ISO/IEC JTC1/SC2/AG1 (IRG) N2968
Date 2026−08−18

Source:	CheonHyeong Sim (沈天珩 / 沈天珩 / 沈 天珩 (しんてんぎょう) / 심천형)
Title:	MetaHan: Work in Progress
Status:	Individual Contribution on IRG #67
Action:	Only for Information

An advertisement is included here to introduce an ongoing personal project: “MetaHan”, a Chinese counterpart to GlyphWiki developed by myself, is currently under active development. The Hangul font MetaHan Myungjo used in this document is the first finished product generated by this project. While the primary focus of this project remains on CJKUI, it can also be used incidentally for designing siniform scripts.

請允許插播一則廣告。中國版字形維基「元瀚字形工坊」，本人的一個專案，正在如火如荼地開發當中。本文檔中所使用的韓文字型MetaHan Myungjo，便是由這個專案生成的第一個成品。當然，這個專案主打方向仍是漢字，只是可以順便用來設計類漢字的文種。

请允许插播一则广告。中国版字形维基“元瀚字形工坊”，本人的一个项目，正在如火如荼地开发当中。本文档中所使用的韩文字体MetaHan Myungjo，便是由这个项目生成的第一个成品。当然，这个项目主打方向仍是汉字，只是可以顺便用来设计类汉字的文种。

ここで、ひとつ宣伝を挟ませていただきます。筆者が開発を進めている「メタハン」は、グリフウィキに相当する中国発の独立プロジェクトであり、現在鋭意開発中である。本ドキュメントで使用されている韓国語フォントMetaHan Myungjoは、本プロジェクトによって生成された最初の完成品である。もちろん、本プロジェクトの主たる開発対象はあくまで漢字であるが、漢字型の書記体系の設計に副次的に活用することも可能である。

양해를 구하며, 본인의 개인 프로젝트에 대하여 한 가지 광고를 덧붙이고자 한다. 글리프위키에 대응하는 중국 독자 개발 프로젝트인 “메타한”은 현재 활발히 개발이 진행 중이다. 본 문서에 사용된 한글 글꼴 메타한명조는 이 프로젝트를 통해 생성된 첫 번째 완성품이다. 물론 이 프로젝트의 주된 대상은 여전히 한자이며, 한자형 문자 체계에도 부수적으로 활용할 수 있다.

Etymology of the Name

Thanks to Zhang Juwei for the naming suggestion. The project operates on a principle similar to MetaFont and primarily serves CJKUI typeface design. The name MetaHan combines “Meta” from “MetaFont” and “Han” from 漢字 (Hànzì; CJKUI).

What can it Do?

Given a stroke library containing over 200 predefined strokes, paired with glyph skeleton data, target vector outlines can be generated automatically via interpolation. Similar in-house development tools may exist at many font companies, but they are not made available to the public. For individual typeface developers, editing only skeleton data undoubtedly reduces both workload and technical difficulty compared with direct vector outline editing.

The project draws inspiration from the hinting mechanism in the Windows XP version of DynaLab MingLiU, which uses composite glyphs combined with hinting instructions to move outline points to designated positions, thereby significantly reducing font file size. Extracting the hinting logic from the font file itself formed the initial prototype of the project.

Since the project stores only glyph skeleton data and does not rely on the stroke outlines of MingLiU proper, copyright issues can be circumvented by redesigning a stroke library, though this process requires a considerable amount of time. In the future, with multiple distinct stroke libraries available, the same skeleton data can be used to directly generate fonts in a variety of styles.

In addition to DynaLab MingLiU, DynaLab KaiU employs the same underlying logic. In the future, with MetaHan, users will be able to edit not only Serif typefaces with ease, but also Kǎi typefaces. Naturally, Serif and Kǎi typefaces require two separate sets of skeleton data, as the spatial arrangement of CJKUI strokes differs drastically between the two styles. These two skeleton datasets will be fully isolated, with no mutual influence or cross-reference. For the equally widely used Sans typeface, most of the Serif skeleton data can be reüsed, with only bulk post-processing adjustments required. All subsequent descriptions in this document are based on the Serif typeface.

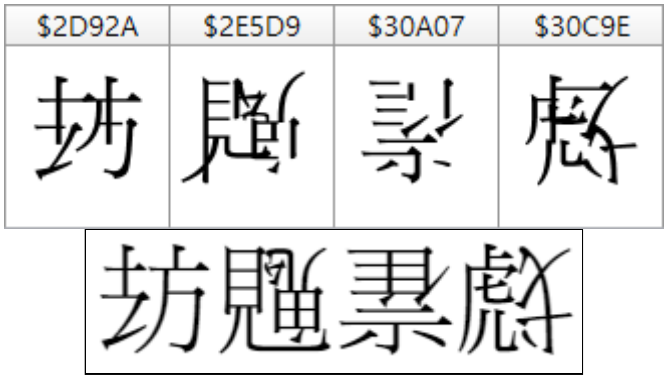
Differences from GlyphWiki

Compared with GlyphWiki created by Prof. Koichi Kamichi, MetaHan has certain limitations, but also features numerous innovations.

As one example of a limitation: the pië stroke in GlyphWiki supports arbitrary curvature adjustment, whereäs MetaHan does not offer this function, and users may only select from dozens of predefined pië strokes in the stroke library. This is nevertheless fully sufficient, as the initial dataset derived from MingLiU already covers all CJKUI in the URO (U+4E00..U+9FA5). In addition, supplementary strokes will be added to the stroke library for certain characters with special stroke forms specified in UTN #43.

Owing to this limitation, MetaHan cannot freely assemble strokes to approximate most non−CJKUI scripts in the way GlyphWiki can, such as Latin letters and Hiragana. This is, of course, outside the intended scope of the project. Even if an existing stroke set were successfully used to approximate the glyph of a given character, applying the same stroke data to a different stroke library would most likely fail. For this reason, such usage will be explicitly prohibited.

The innovations, by contrast, are far more numerous. First, regarding font file generation, MetaHan will provide two versions. The first adopts composite glyphs similar to those in MingLiU, featuring a small file size and a degree of tamper resistance, but requiring a rendering engine with hinting support. The pictures below show the appearance of such fonts when opened in font editing software and their actual rendered output, respectively. The second version uses simple glyphs with all anchor points on the outline fully resolved, delivering WYSIWYG results. In both versions, the outlines are true curves, rather than the pseudo−curves employed in GlyphWiki.



Second, stroke skeleton data in MetaHan incorporates stroke weight information, granting designers a degree of flexibility. In GlyphWiki, skeleton data does not explicitly include weight information; stroke weight is adjusted automatically based on surrounding strokes, and if the result is unsatisfactory, designers must manually modify the weight via patches, which is inconvenient. Beyond granting designers direct control over weight, MetaHan can

scale weight values in bulk via a dedicated algorithm to quickly generate fonts of various weights, and can even generate “gvar” tables to produce variable fonts, which are all unavailable in GlyphWiki.

Third, the stroke order recorded in skeleton data can also be utilized. It can be used to indicate the stroke order of each character, enabling the generation of stroke–by–stroke fonts for educational and other applications. Furthermore, it can reflect regional differences in stroke order for the same character. For instance, for the character 王, the second stroke is horizontal in mainland China, but vertical in Japan. In principle, GlyphWiki is fully capable of implementing this feature as well, but it did not adopt this rule from its inception, making such a change impracticable given its current scale.

The foregoing comparison is in no way intended to favor one platform over the other. I am in fact a heavy user of GlyphWiki, having edited over 100,000 glyphs. It is an undeniable objective fact that both GlyphWiki and MetaHan have their respective strengths and weaknesses. I hope that after MetaHan is officially launched, designers will flexibly utilize both platforms to complement one another and leverage their respective advantages.

At present, MetaHan does not yet have a visual editor, and adjustments can only be made by manually modifying skeleton data, as illustrated in the picture:

Generate!

```
200,10,149,22,96
300,53,209,10,-26
583,52,149,10,8,40
405,54,123,9,85,79
308,158,68,10,-24
202,106,-10,158
623:1,106,-2,10,122,206,150
202,106,29,158
202,106,68,158
305,177,192,10,100
202,177,192,230
610,230,192,10,214,-24,186
202,177,153,230
202,177,114,230
```

Generate!

```
200,10,149,26,96
300,53,209,12,-26
583,52,149,12,8,40
405,54,123,11,85,79
308,158,68,12,-24
202,106,-10,158
623:1,106,-2,12,122,206,150
202,106,29,158
202,106,68,158
305,177,192,12,100
202,177,192,230
610,230,192,12,214,-24,186
202,177,153,230
202,177,114,230
```

```
200,10,149,31,96
300,53,209,16,-26
583,52,149,16,8,40
405,54,123,15,85,79
308,158,68,16,-24
202,106,-10,158
623:1,106,-2,16,122,206,150
202,106,29,158
202,106,68,158
305,177,192,16,100
202,177,192,230
610,230,192,16,214,-24,186
202,177,153,230
202,177,114,230
```

Generate!

Once the visual editor is completed, the editing experience will be nearly identical to that of GlyphWiki.

There is another important purpose in using the character 椢 as an example here: to demonstrate the stroke rotational logic. As can be seen in the pictures, this rotated shùgōu has the stroke ID 623, and its direction is indicated by the notation 623:1. The ":1" suffix affects only the single stroke designated by that line of data, and is completely independent of the presence of other strokes and their ordering. This is not achievable in GlyphWiki. In GlyphWiki, a "rotate" operation itself occupies one line of data. This line contains no visible stroke, but rotates all strokes within the specified range in the lines above it. This mechanism is inherently strongly coupled with stroke order. Since we have decided that data order shall be used to indicate character stroke order, a stroke rotational method independent of data order must be adopted. Naturally, this approach also has drawbacks: a ":1" suffix is required for every rotated stroke, and batch rotations over a selected range is not supported.

As for mirroring operations, MetaHan does not support them. This is because MetaHan stretches and interpolates pre−designed outlines from the stroke library, and direct mirroring would alter the direction of the outlines. For the WYSIWYG font version, this would be straightforward to resolve by reversing the order of anchor points on the outline to restore the correct direction. However, hinting does not support reördering anchor points, so this approach cannot be implemented. All mirrored strokes appearing in CJKUI will be classified as special strokes and added as supplementary entries, with IDs uniformly starting with 8.

About MetaHan Myungjo

As stated earlier, this is the first finished product of MetaHan. Since the visual editor has not yet been developed, the Hangul font MetaHan Myungjo was designed entirely through manual adjustment of skeleton data.

Its stroke library, purpose-built for Hangul typeface design, contains only over 20 basic strokes. The glyph skeleton was designed with reference to the Chosunilbo typeface; while the two appear similar at first glance, there are in fact numerous differences. Most importantly, MetaHan Myungjo provides robust support for YetHangul. The vast majority of YetHangul fonts available on the Internet perform only simple classification during component assembly, based solely on the positional structure of the Vowel Jamo and the presence or absence of a Trailing Jamo, with no further fine-tuning for width and height. Syllables assembled in this way are legible but lack visual harmony. For example, the final consonants of 긴, 길 and 깋 clearly differ in height; for aesthetic balance, the 기 component above them must be compressed at different ratios. MetaHan Myungjo takes this into account in its design, with extremely detailed classification, for instance, Leading Jamos are divided into 66 categories in total. By comparison, in the YetHangul portion of Malgun Gothic, which comes with the Windows system, Leading Jamos are divided into only 5 categories, a difference of an order of magnitude.

Compared with Hangul fonts previously designed by myself using FontCreator, MetaHan Myungjo involves approximately 7 times the design workload, yet takes only about 2.5 times the total time to complete. This directly demonstrates the advantages brought by the MetaHan project. The font is available for download via the link below. As of the completion of this document, the current version is 1.00. The download link will remain unchanged for subsequent updates.

http://ko.cheonhyeong.com/fonts/mthmyung.ttf

In the future, two additional Hangul stroke libraries of distinct styles will be developed, transforming MetaHan Myungjo into a variable font that supports arbitrary blending between three different styles.

Following the release of MetaHan Myungjo, someone remarked in jest that the “Han” in “MetaHan” stands for 韓 (Hangul) rather than 漢 (CJKUI). This is clearly a misunderstanding of the positioning of MetaHan. The reasoning is not difficult to follow: the workload involved in a CJKUI font is on an entirely different order of magnitude compared with a Hangul font. It is simply impossible for a single person to produce a full CJKUI font in a short time before the complete system is launched. This will require collective contribution from the community after the system goes live.

(End of document; Traditional Chinese from the next page)

名稱由來

感謝張巨瑋提供的命名建議。由於專案的工作原理類似於MetaFont，而又主要服務於漢字字型的設計，取MetaFont的Meta和「漢字」的「漢」，組合得到MetaHan。Meta意譯為「元」；Han重新音譯為「瀚」，同時兼顧其字義，表明漢字數量之浩瀚，這便是中文名「元瀚」的由來。

它能做什麼？

給定一套包含兩百多個特定筆劃的筆劃庫，配合字形的骨架資料，即可自動插值得到目標向量輪廓。也許許多字型公司內部都存在類似的開發工具，但那並不會對外公開；對於個人字型開發者而言，比起直接編輯向量輪廓本身，只編輯骨架資料無疑能夠節省不少工作量、降低不少難度。

專案的靈感來源於WindowsXP版本華康細明體中的hinting，其利用複合字形加上hinting程式使輪廓上的點移動到特定的位置，使字型檔的體積大大縮小。將hinting程式的邏輯從字型檔中剝離出來，便得到了專案的雛形。

由於專案只存儲字形的骨架資料，而不依賴於細明體本身的筆劃輪廓，故僅需重新設計一套筆劃庫即可規避版權問題，不過這需要一定的時間。未來若有多套不同的筆劃庫，則可使用相同的骨架資料直接生成多種不同風格的字型。

此外，除了華康細明體，華康標楷體也使用了相同的邏輯。日後，利用元瀚字形工坊，我們不但能夠輕鬆編輯宋體，還能夠編輯楷體。當然，宋體和楷體不得不使用兩套不同的骨架資料，這是因為它們之間漢字筆劃的空間排佈差異巨大。這兩套骨架資料將會是完全隔開的，不會互相影響，也無法互相引用。對於同樣常用的黑體，則可以重複利用宋體的大多數骨架資料，僅需一些後期的批量調整。以下的敘述全部基於宋體。

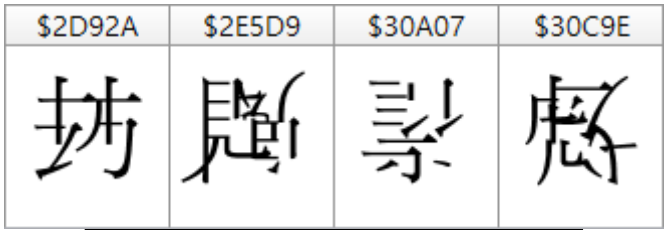
和字形維基的差異

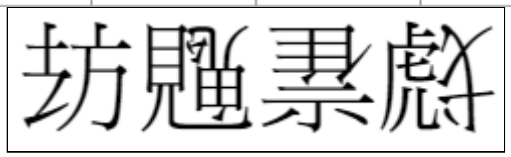
相比上地宏一教授的字形維基，元瀚字形工坊存在一些限制，但同時也有許多創新點。

先舉一個限制的例子。字形維基中的筆劃撇允許任意調節弧度，但元瀚字形工坊不可，只能在筆劃庫中已經給定的幾十種撇當中選擇。當然，這肯定是夠用的，因為細明體的初期資料裡面就已經包含了基本區（U+4E00..U+9FA5）的全部漢字。除此之外，我們還將在筆劃庫中為UTN #43中部分帶有特殊筆劃的漢字額外增補一些筆劃。

受此限制，我們也無法像字形維基那樣用足夠自由的筆劃進行拼接來擬合絕大多數非漢字文種，例如拉丁字母、平假名等。當然，這本就不在專案的工作範圍內。即使利用現有的筆劃成功擬合了某個字元的字形，把這個筆劃資料套用到另一套筆劃庫上，大概率也會失效，因此我們會直接規定不允許使用者這樣做。

而創新點就多了。首先是字型檔的生成，元瀚字形工坊將會提供兩個版本。其一為類似於細明體中那樣的複合字形，體積小，還能一定程度上防止篡改，但需要渲染引擎支援hinting，例如下圖分別展示了字型編輯軟體在打開這類字型時的樣貌以及實際渲染出的效果；其二為將輪廓上的各個錨點完全歸位的簡單字形，所見即所得。無論哪個版本，輪廓都是真正的曲線，而非字形維基那樣的偽曲線。





其次，筆劃的骨架資料當中是包含粗細資訊的，這給了設計者們一定的自由。字形維基的骨架資料中不顯式包含粗細，會自動根據周圍的筆劃調整，若調整得不滿意，設計者需要手動以補丁形式改變粗細，這並不方便。元瀚字形工坊除了給設計者們提供自由之外，將粗細資訊對應的數值以一定的演算法批量放大或縮小，還可快速生成各種不同字重的字型，甚至生成gvar表做成可變字型，這都是字形維基所無法做到的。

再者，骨架資料中各個筆劃的先後順序也可以利用起來，我們可以用它來指示每個字的筆順，生成逐筆順的字型供教學等場合使用，甚至更進一步地，反映同一個字不同地區的筆順差異，例如「王」字，中國大陸的筆順第二筆為橫，而日本的筆順第二筆為豎。其實原則上這一點字形維基也完全有能力做到，但它一開始並沒有遵循這個規則，以如今的體量就已經很難改變了。

以上並沒有任何踩一捧一的意思。甚至本人也是字形維基的重度使用者之一，曾編輯過十餘萬字形。字形維基和元瀚字形工坊各有各的優缺點，這是不可否認的客觀事實，本人希望等元瀚字形工坊正式上線之後，設計者們能夠靈活運用二者，取長補短，發揮各自的優勢。

目前元瀚字形工坊暫未製作視覺化編輯器，只能手工修改骨架資料進行調整，如圖所示：

Generate!

```
200,10,149,22,96
300,53,209,10,-26
583,52,149,10,8,40
405,54,123,9,85,79
308,158,68,10,-24
202,106,-10,158
623:1,106,-2,10,122,206,150
202,106,29,158
202,106,68,158
305,177,192,10,100
202,177,192,230
610,230,192,10,214,-24,186
202,177,153,230
202,177,114,230
```

Generate!

```
200,10,149,26,96
300,53,209,12,-26
583,52,149,12,8,40
405,54,123,11,85,79
308,158,68,12,-24
202,106,-10,158
623:1,106,-2,12,122,206,150
202,106,29,158
202,106,68,158
305,177,192,12,100
202,177,192,230
610,230,192,12,214,-24,186
202,177,153,230
202,177,114,230
```

Generate!

```
200,10,149,31,96
300,53,209,16,-26
583,52,149,16,8,40
405,54,123,15,85,79
308,158,68,16,-24
202,106,-10,158
623:1,106,-2,16,122,206,150
202,106,29,158
202,106,68,158
305,177,192,16,100
202,177,192,230
610,230,192,16,214,-24,186
202,177,153,230
202,177,114,230
```

待視覺化編輯器製作完成之後，可以獲得和字形維基近乎一致的編輯體驗。

這裡用「棚」字舉例還有一個重要的目的：展示筆劃的倒轉邏輯。從圖中可以看到，這個倒轉的豎鉤的筆劃編號是623，它以623:1的形式表明了筆劃的方向。這裡的:1影響的僅有這一行資料所指示的一個筆劃，與其它筆劃是否存在以及它們之間的順序完全無關。字形維基做不到這一點。在字形維基中，「倒轉」操作本身需要佔用一行資料，這行資料不包含任何可見的筆劃，但會倒轉它上面所有行的資料當中落在這一行資料所指定範圍內的筆劃。這顯然和順序強相關。既然我們決定資料的順序要用來指示字的筆順，那必須使用一種與資料順序無關的方式倒轉筆劃。當然，這也有缺點，對於每個被倒轉的筆劃我們都需要一個:1，無法選定範圍批量倒轉。

至於鏡像操作，元瀚字形工坊無法支援。這是因為元瀚字形工坊利用的是筆劃庫中設計過的輪廓來拉伸、插值，而直接鏡像將改變輪廓的方向。如果只是生成所見即所得版本的字型，這很好辦，讓輪廓上各個錨點的順序也倒轉一下，方向就變回來了；但hinting卻不支援變換錨點的順序，只能作罷。對於漢字中所涉及的鏡像筆劃，它們都將被歸類為特殊筆劃，額外增補，編號統一以8起始。

關於MetaHan Myungjo

如前文所述，這是元瀚字形工坊的第一個成品。由於視覺化編輯器暫未開發，韓文字型MetaHan Myungjo就是靠手工修改骨架資料調整做的設計。

這套筆劃庫也是為了韓文字型特地設計的，僅包含二十餘個基本筆劃。字形骨架參考了朝鮮日報體進行設計，風格粗看相似，但實際上也有許多不同點，而且最重要的一點是它能良好地支持古韓文。市面上能夠見到的絕大多數古韓文字型，在各個元件進行拼接時，只根據中聲的位置結構關係以及有無終聲做簡單的分類，而沒有根據它們的寬度、高度做進一步的微調，這樣拼出來的音節，能看，但不夠協調。例如긴、길、길三者的終聲高度顯然不一致，若考慮美觀，其上的ㄱ必然需要以不同比例壓縮。MetaHan Myungjo在設計時考慮到了這一點，做了非常細緻的分類，例如初聲字母一共被分為66類。Windows系統自帶的Malgun Gothic的古韓文部分中，初聲字母僅被分為5類，這是數量級的差異。

對比本人曾經用FontCreator設計過的一些韓文字型，MetaHan Myungjo的工作量大約有7倍，而總耗時卻只有約2.5倍，這直觀地體現了元瀚字形工坊專案的出現帶來的優勢。字型可于下方連結下載，截至本文檔完成時，版本為1.00。後續更新連結不變。

http://ko.cheonhyeong.com/fonts/mthmyung.ttf

未來，將會再另外設計兩套不同風格的韓文筆劃庫，將MetaHan Myungjo變成一款能在三種不同風格之間以任意比例混合的可變字型。

MetaHan Myungjo發佈後，有人戲稱道，原來MetaHan的Han不是「漢」而是「韓」。這顯然是對元瀚字形工坊的定位的誤解。其實也不難理解，漢字字型的工作量和韓文相比根本不是一個數量級，在完整的系統上線前僅憑一個人的力量是無論如何無法在短時間內做出一款漢字字型的，這還需要未來系統上線後發動群眾的力量一同添磚加瓦。

（文檔結束；下一頁起為簡體中文）

名称由来

感谢张巨玮提供的命名建议。由于项目的工作原理类似于MetaFont，而又主要服务于汉字字体的设计，取MetaFont的Meta和“汉字”的“汉”，组合得到MetaHan。Meta意译为“元”；Han重新音译为“瀚”，同时兼顾其字义，表明汉字数量之浩瀚，这便是中文名“元瀚”的由来。

它能做什么?

给定一套包含两百多个特定笔画的笔画库，配合字形的骨架数据，即可自动插值得到目标矢量轮廓。也许许多字体公司内部都存在类似的开发工具，但那并不会对外公开；对于个人字体开发者而言，比起直接编辑矢量轮廓本身，只编辑骨架数据无疑能够节省不少工作量、降低不少难度。

项目的灵感来源于WindowsXP版本华康细明体中的hinting，其利用复合字形加上hinting程序使轮廓上的点移动到特定的位置，使字体文件的体积大大缩小。将hinting程序的逻辑从字体文件中剥离出来，便得到了项目的雏形。

由于项目只存储字形的骨架数据，而不依赖于细明体本身的笔画轮廓，故仅需重新设计一套笔画库即可规避版权问题，不过这需要一定的时间。未来若有多套不同的笔画库，则可使用相同的骨架数据直接生成多种不同风格的字体。

此外，除了华康细明体，华康标楷体也使用了相同的逻辑。日后，利用元瀚字形工坊，我们不但能够轻松编辑宋体，还能够编辑楷体。当然，宋体和楷体不得不使用两套不同的骨架数据，这是因为它们之间汉字笔画的空间排布差异巨大。这两套骨架数据将会是完全隔开的，不会互相影响，也无法互相引用。对于同样常用的黑体，则可以重复利用宋体的大多数骨架数据，仅需一些后期的批量调整。以下的叙述全部基于宋体。

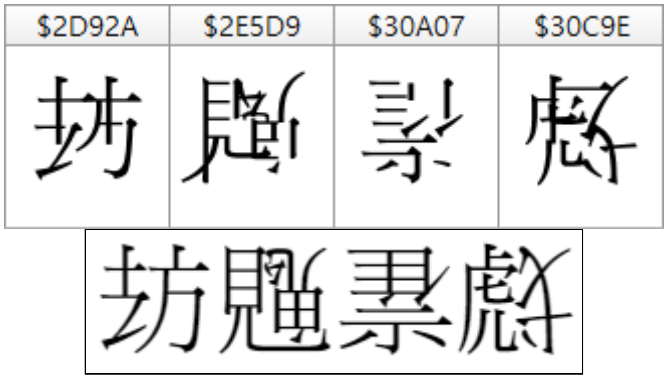
和字形维基的差异

相比上地宏一教授的字形维基，元瀚字形工坊存在一些限制，但同时也有许多创新点。

先举一个限制的例子。字形维基中的笔画撇允许任意调节弧度，但元瀚字形工坊不可，只能在笔画库中已经给定的几十种撇当中选择。当然，这肯定是够用的，因为细明体的初期数据里面就已经包含了基本区（U+4E00..U+9FA5）的全部汉字。除此之外，我们还将在笔画库中为UTN #43中部分带有特殊笔画的汉字额外增补一些笔画。

受此限制，我们也无法像字形维基那样用足够自由的笔画进行拼接来拟合绝大多数非汉字文种，例如拉丁字母、平假名等。当然，这本就不在项目的工作范围内。即使利用现有的笔画成功拟合了某个字符的字形，把这个笔画数据套用到另一套笔画库上，大概率也会失效，因此我们会直接规定不允许用户这样做。

而创新点就多了。首先是字体文件的生成，元瀚字形工坊将会提供两个版本。其一为类似于细明体中那样的复合字形，体积小，还能一定程度上防止篡改，但需要渲染引擎支持hinting，例如下图分别展示了字体编辑软件在打开这类字体时的样貌以及实际渲染出的效果；其二为将轮廓上的各个锚点完全归位的简单字形，所见即所得。无论哪个版本，轮廓都是真正的曲线，而非字形维基那样的伪曲线。



其次，笔画的骨架数据当中是包含粗细信息的，这给了设计者们一定的自由。字形维基的骨架数据中不显式包含粗细，会自动根据周围的笔画调整，若调整得不满意，设计者需要手动以补丁形式改变粗细，这并不方便。元瀚字形工坊除了给设计者们提供自由之外，将粗细信息对应的数值以一定的算法批量放大或缩小，还可快速生成各种不同字重的字体，甚至生成gvar表做成可变字体，这都是字形维基所无法做到的。

再者，骨架数据中各个笔画的先后顺序也可以利用起来，我们可以用它来指示每个字的笔顺，生成逐笔顺的字体供教学等场合使用，甚至更进一步地，反映同一个字不同地区的笔顺差异，例如“王”字，中国大陆的笔顺第二笔为横，而日本的笔顺第二笔为竖。其实原则上这一点字形维基也完全有能力做到，但它一开始并没有遵循这个规则，以如今的体量就已经很难改变了。

以上并没有任何踩一捧一的意思。甚至本人也是字形维基的重度用户之一，曾编辑过十余万字形。字形维基和元瀚字形工坊各有各的优缺点，这是不可否认的客观事实，本人希望等元瀚字形工坊正式上线之后，设计者们能够灵活运用二者，取长补短，发挥各自的优势。

目前元瀚字形工坊暂未制作可视化编辑器，只能手工修改骨架数据进行调整，如图所示：

Generate!

```
200,10,149,22,96
300,53,209,10,-26
583,52,149,10,8,40
405,54,123,9,85,79
308,158,68,10,-24
202,106,-10,158
623:1,106,-2,10,122,206,150
202,106,29,158
202,106,68,158
305,177,192,10,100
202,177,192,230
610,230,192,10,214,-24,186
202,177,153,230
202,177,114,230
```

Generate!

```
200,10,149,26,96
300,53,209,12,-26
583,52,149,12,8,40
405,54,123,11,85,79
308,158,68,12,-24
202,106,-10,158
623:1,106,-2,12,122,206,150
202,106,29,158
202,106,68,158
305,177,192,12,100
202,177,192,230
610,230,192,12,214,-24,186
202,177,153,230
202,177,114,230
```

Generate!

```
200,10,149,31,96
300,53,209,16,-26
583,52,149,16,8,40
405,54,123,15,85,79
308,158,68,16,-24
202,106,-10,158
623:1,106,-2,16,122,206,150
202,106,29,158
202,106,68,158
305,177,192,16,100
202,177,192,230
610,230,192,16,214,-24,186
202,177,153,230
202,177,114,230
```

待可视化编辑器制作完成之后，可以获得和字形维基近乎一致的编辑体验。

这里用“棚”字举例还有一个重要的目的：展示笔画的倒转逻辑。从图中可以看到，这个倒转的竖钩的笔画编号是623，它以623:1的形式表明了笔画的方向。这里的:1影响的仅有这一行数据所指示的一个笔画，与其它笔画是否存在以及它们之间的顺序完全无关。字形维基做不到这一点。在字形维基中，“倒转”操作本身需要占用一行数据，这行数据不包含任何可见的笔画，但会倒转它上面所有行的数据当中落在这一行数据所指定范围内的笔画。这显然和顺序强相关。既然我们决定数据的顺序要用来指示字的笔顺，那必须使用一种与数据顺序无关的方式倒转笔画。当然，这也有缺点，对于每个被倒转的笔画我们都需要一个:1，无法选定范围批量倒转。

至于镜像操作，元瀚字形工坊无法支持。这是因为元瀚字形工坊利用的是笔画库中设计过的轮廓来拉伸、插值，而直接镜像将改变轮廓的方向。如果只是生成所见即所得版本的字体，这很好办，让轮廓上各个锚点的顺序也倒转一下，方向就变回来了；但hinting却不支持变换锚点的顺序，只能作罢。对于汉字中所涉及的镜像笔画，它们都将被归类为特殊笔画，额外增补，编号统一以8起始。

关于MetaHan Myungjo

如前文所述，这是元瀚字形工坊的第一个成品。由于可视化编辑器暂未开发，韩文字体MetaHan Myungjo就是靠手工修改骨架数据调整做的设计。

这套笔画库也是为了韩文字体特地设计的，仅包含二十余个基本笔画。字形骨架参考了朝鲜日报体进行设计，风格粗看相似，但实际上也有许多不同点，而且最重要的一点是它能良好地支持古韩文。市面上能够见到的绝大多数古韩文字体，在各个组件进行拼接时，只根据中声的位置结构关系以及有无终声做简单的分类，而没有根据它们的宽度、高度做进一步的微调，这样拼出来的音节，能看，但不够协调。例如긴、길、긇三者的终声高度显然不一致，若考虑美观，其上的기必然需要以不同比例压缩。MetaHan Myungjo在设计时考虑到了这一点，做了非常细致的分类，例如初声字母一共被分为66类。Windows系统自带的Malgun Gothic的古韩文部分中，初声字母仅被分为5类，这是数量级的差异。

对比本人曾经用FontCreator设计过的一些韩文字体，MetaHan Myungjo的工作量大约有7倍，而总耗时却只有约2.5倍，这直观地体现了元瀚字形工坊项目的出现带来的优势。字体可于下方链接下载，截至本文档完成时，版本为1.00。后续更新链接不变。

http://ko.cheonhyeong.com/fonts/mthmyung.ttf

未来，将会再另外设计两套不同风格的韩文笔画库，将MetaHan Myungjo变成一款能在三种不同风格之间以任意比例混合的可变字体。

MetaHan Myungjo发布后，有人戏称道，原来MetaHan的Han不是“汉”而是“韩”。这显然是对元瀚字形工坊的定位的误解。其实也不难理解，汉字字体的工作量和韩文相比根本不是一个数量级，在完整的系统上线前仅凭一个人的力量是无论如何无法在短时间内做出一款汉字字体的，这还需要未来系统上线后发动群众的力量一同添砖加瓦。

（文档结束；下一页起为日语）

名称の由来

張巨瑋による命名案に感謝する。本プロジェクトはMetaFontと同様の原理で動作し、主に漢字フォントのデザインを対象としていることから、MetaFontの「メタ」と「漢字(hàn)」の「ハン」を組み合わせて「メタハン」と命名された。

何ができるのか？

200種類以上の既定の画からなる画ライブラリと、グリフの骨格データを組み合わせることで、自動補間により目的のベクトル輪郭を生成することができる。多くのフォント会社には同様の社内開発ツールが存在する可能性があるが、一般に公開されてはいない。個人のフォント開発者にとって、ベクトル輪郭を直接編集する場合と比較し、骨格データのみを編集するのは作業量と難易度を大幅に削減できることは明らかである。

本プロジェクトの着想は、WindowsXP版のダイナラブMingLiUにおけるヒンティングに由来する。同フォントは合成グリフとヒンティング命令を組み合わせ、輪郭上の点を指定された位置に移動させることで、ファイルサイズを大幅に削減している。ヒンティングのロジックをフォントファイルから分離したものが、本プロジェクトの原型となった。

本プロジェクトはグリフの骨格データのみを保存し、MingLiU自体の画の輪郭には依存しないため、画ライブラリを再デザインするだけで著作権上の問題を回避できる。ただし、その実施には相当の時間がかかる。将来的に複数の異なる画ライブラリが整備されれば、同一の骨格データから複数のスタイルのフォントを直接生成することが可能となる。

なお、ダイナラブMingLiUのほか、ダイナラブKaiUも同じロジックを採用している。今後、メタハンを用いることで、明朝体だけでなく楷書体も容易に編集できるようになる。もちろん、明朝体と楷書体では漢字画の空間配置が大きく異なるため、それぞれ別個の骨格データセットが必要となる。これら2種の骨格データは完全に分離され、相互に影響を与えず、相互参照もできない。同様に広く用いられるゴシック体については、明朝体の骨格データの大部分を再利用し、一部の一括調整を加えるだけで対応可能である。本稿以降の記述は、すべて明朝体を前提としている。

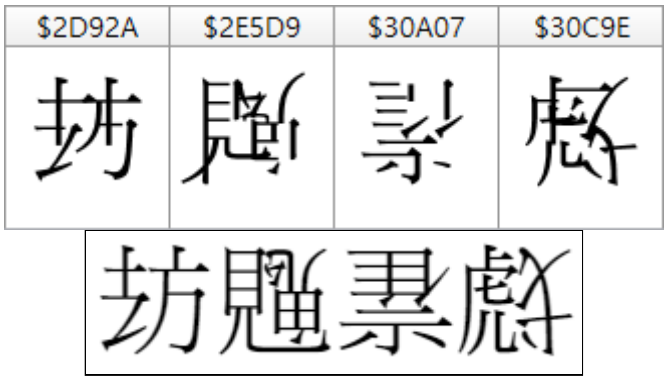
グリフウィキとの相違点

上地宏一教授のグリフウィキと比較すると、メタハンにはいくつかの制約がある一方、多くの革新的な特長も備わっている。

まず制約の例を挙げる。グリフウィキでは左払いという画の曲率を任意に調整できるが、メタハンではこの機能に対応しておらず、画ライブラリにあらかじめ用意された数十種類の左払いの中から選択する形となる。ただし、MingLiU由来の初期データに既にURO（U+4E00..U+9FA5）の全漢字が含まれていることから、実用上十分な水準である。このほか、UTN #43に記載された特殊な画を持つ一部の漢字に対応するため、画ライブラリに追加の画を補充する予定である。

この制約により、グリフウィキのように自由に画を組み合わせてラテン文字やひらがなといった大半の非漢字書記体系を近似的に再現することはできない。もちろん、これは本プロジェクトの本来の目的外の事柄である。既存の画で何らかの文字の字形を近似的に再現できたとしても、そのデータを別の画ライブラリに適用すると大半の場合うまく機能しなくなるため、こうした利用は明示的に禁止する方針である。

一方、革新的な特長は数多く存在する。第一に、フォントファイルの生成方式について、メタハンは2種類のバージョンを提供する。1つはMingLiUと同様の合成グリフ方式であり、ファイルサイズが小さく、ある程度の改ざん防止効果もあるが、ヒンティングに対応したレンダリングエンジンが必要となる。下図は、この方式のフォントをフォント編集ソフトで開いた際の表示と、実際のレンダリング結果をそれぞれ示したものである。もう1つは、輪郭上のすべての点を完全に確定させた単純グリフ方式であり、WYSIWYGを実現する。いずれのバージョンにおいても、輪郭はグリフウィキのような偽曲線ではなく、真の曲線で構成される。



第二に、画の骨格データには太さの情報が含まれており、デザイン者に一定の自由度を提供する。グリフウィキの骨格データには太さ情報が明示的に含まれておらず、周囲の画に応じて自動的に調整される。調整結果が意に沿わない場合、デザイン者はパッチ形式で手動で太さを変更する必要があり、利便性に欠ける。メタハンではデザイン者が太さを自由に制御できるだけでなく、太さの数値を専用のアルゴリズムで一括して拡大・縮小することで、複数のウェイトのフォントを迅速に生成できる。さらにgvarテーブルを生成して

可変フォントを作成することも可能であり、これらはいずれもグリフウィキでは実現できない機能である。

第三に、骨格データに記録された画の順序を活用することもできる。これを利用して各文字の筆順を示し、筆順表示対応フォントを生成して教育用途などに活用できるほか、同一文字における地域ごとの筆順の差異を反映させることもできる。例えば「王」という文字の場合、中国大陸では2画目が横画であるのに対し、日本では2画目が縦画となる。原則的にはグリフウィキもこの機能を実装することは十分可能であるが、当初からこの規則を採用していなかったため、現在の規模では仕様を変更することが困難となっている。

以上の比較は、いずれか一方を優遇したり貶めたりする意図はまったくない。筆者自身もグリフウィキのヘビーユーザーであり、これまでに10万を超えるグリフを編集してきた。グリフウィキとメタハンにはそれぞれ長所と短所があることは、否定できない客観的な事実である。メタハンが正式にリリースされた後、デザイン者たちが両者を柔軟に活用し、互いの長所を生かし短所を補い合うことを期待する。

現在のメタハンにはまだビジュアルエディターが実装されておらず、図に示す通り骨格データを手動で編集して調整を行うしかない。

Generate!

```
200,10,149,22,96
300,53,209,10,-26
583,52,149,10,8,40
405,54,123,9,85,79
308,158,68,10,-24
202,106,-10,158
623:1,106,-2,10,122,206,150
202,106,29,158
202,106,68,158
305,177,192,10,100
202,177,192,230
610,230,192,10,214,-24,186
202,177,153,230
202,177,114,230
```

Generate!

```
200,10,149,26,96
300,53,209,12,-26
583,52,149,12,8,40
405,54,123,11,85,79
308,158,68,12,-24
202,106,-10,158
623:1,106,-2,12,122,206,150
202,106,29,158
202,106,68,158
305,177,192,12,100
202,177,192,230
610,230,192,12,214,-24,186
202,177,153,230
202,177,114,230
```

Generate!

```
200,10,149,31,96
300,53,209,16,-26
583,52,149,16,8,40
405,54,123,15,85,79
308,158,68,16,-24
202,106,-10,158
623:1,106,-2,16,122,206,150
202,106,29,158
202,106,68,158
305,177,192,16,100
202,177,192,230
610,230,192,16,214,-24,186
202,177,153,230
202,177,114,230
```

ビジュアルエディターが完成すれば、グリフウィキとほぼ同様の編集体験が得られるようになる。

ここで「棚」の字を例に挙げるのには、もう1つ重要な目的がある。それは画の回転ロジックを説明することである。図からわかるように、この回転した撓棒の画番号は623であり、623:1という形式で画の向きが示されている。この:1は、その行のデータが示す1つの画にのみ作用し、他の画の有無や画同士の順序とは完全に無関係である。これはグリフウィキでは実現できない仕組みである。グリフウィキにおいて「回転」操作自体が1行のデータを占める。この行には可視の画は含まれないが、それより上の行のデータのうち、指定された範囲内の画をすべて回転させる。この方式は明らかに画の順序と強く結びついている。我々はデータの順序を筆順の表示に用いる方針であるため、データの順序に依存しない方法で画を回転させる必要がある。もちろんこの方式にも欠点はあり、回転させる画ごとに:1を付与する必要があり、範囲を指定して一括で回転させることはできない。

また、鏡映操作については、メタハンは対応していない。これはメタハンが画ライブラリ内でデザイン済みの輪郭を伸縮・補間して利用しているのに対し、単純な鏡映処理では輪郭の向きが変化してしまうためである。WYSIWYG版のフォントであれば、輪郭上の点の順序を反転させることで向きを元に戻せるため対応は容易だが、ヒンティングは点の順序変更に対応していないため、実装を断念せざるを得ない。漢字に登場する鏡映の画については、すべて特殊画として分類して追加実装し、番号はすべて8から始まる体系とする。

MetaHan Myungjoについて

前述の通り、これはメタハンによる最初の完成品である。ビジュアルエディターがまだ未開発であるため、ハングルフォントMetaHan Myungjoは骨格データを手動で編集・調整することでデザインされた。

この画ライブラリもハングルフォントにわざとデザインされており、20種類余りの基本画のみで構成されている。グリフの骨格は朝鮮日報体を参考にデザインされており、一見するとスタイルが似ているが実際には多くの相違点がある。そして最も重要な点として、古ハングルによく対応している。一般に出回っている古ハングル対応フォントの大半は、構成部品を組み合わせる際、中声の位置的構造と終声の有無に基づいて単純な分類を行うだけで、幅や高さに応じた詳細な微調整が行われていない。こうして組み立てられた音節は一応判読はできるものの、視覚的な調和に欠ける。例えば긴、길、길の3つは終声の高さが明らかに異なっており、美観を考慮すれば上部のㄱをそれぞれ違う比率で圧縮する必要がある。MetaHan Myungjoのデザインではこの点が考慮されており、極めて詳細な分類が行われている。一例として、初声字母は全部で66種類に分類されている。Windows標準搭載のMalgun Gothicの古ハングル部分では初声字母がわずか5種類に分類されており、桁違いの細かさである。

筆者が以前FontCreatorでデザインした一部のハングルフォントと比較すると、MetaHan Myungjoの作業量は約7倍に達するが、総作業時間は約2.5倍にとどまる。これはメタハンというプロジェクトの登場がもたらした優位性を端的に示している。本フォントは以下のリンクからダウンロード可能である。本ドキュメント作成時点でのバージョンは1.00であり、今後の更新でもリンクは変更されない。

http://ko.cheonhyeong.com/fonts/mthmyung.ttf

将来的には、さらに2種類の異なるスタイルのハングル画ライブラリをデザインし、MetaHan Myungjoを3種類のスタイルを任意の比率で混合できる可変フォントへと発展させる予定である。

MetaHan Myungjoの公開後、MetaHanのHanは漢ではなく韓のことだったのかと冗談めかして言う者もいた。これは明らかにメタハンの位置づけに対する誤解である。とはいえ無理もないことで、漢字フォントの作業量はハングルフォントとは比較にならないほど膨

大であり、システム全体が公開される前に個人の力だけで短期間に漢字フォントを1本作り上げることは到底不可能である。これは今後システムが公開された後、コミュニティーの力を結集して共同で構築していく必要がある。

（文書の終わり；次のページからは韓国語）

명칭의 유래

장거위 님의 명명 제안에 감사한다. 본 프로젝트는 작동 원리가 MetaFont와 유사하고 주로 한자 폰트 디자인을 지원하므로, MetaFont의 “메타”와 “한자”의 “한”을 결합하여 “메타한”으로 명명하였다.

무엇을 할 수 있는가?

200여 개의 지정된 획으로 구성된 획 라이브러리와 글리프 골격 데이터를 결합하면 자동 보간을 통해 목표 벡터 외곽선을 생성할 수 있다. 다수의 폰트 제작사에 유사한 사내 개발 도구가 존재할 수 있으나, 이는 대중에 공개되지 않는다. 개인 폰트 개발자에게 직접 벡터 외곽선을 편집하는 방식에 비해 골격 데이터만 편집하는 방식은 작업량을 상당히 줄이고 난이도를 낮추는 것은 명백하다.

이 프로젝트는 Windows XP 버전 다이나랩 MingLiU의 힌팅에서 영감을 얻었다. MingLiU는 합성 글리프와 힌팅 명령을 이용해 외곽선의 점을 지정된 위치로 이동시켜 폰트 파일의 용량을 크게 줄였다. 이러한 힌팅 로직을 폰트 파일에서 분리한 것이 프로젝트의 시초이다.

이 프로젝트는 글리프의 골격 데이터만 저장할 뿐 MingLiU 자체의 획 외곽선에 의존하지 않으므로, 획 라이브러리를 새로 디자인하는 것만으로 저작권 문제를 피할 수 있으나 다소 시간이 소요된다. 향후 서로 다른 여러 세트의 획 라이브러리가 갖춰지면 동일한 골격 데이터로 다양한 스타일의 폰트를 직접 생성할 수 있게 된다.

한편 다이나랩 MingLiU 외에도 다이나랩 KaiU도 동일한 로직을 사용한다. 향후 메타한을 이용하면 명조체뿐만 아니라 해서체도 손쉽게 편집할 수 있게 된다. 물론 명조체와 해서체는 한자 획의 공간 배치 차이가 크기 때문에 서로 다른 두 세트의 골격 데이터를 사용해야 한다. 이 두 골격 데이터 세트는 완전히 분리되어 서로 영향을 주지 않으며 상호 참조도 불가능하다. 마찬가지로 널리 쓰이는 고딕체의 경우 명조체 골격 데이터의 대부분을 재활용할 수 있으며, 일부 후속 일괄 조정만 추가하면 된다. 본문 이후의 서술은 모두 명조체를 기준으로 한다.

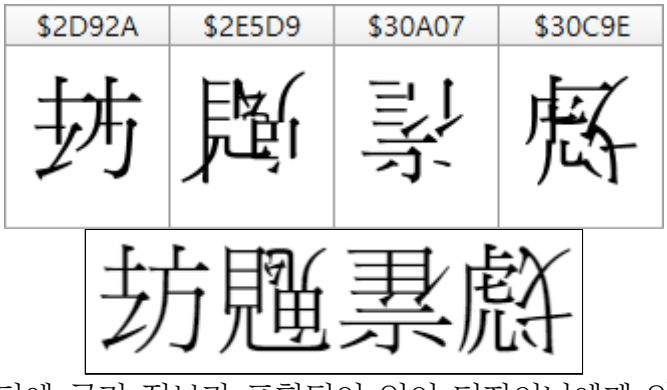
글리프위키와의 차이점

가미치고이치 교수님의 글리프위키와 비교할 때 메타한은 일부 제약 사항이 존재하는 동시에 다수의 혁신점도 지니고 있다.

먼저 제약 사항의 예를 들면, 글리프위키의 삐침 획은 임의로 곡률을 조절할 수 있지만 메타한은 지원하지 않으며, 획 라이브러리에 미리 지정된 수십 종의 삐침 획 중에서 선택해야 한다. 물론 이 정도로도 충분한데, MingLiU 기반 초기 데이터에 이미 기본한자영역(U+4E00..U+9FA5)의 모든 한자가 포함되어 있기 때문이다. 이 외에도 UTN #43에 명시된 특수 획을 지닌 일부 한자를 위해 획 라이브러리에 추가 획들을 보강할 예정이다.

이러한 제약으로 인해 메타한은 글리프위키처럼 자유롭게 획을 조합해 라틴 문자, 히라가나 등 대부분의 비한자 문자 체계를 유사하게 재현할 수는 없다. 물론 이는 본 프로젝트의 본래 목적 범위 밖의 일이다. 기존 획으로 어떤 문자의 글리프를 성공적으로 재현했다 하더라도, 해당 획 데이터를 다른 획 라이브러리에 적용하면 대부분 제대로 작동하지 않게 되므로, 이러한 사용 방식은 명시적으로 금지할 방침이다.

반면 혁신점은 훨씬 많다. 첫째, 폰트 파일 생성 방식에서 메타한은 두 가지 버전을 제공한다. 하나는 MingLiU와 유사한 합성 글리프 방식으로 용량이 작고 일정 수준 변조를 방지할 수 있으나, 힌팅을 지원하는 렌더링 엔진이 필요하다. 아래 사진은 해당 폰트를 폰트 편집 소프트웨어에서 열었을 때의 모습과 실제 렌더링 결과를 각각 보여준다. 다른 하나는 외곽선의 모든 점을 완전히 고정한 단순 글리프 방식으로 WYSIWYG 방식이다. 어떤 버전이든 외곽선은 글리프위키의 유사 곡선과 달리 진정한 곡선으로 구현된다.



둘째, 획 골격 데이터에 굵기 정보가 포함되어 있어 디자이너에게 일정한 자유도를 제공한다. 글리프위키의 골격 데이터에는 굵기 정보가 명시적으로 포함되지 않고 주변 획에 따라 자동으로 조절되는데, 조절 결과가 만족스럽지 않을 경우 디자이너가 수동으로 패치 형태로 굵기를 변경해야 하여 불편하다. 메타한은 디자이너에게 굵기 조절의 자유를 제공할 뿐만 아니라, 굵기 정보에 대응하는 수치를 특정 알고리즘으로 일괄 확대 또는 축소하여 다양한 굵기의 폰트를 빠르게 생성할 수 있으며, 나아가 gvar 테이블을 생성해 가변 폰트를 제작할 수도 있다. 이는 모두 글리프위키로는 구현할 수 없는 기능이다.

셋째, 골격 데이터에 기록된 각 획의 순서를 활용할 수도 있다. 이를 이용해 각 한자의 획순을 표시하고 획순별 폰트를 생성하여 교육 등의 장면에 활용할 수 있으며, 나아가 동일한 한자에 대한 지역별 획순 차이를 반영할 수도 있다. 예를 들어 “王”자의 경우 중국 대륙에서는 두 번째 획이 가로획이지만 일본에서는 세로획이다. 원칙적으로 글리프위키도 이 기능을 충분히 구현할 수 있었으나, 초기부터 이 규칙을 따르지 않았기 때문에 현재의 규모로는 변경하기 어려운 상황이다.

이상의 비교는 어느 한쪽을 편들거나 폄하하려는 의도가 전혀 아니다. 본인은 오히려 글리프위키의 고도 이용자로서 10만 개가 넘는 글리프를 편집한 바 있다. 글리프위키와 메타한은 각각 장단점을 지니고 있다는 것은 부인할 수 없는 객관적 사실이다. 본인은 메타한이 정식으로 출시된 후 디자이너들이 양쪽을 유연하게 활용하여 서로의 장점을 취하고 단점을 보완하며 각자의 강점을 발휘하기를 기대한다.

현재 메타한은 아직 시각적 편집기가 제작되지 않아, 사진에서 보듯 골격 데이터를 수동으로 수정하여 조정해야 한다:

Generate!

```
200,10,149,22,96
300,53,209,10,-26
583,52,149,10,8,40
405,54,123,9,85,79
308,158,68,10,-24
202,106,-10,158
623:1,106,-2,10,122,206,150
202,106,29,158
202,106,68,158
305,177,192,10,100
202,177,192,230
610,230,192,10,214,-24,186
202,177,153,230
202,177,114,230
```

Generate!

```
200,10,149,26,96
300,53,209,12,-26
583,52,149,12,8,40
405,54,123,11,85,79
308,158,68,12,-24
202,106,-10,158
623:1,106,-2,12,122,206,150
202,106,29,158
202,106,68,158
305,177,192,12,100
202,177,192,230
610,230,192,12,214,-24,186
202,177,153,230
202,177,114,230
```

```
200,10,149,31,96
300,53,209,16,-26
583,52,149,16,8,40
405,54,123,15,85,79
308,158,68,16,-24
202,106,-10,158
623:1,106,-2,16,122,206,150
202,106,29,158
202,106,68,158
305,177,192,16,100
202,177,192,230
610,230,192,16,214,-24,186
202,177,153,230
202,177,114,230
```

Generate!

시각적 편집기가 제작 완료되면 글리프위키와 거의 동일한 편집 체험을 제공할 수 있게 된다.

여기서 "棡"자를 예로 드는 데는 또 다른 중요한 목적이 있는데, 바로 획 회전 로직을 설명하기 위함이다. 그림에서 볼 수 있듯 이 회전된 갈고리 획의 번호는 623이며, 623:1 형식으로 획의 방향을 표시한다. 여기서 :1은 해당 행 데이터가 가리키는 단일 획에만 영향을 미칠 뿐, 다른 획의 존재 여부나 획 간 순서와는 전혀 무관하다. 이는 글리프위키로는 구현할 수 없는 방식이다. 글리프위키에서 "회전" 조작 자체가 한 행의 데이터를 차지하는데, 이 행에는 가시적인 획이 포함되지 않으면서 그 위 행들 중 지정된 범위에 속하는 획을 모두 회전시킨다. 이 방식은 명백히 획 순서와 강하게 연동되어 있다. 우리가 데이터 순서를 글자의 획순 표시에 사용하기로 결정했으므로, 데이터 순서와 무관한 방식으로 획을 회전시켜야 한다. 물론 이에도 단점은 있는데, 회전되는 획마다 :1을 붙여야 하며 범위를 지정해 일괄 회전할 수는 없다.

한편 미러링 조작에 대해서는 메타한이 지원하지 않는다. 이는 메타한이 획 라이브러리에 디자인된 외곽선을 신축·보간하여 이용하는 데 비해, 단순 미러링 처리로는 외곽선의 방향이 바뀌어 버리기 때문이다. WYSIWYG 버전 폰트라면 외곽선 점의 순서를 반전시켜 방향을 되돌릴 수 있어 간단하지만, 힌팅은 점 순서 변경을 지원하지 않아 구현할 수 없다. 한자에 등장하는 미러링 형태의 획들은 모두 특수 획으로 분류하여 추가로 보강하며, 번호는 모두 8로 시작하도록 통일한다.

메타한명조에 대하여

앞서 서술한 바와 같이 이는 메타한의 첫 번째 완성품이다. 시각적 편집기가 아직 개발되지 않았으므로, 한글 폰트 메타한명조는 골격 데이터를 수동으로 편집·조정하여 디자인되었다.

```
222,40,179,50,94,179,8
215,17,189,40,179,83,111,105,14
210,92,185,135,153,8
218,153,185,110,139,13,8
215,89,149,110,139,83,190,103,13
200,168,212,203,199,116,66,14
223,89,58,54,19,14,8
210,132,55,181,203,8
214,203,55,9,-28,8,14
208,128,18,156,203,27,12
```

Generate!

이 획 라이브러리 또한 한글 글꼴 디자인을 위해 특별히 제작된 것으로, 20여 개의 기본 획만 포함하고 있다. 글리프 골격은 조선일보체를 참고하여 디자인되었으며, 겉보기에는 스타일이 유사해 보이지만 실제로는 많은 차이점이 존재한다. 그리고 가장 중요한 점은 옛한글을 훌륭하게 지원한다는 것이다. 일반적으로 유통되는 대다수의 옛한글 폰트는 구성 부품을 조합할 때 중성의 위치 구조와 종성 유무에 따라 단순 분류만 할 뿐, 너비와 높이에 따른 세밀한 미세 조정을 진행하지 않는다. 이렇게 조합된 음절은 판독은 가능하지만 시각적 조화가 부족하다. 예를 들어 "긴", "길", "긿" 세 글자의 종성 높이는 분명히 다르므로, 미관을 고려하면 위쪽의 "기" 부분을 각기 다른 비율로 압축해야 한다. 메타한명조는 디자인 시 이 점을 고려하여 매우 세밀한 분류를 진행했는데, 예를 들어 초성 자모는 모두 66가지로 분류된다. Windows 시스템 기본 폰트인 맑은 고딕의 옛한글 부분에서는 초성 자모가 5가지로만 분류되어 있어, 규모 면에서 현격한 차이가 난다.

본인이 이전에 FontCreator로 디자인했던 일부 한글 글꼴과 비교할 때, 메타한명조는 작업량이 약 7배에 달하지만 총 소요 시간은 약 2.5배에 불과하다. 이는 메타한 프로젝트가 가져온 이점을 직접적으로 보여준다. 글꼴은 아래 링크에서 다운로드할 수 있으며, 본 문서 완료 시점의 버전은 1.00이다. 향후 업데이트 시 링크는 변경되지 않는다.

http://ko.cheonhyeong.com/fonts/mthmyung.ttf

향후 서로 다른 스타일의 한글 획 라이브러리를 두 세트 더 디자인하여, 메타한명조를 세 가지 스타일을 임의의 비율로 혼합할 수 있는 가변 폰트로 발전시킬 예정이다.

메타한명조가 공개된 후 "메타한"의 "한"이 "한자"가 아니라 "한글"의 "한" 아니냐는 농담 섞인 말이 나오기도 했다. 이는 분명히 메타한의 포지셔닝에 대한 오해이다. 물론 이해 못 할 바는 아닌 것이, 한자 폰트의 작업량은 한글 폰트와 비교할 수 없을 정도로 방대하여, 완전한 시스템이 공개되기 전에 개인의 힘만으로 단기간에 한자 폰트 하나를 만드는 것은 불가능하다. 이는 향후 시스템이 공개된 후 대중의 힘을 모아 함께 채워나가야 할 부분이다.

(문서 끝)