LuaTEX-ja の近況 2020

北川 弘典

2020-11-21

Online.tex 2020

目次

■LuaTEX-ja の JFM 機能拡張

- JFM 機能
- 源ノ書体の n 倍角ダッシュ

■LuaTEX-ja のルビ

- カタカナへのルビ掛け
- ルビの高さの不揃い
- 和文処理グルーの扱い

全体的に小ネタ × n という感覚.

注意

このスライドのように，
タイトルが白抜きになっているものは
発表では原則飛ばします．

スライドはどこかにアップロードの**予定**

TEX でのフォント指定

LATEX では NFSS2 や fontspec でラップされるが，
最終的には次へ帰着：

欧文フォント	`\font \cs=... [at 42pt]`
横組用和文フォント	`\jfont\cs=... [at 42pt]`
縦組用和文フォント	`\tfont\cs=... [at 42pt]`

本節は ... の指定内容の話.

非 Unicode TEX でのフォント指定

「フォント」「組み方」ごとに異なる TFM を指定.

例　otf パッケージ（upLATEX 版）の和文フォント

$$\mathtt{up}\underbrace{\mathtt{brsg}}_{B}\underbrace{\mathtt{exp}}_{E}\underbrace{\mathtt{minr}}_{S}\underbrace{\mathtt{n}}_{J}\text{-}\underbrace{\mathtt{h}}_{D}\mathtt{.tfm}$$

		候補		
B	ぶら下げの有無[1]	brsg（有）	“”（無）	
E	横/縦組専用仮名	exp（専用）	nml（両用）	ruby
S	ファミリ・ウェイト	計 7 通り		
J	JIS2004 字形[1]	n (JIS2004)	“” (JIS1990)	
D	組方向	h（横）	v（縦）	

[1] ruby（ルビ用仮名）についてはぶら下げ・JIS2004 字形はなし.

LuaTEX-ja での和文フォント指定

LuaTEX-ja の JFM は「組み方」部分だけを司る．

例　前ページの例と同様のことをやると……

$$\underbrace{\texttt{file:HiraMinProN-W3.otf}}_{S\text{（実フォント）}}\texttt{;}\underbrace{\underbrace{\texttt{+hkna}}_{E}\texttt{;}\underbrace{\texttt{+jp04}}_{J}}_{\text{OpenType 機能}}\texttt{;jfm=}\underbrace{\texttt{ujisbrsg}}_{B,\ D}$$ [2]

[2] jfm-ujisbrsg.lua は準備されていない．自分で作る必要あり．

LuaTEX-ja での和文フォント指定

LuaTEX-ja の JFM は「組み方」部分だけを司る．

例　前ページの例と同様のことをやると……

$$\underbrace{\texttt{file:HiraMinProN-W3.otf}}_{S\text{（実フォント）}}\texttt{;}\underbrace{\underbrace{\texttt{+hkna}}_{E}\texttt{;}\underbrace{\texttt{+jp04}}_{J}}_{\text{OpenType 機能}}\texttt{;jfm=}\underbrace{\texttt{ujisbrsg}}_{B,\ D}$$ [2]

ぶら下げ有無などの「組み方」がたくさんある場合，
「その分だけ JFM を準備する」以外の方法は？

[2] jfm-ujisbrsg.lua は準備されていない．自分で作る必要あり．

「組み方」が多い場合：従来の方針

LuaTEX-ja の JFM ファイルは Lua スクリプト
→読み込み時にグローバル変数で制御可能

```
local t = { ... }
if hoge then
    if hoge.burasage then
        t[1].width = ...
    end
end
luatexja.jfont.define_jfm(t)
```

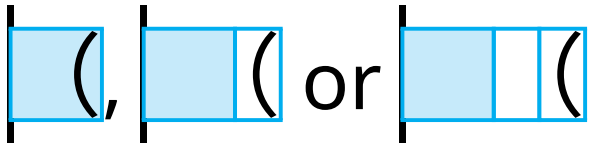
hoge.burasage の値に沿って JFM の中身を変更

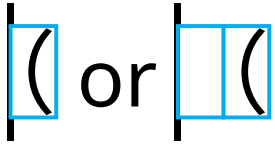
「組み方」が多い場合：従来の方針

LuaTEX-ja の JFM ファイルは Lua スクリプト
→読み込み時にグローバル変数で制御可能

```
local t = { ... }
if hoge then
    if hoge.burasage then
        t[1].width = ...
    end
end
luatexja.jfont.define_jfm(t)
```

例　jlreq クラス[3]

- 

- 

（折り返し行頭）
- 

（ぶら下げ）

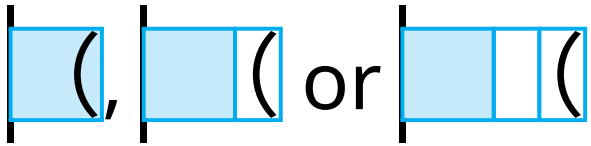
[3] 阿部紀行氏，https://github.com/abenori/jlreq

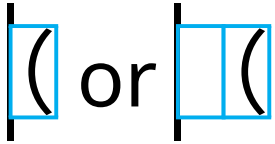
「組み方」が多い場合：従来の方針

LuaTEX-ja の JFM ファイルは Lua スクリプト
→読み込み時にグローバル変数で制御可能

```
local t = { ... }
if hoge then
    if hoge.burasage then
        t[1].width = ...
    end
end
luatexja.jfont.define_jfm(t)
```

例　jlreq クラス

- 

- 

（折り返し行頭）
- 

（ぶら下げ）

これでは「例示のためこの段落だけ……」は対応不可

∵　同じ JFM ファイルの読み込みは一回だけ

JFM 機能

file:HogeMin-Bold.otf;...;jfm=ujis

/{foo,-bar,hoge=2,abc=baz,...};...

JFM 機能の指定

- **JFM 機能の指定が違えば「違う JFM」とみなす**

→ JFM ファイルを都度読み込み

例 \A ≠ \B = \C ≠ \D ≠ \A

```
\jfont\A=...:jfm=ujis;...
\jfont\B=...:jfm=ujis/{a=1,b=2,+foo};...
\jfont\C=...:jfm=ujis/{b=2,a=1,foo};...
\jfont\D=...:jfm=ujis/{a,hoge=z,fuga=false};...
```

順序違い等
正規化の後
比較

JFM 機能

file:HogeMin-Bold.otf;...;jfm=ujis

/{foo,-bar,hoge=2,abc=baz,...};...

JFM 機能の指定

- JFM 機能の指定内容は，JFM ファイルから luatexja.jfont.jfm_feature として参照可能.

  上の指定からは次のテーブルが得られる：

```
{ ["foo"]=true, ["bar"]=false, ["hoge"]="2",
  ["abc"]="baz", ...                          }
```

PEG 構文

`ltj-jfont.lua` 参照.

$$
\begin{aligned}
\langle \text{field} \rangle &\leftarrow (\text{'/\{\};,=' 以外の文字})+ \\
\langle \text{assign} \rangle &\leftarrow \langle \text{field} \rangle \text{ '=' } \langle \text{field} \rangle \\
\langle \text{switch} \rangle &\leftarrow \text{'–'} \langle \text{field} \rangle \ / \ \text{'+'?} \ \langle \text{field} \rangle \\
\langle \text{expr} \rangle &\leftarrow (\langle \text{assign} \rangle \ / \ \langle \text{switch} \rangle) \text{ ','}* \\
\langle \text{list} \rangle &\leftarrow \text{'\{'} \langle \text{expr} \rangle* \text{ '\}'} \ / \ \langle \text{expr} \rangle* \\
\langle \text{jfm name} \rangle &\leftarrow (\text{'/\{\};,=' 以外の文字})+ \\
\langle \text{jfm spec} \rangle &\leftarrow \langle \text{jfm name} \rangle (\text{'/'} \langle \text{list} \rangle)?
\end{aligned}
$$

テーブル化と「違う JFM」判定

1. JFM 機能の指定（コンマ区切り）を Lua テーブル t に変換
2. t の各エントリを ⟨key⟩ の昇順で再文字列化したものをコンマ区切りでつなげた文字列を s とする
   （s は「正規化」された JFM 機能の指定）
3. ⟨jfm name⟩/{s}（と `jfmvar` キー[3]の値の組）で判定

元々の JFM 機能の指定	t[⟨key⟩]	再文字列化
⟨key⟩, +⟨key⟩, ⟨key⟩=`true`	true (bool)	⟨key⟩
−⟨key⟩, ⟨key⟩=`false`	false (bool)	−⟨key⟩
⟨key⟩=⟨value⟩	“⟨value⟩” (string)	⟨key⟩=⟨value⟩

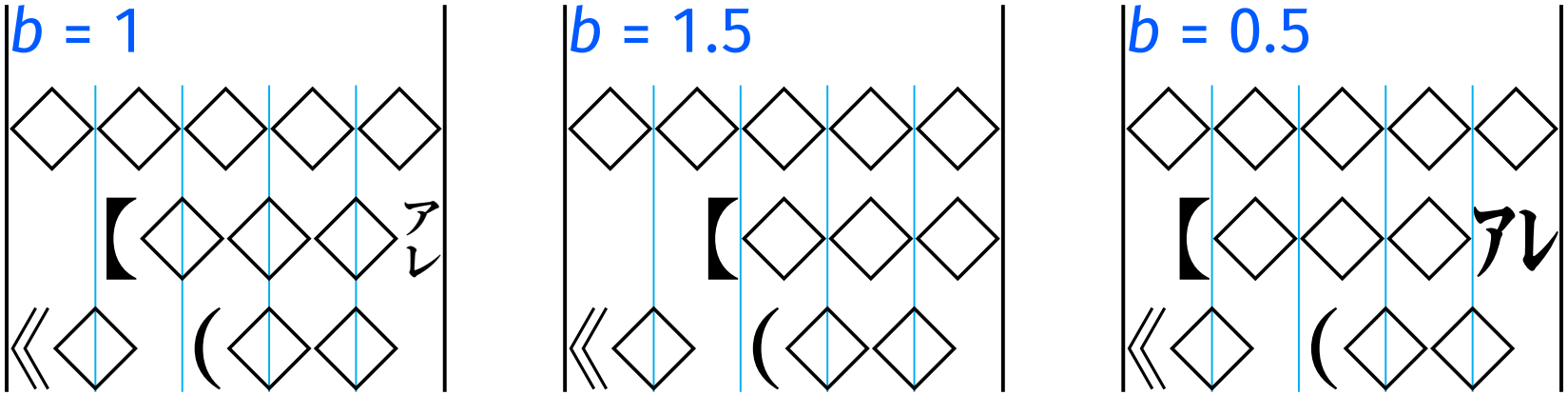
[3] `\jfont\cs=...:jfm=ujis/{...};jfmvar=fuga;+jp04;...`

例


```
\jfont\test=HaranoAjiMincho-Bold.otf:
        jfm=testf/{newline_ob=a,newpar_ob=b}
```

- a　折り返し行頭の始め括弧類の字下げ量
- b　段落冒頭の始め括弧類の字下げ量

例　$a = 0$ のとき



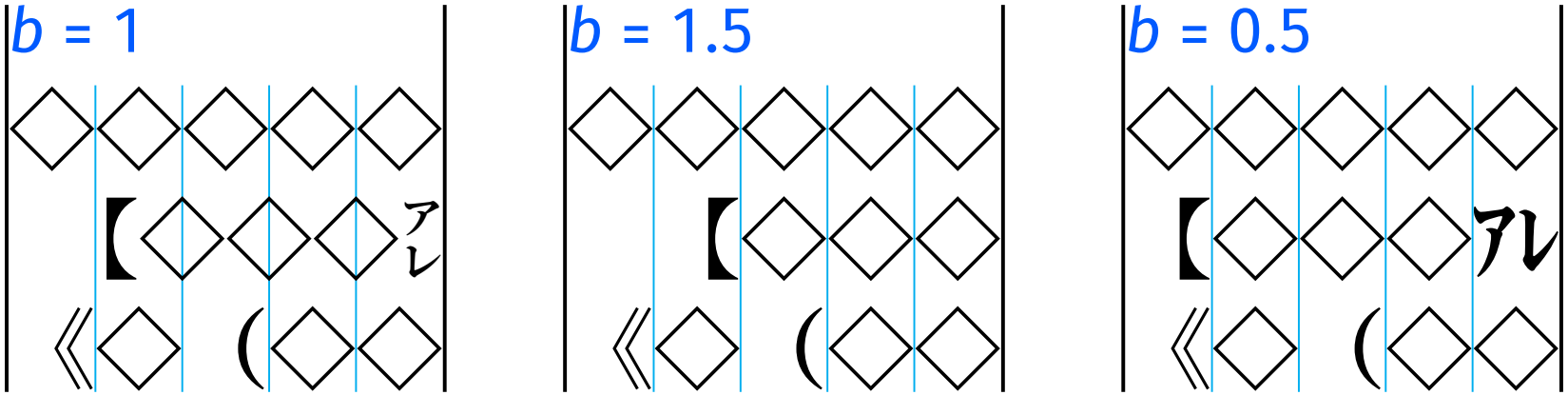
例


```
\jfont\test=HaranoAjiMincho-Bold.otf:
        jfm=testf/{newline_ob=a,newpar_ob=b}
```

a　折り返し行頭の始め括弧類の字下げ量

b　段落冒頭の始め括弧類の字下げ量

例　$a = 0.5$ のとき



他にも ListLee 氏による中国語用 JFM がある．

jfm-testf.lua の内容

```
... -- (jfm-ujis.luaの内容)
local jf = luatexja.jfont.jfm_feature
if jf then
    local npo = jf.newpar_ob and tonumber(jf.newpar_ob)
    if npo then
        t[199].kern = t[199].kern or {}   --「段落頭」は文字クラス 199
        t[199].kern[1] = jf.newpar_ob - 1 -- 始め括弧類は文字クラス 1
    end
    local nlo = jf.newline_ob and tonumber(jf.newline_ob)
    if nlo then
        -- 始め括弧類の幅・オフセットをずらす
        -- 他文字クラスとのグルーを nlo だけ減らす
    end
end
```

例 2

ListLee 氏による中国語用 JFM[4].

jfm-zh_CN.lua（簡体字），jfm-zh_TW.lua（繁体字）

vert	縦組用	
quanjiao	全角式	自（动、“调整”：中英文间。空）白
banjiao	半角式	自(动、“调整”:中英文间。空)白
kaiming	開明式	自(动、“调整”:中英文间。空)白
hwcl	「：」「；」を「文字幅半角・後空き半角」に（jfm-zh_CN.lua，縦組時のみ）	

[4] https://github.com/tanukihee/ChineseJFM

源ノ書体の *n* 倍角ダッシュ

Unicode 列	無指定		日本語[5]	
<2014 2014>	——	[2E3A]	——	[F4DA3]
<2014 2014 2014>	———	[2E3B]	———	[F4DA4]
<2015 2015>	——	[F4DA3]	——	[F4DA3]
<2015 2015 2015>	———	[F4DA4]	———	[F4DA4]
<2E3A>	——	[2E3A]	——	[F4DA3]
<2E3B>	———	[2E3B]	———	[F4DA4]

[ ] 内の値は LuaTEX[6]が割り当てた文字コード

→**和文の *n* 倍角ダッシュは私用領域内に位置**

[5] Language tag: JAN, script tag: hani. locl も有効化.

[6] 厳密には luaotfload パッケージ.

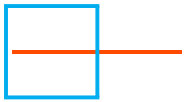
和文 *n* 倍角ダッシュの文字クラス

和文 *n* 倍角ダッシュのグリフは，現行の実装では文字クラス 0（普通の和文文字）固定

- 合字のグリフから文字クラスを決める　→**無理**
  - 私用領域内への割り当て方は環境非依存か？
  - 源ノ書体は Adobe-Japan1 に準拠せず
- 合字の構成要素から文字クラスを決める→**未実装**

  たとえば「ffl」合字を得る方法は 4 つ考えられるが……

  f + f + l,　f + fl,　ff + l,　ffl

→**このままだと**  **とはみ出してしまう**

現行の文字クラスなどの決定手順

c: OpenType 機能による置換後のグリフの文字コード

c_0: 置換前のグリフの「文字コード」（単一のグリフ由来のとき）
そうでなければ，c と同じ

■和文文字

文字クラス	c で JFM を参照→ 0 なら c_0 で参照
{pre,post}breakpenalty kcatcode jaxspmode	いずれも c_0 でパラメータを参照

■欧文文字

{pre,post}breakpenalty, alxspmode とも c で参照[7].

[7]合字なら合字の構成要素まで遡る.

対症療法：round_threshold 指定


```
local t = { -- jfm-ujis.lua（標準の横組用 JFM）
  [0] = {  -- 文字クラス 0（普通の和文文字）限定
    width = 1.0, round_threshold = 0.01, ...
  }, ...
}; ...; luatexja.jfont.define_jfm(t)
```

フォントサイズ比のグリフ幅を W としたとき，
$n = W/1.0$ が整数から ± 0.01 の範囲内にあった場合は
「JFM で指定された幅は $n \cdot 1.0$」と扱う．

対症療法：round_threshold 指定


```
local t = { -- jfm-ujis.lua（標準の横組用 JFM）
  [0] = {  -- 文字クラス 0（普通の和文文字）限定
    width = 1.0, round_threshold = 0.01, ...
  }, ...
}; ...; luatexja.jfont.define_jfm(t)
```

グリフ		幅（全角比）	対応
——	[2E3A]	1.675	専用文字クラス
———	[2E3B]	2.47	専用文字クラス
——	[F4DA3]	2.0	**本機能で対応**
———	[F4DA4]	3.0	**本機能で対応**

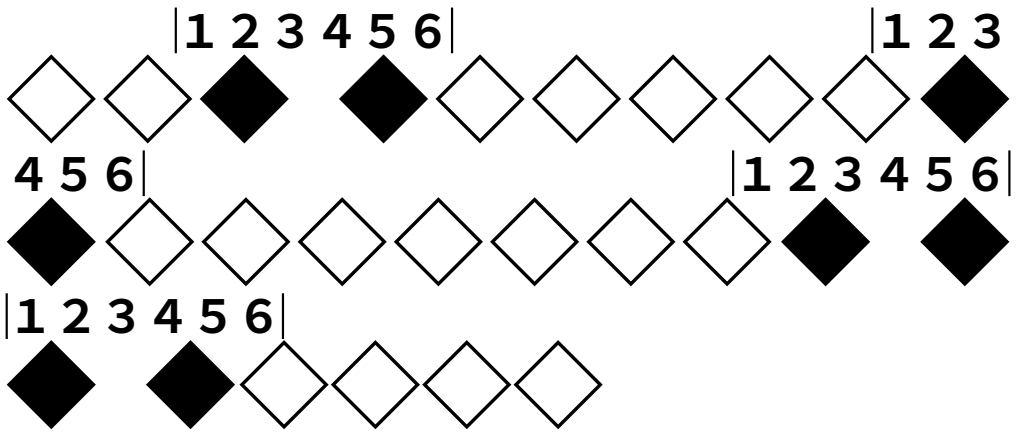
LuaTEX-ja でのルビ (luatexja-ruby)

- 前後の文字に応じて自動でルビ掛け許容量を計算

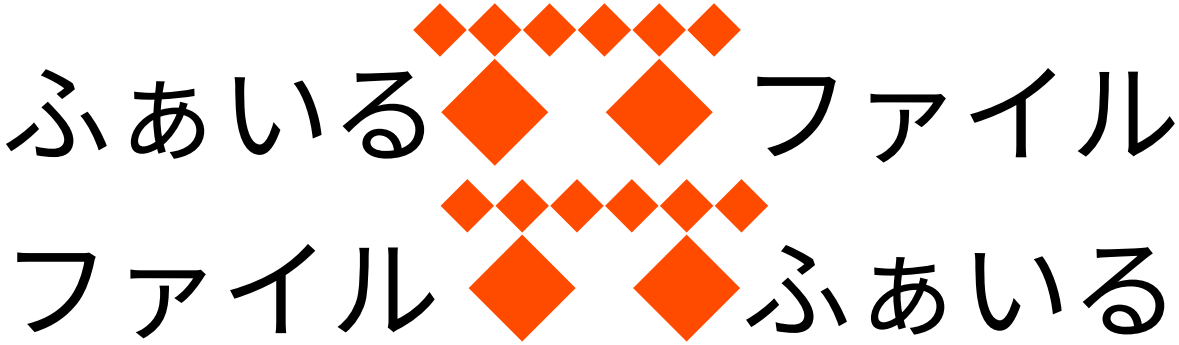
りょうしょう　　りょうしょう
を了 承した　は了 承 事項

りょうしょう　　りょうしょう
を了承した　　は了 承 事項　進入優先例

- 自動で行頭形・行中形・行末形を切り替え



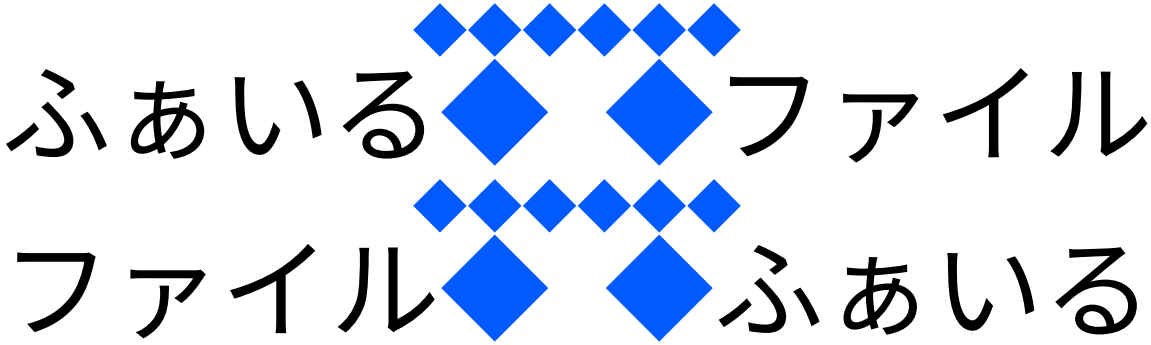
問題点 1：カタカナへのルビ掛け



- JIS X 4051 ではカタカナと漢字は同じ文字クラス
  → JIS X 4051 ではカタカナへのルビ掛け不可.
- 「日本語組版処理の要件」[8]ではカタカナへもルビ掛け可能

[8] https://www.w3.org/TR/jlreq/

問題点 1：カタカナへのルビ掛け→解決



- JIS X 4051 ではカタカナと漢字は同じ文字クラス
  → JIS X 4051 ではカタカナへのルビ掛け不可.
- 「日本語組版処理の要件」[8]ではカタカナへもルビ掛け可能
  → LuaTEX-ja 20201005.0 以降ではこちらを標準

[8] https://www.w3.org/TR/jlreq/

問題点 2：ルビの高さの不揃い

◇xy◇◇pH◇◇漢◇

従来はルビと親文字の間隔 (intergap) のみ指定可

→手動で intergap を指定すれば良いが…….

◇xy◇◇pH◇◇漢◇

0.323 0.138

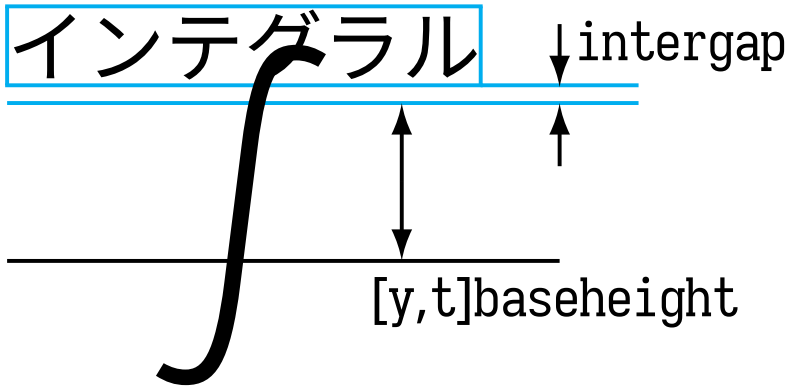
問題点 2：ルビの高さの不揃い→解決

◆◆◆◆ ◆◆◆◆ ◆◆◆◆

◇xy◇◇pH◇◇漢◇

LuaTEX-ja 20201005.0 で，親文字の高さを固定する ybaseheight, tbaseheight キーを追加.

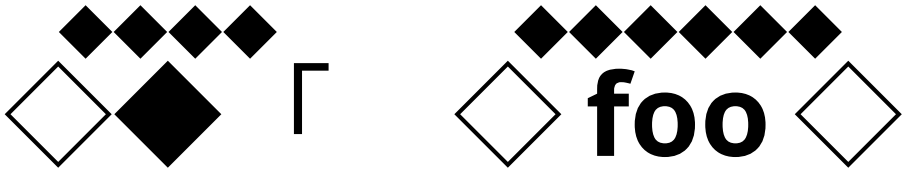
- 負数は「固定しない」
- 標準値：ybaseheight=0.88, tbaseheight=0.5



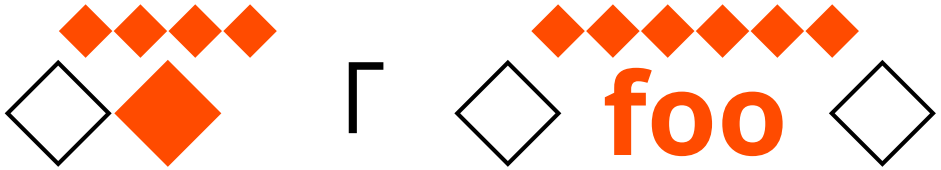

```
\ruby
[ybaseheight=1.0,intergap=0.1]
{$\displaystyle\int$}
{インテグラル}
```

問題点 3：和文処理グルー

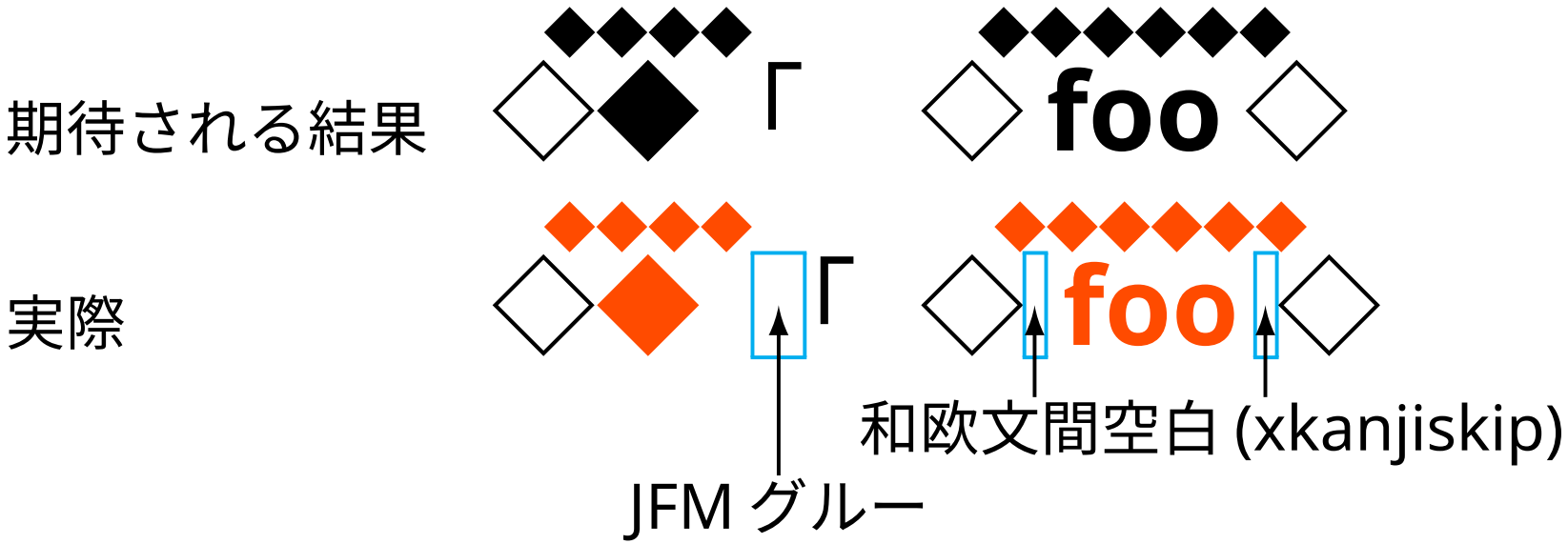
期待される結果



実際

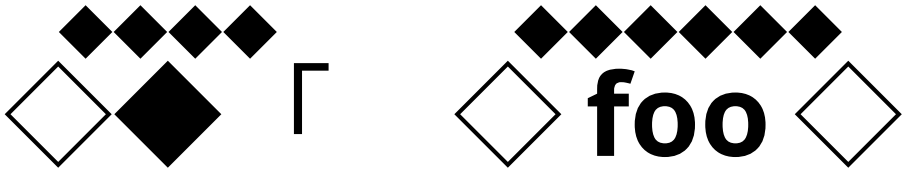


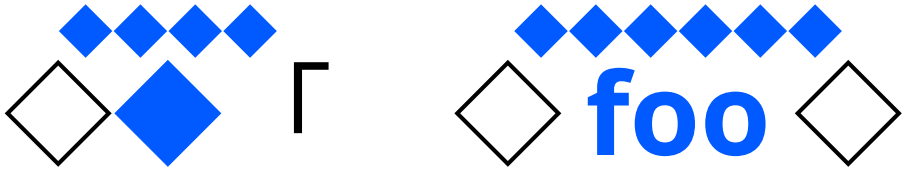
問題点 3：和文処理グルー



- 従来は前後の和文処理グルーは無感知.
  - 進入許容量は前後との文字のみで計算.
  - 「直後の開き括弧類との間の二分空きへのルビ掛け」などは実現できなかった.
  - xkanjiskip が前後にあった場合は余計に空いた.

問題点 3：和文処理グルー

期待される結果 

20201030.0 

- 20201030.0 以降は和文処理グルーの自然長も進入許容量に含められるようにした．

  `intrude_jfmgk` JFM 由来グルーの自然長を算入するか
  `intrude_kanjiskip` 標準の和文間空白 (kanjiskip) の……
  `intrude_xkanjiskip` xkanjiskip の……

  標準ではどれも真．まだ荒削り

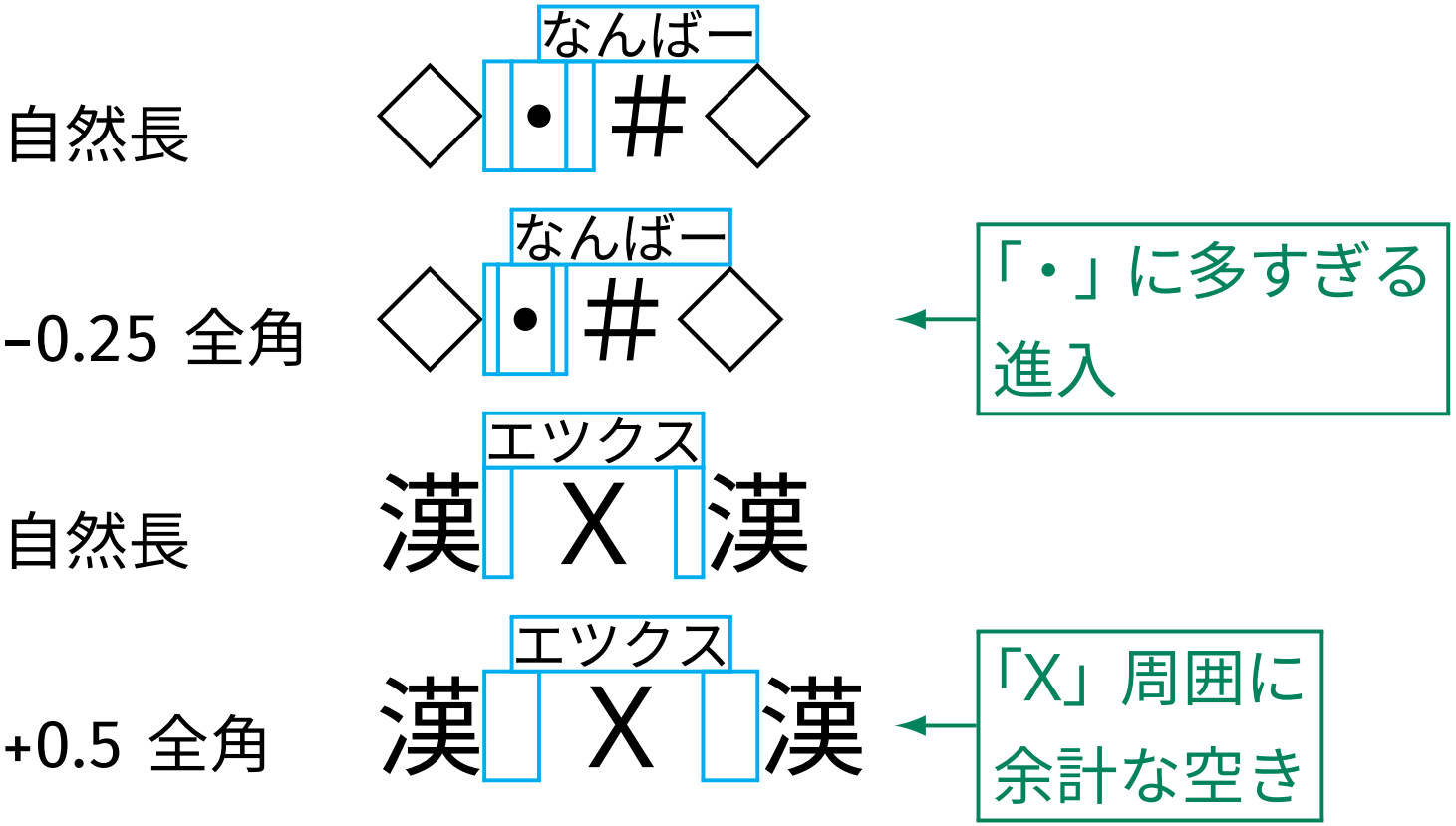
今後の課題：グルーの伸縮

自然長	なんばー ◇・＃◇
-0.25 全角	なんばー ◇・＃◇
自然長	エツクス 漢 X 漢
+0.5 全角	エツクス 漢 X 漢

「・」に多すぎる進入

「X」周囲に余計な空き

今後の課題：グルーの伸縮



進入許容量の計算では自然長のみ考慮

→**和文処理グルーの伸縮分だけおかしくなる**

→**行分割との絡みもあり，悩み中**（気力が……）

■LuaTEX-ja の JFM 機能拡張

- JFM 機能
- 源ノ書体の *n* 倍角ダッシュ

■LuaTEX-ja のルビ

- カタカナへのルビ掛け
- ルビの高さの不揃い
- 和文処理グルーの扱い