InDesignにおける「GSUBフィーチャ文字化け」の解明

小形克宏／直井靖／丸山邦朋

2012年9月28日　rev. 0.9.7

INDEX

0.1　はじめに

本文書は、アドビシステムズによるDTPアプリケーション、InDesignで作成したデータを他のアプリにコピー・アンド・ペースト（以下、コピペ）したり、フォントを変更した際に発生する「GSUBフィーチャ文字化け」を解説したものである。

OpenTypeフォントに内蔵された、文字の表示に変化をもたらす機能を総称して「OpenTypeフィーチャ」という。GSUBフィーチャはそのうちグリフの置換に関するもので、本稿で取り上げるのは、これが原因となって引き起こされる文字化けである（→1.1.2 GSUBテーブルとGSUBフィーチャ）。

本稿はある程度InDesignの操作に習熟した人を対象にしている。もしもこの分野の知識があまりない場合、別に公開されている『InDesignデータ→電子書籍で字形の変化する文字』[1)]から読むことをお勧めする。本文書は、この文書を技術的に敷衍したものだ。

第1部、及び第2部第1章の原稿執筆を小形克宏（@ogwata）が、同じくレイアウトと組版を丸山邦朋（@monokano）が、第2部掲載のリスト作成と全体にわたる監修（正確には「技術的な原作」といえる）を直井靖（@moji_memo）と丸山邦朋が担当した。また、田嶋淳、深沢英次の各氏をはじめ、多くの人から適確なアドバイスを賜ったことを感謝とともに付記する。

本稿には、InDesignの開発元であるアドビシステムズは一切関与していないことをお断りしておく。本文書の内容について同社に問い合わせることは、どうかおやめいただきたい。

最後に、本文中に丸括弧で括り、矢印をつけたのは、参照すべき見出し、図版である。用語はなるべく公知のものを使用したが、必要に応じて著者達が考案したものもある。たとえばタイトルにある「GSUBフィーチャ文字化け」がそうだが、他にも「選択置換」「字形パネル・ダブルクリック」等々がある。その場合それと分かるように断った。

---

1)　田嶋淳「InDesignデータ → 電子書籍で字形の変化する文字」、2012年6月6日
(http://www.pubridge.jp/wp/wp-content/uploads/2012/06/ChangeGlyphCharactor_ID2ebook.pdf)

0.2　定義

本稿では、以下のように独自の用法で使っている言葉がある。

- **InDesign**：本稿で単に「InDesign」と呼ぶ場合、それはInDesign CS5を指す。基本的に他のバージョンでも通用するはずだが、すべての動作確認をしたわけではない。

- **OpenTypeフォント**：現在流通しているOpenTypeフォントは、2種類に分けられる。本稿ではそのうち、CIDフォントから発展したCFFテーブルを内蔵するタイプに限定して論じる。このタイプはDTPでよく使用されており、アドビシステムズ、モリサワ、大日本スクリーン製造などから発売されている。もう1つのタイプはTrueTypeフォントから発展した、glyfテーブルを内蔵するものだ。その代表的なものとしてMS明朝、メイリオ等がある。双方は共に“OpenType Specification”（以下、仕様。URLは「参考文献」を参照）に基づくが、内部構造は大きく異なっている。

- **異体字**：一般には漢字の3要素「形・音・義」のうち、読みと字義が同じだが形のみ違うものを異体字という。本稿では、GSUBフィーチャの適用などによりグリフが置換された文字全般をいう。漢字に限定しないでこの語を使う。

- **グリフ**：文字の分類として、よく抽象部分と具象部分の2つに分けることがおこなわれる。たとえばUnicode Standardの定義ではグリフ（glyph）は後者、つまり具体的な文字の形を指す[2)]。本稿においては「CIDで表現された個々の文字」と定義する（これはUnicode Standardとも合致する）。この定義では、Unicode符号位置によって表現される文字より、ずっと微細な違いを区別する。

- **文字化け**：一般には意図せず符号位置が変わってしまうことをいう。本稿では、加えて意図せずCID（グリフ）が変化することも指す。この結果、一般には文字化けと言わないような微細な変化も、本稿ではそれとして扱っている。

2)　Unicode Consortium, “2.2 Unicode Design Principles” in: *The Unicode Standard 6.1*, 2011 (http://www.unicode.org/versions/Unicode6.1.0/ch02.pdf).

第1部
GSUBフィーチャ文字化けの前提となる解説

この部では、本稿のテーマである「GSUBフィーチャ文字化け」を理解するため必要な事項を説明する。すでにこの分野の知識がある人は第2部にすすまれたい。また、この部での解説は必要最小限に留まるので、より詳細は専門書にあたっていただきたい（付録参照）。

1.1 OpenTypeフォントにおける異体字の機能

GSUBフィーチャ文字化けは、OpenTypeフォントが内蔵する情報を、InDesignが使用することにより発生する。したがってこれを理解するには、InDesignだけでなくOpenTypeフォントの知識が必要となる。本章ではまずこの部分を解説する。

OpenTypeフォントはマイクロソフトとアドビシステムズが共同開発した高機能フォントフォーマットである。その特徴は、(1) クロス・プラットフォーム、(2) プリンタフォントが不要、(3) Unicode対応、(4) 高度な多言語・組版機能の4つにまとめられる。

こうした特徴を実現するため、OpenTypeフォントは多種多様なテーブルを内蔵している。各テーブルは設定値を格納した一種のデータベースと言える。OpenTypeフォントはそうしたデータベースの集合体であり、InDesignはそれらのテーブルを参照することで高度な組版を実現している。そのうち本稿に関係するのがcmapテーブルとGSUBテーブルである。

1.1.1　cmapテーブルとCMapファイル

cmapテーブルはUnicode符号位置とCIDの対応を記述したものだ。CIDについての詳細は「文字を表わす一意な番号としてCIDを使う際の注意点」[3]に譲るが、本稿と関連するポイントを列挙すると以下のようになる。

- CID（Character IDentifier＝文字識別番号）とは、アドビシステムズが策定したフォント用グリフセットで使われる、0から始まる一連の番号である。
- 同社のグリフセット名は3つの要素から成り立つ。例えばAdobe-Japan1-6を例にとると、「Adobe」の部分が「登録者（Registry）」、「Japan1」は「配列（Ordering）、「6」は「追補番号（Supplement）」である。Adobe-Japan1-0（8,283文字）からはじまり、順次拡張を重ねて現在1-6（23,058文字）が最新版である（図1）[4]。

	CID番号の範囲	収録グリフ数	発行日
Adobe-Japan1-0	0〜8283	8,284字	1992年
Adobe-Japan1-1	8284〜8358	8,359字	1993年
Adobe-Japan1-2	8359〜8719	8,720字	1993年
Adobe-Japan1-3	8720〜9353	9,354字	1998年9月
Adobe-Japan1-4	9354〜15443	15,444字	2000年2月21日
Adobe-Japan1-5	15444〜20316	20,317字	2002年9月
Adobe-Japan1-6	20317〜23057	23,058字	2004年3月5日

図1：Adobe-Japan1とそのバージョン（“The Adobe-Japan1-6 Character Collection” p.1）。

- cmapテーブルには、Unicode符号位置とCIDの対応が記述されている。フォントは、OSやアプリケーションから符号位置を受け取ると、cmapテーブルを参照してCIDを取得し、そのCIDによりグリフを引き出してOSやアプリケーションに渡す。つまりcmapテーブルは符号位置とグリフをつなぐ、最も古典的なフォントの役割を担っている。
- ただし、OpenTypeフォントにおいて使用可能な文字は、cmapテーブルに記述されている文字（つまりUnicode符号位置をもつ文字）が全てではない（図2）。

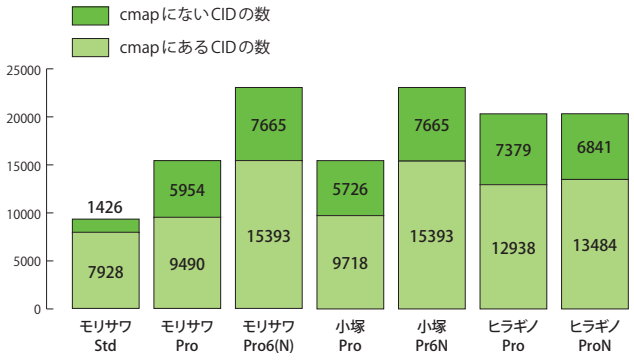

図2：各社のOpenTypeフォントにおける内蔵グリフの内訳
（作成：丸山邦朋）

3)　小形克宏「文字を表わす一意な番号としてCIDを使う際の注意点」2012年5月15日
(http://www.pubridge.jp/info/20120515/)

4)　アドビシステムズ「Adobe-Japan1-6文字コレクションに対応する日本語OpenType®フォントについて」
(http://www.adobe.com/jp/support/type/aj1-6.html)

- cmapテーブルに記述されてない文字、つまり本稿における異体字を制御しているのがGSUBテーブルである（次章参照）。
- 本稿で扱う「GSUBフィーチャ文字化け」は、この異体字が引き起こすものである。
- Unicode符号位置とCIDの対応（つまりcmapテーブルの内容）は、アドビシステムズによって一元管理されており、同社サイトにて最新版が「CMapファイル」として公開されている[5]。
- それらを参照することで、ベンダーによってcmapテーブルの内容はバラバラというわけでなく、Adobe-Japan1準拠フォントの場合おおむね5種類に分類できることが分かる（図3）。

ⓐ	UniJIS-UCS2系	アドビシステムズが開発したAdobe-Japan1-4対応版	アップル以外	リュウミン Pro
ⓑ	UniJIS-UTF32系	アドビシステムズがⓐをAdobe-Japan1-5に対応させたもの。のちに1-6に対応	アップル以外	小塚明朝Pro、リュウミンPr5、リュウミンPr6、イワタ明朝Pr6
ⓒ	UniJIS2004-UTF32系	アドビシステムズがⓑをJIS X 0213:2004に対応させたもの	アップル以外	リュウミン Pr6N、小塚明朝Pr6N、イワタ明朝Pr6N
ⓓ	UniJISX0213-UTF32系	アップルによるⓐの拡張。 のちにAdobe-Japan1-5に対応。	アップルのみ	ヒラギノ明朝Pro
ⓔ	UniJISX02132004-UTF32系	ⓓをJIS X 0213:2004に対応させたもの	アップルのみ	ヒラギノ明朝ProN

図3：CMapファイルの分類。赤字は第2部「InDesignの字形パネルから入力した文字を他のアプリにコピペしたときに化ける例」でサンプルにしたフォント。

5) AdobeSystems, "*CMap Resources*" (http://sourceforge.net/adobe/cmap/home/Home/).

- さらにその変遷を系統図の形にまとめた資料が有志により作成、公開されている（図4）。

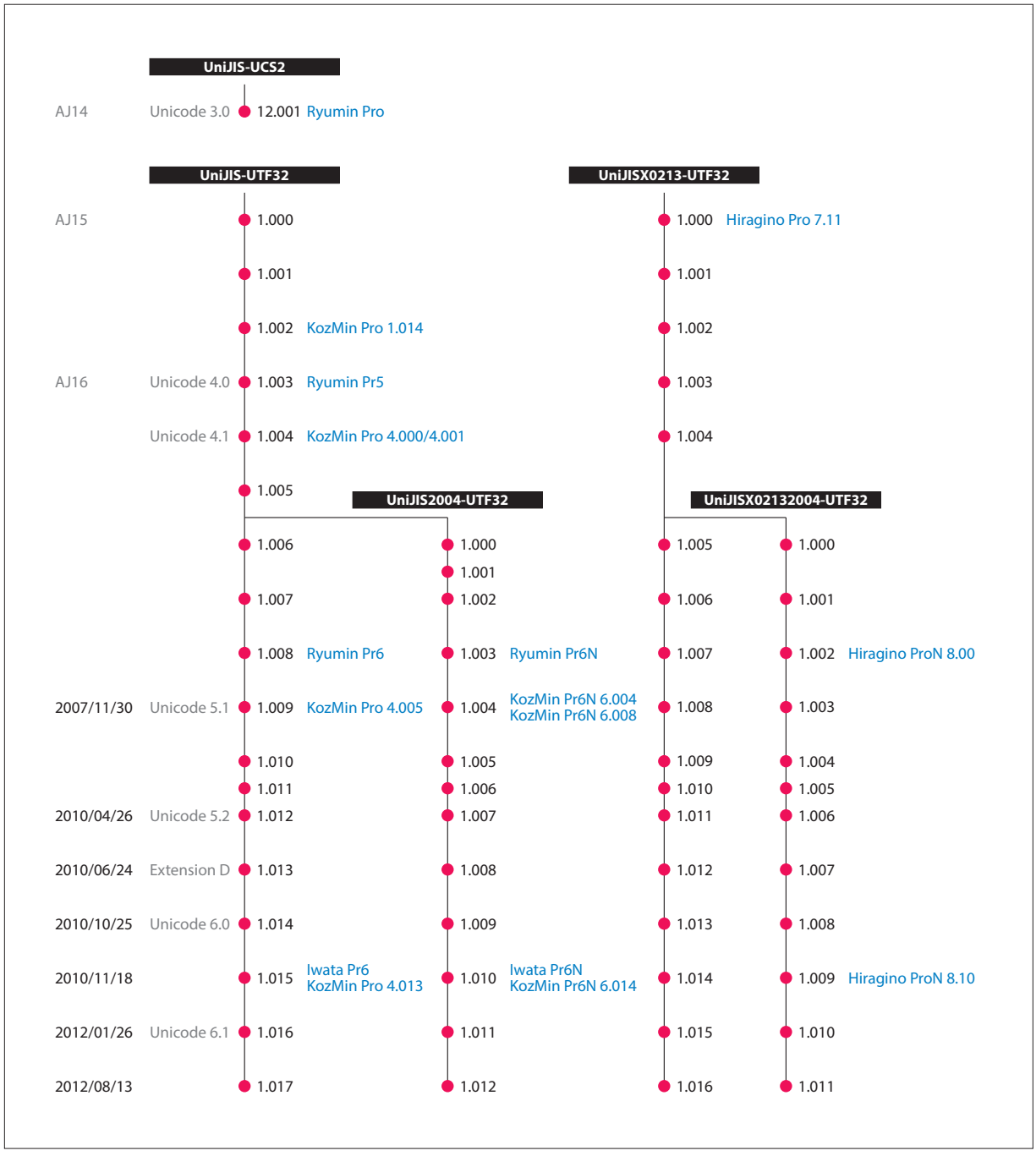

図4：CMapの系統図2012年版（作成：直井靖）[6]。

- こうしたcmapテーブルの違いによっても「GSUBフィーチャ文字化け」は発生する（→第2部「フォント変更にともなう選択置換の文字化けについて」 3.2.2 cmapの違い）。

6) 2012年8月13日のCMapファイルのバージョンアップにともなって改訂した最新版。
前バージョンは次を参照。直井靖「CMapの系統図2011年版」2011年8月30日、Mac OS Xの文字コード問題に関するメモ (http://d.hatena.ne.jp/NAOI/20110830)

1.1.2 GSUBフィーチャとGSUBルックアップ

OpenTypeフォントが内蔵する多くのテーブルの中でも、GSUB、GPOS、BASE、JSTF、GDEFの5つのテーブルを、とくにOpenTypeレイアウト・テーブル（“Advanced Typographic Table”とも）とよぶ。これらによりOpenTypeフォント固有の高度な組版機能が利用できる。「0.1 はじめに」で簡単に述べたが、OpenTypeレイアウト・テーブルが実現する機能を総称して「OpenTypeフィーチャ」（OpenTypeレイアウト・フィーチャとも）という。

OpenTypeレイアウト・テーブルの一つであるGSUBテーブルは、その名の由来が“Glyph Substitution”であることから分かるように、グリフを所定のものに置き換える働きをする。これによりユーザーが望むグリフが提供される。GSUBフィーチャとは、このGSUBテーブルによって実現される一つ一つの機能である。

OpenTypeレイアウト・テーブルのうち、GSUBテーブルとGPOSテーブルは、大まかに全体が4つの階層から構成されている。スクリプト（文字体系）、ランゲージ・システム（言語システム）、フィーチャ、ルックアップである。これはスクリプトを頂点とするツリー構造をなしている（図5）。

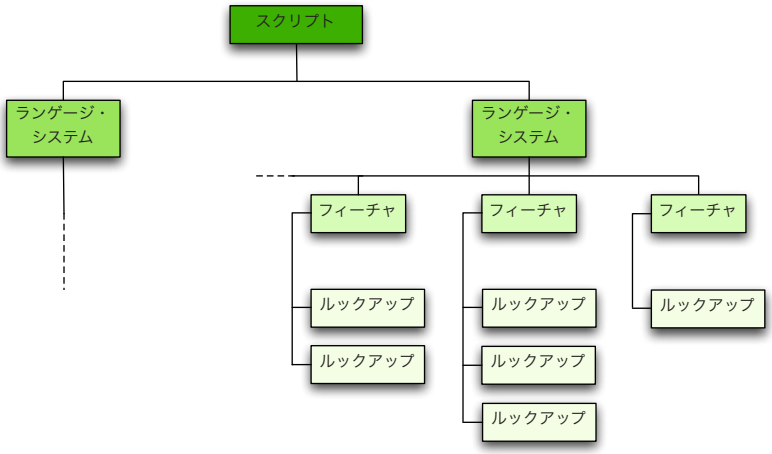

図5：GSUBテーブルとGPOSテーブルの共通構造。全体が大きく4つの階層からなっている。

GSUBテーブルとGPOSテーブルは構造が共通しているが、以下ではGSUBテーブルに限定して話をすすめる。上図にあるスクリプト（文字体系）とは、1つ以上の言語の表記に使われる文字の集合である。ラテン文字のように英語、ドイツ語、ベトナム語、アフリカーンス語等、多くの言語で使用されるスクリプトもあれば、日本語のようにラテン文字、平仮名、片仮名、漢字といった複数のスクリプトを使う言語もある（ただしOpenTypeフォントでは、平仮名と片仮名をまとめて1つのスクリプトとして扱う）。

ランゲージ・システムは英語、ドイツ語、ベトナム語、アフリカーンス語、日本語など、言語ごとの表記の体系をいう。スクリプトとランゲージ・システムを組み合わ

せることにより、置換の対象となるグリフがどのスクリプトや言語に所属するかを特定する。

こうした構造の結果として、GSUBテーブルがおこなう置換はスクリプトごと、ランゲージ・システムごとに振る舞いが規定される。

たとえば現実の表記としてラテン文字を使う多くの言語は「ff」「ffi」「fi」リガチャーを使うが、トルコ語ではこのうち「ffi」「fi」リガチャは使わない。そうした言語の実態を反映して、多くの欧文OpenTypeフォントは同じ振る舞いを再現する。

「ff」「ffi」「fi」の入力後に、GSUBフィーチャ「liga」(欧文合字)を適用してみる。その上で、文字パネルの「言語」で英語やドイツ語等の言語を切り替えてみると、トルコ語を除く全ての言語は「ff」「ffi」「fi」のいずれもリガチャーになる(図6上)。ところがトルコ語だけは「ff」しかリガチャーにならない(図6下)。

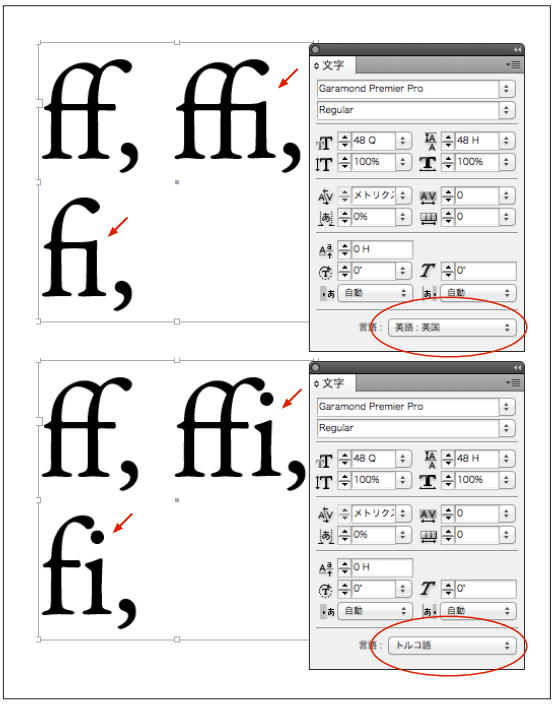

図6：ランゲージ・システムによるリガチャの違い。上は「言語」を「英語」に設定したもの。下はそれをそのまま「トルコ語」に変更したもの。上では合字（リガチャ）になっているものが、下ではなってない。これがランゲージ・システムによるリガチャの使い分けだ。

これはラテン文字を使うランゲージ・システムのデフォルト設定として、「ff」「ffi」「fi」リガチャーに置き換えるフィーチャが設定されている一方で、トルコ語のランゲージ・システムでは「ffi」「fi」リガチャを除外する設定がされているからだ(ちなみに、日本語もデフォルト設定と同じ振る舞いになる)。このように、OpenTypeレイアウト・テーブルの階層構造は、言語に依存したグリフを実装するのに便利な仕組みだ。

さて、4つの階層のうちフィーチャとルックアップこそが、GSUBテーブルの「主役」である。既に述べたようにフィーチャは、GSUBテーブルによって実現される具体的な一つ一つの機能だ。ここでいう機能とは、特定の規格や施策が例示したグリ

フ・デザインに置換したり、デザインの異なる同種の記号類に切り替えたり、全角と半角など文字幅の異なるグリフに置換したりといったことを指す。そして、そうした個々のフィーチャを識別・参照するためのタグがフィーチャ・タグであり、ルックアップとはフィーチャによって置換されるCIDの索引である。

OpenTypeフォントの仕様には、いくつか基本となるグリフ置換の動作が定められている。これをGSUBルックアップ・タイプとよぶ（→1.1.3 GSUBフィーチャの分類）。4つの階層のうちフィーチャでは、個々のフィーチャの振る舞いを実現するために、どのGSUBルックアップをどんな順番で適用するかが定義されている。そしてルップアップの階層では、置換の対象となるCIDの組み合せが記述されたテーブルがおかれている。

このようにOpenTypeフォントは、高度な組版を実現するための、1個の巨大で巧妙なデータベースと言える。しかし、OpenTypeフォント自身がグリフの置換をするわけではない。それをするのは、InDesignなどのOpenTypeフィーチャに対応したアプリケーションである。それらアプリケーションは、OpenTypeフォントに格納された多様なデータに基づいてグリフの置換をおこなう。逆に言えば、InDesignといえどもフォントに組み込まれていないグリフの置換はおこなえない。

1.1.3 GSUBフィーチャの分類

日本語というランゲージ・システムにおいては、前後の文字列のパターンに応じた置換をするような複雑なGSUBフィーチャより、比較的シンプルな置換をするGSUBフィーチャがよく使われる。その反面、多様なフィーチャが使用されるのが特徴と言える。

前章で述べたようにOpenTypeフォントの仕様には、GSUBルックアップ・タイプとして基本となるグリフ置換がいくつか定められている。個々のフィーチャの振る舞いは、これが大きく関わっている。したがって、使用されるGSUBルックアップ・タイプごとに整理することでGSUBフィーチャの分類ができる。以下がそのリストである。

図としてInDesignのメニューと個々のGSUBフィーチャの相関を示したものも掲載した（図7-1〜7-4）。リスト中のリンク先は、仕様にある、“Registered features”（登録されたフィーチャの解説）である[7)]。

ただし、仕様にはフィーチャ名とその振る舞いは定義されているものの、その振る舞いを実現するGSUBルックアップ・タイプの選択は“Recommended implementation”（勧告された実装）として例示されるだけで、どのように実装するかはベンダーに任されている。

たとえば小塚Pro、モリサワPro/Pr6、ヒラギノProなどの「frac」はGSUBルックアップ・タイプ4（合字置換）が実装されているのに対して、小塚Pr6N、モリサワPr6N、ヒラギノProNなどの「frac」は、GSUBルックアップ・タイプ6（連結文脈型グリフ置換）が実装されている[8)]。また、以下で単独置換に分類されている「trad」は、多くのフォントでは部分的にGSUBルックアップ・タイプ3（選択置換）も実装している[9)]。

このように、GSUBフィーチャとGSUBルックアップ・タイプは必ずしも1対1対応するわけではなく、以下のリストでの分類も便宜的なものであることにご注意いただきたい。

---

7) 「OpenTypeレイアウトへの道（5）…GSUBテーブル」2008年8月19日、vanillaの日記（http://vanillasky-room.cocolog-nifty.com/blog/2008/08/opentype5gsub-2.html）

8) 直井靖「「スラッシュを用いた分数」の仕様変更」2010年3月15日、Mac OS Xの文字コード問題に関するメモ（http://d.hatena.ne.jp/NAOI/20100315）

9) 直井靖「‘trad’タグで1対n置換となる例」2007年8月22日、Mac OS Xの文字コード問題に関するメモ（http://d.hatena.ne.jp/NAOI/20070822/1187779020）

■単独置換／Single Substitution（GSUBルックアップ・タイプ1）

1つのグリフを他の1つのグリフに置き換える。

㋓⑦	‘dnom’	Denominators（分母）
㋪❷Ⓑ	‘expt’	Expert Forms（エキスパート字形）
㋕⑫❶	‘fwid’	Full Widths（等幅全角字形）
㋖⑧	‘hkna’	Horizontal Kana Alternates（横組み用かな）
	‘hojo’	Hojo Kanji Forms（JIS X 0212-1990 Kanji Forms）
㋐❽Ⓗ	‘hwid’	Half Widths（等幅半角字形）
㋂⑩	‘ital’	Italics（欧文イタリック）
㋁❸Ⓒ	‘jp78’	JIS78 Forms（JIS78字形）
㋘❹Ⓓ	‘jp83’	JIS83 Forms（JIS83字形）
㋜❺Ⓖ	‘jp90’	JIS90 Forms（JIS90字形）
❻Ⓕ	‘jp04’	JIS2004 Forms（JIS04字形）
㋝❼Ⓔ	‘nlck’	NLC Kanji Forms（印刷標準字体）
㋙⑥	‘numr’	Numerators（分子）
㋑	‘pkna’	Proportional Kana（プロポーショナルかな）
㋞⑪Ⓚ	‘pwid’	Proportional Widths（プロポーショナル字形）
㋗⑩❶	‘qwid’	Quarter Widths（等幅四分字形）
㋒	‘ruby’	Ruby Notation Forms（ルビ用字形）
㋔⑤	‘sinf’	Scientific Inferiors（下付き数字）
㋌⑤	‘subs’	Subscript（下付き文字）
㋍④	‘sups’	Superscript（上付き文字）
㋚❶Ⓐ	‘trad’	Traditional Forms（旧字体）〈タイプ3との複合型〉
㋟❾❶	‘twid’	Third Widths（等幅三分字形）
㋅	‘vert’	Vertical Writing（縦組み用全角字形）
㋇⑨	‘vkna’	Vertical Kana Alternates（縦組み用かな）
㋆	‘vrt2’	Vertical Alternates and Rotation（縦組み用回転字形）
㋃③	‘zero’	Slashed Zero（スラッシュ付きゼロ）

■選択置換／Alternate Substitution（GSUBルックアップ・タイプ3）

1つのグリフを複数の内1つのグリフに置き換える。

㊽	‘nalt’	Alternate Annotation Forms（修飾字形）
㊾	‘aalt’	Access All Alternates（すべての異体字）[10]

〈いずれもフォントによりタイプ1との複合型〉

■合字置換／Ligature Substitution（GSUBルックアップ・タイプ4）

複数のグリフを1つのグリフに置き換える。

㋐	‘afrc’	Alternative Fractions（分数）
㋑	‘ccmp’	Glyph Composition / Decomposition（字体組版／分解）
㋒①	‘dlig’	Discretionary Ligatures（任意の合字）
㋛⑫	‘liga’	Standard Ligatures（欧文合字）
②	‘frac’	Fractions（スラッシュを用いた分数）〈タイプ6の場合あり〉

凡例

図7での番号　‘GSUBフィーチャ名’　仕様上の通称（InDesignでのメニュー項目名）

上記のルックアップ・タイプにつづく括弧内は仕様上の通称とその翻訳名だ。そのうち、“Alternate Substitution”は今のところ定訳がないようだが、本稿ではその振る舞いから「選択置換」と呼ぶことにする。

10）多くの日本語フォントでは、「aalt」はデータのサイズが大きいので拡張形式のGSUBルックアップ・タイプ7で実装されており、その次のレイヤーがGSUBルックアップ・タイプ3（およびタイプ1）となる。

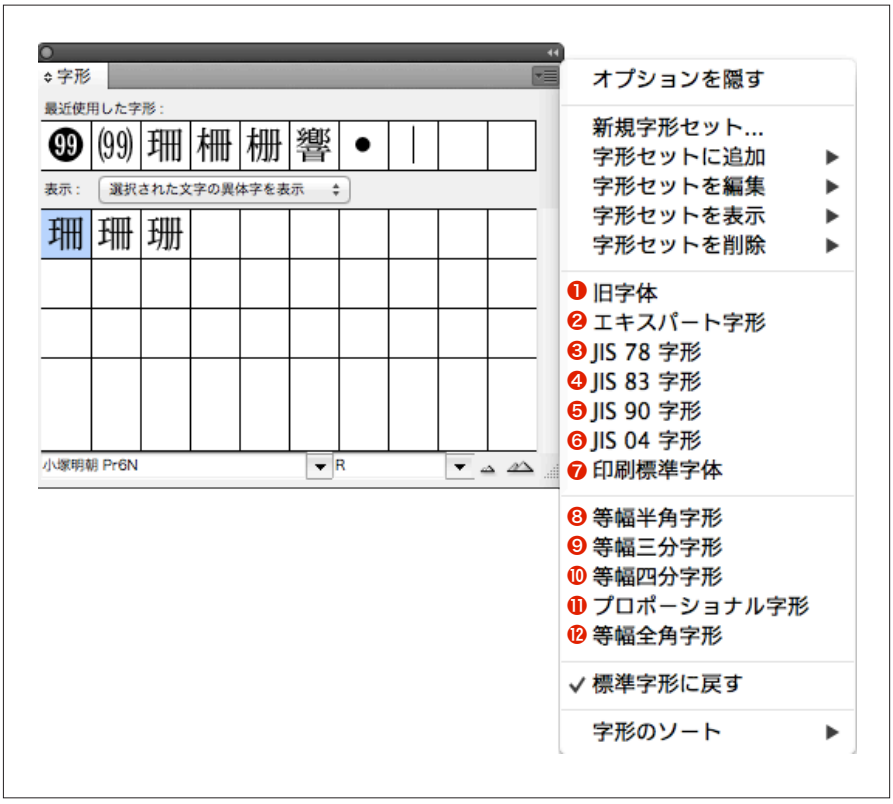

図7-1：字形パネルメニューからアクセスできるGSUBフィーチャ。

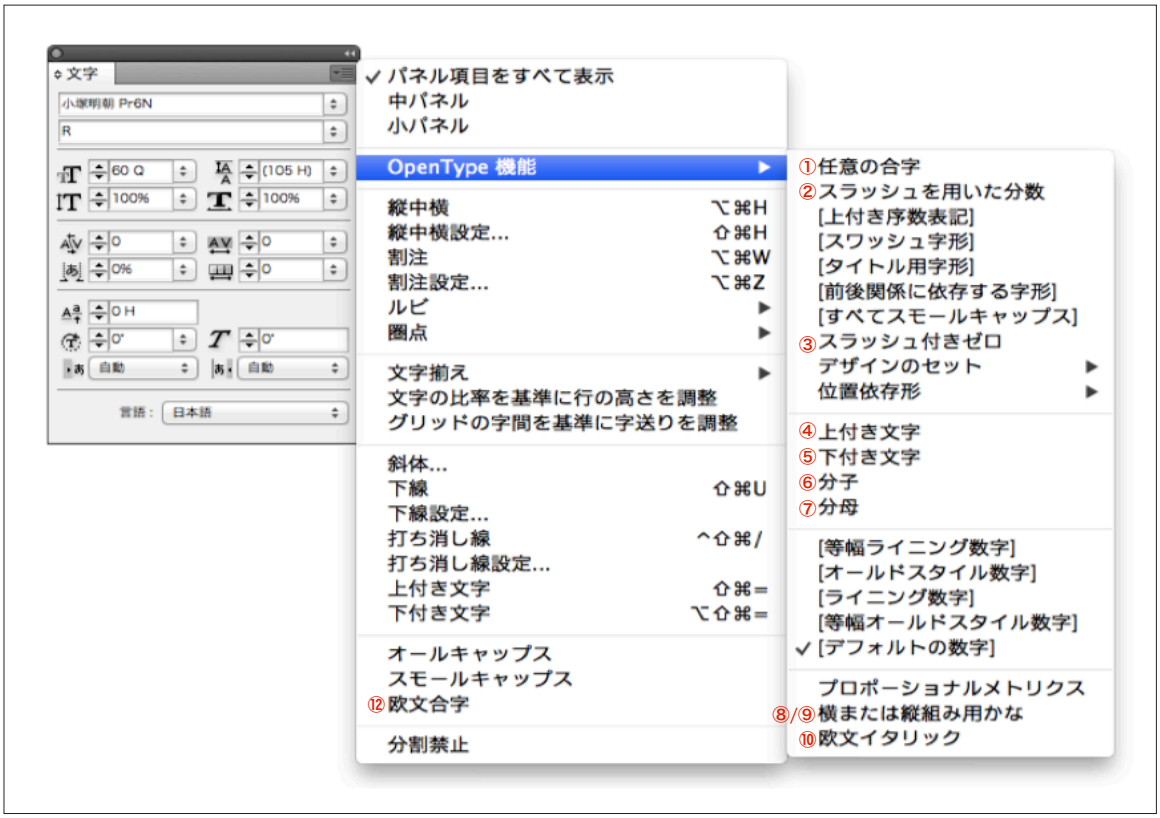

図7-2：文字パネルメニューからアクセスできるGSUBフィーチャ。ここでは日本語フォントによく実装されているGSUBフィーチャに限定して示している。

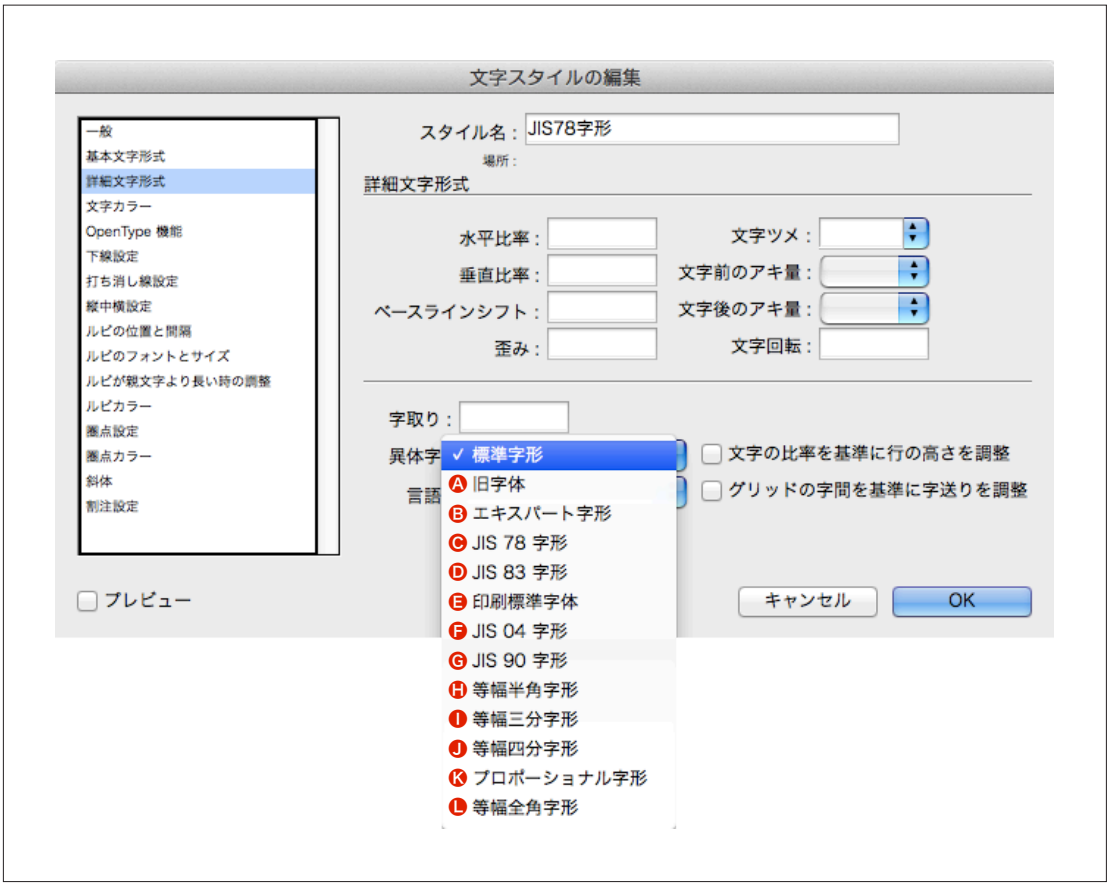

図7-3：文字スタイルの設定メニューからアクセスできるGSUBフィーチャ（段落スタイルの設定メニューも同様）。前図までは選択した文字にGSUBフィーチャを適用するためのメニューだったが、これはスタイルの設定としてGSUBフィーチャを指定することで、そのスタイル全体に一括して適用するという点で上掲のメニューと異なる。

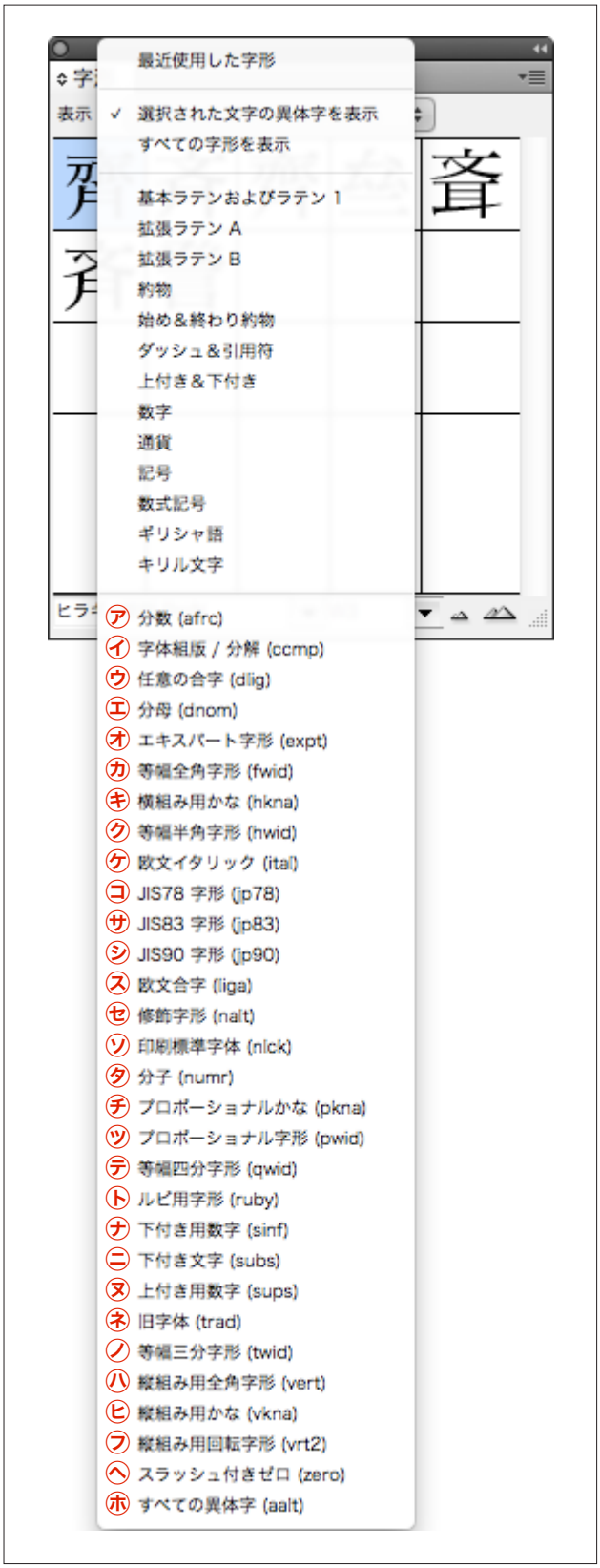

図7-4：字形パネルメニューの「表示」にあるGSUBフィーチャ。前図まではGSUBフィーチャを適用するためのメニューだったが、この「表示」は任意のGSUBフィーチャが適用できるグリフを表示するためのものだ。適用可能なGSUBフィーチャはフォントによって違うので、フォントが変われば表示も変わる。図はヒラギノ明朝ProNのもの。

1.2　InDesignにおける異体字の制御

InDesignにおいて異体字がどのように制御されているかを理解する上で、2つのポイントがある。1つは異体字がInDesign内部でどのように表現されているか、そして異体字がどのような方法で入力／置換されるかだ。互いに両者は関わり合っているが、まずその関係を概観した図を以下に示す（図8）。

入力／置換方法 ＼ 内部表現	Ⓐ字形パネル・ダブルクリック	Ⓑメニューからの適用	ⓒInDesignタグ	Ⓓスクリプト*4	ⒺMac OS XやIMEの文字パレット*5
㋐Unicode符号位置のみ	○		○	○	○
㋑単独置換*1	△	○	○	○	
㋒合字置換*2	△	○	○	○	
㋓選択置換*3	△		○	○	
㋔CIDによる直接指定	△		○		△*6

図8：InDesignにおける異体字の入力方法と内部表現の関係（作成：直井靖）。

凡例

○……可能な内部表現をすべてユーザーが指定できる

△……自動的に内部表現を割り振られる

*1……単独置換のフィーチャのうち、「hojo」（補助漢字のグリフ）と「pkna」（プロポーショナルかな）はInDesignのメニューではサポートされておらず、「aalt」や「nalt」と同様に字形パネル・ダブルクリックなどの方法で入力するしかない。また、「vert」と「vrt2」は縦組みの際にアプリケーションが利用するフィーチャなので、メニューには含まれていない。

*2……合字置換のフィーチャのうち、「afrc」（分数）はInDesignのメニューではサポートされていない。また、「ccmp」（字体組版／分解）は常にオンなので、フィーチャを適用するメニューには存在しない。

*3……「aalt」と「nalt」は選択置換と単独置換の複合型であることが多い。しかしInDesignにおける入力方法と内部表現の関係を見る上では（「aalt」と「nalt」に関しては）単独置換の側面を考慮する必要はないので、この図では選択置換に分類している。

*4……InDesignにおいて使用可能なスクリプトはJavaScript、及びMac版ではAppleScript、Windows版ではVBScriptだ[11]。

*5……Mac OS X 10.5以前の文字パレットまたは同10.6の文字ビューアにおける「表示：グリフ」または「表示：コード表＞選択したフォント内の異体字」、かわせみの文字パレットにおける「コード＞コード体系：CID（Adobe-Japan1）」、ATOK for Macの文字パレットにおける「コード表＞体系＞グリフ（Adobe Japan 1）」で、InDesign側と同じフォントに設定した上で入力した場合の動作。Mac OS X 10.7の「文字ビューア」には、「表示：グリフ」および「選択したフォント内の異体字」というオプションは存在しない。

11) アドビシステムズ「InDesign / スクリプト」(http://help.adobe.com/ja_JP/indesign/cs/using/WS0836C26E-79F9-4c8f-8150-C36260164A87a.html)、AdobeSystems *"Adobe Introduction To Scripting"* 2007 (http://wwwimages.adobe.com/www.adobe.com/content/dam/Adobe/en/products/indesign/pdfs/adobe_intro_to_scripting.pdf)

*6……Mac OS XやIMの文字パレット（文字ビューア）からグリフを入力した場合、㋑〜㋓で表現可能なグリフでも、すべて㋔CIDによる直接指定で表現される。

上図で分かるとおり、ほとんどの入力／置換方法では、入力後の内部表現をユーザーが指定することができるが、Ⓐ字形パネル・ダブルクリックだけはそれができない。この場合は、InDesignの側で一定のルールにしたがって自動的に㋑〜㋔に割り振られる（Ⓔにも「△」があるが、この場合㋐以外は全て㋔になる）。

字形パネルは多くのユーザーにとって馴染み深いと思われるが、これを使って入力する場合、意図せずにGSUBフィーチャが適用され得ることに注意したい。そこで必要になるのは内部表現の種類を見分けることである。

1.2.1　InDesignにおける異体字の内部表現

ここでいう内部表現とは、InDesign内部におけるデータの表現方法のことである。InDesignでは「ウィンドウ→情報」(以下、情報パネル)で確認できる。また図8でⒸInDesignタグとして紹介した「Adobe InDesignタグ付きテキスト」[12]で書き出すことによっても確認できる。

内部表現そのものにはたくさんの種類があるが、ここでは異体字に関係する5つの表現に絞って説明する。InDesignの内部機構はOpenTypeフォントの仕様と密接に関係しているが、5つの表現のうちいくつかは「1.1.3 GSUBフィーチャの分類」で説明したGSUBフィーチャの分類に関係する。

InDesignの内部表現を理解する上でポイントとなるのは、その冗長性だ。外見が同じグリフ(CID)でも、内部表現としては異なる場合が多い。以下は「國」(CID+4467)という同じグリフを表す、複数の内部表現を示したもの(図9)。

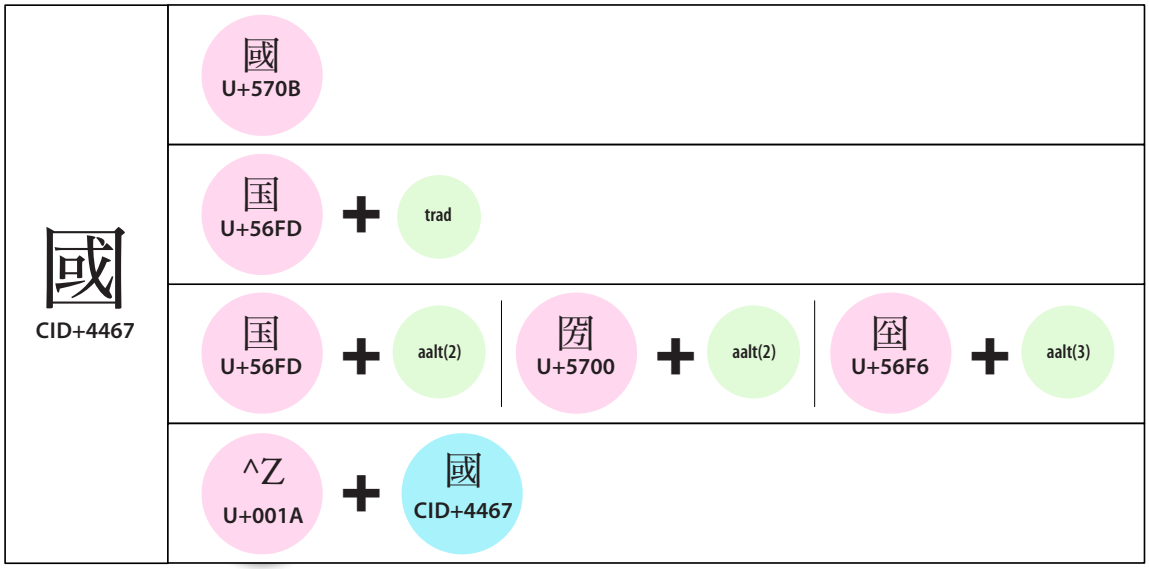

図9：グリフは同じなのに内部表現が異なる例。ピンクはUnicode符号位置、緑はGSUBフィーチャ[13]。水色はCID。なお、円内のグリフはあくまで参考につけたもので、Unicode符号位置やCIDそのものにグリフ形状の情報が含まれるわけではない。

このように、InDesignにおける異体字の内部表現は、同じグリフ(國)を表現するのに、Unicode符号位置だけを使う方法、なんらかのGSUBフィーチャを適用する方法、

12) アドビシステムズ「InDesign CS5タグ付きテキストユーザーガイド」2010年4月30日 (http://help.adobe.com/ja_JP/indesign/cs/taggedtext/indesign_cs5_taggedtext.pdf)

13) 本文で後述するが、選択置換（aaltに関する内部表現）は同じグリフを表すのに何通りもの方法がある。図9の3段目にある表現は、字形パネルなどで確認できる一般的なものではなく、スクリプト等でのみ指定が可能なものであることをお断りしておく（詳細は「1.2.2.1 Ⓐ字形パネル・ダブルクリック」参照)。

あるいはCIDを直接指定する方法など多くの種類がある。つまりInDesignにおいては、グリフの外見だけではその文字の内部表現を区別できない。

1.2.1.1 ㋐Unicode符号位置のみによる内部表現

GSUBフィーチャを伴わない、Unicode符号位置だけの内部表現。CIDを表現するのにUnicode符号位置だけを使用する(図10)。プレーンテキストと同じ内部表現であり、フォントの実装には依存しないので、比較的安定した情報交換が期待できる。本稿の目的であるGSUBフィーチャ文字化けは、この内部表現では発生しない。

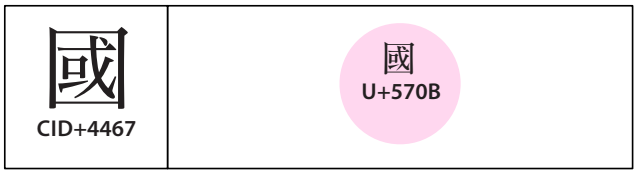

図10：Unicode符号位置1つがCID 1つを表現する方法。なお、以降の図版にも言えることだが、円内のグリフは参考にすぎない。Unicode符号位置及びCID番号にはグリフ形状の情報は含まれておらず、それぞれのフォントの実装に依存する。

次項以降に述べる内部表現は、いずれもUnicode符号位置を親字として、そこに何かを付加する形になっている。言い換えれば、この㋐Unicode符号位置のみによる内部表現は、親字だけの表現とも言える。

1.2.1.2 ㋑単独置換による内部表現

ここから先、1.2.1.4までの3項はGSUBフィーチャを使った内部表現となる。それぞれは置換のバリエーションによって異なる。

まず最もシンプルな単独置換を説明する。下図では親字はU+56FDであり、これとGSUBフィーチャ「trad」が適用されることで「國」を表すCID+4467に置き換えられる(図11)。以下、本稿ではGSUBフィーチャが適用された結果を「GSUB属性」、あるいは「異体字属性」とよぶ。属性とはそのものに備わっている固有の性質のことだ。

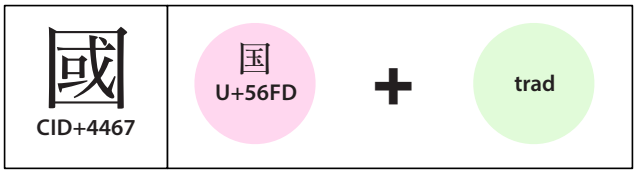

図11：単独置換のGSUBフィーチャを使った内部表現。

この「trad」(旧字体) というGSUBフィーチャは、少数の例外を除き[14]、1つのCIDを1つのCIDに置換する。同じ型のGSUBフィーチャに「jp78」(JIS78字形)、「expt」(エキスパート字形) 等、多数ある (→1.1.3 GSUBフィーチャの分類)。

他アプリへのコピペなどによりGSUB属性が消えてしまった場合、親字に化ける (これは次項以降の全ての内部表現に共通する)。たとえば図11の例でいうと、「trad」が消えて親字 (U+56FD／国) だけがペーストされる。

単独置換されたグリフの場合、親字とは意味や読みが共通することが多い。また次項の合字置換と違って親字1字に対して1つのグリフが置換されるので、コピペにより文字数が変わることはない。逆に言えば、どこが化けたか分かりづらいとも言える。もしもコピペをした場合、原本との照合は不可欠だろう。

1.2.1.3 ㋒合字置換による内部表現

複数のUnicode符号位置が親字となって単一のCID (グリフ) に置換される内部表現。合字置換のGSUBフィーチャがこの型で使用される。適用されるGSUBフィーチャに「liga」(欧文合字)、「frac」(スラッシュを用いた分数)、「afrc」(分数) などがある (ただし一部フォントの「frac」はルックアップ・タイプ6を使う。→1.1.3 GSUBフィーチャの分類)。

下図は「T」(U+0054)、「E」(U+0045)、「L」(U+004C) という文字列が親字となり、GSUBフィーチャ「dlig」(任意の合字) が適用されて、「TEL」(CID+7612) という単一のグリフに置換されている (図12)。つまりn対1の形である。

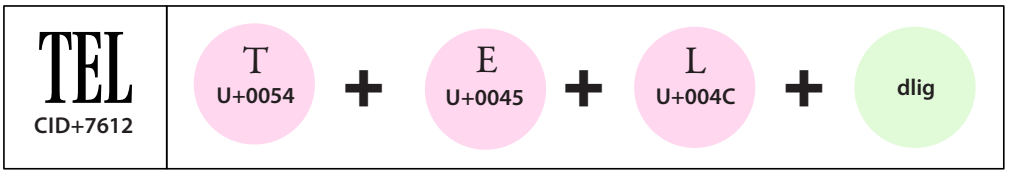

図12：親字が複数のグリフ置換における内部表現。合字置換のGSUBフィーチャが適用される。

合字置換の場合は親字が複数になるので、他アプリへのコピペによりGSUBフィーチャが消えると、文字数が増えることになる。例えば図12の例でいうと、「dlig」が消えると親字のU+0054、U+0045、U+004Cが残るので、元は1文字だったものが3文字に増えてしまう。

また、このタイプの中には「温／U+6E29」「泉／U+6CC9」を親字とし「dlig」を

14) この例外こそが、「1.1.3 GSUBフィーチャの分類」において「trad」をGSUBルックアップ・タイプ3との複合型とした所以だ。註9も参照。

適用することで「♨」(CID+12098) に置換するといったパターンも存在する (→第2部「メニュー選択」他のアプリにコピペすると化ける文字 dlig)。この場合はGSUB属性が消えてしまうと、マークの代わりに文字列が出現することになるので、不都合なことも多いだろう[15]。

1.2.1.4 ㊓選択置換による内部表現

複数の異体字からなるグループから1つのグリフを選択する内部表現。選択置換のGSUBフィーチャ「aalt」(すべての異体字) か「nalt」(修飾字形) が使用される。この内部表現の特徴は、親字のUnicode符号位置、GSUBフィーチャ、及びその異体字番号の3つによりグリフを表現することだ。

下図ではU+56FD「国」という親字と、GSUBフィーチャ「aalt」、そして異体字番号の「2」という3つの情報がセットになって、CID+4467というグリフを表現している (図13)。

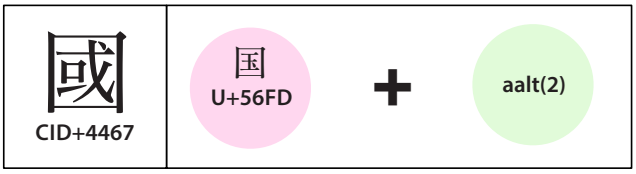

図13：選択置換のGSUBフィーチャを使った内部表現。

上記で異体字番号が「2」ということからも分かるとおり「1」がある。そして、このグループの場合「3」もあり、親字も含めて4つのグリフが1つのグループを構成している (図14)。

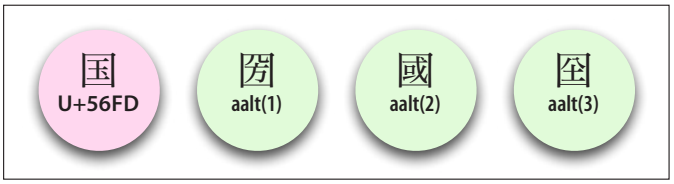

図14：選択置換における「国」のグループ。GSUBフィーチャ「aalt」と異体字番号は、いずれも親字と対になって1つのグリフを表現する。

ところでInDesignの内部表現においては、必ずUnicode符号位置が親字となる。換言すればUnicode符号位置を持っていれば親字になれる。上記の「国」のグループは、すべての文字がUnicode符号位置をもつ。ということは、これらすべてが親字になれることになる (図15)。つまり、それぞれは親字であると同時に、互いに子でもあると

15) 前掲註1、田嶋淳『InDesign データ→電子書籍で字形の変化する文字』p.1 も参照。

いう関係になっている。

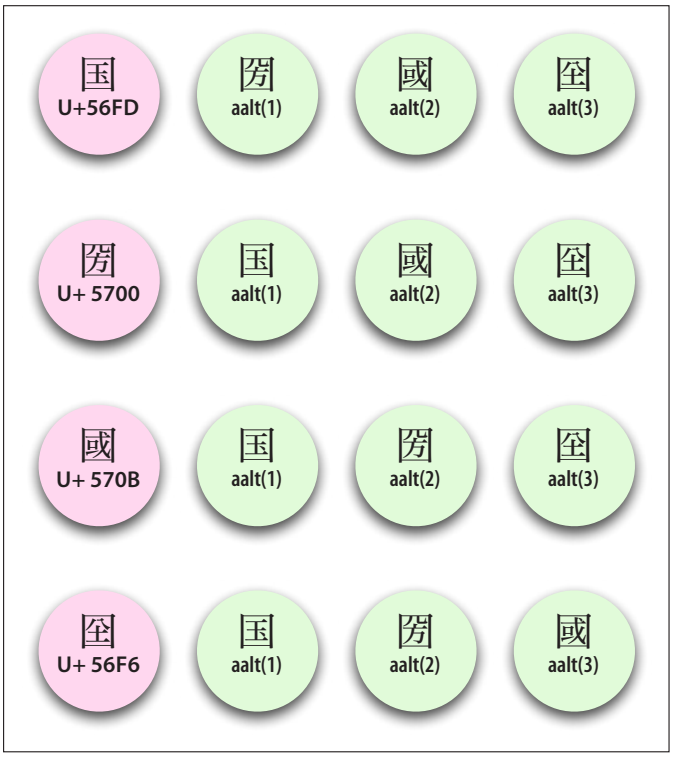

図15：「国」のグループにおける全ての内部表現。ただし、これらは表現として使用可能という意味であり、全てを指定できるのはⓒInDesignタグやⓓスクリプトに限られる。

その結果、1つのグリフを表現する方法はさらに冗長性を増す（図16）。

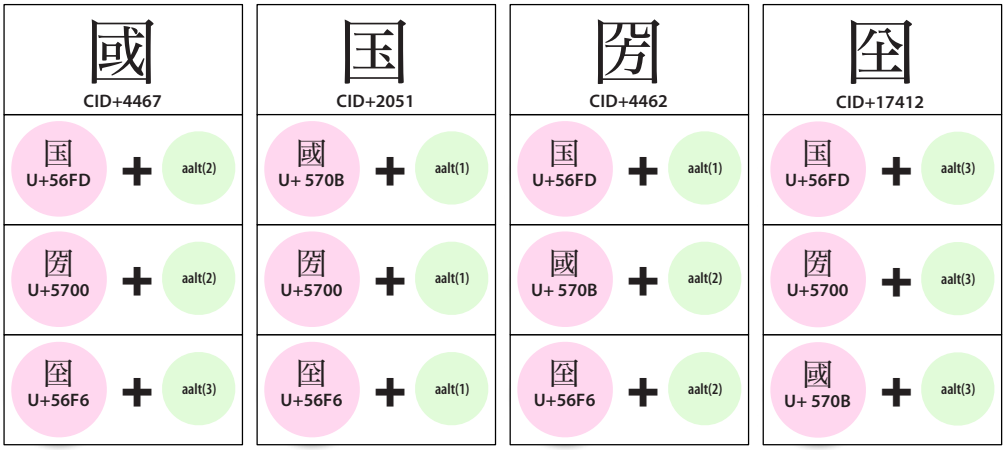

図16：それぞれのグリフを表す選択置換の内部表現。どの表現方法も最上段のグリフを表している。

このⓔ選択置換の内部表現における重要なポイントは、親字と異体字番号の組み合せはフォントに依存していることだ。フォントが変わると組み合わせの内容も変わる場合があり、これによって文字化けが発生する。

ここまで例に挙げた「国」のグループの例では、全ての文字がUnicode符号位置を

持っていたので比較的単純な説明ですんだ。しかし、そのようなケースばかりと限らない。グループ内にGSUBフィーチャやCIDの直接指定でないと使えないようなグリフを含む場合、親字と異体字番号の組み合せは複雑な問題をもたらすことになる。ここでは深く立ち入らないが、第2部「フォント変更にともなう選択置換の文字化けについて」では、この問題について具体的な例を挙げつつ説明している。また、本項では選択置換のうち「aalt」だけを説明しているが、もう1つの「nalt」の振る舞いについても、この記事を参照してほしい。

㋓選択置換の場合も、他アプリへのコピペなどによりGSUB属性が消えれば親字に化けることになる。化け方は㋑単独置換と似ている。つまり、親字とは意味や読みが共通することが多く、また親字1字に対してグリフ1つが置換されるのでコピペによって文字数は変わらない。どこが化けたか分かりづらいのも同じで、やはりコピペした際には原本との照合は不可欠だろう。

1.2.1.5 ㋔CID直接指定による内部表現

以上説明した内部表現は、いずれもUnicode符号位置やGSUBフィーチャを使ったものだった。最後に説明するのは㋔CIDによる直接指定、つまりCIDを表現するのにそのCID自身を使う表現だ。ただし、仕様上の制約で親字としてUnicode符号位置が必要になるので、便宜的に不可視の制御文字(U+001A)が親字に割り当てられている(図17)[16]。別の言い方をすると、InDesignは制御文字を利用することで、CID番号を文字として扱えるようにしたとも言える。

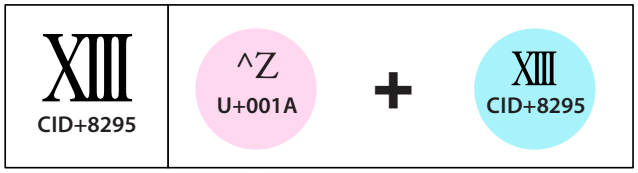

図17：㋔CID直接指定による内部表現。CIDを表現するのにそのCID自身を使う。

この内部表現をとるグリフは記号類が多い。とくにグリフ数という点で目につくのは、数字のバリエーションだ。たとえば丸付き数字は①～㊿まではㇰUnicode符号位置のみで表現できるが、㊿～㉑㊿はCID直接指定によって表現される。あるいは括弧付きの漢数字は㈠～㈩まではㇰUnicode符号位置のみで表現できるが、㈪～㈬はCID直接指定により表現される。このように数字のバリエーションの場合、CIDによる直接指定で数を補っているパターンが多い。

16) U+001Aは、ASCIIなどで使われる制御文字0x1A「SUB」(Substitute)に由来する。その意味合いは、ゲタ「〓」あるいはU+FFFD REPLACEMENT CHARACTER「□」に近い。

前項まで述べたGSUBフィーチャを使う内部表現の場合、他のアプリにコピペしても親字の字義はある程度共通していたので、少なくとも意味が通じることは期待できた。しかし、このCID直接指定では親字が制御文字（U+001A）なのでそうはいかず、純然たる（ある意味分かりやすい）文字化けとなる。

1.2.2 InDesignにおける異体字の入力／置換方法

前節ではInDesignの内部表現について、おもにその冗長性に注目しながら説明した。この節では、InDesignで異体字を入力／置換する方法について、内部表現を踏まえながら説明をすすめる。

内部表現と入力／置換方法の相関関係については、すでに図8として概観した。そこでは入力／置換方法として5種類を挙げたが、ここでは多くのユーザーにとって身近と思われる、Ⓐ字形パネル・ダブルクリックと、Ⓑメニュー選択の2つに絞って述べることにする。

ここで注意してほしいのは、異体字を入力することで勝手に親字の符号位置が変わってしまう場合があることだ。ただし、これはⒶだけに限られ、Ⓑではそういうことは起こらない。この親字が変わる現象は、本稿の目的であるGSUBフィーチャ文字化けの一因ともなるものだ。

1.2.2.1 Ⓐ字形パネル・ダブルクリック

字形パネルを使って入力をするには2つの方法がある。ダブルクリック入力とプレス入力だ。前者は表示されているグリフをダブルクリックすることで入力する方法（図18左）であり、今はこの方法を前提に話をすすめることにする。

なお、プレス入力は字形パネルに表示されたグリフをプレスして表示されるサブパネルを選択することで入力する方法（図18右）だが、ダブルクリック入力との間で入力後の内部表現に違いはない。また、以下に述べる挙動は「表示→選択された文字の異体字を表示」「表示→すべての字形を表示」のいずれかで入力した際のものに限定して説明していることもお断りしておく。

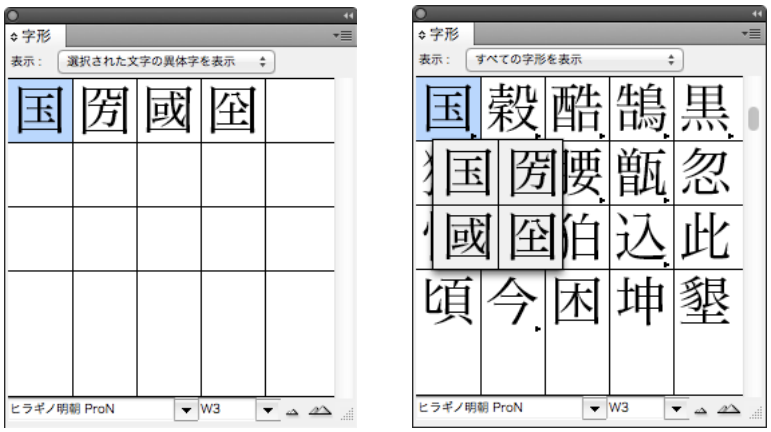

図18：字形パネルにおける2つの入力モード。左がダブルクリック入力、右がプレス入力。

さて、「1.2 InDesignにおける異体字の制御」でも簡単に触れたが、InDesignにおける異体字の入力／置換方法のうち、Ⓐ字形パネル・ダブルクリックだけは入力後の内部表現をユーザーが指定することができない。ここではInDesignの側で自動的に以下の順序で割り振られる。

第1位：Unicode符号位置のみによる内部表現
第2位：GSUBフィーチャを使った内部表現（字形パネルの「表示」メニュー順）
第3位：CID直接指定による内部表現

もしダブルクリックしたグリフが、Unicode符号位置だけで表現可能なグリフであれば、この形が最優先で採用される。それができない場合、次いでGSUBフィーチャによる内部表現が採用され、それでも表現できないものはCID直接指定として表現される（→第2部「フォント変更にともなう選択置換の文字化けについて」1.2.1 ダブルクリック入力における内部表現の優先順位）。

このうちの第2位、GSUBフィーチャによる内部表現で、GSUBフィーチャが採用される優先順序は、字形パネルの「表示」メニュー順だ。字形パネルの中程に「表示：」と書かれた右をプレスすると、以下のようなメニューが表示される（図19）。

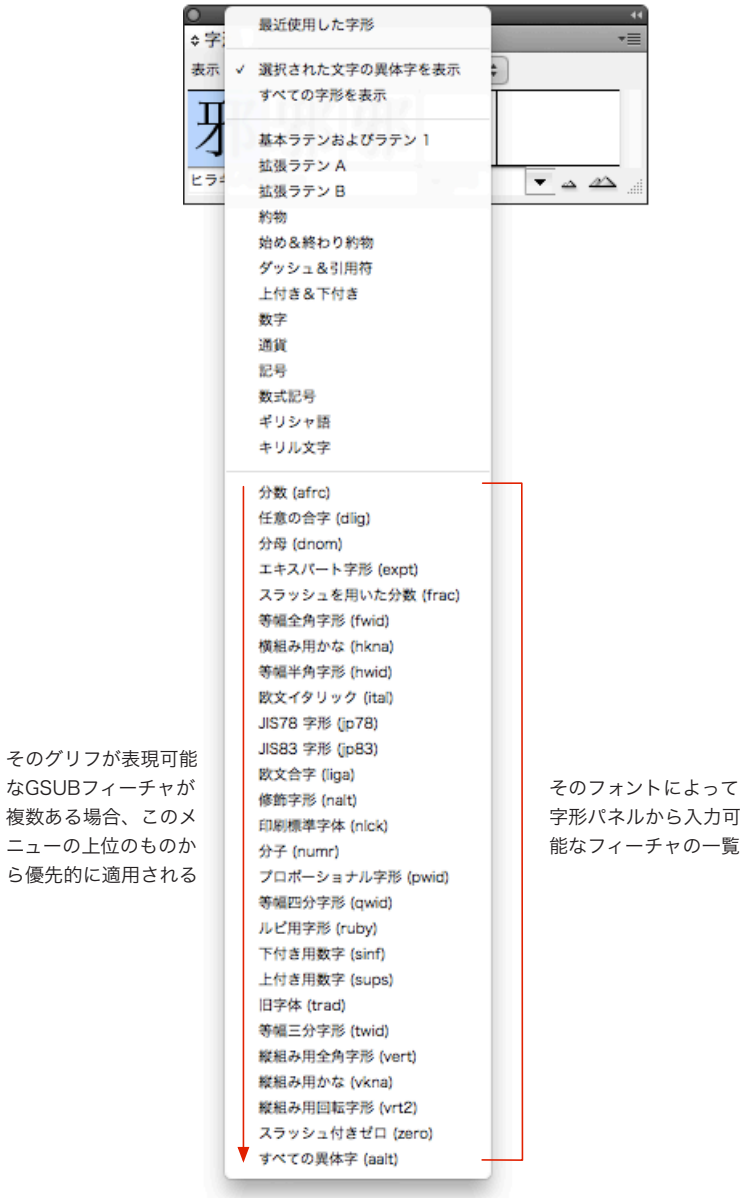

図19：字形パネルの「表示」メニュー。どのGSUBフィーチャが掲載されるかはフォントに依存する。

このメニューはよく見ると4つの区画に分かれており、そのうちの下端が当該フォントで字形パネルから入力できるGSUBフィーチャの一覧だ。この一番上に記載されているGSUBフィーチャから優先的に採用される。表示メニューの内容はフォントに依存するが、「aalt」(すべての異体字)だけは決まって一番下に来ることになっている(→図7-4)。つまり、「aalt」以外のGSUBフィーチャが優先的に適用され、それらで表現できない場合、最後に「aalt」が選択される。

入力後にどのような内部表現になるかは、字形パネルに表示されるツールチップで確認できる(図20)。

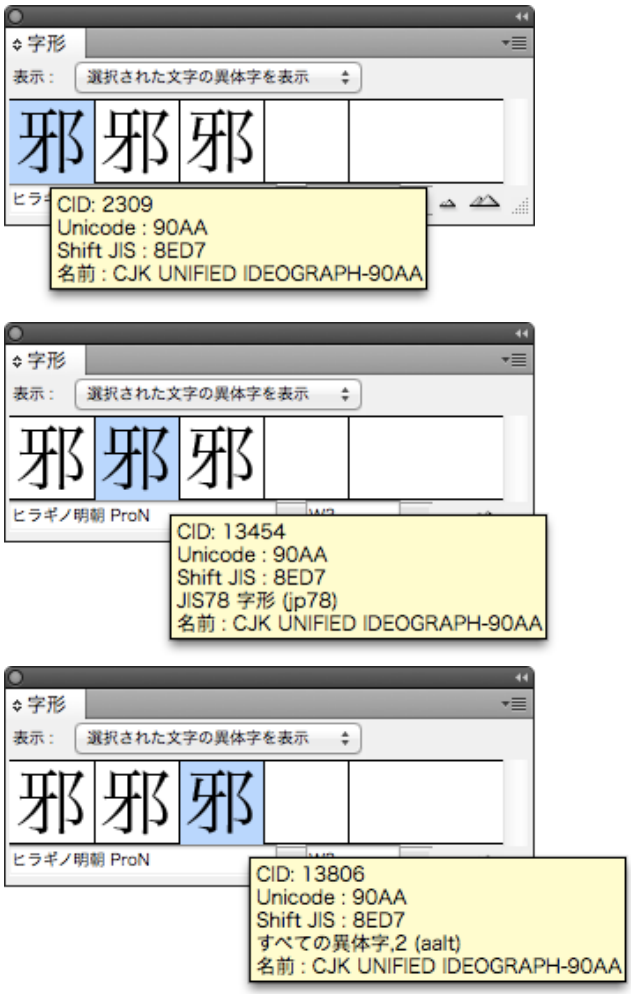

図20：字形パネルにおけるツールチップの表示。

上から説明していこう。字形パネル左端に表示されているCID+2309「邪」は、Unicode等の符号位置だけが表示されており、GSUBフィーチャの情報はない。だからこのグリフを入力した場合の内部表現はUnicode符号位置だけの表現となる。

図20中。CID+13454「邪」は、Unicode符号位置の他に単独置換のGSUBフィーチャ「jp78」が表示されている。つまり、このグリフはUnicode符号位置だけでは表現できず、「jp78」によって表現できる。具体的には親字U+90AA「邪」に「jp78」が適用された形となる。

図20下。CID+13806「邪」は、選択置換のGSUBフィーチャ「aalt」と異体字番号「2」が表示されている。つまり、このグリフはUnicode符号位置では表現できず、数多くあるGSUBフィーチャの中でただ1つ、「aalt」によってのみ表現できる。具体的には親字U+90AA「邪」に「aalt」と異体字番号「2」が適用された形となる。

なお、上図にはないが、入力後にCID直接指定で表現されるグリフの場合、ツールチップにはUnicode符号位置が表示されず、「名前」の欄に「NULL」と表示されることで判別できる。

さて、この字形パネル・ダブルクリックの注意点は、親字が変わってしまう場合があることだ。たとえば、前章で述べた「国」のグループ(図9、図16)がまさにその例だったのだが、以下に示すのは「aalt」において親字が勝手に変更されるケースだ。まずヒラギノ Pro で「音」(U+97F3)を入力してみよう(図21)。

図21：ヒラギノ Pro で「音」(U+97F3)を選択した際の情報パネル（画面上部）と字形パネル右下隅のグリフ（CID+13664）の情報を表示するツールチップ（画面下部）。

この時、情報パネルには「音」の符号位置として「U+97F3」が表示されている。一方、マウスカーソルを字形パネルのうち、一画目が横棒の「音」(CID+13664)の上におくと、このグリフを入力した際の内部表現として、ツールチップには親字が「U+266B」、GSUB フィーチャ「aalt」、異体字番号「8」が表示される。

このグリフをダブルクリックで入力してみると、実際に親字が「U+266B」に変わっていることが確認できる(図22)。

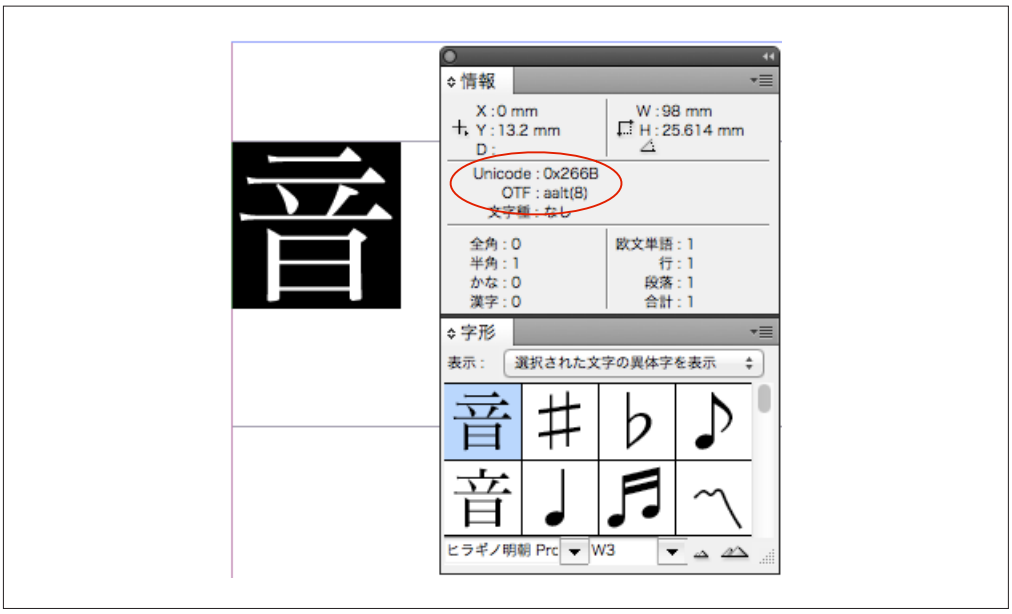

図22：ダブルクリックで一画目が横棒の「音」（CID+13664）を入力した。

「U+266B」とは「♫」だ。つまり、このCID+13664の「音」のグリフを他のアプリケーションにコピペすると「♫」に化けてしまう。なぜこうなるのか？　ヒラギノProにより「音」(CID+13664)を「aalt」で表現する際、ありうる全ての組み合わせを書き出してみた(図23)。

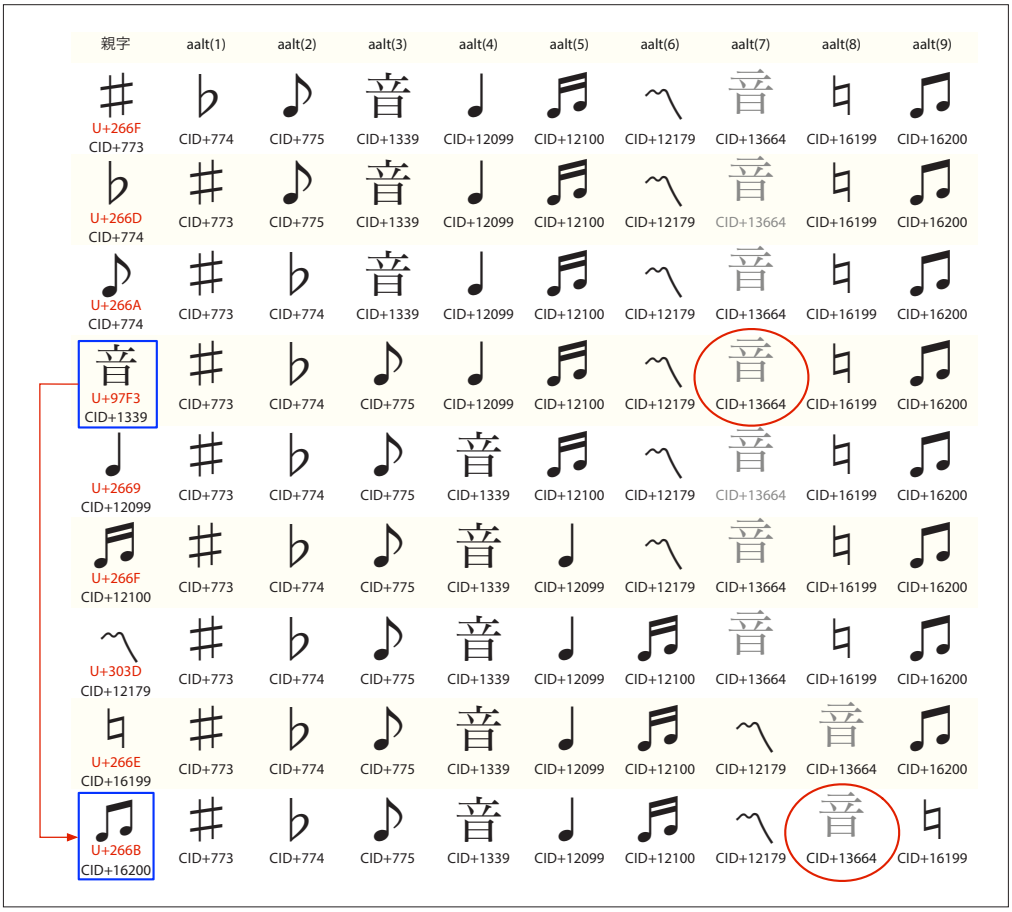

図23：InDesignでヒラギノProを使い、「aalt」で「音」（CID+13664）を表現する場合に、可能な全ての内部表現の組み合せ。淡色のグリフは符号位置を持たないことを表す。
「音」（CID+13664）は符号位置を持たず、aaltでしか表現できない。そういうグリフを入力すると、InDesignは「親字のCID番号が最も大きい内部表現」に変更してしまう。その結果、親字が「♫」に変更される。

ここまで繰り返し述べてきたように、InDesignの内部表現は冗長だ。その傾向は選択置換の内部表現で著しい。このヒラギノProにおける「音」の「aalt」グループも同じだ。ここでCID+13664の「音」はUnicode符号位置を持たず、また「trad」等の単独置換でも表現できず、もちろん合字置換でも表現できない。ただ1つ、このグリフを表現できるのは「aalt」のみ。

こうした「aalt」だけで表現可能なグリフをⒶ字形パネル・ダブルクリックで入力すると、そのグリフを表現可能な方法が複数あれば、InDesignは「親字のCID番号が最も大きい内部表現」を選択してしまう（この奇妙なルールについては→第2部「フォント変更にともなう選択置換の文字化けについて」2.4.1 aaltの親字を参照）。この結果、入力後の内部表現は「U+266B aalt 8」になり、親字は「音」（U+97F3）から「♫」（U+266B）に変更されてしまうというわけなのだ。もちろん、このグリフを他のアプリにコピペすれば「♫」に文字化けする。

このように、Ⓐ字形パネル・ダブルクリックで入力すると、意図せず親字が変わってしまう場合がある。上述した「音」の例はヒラギノProだけのものだが、同様の例として、「!」（U+0021）を選択して字形パネルから「⁉」（CID+12113）を入力すると、親字が「‼」（U+203C）に変更されてしまうケースがある。これはヒラギノPro、ヒラギノProN、小塚Pr6N、リュウミンPr6など、比較的多くのフォントで再現する。

なお、同じ㋓選択置換によるGSUBフィーチャでも、「nalt」における親字の変更では、ごく穏当に「CID番号が最も小さいグリフ」が採用されるので、「aalt」のような心配はない（→第2部「フォント変更にともなう選択置換の文字化けについて」2.4.2 naltの親字）。

1.2.2.2 Ⓑメニュー選択

文字列を選択した上で、字形パネル、及び文字パネルの右上隅からアクセスできるメニュー項目を選ぶと、選択した文字列全体に対してGSUB属性が付与される（図24）。ただし、フォントの側に対応したGSUBフィーチャが実装されていなければグリフは変化しない。

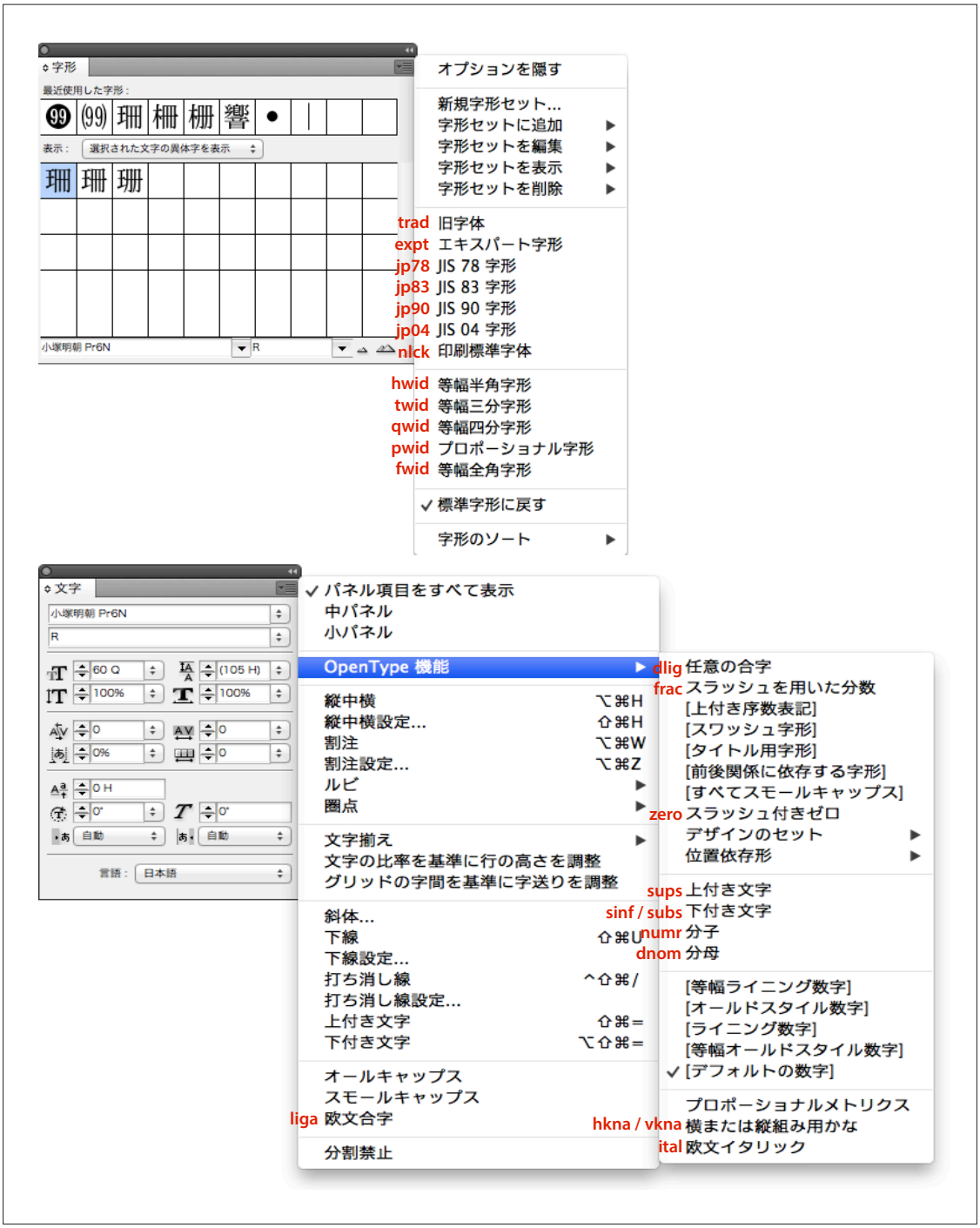

図24：字形パネルメニュー（上）／文字パネルメニュー（下）とGSUBフィーチャの関係。

前項Ⓐ字形パネル・ダブルクリックでは、InDesignの側で自動的に内部表現を割り振ったが、このⒷメニュー選択では、ユーザーの選んだGSUBフィーチャが適用される。したがって知識がありさえすれば、ユーザーは事前にどのような内部表現になるかイメージすることができる。

また、Ⓐは「入力」になるが、Ⓑは「置換」となることにも注意したい。例えばルビつきのグリフを選択しているとしよう。ここで、Ⓐによって異体字にした場合、ル

ビは消えてしまう。これは上書きされた「入力」の振る舞いだ。一方、Ⓑで異体字にしてもルビが消えることはない。つまりグリフが置き換わる「置換」だ。

ここから分かるように、Ⓑの場合は選択した文字に対して単純にGSUB属性が付与されるだけなので、Ⓐのように勝手に親字が変更されるようなことはおきず、親字のUnicode符号位置は保持される。

一方、入力後の内部表現が自動的に割り振られるⒶ字形パネル・ダブルクリックにおいては、原理上「CIDは同じだが内部表現は違う」ということはおきないが、ユーザーが自分の意志でGSUB属性を与えるⒷではそれが起こり得る（図25）。

図25：上はいわゆる「半角のl」（U+0031）に対してGSUBフィーチャ「hwid」（等幅半角字形）を適用したグリフ。下はいわゆる「全角のｌ」（U+FF11）に同じフィーチャを適用したもの。内部表現は違うがグリフは同じだ。なお、「ccmp」（字体組版／分解）も適用されているが、これはサポートするフォントでは全グリフに適用されるフィーチャなので、無視してもらいたい。

その結果、上図にある上下のグリフは、InDesign上では同じに見えるが、他のアプリケーションにコピペした途端、別々の文字（符号位置）になってしまう。

Ⓑメニュー選択には、もう一つ大きな特徴がある。それは、選択した「全て」の文字にGSUB属性がつくということだ。つまりⒷを使う限り、グリフが変化するかどうかに関係なくGSUB属性が付与される（図26）。

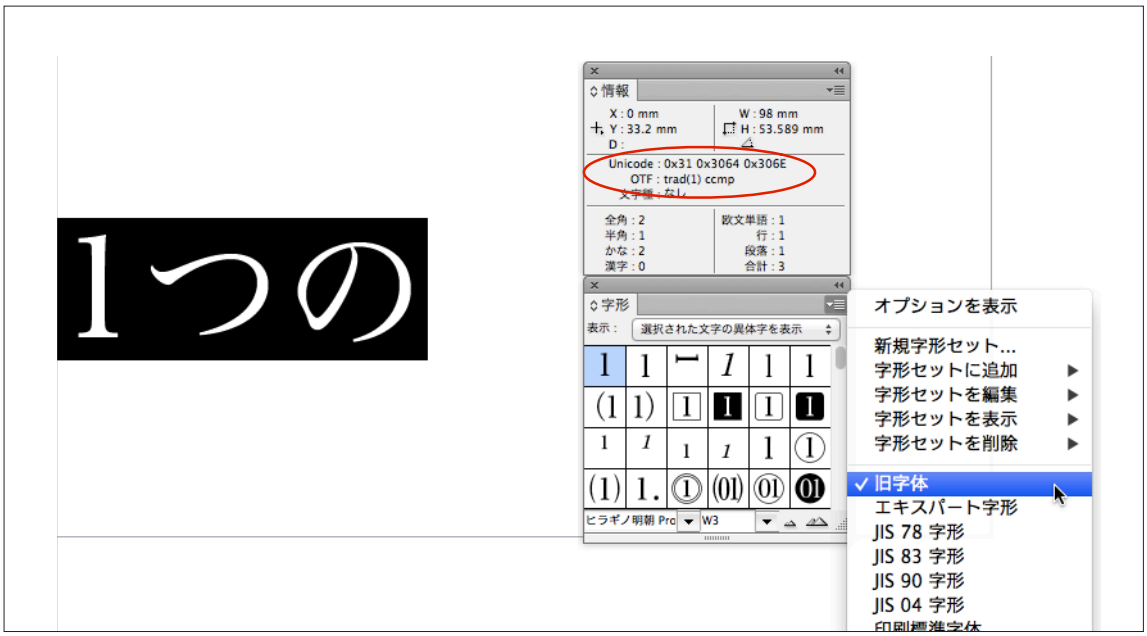

図26：数字と平仮名からなる文字列に対し、GSUBフィーチャ「trad」を適用。情報パネルでわかるように、GSUB属性が付与されている。

上図の「trad」は漢字専用のGSUBフィーチャであり、数字や平仮名には何の影響も及ぼさないはずだ。ところがそうしたこととは無関係に、Ⓑメニュー選択を使えばこれが適用できてしまう。ただしグリフには何の変化も起きない。フォントの側にそのような異体字情報がないからだ。さて、ここで問題なのは、こうして付与されたGSUB属性が意図せず顕在化することだ。「trad」を適用した「1つの」に続けて「国」（U+56FD）と入力してみよう（図27）。

図27：「国」（U+56FD）と入力したのに、隣の文字列の「trad」が続けて適用され「國」になった。情報パネルでもこの字がU+570B（國）などではなく、「U+56FD trad」という内部表現であることが確認できる。

「国」を入れたのに、なぜか「國」が表示された。これは「1つの」という文字列に付与されていたGSUB属性「trad」が、新しく入力された部分にも引き継がれるこ

とで、「國」のグリフが出現したものだ。このようにⒷメニュー選択はグリフを変化させられるかどうかに関係なくGSUB属性が付与でき、さらに隣接する箇所に新規入力するとGSUB属性が引き継がれるという特徴をもつ。

なお、字形パネルや文字パネルのメニューではGSUB属性を付与する対象は選択文字列に限定されるが、「段落スタイル」及び「文字スタイル」にある「詳細文字形式：異体字」で指定すれば、スタイルの一部として特定のGSUB属性が一括して付与することができる（図28）。

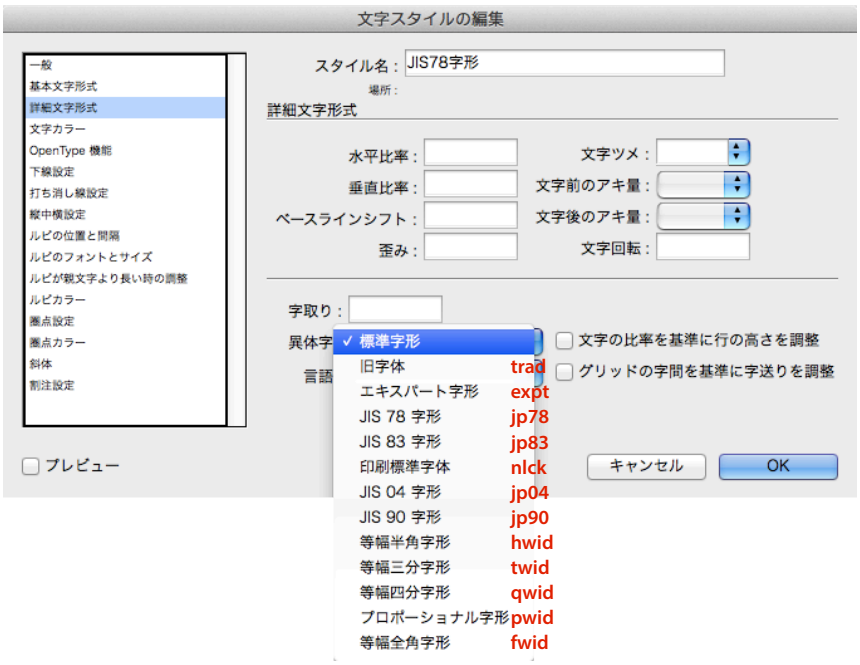

図28：「文字スタイル」で一括適用できるGSUBフィーチャ。「段落スタイル」も同様。

ただし、これらでは対象となったスタイルに含まれる「全ての文字」に対してGSUB属性が付与される。前図のように新しく入力した文字にも適用されることもあって、意図しないグリフが異体字に置き換わってしまう可能性が高い。原稿に含まれる全ての文字の異体字グループが把握できていない場合は、スタイル設定によってGSUBフィーチャを適用するのは、十分に慎重であるべきだろう。

1.2.2.3 グリフの入力／置換方法に由来する内部表現の違い

ここまで異体字を入力／置換する方法を2種類説明した。この違いによって、同じグリフ（CID）を入力した場合でも、内部表現が異なる場合があることに注意したい。Ⓐ字形パネル・ダブルクリックでは、入力結果がㇷUnicode符号位置だけの内部表現になる可能性がある。しかしⒷメニュー選択は、グリフにGSUB属性を付与するだけなので、ㇷにはなり得ず、ㇳ単独置換やㇴ合字置換になる（ㇵ選択置換の属性を与えるメニュー項目はない）。

たとえば、Ⓐ字形パネル・ダブルクリックでCID+13523「𠂉」を入力した場合、小塚明朝Pr6Nではこのグリフに Unicode符号位置U+2090Eが対応づけられているので、入力後の内部表現としてはⓐUnicode符号位置となる（図29）。

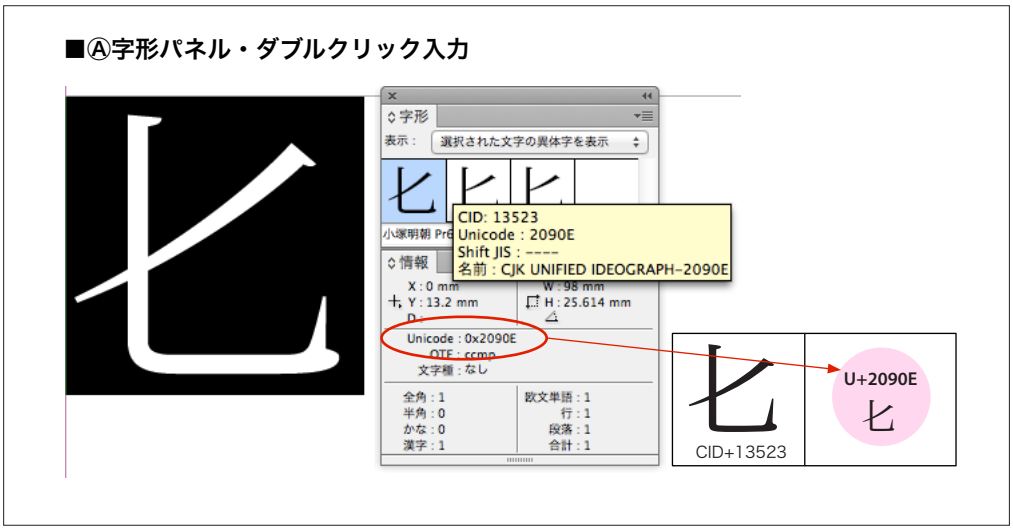

図29：Ⓐ字形パネル・ダブルクリックで異体字を入力した場合の内部表現

他方、同じCID+13523「𠂉」をⒷメニューからの適用で使う場合、まずIMなどでUnicode符号位置U+5315「𠂉」を入力、グリフを選択してしておき、ついでパネルメニューから「JIS78字形」(jp78) か「JIS83字形」(jp83) を選択することでCID+13523「𠂉」に置き換わる（図30）。

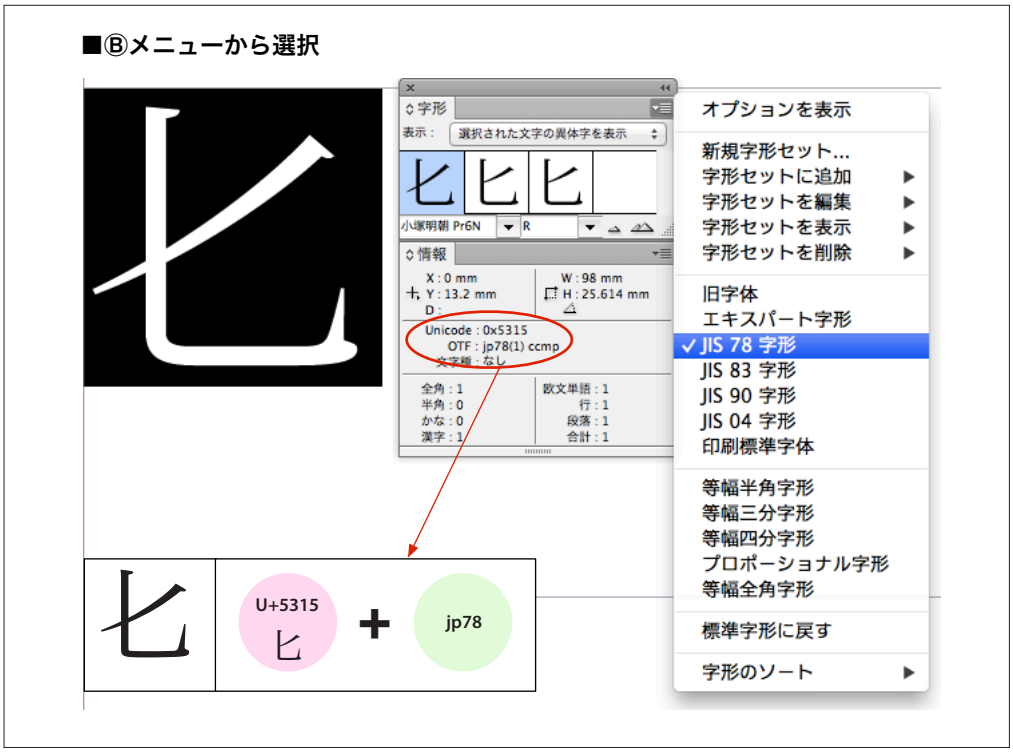

図30：Ⓑメニューからの選択でグリフ置換した場合の内部表現

しかし、図29と図30とでは外見は同じCID+13523「𠂉」であっても、内部表現が異なり、親字も変わっている。こうしたことも次の部で説明するGSUBフィーチャ文字化けの原因ともなりうる。InDesignでは同じだったグリフが、他のアプリケーションにコピペした途端、違う符号位置の文字として現れるからだ。

第2部
GSUBフィーチャ文字化けの原理とそのリスト

2.1　GSUBフィーチャ文字化けの基本原理

第1部まで、OpenTypeフォントとInDesignの内部構造について少し立ち入った説明をした。この部ではそうした内部構造の結果として発生するGSUBフィーチャ文字化けが、具体的にどういうグリフで発生するのかをリストにして提供する。

その前に、まず文字化けの原理を概括しておく。前部でも繰り返し述べたが、InDesignにおいてGSUBフィーチャがついている文字は、他のアプリケーションにコピペすると、高い確率で別のグリフ（CID）に化ける。それはなぜか。

第1部で述べたようにInDesignの内部表現には、㋐Unicode符号位置のみ、㋑単独置換、㋒合字置換、㋓選択置換、㋔CID直接指定の5種類がある。このうち㋐は、1つのグリフが1つのUnicode符号位置に対応するシンプルな内部表現だ。これは汎用性が高く、他のアプリケーションにコピペしても特殊な場合を除き化けることはない。

しかし残りの㋑〜㋔はInDesignだけが理解できる独自の表現方法だ。たとえ外見はテキスト・データのように見えても、InDesign以外のアプリケーションはこれを認識できない。したがって、これらの内部表現の文字は他のアプリケーションにコピペすると、親字であるUnicode符号位置だけがペーストされることになる。結果としてコピペの前後でグリフが変わってしまう。これがGSUBフィーチャ文字化けだ。

ここで重要なことは、グリフの外見だけから内部表現の違いを判別することは不可能ということだ。もちろん、単純に全選択してコピペをしようものなら、ペースト後にはどこが化けたか分からないテキストが出現することになる。

特にGSUBフィーチャが「aalt」の場合、「1.2.2.1 Ⓐ字形パネル・ダブルクリック」で述べた親字決定のロジックにより、普段はまず目にしない難解な漢字が親字になっているかもしれない。そうした文字がペースト後にいきなり出現すれば、驚く人も多いだろう（図31）。

図31：InDesignにおける「U+279B4 aalt 3」という内部表現で表されたグリフ「齊」（左）を、テキストエディットにコピペした結果（右）。GSUBフィーチャがとれることで、まったく違うグリフ「詧」に化けてしまった。これがGSUBフィーチャ文字化けだ。

次にリストについて説明する。これらはかなり大部のものなので、本稿には含めず別添の形で公開することにした。次ページに掲載したのは、個々のリストのタイトルであり、それぞれには配布先へのリンクが埋め込まれている。

構成についても一言述べておこう。リストは、第1部で説明したⒶ字形パネル・ダブルクリック、及びⒷメニュー選択という、2つの入力／置換方法ごとに大別してある。Ⓐはさらに CMap の種類が異なる5つのフォントごとに分かれている（→1.1.1 cmap テーブルと CMap ファイル）。一方、ⒷはGSUBフィーチャごとに分けてある。

リストの冒頭においた「フォント変更にともなう選択置換の文字化けについて」では、字形パネルの基本原理とそこでの内部表現を糸口にして、選択置換のGSUBフィーチャ「aalt」「nalt」が原因となる少し複雑な文字化けについて解説する。GSUBフィーチャ文字化けの、一歩踏み込んだ理解に役立ててほしい。

資料リンク

フォント変更にともなう選択置換の文字化けについて

InDesignの字形パネルから異体字を入力する2つの方法

選択置換のしくみとInDesignにおける内部表現

フォントの変更による文字化け

XMDFビルダーにおけるIDMLデータの文字化けリスト

InDesignの字形パネルから入力した文字を他のアプリにコピペしたときに化ける例

Ryumin Pro

Ryumin Pr6

Kozuka Pr6N

Hiragino Pro

Hiragino ProN 8.10

InDesignのメニュー選択でグリフ置換した文字を他のアプリにコピペしたときに化ける例

trad（旧字体）／expt（エキスパート字形）／jp78（JIS78字形）
jp83（JIS83字形）／jp90（JIS90字形）／nlck（印刷標準字体）
dlig（任意の合字）／liga（欧文合字）／sups（上付き文字）
subs/sinf（下付き文字）

付録　参考文献

■書籍

- Yannis Haralambous, P. Scott Horne（trans）, *Fonts & Encodings*, California: O'Reilly Media, 2007.
- 『カラー図解 DTP &印刷スーパーしくみ事典 2012年度版』ワークスコーポレーション、2012年
- ケン・ランディ著、小松章／逆井克己訳『CJKV 日中韓越情報処理』オライリー・ジャパン、2002年

■ウェブサイト

- AdobeSystems., *"The Adobe-Japan1-6 Character Collection"*, 15 Feb. 2008（http://www.adobe.com/content/dam/Adobe/en/devnet/font/pdfs/5078.Adobe-Japan1-6.pdf）.
- Microsoft Corp., *"OpenType specification"*, 21 Sep. 2009（http://www.microsoft.com/typography/otspec/）.
- 狩野宏樹「フォントのしくみ」2011年（http://www.iwatafont.co.jp/news/img/about_font.pdf）
- 川幡太一「OpenType」『漢字データベースプロジェクト』（http://kanji-database.sourceforge.net/fonts/opentype.html）
- 直井靖「Mac OS Xの文字コード問題に関するメモ」（http://d.hatena.ne.jp/NAOI/）