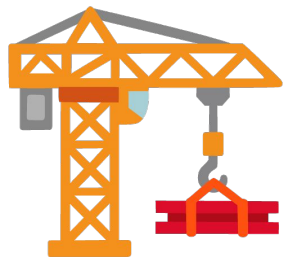




Better-Engineered Font-Formats

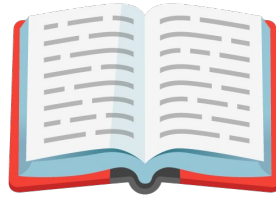
yet another update



GUADEC 2024—Denver
behdad.org



What's in a name?



engineered

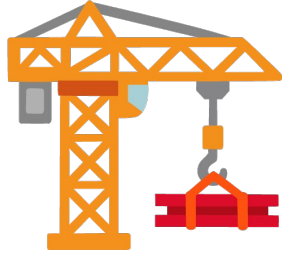
adjective

1. (of an organism) modified by manipulation of genetic material.
"genetically engineered plants"
2. skillfully and deliberately arranged rather than arising naturally or spontaneously.
"they plan to spy on Hamlet at a future, engineered meeting with Ophelia"

Part 1: Boring Expansion



Part 2: Better Ergonomics



Part 3: Beyond Emulation

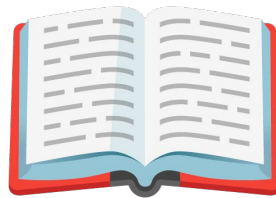




Part 1 of 3: **Boring Expansion**

Business as usual

Boring Expansion: What's in a name?



bor·ing

adjective

not interesting; tedious.

bore

verb

make (a hole) in something, especially with a revolving tool.



Boring Expansion: What is it?

Extend the font format:

- To address decades-long limitations.
- Add highly-desirable features.
- It's okay if adoption takes time.



Boring Expansion: So far...

- avar2
<https://github.com/harfbuzz/boring-expansion-spec/blob/main/avar2.md>
- Cubic **glyf** outlines
<https://github.com/harfbuzz/boring-expansion-spec/blob/main/glyf1-cubicOutlines.md>
- VarComposites
<https://github.com/harfbuzz/boring-expansion-spec/blob/main/VARC.md>
- beyond-64k
<https://github.com/harfbuzz/boring-expansion-spec/blob/main/beyond-64k.md>



avar2: What is it?

- New version of the **avar** table that supports arbitrary mapping across variation axes.
- Specification:
github.com/harfbuzz/boring-expansion-spec/blob/main/avar2.md
- Support: FontTools, HarfBuzz, FreeType, macOS/iOS

avar2: Use cases

- Designspace warping
- Higher-order interpolation (HOI, aka Non Linear Interpolation)
- Parametric fonts
- Designspace fences





avar2: Parametric fonts

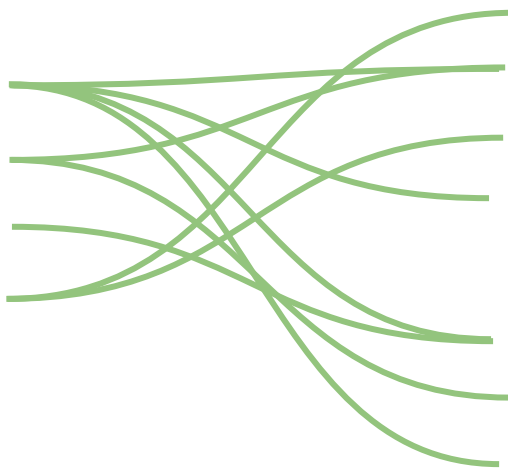
Vastly fewer corner masters needed in parametrically-designed fonts.

Input:

Weight
Width
Optical Size
Grade

Output:

XOPQ
XTRA
YOPQ
YTLC
YTUC
YTAS
YTDE
YTFI



avar2: Results

- Prototypes: Parametric fonts:
 - 80% file-size savings in Roboto Flex
 - 70% file-size savings in Amstelvar





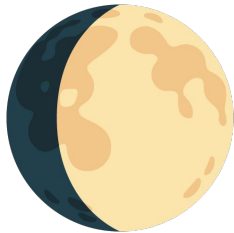
Cubic **glyf** outlines: What is it?

- Extension to the **glyf** table to support cubic Bezier curves.
- Specification:
github.com/harfbuzz/boring-expansion-spec/blob/main/glyf1-cubicOutlines.md
- Experimental support: FontTools, HarfBuzz
Patch: FreeType



Cubic `glyf` outlines: Use cases

- Higher-fidelity representation of outlines, especially important for:
 - lower units-per-EM fonts,
 - very light weights,
 - variable fonts with many interacting masters, as conversion errors accumulate.
- Slightly smaller fonts.



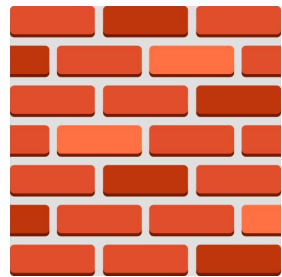
Cubic **glyf** outlines: Results

- Roughly 3% to 10% savings by just rebuilding fonts from existing sources, depending on how curve-heavy the design is.



VarComposites: What is it?

- New, **VARC** table to support *variable components*, aka “smart components”.
- Specification:
github.com/harfbuzz/boring-expansion-spec/blob/main/VARC.md
- Experimental support: FontTools, HarfBuzz



Classic components

- References another glyph *at the font's current variations settings*
- Translation offset: *variable*
- 2x2 transformation matrix for scale, rotate and shear: *not variable!*

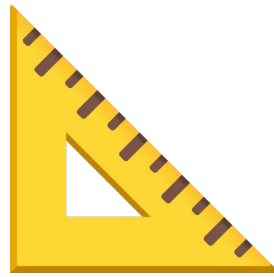


Variable components

- References another glyph *an arbitrary variable location* in the variation space
- Translation offset: *variable*
- Rotation: *variable*
- Scale: *variable*
- Shear: *variable*
- Center of transformation: *variable*

VarComposites: Use cases

- Han fonts
- Hangul fonts
- Tibetan fonts
- Latin stencil / modular fonts
- Icon fonts
- ...





VarComposites: How it works

- Each VarComposite glyph refers to components at certain location in the component glyphs' designspace.
- Same component can be used to provide a variety of shapes.

2 2 2 2 2 2 2 2

2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

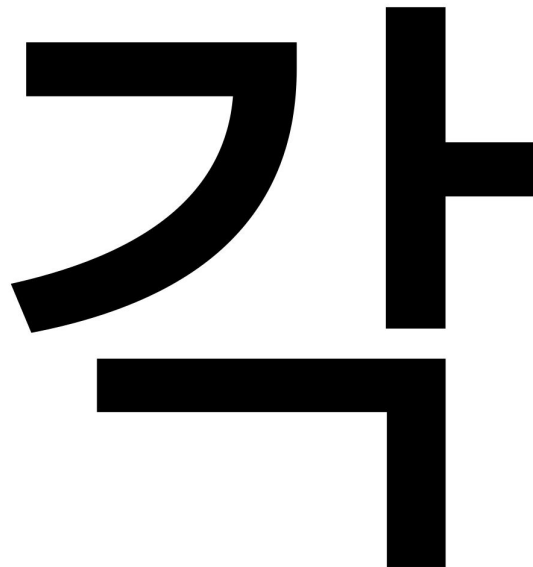
VarComposites: Results

- Han variable-font with 2 weights & ~44k ideographs:

67% file-size savings!

- Hangul variable-font with 2 weights & ~11k syllables:

92% file-size savings!!





beyond-64k: What is it?

- Extension to the font format to support more than the current glyph limit: 65,535!
- Specification:
github.com/harfbuzz/boring-expansion-spec/blob/main/beyond-64k.md
- Experimental support: HarfBuzz
Proof-of-concept: FontTools, FreeType



beyond-64k: Use cases

- Massive CJK fonts

Currently Noto CJK / Source Han CJK fonts at maximum glyph count, and missing many CJK characters.

- Pan-Unicode fonts

Currently pan-Unicode fonts use 100+ separate files that are hard to organize / order, with distinct family names. This feature allows eg. having one **NotoSans.ttf** as One Font to Rule Them All.

Simplifies document fallback where file-size is not constrained, or incremental transfer is available

beyond-64k: How it works

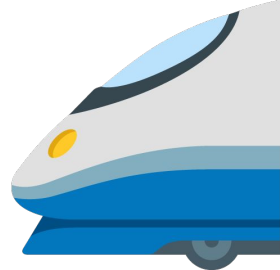


- 24-bit glyph indices instead of 16-bit, allowing 16m glyphs. *Ought* to be enough for everyone, right?

For context, Unicode allows up to ~1m codepoints, of which currently ~150k are used.

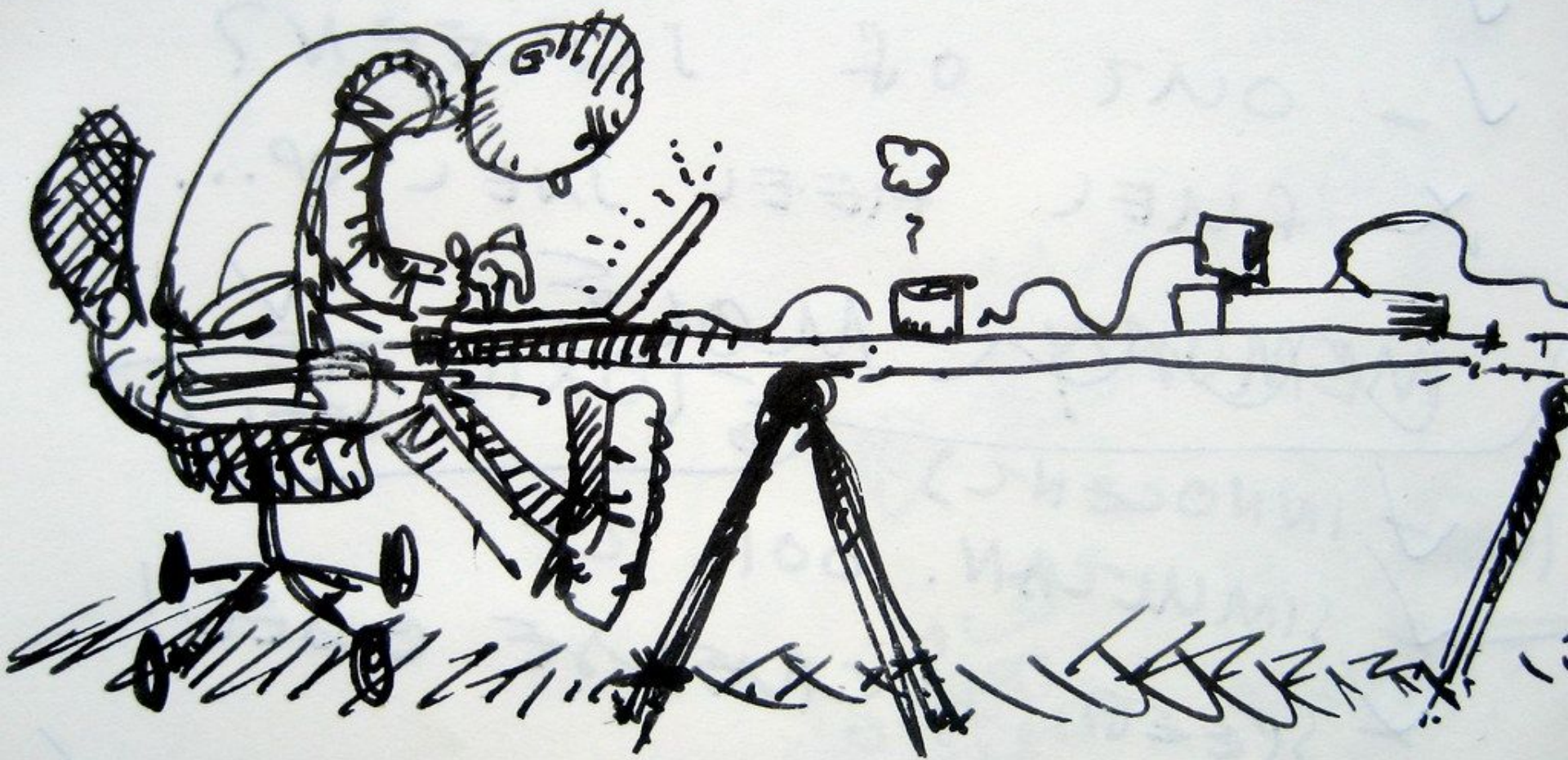
- Noto fonts use less than 200k glyphs total (due to many scripts' conjuncts)
- Many tables updated to allow more than 64k glyphs.
- Most intensive changes applied to the **GDEF** / **GSUB** / **GPOS** tables, also alleviating the long-standing *offset-overflow* issues. Designed for code-reuse via templates.

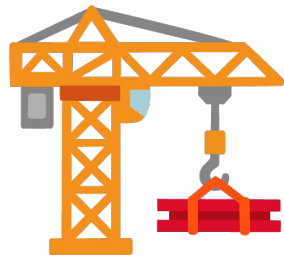
beyond-64k: Results



- 15MB merged **NotoSans.ttf**, containing all scripts except for CJK.
- Over 100,000 glyphs.
- **GPOS** table over 1.5MB, compiled successfully without overflows.

- Drawback: HarfBuzz binary size growth +100kb (+%10)

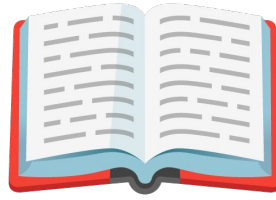




Part 2 of 3: Better Ergonomics

The one where we rebuild everything

Better Ergonomics: What's in a name?



er·go·nom·ics

noun

the study of people's efficiency in their working environment.



Better Ergonomics: A tangent

- Google Fonts effort to rewrite everything in *Rust* (from Python/C/C++):
 - No more security vulnerabilities, yay!
 - Super-fast font compiler.

github.com/googlefonts/oxidize

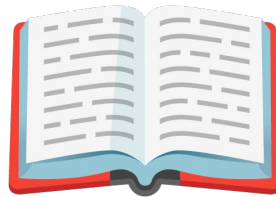
- FreeType replacement (Skrifa)
- HarfBuzz replacement (TBD: [HarfRuzz](#), forked from [RustyBuzz](#))
- fontmake replacement ([fontc](#))
- All built on top of [Fontations](#).





Part 3 of 3: Beyond Emulation

Putting the *smart* into fonts



Beyond Emulation: What's in a name?

em·u·la·tion

noun

effort to match or surpass a person or achievement, typically by imitation.
"their success is inspiring emulation from others"

COMPUTING

reproduction of the function or action of a different computer, software system, etc.

"software emulation of complete systems"

Beyond Emulation: Problem statement



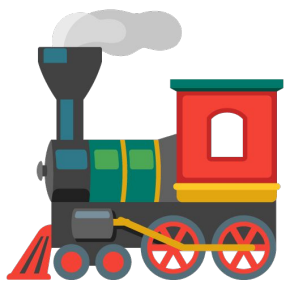
OpenType
shaping





Beyond Emulation: Problem statement

- Complex scripts bump up against the limits of OpenType Layout
- When you need to calculate things
 - “Flow fonts”, connected shorthand, etc.
- When you don’t know how many things you have until run-time
 - `sub @flagged @any @any @any @any @any @any @any @any @above;`
`sub @flagged @any @any @any @any @any @any @any @above;`
...
- When you don’t know what things you have until run-time
- When you don’t know where things are until run-time
 - Egyptian hieroglyphs
 - Arabic dot collision avoidance
 - Automated kerning



Beyond Emulation: What is it?

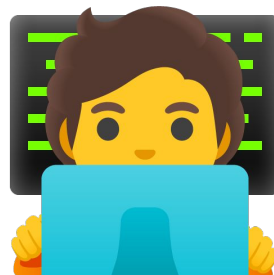
- WebAssembly for President!
- Optionally embed font shaping & drawing logic in the font's **Wasm** table.
- Interpreted by the font engine in a *secure* environment.
- Puts the font engineer in the driver's seat.
- <https://github.com/harfbuzz/harfbuzz/pull/4131>
<https://github.com/harfbuzz/harfbuzz/blob/wasm/src/hb-wasm-api.h>



Beyond Emulation: Use cases

- Graphite, DecoType ACE, other alternate shaping engines
- Custom programmatic shaping (collision avoidance, etc)
- Custom effects

Beyond Emulation: How it works



```
1 #define HB_WASM_INTERFACE(ret_t, name) __attribute__((export_name(#name))) ret_t name
2
3 #include <hb-wasm-api.h>
4
5 bool_t
6 shape (void *shape_plan,
7        font_t *font, buffer_t *buffer,
8        const feature_t *features, uint32_t num_features)
9 {
10     if (!shape_with (font, buffer, features, num_features, "ot"))
11         return false;
12
13     // Modify buffer;
14
15     return true;
16 }
```



Beyond Emulation: [Demos](#)

- SIL Graphite2
- Arabic collision-avoidance / kerning
- Arabic baseline-balancing
- Egyptian Hieroglyphs
- Dotted handwriting font
- Embedded rasterization

And:

- Bad Apple
- llama.ttf
- translate.ttf

Beyond Emulation: SIL Graphite2

- 90kb wasm payload (40kb in WOFF2)
- 3x slower than native, with JIT



Beyond Emulation: Arabic collision-avoidance



نچھے جرپیل

OpenType
(No mitigations)

نچھے جرپیل

OpenType
(Best effort)

نچھے جرپیل

WASM

Beyond Emulation: Arabic baseline-balancing



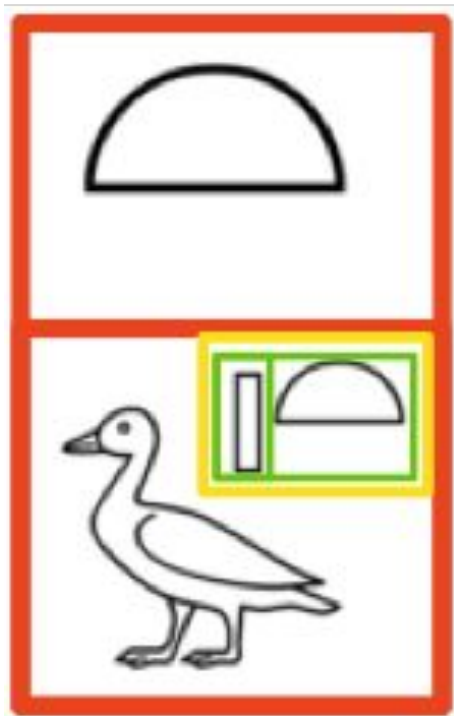
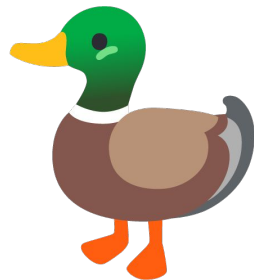
OT
(Best effort)

أَفْسُوا السَّلامَ بَيْنَكُمْ

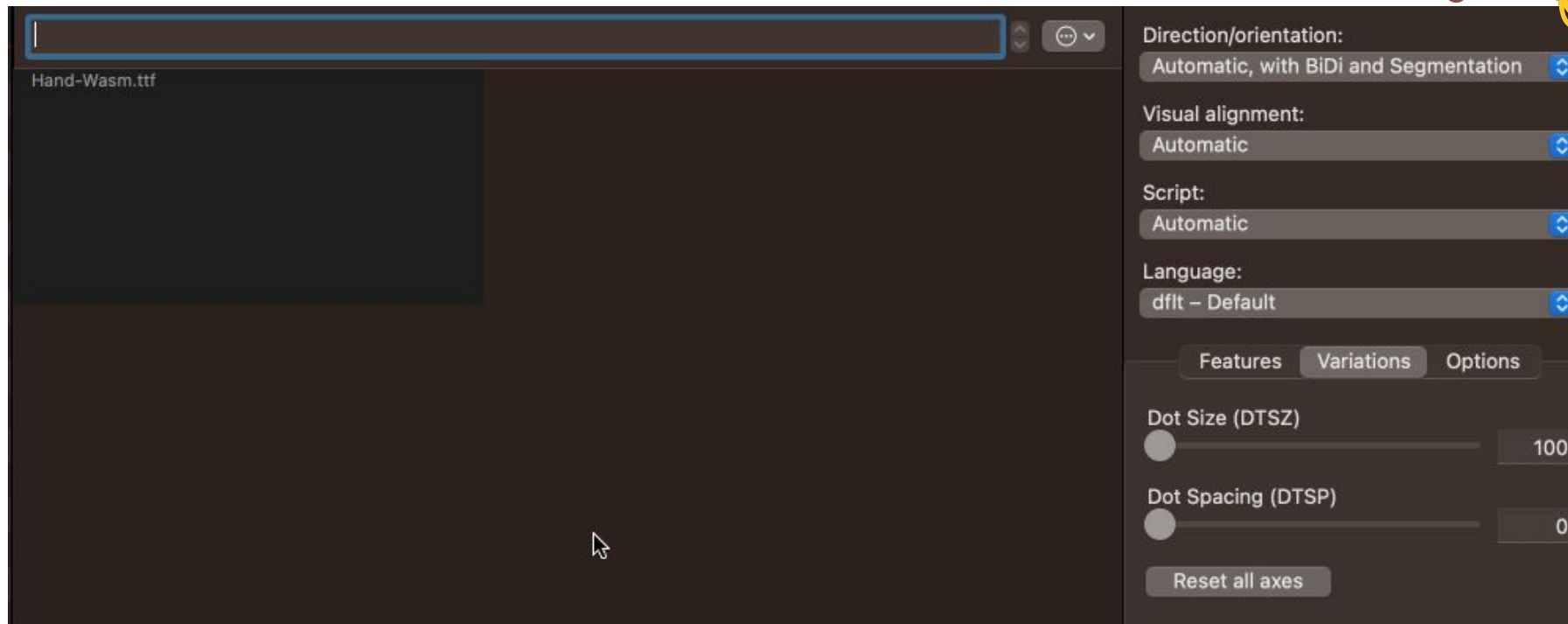
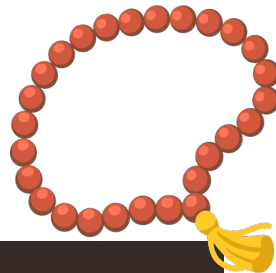
WASM

أَفْسُوا السَّلامَ بَيْنَكُمْ

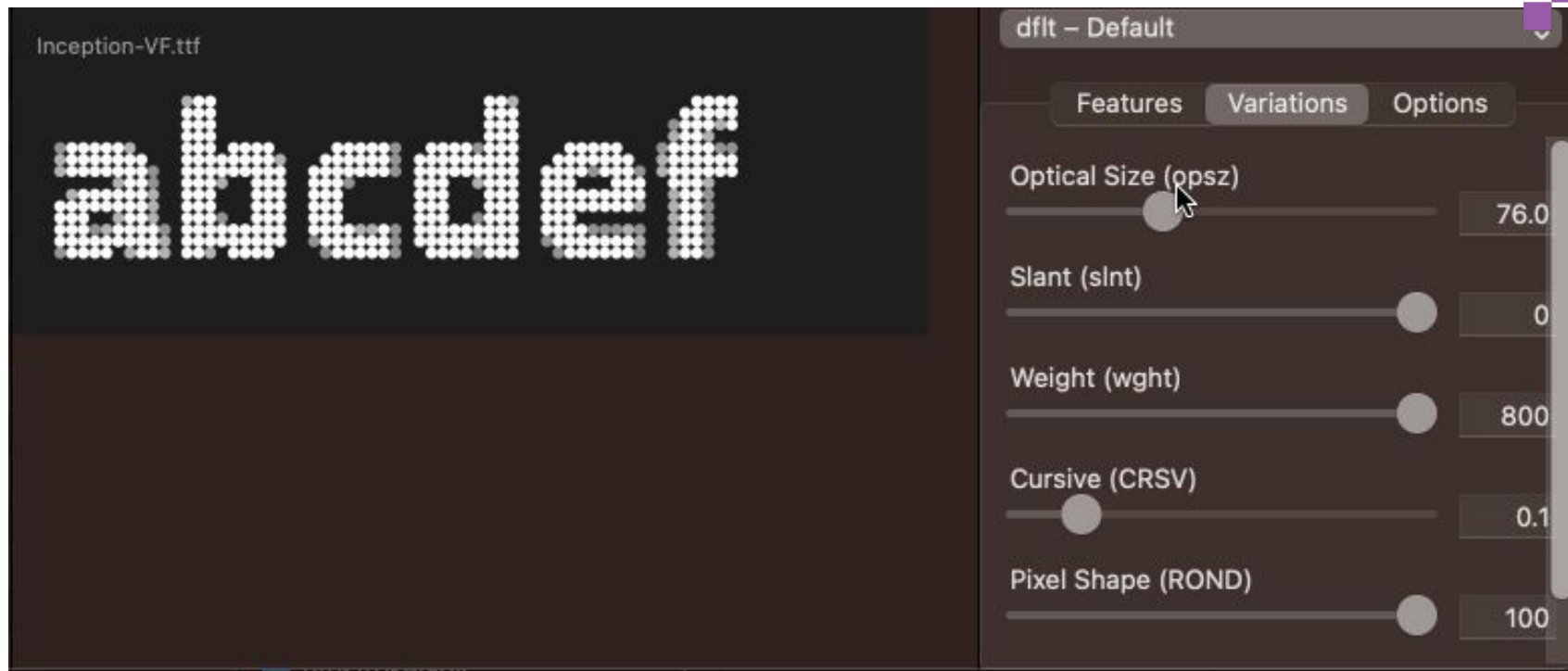
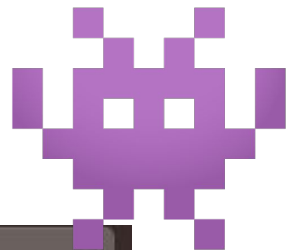
Beyond Emulation: Egyptian Hieroglyphs



Beyond Emulation: Dotted Handwriting Font



Beyond Emulation: Embedded rasterization



Beyond Emulation: BIG question, again:



Why?



Beyond Emulation: Take Arabic typesetting

- Graphite
 - DecoType
 - DigitalKhaatt
 - Other custom-built solutions, some only available in Iran / Pakistan
-
- We don't *want* to standardize one solution. It would be way too specific for a general font format.

Beyond Emulation: Interpretations

- Pluggable shapers
- For typesetting
- Universal extension mechanism
- Finally putting the *smart* in fonts





Beyond Emulation: Precedent

- Graphics: OpenGL Shading Language, WebGPU Shading Language
- The Web: Javascript, WebAssembly
- Web serving: Containers, Wasm in Edge Computing
- Typesetting: METAFONT, TeX, luaTeX



Beyond Emulation: A philosophical debate

- AAT / Graphite chose a state-machine approach to shaping.
- OpenType chose a per-script shaping model.
- The OpenType approach proved hard to scale, so the [Universal Shaping Engine](#) was born.
- It is impossible to envision all needs *and* encode them in the font format.
- The choices are to either give up, or enable arbitrary computation.
- As demonstrated already, OpenType lookups are Turing-complete; just tedious to work with, as is the Turing machine itself.
- One Nastaliq font has a **GPOS** Lookup with more than 3,800 sublookups!
 - The equivalent WASM font is $\frac{1}{3}$ of the size.

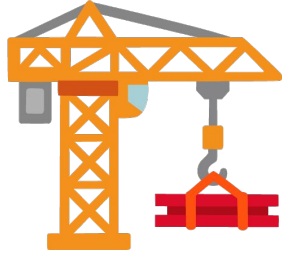
Goofy

- [Bad Apple](#)
- [llama.ttf](#)
- [translate.ttf](#)

Part 1: Boring Expansion



Part 2: Better Ergonomics



Part 3: Beyond Emulation



Conferences



MPEG & other standards

github.com/harfbuzz/boring-expansion-spec



On HarfBuzz...

Proliferation in market share:

- Linux Desktop, Android, ChromeOS
- Chrome, Firefox, GNOME Web, Microsoft Edge
- LibreOffice, XeTeX, LuaTeX
- Adobe Photoshop, Illustrator, InDesign
- PlayStation, game engines, ...
- Smart TVs, cars, ...

Innovating but slower in roll-out, unlike Apple / Microsoft.



Thanks

[Google Fonts](#), for:

- Supporting my work
- Investing in the Open Source fonts ecosystem

In another note, I wrote my thoughts on the ecosystem recently at:

State of Text Rendering 2024 (behdad.org/text2024)



And now...