

luatexja-ruby パッケージ

LuaTeX-ja プロジェクトチーム

2021-05-04 v0.52 (2021 年 9 月 18 日)

概要

luatexja-ruby パッケージは, LuaTeX-ja の機能を利用してルビの組版処理を行う追加パッケージである. LuaTeX, LuaTeX-ja の内部処理に割り込むことにより, 熟語ルビ中の行分割や, 行頭形・行中形・行末形の自動検出, また進入許容量の自動設定などを可能とした.

v0.3 より前とは親文字の高さの扱いが変わっている (`baseheight` キーを参照せよ) ほか, `rubypreintrusion`, `rubypostintrusion` パラメータの初期値が変わっているので注意すること.

目次

1	利用方法	2
1.1	用語	2
1.2	命令	2
1.3	グループの指定	5
2	注意点	6
3	実装について	9
3.1	進入量の計算	9
3.2	ノードの扱い	10
4	いくつかの例	12
5	『日本語組版処理の要件』 20120403 の例	14

1 りようほうほう 利用方法

パッケージ読み込みは、`\usepackage{luatexja-ruby}` で良く、必要ならば自動的に LuaTeX-ja 本体を読み込む。plain LuaTeX でのロードはまだサポートしておらず、 $\text{\LaTeX} 2_{\epsilon}$ のみサポートしている。

1.1 ようご 用語

「進入 (intrusion)」「突出 (protrusion)」という用語は、[pxrubrica](#) パッケージでの用法に準ずる。

進入あり：あかつきとあかつき暁の
進入なし：あかつきとあかつき暁の
突出あり：ちようしゆう聴衆
突出なし：ちようしゆう聴衆

なお、本パッケージでは親文字と直前・直後の文字の間に 0 でない和文処理グルー*1 がくることがも考慮しているため、「前後の文字への進入 (許容) 量」と「進入 (許容) 量」とは異なる可能性がある。この文書では次のように称する：

文字進入量 前後の文字ヘルビ文字が実際にかかる長さ。常に下線を引くことにする。

進入量 前後の文字、およびそれとルビの間の和文処理グルーにかかる長さ

多くの場合、和文処理グルーは 0 以上の長さのため、進入量は文字進入量以上である。

例えば次の例では、直前の文字「来」への前文字進入量は 0 であるが、前進入量は（和欧文間空白にかかる分まで含めるので）正である。

ほげほげふがふが
本来 foohoge においては……
本来 foohoge においては……

1.2 めいれい 命令

■`\ltjruby` ルビ出力用命令の本体。`\ruby` という別名を定義している。

```
\ltjruby[<option>]{親|文|字}{おや|も|じ}
```

のように親文字→ルビの順序で指定する。第 2・第 3 引数内の「|」はグループの区切りを表す。詳細は 1.3 小節を参照。

<option> には以下の内容を key-value リストで指定可能である。*<real>* は 10 進の実数値を表す。*<bool>* は真偽値 `true` (真) か `false` (偽) であり、値を省略したときには `true` の意味になる。

pre=*<real>* 前文字進入許容量をルビ全角単位で指定。負の長さを指定した場合は、ルビの状況や直前の文字に応じた自動指定を意味する。既定値は負（つまり、自動指定）。

post=*<real>* 同様に、後文字進入許容量を指定する。既定値は負（自動指定）。

mode 進入処理のモードを表す bit vector。下位 2 bit は、`pre` や `post` が負である場合にしか効力を発揮しない。既定値は $(0001)_2 = 1$ 。

bit 0 前後の文字への進入を無効にするならば 0、有効にするならば 1。

*1 JFM で指定されたグルーや、標準の和文間空白 (`kanjiskip`)、標準の和欧文間空白 (`xkanjiskip`)。

bit 1 前進入許容量 B と後進入許容量 A が異なった場合、そのまま処理する場合は 0、小さい方に揃えるならば 1.

bit 2–3 ルビ文字の突出量から実際の前・後進入量の計算方法を指定する. 詳しい計算方法については 3.1 小節を参照.

intrude_jfmgk= $\langle bool \rangle$ 進入量算出の際に、前後の JFM グルーの自然長を考慮するか否か. 既定値は真.

intrude_kanjiskip= $\langle bool \rangle$ 進入量算出の際に、ルビ前後に挿入される標準の和文間空白 ([kanjiskip](#)) の自然長を考慮するか否か. 既定値は真.

intrude_xkanjiskip= $\langle bool \rangle$ 進入量算出の際に、ルビ前後に挿入される標準の和欧文間空白 ([xkanjiskip](#)) の自然長を考慮するか否か. 既定値は真.

stretchruby={ $\langle left \rangle$ }{ $\langle middle \rangle$ }{ $\langle right \rangle$ } 親文字の合計長がルビ文字の合計長より長い時に、ルビ文字の前・間・後に入れる空白の割合であり、それぞれ 0–7 の自然数で指定する. 既定値は {1}{2}{1} である. $\langle left \rangle$ はルビ文字の先頭までの空き量、 $\langle middle \rangle$ はルビ文字間の空き量、 $\langle right \rangle$ はルビ文字の末尾からの空き量 (の比) を表す. 以下が例である.



```
1 \ltjruby[stretchruby=123,maxmargin=2]%
2 {○○○○}{◆◆}
```

stretch={ $\langle left \rangle$ }{ $\langle middle \rangle$ }{ $\langle right \rangle$ } 行中形でルビ文字の方が長い場合、親文字の前・中・後に入れる空白の割合. 既定値は {1}{2}{1} である. それ以外の代表的な値としては、例えば次のようなものがある.

親文字均等割禁止 {1}{0}{1} など $\langle middle \rangle$ を 0 にした値

前突出禁止 {0}{1}{1}

後突出禁止 {1}{1}{0}

stretchbol={ $\langle left \rangle$ }{ $\langle middle \rangle$ }{ $\langle right \rangle$ } 行頭形に対する stretch と同様の指定. 既定値は {0}{1}{1} である.

stretcheol={ $\langle left \rangle$ }{ $\langle middle \rangle$ }{ $\langle right \rangle$ } 行末形に対する stretch と同様の指定. 既定値は {1}{1}{0} である.

maxmargin= $\langle real \rangle$ 親文字の方がルビより長い時に、ルビの先頭と親文字の先頭、及びルビ末尾と親文字の末尾の間に許される最大の空白量. **親文字全角単位**で指定し、既定値は 0.5.

size= $\langle real \rangle$ ルビ文字の親文字に対する大きさ. 既定値は 0.5.

intergap= $\langle real \rangle$ ルビ文字と親文字との空きを親文字全角単位で指定. 既定値は 0.

rubysmash= $\langle bool \rangle$ ルビの高さを 0 にするか. 既定値は偽. 次が例である.

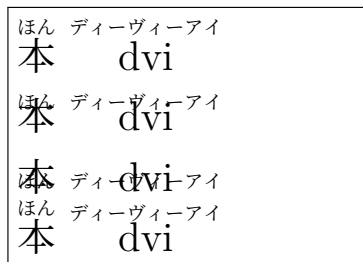


```
1 \vrule width 0pt height 2\zw depth 1\zw
2 \frame{\ltjruby[rubysmash=false]{本}{ほん}}\
3 \frame{\ltjruby[rubysmash=true]{本}{ほん}}\
4 \frame{\ltjruby[rubysmash=false,intergap=0.2]
5 {本}{ほん}}\
6 \frame{\ltjruby[rubysmash=true,intergap=0.2]
7 {本}{ほん}}\
8 \frame{\ltjruby[rubysmash=false,intergap=-1.5]
9 {本}{ほん}}
```

ybaseheight= $\langle real \rangle$ 非負の値が指定された場合、**縦組以外での**ルビの親文字の高さを全角高さの $\langle real \rangle$ 倍と強制的に固定する. 負の値が指定された場合は「固定しない」(すなわち、v0.3 以前の挙動と同じになる). 既定値は 0.88.

tbaseheight= $\langle real \rangle$ ybaseheight と同様だが、こちらは**縦組での**ルビの親文字の高さを指定する. 既定値は 0.5.

`baseheight=<real> ybaseheight, tbaseheight` を同時に指定したことと同義.



```
1 \noindent
2 \ltjruby[baseheight=0.88]{本}{ほん}\
3 \ltjruby[baseheight=0.88]{dvi}{ディーヴィーアイ}\
4 \ltjruby[baseheight=0.5]{本}{ほん}\
5 \ltjruby[baseheight=0.5]{dvi}{ディーヴィーアイ}\
6 \ltjruby[baseheight=0]{本}{ほん}\
7 \ltjruby[baseheight=0]{dvi}{ディーヴィーアイ}\
8 \ltjruby[baseheight=-1]{本}{ほん}\
9 \ltjruby[baseheight=-1]{dvi}{ディーヴィーアイ}
```

`kenten=<command>` 各文字につく圏点の出力命令を指定する. 既定値は「`\textbullet`」である.

`fontcmd=<command>` ルビ用のフォント切り替え命令を指定する. このキーの内容が実行された後に `\fontsize... \selectfont` が実行されるので, このキーの指定では最後に `\selectfont` を加える必要はない.

この `fontcmd` キーの内容は多くの回数実行される. 例えば, `luatexja-fontspec` パッケージを用いて OpenType フォントを用いる場合,

```
fontcmd=\addfontfeatures{Style=Ruby}
```

のようにしてルビ用字形を用いることが可能だが, 現在の実装ではタイプセットに時間がかかるようになる.

次の2つは, 以上で説明した複数のオプションを一度に設定するためのものである. 普通はこの2つのうちいずれかを設定するだけで足りるだろう.

`naka` 以下のオプションを同時に設定する. 主に中付きルビを組むときに用いる.

```
mode=1, stretch={1}{2}{1}, stretchruby={1}{2}{1}
```

`kata` 同様に, 肩付きルビ用に, 次を設定する.

```
mode=9, stretch={1}{2}{1}, stretchruby={0}{0}{1}
```

■`\ltjsetruby{<option>}` `<option>` の既定値を指定する. `luatexja-ruby` 読み込み時の値は各項目の所で既に説明してあるが, 次のようになっている.

```
pre=-1, post=-1, mode=1,
stretchruby={1}{2}{1}, stretch = {1}{2}{1},
stretchbol={0}{1}{1}, stretcheol={1}{1}{0},
maxmargin=0.5, size=0.5, intergap=0, rubysmash=false,
kenten=\textbullet, fontcmd=\relax, ybaseheight=0.88, tbaseheight=0.5,
intrude_jfmgk, intrude_kanjiskip, intrude_xkanjiskip
```

■`\ltjsetparamater` に追加されるキー


`rubypreintrusion={<chr_code>, <pre_int>}` 文字 `<chr_code>` に, その直後のルビによって掛けられるルビ文字列の最大長 (つまり, 前文字進入許容量) をルビ全角単位で指定.

v0.5 以降での変更点: もし $-a$ ($a > 0$) を指定した場合は, 実際の前文字進入許容量は a となるが, 文字 `<chr_code>` の前の JFM グルーに進入が発生した場合には, 前文字進入許容量が 0 でなかった場合は, の後の JFM グルーにはルビの進入は発生しない.

`rubypostintrusion={⟨chr_code⟩, ⟨post_int⟩}` 文字 `⟨chr_code⟩` に、その直前のルビによって掛けられるルビ文字列の最大長（つまり、後文字進入許容量）をルビ全角単位で指定。

v0.5 以降での変更点：もし $-a$ ($a > 0$) を指定した場合は、実際の後文字進入許容量は a となるが、実際の後文字進入量が 0 でなかった場合は、文字 `⟨chr_code⟩` の後の JFM グルーにはルビの進入は発生しない。

以下の文字は `rubypreintrusion`, `rubypostintrusion` とも初期値は 1 である：

ひらがな (U+3040–U+309F), カタカナ (U+30A0–U+30FF), 仮名補助など (U+1B000–U+1B16F), 分離禁止文字 


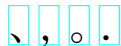
次に、以下の文字は `rubypreintrusion`, `rubypostintrusion` とも初期値は 0.5 である：

中点類 

また、以下の文字は `rubypreintrusion` のみ初期値は -1 である：

始め括弧類 , 「文字コード -1 の文字^{*2}」

さらに、以下の文字は `rubypostintrusion` のみ初期値は -1 である：

閉じ括弧類 , 句読点類 

以上で記述されていない `rubypreintrusion`, `rubypostintrusion` の初期値はすべて 0 である。

■ `\ltjkenten` 圏点を出力する命令で、`\kenten` という別名を定義している。

`\ltjkenten[⟨option⟩]{親文字}`

のように使用する。複数文字に圏点をつける場合でも、`\ltjruby` のように「|」を使って文字を区切る必要はない（`\kenten` 内では「|」は特別な意味を持たない）ことに注意してほしい。


圏点として出力される文字は `kenten` キーによって指定し、圏点自身の大きさは（ルビと同様に）`size` キーで指定する。

1.3 グループの指定^{している}

`\ltjruby[⟨option⟩]{親|文|字}{おや|も|じ}`

のように、`\ltjruby` の第 2・第 3 引数内の「|」はグループの区切りを表す。グループの数は両者で一致しなければならないが、`\ltjruby{紋章}{もん|しよう}` のようにはできない。

1 グループのみのルビ（単純グループルビ）はグループルビとして組まれる。そのため、もしモノルビを使いたければ、面倒でも



`1` の `\ltjruby{紋}{もん}\ltjruby{章}{しよう}` が

のように、複数回使用すること。また、全てのグループにおいて「ルビ文字列の長さは親文字列以下」^{*3}であれば、単純グループルビの並びとして扱われる。すなわち、次ページ冒頭の 2 行は全くの等価となる。

^{*2} 段落開始の `\parindent` 分インデントを表す。通常の Lua_TE_X-ja における指定では「文字コード -1 」は文中数式境界を表していることに注意。

^{*3} 実際には T_EX での長さの計算誤差 (2^{-16} pt の整数倍として計算していることによる) を考慮し、親文字全角の 1/1000 だけルビ文字列が長くなることを許容している。

\ltjruby{普通|車}{ふ|つう|しや}

\ltjruby{普}{ふ}\ltjruby{通}{つう}\ltjruby{車}{しや}

複数グループかつ上記の条件を満たさない場合は、所謂「可動グループルビ」であり、ルビの前後や各グループの切れ目で行分割が可能となる。例えば

…の\ltjruby{表|現|力}{ひよう|げん|りよく}は…

という入力からは得られる組版結果は、次のいずれかになる。

改行なし（行中形）	…の ^{ひようげんりよく} 表現力は…
直前で改行	^{ひようげんりよく} 表現力は…
	…の ^{ひようげんりよく} 表現力は…
	…の ^{ひようげんりよく} 表現力は…
直後に改行	…の ^{ひようげんりよく} 表現力

- これらの行分割によってペナルティは発生しない。
- 上記の例で見られるように、2ブロック以上をまとめて組むときは、全体を1つのグループルビのように組版する（JIS X 4051 と同様）。『日本語組版処理の要件』では、附属書 F に「熟語の構成、さらにその熟語の前後にくる文字の種類を考慮して配置する方法」として別の方法を解説しているが、こちらの方法は現時点ではサポートしていないので、面倒でも手動で pre, post などを使って頑張る欲しい。
- 実装方法の都合により、ルビの直前・直後・途中で2箇所以上の改行が起きる場合、例えば

…の^{ひようげんりよく}表現力 | ^{ひようげんりよく}表現力は… | ^{ひようげんりよく}表現力

などの組み方は想定していない。エラーが発生して止まることもあるし、エラーが発生しなくても正しく組まれない。

2 ちゆういてん 注意点

■前後からのルビ文字のはみ出し 1 「日本語組版処理の要件」の図 117*4のように、前後からのルビ文字のはみ出しが繋がらないようにする処理が組み込まれている。例えば、

りよう みささぎ 陵と 陵
りよう みささぎ 陵と 陵

- 1 \ltjruby{陵}{りよう}と\ltjruby{陵}{みささぎ}\
- 2 \ltjruby{陵}{りよう}と\ltjruby[pre=1]{陵}{みささぎ}

において、1行目右側の^{みささぎ}「陵」のルビが前の「と」にかかる量は次のように決まる：

1. 1回目の実行では、行分割前に^{りよう}「陵」の後文字進入量は前もって知ることはできない。そのため、^{りよう}「陵」は行中形で組まれるものと想定し、^{みささぎ}「陵」の前文字進入許容量は

$$\frac{0.5 \text{ zw}}{\text{元々の許容量}} - \frac{0.25 \text{ zw}}{\text{前のルビの後文字進入量 (行中形)}} = 0.25 \text{ zw}$$

となる。行分割後、^{りよう}「陵」の実際の後文字進入量は <jobname>.ltjruby ファイルに記述される。

*4 2020-08-11 版での番号。2012-04-03 版では図 3.82。

2. 2 回目以降の実行では、`<jobname>.ltjruby` ファイルに保存された「^{りょう}陵」の後文字進入量を用いて、「^{みささぎ}陵」の前文字進入許容量を次のように計算する：

$$\frac{0.5 \text{ zw}}{\text{元々の許容量}} - \frac{0.25 \text{ zw}}{\text{前のルビの後文字進入量(from .ltjruby)}} = 0.25 \text{ zw.}$$

`<jobname>.ltjruby` ファイルに保存する際、各 `\ltjruby` 命令の呼び出しを識別するキーが必要になるが、そのキーとしては単純に「何個目の `\ltjruby` 命令か」である。

なお、以上の処理は、1 行目と 2 行目を比較すれば分かるように、「^{みささぎ}陵」の前文字進入許容量指定 (pre) が自動になっている場合のみ実施される。

■**前後からのルビ文字のはみ出し 2** また、本パッケージの `v0.**`以降では、「日本語組版処理の要件」にある

後ろにくる終わり括弧類、句点類若しくは読点類、又は前にくる始め括弧類には、最大でルビ文字サイズの全角までルビ文字を掛けてもよい。この場合、後ろにくる終わり括弧類、句点類若しくは読点類の後ろの空き量、又は前にくる始め括弧類の前の空き量に掛けてはならない。

という処理も組み込まれており、`<jobname>.ltjruby` に「前後の和文処理グルーに正の量だけ進入したか」という情報を保存することによって実装されている。

■**段落末尾のルビ** 段落がルビで終わった場合、そのルビが行末形で組まれることはない。これは、段落の「本当の」末尾には `\penalty10000\parfillskip` があるためで、ルビ処理用に作った最後のグルー (3.2 小節の説明では g_2) が消去されないことによる。

`\parfillskip` の長さ (や、場合によっては `\rightskip`) を実測し、それによって処理を変えるのも可能だが、そのようなことはしなかった。段落がルビで終わることは普通ない (最低でも句点が続くだろう) と思うからである。

■**段落先頭のルビ** 同様に、段落先頭のルビは行頭形にはならない。pre が負 (つまり、自動指定) のとき、段落最初の `\parindent` 分への進入は可能である。ここ `\parindent` 分のインデントへの進入許容量は「文字 -1」に対する `rubypreintrusion` (既定値は 1, ルビ全角単位) と `\parindent` の長さのうち小さい方である。

<pre> 0 1 2 3 4 5 ^{みささぎ} 陵は…… ^{うけたまわ} 承り…… ^{みささぎ} 陵は…… </pre>	<pre> 1 \parindent1\zw\noindent 0 1 2 3 4 5\par 2 \ltjruby{^{みささぎ}陵}{^{みささぎ}は……}\par 3 \ltjruby{^{うけたまわ}承り}{^{うけたまわ}り……}\par 4 \parindent0.25\zw\ltjruby{^{みささぎ}陵}{^{みささぎ}は……} </pre>
--	---

■**和文処理グルーの伸縮** 現バージョンでは、進入量調整に和文処理グルーを考慮させる、

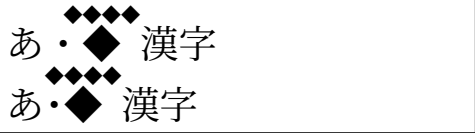
`intrude_jfmgk`, `intrude_kanjiskip`, `intrude_xkanjiskip` キー

の値が true (真) であった状況でも、考慮されるのは自然長の値のみである。そのため、行の調整処理が発生した場合は意図しない結果となる。

例えば、標準設定での中黒「・」の直後のルビからの中黒への進入許容量は

中黒「・」への前進入許容量はルビ全角の半分で、中黒本体の後の四分空きには進入可能

となっている。そのため、下の例の 1, 2 行目ともルビの前進入量は 0.5 zw となる。しかし、2 行目では詰め量の 0.5 zw がほとんど中黒周囲の四分空きで負担されるため、実際には「中黒本体にほぼ 0.5 zw が進入する」という望ましくない結果が得られている。

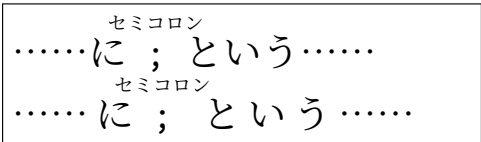


```

1 \leavevmode\hbox{あ・\ltjruby{◆}{◆◆◆◆}漢字}\
2 \hbox spread=0.5\zw{あ・\ltjruby{◆}{◆◆◆◆}漢字}

```

また、次の例では2行目では前後の文字進入量がルビ1字分と等しくなっているが、3行目のように1全角伸ばすという調整が行われた後は文字進入量が不揃いになってしまっている。これはもともと「;」には後側にのみ和欧文間空白が入ることと、3行目ではこの和欧文間空白が伸びているためである。



```

1 \leavevmode\hbox{……に%
2 \ltjruby{\texttt{;}}{セミコロン}という……}\
3 \hbox spread\zw{……に%
4 \ltjruby{\texttt{;}}{セミコロン}という……}

```


3 ^{じつそう}実装について

3.1 進入量の計算

ルビ文字を自然に組んだときの幅が親文字のそれより多い場合、ルビの前後への進入量は次のように決定される。

Step 1 前文字進入許容量 B_0 , 後進入文字許容量 A_0 の算出。

ルビ全角の長さを r とする。

- (a) `pre` の指定値が非負であった場合は、それに r を掛けたものを B_0 とする。
そうでなかった場合は、「ルビの直前の文字」に対する `rubypreintrusion` の値に r を掛けたものを B_0 とする*5。
- (b) `post` の指定値が非負であった場合は、それに r を掛けたものを A_0 とする。
そうでなかった場合は、「ルビの直後の文字」に対する `rubypostintrusion` の値にルビ全角の値を掛けたものを A_0 とする。
- (c) `mode` の最下位ビット (bit 0) が 0 であった場合は、 $B_0 \leftarrow 0, A \leftarrow 0$ とする。
- (d) もし 2 つ前の文字がルビで、その直後 (つまりいま処理しているルビから見れば直前) の文字へ a' だけの進入があった場合、現在のルビについて $B_0 \leftarrow \min(0, B_0 - a')$ とする。

Step 2 前進入許容量 B , 後進入許容量 A の算出。

- (a) まず $B \leftarrow B_0, A \leftarrow A_0$ とする。
- (b) ルビとその直前の文字の間に和文処理グルー g が挿入された場合、
 - g が JFM グルーの場合は、`intrude_jfmgk` が真の場合に、
 - g が標準の和文間空白 (`kanjiskip`) の場合は、`intrude_kanjiskip` が真の場合に、
 - g が標準の和欧文間空白 (`xkanjiskip`) の場合は、`intrude_xkanjiskip` が真の場合に、それぞれ g の自然長を B に加算する。
ルビとその直後の文字との間に和文処理グルーが挿入された場合も同様である。
- (c) `mode` の 2 番目のビット (bit 1) が 0 であった場合は、 $B, A \leftarrow \min(B, A)$ とする。

Step 3 実際の前進入量 b , 後進入量 a の計算。

ルビ文字の突出量を x , 親文字の文字数を $k + 1$, 親文字の前に入る空白量・間の空白量・後ろの空白量の比を $p : q : r$ とする。このとき、`mode` の bit 2, 3 の値によって b, a を次のように算出する：

$$00 \quad b = \min(B, xp/(p + kq + r)), \quad a = \min(A, xr/(p + kq + r))$$

$$01 \quad b = \min(B, x), \quad a = \min(A, \max(x - b, 0))$$

$$10 \quad a = \min(A, x), \quad b = \min(B, \max(x - a, 0))$$

11 $M = \min(B, A)$ とおく。もし $x \leq 2M$ ならば $b = a = x/2$ 。そうでなければ

$$b = \min\left(B, M + \frac{(x - 2M)p}{p + kq + r}\right), \quad a = \min\left(A, M + \frac{(x - 2M)r}{p + kq + r}\right).$$

組み方の具体例を実際に示す。例示のため、平仮名にはルビが 1 字まで、「立」にはルビを 0.5 字分までかけてよいことにしている。

00 は^{うつく}美しい は^{ちようしゆう}聴衆と は^{あかつき}暁立 は^{とうげ}峠立 は^{ちようしゆう}聴衆立
01 は^{うつく}美しい は^{ちようしゆう}聴衆と は^{あかつき}暁立 は^{とうげ}峠立 は^{ちようしゆう}聴衆立

*5 なお、「ルビの直前の文字」が段落最初の `\parindent` 分のインデントであった場合、 B_0 を、「文字コード -1 の文字」`rubypreintrusion` の値に r を掛けた値と `\parindent` のうち小さい方とする。

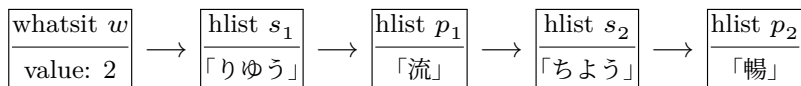
10 ^{うつく}は美しい ^{ちようしゆう}は聴衆と ^{あかつき}は暁立 ^{とうげ}は峠立 ^{ちようしゆう}は聴衆立
 11 ^{うつく}は美しい ^{ちようしゆう}は聴衆と ^{あかつき}は暁立 ^{とうげ}は峠立 ^{ちようしゆう}は聴衆立

3.2 ノードの扱い

次の例で内部実装の大まかな方法を説明する。

……^{りゆうちよう}を流暢に……₁ ……を\ltjruby{流|暢}{りゆう|ちよう}に……

1. \ltjruby コマンド自体は、一旦次の node list を値とする whatsit W を作って、現在の水平リストへと挿入する（必要ならば\leavevmode も実行）：



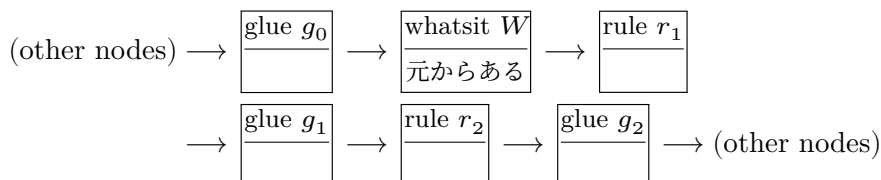
ここで、最初の w の値 2 は、ルビが 2 つのパーツ「流」「暢」からなっていることを表している。この値を cmp とおこう。 s_i 達の中の文字は既にルビの大きさである。

2. LuaTeX-ja の和文処理グルー挿入処理において、この whatsit W はまとめて「先頭が『流』、最後が『暢』であるような hbox を \unhbox で展開したもの」と扱われる。言い換えれば、ルビ部分を無視した単なる「流暢」という和文文字の並びとして扱われる*6。次のサンプルを参照

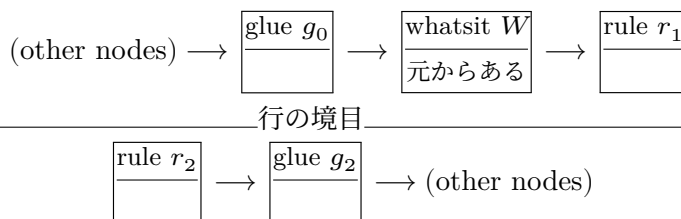
. A
. A ₁ \leavevmode\hbox{. }A\
₂ %↑xkanjiskip
₃ \ltjruby{. }{A
₄ %↑2分

3. 和文処理グルーの挿入が終わった後で、可動グループルビのためのノードの挿入に入る。

(a) W の前後に $2cmp + 1 = 5$ 個のノードが挿入され、 W の周辺は次のようなノード列になる。



(b) このようにノードを挿入する目的は、TeX の行分割処理自体に影響を加えずに可動グループルビを実現させることにある。



のようになったとしたら、「流」「暢」の間で行分割が起きた、ということがわかり、 g_i, r_i 達のノードを適切に置き換えればよい（後で詳しく説明する）。

(c) なお、 r_i 達の高さ・深さは組み上がった後のそれである。 g_i, r_i 達の幅は、図 1 に示したような対応に沿って算出する。例えばこの場合、行中形 n_5 に対して

$$g_0 + r_1 + g_2 + r_2 + g_2 = (3 - 0.25 \times 2) zw = 2.5 zw$$

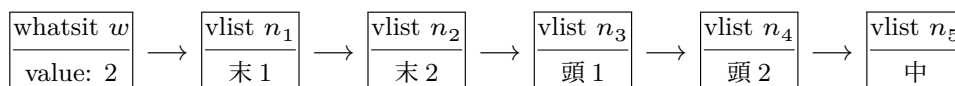
という方程式が立つ (zw は親文字全角の幅、進入量込)。 n_1 から n_5 まで計 5 本の方程式が立つが、これらは Gauß の消去法で解くことができ、 g_i, r_i 達の幅が求まる。

*6 「流」「暢」の間のグルーは既に入っている、と扱われる。

ノード	組み方	サンプル	対応するノード並び
n_1	行末 1 グループ	$\begin{array}{c} \text{りゅう} \\ \boxed{\text{を流}} \end{array}$	$g_0 \rightarrow W \rightarrow r_1$
n_2	行末 2 グループ	$\begin{array}{c} \text{りゅうちよう} \\ \boxed{\text{を流暢}} \end{array}$	$g_0 \rightarrow W \rightarrow r_1 \rightarrow g_2 \rightarrow r_2$
n_3	行頭 1 グループ	$\begin{array}{c} \text{ちよう} \\ \boxed{\text{暢に}} \end{array}$	$r_2 \rightarrow g_2$
n_4	行頭 2 グループ	$\begin{array}{c} \text{りゅうちよう} \\ \boxed{\text{流暢に}} \end{array}$	$W \rightarrow r_1 \rightarrow g_2 \rightarrow r_2 \rightarrow g_2$
n_5	行中	$\begin{array}{c} \text{りゅうちよう} \\ \boxed{\text{を流暢に}} \end{array}$	$g_0 \rightarrow W \rightarrow r_1 \rightarrow g_2 \rightarrow r_2 \rightarrow g_2$

図 1 ルビの組み方と対応するノード並び

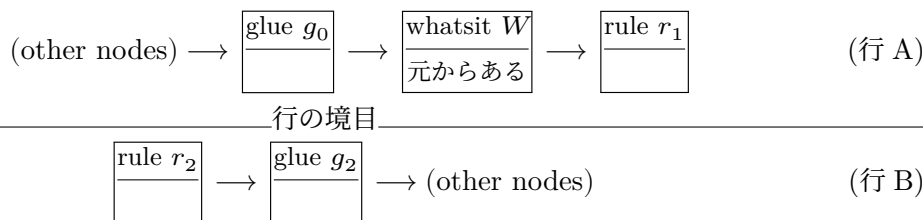
(d) また、ルビ処理を統括している whatsit W の値も



に置き換えておく。

4. LuaTeX の行分割処理を普通に行う。
5. 行分割の結果に従って、 g_i, r_i 達を適切に置換する。

例えば行分割の結果



のようになったとしよう。

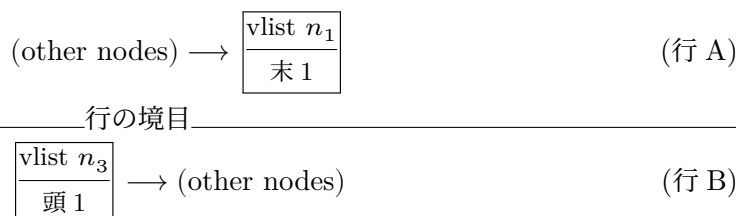
- (a) 処理は段落の上の行から順番に行われる。行 A の処理がまわってきたとしよう。
- (b) 行 A の先頭から順番に眺めていく。すると「whatsit W 由来」のノード、 g_0, W, r_1 が見つかり、行 A はここで終わっている。

まず、行 A の hbox の中身から whatsit W を消去（リストから取り除くだけで、 W のメモリを解放するわけではない）する。 $g_0, (W), r_1$ というノードの並びは、「行末 1 グループ」 n_1 に対応しているので、 g_0, r_1 を行 A から除去・メモリ解放し、代わりに n_1 を行 A の中身に追加する。

- (c) 次に行 B の処理にうつる。行 A でルビの処理は完了していない（2 グループのルビなのにまだ 1 グループしか使っていない）ので、「whatsit W 由来」のノードがいくつか行 B 内に残っているはずである。

案の定、 r_2, g_2 というノード列が見つかった。これは「行頭 1 グループ」 n_3 に対応しているので、 r_2, g_2 を行 B から除去・メモリ解放し、代わりに n_3 を行 B の中身に挿入する。

- (d) これで 2 グループとも使い切ったことになるので、隔離しておいた W を、（使われなかった n_2, n_4, n_5 などと共に）メモリ解放する。結果として次のようになった：



4 いくつかの例

ゴールデンゲートブリッジ
 ああああ黄金橋いうえおかきくけこあ
ゴールデンゲートブリッジ ゴールデンゲートブリッジ
 黄金橋いうえおかきくけこあ黄金橋
ゴールデンゲートブリッジ
 いうえおかきくけこあ黄金橋いうえおか
ゴールデンゲートブリッジ
 きくけこあ黄金橋いうえおかきくけこあ
ゴールデンゲートブリッジ
 黄金橋いうえおかきくけこ

baseheight=0.88 での例

ディーヴィーアイ ほーげーふーが
 ふあいる dvi ファイル oo 漢字
ディーヴィーアイ ほーげーふーが
 ファイル dvi ふあいる oo 漢字
ふがふがふがふが ふがふがふがふが ふがふがふがふが ふがふがふがふが
 ああ (ほげ) ほげ 「ほげ」 ほげ…

こうづ
 あ国府津いうえおかきくけこあ国府津
こうづ
 いうえおかきくけこあ国府津いうえおか
こうづ
 きくけこあ国府津いうえおかきくけこあ
こうづ
 国府津いうえおかきくけこあ国府津いう
こうづ
 えおかきくけこ

こうづ
 あ●●◆いうえおかきくけこあ●●
こうづ
 ◆いうえおかきくけこあ●●◆いうえお
こうづ
 かきくけこあ●●◆いうえおかきくけこ
こうづ
 イあ●●◆いうえおかきくけこあ●●
こうづ
 ◆いうえおかきくけこウあ●●◆いう
こうづ
 えおかきくけこエあ●●◆いうえおか
こうづ
 きくけこあ●●◆いうえおかきくけこあ
こうづ
 ●●◆いうえおかきくけこ

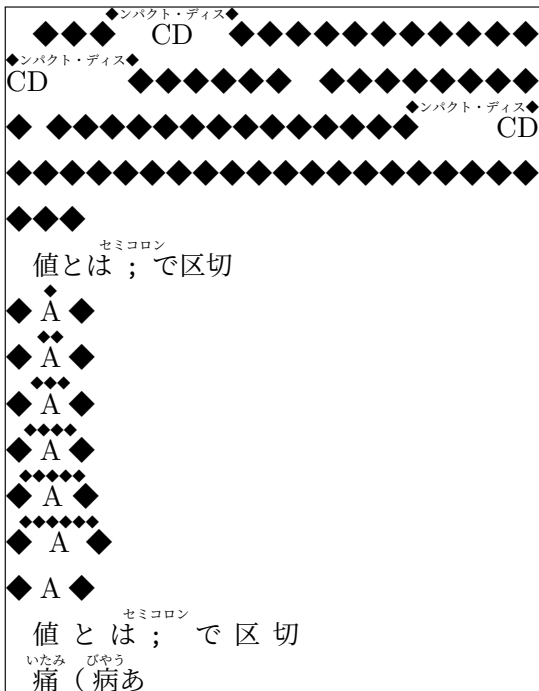
こうづ
 あ●●□いうえおかきくけこあ●●□い
こうづ
 うえおかきくけこあ●●□いうえおかきく
こうづ
 けこあ●●□いうえおかきくけこイあ●●
こうづ
 □いうえおかきくけこあ●●□いうえお
こうづ
 かきくけこウあ●●□いうえおかきくけこ
こうづ
 エあ●●□いうえおかきくけこあ●●□い
こうづ
 うえおかきくけこあ●●□いうえおかきく
こうづ
 けこ

いよう いよう
 あ異様いうえくけあ異様いうえくけこあ
いよう いよう
 異様いうえくけこあ異様いうえくけこイあ
いよう
 異様いうえおかきくけこ

いよう
 あ□■□いうえおかきくけこうえおかきくけこ
いよう
 あ□■□いうえおかきくけこうえおかきくけこあ
いよう
 □□□□いう□おかきくけこうえおかきくけこあ□
いよう
 □□□■□いう□おかきくけこうえおかきくけこあ□■
いよう
 □□□□いう□おかきくけこうえおかきくけこあ□■□
いよう
 いうえおかきくけこ

標準 うけたまわ 又 そ 承る にわか 疎 にわか は俄勉強 にわか 後俄勉強 あかつき は暁には俄に きゆうけいちゆう 休憩中 しちようちゆう かつ視聴中
 肩つき うけたまわ 又 そ 承る にわか 疎 にわか は俄勉強 にわか 後俄勉強 あかつき は暁には俄に きゆうけいちゆう 休憩中 しちようちゆう かつ視聴中

- 1 `{\ltjsetruby{stretch=101}}%` 親文字均等割り禁止
- 2 `\ruby{休|憩|中}{きゆう|けい|ちゆう}かつ\ruby{視|聴|中}{し|ちよう|ちゆう}%`
- 3 `}\quad`
- 4 `\textgt{\ruby{勉|強}{べん|きよう}と%`
- 5 `\ruby[fontcmd=\mcfamily]{勉|強}{べん|きよう}}\quad%` ルビは明朝体
- 6 `\ruby{コギト・エルゴ・スム}{Cogito ergo sum}\quad` % 欧文空白は伸長しない
- 7 `\ruby[size=1]{Cogito ergo sum}{コギト・エルゴ・スム}%` 欧文空白は伸長しない



5 『日本語組版処理の要件』 20120403 の例

■3.3.1 節

3.49	君子は和して同ぜず
3.50	人に誨えて倦まず
3.51	鬼門の方角を凝視する
3.52	鬼門の方角を凝視する
3.53	茅場町 茅場町
3.54	紫陽花 埧塙 田舎
3.55	模型 顧客 境界面 避難所
3.56	編集者 editor

- ```
1 \obeylines
2 3.49 \ruby{君|子}{くん|し}は\ruby{和}{わ}して\ruby{同}{どう}ぜず
3 3.50 \ruby{人}{ひと}に\ruby{誨}{おし}えて\ruby{倦}{う}まず
4 % モノルビ．面倒でも複数回の実行が必要
5 3.51 \ruby{鬼}{き}\ruby{門}{もん}の\ruby{方}{ほう}\ruby{角}{かく}を%
6 \ruby{凝}{ぎよう}\ruby{視}{し}する
7 % 熟語ルビ
8 3.52 \ruby{鬼|門}{き|もん}の\ruby{方|角}{ほう|かく}を\ruby{凝|視}{ぎよう|し}する
9 3.53 \ruby{茅場町}{かやばちよう}\quad\ruby{茅|場}{かや|ば}\ruby{町}{ちよう}
10 % 熟字訓
11 3.54 \ruby{紫陽花}{あじさい}\quad\ruby{埧塙}{るつぼ}\quad\ruby{田舎}{いなか}
12 % グループルビ
13 3.55 \ruby{模型}{モデル}\quad\ruby{顧客}{クライアント}\quad%
14 \ruby{境界面}{インターフェース}\quad\ruby{避難所}{アジール}
15 3.56 \ruby{編集者}{editor}\quad\ruby{編集者}{エディター}
```

### ■3.3.3 節

|      |              |
|------|--------------|
| 3.58 | に幟を に幟を 韋編三絶 |
| 3.59 | に幟を          |
| 3.60 | 韋編三絶 韋編三絶    |

- ```
1 \obeylines
2 3.58 に\ruby{幟}{のぼり}を\quad に\ruby[kata]{幟}{のぼり}を\quad%
3     \ruby{韋}{い}\ruby{編}{へん}\ruby{三}{さん}\ruby{絶}{ぜつ}
4 % 三分ルビ．JY3/mc/mc は本文書のプレアンブルで独自に定義
5 3.59 に\ruby[fontcmd=\kanjifamily{mc}\kanjiseris{mc}]{幟}{のぼり}を
6 % ルビ文字を小さくする
7 3.60 {\Large%
8     \ruby{韋}{い}\ruby{編}{へん}\ruby{三}{さん}\ruby{絶}{ぜつ}\quad% 比較用
9     \ltjsetruby{size=0.375}% 0.5 -> 0.375
10    \ruby{韋}{い}\ruby{編}{へん}\ruby{三}{さん}\ruby{絶}{ぜつ}}
```

■3.3.4 節 3.61 図（両側ルビ）は未サポートにより省略

■3.3.5 節 モノルビ

3.62 の^{やく}葯に
 3.63 版面の^ち地に 版面の^ち地に
 3.65 の^{とりで}砦に
 3.66 上 の^{しゆん}句に 後^{しゆん}句に
 3.66 下 の^{しゆん}句又 後^{しゆん}句又

- 1 \obeylines
- 2 3.62 の\ruby{葯}{やく}に
- 3 3.63 版面の\ruby{地}{ち}に\quad 版面の\ruby[kata]{地}{ち}に
- 4 % 横組肩つきはしないが，現状では縦組未サポートだし，仕方ないね
- 5 3.65 の\ruby{砦}{とりで}に
- 6 {\ltjsetruby{kata}%
- 7 3.66上 の\ruby{句}{しゆん}に\quad 後\ruby{句}{しゆん}に
- 8 3.66下 の\ruby{句}{しゆん}又\quad 後\ruby{句}{しゆん}又}

■3.3.6 節 グループルビ

3.67 は^{コーデツクス}冊子体と
 3.68 ^{モデル}模型 ^{ライセンス}利用許諾
 3.69 ^{モデル}模型 ^{ライセンス}利用許諾
 3.70 ^ピなげきの^エ聖母^タ像 ^ピなげきの^エ聖母^タ像
 3.71 ^{クライアント}顧客 ^{インターフェース}境界面
 3.72 ^{クライアント}顧客 ^{インターフェース}境界面

- 1 \obeylines
- 2 3.67 は\ruby{冊子体}{コーデツクス}と
- 3 3.68 \ruby{模型}{モデル}\quad \ruby{利用許諾}{ライセンス}
- 4 % 両端を揃える流儀
- 5 3.69 {\ltjsetruby{stretchruby=010}%
- 6 \ruby{模型}{モデル}\quad \ruby{利用許諾}{ライセンス}}
- 7 % ルビが極端に短い場合
- 8 3.70 \ruby{なげきの聖母像}{ピエタ}\quad% ルビ全角まで許容
- 9 \ruby[maxmargin=0.75]{なげきの聖母像}{ピエタ}\\% ルビ全角1.5倍まで
- 10 % ルビが長い場合
- 11 3.71 \ruby{顧客}{クライアント}\quad \ruby{境界面}{インターフェース}
- 12 3.72 {\ltjsetruby{stretch=010, stretchbol=010, stretcheol=010}% はみ出さない流儀
- 13 \ruby{顧客}{クライアント}\quad \ruby{境界面}{インターフェース}}

■3.3.7 節 熟語ルビ

3.73	^{きゆう} 杞憂	^{いふ} 畏怖	^{きゆう} 杞憂	^{いふ} 畏怖	
3.74	^{りゆうぎ} の流儀を	^{むじよう} の無常を	^{じようじゆ} の成就を	^{もんしよう} の紋章を	^{しようちよう} の象徴を
3.75	^{りゆうぎ} の流儀を	^{むじよう} の無常を	^{じようじゆ} の成就を	^{もんしよう} の紋章を	^{しようちよう} の象徴を
3.76×	^{りゆうぎ} の流儀を	^{むじよう} の無常を			
3.77	ああああああの ^{りゆうぎ} 流儀		……等の ^む 無常を		

```

1 \obeylines
2 3.73 \ruby{杞|憂}{き|ゆう}\quad \ruby{畏|怖}{い|ふ}\quad%
3     \ruby[kata]{杞|憂}{き|ゆう}\quad \ruby[kata]{畏|怖}{い|ふ}
4 3.74 の\ruby{流|儀}{りゆう|ぎ}を\quad   の\ruby{無|常}{む|じよう}を\quad%
5     の\ruby{成|就}{じよう|じゆ}を\quad   の\ruby{紋|章}{もん|しよう}を\quad%
6     の\ruby{象|徴}{しよう|ちよう}を
7 % 熟語の構成を考慮した方法は現行ではできない。
8 % 行分割はできるのだが、「他の漢字にルビ全角までかかって良い」は難しい。
9 3.75 {\ltjsetruby{kata}%
10     の\ruby{流|儀}{りゆう|ぎ}を\quad   の\ruby{無|常}{む|じよう}を\quad%
11     の\ruby{成|就}{じよう|じゆ}を\quad   の\ruby{紋|章}{もん|しよう}を\quad%
12     の\ruby{象|徴}{しよう|ちよう}を}
13 % モノルビ配置。望ましくない
14 3.76× の\ruby{流|儀}{りゆう|ぎ}\ruby{儀}{ぎ}を\quad   の\ruby{無}{む}\ruby{常}{じよう}を\quad%
15
16 3.77 {\ltjsetruby{stretchbol=121, stretcheol=121}% 行頭・行末揃えず
17     \hbox{\vrule\box{\hsize=10\zw\noindent\kern.75\zw
18     ああああああの\ruby{流|儀}{りゆう|ぎ}がある。}\vrule}\quad%
19     \hbox{\vrule\box{\hsize=5\zw   ……等の\ruby{無|常}{む|じよう}を}\vrule}}

```

■3.3.8 節 ルビはみ出し

3.78	^{ひと} 人は死して ^な 名を ^{のこ} 残す
3.79	漢字の部首には ^{へん} 偏・ ^{かんむり} 冠・ ^{きやく} 脚・ ^{つくり} 旁がある
3.79	漢字の部首には ^{へん} 偏, ^{かんむり} 冠, ^{きやく} 脚, ^{つくり} 旁がある

```

1 \obeylines
2 3.78 \ruby{人}{ひと}は\ruby{死}{し}して\ruby{名}{な}を\ruby{残}{のこ}す
3 3.79 漢字の部首には\ruby{偏}{へん}・\ruby{冠}{かんむり}・\ruby{脚}{きやく}・%
4     \ruby{旁}{つくり}がある
5 3.79 漢字の部首には\ruby{偏}{へん}, \ruby{冠}{かんむり}, \ruby{脚}{きやく}, %
6     \ruby{旁}{つくり}がある

```


3.79 この^{うわさ}噂の好きな人は^{ふところ}懐ぐあいもよく、^{ひのき}檜を

3.80 漢字の部首には「^{へん}偏」「^{かんむり}冠」「^{きやく}脚」「^{つくり}旁」がある

3.80 この^{うわさ}噂好きな人は^{ふところ}懐 具合もよく、^{ひのき}檜材を

3.81× に^{あかつきおもむき}暁の趣を（良くない例）

3.82 に^{あかつき おもむき}暁 趣を

3.83 この^{うわさ}噂の好きな人は^{ふところ}懐ぐあいもよく、^{ひのき}檜を

3.83 この^{うわさ}噂好きな人は^{ふところ}懐 具合もよく、^{ひのき}檜材を

3.84 この^{うわさ}噂の好きな人は^{ふところ}懐ぐあいもよく、^{ひのき}檜を

3.84 この^{うわさ}噂好きな人は^{ふところ}懐 具合もよく、^{ひのき}檜材を

```

1 \obeylines
2 3.79 この\ruby{\噂}{うわさ}の好きな人は\ruby{懐}{ふところ}ぐあいもよく、\ruby{檜}{ひのき}を
3 3.80 漢字の部首には「\ruby{偏}{へん}」「\ruby{冠}{かんむり}」「\ruby{脚}{きやく}」%
4 「\ruby{旁}{つくり}」がある
5 3.80 この\ruby{\噂}{うわさ}好きな人は\ruby{懐}{ふところ}具合もよく、\ruby{檜}{ひのき}材を
6 3.81× に\ruby{暁}{あかつき}の\kern-1\zw の\ruby{趣}{おもむき}を（良くない例）
7 3.82 に\ruby{暁}{あかつき}の\ruby{趣}{おもむき}を
8
9 % 漢字・ひらがな・カタカナにルビを2分まで掛けても良い流儀
10 {%
11 \catcode`\<12%
12 \makeatletter\count@="3040\loop\relax\ifnum \count@<"30FF%
13 \ltjsetparameter{rubypreintrusion={\the\count@,0.5}, %
14 rubypostintrusion={\the\count@,0.5}}%
15 \advance\count@1 \repeat
16 \ltjsetparameter{rubypostintrusion={`好,0.5}}
17 \ltjsetparameter{rubypostintrusion={`具,0.5}}
18 \ltjsetparameter{rubypostintrusion={`材,0.5}}
19 3.83 この\ruby{\噂}{うわさ}の好きな人は\ruby{懐}{ふところ}ぐあいもよく、\ruby{檜}{ひのき}を
20 3.83 この\ruby{\噂}{うわさ}好きな人は\ruby{懐}{ふところ}具合もよく、\ruby{檜}{ひのき}材を
21 }
22 % 平仮名にもルビを掛けない流儀
23 {\catcode`\<12%
24 \makeatletter\count@="3040\loop\relax\ifnum \count@<"30A0%
25 \ltjsetparameter{rubypreintrusion={\the\count@,0}, %
26 rubypostintrusion={\the\count@,0}}%
27 \advance\count@1 \repeat
28 3.84 この\ruby{\噂}{うわさ}の好きな人は\ruby{懐}{ふところ}ぐあいもよく、\ruby{檜}{ひのき}を
29 3.84 この\ruby{\噂}{うわさ}好きな人は\ruby{懐}{ふところ}具合もよく、\ruby{檜}{ひのき}材を
30 }

```

		……の徑 <small>こみち</small>
3.85	を…… 徑を…… <small>こみち</small>	……の ……の徑 <small>こみち</small>
3.86	を…… 徑を…… <small>こみち</small>	……の ……の徑 <small>こみち</small>
3.87	等…… 飾り等…… <small>アクセサリー</small>	……共飾り <small>アクセサリー</small> ……共

```

1 \obeylines
2 3.85\ {\ltjsetruby{stretchbol=121, stretcheol=121}% 行頭・行末揃えず
3 \hbox{\vrule\ vbox{\hsize=15\zw
4 \hskip9.5\zw……の\ruby{徑}{こみち}を……%
5 \hskip9\zw……の\ruby{徑}{こみち}を……}\vrule}}
6 3.86\ \hbox{\vrule\ vbox{\hsize=15\zw
7 \hskip10\zw……の\ruby{徑}{こみち}を……%
8 \hskip9\zw……の\ruby{徑}{こみち}を……}\vrule}}
9 3.87\ \hbox{\vrule\ vbox{\hsize=15\zw
10 \hskip8\zw……共\ruby{飾り}{アクセサリー}等……%
11 \hskip9\zw……共\ruby{飾り}{アクセサリー}等……}\vrule}}

```

■**圈点の例（常用漢字表前書きより）** この表は、法令、公用文書、新聞、雑誌、放送など、一般の社会生活において現代の国語を書き表す場合の漢字使用の自安を示すものである。

「\」の大きさを親文字の0.33倍にした例 この表は、法令、公用文書、新聞、雑誌、放送など、一般の社会生活において現代の国語を書き表す場合の漢字使用の自安を示すものである。