* にします。

.br 改行です。

.brp 改行し、出力行を引き延ばします。**\p** と同じです。

.break while ループを終了します。

.c2 非改行制御文字を“’”に戻します。

.c2 *c* 非改行制御文字を *c* に設定します。

.cc 制御文字を‘.’に戻します。

.cc *c* 制御文字を *c* に設定します。

.ce 次の入力行をセンタリングします。

.ce *N* 次に来る *N* 行の入力行をセンタリングします。

.cf *filename*
ファイル *filename* の内容をそのまま標準出力またはディバージョンへコピーします。

.cflags *mode c1 c2 ...*
mode 番号に従って、文字 *c1*, *c2*, ... を扱います。

.ch *trap N*
trap の場所を *N* に変更します。

.char *c anything*
文字 *c* を 文字列 *anything* と定義します。

.chop *object*
マクロ、文字列、ディバージョン *object* の最後の 1 文字をとり除きます。

.close *stream*
ストリーム *stream* をクローズします。

.continue
while ループにおける現在の繰り返し処理を終了します。

.cp 互換モードを有効にします。

.cp *N* *N* が 0 なら互換モードを無効にします。それ以外なら有効にします。

.cs *font N M*
フォント *font* の固定文字幅モードを *N*/36 em に設定します (em は値 *M*)

.cu *N* nroff の場合の連続アンダーライン。troff のリクエスト **.ul** に相当します。

.da 現在のディバージョンを終了します。

.da *macro*
マクロ *macro* に転換･追加します。

.de *macro*
リクエスト **..** が呼ばれるまでマクロ *macro* を定義 (再定義) します。

.de *macro end*
リクエスト **.end** が呼ばれるまでマクロ *macro* を定義 (再定義) します。

.de1 *macro*
リクエスト **.de** と同じですが、マクロ展開の時に互換モードが無効になります。

.de1 *macro end*
リクエスト **.de** と同じですが、マクロ展開の時に互換モードが無効になります。
.dei *macro*
文字列レジスタ *macro* に名前が含まれるマクロを、 **..** が呼ばれるまで定義 (再定義) します。
.dei *macro end*
間接的にマクロを定義 (再定義) します。 *macro* および *end* は文字列レジスタであり、この内容がそれぞれマクロ名と終了マクロに 挿入されます。
.di 現在のディバージョンを終了します。
.di *macro*
マクロ *macro* に転換します。
.do *name*
リクエスト **.name** を、互換モードを無効にして解釈します。
.ds *stringvar anything*
文字列変数 *stringvar* に文字列 *anything* を設定します。
.dt *N trap*
ディバージョンのトラップ位置を *N* に設定します (デフォルトの単位指定子は **v**です)。
.ec エスケープ文字を '****' に戻します。
.ec *c* エスケープ文字を *c* に設定します。
.ecr **.ecs** を用いて保存されているエスケープ文字を復元します。
.ecs 現在のエスケープ文字を保存します。
.el *anything*
ifelse リクエスト (*.ie*) の else ブロックです。
.em *macro*
入力が終わった後にマクロ *macro* を実行します。
.eo エスケープ文字の処理を抑制します。
.ev 直前の環境に変更します。
.ev *env* 環境 *env* (番号または名前で指定) をプッシュして切り替えます。
.evc *env*
環境 *env* の内容を現在の環境にコピーします。環境のプッシュまたはポップは行いません。
.ex roff の処理を終了します。
.fam 以前のフォントファミリを返します。
.fam *name*
現在のフォントファミリを *name* に設定します。
.fc フィールド機構を無効にします。
.fc *a* フィールド区切りを *a* に設定し、パディング文字を空白にします。
.fc *a b* フィールド区切りを *a* に、パディング文字を *b* に設定します。
.fi 出力行を埋めます。
.fl 出力バッファをフラッシュします。
.fp *n font*
位置 *n* にフォント *font* をマウントします。
.fp *n internal external*
長い名前 *external* のフォントを短い名前 *internal* のフォントとし、位置 *n* にマウントします。
.fspecial *font s1 s2...*
現在のフォントが *font* の場合に、*s1*, *s2*, *...* を特別フォントにします。
.ft 直前のフォントに戻します。リクエスト **\fP** と同じです。
.ft *font*
フォント名または番号 *font* に変更します。エスケープシーケンス **\f[***font***]** と同じです。
.ftr *font1 font2*
フォント *font1* をフォント *font2* に変換します。
.hc 追加ハイフネーション指定文字を削除します。
.hc *c* 追加ハイフネーション指定文字として *c* を設定します。
.hcode *c1 code1 c2 code2 ...*
文字 *c1* のハイフネーションコードを *code1* に、文字 *c2* のコードを *code2* のように設定します。

.hla *lang*
現在のハイフネーション言語を *lang* に設定します。
.hlm *n* ハイフンされた行の最大連続数を *n* に設定します。
.hpf *file*
ハイフネーションのパターンをファイル *file* から読み込みます。
.hw *words*
例外的なハイフネーションをする単語のリストを *words* で指定します。
.hy *N* ハイフネーションモードを *N* に変更します。
.hym *n* ハイフネーションの余白を *n* に設定します (デフォルトの単位指定子は **m** です)。
.hys *n* ハイフネーションの空白を *n* に設定します。
.ie *cond anything*
条件式 *cond* が真ならば *anything* を処理します。偽の場合はリクエスト **.el** へ移動します。
.if *cond anything*
条件式 *cond* が真ならば *anything* を処理します。偽の場合は何もしません。
.ig リクエスト **..** が呼ばれるまでテキストを無視します。
.ig *end* リクエスト **.end** が呼ばれるまでテキストを無視します。
.in インデント量を直前の値に戻します。
.in ±*N* 引数 ±*N* に従ってインデント量を変更します (デフォルトの単位指定子は **m**です)。
.it *N trap*
入力行のカウントトラップを位置 *N* に設定します。
.kern ペアワイズカーニングを有効にします。
.kern *n* *n* が 0 ならばペアワイズカーニングを無効にします。0 でなければ有効にします。
.lc リーダ繰り返し文字の定義を削除します。
.lc *c* リーダ繰り返し文字を *c* に設定します。
.length *register anything*
文字列 *anything* の文字列長をレジスタ *register* に書き込みます。
.linetabs
行タブモードを有効にします (つまり、出力行に比例したタブ位置を計算します)。
.linetabs *n*
n が 0 の場合、行タブモードを無効にします。それ以外の場合は、行タブモードを有効にします。
.lf *N file*
入力する行数を *N* に、ファイル名を *file* に設定します。
.lg *N* 引数 *N* が 0 より大きければリガチャ (合字) モードにします。
.ll 行幅を直前の値に戻します。
.ll ±*N* 行幅を引数 ±*N* に従って設定します (デフォルトの設定は `6.5`**i** で、単位指定子は **m** です)。
.ls 追加の行間スキップ量を直前の値に戻します。
.ls *N* 追加の行間スキップ量を *N* に設定します。 つまり、テキストの出力行それぞれの後に *N*−1 行の空白行を挿入します。
.lt ±*N* タイトルの長さです (デフォルトの単位指定子は **m** です)。
.mc 余白文字を無効にします。
.mc *c* それぞれのテキスト行の後、 右側余白から現在設定されている距離に文字 *c* を出力します。
.mc *c N* 余白文字を *c* に、右側余白からの距離を *N* に設定します (デフォルトの単位指定子は **m** です)。
.mk *register*
現在の垂直位置を *register* にマークします。
.mso *file*
リクエスト .so と同じですが、tmac ディレクトリにある *file* が検索される点が違います。
.na 出力行の位置揃えを行いません。
.ne 1 行分の行送りが必要であることを指定します。
.ne *N* *N* 行分の行送りが必要であることを指定します (デフォルトの単位指定子は **v** です)。
.nf 出力行に行詰めや位置揃えを行いません。
.nh ハイフネーションをしません。
.nm 行番号モードを無効にします。

.nm ±*N M S I*
行番号モードの、行番号、行番号出力間隔、空白、インデントを設定します。
.nn 次の行に行番号をつけません。
.nn *N* 次の *N* 行に行番号をつけません。
.nop *anything*
anything を常に実行します。
.nr *register* ±*N M*
レジスタの値を、インクリメント値 *M* で ±*N* に設定・変更します。
.nroff 組み込み条件式 n を真に、t を偽にします。
.ns 空白なしモードにします。
.nx *filename*
次のファイルへ処理を移します。
.open *stream filename*
ファイル **filename** を書き込みモードでオープンし、名前 **stream** を持つストリームに関連づけます。
.opena *stream filename*
リクエスト **.open** と同じですが、追加モードでファイルをオープンします。
.os リクエスト **.sv** で指定された行送り量を出力します。
.pc ページ番号文字を‘**%**’に戻します。
.pc *c* ページ番号文字を設定します。
.pi *program*
プログラム *program* に出力をパイプします (nroff のみ)。
.pl ページ長をデフォルトの 11**i** に設定します。現在のページ長はレジスタ **.p** に格納されています。
.pl ±*N* ページ長を ±*N* に変更します (デフォルトの単位指定子は **v** です)。
.pm マクロ名とサイズを出力します (サイズは 1 ブロック 128 バイトのブロック数です)。
.pm t マクロ全体のサイズのみを出力します (サイズは 1 ブロック 128 バイトのブロック数です)。
.pn ±*N* 次のページ番号を *N* に設定します。
.pnr 現在定義されている数値レジスタの名前と内容を標準エラー出力に 出力します。
.po ページオフセットを直前の値に戻します。 現在のページオフセットはレジスタ **.o** に格納されています。
.po ±*N* ページオフセットを *N* に設定します。
.ps ポイントサイズを直前の値に戻します。
.ps ±*N* ポイントサイズを指定します。エスケープシーケンス **\s[**±*N***]** と同じです。
.psbb *filename*
PostScript 画像 *filename* のための矩形領域を確保します。
.pso *command*
リクエスト **.so** と同様ですが、*command* の標準出力から入力します。
.ptr すべてのトラップの名前と位置を標準エラー出力に出力します (入力行のトラップとディバージョンのトラップは含まれません)。
.rchar *c1 c2...*
文字定義 *c1*, *c2*, ... を削除します。
.rd *prompt*
標準入力からの入力を読み込みます。
.return
マクロから戻ります。
.rj *n* 次の *n* 行の入力行を右寄せします。
.rm *name*
name で指定されたリクエスト、マクロ、文字列を削除します。
.rn *old new*
old で指定されたリクエスト、マクロ、文字列の名前を *new* に変更します。
.rnn *reg1 reg2*
レジスタ名 *reg1* を *reg2* に変更します。
.rr *register*
レジスタ *register* を削除します。
.rs 空白を復活させます。つまり空白なしモードを無効にします。

.rt ±*N* (上方向のみ、) マークしておいた垂直位置まで戻します (デフォルトの単位指定子は **v** です)。

.shc ソフトハイフン文字を **\(hy** に戻します。

.shc *c* ソフトハイフン文字を *c* に設定します。

.shift *n*
マクロにおいて、引数を位置 *n* にシフトします。

.so *filename*
ソースファイルをインクルードします。

.sp 1 行スキップします。

.sp *N* *N* の行送りを挿入します。*N* の符号によって上下方向が決まります (デフォルトの単位指定子は **v** です)。

.special *s1 s2 ...*
フォント *s1*, *s2*, などを特別フォントとします。現在のフォントにない文字をこれらから検索します。

.ss *N* 空白文字のサイズを *N*/12 に設定します。単位は現在のフォントの空白幅です。

.ss *N M* 空白文字のサイズを *N*/12 に、文の空白サイズを *M*/12 に設定します。単位は現在のフォントの空白幅 (=1/3 em) です。

.sty *n style*
位置 *n* のフォントをスタイル *style* に関連づけます。

.substring *register n1 n2*
レジスタ *register* の文字列の中の部分文字列 *n1* を *n2* に置き換えます。

.sv 行送り量を 1**v** に設定します。

.sv *N* リクエスト **.os** で出力される空白行の送り量を *N* に設定します。

.sy *command-line*
プログラム *command-line* を実行します。

.ta T *N* タブ位置を *N* の倍数に設定します (デフォルトの単位指定子は **m** です)。

.ta *n1 n2 ... nn* **T** *r1 r2 ... rn*
位置 *n1*, *n2*, … , *nn* のタブ位置をそれぞれ *nn+r1*, *nn+r2*, … *nn+rn* に、さらにそれ以降を *nn+rn+r1*, *nn+rn+r2*, … *nn+rn+rn* のように設定します。

.tc タブ繰り返し文字を削除します。

.tc *c* タブ繰り返し文字を *c* に設定します。

.ti ±*N* 次の行を一時的にインデントします (デフォルトの単位指定子は **m** です)。

.tkf *font s1 n1 s2 n2*
フォント *font* のトラックカーニングを有効にします。

.tl '*left*'*center*'*right*'
3 つの部位をもつタイトルです。

.tm *anything*
anything を端末 (UNIX の標準的なメッセージ出力先) に出力します。

.tm1 *anything*
anything を端末 (UNIX の標準的なメッセージ出力先) に出力します。 その際、*anything* が **"** で始まっている場合は、先頭を空白で始めることができます (**"** 自体は取り除かれます)。

.tmc *anything*
.tm1 と似ていますが、末尾の改行を出力しません。

.tr *abcd....*
出力で *a* を *b* に、*c* を *d* のように変換します。

.trf *filename*
ファイル *filename* の内容をそのまま出力します。

.trnt *abcd....*
リクエスト **.tr** と同じですが、**\!** によってディバージョンへと出力されるテキストは変換されません。

.troff 組み込み条件式 t を真に、n を偽に設定します。

.uf *font*
アンダーラインフォントセットを *font* に設定します (リクエスト **.ul** で切り替わります)。

.ul *N* 入力行を *N* 行、アンダーライン付きにします (troff では イタリック体になります)。

.unformat *diversion*
diversion 内のフォント情報を保存しつつ、空白文字およびタブ文字を アンフォーマットします。

.vpt *n* *n* が 0 でなければ垂直位置のトラップを有効にします。0 ならば無効にします。
.vs 行送りの基準線間隔を直前の値に戻します。
.vs *N* 行送りの基準線間隔を *N* に設定します。デフォルト値は 12**p** です。
.warn *n* 警告コードを *n* に設定します。
.wh *N trap*
位置によるトラップを設定します。負の値の場合はページ末尾からになります。
.while *cond anything*
条件式 *cond* が真ならば *anything* を入力として受理します。
.write *stream anything*
ストリーム *stream* へ *anything* を書き込みます。

これらの基本的な groff リクエストの他にも、マクロの呼び出しがあります。これらはマクロパッケージ (概要は roff(7) を参照) やプリプロセッサに由来するものです。

プリプロセッサマクロを認識するのは容易です。これらは独特のマクロの対で括られています。

プリプロセッサ	開始マクロ	終了マクロ
eqn	**.PS**	**.PE**
grap	**.G1**	**.G2**
grn	**.GS**	**.GE**
pic	**.PS**	**.PE**
refer	**.R1**	**.R2**
soelim	なし	なし
tbl	**.TS**	**.TE**

エスケープシーケンス

エスケープシーケンスは、行中の要素であり、通常は バックスラッシュ‘****’で開始され、エスケープ名がそれに続きます。 場合によっては、必要な引数をつけることもあります。 入力処理は、エスケープ文字あるいはその引数の直後から再開されます。 引数には区切り文字がはさまっているわけではありません。そのため、エスケープ名や引数の終わりを決定する方法が必要です。

これは、名前 (エスケープ名、および変数名からなる引数) を 角括弧 **\[***name***]** で囲い、定数引数 (数式および文字) を アポストロフィ (ASCII 0x27) で *'constant'* のように囲うことによって実現します。

エスケープ名には短い名前の省略形があります。2 文字のエスケープ名は開き括弧で **\(***xy* のように指定され、閉じ括弧は必要ありません。そして、特殊文字である‘**[**’と‘**(**’以外の 1 文字の名前はすべて *****c* のようにマーカなしで指定することができます。

長さ 1 の定数の引数もアポストロフィのマーカを省略できますが、2 文字の名前のときには省略できません。

1 文字のエスケープシーケンスは主に行中での 関数とシステムに関連したタスク用に使われるのに対して、**\(** が続いた 2 文字の名前は roff システムで定義済みの特殊文字用に 使われます。2 文字より多い文字を持った名前 **\[***name***]** は、ほとんどユーザ定義の文字を表しています (リクエスト **.char** を参照してください)。

1 文字のエスケープシーケンス

\" コメントの開始です。行末までのものすべてが無視されます。
\# 次の改行文字までのものすべてが無視されます。 これは、コピーモードで解釈されます。 **\"** と似ていますが、終端となる改行を無視する点が異なります。
(st* 2 文字の名前 *st* を持った文字列変数に格納されている文字列です。
[stringvar***]**
任意の長さの名前 *stringvar* を持った文字列変数に格納されている文字列です。
\$0 現在のマクロが呼び出されている名前。リクエスト **.als** は、1 つのマクロに複数の名前をつけることができます。
\$(*xy* 2 桁の番号 *xy* 番目のマクロの引数。
\$[*nexp***]**
nexp 番目のマクロの引数。ここで、*nexp* は 1 以上の整数に評価される数値表現です。
\$* マクロにおいて、引数すべてを空白文字区切りで連結したもの。

\$@ マクロにおいて、引数すべてをそれぞれダブルクォートで囲い、 空白文字で区切って連結したもの。

**** バックスラッシュ 1 個に縮退します。 コピーモードでエスケープ文字として解釈されるのを遅らせるときに便利です。表示可能なバックスラッシュには、**\e** を使用してください。

\' 揚音アクセント符号 ´ 。**\(***aa* と同じです。エスケープなし: アポストロフィ、右クォーテーション記号、シングルクォート (ASCII 0x27)。

\` 抑音アクセント符号 。 **\(***ga* と同じです。 エスケープなし: 左クォート、バッククォート (ASCII 0x60)。

\- 現在のフォントでの − 符号。

\. 解釈されないドット (ピリオド)。行頭にあってもそうなります。

\% デフォルトでのオプションのハイフン文字。

\! 透過行指定子。

\?*anything***?**
ディバージョンにおいて、そのまま *anything* を組み込みます。コピーモードで *anything* は読み込まれます。エスケープシーケンス **\!** および **\?** も参照してください。

*****space*
パディングされない、空白の大きさを持った空白文字 (改行もされません)。

\0 数字の幅の空白。

\| 1/6 em の狭い空白文字。nroff では幅 0 です。

\^ 1/12 em のさらに狭い空白文字。nroff では幅 0 です。

\& 表示されない、幅 0 の文字。

\) **\&** と同様ですが、文の終わりを認識する目的で、cflags リクエストを 用いて透過になるように宣言された文字のように振る舞うという 点で異なります。

\/ 次の文字がローマン体であるときには、それの前の文字の幅を増やして 次の文字との間の空白が修正されるようにします。

\, 前の文字がローマン体であるときには、次の文字の空白を変更して 前の文字との間の空白が修正されるようにします。

\~ 改行されない空白。行を調整するとき、通常の単語間の空白のように 引き延ばされます。

\: 幅 0 のブレークポイント (**\%** と似ていますが、ソフトハイフン文字は使いません)。

*****newline*
無視される改行。行を連続させるときのためです。

\{ 条件入力の開始。

\} 条件入力の終了。

\(*st* 2 文字の名前を持った特殊文字。セクション **特殊文字** を参照してください。

\[*name***]**
任意の長さの名前 *name* を持った文字

\a 解釈されないリーダ文字。

\A'*anything***'**
anything が文字列、マクロ、ディバージョン、レジスタ、環境、あるいは フォント名として受け付けられるものの場合、1 に展開します。そうでない場合 0 になります。

\b'*abc...***'**
角括弧を作る関数。

\B'*anything***'**
anything が正しい数値表現として受け付けられるものの場合、1 に展開します。そうでない場合、0 になります。

\c テキスト処理を中断します。

\C'*char***'**
char と呼ばれる文字。**\[***char***]** と同じですが、他の roff バージョンと互換性があります。

\d 垂直単位で 1/2 em だけ前方 (下) に移動します (nroff では 1/2 行です)。

\D'*charseq***'**
charseq 中の文字で定義されたグラフィカルな要素を描きます。詳細は groff info ファイルを参照してください。

\e 現在のエスケープ文字を表示する時の表記。

\E エスケープ文字と等価ですが、コピーモードでは評価されません。

\f*F* 1 文字の名前もしくは 1 個の数字 *F* を持つフォントに変更します。

\f(*fo* 2 文字の名前もしくは 2 個の数字 *fo* を持つフォントに変更します。

\f[*font***]**
任意の長さの名前もしくは数式 *font* で表されるフォントに変更します。

\g[*reg***]**
リクエスト **.af** に適した、名前 *reg* を持ったレジスタのフォーマットを返します。別の形式として **\g(***xy* および **\g***x* があります。

\h'*N***'** 局所的な水平移動。右側へ *N* だけ移動します (負の値のときは左側へ移動します)。

\H'*N***'** 現在のフォントの高さを *N* に設定します。

\k[*reg***]**
任意の長さの名前 *reg* を持つレジスタ中の水平入力位置をマークします。 別の形式では **\k(***xy* と **\k***x* です。

\l'*Nc***'**
水平線描画関数 (文字 *c* を用いることも可能です)。

\L'*Nc***'**
垂直線描画関数 (文字 *c* を用いることも可能です)。

\n*r* 1 文字の名前 *r* を持つレジスタ変数に格納されている数値。

\n(*re* 2 文字の名前 *re* を持つレジスタ変数に格納されている数値。

\n[*reg***]**
任意の長さの名前 *reg* を持つレジスタ変数に格納されている数値。

\N'*n***'** 現在のフォントで、コード *n* で指定された文字を出力します。 特別フォントは検索されません。リクエスト **.char** を用いて文字にフォントを追加するときに便利です。

\o'*abc...***'**
文字 *a*, *b*, *c*, などを 2 度打ちします。

\p 改行し、その出力行を引き延ばします。

\r 垂直方向に 1 em 逆戻りします (nroff では 1 行逆戻りします)。

\R'*name* ±*n***'**
リクエスト **.nr** *name* ±*n* と同じです。

\s[±*N***]**
ポイントサイズを *N* スケールポイントに設定します。 別の書式として ¥s±[*N*], ¥s'±*N*', ¥s±'*N*', **\s(**±*xy*, ¥s±(*xy*, **\s**±*x* が使えることに注意してください。これは、リクエスト **.ps** と同じです。

\S'*N***'** 出力を *N* 度傾けます。

\t 解釈されない水平タブ。

\u 垂直方向に 1/2 em 逆戻りします (nroff では 1/2 行逆戻りします)。

\v'*N***'** 局所的な垂直移動。*N* だけ下がります (負の値のときは上がります)。

\V[*env***]**
環境変数 *env* の内容。別の書式では、**\V(***xy* と **\V***x* が使えます。

\w'*string***'**
文字列 *string* の幅。

\x'*N***'** さらに行送りする関数 (負の値なら前へ、正の値なら後ろへ行送りします)。

\X'*string***'**
string をデバイス制御関数として出力します。

\Y[*name***]**
文字列変数あるいはマクロ *name* をデバイス制御関数として解釈しないように出力します。別の書式では、**\Y(***xy* と **\Y***x* になります。

\z*c* 幅 0 (空白なし) で *c* を出力します。

\Z'*anything***'**
anything を出力し、水平・垂直位置を元に戻します。 *anything* にはタブや先頭文字は含まれません。

エスケープシーケンス **\e**, **\.**, **\"**, **\$**, *****, **\a**, **\n**, **\t**, **\g**, および *****newline* はコピーモードで解釈されます。

\(あるいは **\[** で始まるエスケープシーケンスは 1 文字のエスケープシーケンスを 表すものではなく、2 つ以上の文字を持ったエスケープ名の 開始を示します。

バックスラッシュの後に定義済みのエスケープシーケンスを 形成しない文字が続く場合は、バックスラッシュは黙って無視され、文字そのものがマップされます。

特殊文字

よく使われる特殊文字は、文字 *x* と *y* を用いた **\(***xy* の形式のエスケープシーケンスであらかじめ定義されています。 特殊文字の中には通常のフォントに含まれているものも一部ありますが、 ほとんどは特別フォントでしか利用できません。最も重要なグリフを抜き出したものを次に示します。完全なリストは、groff_char(7) にあります。

\(bu	·	中黒.
\(co	©	著作権記号.
\(ct		セント記号 (通貨).
\(dd	‡	2 重ダガー.
\(de	°	度記号.
\(dg	†	ダガー.
\(em		全角ダッシュ.
\(hy		ハイフン.
\(rg	®	登録記号.
\(sc	§	セクション記号.
\(ul	_	アンダーライン文字.
\(==	≡	等号.
\(>=	≥	以上.
\(<=	≤	以下.
\(!=	≠	不等号.
\(->	→	右矢印.
\(<-	←	左矢印.
\(+-	±	プラスマイナス記号.

レジスタ

レジスタは値を格納する変数です。groff では、ほとんどのレジスタは数値を格納しますが (前述セクション **数式** を参照してください)、なかには文字列値を保持できるものもあります。

各レジスタには名前が与えられています。任意のレジスタを定義でき、リクエスト **.nr** *register* で定義・設定できます。

レジスタに格納された値は、エスケープシーケンス **\n** を用いて取り出すことができます。

最も有用なのは、定義済みのレジスタです。次からは、レジスタのことを話しているのだということを明確にするため、*name* という表記を用いて **name** レジスタのことを示します。**\en[]** 修飾はレジスタ名の一部ではないことを気にとめておいてください。

読み込み専用レジスタ

次のレジスタは、ユーザが書き換えられない定義済みの値を持った レジスタです (通常、ドットで始まるレジスタは読み込み専用です)。ほとんどは、現在の設定についての情報を提供したり、リクエスト呼び出し からの結果を保存したりするものです。

\n[**.$**] 現在のマクロの引数の個数。

\n[**.a**] 最後に **\x'***N***'** を用いて指定した行送り幅。

\n[**.A**] オプション –A が使用されているときに troff 内で 1 が設定されます。nroff 内では常に 1 です。

\n[**.c**] 現在の入力行番号。

\n[**.C**] 互換モードが有効になっている場合は 1 で、そうでない場合 0 です。

\n[**.cdp**]
現在の環境に追加された最後の文字の深さ。 最後の文字が基準線から下に突き出ている場合に正になります。

\n[**.ce**] リクエスト **.ce** で設定された、センタリングされる行の残数。

\n[**.cht**]
現在の環境に追加された最後の文字の高さ。 基準線よりも上に文字が突き出ている場合に正になります。

\n[**.csk**]
現在の環境に追加された最後の文字の傾き。文字の傾きとは、文字の上についたアクセント記号が文字の中央からどれだけ 右に離れて置かれるかということです。

\n[**.d**] 現在のディバージョンでの垂直方向の位置です。レジスタ **nl** と同じです。

\n[**.ev**] 現在の環境の名前もしくは番号です (文字列の値です)。

\n[**.f**] 現在のフォント番号です。

\n[**.fam**]
現在のフォントファミリです (文字列の値です)。

\n[**.fp**] 次に空いているフォント位置番号。

\n[**.g**] GNU troff では常に 1 です。マクロ中で groff で動作しているかどうかをテストする場合、このレジスタを使用すべきでしょう。

\n[**.h**] 現在のページあるいはディバージョンでのテキスト基準線の 最高点です。
\n[**.H**] 利用可能な水平方向の解像度 (基本単位です)。
\n[**.hla**]
リクエスト .hla で設定された現在のハイフネーション言語。
\n[**.hlc**]
直前で連続したハイフネーション行数。
\n[**.hlm**]
連続したハイフネーション行数の許される最大値。これは、リクエスト **.hlm** で設定されます。
\n[**.hy**] 現在のハイフネーションフラグ (リクエスト **.hy** で設定されます)。
\n[**.hym**]
現在のハイフネーションの余白 (リクエスト **.hym** で設定されます)。
\n[**.hys**]
現在のハイフネーションの空白 (リクエスト **.hys** で設定されます)。
\n[**.i**] 現在のインデント。
\n[**.in**] 現在の出力行に対して適用されるインデント。
\n[**.int**]
最後に出力した行に **\c** が含まれていれば正です。
\n[**.kern**]
ペアワイズカーニングが有効になっている場合 1 で、無効になっていれば 0 です。
\n[**.l**] 現在行の長さです。
\n[**.lg**] 現在の合字モード (リクエスト **.lg** で設定されます)。
\n[**.linetabs**]
現在の行タブモード (**.linetabs** リクエストによって設定されます)。
\n[**.ll**] 現在の出力行に対して適用される行の長さ。
\n[**.lt**] タイトルの長さ (リクエスト **.lt** で設定されます)。
\n[**.n**] 直前の出力行でのテキスト部分の長さ。
\n[**.ne**] トラップを生起させる原因となった最後のリクエスト **.ne** が必要とした空白の量。このレジスタは、**.trunc** と一緒に使用すると便利です。
\n[**.ns**] 空白なしモードであれば 1 で、そうでなければ 0 が設定されます。
\n[**.o**] 現在のページのオフセット。
\n[**.p**] 現在のページの長さ。
\n[**.pn**] 次ページの番号。これは、リクエスト **.pn** で定義された値もしくは現在ページに 1 を足した番号の いずれかです。
\n[**.ps**] スケールポイントで表した現在のポイントサイズ。
\n[**.psr**]
最後に要求された、スケールポイントで表したポイントサイズ。
\n[**.rj**] リクエスト **rj** によって設定された右寄せされる行の数。
\n[**.s**] 10 進小数での現在のポイントサイズ。
\n[**.sr**] 10 進小数での最後に要求されたポイントサイズ (文字列値)。
\n[**.t**] 次のトラップまでの距離。
\n[**.T**] オプション **-T** が使われている場合 1 に設定されます。
\n[**.tabs**]
現在のタブ設定の文字列表現。リクエスト **.ta** に対する引数として使用するのに適しています。
\n[**.trunc**]
最近の垂直位置トラップによって切り詰められた垂直スペースの量。リクエスト **.ne** によるトラップが引き起こされた場合は、そのリクエストによって生じた 垂直移動量をさらにマイナスした値になります。 別の言い方をすれば、トラップが発生した時点での、トラップが起きなかった場合の垂直位置と現在の垂直位置との差を 表しています。リクエスト **.ne** と一緒に使用すると便利です。
\n[**.ss**] リクエスト **.ss** の第 1 引数によって設定されたパラメータの値。
\n[**.sss**]
リクエスト **.ss** の第 2 引数によって設定されたパラメータの値。
\n[**.u**] 行連結モードのときは 1 で、そうでなければ 0 です。
\n[**.v**] 現在の垂直方向の行送り量。
\n[**.V**] 利用可能な垂直方向の解像度 (基本単位です)。

\n[**.vpt**]
垂直位置トラップが有効なら 1 で、無効なら 0 です。
\n[**.w**] 直前の文字の幅。
\n[**.warn**]
現在有効になっている警告の番号コードの和。
\n[**.x**] メジャーバージョン番号。
\n[**.y**] マイナバージョン番号。
\n[**.Y**] groff のリビジョン番号。
\n[**.z**] 現在のディバージョン名。

書き込み可能なレジスタ

次のレジスタは、ユーザによって読み書き可能です。 定義済みのデフォルト値を持っていますが、ドキュメントを カスタマイズするために変更できます。

\n[**%**] 現在のページ番号。
\n[**c.**] 現在の入力行番号。
\n[**ct**] 文字の種類 (幅関数 **\w** で指定されます)。
\n[**dl**] 最後に行われたディバージョンの最大幅。
\n[**dn**] 最後に行われたディバージョンの高さ。
\n[**dw**] 現在の曜日 (17)。
\n[**dy**] 現在の日 (131)。
\n[**hp**] 現在の入力行における水平位置。
\n[**llx**] 与えられた PostScript 画像 (**.psbb** で設定されます) の左下 x 座標 (PostScript 単位)。
\n[**lly**] 与えられた PostScript 画像 (**.psbb** で設定されます) の左下 y 座標 (PostScript 単位)。
\n[**ln**] 出力行番号。
\n[**mo**] 現在の月 (112)。
\n[**nl**] 最後に表示されたテキストの基準線の垂直方向の位置。
\n[**rsb**] **sb** と同様ですが、文字の高さおよび深さを勘定に含めています。
\n[**rst**] **st** と同様ですが、文字の高さおよび深さを勘定に含めています。
\n[**sb**] 文字列の、基準線の下側の深さ (幅関数 **\w** で生成されます)。
\n[**skw**] **\w** 引数の最後の文字の中央からの右スキップ幅。
\n[**slimit**]
0 よりも大きければ、入力スタック上のオブジェクトの最大数。 0 以下であれば、制限はありません。つまり、仮想メモリを使い果たすまで再帰呼び出しし続けることができます。
\n[**ssc**] 添字の直前にある文字に追加すべき水平方向の空白量 (幅関数 **\w** で生成されます) (負の値になる可能性もあります)。
\n[**st**] 文字列の、基準線の上の高さ (幅関数 **\w** で生成されます)。
\n[**systat**]
最後の **.sy** リクエストによって実行された *system()* 関数の戻り値。
\n[**urx**] 与えられた PostScript 画像 (**.psbb** で設定されます) の右上 x 座標 (PostScript 単位)。
\n[**ury**] 与えられた PostScript 画像 (**.psbb** で設定されます) の右上 y 座標 (PostScript 単位)。
\n[**year**]
現在の年 (2000 年問題対応です)。
\n[**yr**] 現在の年から 1900 を引いたものです。 2000 年問題対応にするにはレジスタ **year** を代わりに使用してください。

警告

groff が生成する警告は各々名前およびコード番号で識別されます。 コードは 2 の累乗になっていて、1 つの整数の上にビットエンコード できるようになっています。また警告のグループを参照するのに使用することのできる名前もあります。

警告と関連のある名前は、 **-w** および **-W** オプションで使用されます。 コード番号は リクエスト **.warn** および **\n[***warn***]** レジスタで使用されます。

all *group*
di, **mac** および **reg** を除いたすべての警告です。伝統的なマクロパッケージでの警

告をすべて網羅しています。

break *4*
行連結モード時に、1 行の長さよりも短くなるように 行を分割できませんでした。デフォルトでは有効です。

char *1*
存在しない文字です。デフォルトでは有効です。

delim *8*
閉じ区切り記号がないか、もしくは対応が取れません。

di *256*
現在ディバージョンが存在しないのに、引数なしで **.di** あるいは **.da** を使っています。

el *16*
対応するリクエスト **.ie** が存在しないのに、**.el** リクエストを使っています。

escape *32768*
認識されないエスケープシーケンスです。そのためエスケープ文字は無視されます。

font *131072*
存在しないフォントです。デフォルトでは有効です。

ig *262144*
リクエスト **.ig** を使って無視されているテキスト中で不正なエスケープがあります。無視されるテキストの外側でこの警告が発生する場合、エラーとなる状態です。

mac *512*
定義されていない文字列、マクロ、およびディバージョンが使われました。 自動的に空文字として扱われます。通常は、名前ごとに 1 つしか警告は出ません。

missing *8192*
オプションではない引数が指定されていないリクエストです。

input *16384*
不正な入力文字です。

number *2*
不正な数式です。デフォルトでは有効です。

range *64*
引数が範囲外です。

reg *1024*
定義されていない番号レジスタを使っています。自動的に値 0 をもつレジスタとして扱われます。通常は、名前ごとに 1 つしか警告は出ません。

right-brace
4096
数字を指定すべき場所で **\}** が使われました。

scale *32*
意味のない単位指定子です。

space *65536*
リクエストあるいはマクロとその引数との間に空白がありません。そのため、自動的にマクロは定義されません。 デフォルトでは有効です。 この警告は互換モードでは絶対に発生しません。

syntax *128*
数式中の構文が曖昧です。

tab *2048*
タブ文字の使い方が適切ではありません (クォートされていない マクロ引数中や数字を指定すべきところにタブ文字がある)。

w *group*
すべての警告です。

Bit	Code	警告	Bit	Code	警告	Bit	Code	警告
0	*1*	char	8	*256*	di	16	*65536*	space
1	*2*	number	9	*512*	mac	17	*131072*	font
2	*4*	break	10	*1024*	reg	18	*262144*	ig
3	*8*	delim	11	*2048*	tab			
4	*16*	el	12	*4096*	rightbrace			
5	*32*	scale	13	*8192*	missing			
6	*64*	range	14	*16384*	input			
7	*128*	syntax	15	*32768*	escape			

互換性

groff は、古典的な *troff* 用に書かれた roff コードや他の roff 実装用の roff コードを同じ方法で処理できるようにする **互換モード** を提供します。

互換モードはコマンドラインオプション **-C** を用いて有効にでき、リクエスト **.cp** で有効にしたり無効にしたりできます。番号レジスタ **\n(.***C* は、互換モードが有効であるとき 1 で、無効であるとき 0 です。

長い名前に対する GNU の考え方によってある種の非互換性が 生まれてしまうのでこれが必要になりました。*古典的な troff* は、

```
.dsabcd
```

を cd という中身を持った文字列 ab を定義しているものとして解釈します。*groff* は、通常これを **dsabcd** という名前のマクロ呼び出しとして解釈します。

さらに、*古典的な troff* では ***[** または **\n[** を **[** と呼ばれる文字列レジスタあるいは番号レジスタへのリファレンスとして 解釈します。しかし、*GNU 独自の モード* では通常これを長い名前の始まりとして解釈してしまいます。

互換性 モードでは、groff はこれらを古典的な方法で解釈するようになりますが、長い名前は 認識されなくなります。

これに対して、*GNU 独自モード* での groff は、文字列やマクロ、変換、番号レジスタ、フォントあるいは 環境名にエスケープシーケンス **\e**, **\|**, **\^**, **\&**, **\}**, **\{**, **** (スペース), **\'**, **\`**, **\-**, **_**, **\!**, **\%**, **\c** を使うことはできません。これに対して *古典的な troff* ではこれらのエスケープシーケンスを使えます。エスケープシーケンス **\A** は、名前の中でこれらのエスケープシーケンスを使わないようにするとき 役に立ちます。

小数のポイントサイズは、顕著な非互換性を生み出します。*古典的な troff* では、**.ps** リクエストは単位指定子を無視しますので、

```
.ps 10u
```

とするとポイントサイズを 10 ポイントに設定します。これに対して groff 独自のモードでは、ポイントサイズは スケールポイントで 10 ポイントに設定します。

groff モードでは、整形されていない入力と整形された出力文字との間に 基本的な違いがあります。出力文字がどのように出力されるかに影響を与えるものは、 すべてその文字と一緒に格納されます。一度出力文字が作られれば、その後でどのようなリクエストが実行されても出力文字は影響を受けません。**.bd**, **.cs**, **.tkf**, **.tr**, **.fp** のいずれのリクエストでも同様です。

通常、出力文字は、入力文字を現在の出力行に追加する直前に作られます。　　マクロ、ディバージョン、文字列は、実はすべて 同じオブジェクトタイプです。 これらは、どのような組み合わせでも入力文字のリスト ならびに出力文字のリストを持っています。

マクロを処理する目的では、出力文字は入力文字と同じような 振る舞いはしません。出力文字は、自分の構築元となった入力文字ならば持っていたであろう 特別な属性を一切引き継ぎません。次の例は、これらのことをもっと明解に示しています。

```
.di x
¥¥¥¥
.br
.di
.x
```

GNU モード では、これは **** として表示されます。つまり、入力されたバックスラッシュの対‘****’はそれぞれ 1 つのバックスラッシュ‘****’に変換されます。 そして、結果として出力されるバックスラッ

シュは、再度読み込まれるときにはエスケープ文字としては解釈されません。

古典的な troff では、こうしたバックスラッシュは再度読み込まれるときには エスケープ文字として解釈されるので、最終的には 1 つのバックスラッシュ‘****’として出力されるでしょう。

出力可能な‘****’を得る正しい方法は エスケープシーケンス **\e** を使うことでしょう。 これは、ディバージョン内で使用されているかどうかに関わらず、現在のエスケープ文字を常に 1 つだけ出力します。さらにこれは GNU モードでも互換モードでも動作します。

ディバージョン内に、再度読み込まれたときに解釈したい エスケープシーケンスを格納するには、伝統的な 透過出力ファシリティ **\!** あるいは新しいエスケープシーケンス **\?** のどちらかが使用できます。

バグ

現在、groff システムのドキュメントは変更・刷新途上にあります。 それぞれのマニュアルには小さな不一致がある可能性があります。

警告 セクションは **troff**(1) に属しています。

作者

このドキュメントは groff、すなわち GNU roff 配布物の一部です。 Bernd Warken <bwarken@mayn.de> が書きました。

このドキュメントは、FDL (GNU Free Documentation License) バージョン 1.1 以降の条項のもとに配布されています。システムに FDL のコピーがあるはずですし、オンライン

http://www.gnu.org/copyleft/fdl.html

でも入手できます。

もともと、groff 言語拡張については **troff**(1) マニュアルページで管理されていました。このドキュメントは groff 言語拡張の不可欠な部分を含んではいますが、詳しい説明については、groff info ファイルの中にあります。

関連項目

groff 言語の主な情報源は **groff info**(1) ファイルです。

roff および groff システムを調べたり、さらなるドキュメントへの ポインタを得るには、**roff**(7) を参照してください。

フォーマッタプログラムについては **groff**(1) および **troff**(1) で説明されており、前もって定義されているグリフ名のすべては **groff_char**(7) で説明されています。

古くからある *troff* のドキュメントはオンライン

http://cm.belllabs.com/cm/cs/cstr.html

および

http://www.kohala.com/start/troff/

にあります。