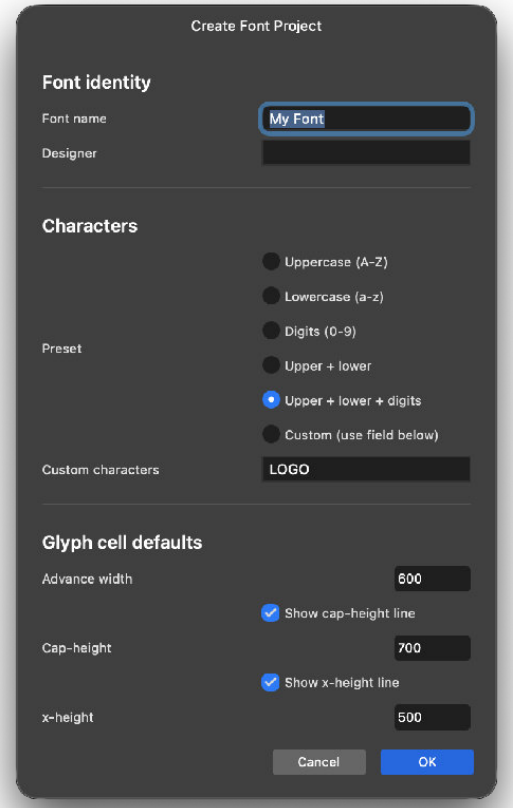
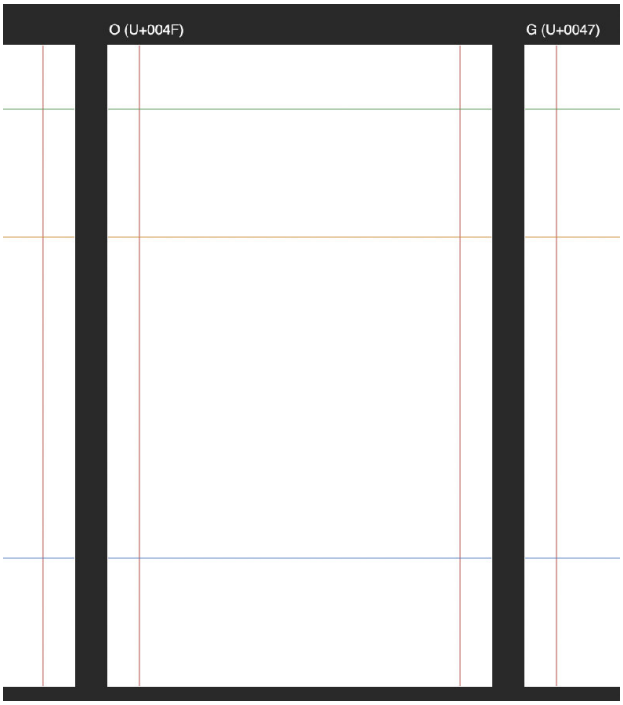
# affinity-font-setup

Creates a new Affinity Designer document set up as a font workspace: one artboard per glyph, laid out in rows of 10, each with locked metric guides. Run this first, draw your glyph artwork into the artboards, then run `affinity-font-export.js` to build the `.otf` file.






# Installation

1. Download the zip file and unzip it.
2. Run the script in one of two ways: **a) Run directly** — Open the Script Editor panel ( **Window > Scripting > Script Editor**), paste the contents of `affinity-font-setup.js` into the editor, and press **Run**. **b) Save to Script Library first** — Follow step (a) above, then click **Save As...** to save the script to the Script Library. Open the Script Library panel (**Window > Scripting > Script Library**) and click the saved script once to run it.

# Purpose

Turns a font idea into a drawing workspace in one click. You choose which characters the font will contain and the basic metrics, and the script builds a 1000 dpi document where every glyph gets its own artboard — one em-square tall — with colour-coded metric guides already in place and locked:

- **Red verticals** — left and right sidebearings (LSB, RSB). The distance between them becomes the glyph's advance width on export.
- **Blue horizontal** — baseline. Glyph artwork sits on this line.
- **Green horizontal** — cap-height (optional).
- **Orange horizontal** — x-height (optional).

Artboards read top-to-bottom as Uppercase  Lowercase  Digits  Other, both on the spread and in the Layers panel. Each artboard is named `Name (U+XXXX)`, e.g. `G (U+0047)` — the export script parses these names, so do not rename them.

# Workflow (run order)

1. **Run `affinity-font-setup.js`** — pick a preset, accept or adjust the metrics, press OK. A new document opens with empty artboards.
2. **Draw each glyph** inside its artboard with the Pen/Node tools, using the guides for alignment. Anything you draw becomes part of the glyph; the guides themselves are ignored on export. Leave an artboard empty and it is skipped with a note, not an error.
3. **Run `affinity-font-export.js`** with that document open — it reads the artboards and writes the `.otf` file to your Desktop.

# Dialog

- **Font name** — working title of the project (default `My Font`).
- **Designer** — optional, kept for future use.
- **Preset** — which characters to create artboards for: `Uppercase (A-Z)`, `Lowercase (a-z)`, `Digits (0-9)`, `Upper + lower`, `Upper + lower + digits` (default), or `Custom (use field below)`.
- **Custom characters** — only used with the Custom preset. Every unique non-space character in the field gets an artboard (duplicates and whitespace are dropped).
- **Advance width** — default advance for every cell, 100–2000 (default 600). Per-glyph widths come from the RSB marker position at export time — drag the red RSB line to change a single glyph's width.
- **Show cap-height line / Cap-height** — green guide, 100–1000 (default 700).
- **Show x-height line / x-height** — orange guide, 100–1000 (default 500).

Cancelling the dialog creates nothing. Choosing a preset with no characters (e.g. an empty Custom field) logs a note and stops.

# Drawing tips

- Draw with closed outlines. Open strokes export as-is and may not fill the way they look on screen.
- Keep artwork inside the LSB/RSB lines unless you mean to kern by hand — anything outside still exports, shifting the visual spacing.
- To change one glyph's width, unlock and drag its red RSB line; advance width is measured from the markers, not the artboard edge.
- To change the metrics globally (taller caps, say), move the green guide in every artboard — or re-run the script with new values and copy your artwork across.
- The guides are locked so they can't be moved by accident. Unlock a guide only to reposition it, then lock it again.

# Requirements

- Affinity by Canva 3.3 (per script header).
- No document needs to be open — the script creates a new one.

# What this is (and is not) for

This is a **hobby/small-project font maker, not a professional type design tool**. There is no kerning, no hinting, no multiple weights, no OpenType features (ligatures, alternates, small caps), and the vertical metrics are fixed (ascender 800, descender −200). What it is genuinely good for: **dingbat, symbol, icon, and logo fonts** — a handful of shapes you want to type instead of place, like a logo sheet, UI icons, or decorative ornaments. If you need a text face for body copy, use a real font editor.

# Known limitations

- Artboards may come up with mixed expanded/collapsed state in the Layers panel (which ones varies run to run — a panel refresh artifact, not script data). One click on Collapse All tidies it; there is no scriptable disclosure state in the SDK.
- Renaming an artboard so it no longer contains `(U+XXXX)` makes the export script skip it ("could not parse name"). Deleting a marker line makes it skip the glyph ("missing required marker").
- The RSB marker is clamped inside the artboard; advance widths wider than the cell need a wider Advance width up front.
- All creation runs as a short command sequence per artboard, so reverting a full setup takes several undo steps.

# Changelog

1.1.0 - Failures now show an alert dialog instead of only logging to the console; settings moved into a `CONFIG` object

1.0.1 - Removed no-op `isLandscape` assignment; clear end-of-run selection so artboards come up uniformly collapsed

1.0.0 - Initial version