A brief introduction to drawing type in RoboFont

1 Formats and tools

1.1 Fonts and work files. The working format we will be using for the source files of our fonts is UFO. The *Unified Font Object* (.ufo) is “a cross-platform, cross-application, human readable, future proof format for storing font data” (from the specification). From these source files, we will generate installable *OpenType* fonts (.otf). The OpenType format is the current de facto standard for font files and is widely supported across platforms. Specifically, we will be dealing with *PostScript flavored fonts,* which mainly means that they use *quadratic beziers:* the kinds of curves you know from other vector-editing applications, which are much nicer to handle than the alternative (which is *TrueType*).

Tip: Always zip (compress) UFOs before emailing them. They technically consist of lots of small text files, which makes them a bit unwieldy in some contexts, like email or sharing over DropBox.

1.2 Toolspace and workflow. UFO is an open format, and a UFO workflow does not limit you to any one program; one file can be passed between software built for different tasks. We may encounter a couple of other, specialized tools later in the process; for now we will focus on *RoboFont,* the font editor in which we will create the drawings and their spacing.

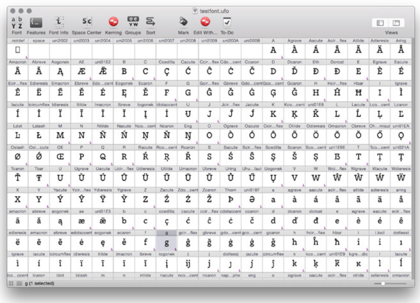
2 RoboFont: The workspace

2.1 Concept and overview. RoboFont’s interface is fairly modular, using individual windows for individual tasks. The environment may also strike you as rather minimal. The idea is for it to be an open platform that can be customized, rather than locking you into one prefabricated workflow from the start.
If you are looking for specific functionality, generally these are the places to look:

- Menus
- Context menus in relevant windows
- Buttons/flyout menus on relevant windows
- Hot keys (see preferences)
- Or it might actually not be built in, in which case there might be an extension for it (see point 2.7).

Tip: If you don’t enjoy handling multiple windows, you may want to try switching to Single Window Mode (Preferences > Misc).

2.2 The Font Collection. This is the first window you’ll see when you open a font or start a new one. It displays an overview of the glyphs (i.e. all the individual character representations/drawings) contained in the font, shown as a grid of cells or alternately as a list (you can switch between these views in the bottom-left corner of the window). This is where you manage your glyphs, sort them, create new ones or remove them.



Glyph display. Glyphs that are contained in the font are displayed as white cells with a preview of the drawing. Glyphs that are provided in the template but haven't been added to the font yet are shown in the Font Collection as *template glyphs:* grey cells that display a preview of what the glyph looks like in an existing font.
Glyphs that have been changed since the last save are highlighted.
Depending on the size of the glyph cells (slider in the bottom-right corner of the window) they will display additional information:

- A top bar with the glyph name (hidden at small sizes),
- Abbreviations that indicate whether a glyph contains additional layers (**L**) or notes (**N**),
- At large sizes, color-coded overlay of any additional layers.

Handling glyphs

- Double-clicking a glyph opens it in a glyph window.
- To add a glyph, if it is already present in the template you can just double-click the template cell and a blank glyph will be created for it. Otherwise hit ⌘⌥**G** (or select **Font > Add Glyphs…** from the menu) and enter the glyph name.
- Deleting a glyph (with the backspace key) removes it from the font, but still shows a template glyph in its place. To remove a glyph including its template glyph, hold ⌥ while deleting.
- Copy and paste glyphs or template glyphs (⌘**C**, ⌘**V**). If no other glyphs are selected, pasting will append the copied glyphs to the end of the font. If a number of glyphs are selected that matches the number of copied glyphs, pasting will replace those.

Tip: Usually only one glyph window per font is opened. If you would like to open a second one, hold ⌥ as you double-click the cell.

Finding glyphs

- When looking for a specific glyph, start typing its name to select it.
- ⌘**F** opens an interface for advanced searching and filtering. See the documentation for an explanation of the options: http://doc.robofont.com/documentation/workspace/font-collection/search-and-find/

Sorting glyphs

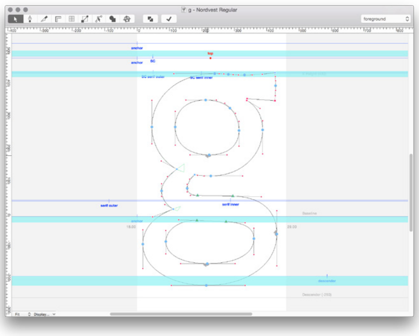
- Having a sort order that makes sense to you will make it easier to find things; sort order is also preserved in the generated files, and can be made to show up e.g. in InDesign's glyph palette.
- Use the **Sort** button in the top bar of the window to sort glyphs according to a given character set (see below) or a custom sort order.
- You can also select glyphs (or template glyphs) and drag them to change the order manually.

Tip: When dragging and dropping glyphs be careful to drop them *between* (rather than on top of) other glyphs — otherwise you'll replace the drawings in the glyph slots you are dropping them on.

The Font Info window

- The Font Info window is accessible from the top bar of the window. This is where you set up your font; more on these options below.

2.3 The Glyph Editor. This is the detail view of any one glyph, which is where any editing of the individual glyphs is done.



The buttons on the left side of the top bar show the available tools: the editing tool, the drawing tool, the slice tool, the measuring tool, and any custom tools you may have installed. How to use these tools will be discussed in section 6 below.

The dropdown menu on the right side of the top bar allows you to switch between the layers in the glyph. More on using layers in point 6.7.

The **View Options** flyout menu in the bottom-left of the window gives you fine-grained control over which aspects of the glyph you would like displayed.

If you don't like how any aspect of the glyph representation looks, chances are there's a setting to change it in the preferences. For a full documentation of the options available in the Preferences see: http://doc.robofont.com/documentation/workspace/preferences/glyph-view/

Tip: Of particular note for people using MacBooks with Retina screens is the setting "Draw strokes using antialias" in the Preferences, which makes the outlines and other 1-pixel lines a little less thin on high-resolution screens.

Tip: You can make working in the Glyph Editor faster and more powerful by assigning custom "hot keys". These are shortcuts used without modifier keys. Go to Preferences > Glyph Window > Hot Keys to set these up. In fact, many functions in the Glyph Window are only accessible this way.

Navigation. To navigate to another glyph while inside the glyph window, you can of course close the glyph window and double-click another glyph in the Font Collection; faster ways are:
- Using ⌘**J** to jump to another glyph (just start typing its name);
- Using hot keys to navigate to the previous or next glyphs.

2.4 The Inspector Palette. This "lifesaver" palette should likely remain open most of the time. It gives access to detailed information about the selected/current glyph, and a number of handy operations:

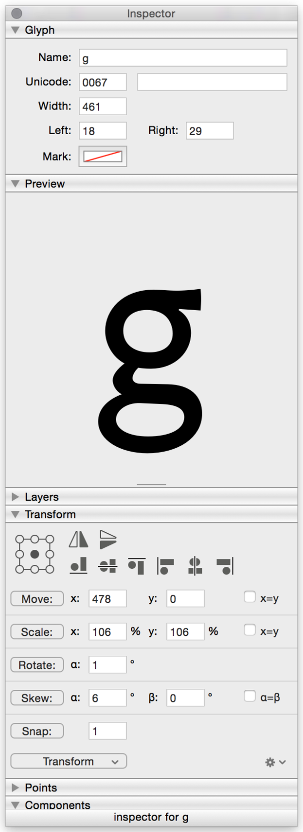
- *Names and Unicode information.* If you ever need to rename a glyph you would do that here, for instance after copying a glyph. (While the preset glyph names may not be immediately intuitive, you should generally leave them as they are: they automatically map to their Unicode representation, which makes these glyphs accessible and "typable" in other applications.)

- *Layers.* This is where you set up the number and sequence of layers, and set view options for them (for instance, whether a layer should be visible if it is in the background).

- *Align and transform.* This palette offers quick access to a number of transformations and alignment options that will largely be familiar to you from illustration software. Set the desired transformation options and then hit enter, or click the button in question.

- The palette also shows a preview of the current glyph, lists its various constituent elements, and lets you affix a note to it.

For a more exhaustive overview and explanation of available controls in the Inspector palette see the official documentation: http://doc.robofont.com/documentation/workspace/inspector/

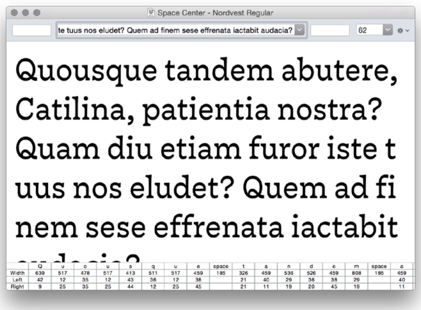


Name: g
Unicode: 0067
Width: 461
Left: 18 Right: 29
Mark:

Move: x: 478 y: 0 x=y
Scale: x: 106 % y: 106 % x=y
Rotate: α: 1 °
Skew: α: 6 ° β: 0 ° α=β
Snap: 1
Transform

inspector for g

2.5 The Space Center. This window allows you to type in your font. This is where you work out the spacing, and it is also generally useful for testing and previewing. Open the Space Center with the button from the Font Collection window, by selecting it from the Window menu, or by hitting ⌘⌥**S**.


Space Center - Nordvest Regular

te tuus nos eludet? Quem ad finem sese effrenata iactabit audacia? | 62

Quousque tandem abutere, Catilina, patientia nostra? Quam diu etiam furor iste tuus nos eludet? Quem ad finem sese effrenata iactabit audacia?

	Q	u	o	u	s	q	u	e	space	t	a	n	d	e	m	space	a
Width	639	517	478	517	413	511	517	459	185	326	459	536	526	459	808	185	459
Left	42	12	35	12	43	36	12	38		21	40	29	36	38	29		40
Right	9	25	35	25	44	12	25	45		21	11	19	20	45	19		11

Tip: Usually only one Space Center window is opened per font. If you would like to open a second one, hold ⌥ as you click the Space Center button.

Text entry. Use the large text field on the top of the window:
- Either just type continuous text, like so.
- Inputting a glyph name preceded by a slash will render the glyph with that name: **/adieresis** will display as **ä**.
- You can mix these two modes of representation. NB: When switching from a typed-out glyph name back to continuous text, you will need to insert a space: **Mot/odieresis rhead** will display as **Motörhead**.
- The **/?** wildcard shows the current glyph (the current glyph is the one selected in the Font Collection, or the one open in the Glyph Editor if that window has the main focus).
- Type **\n** to break to a new line.

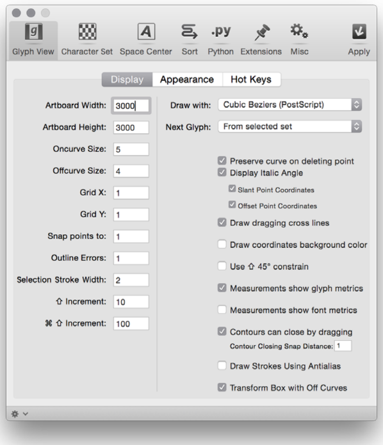
Note the little additional text entry fields to the left and right of the main text entry box. These are for optional *context glyphs* to be inserted before/after every glyph in the main text input. This is super useful for spacing, see section 7 below.

The Space Center has a range of *view options,* accessible from the flyout menu (gear icon) in the top-right corner of the window. You can display text as *Multiline* (continuous text that wraps to multiple lines), *Single Line,* or *Waterfall* (one line, rendered in a range of sizes). *x-Height Cut* shows only the x-height region, reversed (this can be helpful for spacing), while *Inverse* inverts the view, and *Upside Down* flips it. Take some time to familiarize yourself with the options; also note *View Metrics* and the *Beam,* useful for measuring.

NB: RoboFont is not a typesetting application, and rendering text can slow it down significantly. Don't put too much text in the Space Center, and close the Space Center window whenever you aren't using it.

Many of the view options described above can be assigned to custom hot keys. It's worth taking some time to review these options under **Preferences > Space Center > Hot Keys**.

2.6 The Preferences window contains many useful settings for various windows, as well as determining how RoboFont handles things internally: You'll find switches for sorting modes, mark colors, interface options etc. Press the **Apply** button after changing preferences for them to take effect; some settings require reopening the font or relaunching RoboFont, which will be indicated in the bottom of the window.
Sets of preferences can also be saved and loaded (with the little gear icon bottom left), which can be useful for instance for setting up your workspace on a new computer.



Artboard Width: 3000
Artboard Height: 3000
Oncurve Size: 5
Offcurve Size: 4
Grid X: 1
Grid Y: 1
Snap points to: 1
Outline Errors: 1
Selection Stroke Width: 2
⇧ Increment: 10
⌘ ⇧ Increment: 100

Draw with: Cubic Beziers (PostScript)
Next Glyph: From selected set

✓ Preserve curve on deleting point
✓ Display Italic Angle
✓ Slant Point Coordinates
✓ Offset Point Coordinates
✓ Draw dragging cross lines
☐ Draw coordinates background color
☐ Use ⇧ 45° constrain
✓ Measurements show glyph metrics
☐ Measurements show font metrics
✓ Contours can close by dragging
Contour Closing Snap Distance: 1
☐ Draw Strokes Using Antialias
✓ Transform Box with Off Curves

2.7 Scripts and extensions. The functionality of RoboFont can be customized and expanded. Many designers use a set of custom scripts and extensions (plugins) made by a variety of RoboFont users. A few potentially useful extensions are pointed out throughout this tutorial. Most of them can be downloaded for free, and they are installed simply by double-clicking a *.roboFontExtension* file.

Extension Tip: You can find an overview of available extensions on robofontextensions.com.
One especially useful "meta" extension is RoboFont Mechanic, an extension manager that lets you easily find and install other extensions, and alerts you when they are updated. www.robofontmechanic.com

3 Working from existing artwork

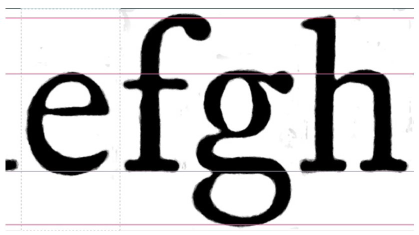
When working from reference material like manual drawings or scans, the common way is to move directly from scans/bitmaps to RoboFont. It may be tempting to autotrace in Illustrator, but the quick vectorizing usually isn't worth the lack of control and messy point structure this generates.

Preparing your artwork

- Images can be copy-pasted into glyph windows in RoboFont one-by-one, and moved and scaled to match; RoboFont has a pretty nifty interface for setting the scale of an image. However this only treats one image at a time, and having to scale all of them separately gets cumbersome and hard to control. I would recommend setting the image material to the correct scale factor beforehand and then copy-pasting at the correct size. This has the nice side effect that it lets you determine the vertical proportions and measurements before having to set them up in your font.
- The first thing is to make sure your source material is consistent and representative. When working from scans of printed type, you may want to look at more than one instance of each glyph, maybe overlay them to get a good idea of what the shape is that you're aiming to reproduce. You should also make the shapes as clear and sharp as possible, which means setting the levels (potentially with the Threshold function) to produce a crisp black and white image that will be easier and less ambiguous to follow once it's seen faintly in the background of your glyph drawings.
- Make sure the images are rotated for the baseline to be exactly horizontal, and share the same scale.

Determining vertical metrics

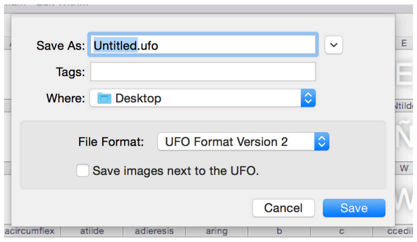
- I recommend collecting source scans of all letters/characters in a wide one-line overview, lining them up on a common baseline. Most likely, you'll find that the vertical proportions are fairly consistent: The x-line (top end of the small lowercase letters; look especially at flat ones like *x* and *z*), cap line (top end of caps, again mostly focusing on flat ones), as well as the length of ascenders & descenders are parameters you can mark across the font; see diagram on the right. Mark them with actual thin lines, not just guidelines, so you'll be able to refer back to these measurements outside of Photoshop too.



- Crop the height of the image to include the entire space from the deepest descenders to the tallest ascenders and caps, plus adding just a little bit of space on the top and bottom. This total height is going to be the *body height* of the font. For this concept, it helps to think of a sort of metal type: the *body size* of the type is the entire surface area of this little block of metal, and the *face* or visible image of the letters is typically a little bit smaller/shorter than the full height of the body. This is so that extenders don't touch when the type is set solid (without additional leading). *How much* padding to add on top and on the bottom is up to you. This is a big part of the reason why different fonts can vary wildly in apparent size when set at the same point size: When type is set at 12 points, it is the *body size* that is scaled to 12 points; and "how large the type is on the body" is up to the designer.

NB: In digital type design, it is not a big problem if some accents or other rarely-occuring things extend beyond the 1000-unit grid. Just be aware that if your font is set solid, its lines of 1000-unit-tall drawings will directly abut each other, and things that stick out of this body area can collide with the text above and below.

- Now resize your image proportionally to measure 1000 pixels vertically. The image height of 1000 pixels will translate directly into the 1000 units of vertical space that you have available as a “canvas” for drawing your glyphs in RoboFont. So you can now select, copy and paste the image of each glyph into the background layers of your glyphs in RoboFont and they’ll already be sized correctly; the only thing you’ll need to do is shift them to align with the baseline.
- Refer to the marked metrics lines in your scans to set up the vertical metrics in your digital designspace, as will be explained below.
- Once you have added images to your UFO file in RoboFont, re-save the file and make sure the option to “Save images next to the UFO” is activated. Otherwise RoboFont may lose the images from the file!


Save As: Untitled.ufo
Tags:
Where: Desktop
File Format: UFO Format Version 2
☐ Save images next to the UFO.

Make sure that bottom option is checked.

4 Starting a new font

4.1 Character set. The character set defines the scope and sequence of what glyphs you want to include in your font. This determines which template glyphs are going to show up, and in which order.

We have defined the basic character set on the right. This is the scope that we would eventually expect you to cover. You can always add more (and tips on expanding your character sets will be forthcoming later in the semester), but please try to resist the temptation of spending your time adding more glyphs rather than refining the central ones.

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z a b c d e f g h i j k l m n o p q r s t u v w x y z zero one two three four five six seven eight nine space period comma colon semicolon exclam question hyphen endash emdash parenleft parenright bracketleft bracketright slash asterisk ampersand dollar at quoteleft quoteright quotedblleft quotedblright

There are multiple ways to start out your font using this character set:
- See the template UFO file *Yale Basic Charset.ufo;* you’re welcome to copy this (empty) file as a starting point for your font.
- You can also go to **Preferences > Character set**. Below the list of character sets choose **Import > Import from UFO** and point that towards the abovementioned UFO. This will bring the character set into your preferences. Make sure to also select it on the very top of this preference pane as the default character set for new fonts. Now if you create a new font, it will use this character set.
- To make this latter process easier you can also import the RoboFont preferences file, *yale743a.roboFontSettings.* This is the default RoboFont configuration, except with this character set as default. (This will clear any preferences you have already set!) Be sure to apply the new preferences, then hit **⌘N** to create a new font. This should now contain an accurate overview of template glyphs.

4.2 Naming

Next open the Font Info window (accessible from the top bar of the Font Collection) to set up the basic settings of the design.

Choose a name. (Don’t fret if you can’t think of a good one right away; this is arguably one of the harder parts of type design. For now, a temporary working title will do just fine.) Enter the name in the **Family Name** field. Font names are used to identify the font in a variety of contexts and platforms, and while this is actually pretty complicated business behind the scenes, RoboFont allows you to skip most of that for now. One thing that is good to keep in mind is to limit names to a basic alphabetic range (no accents or special characters).

Other settings. Unless you're drawing a family of styles, you can leave the style name at its default **Regular**; the other entries in the **Identification** portion should also be fine at their default values for now.

4.3 Vertical proportions

Under the **Dimensions** heading, the **Units per em** value sets up the size of your design space. For these PostScript-based fonts, this value ought to be set to the default 1000. What this means is that the vertical space you have for each glyph drawing is divided up into 1000 units. The units are the same size in the horizontal direction, laying an orthogonal grid over your drawing space.

While the width and horizontal proportions of your drawings will typically vary from glyph to glyph, the structuring of the vertical space is typically fairly consistent across a font. As you have determined the vertical proportions in preparing your artwork, you can now just measure those pixel values from the baseline in the original art and set the values here accordingly. The **cap height** and **x-height** can and should be set to their actual expected values as distances from the baseline; the baseline sits at the zero position by definition. There is one funny thing in how RoboFont treats vertical dimensions, and that is that the **ascender** and **descender** values should sum to the full 1000. Set this up so they include the entire height above the baseline: the actual ascender height plus any space you'd like to add beyond that, and same for the descenders. We will in the next step set up guidelines to mark the actual/physical length of ascenders and descenders.

4.4 Global guides

Next you will need to set up some other measurements in the form of global guidelines. In any glyph, drag horizontal guides from the top ruler onto the workspace. Either drag them to their vertical position, or right-click and enter the value numerically. Also in the right-click context menu, set the guidelines to *global* (which means they should be visible throughout the font rather than just in this one glyph); and it's a good idea to assign them names too, so you'll remember which line is what. You'll need the following guidelines:

- Ascender
- Descender
- Overshoots at vertical metrics. While those measurements prescribe exact values for the vertical positions of features, optics demand a bit more fuzziness: round or pointy shapes need to be slightly taller than square/flat ones to appear the same size. Marking such alternate measurementswith global guides can help ensure consistency.

Further reading on overshoots and why they are needed, from Tobias: frerejones.com/blog/typeface-mechanics-001

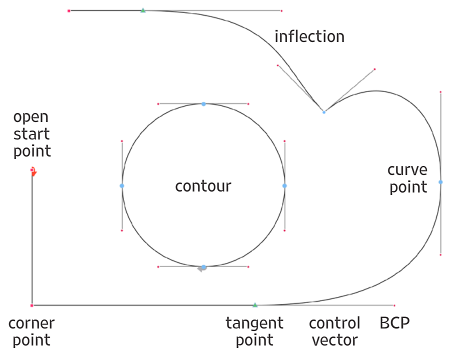
4.5 Save and backup!

Next and very importantly, save your file. Give it an easily-identifiable filename that at least reflects the font name. You should save this file often; get in the habit of saving incremental backups, so you can go back and compare to earlier versions. Some designers include a version number inside the font name, which gets incremented with each revision; others make a new folder for each day they work on the files; some do both. The important thing is to name/organize them consistently, so that you can easily refer back to them when you need to.

5 Bezier drawing for type

5.1 Mechanics. As we are making PostScript flavored OpenType fonts, we will be drawing them using cubic beziers; take a moment to make sure your drawing mode is set to **Cubic beziers (PS)** under **Preferences > Glyph View > Display.**

Cubic beziers use *nodes,* or *on-curve points,* which sit on the actual outline; and *BCPs* (bezier control points), or *off-curve points,* pairs of which determine the shaping of curve segments between on-curve points. Straight segments, or vectors, form a straight connection between two on-curve points, and do not have BCPs.

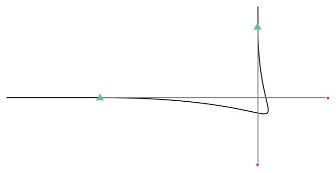


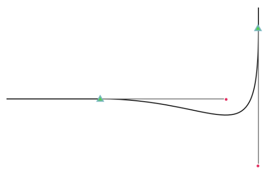
5.2 Special considerations. So far there are lots of parallels to general vector-drawing fundamentals as you know them from other illustration tools. There are also some special considerations that are specific to drawing type. If these seem restrictive, remember that we are making drawings intended to be used in great number at a fairly tiny size; and that the rendering of the shapes in a font falls to the respective platform/hardware and software that your type is displayed by, be it an operating system, a browser, or a printer, to name just a few. To make sure they'll have the best chances to survive well in all of these applications, glyph drawings need to be not only well-designed, but also well-constructed.

- All points, oncurve or offcurve, should be placed at **integer coordinates** in the 1000-unit design space.
- You should generally have **nodes at horizontal and vertical extrema:** that is, place points at the leftmost, rightmost, topmost and bottommost points of curves. Their control handles will then project in horizontal and vertical directions. This has partly technical reasons; the rendering of type can behave unexpectedly in some platforms if this rule is not regarded and the renderer has to "guess" where the horizontal and vertical extremes of the shape are. But this strategy also makes it easier to evaluate and compare dimensions and draw clean shapes. Finding the topmost point on a curve, for instance, is usually a manageable task; and then you only need to worry about the exact position of the node and the distance of its BCPs, and not their angle as well.

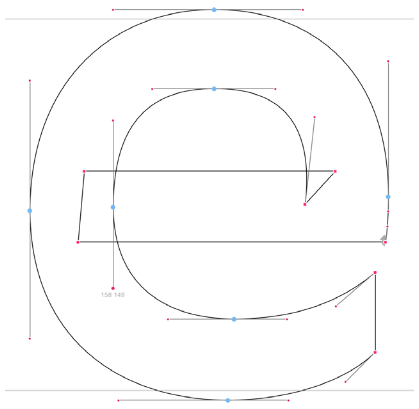
Tip: To automatically add points at extrema, use RoboFont's "Add extreme points" command from the context menu.

- In addition to the extremum points that mark the top, bottom, left, and right extremes of shapes, you will of course need additional points at corners and other places that define a design, like the terminals of a 'C'. The control vectors of these do not have to be forced to be vertical or horizontal!
- Generally, you should aim to work with a **minimal number of nodes**. It may be tempting to add more nodes, but that actually tends to make curves harder to control and shape right.
- Make sure that **each curve segment is shaped by two BCPs**, whose control vectors should be carefully balanced in length; if one is much shorter than the other that can indicate a badly shaped curve.
- Make sure that the two BCP handles that control a curve segment **do not overlap** directly, or even in their trajectories.


Not good: Overlapping control vectors


Not good: Overlapping trajectories of control vectors

- **Inflection points** happen when curve segments change direction midway through, so when the two BCPs that control them are on opposite sides of the curve (see illustration above). These constructions can sometimes be useful for drawing, but when you are finishing up your outlines you should add an explicit point at the spot where the direction changes to make sure the curve will be rendered as intended.
- **Open and closed contours.** Generated fonts will only consist of closed contours, and generally it's good to get used to working with closed ones; though opening them temporarily can be useful for editing, like to splice in a new part.
- **Path direction.** Paths in fonts have a defined direction, meaning a sequence in which their points are drawn. This sequence determines if a path gets filled with black, or "punched out" of other black shapes as a white counter. In PostScript outlines, if you picture yourself walking along the path in the path direction, the black area would always be to your left. This means that outer contours generally run counterclockwise and counters (counter-intuitively, heh) run clockwise. Usually you don't have to think about this too much, but if you do run into problems, RoboFont can fix the direction for you (select **Correct Direction (PS)** from the glyph window context menu).
- **Overlaps and point structures.** It is not a problem to make up a coherent black shape from multiple parts. In fact, it can be useful to draw logical parts of the shape separately; for instance in a lowercase *e*, drawing the crossbar as a separate overlapping part will make it much easier to keep the left-hand curve continuous (see example on the right).
  Overlaps will have to be removed for generated font files, but RoboFont can do this automatically on export. (When generating a font you'll notice a "Remove Overlap" checkbox; for test-installing, be sure to activate the "Remove Overlap" option under **Preferences > Misc > Test Install Options**.



Can be useful: Overlapping drawing structures

5.3 Feedback and warnings. RoboFont has a couple of built-in ways to alert about potential problems, such as vectors that are just off the vertical or horizontal axes or double points overlaying each other. Enable **Outline Errors** from the Display Options menu in the bottom-left corner of the Glyph Window.

Extension tip: Glyph Nanny helps detect and fix a wide range of possible technical errors in glyph drawings: robofontextensions.com/glyph-nanny/

6 Drawing in RoboFont

The drawing of your glyphs will be done in the glyph window. It comes with four default tools: The editing tool (arrow icon), the drawing tool (pen), the slicing tool (knife), and the measurement tool (ruler). You can easily switch between these tools with the number keys, the tools being numbered in the sequence in which they appear in the menu bar.

Tip: You can also install additional custom tools — an extension tip here would be the Shape Tool that quickly lets you draw rectangles and ellipses: robofontextensions.com/shapetool/

6.1 Drawing new outlines

Draw new outlines with the pen tool (**2**).

- Click and release to add a corner point; click, hold and drag BCP handles out for a curve point (hold ⇧ to constrain to horizontal/vertical).
- Double-click to automatically close the contour, or drag one of the endpoints on top of the other to close it manually.

6.2 Selecting elements

Select elements with the arrow (edit) tool (**1**).

- Click on a node or an offcurve point to select it. Hold ⇧ to keep adding to the selection.
- Click on a segment (stretch between two nodes) to select it.
- Double-click on a segment to select the entire contour.
- ⌘-click anywhere in the design space selects the closest contour.
- Marquee (drag over) multiple points to select; hold ⌥ to only select offcurves.
- **⌘A** selects all contours in current glyph.
- You can also select elements via the listings in the Inspector palette.
- Triple-click a contour for Transform mode that allows for quick scaling (grab and drag corner/center points), rotating (⌥), moving (⇧).

6.3 Editing elements

Edit selected parts of drawings with the arrow (edit) tool (**1**).

Transformations

- Apart from Transform mode (triple-click), you can perform transformations like scaling, skewing, flipping, or rotating, as well as aligning with more fine-grained control via the Inspector palette.
- You may also want to look at available Hot Key settings.

Removing nodes, opening and closing contours

- Deleting a node with the delete/backspace key will keep the contour intact/closed. Depending on a Preference setting (**preserve curve on deleting point**), RoboFont can rebalance the handles to keep the shape as close as possible to the existing one after deleting the point.
- **To open a contour:**
  - Select a node and hold ⌥ while hitting backspace to cut the contour at that point;
  - Select segments and instead of deleting, cut (**⌘X**) to remove the selected part and leave an open contour;
  - Use the preset hot key **b** to break the contour.
- **To close a contour:**
  - Manually: Drag endpoints on top of one another (snap distance can be set in **Preferences > Glyph View > Display > Snap Points to:**)
  - To automatically close open contours, select **close/join open contours** from the context menu; though depending on your point structures, the results may not be what you wanted.

Adding nodes to outlines

- Using the pen tool (**2**) will start a new contour. To add a node on an existing outline, ⌘⌥-click on a segment or drag the slice tool (**3**) across it. (Dragging the slice tool across an entire closed outline cuts the outline into two separate contours, both closed.)

Editing nodes

- Double-click a point to change its smoothness; a smooth point will have the angle of its handles constrained to the tangent.
- To add a BCP to a node that does not have two BCPs yet, hold ⌥ and drag the BCP out from the node in the desired direction.
- To remove a BCP just select and delete it. (Because curve segments should be defined by pairs of BCPs, when deleting one of them RoboFont will automatically remove the other one too, turning the segment into a straight vector.)

Moving nodes

- Use the Inspector to move selected points by precise amounts, or the arrow keys to nudge them along unit for unit; hold ⇧ for increments of 10 units, ⌘⇧ for increments of 100.
- Click and drag selected points freehand.
  - Holding ⇧ when dragging constrains the movement to horizontal or vertical. (A good habit when dragging offcurves at extrema, to make sure the handles don't go off-vertical/horizontal.)
  - Holding ⌘ generally keeps the angle of the control vector constant, even if it is not horizontal or vertical.
  - Some special applications:
  - Click on a curve segment, hold ⌘ and drag to change the length, but not the angle, of its two BCP handles, thereby altering curve shape and tension.
  - Hold ⌘ to slide one or more curve nodes while keeping the offcurve points in place (this can be really helpful for finding their exact right location).
  - Select offcurve point(s) on one side of a node, hold ⌘ and ⌥ and then drag to make a symmetrical handle.

6.4 Measuring

Use the measuring tool (**4**) to drag a measuring line across your drawing. Hold ⇧ to constrain to a horizontal or vertical direction. When measuring from an outline, holding ⇧ will constrain the angle to a right angle to the contour.

Tip: Hold ⌥ to add more than one measuring line.

The line will display measurements between contours (and, optionally, horizontal and vertical metrics; this can be set in **Preferences > Glyph View > Display: Measurements show glyph metrics** for sidebearings, **Measurements show font metrics** for vertical metrics).

Select **Measurement info** from the Display options menu in the bottom-left of the glyph window to activate an extended display that also shows the angle.

Tip: Guidelines can also display measurements (right click on a local or global guide to access this option).

6.5 Viewing

Zoom. There are different ways to zoom in and out of the glyph display:

- Hold ⌥ and scroll up/down (can be slow)
- Hot keys: by default **z** to zoom in, **x** to zoom out
- **⌘0** to fit entire glyph, **⌘+**/**⌘–** to zoom in and out
- Zoom factor in bottom-left of glyph window

Moving the view. Scrolling moves the view. When in editing mode, you can also hold space to get the little scrubbing hand.

Previews. It will help a lot to switch frequently between editing view (showing the editable structure of your outlines) and a clean black-and-white preview, which makes it easier to evaluate the shapes.
The ` key is set up as a default hot key for the preview.
Apart from this, I recommend keeping the Inspector palette open for a small preview of the glyph you are drawing.

Tip: Viewing the preview inverted can be helpful. You can set a system-wide keyboard shortcut for inverting the screen from System Preferences > Keyboard Shortcuts > Accessibility > Invert colors.

6.6 Working with guides

Consistency is important in type design, and when used judiciously, guides can be very helpful in visually referencing recurring values that shapes should align/refer to. For guides to be most useful, only use the ones you really need, and name them so you can remember which one does what. Also remember that guides can be local (meaning they just appear in the current glyph) or global (for the entire font).
To place a new guide, drag it out of the top or left side ruler.
Right-clicking on a guide reveals a few optional settings, such as for naming/labeling the guide, setting a custom angle, displaying measurements along the guide, or making it slice the glyph.

6.7 Working with layers

While the foreground layer is the only one that gets exported into generated fonts and displayed when the font is used, UFOs can contain any number of additional layers. Other layers may for instance be useful for keeping reference images, scans, or old/alternate versions of drawings that can be displayed in the background.
To switch between layers, use the dropdown menu on top-right of the glyph window, the Layers palette in the Inspector, or (this is the fastest way) assign custom hot keys. This will for instance make it easy to quickly switch back and forth between layers to evaluate alternate drawings. You can also set a hot key for copying the contents of the current layer to another layer.
To create new layers, set display options, or delete existing ones, use the Layers palette in the Inspector. (Layers are defined per font, so if you delete a layer from one glyph it will also be gone from all others!)

NB: While layers of a glyph can contain different artwork, they all share the same width.

7 Spacing

Spacing, or the setting of horizontal metrics for each glyph, i.e. deciding where the sides of its body end, should not be regarded as a separate step in the process: It is deeply intertwined with drawing and needs to be addressed and refined concurrently. It may sound esoteric, but the white parts of the glyph body are as much part of the design as the black parts; there's really no way you can judge the shapes you draw without taking their spacing and rhythm into consideration. In fact, a typeface that's badly drawn but evenly spaced has a better chance of being legible in running text than the other way round.

There are two general concerns with spacing: Determining the overall "fit" of your typeface, so the *amount* of space between letters, how *tight* or *loose* the fitting is; and making the rhythm of shapes as *even* and regular as possible.

Spacing is a lot about balancing the space enclosed by the black glyph bodies with the space surrounding them. There is no one simple recipe for this relationship; it depends on many factors, including the target size you are designing for, or whether your font has serifs. If you make a serif typeface for text, you can usually start by basing the space between adjacent 'n's on the space enclosed by the 'n'; so start the 'n' off with sidebearings of about half the width of the counter of the 'n'. If your font is for large sizes or is a sansserif, all the sidebearings will have to be tighter.

One important thing to realize is that sidebearings give a linear distance measurement; but spacing is all about balancing *areas* of space. So the sidebearing value will not directly correspond between different shapes: Sidebearings of round shapes will measure less than those of flat shapes, and those of pointy shapes even less. The sidebearings of symmetrical shapes like the 'o' should be symmetrical. In a conventionally-shaped 'n', however, the distance from the right stem to the sidebearing will also be a bit less than that of the left one, because there is an additional bit of space tucked in above the arch, where the left side has a stem and perhaps a serif.

7.1 Basic method. The problem in getting spacing to be even by setting one sidebearing at a time is that it is hard to precisely identify where problems originate. In running text, the right sidebearing of one letter combines with the left sidebearing of the next one, so we are always looking at two variables at the same time. The solution is to go about spacing in a systematic way that isolates these variables and allows us to solve for one of them at a time. This is where *spacing standards* come in: Reference glyphs to work out first, which other glyphs can then refer back to.

- The first thing to do is to look at the spacing and rhythm of (a) flat-sided glyphs, and (b) round-sided glyphs. In the caps, these are typically HHHHH and OOOOO; in the lowercase, nnnnn and ooooo. Look at these in various sizes, but mostly at the intended target size of the design. The fit of these will determine the overall rhythm of your design. You should look at the lowercase and the caps separately at first, and while their numbers won't match up, they should behave similarly, on a different scale.

Definitions

Spacing: Setting the sidebearings of each glyph to define the spaces between the drawings.

Tracking: Evenly increasing (tracking out) or decreasing (tracking in) the amount of space between the glyphs. This is typically done in layout applications, and corresponds to the "letter-spacing" CSS property.

Kerning: Individual adjustment of the space between specific pairs of glyphs whose shapes are particularly uncooperative when treated as the rectangles they sit on (such as 'AV' or 'To'). This somewhat painstaking process of pair-by-pair refinement should only be considered later in the process, after the overall spacing has been finetuned, and will be introduced in more detail later on.

Sidebearings: Horizontal distance between the leftmost/rightmost part of a design and the border of the glyph body. Sidebearings can also be negative.

Advance width (also called **set width**): The total width of a glyph including the width of the bounding box of the drawing plus left and right sidebearings.

- Once your HHHHH and OOOOO, and your nnnnn and ooooo, seem well-balanced in themselves, it's time to mix them to make sure they relate well to one another. The best test for this is to immediately add a third letter that directly combines both of these basic shape elements; that would be a D for the caps, and a p for the lowercase:
  HHDHOHODOO
  nnpnonopoo

Tip: You will find all of these test strings in the supplied InDesign proofing templates.

- The special feature of these test strings is that they contain mostly flat sides at the start and mostly round sides at the end; so if there is a mismatch between how the two behave, these sequences will amplify that, making it easier to spot.
- In a third step, see how well your solutions for capitals and lower-case letters combine by testing
  HHnnHH OOooOO
- Once you have nailed down these "standards," you can apply the principles found here to other glyphs. The C, for instance, is likely to share the O's left side profile and therefore, sidebearing, but how about its right side? The chart in the sidebar, from Walter Tracy's *Letters of Credit,* gives a rough guideline for how sidebearings across the alphabet can be approximated. Just keep in mind this was conceived for serif text typefaces, and for sansserifs (or designs that diverge in other ways from this traditional model) the recipe will differ somewhat.

Spacing approximations after Walter Tracy's "Letters of Credit"

Lowercase

same as left side of n	**b**	same as o
same as o	**c**	slightly less than o
same as o	**d**	same as left side of n
same as o	**e**	slightly less than o
slightly more than left side of n	**h**	same as right side of n
slightly more than left side of n	**i**	same as left side of n
same as left side of n	**j**	same as left side of n
slightly more than left side of n	**k**	minimum
slightly more than left side of n	**l**	same as left side of n
same as left side of n	**m**	same as right side of n
slightly more than left side of n	**p**	same as o
same as o	**q**	same as left side of n
same as left side of n	**r**	minimum
same as right side of n	**u**	same as right side of n
minimum	**v**	minimum
minimum	**w**	minimum
minimum	**y**	minimum

(The double-storey **a** and the **f g s t z** must be spaced visually, between standards.)

Uppercase

minimum	**A**	minimum
same as H	**B**	about half of H
same as O	**C**	about half of H
same as H	**D**	same as O
same as H	**E**	about half of H
same as H	**F**	about half of H
same as O	**G**	slightly less than H
same as H	**I**	same as H
minimum	**J**	same as H
same as H	**K**	minimum
same as H	**L**	minimum
slightly less than H	**M**	same as H
slightly less than H	**N**	slightly less than H
same as H	**P**	same as O
same as O	**Q**	same as O
same as H	**R**	minimum
minimum	**T**	minimum
same as H	**U**	slightly less than H
minimum	**V**	minimum
minimum	**W**	minimum
minimum	**X**	minimum
minimum	**Y**	minimum
about half of H	**Z**	about half of H

(the **S** must be spaced visually, between standards.)

- All that said, many glyphs will not behave entirely predictably; but now you have a stable context to compare them to, so you can/should evaluate the spacing of all characters between series of either 'H's or 'O's (or for lowercase letters between 'n's and 'o's), and check that the white areas around them look evenly balanced. Not all letters are going to look equally good between straight forms and round forms; the goal is to find a compromise that works for both, rather than perfecting one at the expense of the other.

7.2 Spacing in RoboFont. While it is possible to set sidebearings also in the glyph window (drag) or the Inspector (enter values), really where you should work out the spacing of your font is in the Space Center.

- For sending glyphs to the Space Center, select one or more glyphs in the Font Collection and open the Space Center (with the button in the top bar of the Font Collection window, selecting **Window > Space Center** or pressing **⌘⌥S**). See how useful the context fields are for quickly viewing a set of letters in the Space Center between context glyphs or standards?
- For editing the sidebearings, there are two main methods:
  - Select a glyph in the preview, and use the arrow keys to change the sidebearings. ←→ change the right sidebearing of the glyph; hold ⌥ to change the left sidebearing instead. Holding ⇧ or ⌘⇧ for intervals of 10 or 100 units, respectively, works here, too.
  - For working directly with the numbers, make sure the Space Center displays the Space Matrix (selectable from the Display Options flyout menu in the top-right of the window). The space matrix lists values for left, right sidebearing and the total advance width of the glyph. In the space matrix, use ⇥ ⇧⇥ and ⌥↑ ⌥↓ to navigate, ↑ ↓ to edit values. You can also just type in values, or even references to other glyphs: To set the left sidebearing of the D to

the same value as the H, instead of looking up the value of H and copying that, you can also just type **=H** into this box, a reference that will instantly be converted to the correct number. (NB: This only inserts the correct value once and should not be misunderstood as somehow linking the spacing of these two glyphs.)

- Sidebearings are typically measured from the extreme point of the drawing. To find the sidebearing at any given point on an outline, activate the **Beam** from the Display Options menu. It acts much like a measurement line in the Space Center that can be moved vertically; the values in the space matrix will update to reflect the measurements at that particular vertical position.

Tip: I've found that entering values into the space matrix with the beam on can be a bit buggy, so I prefer to just use this for diagnosis and then switch it off again to enact changes.

Tracking tip: If you're not sure whether your font is overall too tight or too loose, a quick way to prototype variants is to typeset your spacing strings or sample text in InDesign, and use the controls there to track the text in or out. *The units used there for tracking translate directly to the font units you use in RoboFont.* So if your text looks better at, say, 10 tracking in InDesign, try adding 10 units to the width of every glyph (5 on the left side, 5 on the right) to achieve the same effect directly in the font.

8 Evaluating your work

A big part of type design is testing and evaluating your work, staring at it in different ways, learning to see what needs fixing, and then iteratively applying these insights to refine the design. While we draw letters in isolation, they need to be evaluated in context with other letters (and at a more realistic scale) to make sure they all play nice together.

8.1 Evaluating your work in RoboFont: While the Space Center is primarily for, well, spacing, it's also a great place for previewing and testing your font. View your font with different kinds of text, in different sizes. Try out the display options, view it upside down, invert the screen.

Extension Tip: word-o-mat is a RoboFont extension that will find actual sample words in a number of languages and display them in the Space Center. This makes it easy to preview your font in real words even while your character set is still limited. robofontextensions.com/word-o-mat/

8.2 Evaluating your work in layout applications:

- To test your work outside of RoboFont, the handy **Test Install** command (in the File menu) installs a temporary copy of the font on your system while RoboFont is open. After test installing, the font will be available to use in all applications, just like any other installed font. If you edit the font, run Test Install again to update it across the system. When the font is closed or RoboFont is quit, the temporary font is cleared off the system.
- Use the supplied InDesign templates for testing and proofing. These have been prepared in ways that should make them most useful and informative in the various stages of your design process. You should adapt them to a size that is appropriate for your design.
- When printing showings and proofs, do pay attention to printer settings to get an image that is as close as possible to the shapes you actually drew. For recommended settings with the Graphic Design printers, refer to the separate printer guideline. Generally, for proofing type you should always use monochrome printers and be sure to set the resolution to 1200 dpi.

Tip: If test installing does not seem to work, open the Output Window from the Python menu, and see if there are any error messages that might indicate why. You cannot, for instance, test install a font that has no name set yet. And you should keep RoboFont active/in the foreground while Test Installing is in progress.

9 Generating font files

Generating actual font files is almost as simple as test installing. From the File menu select **Generate Font.** You are presented with a save dialog that includes a few options. You will want to select **otf** (not **ttf**) as the format, and you'll probably want to activate the **Remove Overlaps** option if your design contains any overlapping parts (these can cause problems in PDFs). Don't worry, RoboFont only flattens the overlaps in the exported font, not in your source file. For now it is best to deactivate autohinting.
To install the generated .otf font files, simply copy them to either your system-wide or your user-specific Fonts folder.

10 Some more workflow tips

- **Save and backup.** Really. Harddisks break. They especially break when you least expect them to. Backups are your friends.
- **Mark colors.** It can be very useful to use color highlighting of glyphs in the Font Collection window for instance to track their status, highlight what glyphs need attention or have been fixed, etc. Simply select a color from the Mark menu in the Font Collection (or the Inspector window). You can define your own preset colors and labels in **Preferences > Misc**.
- **Notes.** You can attach notes to glyphs in the Inspector. This can for instance be useful as a "note to self", like if you spot something that needs to be checked or fixed later. You can also attach a note to the entire font in the Font Info (in the bottom of the **General** tab).
- **Scripting.** RoboFont is fully scriptable in Python (in fact the program is itself written in Python), and if you are serious about type design, picking up a bit of scripting knowledge is a very useful superpower to have; you can use it to automate processes, ensure consistency, or quite generally tailor workflows to your own needs. There is sadly no room here for a complete introduction to Python, but we may be able to put on an extra workshop if you are interested.

11 Further resources

- **RoboFont Documentation:** not complete/exhaustive, but partially very useful (there are even some screencasts):
  http://doc.robofont.com/documentation/
- **RoboFont Forums:**
  http://doc.robofont.com/forums/